Руденко С.Ю.

УПРАВЛЕНИЕ БАЗАМИ ДАННЫХ

Paradox 9

Часть 1.
Paradox

Часть 2.
Paradox Application Language
(с использованием примеров из реальных программ)

для школьников, студентов, начинающих и опытных предпринимателей,
желающих управлять процессами своего хозяйства.

Российский Новый Университет
Москва
2011

Получить как можно больше и заплатить как можно меньше - естественное желание каждого человека. Paradox (именно поэтому он получил свое название) единственный из языков программирования, который отвечает этому требованию.

Когда – то была фраза, что «даже кухарка может управлять государством». Paradox – это такая Система Управления Базами Данных (СУБД), которая позволяет практически любому человеку стать программистом. Главное, знать что хочется получить.

Т.е. книга рассчитана для:

- Школьников, желающих систематизировать свои проблемы.
- Молодым матерям для контроля развития их детей,
- Владельцам небольших магазинчиков для учета прихода, расхода и потребности в товаре,
- Для скучающих вечерами, для поваров, экономистов, медицинских работников и вообще всех, кого интересует быстрое получение результата из какой – либо базы данных или создание самих этих баз.

Книга состоит из двух частей.Первая часть описывает возможности СУБД Paradox for Windows (версия Word Perfect Office 12, .Paradox 9)

Вторая часть – язык приложений для СУБД Paradox – PAL, который упрощает весь процесс работы с готовой задачей до нажатия двух клавиш. Этот язык встроен в Paradox и никаких дополнительных программных средств не требуется.

Описание этих частей в книгах отличается. Первая книга содержит примеры из эталонных баз. Эти эталонные базы устанавливаются сразу с Paradox и находятся в каталоге sample, т.е. установив Paradox вы сможете проверить примеры из каждой главы этой книги. Ответы должны полностью соответствовать рисункам. В некоторых главах будет предложено изменить содержание таблиц, но, если вы их не скопировали, всегда можно переустановить Paradox. В этом случае все эталонные базы и все основные компоненты будут восстановлены. Содержание BDE , которое можно изменить и из «Панели Управления Windows», останется пользовательским.

Для тех, кто пользовался уникальнейшим Paradox for DOS не стоит огорчатся. Все эти старые программы (проверено на 2-х ядерных процессорах, где DOS игрушки не идут) работают нормально. Это будет показано во второй части. Это удобно, когда надо много печатать с минимальной стоимостью (например, на матричных принтерах квитанции об оплате).

Вторая, которая описывает PAL (Paradox Application Language – устанавливается сразу с Paradox), построена на описании отдельных частей конкретных рабочих задач. Она будет интересна и профессиональным программистам для сравнения трудоемкости реализации алгоритмов на различных языках.

Paradox работает на всех системах Windows (95, 98, 2000, XP, Vista, 7) и на тех компьютерах, для которых писались эти системы. Например, на двухядерном компьютере с Windows XP отрабатывалась задача для PC-386 с печатью в DOS на матричном принтере (для печати подключались модули Paradox for DOS)

ISBN 978-5-86746-111-4

Введение

WordPerfect Office 12 - лучшая альтернатива пакету Microsoft Office, при этом гарантирующая полную совместимость с ним при меньшей стоимости (Разработчик: Corel Corporation). При использовании пакета WordPerfect Office 12, совместимого практически со всеми популярными офисными инструментами, пользователь может упорядочить свою совместную работу с другими сотрудниками, в частности, с теми, кто работает с пакетом Microsoft Office.

Текстовый редактор WordPerfect 12: Получивший широкое признание пакет для работы с текстами WordPerfect 12 предлагает полный контроль над документами, при этом пользователю предлагается возможность быстрого и удобного импорта / экспорта файлов Microsoft Word. С помощью пакета WordPerfect 12 пользователь может создавать профессионально оформленные письма, статьи, доклада, коммерческого предложения, книги, юридического документа, брошюры и многого другого. Кроме того, при подготовке документов к распространению пакет WordPerfect 12 предлагает широкие возможности для работы с технологиями HTML, CSS (Cascading Style Sheets), XML и PDF.

Quattro Pro 12: Эта мощная электронная таблица позволяет систематизировать и анализировать данные, готовить отчеты, публиковать важные данные или финансовую информацию. Расширенная совместимость с файлами пакета Microsoft Excel гарантирует возможность обмена данными между электронными таблицами Microsoft Excel и Quattro Pro 12.

Presentations 12: Пакет Presentations 12 содержит все необходимые инструменты для создания слайд-шоу, описаний проектов, интерактивных отчётов, презентаций, мультимедийных роликов, листовок, вывесок, баннеров, PDF- документов, файлов Macromedia Flash для web- страниц и многого другого. Пакет Presentations 12 предлагает технологию автономного просмотра презентаций Show on the Go, которая позволяет просматривать презентации на любом компьютере, даже если там не установлен пакет Presentations.

Paradox: Поставляемая в состав редакции Professional и Education пакета WordPerfect Office 12, СУБД Paradox представляет собой удобную реляционную базу данных, которая предлагает множество способов хранения и извлечения данных. Также из сохранёных в СУБД Paradox данных можно создавать качественные формы для заполнения, диаграммы и отчеты. Также можно публиковать данные в файлах различного формата, например, в форматах Quattro Pro (QPW), Microsoft Excel (XLS), Lotus 1-2-3 и dBASE (DBF), решать задачи по учету коммунальных услуг, составлению архивов и работе с ними с учетом графики (архив музея, кдровый учет, учет в поликлиниках) и т.д

К вспомогатеьльным программам и утилитам относятся:

PerfectExpert: Утилита PerfectExpert позволяет быстро создавать сложные документы на базе созданных профессиональными дизайнерами шаблонов, которые можно скорректировать в соответствии со своими потребностями. С помощью мастеров пользователь может создать даже такой сложный проект, как бизнес-план, а также решить не такие сложные задачи, такие, как вставка изображения из набора графики. В утилите PerfectExpert прилагается несколько галерей профессионально сконструированных шаблонов, а также средства редактирования компоновки и содержимого этих шаблонов.

Словарь **Pocket Oxford Dictionary:** Английский толковый словарь Pocket Oxford Dictionary предлагает определения и произношения для 30 000 слов.

Адресная книга: Этот мощный инструмент помогает фиксировать и публиковать сведения о контактах. В адресной книге можно добавлять собственные поля и проводить поиск нужных записей. В пакете WordPerfect Office 12 поддерживается несколько типов адресных книг, включая книги Outlook, MAPI и GroupWise.

Шрифты, графические элементы и фотографии: В комплект поставки пакета WordPerfect Office 12 входит обширная коллекция шрифтов, изображений и фотографий для оформления документов. Модуль вставки графических элементов Scrapbook обеспечивает быстрый и удобный доступ к папкам с готовыми графическими элементами, разбитыми по категориям. Пользователь может просмотреть имеющиеся графические элементы, а затем перетаскивать или переносить их через буфер обмена в диаграммы, презентации, электронные таблицы, аналитические отчёты и другие документы. Также модуль Scrapbook позволяет просматривать аудио - и видео материалы, а также создавать собственные категории для систематизации ресурсов.

Как видно из стандартного перечня разделов WordPerfect возможности у пакета практически безграничные. Если большинство из них знакомо пользователям Microsoft Office, то Paradox является исключением. Объяснить это трудно, т.к. Paradox появился еще во времена DOS и является очень мощным, простым и удобным средством управления базами данных (написание новых программ или выборка одним запросом ответа из задачи, например, 1СБухгалтерия и написанной на Foxpro).

Данная книга знакомит с работой в Paradox-9 (WordPerfect Office 2000). Эта версия является промежуточной между Paradox for Windows и WordPerfect Office 12, т.е. все, что описано ниже, работает с любой версией. Программы можно писать на любой версии и везде они работоспособны (проверено с версиями 7 и 9 на большом количестве программ).

Глава 1 Основные понятия

Объекты Paradox

Paradox предоставляет большой выбор способов хранения, отображения и представления данных. Компоненты, которые используются для хранения и представления данных, называются объектами (objects). В системе Paradox существуют следующие объекты: таблица (table), форма (form}, отчет (report), запрос (query), программа (script), библиотека программ (library).

Paradox использует объекты для хранения и представления информации. С точки зрения простого пользователя к объектам относятся:

1	Файлы на диске
2	Таблицы, формы, отчёты, запросы, программы и библиотеки

Объекты интерфейса - это объекты, которые размещаются в формах и отчётах с помощью кнопок, расположенных на панели управления в окнах конструктора. К объектам интерфейса относятся:

1	Текстовые объекты
2	Рамки, линии и эллипсы
3	Поля и таблицы
4	Перекрестные таблицы и диаграммы
5	Объекты -наборы записей
6	Кнопки
7	Графические объекты
8	OLE-объекты
9	Страницы документов

Таблицы

Paradox размещает данные в таблицах. Таблицы состоят из рядов (row) и колонок (column). Каждый ряд содержит всю доступную информацию о конкретном предмете (item) и называется запись (record), а каждая колонка - одну категорию данных, называемую полем (field).

Типы полей в Paradox

Paradox разделяет поля на несколько типов. Каждый тип поля определяет вид данных, которые оно содержит.

При выборе поля предлагается список, который приведён ниже на рис.1.1.

Это окно открывается, когда создается или редактируется тип поля таблицы. Для его вызова надо, чтобы курсор находился в поле «тип» и потом нажать правую кнопку мыши.

Когда таблица создана, количество знаков, которые выводятся на экран можно для некоторых полей можно регулировать. Для этого при просмотре таблицы надо поставить курсор на поле (любое в колонке) и нажать правую кнопку мыши и выбрать формат.

A	Alpha
N	Number
$	$(Money)
S	Shot
I	Long Integer
#	#(BCD)
D	Date
T	Time
@	@(Timestamp)
M	Memo
F	Formatted Memo
G	Graphic
O	OLE
L	Logical
±	Autoincrement
B	Binary
Y	Bytes

Рис.1.1 поля таблицы Paradox

Например, для цифрового поля (number format) можно добавить тип, который показывал четыре знака после запятой и дать ему любое название (например, тариф).

В дальнейшем это поле будет показано в этой таблице под выбранным нами именем.

Новый тип должен отвечать требованиям ранее используемого и соответствовать таблице 1.1

Таблице 1.1 Типы полей Paradox

Тип поля	размер	Описание
Алфавитно-цифровое (Alpha)	от 1 до 255	Хранит буквы, числа, спецсимволы и любые другие печатные символы.
Числовое (Number)	не нужен	Хранит числа в диапазоне от -10307 до 10308 с 15 значащими цифрами. Числовое поле лучше всего подходит тогда, когда вы хотите выполнять вычисления над значениями полей.Используйте алфавитно-цифровое поле, а не числовое, если вы хотите вводить скобки или тире (как в случае телефонных номеров или почтовых индексов),
Денежное (Currency)	не нужен	Хранит числа в диапазоне от -10307 10308 с 15 значащими цифрами. Денежные («Валютные») поля -абсолютно такие же, как и числовые, но при отображении форматируются таким образом, чтобы выделять десятичные позиции сотен, тысяч, миллионов и знак валюты. Независимо от количества показанных десятичных позиций, Paradox распознает до шести десятичных позиций при выполнении внутренних вычислений над денежными полями
Дата (Date)	не нужен	Содержит любую допустимую дату от 1 Января 100 до 31 Декабря 9999. Paradox правильно обрабатывает високосные годы и столетия и проверяет все даты на допустимость.
Короткое целое (Short Number)	не нужен	Содержит целые числа в диапазоне от -32767 до 32768. Короткое целое поле не позволяет использовать те же опции форматирования, что и числовое поле, и его следует использовать только опытным пользователям Paradox.
Длинное целое (Long Number)	не нужен	Содержит целые числа в диапазоне от -2147483647 до 2147483647. В остальном, оно аналогично Short Number
Memo (Memo)	от 1 до 240 в .DB файле; неограничен в .MB файле	Содержит текст переменной длины и обычно слишком длинный, чтобы храниться в алфавитно-цифровом поле. Поля memo могут хранить буквы, числа, спецсимволы (та-кие как %, &, #, и =) или любые печатные символы, а также символы перехода на новую строку (LF), возврата каретки (CR), табуляции и другие символы управления печатью.

		Поля memo могут быть практически любой длины. Значение задаваемого размера относится к части memo - поля, которая хранится в таблице, Она может быть от 1 до 240 символов. Оставшуюся часть memo - поля Paradox хранит вне таблицы в файле с расширением .MB. Paradox считывает данные из файла .MB, когда вы листаете записи в таблице. Количество данных, которое может содержаться в этом поле ограничено только доступным дисковым пространством на вашем компьютере
Форматированное мемо Formatted memo)*	от 1 до 240 в .DB файле; неограничен в .MB файле	Форматированное memo - поле не отличается от memo - поле кроме того, что вы можете хранить в нем отформатированный текст. Paradox распознает и хранит наряду с данными атрибуты текста (оформление различными шрифтами, стили, цвета и размер букв), а также и формат параграфов (позиции табуляций, отступы и выравнивание).
Двоичный (Binary)*	от 1 до 240 в .DB файле; неограничен в .MB файле	Содержит данные, которые Paradox не может интерпретировать. Принято использовать двоичные поля для хранение звуковых данных. Paradox не может отобразить или интерпретировать двоичные данные, но при помощи ObjectPAL можно к ним получать доступ.
Графическое (Graphic)*	от 1 до 240 в .DB файле; неограничен в .MB файле	Содержит данные, которые Paradox не может интерпретировать. Принято использовать двоичные поля для хранение звуковых данных. Paradox не может отобразить или интерпретировать двоичные данные, но при помощи ObjectPAL можно к ним получать доступ.
OLE*	от 1 до 240 в .DB файле; неограничен в .MB файле	Содержит объекты, помещённые в вашу таблицу из других приложений Windows, которые поддерживают OLE (Object Linking and Embedding -Связывание и Встраивание Объектов) как сервер. Преимущество использования поля типа OLE в том, что при помощи OLE вы получаете доступ и возможность изменения OLE - объекта в другом приложении Windows вне системы Paradox.

* Задание размера поля необязательно.

BLOB-поля

Memo, форматированные memo, графические, OLE и двоичные поля могут содержать так называемые Двоичные Файлы Больших Объектов (BLOB - Binary Large Object). К этой группе полей применимы обделённые правила, поэтому они иногда собирательно обозначаются как BLOB - поля. Мемо - поле dBASE - таблицы также является BLOB - полем.

Типы полей dBASE

Paradox позволяет вам создавать и использовать таблицы формата dBASE так же просто, как и таблицы формата Paradox. Если вы используете dBASE - таблицы, следует учесть, что типы их полей отличаются от типов полей Paradox - таблиц.

Таблица 1.2 Типы полей dBASE

Тип поля	Размер	Описание
Символьное (Character)	от 1 до 254	Хранит любой печатный символ (включая пробелы).
Действительное число (Float number)	от 1 до 20	Хранит численные данные в двоичном формате с плавающей точкой.
Число (Number)	от 1 до 20	Содержит числовые данные в формате BCD (Binary Coded Decimals). Используйте числовые поля, если вам нужно выполнять точные вычисления. Вычисления над - числовым полем занимают больше времени, но дают большую точность, чем в случае типа поля действительные число(с плавающей точкой).
Дата (Date)	8 (автоматически)	Содержит даты. Формат по умолчанию для ввода и отображения дат устанавливается в Windows Control Panel, но вы можете форматировать поля даты dBASE точно также, как и поля даты Paradox, «инспектируя» поле даты в таблице или в форме. Не нужно задавать размер для поля даты- он всегда по умолчанию 8.
Логическое (Logical)	не нужен	Содержит величины имеющие значения Истинно или Ложно (True or False). Можно задавать формат значений, по желанию, «инспектируя» логическое поле и выбирая в списке свойств Logical Format (Формат Логического Поля)
Memo (Memo)	не нужен	Содержит блоки текста, слишком большие для того, чтобы хранить их в символьном поле. Содержимое memo - полей хранится в отдельном файле с именем таблицы и расширением .D8T, Размер поля для memo - полей задавать не нужно.

Временные таблицы

При выполнении некоторых операций Paradox создаёт временные таблицы, существующие до тех пор, пока вы не смените личный (private) каталог или не завершите сеанс работы с Paradox. Paradox хранит все временные таблицы в вашем личном каталоге. Вы можете редактировать временные таблицы и делать запросы к ним так же, как и к другим таблицам. Если вы хотите сохранить одну из этих таблиц, используйте для этого команду File / Utilities / Rename.

Не следует использовать имена временных таблиц в качестве имен объектов Paradox. Если вы назовете объект именем временной таблицы, то Paradox удалит ваш объект, как только вы смените личный каталог или завершите сеанс работы.

Таблица 1.3 Временные таблицы

Имя	Содержимое	Создается во время операции
Answer	Результат запроса	Запрос
Changed	Копии первоначального содержания изменённых записей	Запрос CHANGETO или операция добавления Add
Crosstab	Кросстаблица (crosstab)	Создание кростаблицы в форме
Deleted	Удалённые записи	Запрос DELETE
Errchng	Записи, которые не могут быть изменены	Запрос CHANGETO
Errdel	Записи, которые не могут быть удалены	Запрос DELETE
Errins	Записи, которые не могут быть добавлены	Запрос INSERT
Export	Специальная таблица для экспортирования текста с полями фиксированной длины	Операция Export
Import	Специальная таблица для импортирования текста с полями фиксированной длинны	Операция Import
Inserted	Добавленные записи	Запрос INSERT
Keyvoil	Записи с дублированным значением ключевого поля	Изменение структуры таблицы (Restructure) или операция добавления записей Add
Locks	Все активные в данном сеансе работы с Paradox заблокированные записи	File / Multiuser / Display Locks
Pal$src	Список исходных кодов, объектов и методов в форме	Language / Browse Sources
Problems	Непреобразованные записи	File / Utilities / Import или Restructure
Struct	Определения полей таблицы	Create или Restructure

Формы

В большинстве случаев более удобно работать с данными в отдельных записях, а не со всей таблицей целиком. Формы позволяют вам видеть столько данных из таблицы, сколько вы пожелаете, и в том формате, который вы предпочитаете. При работе с формой вы видите те же данные, что и в таблице, но представленные в другом виде. Если вы редактируете данные в форме, то Paradox обновляет соответствующую информацию в таблице.

Вы можете использовать средства разработки Paradox (Design tools) для создания форм с необходимой вам структурой. Paradox позволяет отображать в форме несколько записей одной таблицы или записи из нескольких разных таблиц одновременно.

Отчеты

При работе с базами данных часто бывает нужно распечатать необходимую информацию. Paradox предоставляет мощные средства генерации отчетов. Вы можете сортировать и группировать записи, производить необходимые вычисления над полями, а также упорядочивать и представлять данные практически в любом формате.

При создании отчётов, также как и форм, доступны средства разработки Paradox (Design tools). Используя их, вы можете сконструировать вид отчёта, который вам необходим. А так как Paradox позволяет связывать данные из разных таблиц, то вы легко можете создавать сложные отчёты, использующие несколько таблиц базы данных.

Запросы

Запрос в Paradox - это некий вопрос об информации в вашей базе данных. При помощи запросов вы можете:

- Вести поиск или выбор данных в таблице
- Комбинировать данные из нескольких таблиц
- Производить вычисления над данными
- Вставлять данные
- Удалять данные
- Изменять данные
- Определять группы и наборы данных, над которыми будут производиться вычисления или сравнения

Paradox предоставляет вам простой, но гибкий и мощный способ создания запросов. В окне Query вы можете выбрать таблицы, по которым вы хотите сделать запрос. Затем вы вводите образец (example) данных, удовлетворяющих вашим условиям, а Paradox выбирает из таблиц те данные, которые соответствуют этому образцу. Это называется Запрос По Образцу (QBE - Query By Example). Практически система запросов позволяет моментально получать любые выборки из любого количества таблиц, в которых есть одинаковые поля. Paradox представляет возможность составления запроса в графическом виде, что делает эту возможность уникально простой.

Программы

Программы - это последовательность команд на языке ObjectPAL (языке разработки приложений Paradox), которые позволяют автоматизировать выполнение определённой последовательности действий над базой данных. Коды ObjectPAL обычно «присоединяются» (attach) к объектам формы, но вы можете написать и отдельные программы, которые будут выполнять независимо от какой-либо формы. Например, вы можете написать программу для открытия определённой таблицы и выполнения вычислений над одним или более полями этой таблицы. Программы такого типа запускаются непосредственно из основного окна Paradox (Desktop), а не в результате того, что произошло какое-либо событие, которое запустило на выполнение программу (метод), присоединённую к некоторому объекту формы.

При инсталяции Paradox его расширения становятся понятны Windows, что позволяет вывести иконку начальной программы, которая задает исходные данные, на рабочий стол и в дальнейшем использовать ее для запуска.

Библиотеки программ

Библиотека - это объект, который вы можете использовать для хранения команд ObjectPAL. Это дает вам возможность создавать процедуры, доступные различным формам, программам и другим библиотекам.

Конструкционные объекты

Кроме рассмотренных выше объектов Paradox предоставляет набор так называемых конструкционных объектов (design object), при помощи которых разрабатывается структура форм и отчетов. Вы можете создавать эти объекты используя специальные средства, находящиеся на SpeedBar (линейке, расположенной в верхней части экрана и содержащая набор кнопок - иконок для вызова различных операций), и размещать их на документе, который вы разрабатываете, будь то форма или отчет.

В случае формы - страница, на которой вы размещаете объекты, сама является объектом. Вы можете изменить ее свойства, например, цвет, или присоединить к ней методы (метод (method) - это последовательность команд ObjectPAL, выполняющая определённые действия). Методы, присоединённые к странице, могут начать выполняться при открытии или закрытии документа, при щелчке или двойном щелчке клавишей мыши или при выполнении другого события.

Вы можете создавать конструкционные объекты при помощи инструментария (design tools), который есть на SpeedBar окна разработки каждого документа.

Текстовые объекты

Текстовый объект (text object) - это объект, содержащий текст. Вы можете создать рамку, в которую можно будет поместить текст. Текст может быть любой длины и любого формата,

Чаще всего текстовые объекты используются для размещения заголовков на формах и отчетах или для задания имен полей и таблиц.

Прямоугольники, линии и эллипсы

Прямоугольники, линии и эллипсы (boxes, lines, ellipses) - это объекты, которые вы можете размещать в форме или отчете для придания документу более привлекательного вида. Вы можете рисовать прямоугольники или эллипсы вокруг полей или таблиц, использовать дополнительные линии для того, чтобы указать на какую-либо важную особенность в документе.

Поля

Вы можете размещать поля из ваших таблиц в форме или отчете. Используйте инструмент Field на SpeedBar для того, чтобы начертить рамку (границу поля) (frame), а затем задать ему те свойства, которые вам нужны. Вы можете определить объект типа поле из уже существующей таблицы или создать вычисляемое или итоговое поле для выполнения действий над вашими данными.

Таблицы

Таблица - один из основных объектов Paradox. В разрабатываемых документах (формах или отчетах) вы можете использовать инструмент Table для того, чтобы создать столбцы и ряды таблицы, а затем задать и саму таблицу с данными. В таблицах на экранных формах и отчетах вы можете размещать поля и их заголовки гораздо свободнее и разнообразнее, чем в режиме просмотра и редактирование таблиц в окне Table.

Кросстаблицы

Кросстаблица (crosstab) преобразует данные из структуры таблицы базы данных в структуру, подобную электронной таблице. Она подводит итоги по одному полю, группируя записи в этом поле и основываясь на значениях одного или нескольких полей (например, можно узнать объем продаж различных изделий по месяцам). Кросстаблицы дают возможность анализировать данные по одному или нескольким факторам.

Предположим, вы хотите определить, в какие месяцы клиенты предпочитают расплачиваться наличными, а в какие -по перечислению. Вы можете создать кросстаблицу, которая покажет суммы отпуска товаров по накладным в каждом месяце , сгруппированные по способам оплаты.

Графики

Иногда более удобно анализировать и представлять ваши данные в виде графиков (graph) и диаграмм. Paradox дает возможность легко создавать графики по вашим данным. Вы можете варьировать тип графика, его строение и свойства. Paradox автоматически обновляет график, если изменяются данные в таблице (даже если изменения произошли при работе другого сетевого пользователя этой таблицы).

Многозаписные объекты

Многозаписные объекты (multy-record object) представляет собой повторяющийся поля нескольких записей. Вы задаёте расположение полей одной записи и указываете, сколько раз по вертикали и горизонтали повторяется этот образец. Многозаписный объект позволяет отображать одновременнно несколько записей (как и в таблицах) и так располагать поля, чтобы вам было удобно (как в формах).

Кнопки

Кнопки (button) - это объекты Paradox, которые вы можете размещать в формах и присоединять к ним методы ObjectPAL Работая с формой вы можете щелкнуть мышью на кнопке для того, чтобы выполнить действия, определяемые присоединенным методом. К кнопке вы можете добавить любой текст или рисунок, поясняющий ее назначение.

Хотя вы можете присоединять методы к любому объекту формы, кнопки специально предназначены для этой цели.

В форме можно разместить сколько угодно кнопок и присоединить к ним различные методы. Вы можете присоединить к одной кнопке несколько различных методов, каждый из которых активизируется отдельным событием (разработка методов описана в руководстве по ObjectPAL).

Графика

Графика (graphics) - это графические образы, которые вы можете помещать в поля графического типа Paradox - таблиц, или размещать как независимый графический объект в форме или отчете.

Paradox может импортировать графику из файлов формата .BMP, .EPS, .PCX, .TIP, .GIF или из буфера Windows Clipboard.

Это удобно использовать, например, для создания архива фотографий.

OLE-объекты

OLE (Object Linking and Embedding) - технология связывания и встраивания объектов. Используя технологию OLE, вы можете создавать «контейнеры», которые будут содержать объекты из других приложений Windows.

Так как технология OLE обеспечивает связь между таблицей и исходным файлом встроенного объекта, вы можете, щелкнув дважды мышью над этим объектом, запустить то приложение, в котором был создан этот объект.

Файлы объектов Paradox

В нихе приведены расширения файлов, в которых Paradox хранит основные объекты.

Таблице 1.4 Расширения файлов объектов Paradox

Расширение	Тип объектов
.CFG	Файл конфигурации
.DB	Paradox - таблица
.DBF	dBASE - таблица
.DBT	Файл memo - поля dBASE-таблицы
.FAM	Список связанных файлов Paradox
.FDL	Оттранслированная форма
.FSL	Сохраненная форма
.FTL	Временная форма
.INI	Файл конфигурации системы
.LDL	Оттранслированная библиотека
.LSI	Сохранённая библиотека
.LTL	Временная библиотека
.MB	Файл memo - поля Paradox - таблицы
.MDX	Поддерживаемый индекс dBASE - таблицы
.NDX	Неподдерживаемый индекс dBASE - таблицы
.PX	Первичный индекс Paradox-таблицы
.QBE	Сохранённый запрос
.RDL	Оттранслированный запрос
.RSL	Сохранённый запрос
.RTL	Временный запрос
.SDL	Оттранслированная программа
.SSL	Сохранённая программа
.STL	Временная программа
.TV	Установки параметров Paradox-таблицы
.TVF	Установки параметров dBASE-таблицы
.VAL	Критерии допустимых значений и системы ссылок Paradox-таблицы
.Xnn	Вторичный простой пронумерованный индекс Paradox- таблицы
.Ynn	Вторичный простой пронумерованный индекс Paradox- таблицы
.XGn	Составной вторичный индекс Paradox-таблицы
.YGn	Составной вторичный индекс Paradox-таблицы

Основы представления данных

Paradox представляет собой реляционную систему управления базами данных для персональных компьютеров. В этом разделе вводятся понятия ключей, индексов и системы ссылок между таблицами .

Ключи

Paradox поддерживает два типа формата таблиц - Paradox и dBASE. При использовании таблиц Paradox - формата следует понимать, как работают ключи таблиц (dBASE тоже использует индексы, но в dBASE (до VFP6) нет понятия первичного ключа в том смысле, как в Paradox).

Первичным ключом (primary key), который иногда называют просто ключом (key), является поле (или группа полей), содержащее данные, однозначно идентифицирующие каждую запись в таблице.

Значение ключа должно быть уникальным для каждой записи таблицы. Таблица, у которой определен первичный ключ, называется индексированной (keyed table).

Ключ устанавливает порядок сортировки по умолчанию записей таблицы. Paradox сортирует записи таблицы на основании значений поля (полей - в случае составного первичного ключа), которое вы задали как ключевое. Это позволяет быстро находить записи по значению ключа и совершать другие операции над записями индексированной таблицы.

Paradox допускает пустое значение ключа только у одной записи таблицы. Все последующие записи с пустым значением ключа считаются записями с дублирующимся ключом и в таблицу не допускаются.

Составной первичный ключ

Вы можете задать в качестве ключа либо отдельное поле либо группу полей. Когда в качестве ключа определена группа полей, его называют составным первичным ключом (composite key).

Paradox не допускает присутствие в таблице записей с дублирующимся значением первичного ключа. В случае, когда в таблице создан составной ключ, Paradox позволяет значениям отдельных полей, составляющих первичный ключ, повторяться, но только в тех случаях, когда набор значений полей, составляющих ключ, остается уникальным для каждой записи. Другими словами, поля, составляющие ключ как целое, должны однозначно идентифицировать запись.

Например, таблица Customer (Клиенты) может иметь несколько записей, имеющих значение поля Name1 (Фамилия) «Иванов». Аналогично, может быть несколько записей со значением поля Name2 (Имя) «Петр». Ни одно из этих полей не идентифицирует запись однозначно. Но предположим, что их комбинация в упрощенном случае (Петр Иванов) уникальным образом идентифицирует запись (т.е. среди клиентов нет двух с одинаковыми фамилией и именем). Тогда можно создать для этой таблицы составной первичный ключ, состоящий из сочетания полей Name1 и Name2. Конечно, этого может быть в реальном случае не достаточно. Как правило, следует всегда включать в таблицу достаточное количество полей, чтобы обеспечить уникальность каждой записи таблицы.

Если вы не можете разумным способом создать составной ключ, выходом из этой ситуации является определение поля идентификатора записи (ID field), которое имеет единственное значение для каждой записи таблицы. Например, для таблицы Customer можно ввести поле Customer No (идентификатор клиента), задав каждому клиенту уникальный (и в достаточной степени произвольный) номер.

Индексы

Индекс (index) определяет порядок, в котором Paradox имеет доступ к записям таблицы. Как Paradox, так и dBASE позволяют создавать у таблицы несколько индексов, определяющих различные порядки доступа к записям. Но Paradox и dBASE работают с индексами различным образом.

Когда вы определяете индекс, Paradox создает файл, содержащий значения индексированных полей и порядковые номера записей с этими значениями индекса. Paradox использует индексный файл для определения местоположения записи в таблице по значению индекса.

Индексы можно использовать для просмотра записей в порядке, отличном от определяемого по умолчанию первичным ключом или физическим порядком расположения записей (в случае отсутствия ключа). При этом пересортировки и изменения физического порядка хранения записей в таблице не происходит.

Первичный индекс Paradox - таблицы

Paradox упорядочивает записи в индексированной таблице в соответствии со значениями поля ;(полей), являющегося ключом таблицы. Этот порядок называется первичным индексом (primary index).

По умолчанию все индексы (как первичные, так и вторичные) упорядочивают и позволяют получить доступ к записям в возрастающем порядке значений (от A до Z или от O до 9). Например, если в качестве индекса служит алфавитно-цифровое поле, записи будут упорядочены в естественном для каждого национального языка порядке. Если этот индекс первичный - записи с дублирующимися значениями недопустимы.

В случае составного ключа Paradox создает составной первичный индекс, который упорядочивает записи сначала по первому из полей, составляющих ключ (в соответствии со структурой таблицы), затем по следующему полю и так далее. Причём значения отдельных полей ключа могут быть одинаковы для отдельных записей, но сочетание полей в целом должно быть уникальным.

Вторичные индексы Paradox - таблицы

Работая с таблицами в Paradox, вы можете использовать вторичный индекс (secondary index) для того, чтобы задать альтернативный порядок доступа и отображения записей.

Вторичные индексы могут быть как автоматически поддерживаемые (automatically maintained), так и неподдерживаемые (non-maintained) системой Paradox (первичный индекс всегда поддерживаемый). Если индекс поддерживаем, то Paradox обновляет индексный файл всякий раз, когда вы изменяете таблицу (редактируете значения полей, составляющих этот индекс, добавляете или удаляете записи). Файл неподдерживаемого индекса не обновляется при изменениях таблицы, но может быть открыт явным образом для использования. Для этого необходимо войти в диалоговое окно Order/ Range, чтобы задать индекс, определяющий порядок записей во время работы с таблицей. Paradox позволяет открыть только один неподдерживаемый индекс одновременно.

Когда вы просматриваете таблицу, используя вторичный индекс, физическое расположение записей остается неизменным.

Вторичные индексы могут использоваться также для связывания (linking) нескольких таблиц. Paradox допускает создание составного вторичного индекса (composite secondary index), использующего группу полей таблицы.

Индексирование dBASE - таблиц

Хотя Paradox поддерживает индексные файлы двух форматов: .MDX и .NDX, рекомендуется использовать только формат .MDX. Использование формата .CDX вносит некоторые ограничения. Например, нельзя изменить порядок записей (выполнить сортировку), но если удалить файл с этим расширением (или скопировать его в другое место), то проблемы исчезают.

Система ссылок между таблицами

Система ссылок (referential integrity) обеспечивает соответствие множества значений поля или груп пы полей одной таблицы, называемой дочерней (child table), множеству значений первичного ключа другой таблицы - родительской (parent table). Поля в дочерней таблице, по которым обеспечивается связь таблиц в единую базу данных на основе значений ключа из родительской таблицы, называются заимствованным ключом (foreign key). Система ссылок предоставляет вам несколько способов воздействия на значения заимствованных ключей во всех дочерних таблицах при изменении значений ключа в родительской таблице (поддержание системы ссылок обеспечивается только для Paradox-таблиц.).

Допустим, таблица Orders (Счета - заказы) имеет поле Customer No (Номер Клиента). Вы хотите быть абсолютно уверены, что любое значение этого поля представляет номер, который был присвоен клиенту при занесении его в таблицу Customer (Клиент), связанную с Orders по этому полю. Чтобы обеспечить такое строгое соответствие (для того, чтобы у вас не было счетов, выписанных неизвестно кем), вы можете объявить поле Customer No заимствованным, используя связь с ключом из таблицы Customer. Тогда Paradox каждый раз, как только вы заполняете новый счет и вводите в поле Customer

No идентификационный номер клиента, проверяет, допустимо ли это значение, и есть ли в базе данных клиент с таким номером.

Каскадное обновление

Предположим, что вам нужно изменить значение ключа в родительской таблице. Система ссылок позволит вам автоматически изменить на новое значение все записи в дочерней соответствующими значениями заимствованного ключа.

Продолжая предыдущий пример, предположим, что вы изменили у какого-либо клиента в таблице Customer его идентификационный номер Customer No. Если вы не будете использовать систему ссылок, то все счета-заказы в дочерней таблице Orders не будут отслеживать изменения, происходящие с данными о клиентах из таблицы Customer. При использовании системы ссылок Paradox самостоятельно произведет каскадное обновление (cascade update) соответствующих записей в таблице Orders. Paradox найдет все записи из таблицы Orders, для которых значения заимствованного ключа совпадает со значением ключа родительской таблицы Customer, и заменит их на новое значение идентификационного номера клиента.

Псевдоним

Псевдоним (alias) - это имя, которое можно присвоить каталогу DOS для краткости. Предположим, что вы работаете с базой данных, состоящей из таблиц, текстовых файлов, форм, отчетов, программ и графиков, находящихся в одном и том же каталоге: C:\PARADOX\PRJ\NEW\PLAN. Используя диалоговое окно Alias Manager, вы можете дать этому каталогу псевдоним. Например, если вы зададите псевдоним :MYPLAN:, то тогда вместо полного пути C:\PARADOX\PRJ\NEW\PLAN, вы сможете использовать :MYPLAN: когда вам потребуется доступ к какому-либо из файлов этого каталога.

Использование псевдонимов дает вам несколько преимуществ:

1. Вы избавляетесь от необходимости печатать длинные имена каталогов DOS.
2. Ссылки к файлам в формах, отчетах и подобных объектах Paradox могут использовать имена псевдонимов вместо указания полного пути доступа к ним. Это делает ваше приложение переместимым. Вы можете перенести всё приложение целиком в другой каталог, не изменяя ссылок к файлам, а просто поменяв определение псевдонима. Иными словами, псевдоним является переменной, хранящей наименование каталога в DOS.
3. Вы в любой момент можете изменить определение псевдонима. Тогда все формы, отчеты и другие объекты Paradox автоматически будут ссылаться к файлам из другого каталога. На пример, создав многотабличную форму, использующую данные, расположенные на винчестере вашего компьютера, вы сможете передать эти данные для совместного сетевого использования несколькими пользователями, просто переопределив псевдоним каталога данных на имя сетевого каталога, и тогда форма будет знать, где расположены данные.

Рабочий каталог

Рабочий каталог Paradox - это каталог содержащий таблицы, с которыми вы работаете в данный момент (соответствует текущему каталогу DOS). Рабочий каталог Paradox определяет, какие файлы будут показаны в диалоговом окне, которое открывается если использовать команды меню File / Open или File / Save. Когда вы устанавливаете Paradox на отдельной машине, не подключений к локальной сети, Paradox создает каталог с именем WORKING в своем системном катало-ге Это ваш рабочий каталог по умолчанию.

Вы можете всегда определить любой каталог в качестве рабочего. Paradox присваивает рабочему каталогу псевдоним :WORK:, и если этому каталогу был присвоен ранее другой псевдоним, Paradox будет все равно использовать для ра-бочего каталога псевдоним :WORK:.

Личный каталог

В многопользовательской среде (сетевой вариант) вам необходимо место для размещения временных объектов. Временные таблицы такие как Answer или Inserted (создающиеся в результате запросов) должны храниться в неразделяемом каталоге, иначе другой пользователь, работающий одновременно с вами и запустивший на исполнение запрос после вас, может перезаписать эти таблицы. В локальной сети каждый пользователь Paradox должен задать свой личный каталог для хранения временных объектов.

Файлы, содержащиеся в вашем личном каталоге, будут показаны в диалоговом окне, которое появляется при выполнении команд File / Open или File / Save вместе с файлами из рабочего каталога. Они располагаются в конце списка файлов с префиксом :PRIV: и доступны вам и никому более из пользователей сети.

Задать личный каталог можно командой Tools / Settings / Preferences / Data-base. Paradox присваивает ему псевдоним :PRIV:.

При установке Paradox на машине, не подключенной к сети, личным катало-гом по умолчанию будет каталог с именем PRIVATE в системном каталоге Paradox.

Инспектор объекта

Каждый объект Paradox содержит в себе меню. Для большинства объектов Paradox - таблиц, форм, запросов -это меню содержит команды (такие, как View, Design, Run). В случае конструкционных объектов это меню предоставляет выбор свойств данного объекта (например, цвет, формат представления чисел или стиль отображения текста). Вы получаете доступ к этому типу меню, инспектируя (inspecting) объект. Эта возможность присуща Paradox for Windows, Quattro Pro for Windows и другим продуктам фирмы Borland и называется инспектор Объекта (object Inspector).

Самый простой способ проинспектировать объект щелкнуть объект правой клавишей (клавишей мыши). Появится меню объекта (установить курсор мыши на объект и щелкнуть правой клавишей мыши). Выберите из него нужное свойство или команду.

Предположим, что вы хотите изменить тип рамки прямоугольника. Сначала проинспектируйте его. Появится меню присущих прямоугольникам свойств. Из этого меню выберите Frame / Style - появится палитра Frame Style (Стиль Рамки). Щелкните на стиле, который вы хотите выбрать и Paradox изменит вид рамки прямоугольника. Палитра (palette) . это особый тип меню, визуально представляющий возможности выбора. На любой палитре Paradox есть кнопка - защёлка (snap), позволяющая «прикрепить» эту палитру так, что она не исчезает после однократного выбора свойства, а остается видимой до тех пор, пока ее не «открепят».

Допустим, вы хотите изменить конструкцию формы. В окне Folder или Browser проинспектируйте иконку (icon) объекта формы. Появится меню допустимых команд. При выборе пункта Design, Paradox открывает эту форму в окне Form Design.

Инспектировать в Paradox можно практически все. Конструкционные объекты или иконки объектов - не единственное, что можно инспектировать. Щелкните правой кнопкой мыши, где вам угодно - на сетке таблицы, на одном из инструментов SpeedBar, на оси графика - появится меню данного объекта.

Проинспектировав и изменив свойства объекта, вы модифицируете только данный объект. Например, изменив цвет прямоугольника, вы не меняете цвета всех других прямоугольников на форме. Каждый объект уникален и имеет свои собственные значения свойств. Для сложных объектов, таких как таблицы, доступен большой набор изменяемых свойств. Таблица может иметь у каждого ее столбца, заголовка, линии сетки отдельно задаваемые свойства.

Если вы предпочитаете работу с клавиатурой, то для инспектирования:

- Нажмите Tab для выбора объекта и затем нажмите F6 (в окне Form Design или Report Design).
- Выберите часть таблицы, которую вы хотите проинспектировать, и нажмите соответствующую комбинацию клавиш (F6 - для выбранного поля, Shift+F6 - для всех полей, Ctrl+H – для заголовков, Ctrl-f-G - для сетки и т.п.) (полный список комбинаций клавиш представлен в приложении А).

Для выбора пунктов меню объекта используйте стрелки управления курсором, а затем нажмите Enter.

Глава 2 Работа в Paradox Desktop

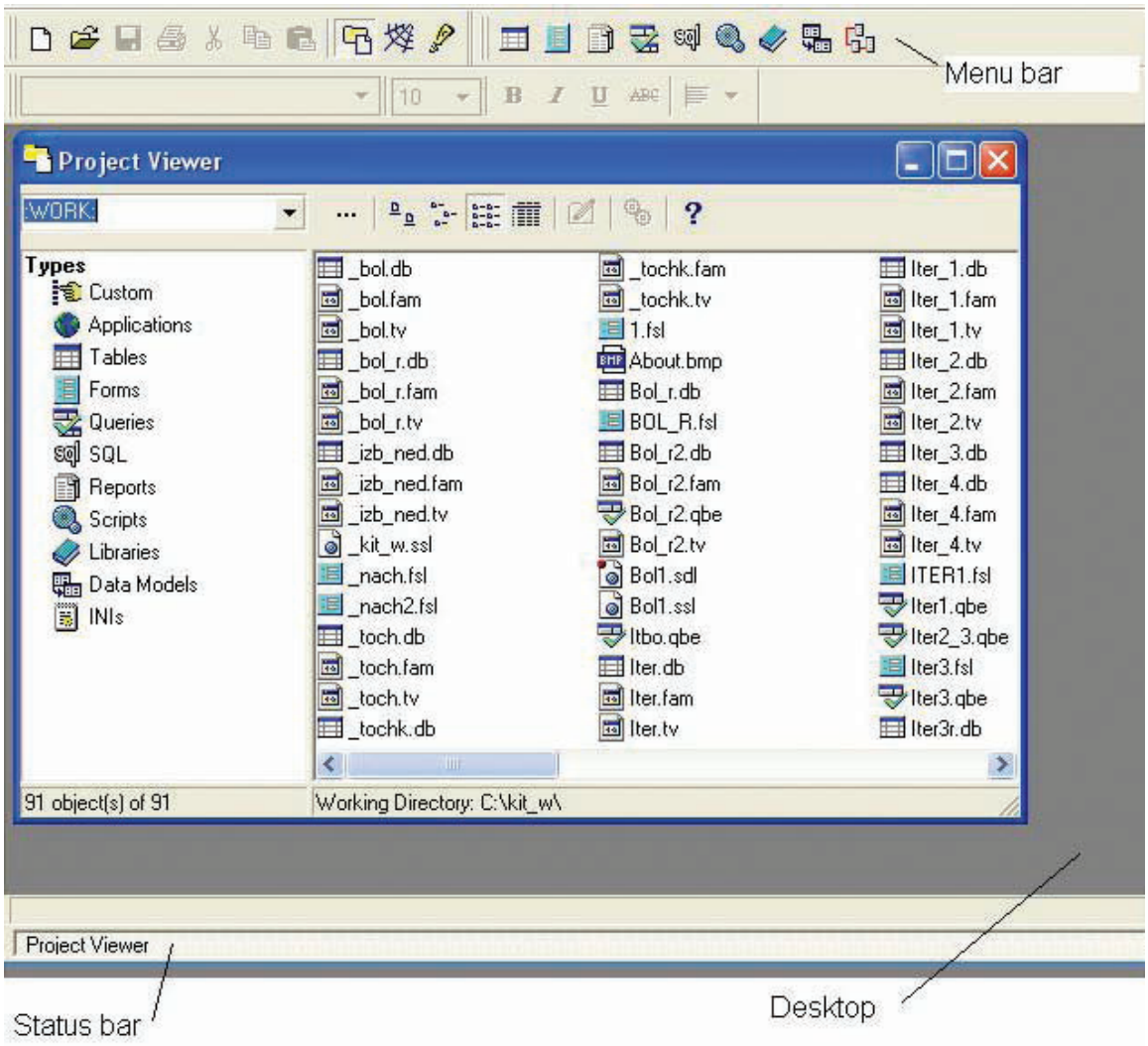


Рис. 2.1 Paradox Desktop

Desktop является «родительским» (parent) окном (рис. 2.1) для любого другого окна Paradox. Используя Paradox Desktop, вы можете:

- Управлять файлами
- Задавать рабочие параметры среды
- Управлять многопользовательским доступом к данным
- Определять и задавать предпочитаемые текущие установки (preferences) и установки по умолчанию (defaults)

Многие из установленных вами параметров среды остаются действительными на протяжении всего сеанса работы. Сеанс работы (session) с Paradox - это время с момента запуска системы Paradox и до завершения работы с ней. Paradox позволяет сохранить во время сеанса параметры среды для использования их в дальнейшем.

Desktop является основным рабочим окном Paradox. Все остальные окна открываются на фоне Desktop.

Каждый видимый объект Paradox (такой, как таблица, форма, запрос, отчет) отображается в своем особом типе окна. Например, формы всегда отображаются в окнах типа Form, запросы - в окне Query, таблицы - в окне Table.

Каждый тип окна обладает специфичным ему набором команд и функций, применимых только к нему. Но так как Desktop содержит все другие окна, команды и функции Desktop доступны им всем. Desktop - это то, что вы увидите, запустив Paradox.

Главное окно Paradox - это его первичная рабочая область. Все окна открываются из главного окна и содержатся в главном окне.

Главное окно является родительским по отношению ко всем остальным окнам Paradox. Все остальные окна являются дочерними. Это означает, что каждое окно является относительно самостоятельным, однако, не может существовать при закрытом главном окне Paradox.

Объект каждого типа (например, таблица, запрос или отчет) появляется на экране в окне особого типа. Например, формы появляются в окне формы, а запросы - в окне запроса.

Окну каждого типа соответствует свой собственный набор команд и функций, применимых только к объекту данного типа. Все команды и функции главного окна Paradox относятся ко всем дочерним окнам.

Пиктограммы объектов

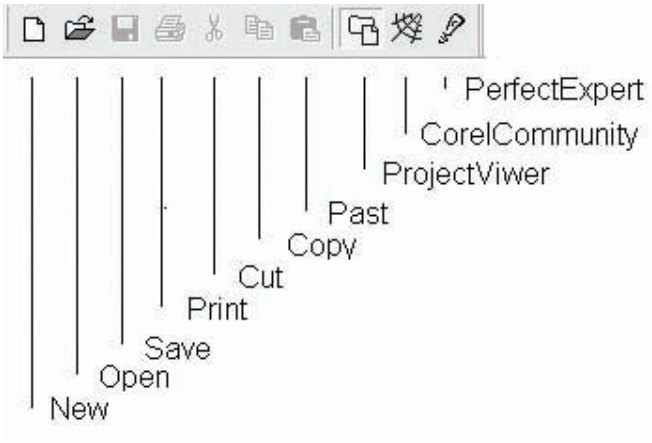


Рис.2.2а Кнопки SpeedBar

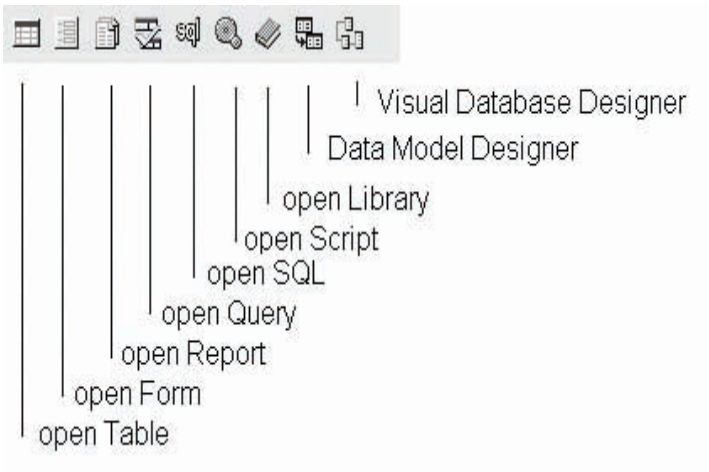


рис.2.2б Кнопки SpeedBar

На рис.2.2а и 2.2б показаны основные кнопки SpeedBar. Полный их комплект можно просмотреть в режиме Tools / Settings / Customize Toolbars. Там же можно указать какие кнопки вывести на экран.

Пиктограммы соответствуют минимизированным дочерним окнам Paradox. Для того чтобы получить справочную информацию о любом из этих окон - укажите на соответствующую ему пиктограмму.

Пиктограммы в папке или окне поиска файлов соответствуют объектам Paradox или ссылкам на эти объекты, например, таблицам, формам, отчетам или запросам. Объекту каждого типа соответствует своя пиктограмма.

Основными составляющими элементами баз данных Paradox являются таблицы. Любые действия, которые выполняет Paradox, всегда непосредственно связаны с таблицами. Данные можно просматривать с помощью таблиц, форм, отчеётов, диаграмм или перекрёстных таблиц и редактировать с помощью таблиц или форм, однако, хранятся данные в таблицах. Нажмите кнопку «Открыть таблицу», чтобы начать работу с таблицей. Нажатие кнопки «Открыть таблицу» на панели управления эквивалентно выбору команды Файл / Открыть / Таблица.

После того как вы нажмите кнопку Открыть таблицу, Paradox откроет окно диалога «Открыть таблицу». Введите имя нужной таблицы или выберите его из списка, затем выберите OK.

Кроме того, можно указать на эту кнопку и нажать правую кнопку мыши, а затем выбрать Открыть.

С помощью форм можно представить информацию (табличные данные, графические изображения, результаты вычислений и т.п.) в надлежащем виде. Кроме того, в форме можно объединить информацию из нескольких таблиц и разместить в ней объекты интерфейса. Использование форм позволяет существенно упростить ввод данных. Нажмите кнопку «Открыть форму», чтобы начать работу с формой.

Нажатие кнопки «Открыть форму» на панели управления эквивалентно выбору команды Файл / Открыть / Форма.

После того как вы нажмите кнопку «Открыть форму», Paradox откроет окно диалога «Открыть документ». Введите имя нужной формы или выберите его из списка. Выберите «Просмотр» (чтобы

просмотреть данные с помощью формы) или Конструктор (чтобы изменить макет формы), затем выберите OK. Paradox откроет указанный файл в окне формы.

Для того чтобы открыть форму как отчет, выполните описанные выше действия, а также выберите Отчет в разделе «Открыть как» в окне диалога «Открыть документ».

Кроме того, можно указать на эту кнопку и нажать правую кнопку мыши, а затем выбрать Открыть, чтобы открыть форму выбрать Создать, чтобы создать новую форму.

Запрос в Paradox - это вопрос о табличных данных, заданный пользователем. Запросы могут быть простыми, извлекающими информацию из одного поля таблицы, или сложными, объединяющими информацию из нескольких таблиц.

Для составления запросов Paradox использует способ, который называется запросом по образцу. Этот способ заключается в следующем: пользователь описывает образец данных, которые следует получить, а Paradox пытается найти в указанных пользователем таблицах информацию, удовлетворяющую этому образцу. Такое распределение обязанностей позволяет пользователю сосредоточиться на том, что следует получить, и не задумываться над тем, какие действия следует выполнить, чтобы извлечь указанную информацию.

Составление запросов в Paradox является гибкой, интерактивной и итеративной процедурой. Если запрос не дает нужного результата, его можно быстро исправить и выполнить еще раз. Кроме того, Paradox позволяет составлять запросы по результатам предыдущих запросов, тем самым проигрывая ситуации типа «А что если?».

Нажмите кнопку Открыть запрос, чтобы просмотреть или выполнить запрос. Нажатие кнопки Открыть запрос на панели управления эквивалентно выбору команды Файл / Открыть / Запрос.
После того как вы нажмёте кнопку Открыть запрос, Paradox откроет окно диалога «Выбор файла». Введите имя нужного запроса или выберите его из списка, затем выберите OK. Paradox откроет указанный файл в окне запроса.

Кроме того, можно указать на эту кнопку и нажать правую кнопку мыши, а затем выбрать Открыть, чтобы открыть запрос выбрать Создать, чтобы создать новый запрос.

Формирование (использование) SQL запросов. Структура их соответствует другим языкам программирования, но надо заметить, что в Paradox система Запросов намного проще и мощнее.

Нажмите кнопку Открыть программу, чтобы просмотреть или отредактировать автономную программу на языке ObjectPAL. Нажатие кнопки Открыть программу на панели управления эквивалентно выбору команды Файл / Открыть / Программа.

Автономная программа - это форма, у которой нет окна и в которой не размещены никакие объекты. Текст автономной программы редактируется точно так же, как и любой другой объект Paradox.

После того как вы нажмёте кнопку Открыть программу, Paradox откроет окно диалога «Открыть документ». Введите имя нужной программы или выберите его из списка. Выберите Запуск, чтобы запустить программу, или Конструктор, чтобы открыть окно редактора ObjectPAL, в котором можно отредактировать программу.

Кроме того, можно указать на эту кнопку и нажать правую кнопку мыши, а затем выбрать Открыть, чтобы открыть программу выбрать Создать, чтобы создать новую программу.

Нажмите кнопку Открыть библиотеку, чтобы просмотреть или отредактировать библиотеку ObjectPAL. Нажатие кнопки Открыть библиотеку на панели управления эквивалентно выбору команды Файл / Открыть / Библиотека.

Библиотека - это объект Paradox, который используется для хранения программ на языке ObjectPAL. Содержимое библиотеки может использоваться совместно разными формами, программами и другими библиотеками.

После того как вы нажмёте кнопку Открыть библиотеку, Paradox откроет окно диалога «Открыть документ». Введите имя нужной библиотеки или выберите его из списка. Paradox откроет окно библиотеки.

Вызовите меню окна библиотеки и выберите метод, который требуется изменить. Paradox откроет окно редактора и загрузит в него указанный метод.

Кроме того, можно указать на эту кнопку и нажать правую кнопку мыши, а затем выбрать «Открыть», чтобы открыть библиотеку выбрать Создать, чтобы создать новую библиотеку.

Для того чтобы добавить объект в папку, выберите Папка / Добавить или нажмите кнопку До-

бавить объект в папку на панели управления. На экране появится окно диалога «Выбор файла». Выберите файл, который следует добавить в папку.

Для того чтобы включить в список файлы другого типа, используйте список Тип. Для того чтобы включить в список файлы из другого каталога, используйте кнопку Поиск.

Примечание: При желании можно добавить в папку сразу несколько пиктограмм объектов. Для этого в окне диалога «Выбор файла» выделите все нужные файлы и выберите OK.

Все пиктограммы в папке являются ссылками. Пиктограмма показывает имя файла, который содержит данный объект, и его расширение. В строке состояния отображается полное имя или псевдоним каталога, в котором находится этот объект. Paradox позволяет поместить в папку две пиктограммы с одинаковыми именами, если они ссылаются на файлы, расположенные в разных каталогах.

Для того чтобы включить в папку все объекты Paradox, которые находятся в рабочем каталоге, выберите Папка / Включить все файлы.

Для того чтобы удалить пиктограмму объекта из папки, выберите Папка / Удалить или нажмите кнопку Удалить объект из папки на панели управления.

На экране появится окно диалога «Удаление объекта из папки». В нем перечислены все объекты, которые в текущий момент содержатся в папке. Выберите объекты, которые следует удалить из папки. Объекты, которые следует удалить, необходимо выделить в этом окне диалога. При этом не учитываются пиктограммы, выделенные ранее в папке.

Удаление объекта из папки не удаляет его с диска. Файл продолжает существовать, он просто не отображается в папке.

Примечание: Кроме того, можно удалить пиктограмму из папки, если выделить ее и нажать Del. Файл продолжает существовать, он просто не отображается в папке.

Кнопка «Open project viewer» (окно файлов, которые подключены к пиктограммам) открывает окно, которое показано ниже. Все перечисленные действия относятся и к пиктограммам, находящимся в этом окне. Кроме этого можно очень легко изменять Рабочий каталог (Working Directory).

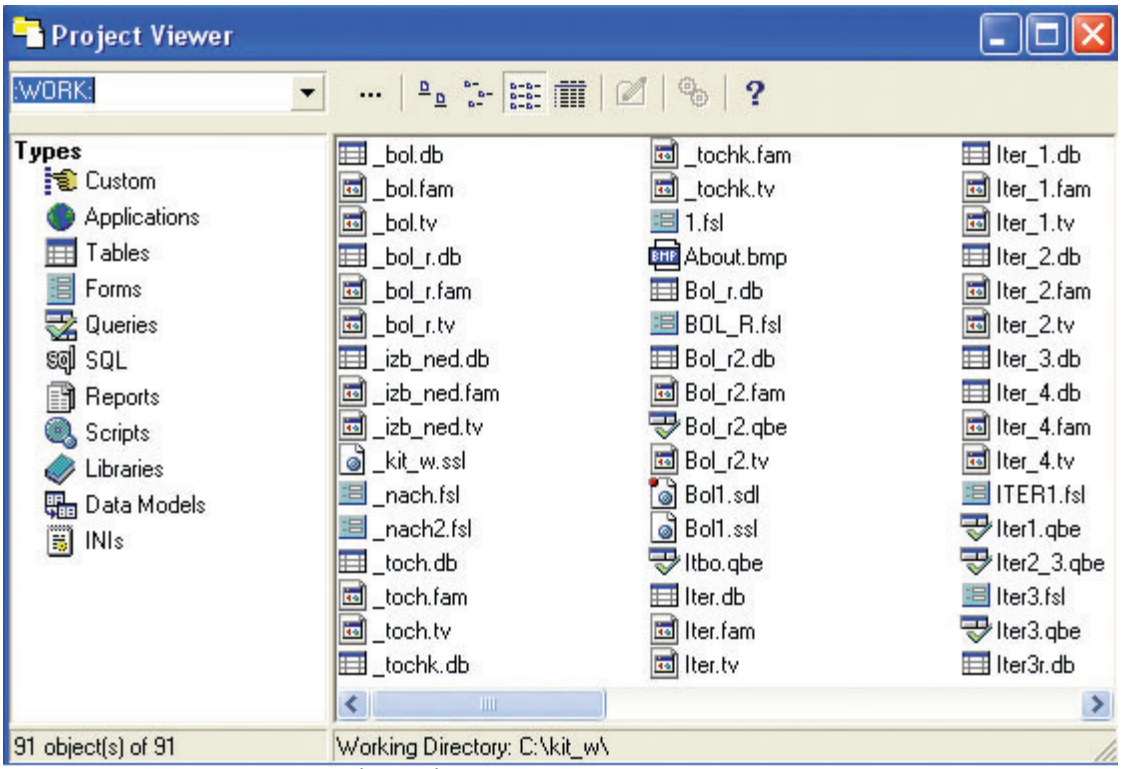


Рис.2.3 Окно Project Viewer

Установка характеристик Desktop

Paradox хранит установки параметров (свойств) Desktop в файле PDOXWIN.INI. Выбрав Tools / Setting / Preferences вы откроете диалоговое окно Preferences (Рис. 2.4).

В поле Title можно ввести текст заголовка окна Desktop - он заменит текст по умолчанию «Paradox».

При желании можно задать рисунок, который будет появляться в качестве фона Desktop. Для этого в поле Background Bitmap введите имя графического файла или нажмите кнопку Find для того, чтобы открыть диалоговое окно Select File (выбор файла).

В списке файлов будут показаны все графические файлы в рабочем и личном каталоге. Вы можете выбрать один из них или используя список псевдонимов Path, выбрать файл из другого каталога, или, выбрав Browse, открыть диалогового Browser в нем осуществить выбор диска, каталога и графического файла.

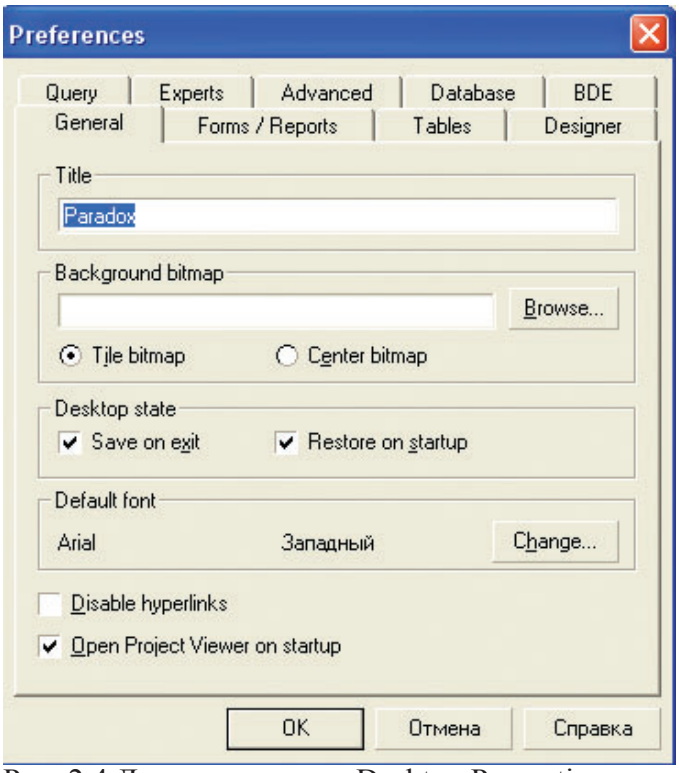


Рис. 2.4 Диалоговое окно Desktop Properties

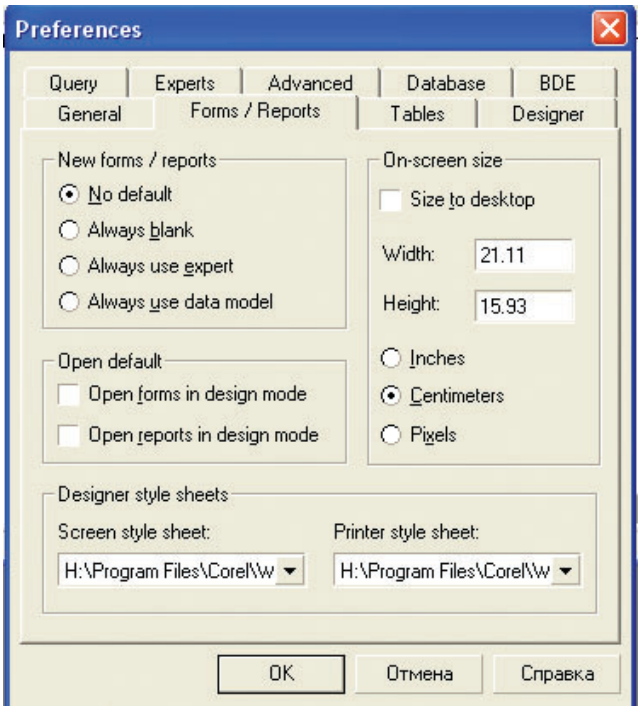


Рис.2.5 установки для форм и отчетов

Кнопка Change в разделе Default Font позволяет выбрать тип шрифта, который будет использоваться «по умолчанию»

Кнопка Forms and Reports открывает диалоговое окно установки исходных значений для начала проектирования форм и отчётов, вид которого показан на рис.2,5. Блок New Form / Reports задет начальные условия:

- No default –(по умолчанию) означает, что этих условий нет
- Always Blank –всегда чистый
- Always Use Expert –всегда помощь эксперта (помощника)
- Always Use Data Model –всегда окно с перечнем баз, включенных в форму - отчет

В блоке Form Screen Page Size можно задать стандартные размеры для каждой новой формы. В исходном состоянии они привязываются к размеру рабочего стола Windows, а на рисунке показан размер для окна 600x800.
В блоке Open Default можно установить, чтобы все формы / отчеты открывались в окне редактора.

Аналогично производятся установки для остальных окон Preferences. Обычно все режимы оставляются без изменений. Исключение составляют два окна:

- Designer - для установки параметров сетки, которая используется при проектировании форм и отчетов
- Database – для установки личного (:PRIV:) каталога

Создание новых объектов

Рис.2,6 открывает окно при выполнении пунктов меню File / New /... Далее можно выбрать тип объекта, который будем создавать.

Не выделенные пункты меню (бледный шрифт) – не активизированны. Метод ак-тивации только тех пунктов меню, которые подходят к конкретной пиктограмме, используется во всех режимах.

Вызов существующего объекта

Для вызова существующего объекта надо выбрать режим File / Open, после чего появится подменю рисунка 2.6. После выбора объекта в очередном окне высветятся все эти объекты, находящиеся в Рабочем (Working) и Личном (Private) каталогах.

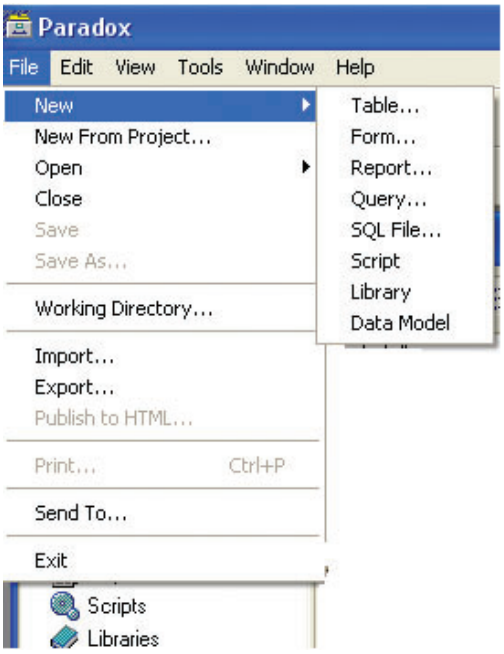


Рис.2.6 Пункты меню File/new

Создание в вызов объектов через Project Viewer

Окно Project Viewer показано на рис.2.3. Став на объект надо нажать правую кнопку мыши, надо выбрать New для создания нового объекта. Все существующие объекты выбранного типа высветятся в окне Project Viewer. Если курсор находится в окне перечня объектов, то поиск можно ускорить, нажав первую букву имени или очень быстро две буквы. Файлы личного каталога находятся после признака :priv:

Отличие между Working Directory… и Private Directory…

Как говорилось выше, результаты некоторых действий помещаются в Личный каталог. Это относится к вставкам, запросам, удалениям, где создаются соответствующие временные таблицы. Вторая причина - работа в сети (многопользовательский режим), где надо исключить влияние других пользователей на результат своей работы.

При использование языка ObjectPAL для файлов рабочего личного каталога перед файлом надо указывать его принадлежность к этому каталогу. Для рабочего этого делать не надо.

Использование Aliases

Иногда возникают ситуации, когда в работе используются базы других за-дач, например, делать выбор оплат по абонентской плате. Каталог, где находятся эти данные, находится далеко. Чтобы избежать необходимости каждый раз совершать далекие путешествия по диску, можно один раз задать этот каталог и потом просто его использовать в работе. Для задания надо выбрать Tools / Alias Manager.

Внешний вид окон при открытии объектов

Если Вы делаете случайные выборки, программируете или выполняете другие действия, которые не заботятся о внешнем виде, но дают максимальный доступ к базам, то обычно работают с Project Viewer.

На рис. 2.6 перечень объектов при открытии или создании нового. О общих чертах для всех них процесс одинаковый, но в каждом случае имеются свои нюансы.

Ниже показаны окна, которые открываются для каждого из указанных объектов при работе с Project Viewer.

На рис 2.7. показано окно для открытия таблиц. Окно Drive (or Alias) дополнительно открыто, чтобы показать возможность выбора баз данных. Как видно на рисунке, выбор возможен из рабочего

каталога, трех приложений (Alias), личного каталога и любого диска.

На рис 2.8. показано окно, которое открывается при переходе курсора на конкретную таблицу и нажатии правой кнопки мыши.

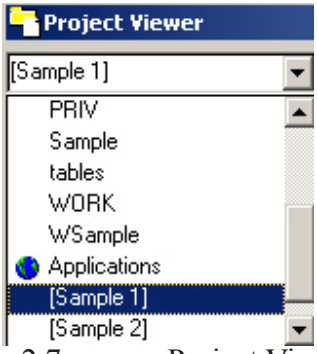


Рис.2.7 режим Project Viewer выбор директории

Дальше будут рассматриваться режимы открытия объектов. Создание новых и более детальная работа с уже созданными будет рассматриваться детально в каждой соответствующей главе.

На рис 2.9.показано окно открытия форм. Режим View Data открывает окно с данными в том режиме, как оно сохранено (в обычном виде конструктора с пиктограммами или в диалоговом режиме, где только те объекты, которые установил программист)

Режим Design всегда открывает окно конструктора.

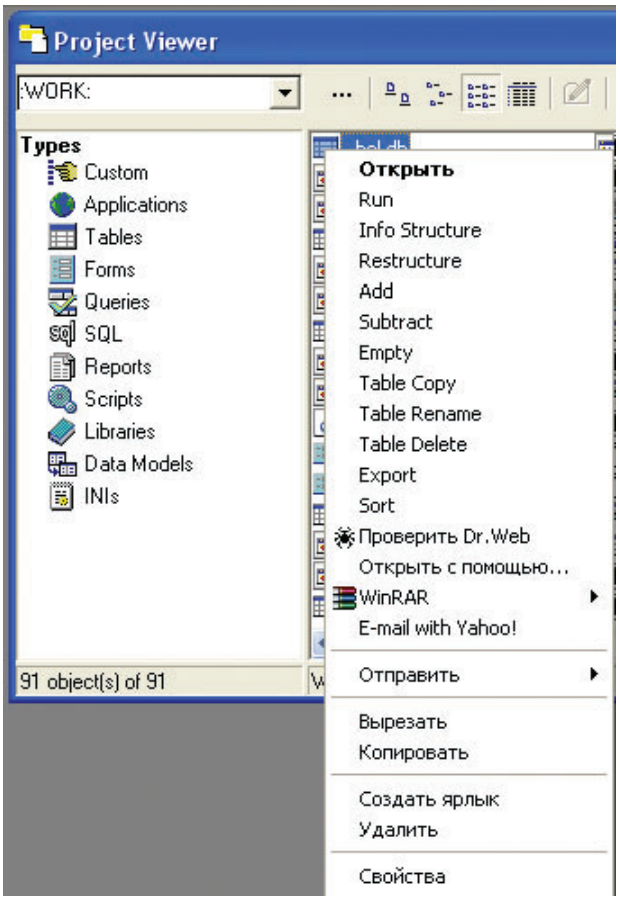


Рис.2.8. Table - окно баз - правая кнопка мыши

Режим Open As позволяет открыть форму в виде экранной – form (что видим на мониторе) или в форме отчета – report (что выводится на печать)

При открытии запроса (рис 2.10) режим Run the query выполняет запрос, сам запрос и результат открывается. В режиме Design открывается сам запрос, но на выполнение не отправляется. В случае, если поле таблицы, используемое в запросе удалено (произвели реструктуризацию таблицы и удалили, переименовали поле или изменили тип), то запрос не откроется и будет выдано сообщение об ошибке. Добавление новых полей и изменение их порядка роли не играет.

Окно запросов SQL аналогично предыдущему.

При открытии отчета (Reports)

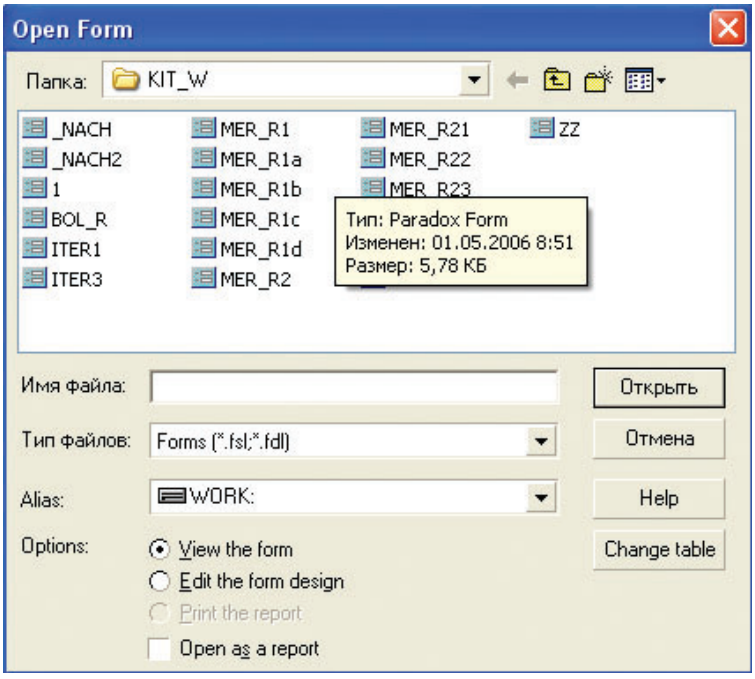


Рис.2.9 Окно открытия форм

отличие от окна Form в том, что отчет можно сразу распечатать (режим Print). В остальном окна одинаковы.

Окно Scripts похоже на предыдущие. В режиме Play программа просто выполняется. В режиме конструктора (design) открывается листинг программы, но на выполнение не отсылается.

Открытие библиотек (libraries) происходит только в режиме Design и окно аналогично предыдущим

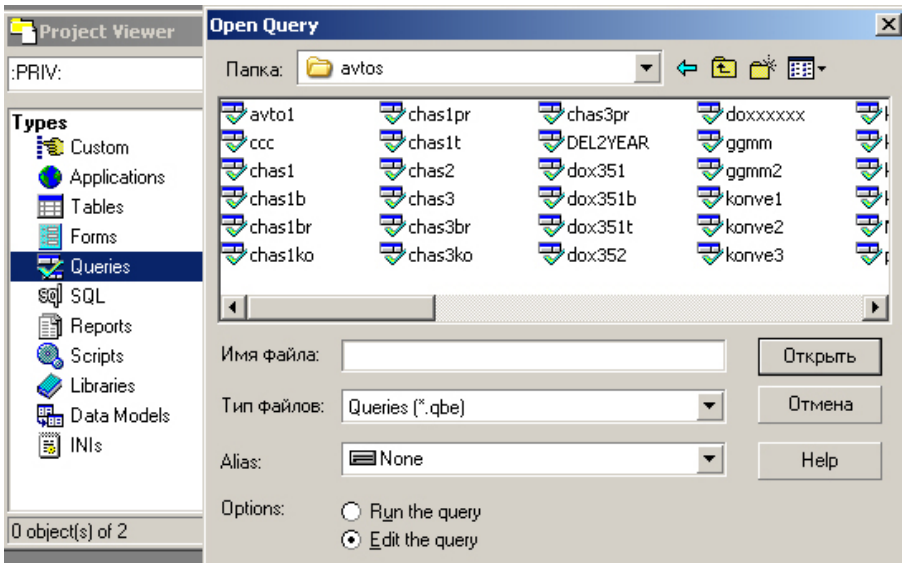


Рис. 2.10 окно открытия Запроса

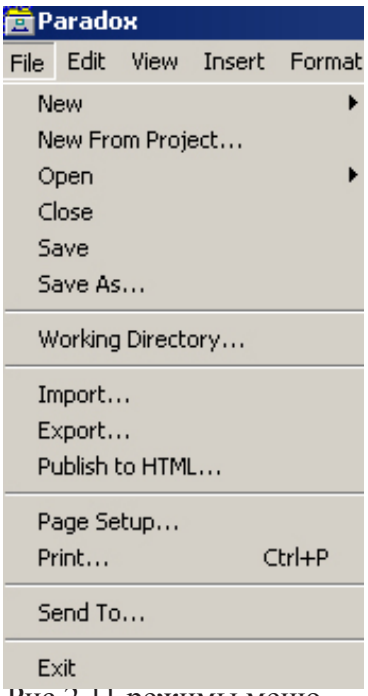


Рис.2.11 режимы меню File

Сохранение объектов

Если открыт уже созданный объект и нажата хоть одна кнопка клавиатуры, то при стандартном закрытии окна Paradox сам предложит сохранить или игнорировать изменения.

В отличии от рис 2.6 в режиме меню File будут активизированы закрытые окна (рис 2.11.) Работая в диалоговом окне Save File As, вы можете: задать новое имя объекту. При создании нового имени надо иметь в виду, что точно такого имени не должно быть ни в одном из используемых Вами каталогов. Иначе возникает ситуация, когда новый записывается вторым, а в работу берется первый.

При создании имени во всех случаях желательно, чтобы первой была буква латинского алфавита (это относится и именам полей в таблицах)

Печать документов

Для простых пользователей, которые работают в «Диалоговом режиме» этот раздел значения не имеет, но для программистов Этот раздел является наиболее неопределенным, но не по причине недостатков Paradox, а из – за различия в кодировках и самих системах Windows (win 9x, 3.1,2000,XP. Дело в том, что Paradox создает свои условия работы принтера, а Windows накладывает свои. И в этом случае Windows обычно побеждает. Исходя из этого, отчёты и формы (в виде отчетов) следует настраивать по установкам Windows (для всех остальных объектов это роли не играет, т.к. они не используются в виде «документа»)

Как и в Windows, Paradox использует для печати пиктограммы принтера (или file/print - тут используются стандартные установки Windows). Если открыта форма, ее можно распечатать или как форму или как отчет. Выбор способа представления определяется в окне Page Setup (рис 2.12)

При работе с этим режимом обязательно делать соответствие с установками Windows.

Иконки объектов Paradox

Иконка (icon) - это графическое представление объекта. Когда окно минимизировано, оно представлено на Desktop в виде иконки. Каждый тип объекта имеет свой собственный тип иконки. Имя объекта, который иконка представляет, располагается непосредственно под ней.

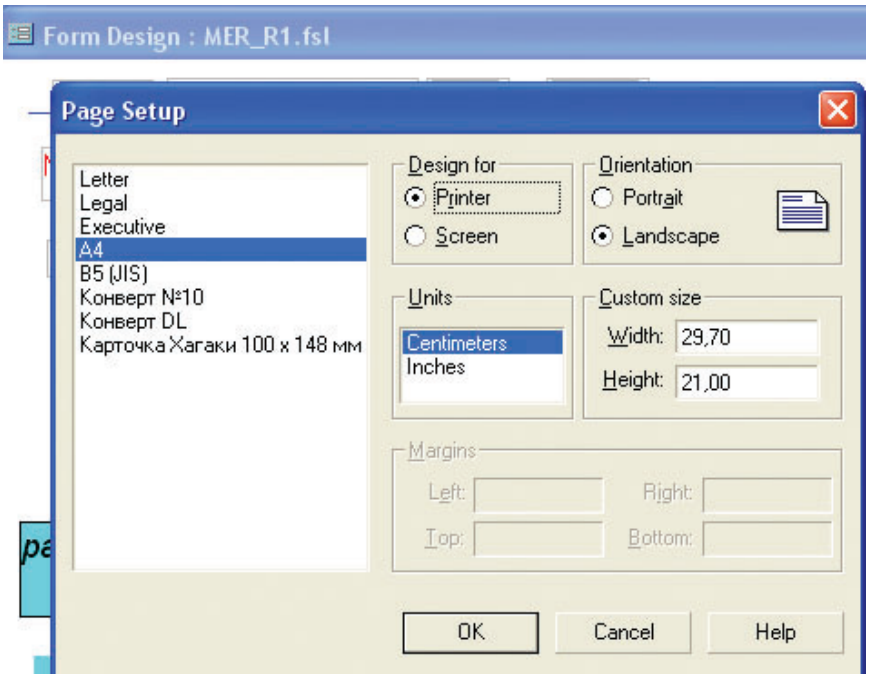


Рис. 2.12 Диалоговое окно Printer Setup

Упорядочивание иконок

В окне Project Viewer выбирается п иктограмма вида объектов (эти пиктограммы) находятся рядом с окном названия директории). Если перечень объектов представлен в полном виде, то ,как и в Windows, их можно отсортировать по имени, размеру, нахождению, времени изменения (создания) щелкнув мышью по соответствующему названию.

Все файлы можно просмотреть выбрав Custom в окне Project Viewer.

Удаление иконок

В окне Project Viewer выбирается объект, правой кнопкой вызывается окно свойств, где выбирается режим удаления. В результате файл перемещается в корзину Windows.

Получение справки

Одним из способов доступа к справочнику системы Paradox является меню Help. Кроме того, вы можете щелкнуть мышью любую кнопку с надписью Help или нажать F1 в любой момент времени.

При нажатии клавиши F1 Paradox сам выбирает подходящую в данных обстоятельствах справочную информацию (контекстно зависимая справка).

- Меню Help используется для выбора предмета, по которому необходимо получить справку.
- Если вы щелкните кнопку Help в диалоговом окне, вы получите справку, касающуюся работы в данном окне.
- При нажатии F1 Paradox сам выберет подходящую в данном случае справку

Установка параметров многопользовательского доступа

Для установки параметров режима использования баз данных в локальной сети и для получения ин¬формации о других пользователях многопользовательской среды используйте BDE Administrator (можно открыть через ПУСК / Панель управления / BDE Administrator или ПУСК / Все программы / Paradox 9 / Utilties / Borland Database / Engine строка NET DIR (рис. 2.13).

Традиционно, термин «многопользовательский» относится к локальным сетям. Но иногда вы можете попасть в многопользовательскую ситуацию, работая на машине, не подключенной к сети.

Например, если вы открыли таблицу, Paradox блокирует к ней доступ (lock). Это означает, что вы можете быть уверены в том, что пока вы просматриваете таблицу, она никем не изменяется и не может быть удалена.

Иногда эта автоматическая блокировка не позволяет вам совершать некоторые действия с та-

блицей, например, ее удаление. В данном случае различные окна выступают как отдельные пользователи.

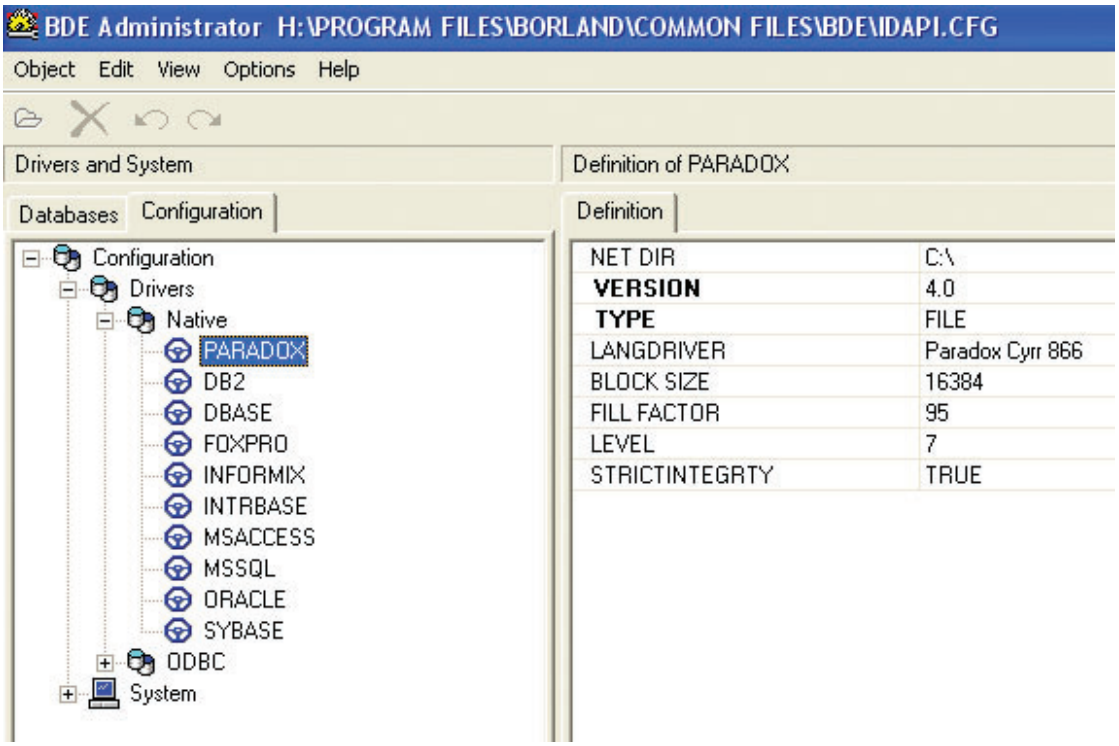


Рис. 2.13 установка для локальной сети

Просмотр информации о блокировках таблиц

Выберите пункт меню File / Multiuser / Display Locks для того, чтобы просмотреть информацию о блокировках таблицы. В появившемся диалоговом окне Select File укажите интересующую таблицу. Если вы работаете в многопользовательской среде, вам необходимо знать правила блокировки отдельных записей и таблицы в целом.

Установка блокировок

Для некоторых операций (например, редактирования и ввода данных) вам требуется доступ для изменения только одной записи в данный момент времени. Paradox управляет такого рода доступом, используя автоблоки-ровку записи (record lock), которую автоматически накладывает и снимает. Блокировка записи предотвращает ситуацию, когда два пользователя одновременно редактируют одну и ту же запись. Как только вы перешли на другую запись, Paradox снимает блокировку с предыдущей.

Вы можете блокировать записи вручную, выбрав команду Record / Lock или нажав клавишу F5, находясь в режиме редактирования Edit. Как только вы заблокировали запись, никакой другой пользователь не сможет изменить или удалить ее до тех пор, пока вы не снимете блокировку. Блокировка снимается при следующих операциях:

- Переходе на другую запись
- выборе команды Record / Unlock
- Нажатии комбинации клавиш Shift+F5

Paradox заносит отредактированную запись в таблицу после снятия блокировки. Если вы хотите занести изменённую запись в таблицу не снимая блокировки, выберите пункт Record / Post / Keep Locked или нажмите Ctrl+F5. Некоторые операции (например, изменение структуры таблицы или ее удаление) требуют большего, чем блокировка одной записи. При этих операциях вам требуются большие права доступа к таблице, чем другим пользователям. Paradox предоставляет возможность на уровне Desktop явно запросить и отменить блокировку файла.

Блокировка этого уровня отличается от блокировки записи тем, что

- Блокируется вся таблица целиком.
- Обеспечивается больший уровень защиты.
- Эту блокировку следует накладывать и снимать явным образом (вручную).

Для блокирования таблицы выберите пункт Tools / security /Set Locks, чтобы открыть диалоговое окно Table Locks (рис. 2.14). dBASE -таблицы не поддерживают Read Lock (Блокировка Чтения), поэтому когда вы выбираете из списка таблиц .DBF -файл, эта опция недоступна.

Если вы хотите просмотреть текущие блокировки таблицы (на уровне записи или всей таблицы), которые наложили вы или другие пользователи, используйте команду Tools / Multiuser / Display Locks.

Для блокирования таблицы выберите ее и задайте тип блокировки. Выбрать можно только один из перечисленных ниже типов:

- No Lock - означает, что вы снимаете все блокировки на уровне Desktop. Это не означает, что таблица обязательно будет целиком разблокирована. Другие пользователи при этом могут наложить блокировку на отдельные записи или на всю таблицу.
- Open Lock - означает блокировку, накладываемую системой Paradox автоматически на любую таблицу, которую любой пользователь, включая вас, открывает. Такой тип блокировки сохраняет таблицу от наложения на нее единоличной (exclusive) блокировки любым из пользователей. Все пользователи должны снять Open Lock для того, чтобы на эту таблицу можно было наложить единоличную блокировку. Этот тип блокировки можно наложить не открывая таблицы.
- Read Lock - означает, что вы можете и читать и писать данные в таблицу. Все другие пользователи могут только читать данные. Когда вы наложили на таблицу блокировку чтения, ни один из пользователей не может наложить на нее блокировку, которая не позволит вам читать из нее ваше право на чтение данных гарантировано. Однако другие пользователи могут наложить блокировку записи, что даст им возможность записывать данные, а вам это будет запрещено.
- Write Lock - означает, что вы можете читать и записывать данные в таблицу. Все другие пользователи могут читать данные, но не могут редактировать записи. Другие пользователи могут наложить на таблицу, которую вы блокировали от записи, блокировку чтения.
- Exclusive Lock - означает, что вы имеете право чтения и записи данных, и никто другой не может этого делать. Единоличное право доступа можно получить, если ни один из пользователей не наложил на таблицу блокировку типа Open, Read, Write.

Paradox сохраняет блокировку до тех пор, пока вы не завершите сеанс работы или не снимите блокировку командой No Lock.

Уровень блокировки	**Ваши права**	**Права других пользователей**	**Блокировки, которые могут наложить другие пользователи**
None	никаких	полные	Любые
Open	Read (и write, если ни один из других пользователей не наложил блокировку read)	Read, write	Любые кроме exclusive, если не заперта ни одна запись, в противном случае только open
Read	Read (и write, если ни один из других пользователей не наложил блокировку read)	Read	Pen, Read
write	Read, write	Read	Open
exclusive	полные	никаких	Никаких

Установка времени обновления экрана

Для того, чтобы задать период, который проходит перед автоматическим обновлением отображения таблиц на экране, используйте Tools / Settings / Preferences / Database/ Refresh. Это удобно, когда вы работаете с таблицами в локальной сети, которые могут изменяться без вашего участия. Эта возможность доступ-на только для Paradox-таблиц. Когда вы выберете Auto Refresh, появится диало-говое окно Network Refresh Rate (рис. 2,14). Введите в Number of Seconds for Refresh период обновления экрана в секундах (0 - для того, чтобы выключить Auto Refresh).

Учтите, чем чаще вы обновляете экран, тем чаще вы обращаетесь к локальной сети.

Интерпретация пустых числовых полей

При необходимости вы можете указать системе Paradox, чтобы пустые числовые поля интерпретировались как число 0. По умолчанию Paradox рассматривает пустые числовые поля в запросах, формах и отчетах как не отличающиеся от пустых алфавитно-цифровых полей и отображает в них пробелы. Для установки типа отображения пустых полей используется метка Treat blank fields as zeros (рис.2.14)

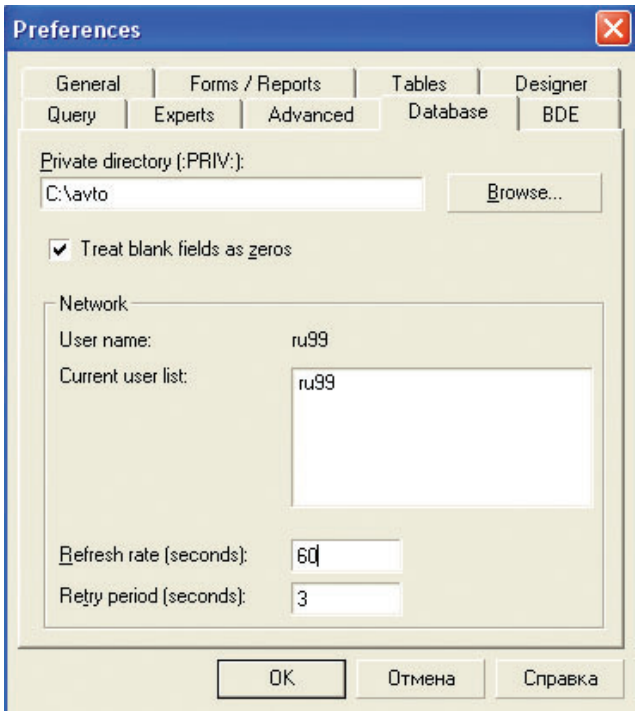


Рис.2.14 диалоговое окно Network Refresh Rate

Информация о драйверах

Выберите пункт Tools / Settings / Preferences/ BDE для того, чтобы открыть диалоговое окно Drivers. В нем показаны типы драйверов баз данных, которые вы можете использовать.

IDAPI-информация (справочно) для Paradox7

ODAPI - это ядро системы управления базами данных Paradox. Для того, чтобы просмотреть текущие установки параметров ODAPI, выберите Tools /System Settings/ IDAPI и откройте диалоговое окно IDAPI System Information.

Глава 3 Разработка и изменение структур таблиц

Таблицы - основные строительные блоки системы Paradox : все действия, которые производятся в Paradox, каким-либо образом связаны с таблицами.

В этой главе рассматриваются вопросы разработки и изменения структуры Paradox и dBASE-таблиц. Вы узнаете, как определять имена, типы и размеры полей таблицы.

Разработка таблиц

Первым шагом разработки таблицы является продумывание ее структуры. Вам необходимо решить, какую информацию будет таблица содержать и в каком порядке она должна располагаться. При разработке таблицы придерживайтесь следующих правил :

- Избегайте повторения полей. Это обеспечит более гибкое хранение данных и простой доступ к ним. В этом состоит отличие разработки таблиц базы данных от организации данных в системах электронных таблиц.
- Если будет необходимость просмотреть данные в формате, аналогичном электронным таблицам, можно по вашей таблице построить кросстаблицу.
- Старайтесь быть исчерпывающими. Включайте в таблицу поля для всей информации, которая вам понадобится, но не забивайте таблицы ненужными данными. Если впоследствии вам понадобится какое-либо дополнительное поле, вы сможете без проблем ввести его в вашу таблицу.
- Используйте небольшие таблицы. Если вам необходимо организовать в базу данных большой объем информации, то обычно, лучше разместить ее в несколько небольших взаимосвязанных таблиц, чем в одну всеобъемлющую.
- Разрабатывайте таблицы с привычной для вас структурой. Лучше, чтобы ваши таблицы были похожи на форматы документов и файлов, с которыми вам уже приходилось работать.
- Избегайте избыточности. Не следует дублировать информацию в таблицах за исключением полей, служащих для организации связей между таблицами.
- Решите, какой тип таблиц вам нужен. Взвесьте преимущества и недостатки Paradox и dBASE-таблиц. Например, dBASE - таблицы позволяют хранить логические величины, в то время как Paradox - таблицы могут содержать форматированные memo-поля, графики и OLE-поля. Перед тем как выбрать тип таблицы, решите, какого вида информация должна в ней храниться.

Выбор типа таблицы

Для создания новой таблицы выбрать File/new/ Table или в Project Vievers выбрать Table/правая кн.мыши/ New. В результате сначала будет предложен выбор создания таблицы при помощи Эксперта или самостоятельно (blank), зхатем окно выбора типа таблицы (рис.3.1), где будут указаны те типы таблиц, которые Paradox воспринимает без изменения их структур. Для выбора типа, отличного от Paradox for Windows, щелкните мышью маркер раскрывающегося списка, и выберите желаемый тип таблицы.

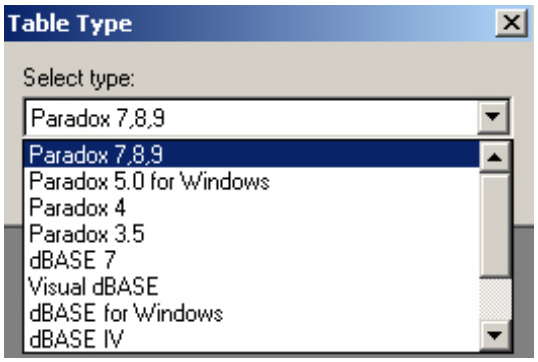


Рис. 3.1 Диалоговое окно Table Type

Если вы выберите тип Paradox for Windows в диалоговом окне TaЬIe Type, то вслед за этим на экране появится диалоговое окно Create TaЬIe (рис. 3.2). Оно служит для :

- Ввода имен полей таблицы
- Определения типа и размеров полей

Кроме того, вы можете :

- Определить ключевые поля
- Назначить проверку значений в каждом поле
- Определить вторичный индекс таблицы
- Назначить для данной таблицы таблицу-справочник

- Определить систему ссылок к другим таблицам
- Назначить пароль доступа к таблице или к ее отдельным полям

Пример.
Создание Paradox - таблицы

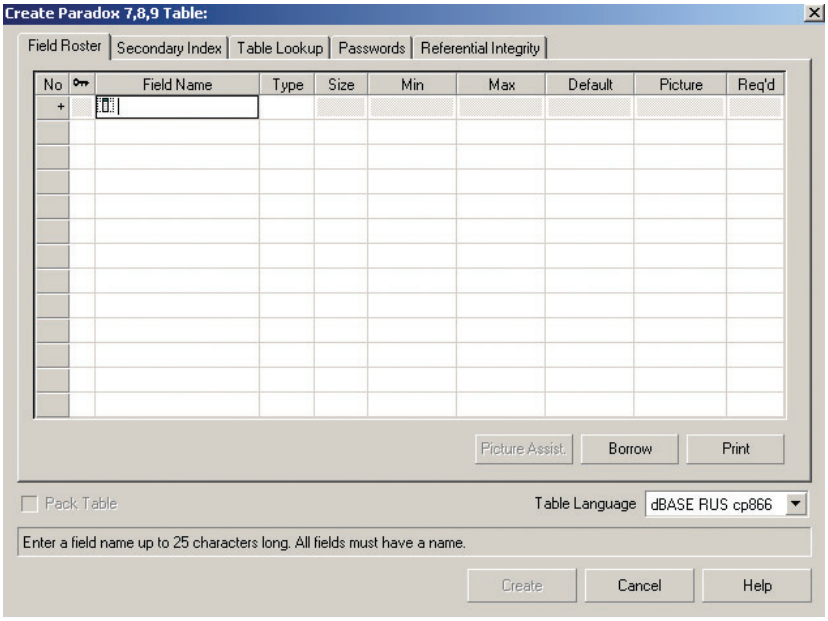


Рис. 3.2 Диалоговое окно Create Table

Предположим, вы хотите создать простую таблицу без индексов.

1. Введите имя первого поля в колонке Field Name перечня полей (Field Roster) Правила выбора имен приведены ниже в данной главе.
2. Переместитесь в колонку Type. Перемещаться по колонкам перечня полей можно клавишами Tab, Shfft+Tab, Enter, а также клавишами управления курсором или мышью. При этом Paradox автоматически пропускает ненужные колонки.
3. Нажмите Spacebar на клавиатуре или щелкните правой клавишей мыши, находясь в колонке Type, и вам будет предоставлен список возможных типов полей. Введите символ желаемого типа поля. Типы полей приведены ниже в данной главе.
4. Переместитесь в колонку Size и введите желаемый размер поля (если это необходимо). Информация о размерах полей приведена ниже в данной главе.
5. Переместитесь на вторую строку перечня полей и повторите проделанные операции для остальных полей таблицы.
6. Нажмите мышью кнопку Save As для сохранения таблицы и выбора ее имени.

Определение полей

Перечень полей (Field Roster) служит для определения полей новой таблицу. Перемещаться по колонкам перечня полей можно клавишами Tab, Shift+Tab, Enter, а также клавишами управления курсором или мышью. При этом в строке системных сообщений, находящемся в нижней части диалогового окна, показывается список значений, допустимых для ввода в текущую позицию перечня полей. Если в перечень введено полей больше, чем может одновременно быть показано на экране, то справа от перечня появляется линейка вертикальной прокрутки.

Имена полей

Имена полей вводятся в колонку Field Name перечня полей. Ниже приведены требования к именам полей.

- Максимальная длина имени поля составляет 25 символов, но с учетом использования различных систем, лучше использовать 8 символов.
- Имя не должно начинаться с символа пробела, но может его содержать.
- Каждое поле в таблице должно иметь уникальное имя. Т.е. нельзя сделать имя уни-кальным:
    - добавлением пробелов в конце имени
    - изменением верхнего и нижнего регистров букв
- Имя поля не должно содержать следующие символы:
    - квадратные [ ], фигурные {} и круглые ( ) скобки
    - комбинацию ->
    - один символ # (допустимы его сочетания с другими символами, на-пример, Phone #)

Типы и размеры полей Paradox-таблиц

Тип поля определяет вид хранимой в нем информации.Для определения типа поля переместитесь в колонку Type и воспользуйтесь одним из следующих методов, описанных выше. Для каждого типа таблиц будет показан только тот перечень, который в ней используется.

В следующей таблице перечислены символы, обозначающие типы полей и условия, накладываемые на их размеры для Paradox – таблиц.

Таблица 3.1 Типы полей Paradox - таблиц

Тип поля	Символ	Размер
Alphanumeric (алфавитно -цифровое)	A	1-255 (указывается обязательно)
Number (числовое)	N	нет
Short number (короткое число)	S	нет
Currency (денежное поле)	$	нет
Date (дата)	D	нет
Memo (memo - поле)	M	1-240 (обязательно)
Formatted memo (форматированное memo)	F	1-240 (необязательно)
Graphic (графическое)	G	0-240 (необязательно)
OLE	0	0-240 (необязательно)
Binary (бинарное поле)	B	0-240 (необязательно)

Memo-поля, форматированные memo-поля, графические, OLE и бинарные поля считаются BLOB-полями (binary large object fields). В таблицах формата Paradox 3.5 BLOB-поля отсутствуют.

Замечание о memo-полях

Memo и форматированные memo-поля могут быть практически любой длины. Размер поля, который вы указываете в диалоговом окне Create Table, определяет часть поля, хранимую непосредственно в таблице (от 1 до 240 символов). Само memo-поле хранится вне таблицы в специальном файле с расширением .MB и извлекается из него по мере необходимости при перемещении по записям таблицы.

Если все ваши memo - поля короче определённой величины (например, 200 символов), вы можете сэкономить место на жестком диске и сократить время доступа к memo - полю, если укажете его длину меньше или равной данной величине. В этом случае .MB файл все равно будет создан, но обращение к нему производится не будет.

Вставка полей и удаление полей

Чтобы вставить поле между двумя уже существующими в перечне, переместитесь к полю, которое должно оказаться под новым полем, и нажмите клавишу Ins. Paradox откроет чистую строку над текущим полем.

Чтобы удалить поле из перечня полей, выберите его и нажмите Ctrl+Del. Paradox удалит из перечня текущую строку.

Помните, что все пустые строки необходимо удалить из перечня полей до сохранения структуры таблицы.

Изменение порядка следования полей

Установить курсор на крайнюю левую колонку. Нажать левую кнопку мыши (в результате строка выделится двумя полосами). Не отпуская кнопку, переместить поле в нужное место. Четкое соответствие полей необходимо только в случае сложения дух таблиц. Для других случаев (форма, отчет, запрос…) порядок полей роли не играет.

Ключевые поля Paradox-таблиц

Этот раздел описывает ключевые поля таблиц и относится только к Paradox - таблицам. Ключевые поля определяют первичный индекс и порядок сортировки записей в таблице. Ключевое поле подразумевает также, что находящееся в нем значение должно быть уникальным.

Ключевые поля необходимы при связывании таблиц и организации системы ссылок между таблицами.

Определение ключевых полей

При определении ключевых полей помните о следующем :

- Таблица может иметь только один ключ, состоящий из одного или нескольких полей.
- Ключевые поля таблицы должны быть первыми в перечне полей.
- Если вы определяете несколько полей как ключевые, то создается составной ключ. Набор значений в этих полях должен быть уникальным для каждой записи таблицы. Составной ключ должен начинаться с первого поля перечня полей.

Вы в любой, момент можете переместить поле в другую позицию перечня по-лей для получения иного порядка полей в таблице (см. «Изменение порядка следования полей» ниже в этой главе).

Чтобы определить поле как ключевое, переместитесь в колонку Key перечня полей и произведите двойной щелчок мышью или нажмите любую клавишу. В данной позиции перечня появится индикатор ключа (*). Paradox проиндексирует таблицу по данному ключевому полю.

Удаление ключей

Чтобы удалить ключ с одного или нескольких полей, переместитесь в колонку Key перечня полей и двойным щелчком мыши или нажатием любой клавиши удалите индикатор ключа .

Если вы удалили ключ с поля, которое расположено в списке выше других ключевых полей, вам необходимо будет перегруппировать поля в перечне так, чтобы он начинался с ключевых полей.

Заимствование готовой структуры таблицы

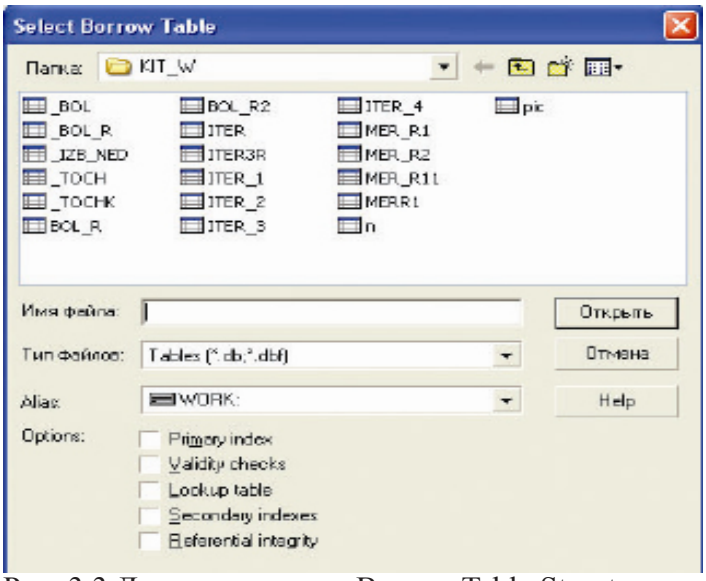


Рис. 3.3 Диалоговое окно Borrow Table Structure

Иногда необходимо создать таблицу со структурой подобной или идентичной уже имеющейся таблицы. Вы также можете воспользоваться структурой готовой таблицы, чтобы модифицировать ее по своему усмотрению:

1. Выберите кнопку Borrow в диалоговом окне Create Table. На экране появится диалоговое окно Borrow Table Structure (рис.3.3).
2. Укажите таблицу, структуру которой вы хотите заимствовать.
3. Отметьте при необходимости опции, которые желательно заимствовать вместе со структурой таблицы.
4. Нажмите мышью кнопку ОК для возврата в окно Create Table. Структура заимствованной таблицы появится в списке полей.

Кнопка Borrow диалогового окна Create Table доступна только при пустом перечне полей.

Опции заимствования

Кроме самой структуры таблицы вы можете заимствовать ее первичный и вторичные индексы, контроль на допустимые значения полей, таблицы-справочники, систему ссылок на другие таблицы. Отметьте на панели Options все, что вы желаете заимствовать у данной таблицы.

Учтите, если вы заимствуете ключ таблицы (опция Primary Index), вы должны быть уверены, что это ключевое поле будет первым в перечне полей новой таблицы.

Редактирование имени поля

Вы можете изменить ранее введённое имя поля. Двойным щелчком мыши или клавишей F2 введите текстовый курсор внутрь поля. Используйте клавишу Backspace для удаления символа слева от курсора или Del для удаления символа справа от курсора. Чтобы полностью изменить имя поля, следует переместиться к нему так, чтобы оно было выделено цветом, и просто начать вводить символы нового имени.

Контроль корректности данных

Контроль значений (validity checks)- это условия, которым должны удовлетворять вводимые в поля данные. Данный раздел, описывающий контроль значений, относится только к Paradox-таблицам.

Таблица 3.2 Типы контроля корректности данных Paradox

Тип контроля	Значение
Required field	Это поле каждой записи таблицы должно содержать непустое значение
Minimum	Значения, вводимые в данное поле, должны быть не меньше указываемой вами величины
Maximum	Значения, вводимые в данное поле, должны быть не больше указываемой вами величины
Default	Указываемая вами величина автоматически заносится в поле. Вы можете заменить ее вручную на другое значение.
Picture	Вы определяете строку символов, которая работает как шаблон для вводимых данных. Данные автоматически форматируются в соответствии с шаблоном.

Задание допустимых значений

Чтобы установить контроль значений для поля, сначала выберите его в перечне. Данное поле должно иметь уже определенные имя, тип и (при необходимости) размер. Выберите пункт Validity Checks, находящийся в списке Table Properties (рис. 3.4). В окне Create Table появятся все доступные типы контроля значений. Введите значения, необходимые для того или иного типа контроля значений данного поля. Вы можете составить любую комбинацию типов контроля значений.

Пример. Задание контроля значений типа Default.
Предположим, вы хотите, чтобы поле State/Prov таблицы Customer имело значение по умолчанию CA.

1. Выберите пункт Validity Checks в списке Table Properties.
2. Выберите поле State/Prov в перечне полей.
3. Введите CA в окошко Default.

Когда вы вставите в таблицу новую запись, Paradox автоматически введет в поле State / Prov значение CA. (При необходимости вы можете переместиться к этому полю и отредактировать его.)

Когда вы сохраняете структуру таблицы, заданные вами правила контроля значений полей Paradox записывает в файл с расширением .VAL .

Просмотр правил контроля значений

Paradox показывает все правила контроля значений, относящиеся к выде-ленному в данный момент в перечне полю. Вы можете просмотреть также правила контроля значений полей данной таблицы в диалоговом окне Structure Information. Для этого выберите пункт меню Tools / Utilities / Info Structure меню Desktop, либо проинспектируйте иконку таблицы в окне Browser или Folder и выберите пункт Info Structure, либо войдите в пункт Table / Info Structure окна таблицы.

Удаление контроля значений

Удалить контроль значений полей можно, находясь либо в окне Create Table либо в окне Restructure Table (окно Restructure Table описывается ниже в данной главе). Удаление контроля значений никак не изменяет уже находящиеся в таблице данные. Оно просто снимает ограничения на значения, которые будут вводится в дальнейшем. .

Пример. Удаление контроля значений.
Чтобы удалить контроль значений поля, следует :

1. Выбрать это поле в списке полей.
2. Удалить все величины из текстовых окошек, задающих контроль значений. (Чтобы убрать контроль типа Required Field, следует просто убрать отметку из его окошка.).

Поля, обязательные для заполнения (Required Fields)

В данное поле должны быть введены данные, прежде чем вы сможете перейти к другой записи. Чтобы сделать поле обязательным для заполнения, выберите его в списке полей и установите опцию Required Field.

Пример. Установка контроля типа Required Field.
Если вы хотите, чтобы поле Customer No таблицы Customer было бязательным для заполнения:

1. Выберите поле Customer No в списке.
2. Установите опцию Required Field. Теперь в таблице Customer есть обязательное для заполнения поле, и если вы не введете в него данные, Paradox проинформирует вас о том, что в поле находится недопустимое значение и не позволит покинуть текущую запись или выйти из режима редактирования, пока в поле Customer No не будет введено значение.

Вы можете сделать обязательными для заполнения поле любого типа, включая BLOB-поля

Контроль на минимальные и максимальные значения (Minimum and maximum values)

Контроль на минимум определяет минимально допустимое значение поля, контроль на максимум соответственно максимально допустимое значение.

Пример. Установка контроля на минимум и максимум.
Предположим, вам нужно, чтобы минимальное значение поля Qty таблицы Lineitem равнялось 1, а максимальное 1000.

1. Войдите в пункт Validity Checks из списка Table Properties.
2. Выберите поле Qty в списке полей.
3. Введите число 1 в текстовое окошко Minimum.
4. Введите число 1000 в текстовое окошко Maximum.

Теперь, когда вы будете вводить данные в поле Qty таблицы Lineitem, Paradox не примет значения, не входящие в диапазон от 1 до 1000.

Когда вы определяете минимум или максимум числового поля, их значения следует вводить в текущем числовом формате, который устанавливается с помощью Control Panel системы Windows. В случае поля типа даты можно использовать любой формат. Контроль на минимум и максимум можно применять к алфавитно-цифровым, числовым, денежным и полям типа дата.

Значения по умолчанию (Default values)

Paradox автоматически вводит определенное вами значение в нужное поле при вставке в таблицу новой записи. Например, если большинство ваших клиен-тов находится в Англии, вы можете определить символы UK как значение по умолчанию для поля Country таблицы Customer. Когда вы вставляете в таблицу новую строку, она появляется уже со значением UK в поле Country.

Значение, введённое по умолчанию, можно изменить, переместившись в данное поле и введя другую величину. Вы можете удалить значение по умолчанию и оставить это поле пустым, если оно не является обязательным для заполнения. Когда вы определяете значение по умолчанию для числового поля, его следует вводить в текущем числовом формате, который устанавливается с помощью Control Panel системы Windows. Значения по умолчанию можно присваивать только алфавитно-цифровым, числовым, денежным и полям типа дата.

Шаблоны (Picture patterns)

Шаблон задает формат вводимых в поле данных. Например, если вы зададите шаблон (###)###-#### (типичный вид телефонных номеров в США) и введете значение 4099551234, Paradox преобразует его к виду (409)955-1234.

В таблице 3.3 приведены символы, которые можно использовать в шаблоне, и их значения.

Если вы в шаблоне используете любой символ, отличный от перечисленных в таблице, Paradox будет считать его константой. Когда вы вводите данные в поле с шаблоном и доходите до константы, Paradox вводит ее автоматически. Например, если вы задали шаблон (409)###-#### , а затем ввели 9551234, то Paradox поместит в это поле значение (409)955-1234.

Чтобы задать шаблон, просто введите его в текстовое окошко Picture либо нажмите мышью кнопку Assist. В результате откроется окно, показанное на рис.3.4

Таблица 3.3 Символы, использующиеся в шаблонах.

Символ	Значение
#	Любая цифра
?	Любая буква
&	Любая буква (преобразуется в верхний регистр)
@	Любой символ
I	Любой символ (преобразуется в верхний регистр)
!	Следующий символ является литералом, а не специальным символом шаблона
*	Следующий символ может повторяться любое количество раз, либо определяет требуемое количество повторений
[ ]	Символы внутри скобок необязательны
{}	Группирует символы внутри скобок
,	Альтернативные значения

Получение информации о шаблоне

Когда вы нажмете кнопку Assist окна Create Table или Restructure Table, на экране появится диалоговое окно Picture Assistant (рис. 3.4).

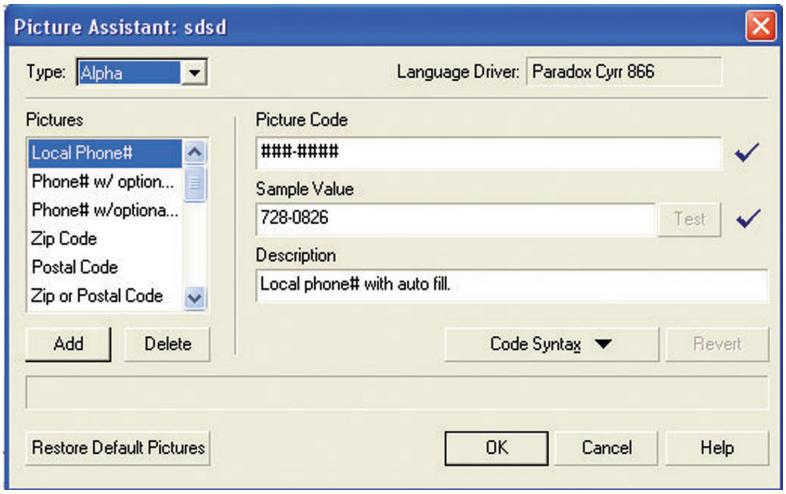


Рис.3.4 создание шаблона

В этом окне вы можете :

- Ввести желаемый шаблон в текстовое окошко Picture
- Нажать кнопку Code Syntax для тестирования шаблона, находящегося в окошке Picture
- Нажать Restore Default, чтобы отменить изменения, сделанные вами и вернутся к исходным значением окошка Picture
- Ввести какое-либо значение в текстовое окошко Sample Value и нажать кнопку Test, чтобы убедиться в том, что ваш шаблон работает правильно
- Щелкнуть мышью над указателем раскрывающегося списка Sample Pictures для получения списка примеров шаблонов. Над окошком Sample Pictures Paradox приводит объяснения, касающиеся каждого шаблона из списка примеров.
- Нажать кнопку Add To List, чтобы поместить содержимое окошка Picture в список примеров шаблонов Sample Pictures.
- Нажать кнопку Delete From List для удаления какого-либо шаблона из списка примеров.

Все перечисленные вспомогательные действия вы можете производить как с разработанными вами шаблонами, так и с примерами шаблонов, предоставляемыми вам системой Paradox.

Пример . Ввод собственного шаблона.

Чтобы ввести собственный шаблон, следует :

1. Ввести его в текстовое окошко Picture, используя специальные символы. Напри-мер, для ввода даты можно ввести ##.##.## (две решетки точка две решетки точка две решетки). В этом случае точки будут ставиться автоматически при наборе даты.
2. Выбрать пункт Code / Syntax, чтобы убедиться в том, что Paradox способен интерпретировать содержимое окошка Picture как шаблон.
3. Если шаблон задан правильно, в нижней части окна Picture Assistance появится соответствующее сообщение The picture is correct.
4. Нажмите Add to List, чтобы сохранить шаблон в списке. OK, чтобы сохранить шаблон и закрыть диалоговое окно. При создании нового шаблона ему надо будет дать имя.Пример. Использование готового примера шаблона.

Пример. Использование готового примера шаблона.

Paradox предоставляет несколько стандартных шаблонов, которые можно выбрать в диалоговом окне Picture Assistance. Для выбора одного из них следует:

1. Выбрать шаблон из раскрывающегося списка примеров окошка Sample Pictures. При этом в специальной области над окошком будет приведено краткое объяснение данного шаблона. Например, если вы выберите шаблон 5#[-4#], то увидите сообщение о том, что этот шаблон служит для 5-ти или 9-ти цифрового почтового zip - кода США.
2. При необходимости, вы можете изменить стандартный шаблон, находящийся в окошке Picture. Если вы сделали ошибку, нажмите кнопку Restore Original, и вы получите исходный шаблон, который вы скопировали из списка примеров.
3. Когда нужный вам шаблон готов и находится в окошке Picture, нажмите ОК.

Учтите, если вы разработаете шаблон в процессе изменения структуры таблицы, уже содержащей информацию, Paradox не изменит находящиеся в таблице данные в соответствие с шаблоном.

Задание таблицы-справочника

Данный раздел, описывающий возможности использования таблиц-справочников (lookup table), относится только к Paradox-таблицам.

Использование таблицы-справочника означает, что вы обязаны вводить в вашу таблицу только те данные, которые уже содержатся в другой таблице - таблице-справочнике. «Присоединение» таблицы-справочника к какому-либо полю приводит к следующему:

- Заставляет вас вводить те значения, которые уже существуют в первом поле таблицы-справочника.
- Позволяет найти и автоматически скопировать данные из таблицы-справочника в вашу таблицу.

Таблицы-справочники используются, в основном, при вводе данных. В отличие от системы ссылок (referential integrity), этот режим не отслеживает и не контролирует изменения, которые вы вносите в таблицу-справочник. Использование справочника обеспечивает безошибочное копирование из одной таблицы в другую, в то время как система ссылок сохраняет неразрывность связей между данными в различных таблицах. Система ссылок будет рассмотрена ниже в этой главе.

Основное преимущество использования таблиц-справочников - возмож-ность автоматически вводить правильные данные в вашу таблицу.

Таблицы-справочники полезны также при использовании Paradox-таблиц версии 3.5, т.к. системы ссылок для этих таблиц не поддерживаются.

Помните о следующих правилах при определении таблицы-справочника:

- Таблица-справочник содержит данные, которые вы собираетесь скопировать в другую таблицу.
- Данные из таблицы-справочника, которыми вы собираетесь воспользоваться, должны находиться в ее первом поле.
- Наличие ключа у таблицы-справочника поможет ускорить доступ к ней (См. «Ключевые поля Paradox-таблиц» выше в этой главе).

Пример. Задание таблицы-справочника

Чтобы указать таблицу-справочник для поля:

1. Выберите пункт Table Lookup из списка Table Properties в диалоговом окне Creat Table (или Restructure Table). При этом станет доступной кнопка Define и ниже будут приведены все доступные таблицы - справочники.
2. Нажмите Table Lookup.
3. Выберите таблицу-справочник из списка (в нем присутствуют все таблицы из рабочего каталога). При этом в рамке-Lookup Field появится первое поле выбранной вами таблицы.
4. Из списка Fields выберите поле, которое будет обращаться к справочник. Paradox поместит его в рамку Field Name.
5. Задайте желаемые режимы использования таблице - справочника.
6. Нажмите ОК для выхода из диалогового окна Table Lookup. Ниже кнопки Define диалогового окна Create Table (или Restructure Table) появится имя таблицы - справочник

Вы можете использовать таблицы-справочники, находящиеся в разных каталогах. Используйте для этого раскрывающееся меню Path или кнопку Browse.

Режимы использования таблицы-справочника

Paradox предлагает два способа работы с таблицей-справочником:

- Just Current Field: Значение текущего поля - единственная величина из таблицы-справочника, которую Paradox будет проверять или вносить в данную таблицу.
- All Corresponding Fields: Paradox проверяет поле, для которого задана таблица-справочник, и заполняет все поля данными из соответствующих полей таблицы-сравнения. Соответствующие поля обеих таблиц должны иметь одинаковые имена и совместимые типы.

Paradox дает два варианта доступа к таблице-справочнику. Они определяют возможность просмотра таблицы-справочника во время редактирования.

- Help and Fill: вы можете видеть таблицу-справочник из редактируемой таблицы.
- Fill No Help: Вы не можете видеть таблицу-справочник.

Если выбрано Fill No Help, вы не можете автоматически открыть таблицу-справочник. Однако, она может быть доступна из ее собственного табличного окна.

Определение вторичных индексов

Вы можете присвоить полю или группе полей вторичные индексы, чтобы:

- Производить быстрый поиск значений в определённых полях
- Иметь возможность другого порядка просмотра таблицы
- Связывать таблицы

Чтобы просмотреть записи в таблице, имеющей ключ, в другом порядке, вам необходимо использовать вторичный индекс. Только таким образом вы можете временно скрыть физический порядок записей заданный ключом таблицы.

Примером использования вторичного индекса может служить задача связывания таблиц Customer и Orders таким образом, чтобы были видны заказы каждого клиента. Таблица Orders имеет вторичный индекс, связанный с полем Customer No. Это означает, что Paradox может быстро найти все записи с данным значением Customer No. Когда вы связываете таблицы, Paradox для каждого значения Customer No в таблице Customer находит и показывает все совпадающие значения Customer No в таблице Orders. Используя такую связь, вы можете создать форму, содержащую все заказы, сделанные каждым клиентом.

Таблица может иметь несколько вторичных индексов. Вы можете также создавать составные вторичные индексы, объединяя два и более полей. Количество сложных вторичных индексов может достигать 16, а максимальное количество простых вторичных индексов равно количеству полей в таблице.

Помните, что нельзя создать вторичный индекс по BLOB-полям (memo, форматированное memo, двоичное, OLE или графическое).

Чтобы создать вторичный индекс для таблицы:

1. Выберите пункт Secondary Index (рис. 3.2) из списка Table Properties в диалоговом окне Create Table (или Restructure Table). Кнопка Define становится доступной, и появляется список всех существующих вторичных индексов.
2. Нажмите кнопку Define для входа в диалоговое окно Define Secondary Index. (рис. 3.5). Меню Fields показывает поля, которые можно использовать для создания вторичных индексов (BLOB-поля недоступны).
3. Двойным щелчком отметьте поле, по которому вы хотите создать вторичный индекс (или выделите его, а затем нажмите стрелку Add Field или нажмите Alt+A. Paradox поместит это поле в список Indexed Fields.
4. Выберите опции по желанию. Они будут описаны в этой главе ниже.
5. Нажмите OK для создания вторичного индекса и закройте диалоговое окно.

Paradox автоматически присваивает индексам имена полей, по которым они созданы, и помещает их в список ниже кнопки Define в диалоговом окне Create Table (или Restructure Table). Для создания другого вторичного индекса снова нажмите кнопку Define.

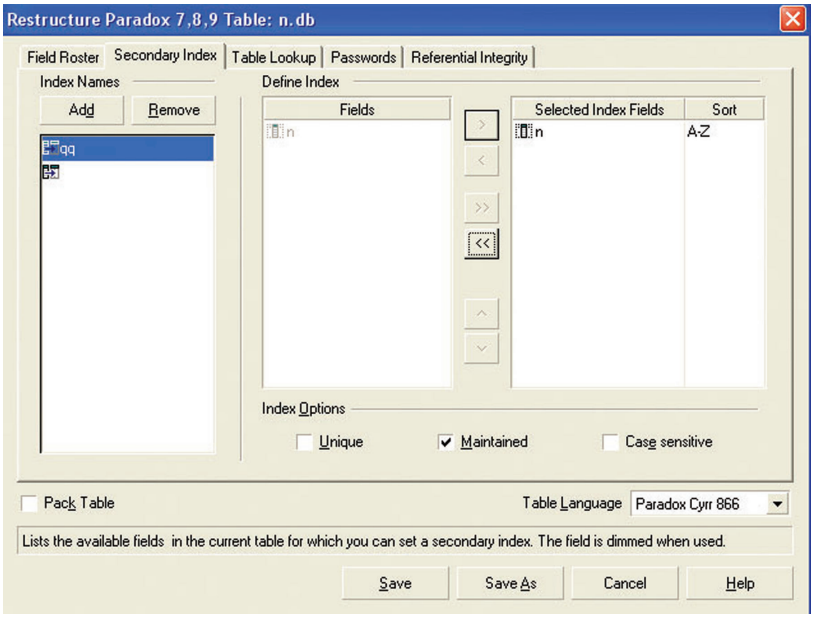


Рис.3.5 Создание вторичных индексов

Опция Case Sensitive

Если вы установите опцию Case Sensitive в диалоговом окне Define Secondary Index, то Paradox будет различать строчные и прописные буквы при сортировке слов.

Учтите, что драйверы некоторых языков могут быть нечувствительны к данной опции.

Пример. Влияние опции Case Sensitive на сортировку данных

Case Sensitive	Case Insensitive
Abed	aaa
aBcd	Abed *
aaaa	aBcd *

* Значения Abed и aBcd неразличимы в случае case-insensitive. Они появляются в том порядке, в котором были введены в таблицу.

Наличие букв, написанных в верхнем регистре, никак не проявляется в индексе, для которого опция Case Sensitive выключена.

Если опция включена и индекс создан на основе какого-то одного поля, Paradox сам присвоит индексу имя этого поля. В остальных случаях вы лично должны присваивать имена индексам при сохранении. Это позволяет создать два индекса на основе одного поля - чувствительный и не чувствительный к регистру.

Составные вторичные индексы (composite secondary indexes)

Этот раздел применим только к таблицам Paradox for Windows. Вы можете создать составной вторичный индекс внесением в список Indexed Fields более од-ного поля. Откройте диалоговое окно Define Secondary Index и добавьте в список Indexed Fields поля, которые будут использоваться для индексирования. Для этого двойным щелчком отметьте поле (или выделите его в списке Fields, а затем выберите стрелку Add Field или нажмите Alt+A). Paradox поместит это поле под текущим полем в списке Indexed Fields.

Paradox создает сложный индекс в порядке появления полей в списке Indexed Fields. При использовании такого индекса Paradox сортирует таблицу по полям последовательно, начиная с верхнего поля. Чтобы измененить порядок сортировки записей по данному индексу, вы должны изменить порядок полей в списке . Для перемещения поля в списке Indexed Fields выберите это поле и поместите его в нужное место с помощью кнопок-стрелок Change Orders. Эти стрелки доступны, когда в упомянутом списке имеется более одного поля.

Для удаления поля из списка Indexed Fields выделите его и нажмите мышью кнопку-стрелку Remove Field (или клавиши Alt+R). Для удаления всех полей из списка щелкните кнопку Clear All (или нажмите клавиши Alt+L).

Вы должны самостоятельно присвоить имена сложным вторичным индексам. Для этого нажмите OK, после этого вы увидите диалоговое окно Save Index As. Если индекс создан на основе одного поля и его опция Case Sensitive включена, Paradox автоматически присвоит ему имя этого поля. В остальных случаях впечатайте какое-либо имя в текстовое окошко Index Name. Имя не должно содержать более 25 символов. Символы должны быть печатными. После нажатия кнопки OK в диалоговом окне Save Index As, оно, как и окно Define Secondary Index, закро-ется, а имя индекса появится в диалоговом окне Create Table (или Restructure Table).

Paradox предупредит вас о возможной перезаписи индекса, если присвоенное вами имя уже используется.

Изменение вторичных индексов

Для внесения изменений во вторичный индекс выберите его из списка вторичных индексов в диалоговом окне Create Table (или Restructure Table) и нажмите кнопку Modify. Откроется окно Define Secondary Index со всеми опциями индекса. Внесите желаемые изменения и нажмите ОК.

Если вы хотите удалить вторичный индекс, выберите его имя из списка вторичных индексов в диалоговом окне Create Table (или Restructure Table) и нажмите Erase. Paradox удалит выбранный индекс.

Определение системы ссылок между таблицами

Этот раздел применим только к Paradox-таблицам. Система ссылок подразумевает соответствие поля или группы полей одной таблицы («дочерней») к ключу другой таблицы («материнской»). Для определённых полей дочерней таблицы Paradox считает верными только те значения, которые существуют в ключевых полях материнской таблицы.

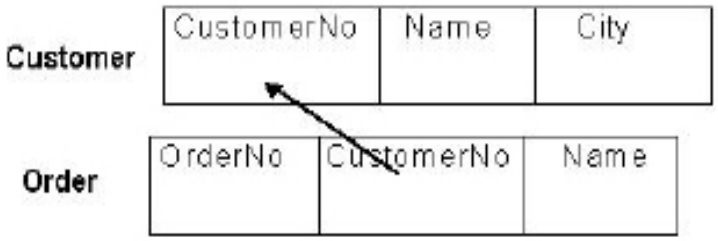


Рис. 3.6 Пример системы ссылок

Paradox запрещает вам вводить значение поля CustomerNo в таблице Orders, которое не соответствует какому-либо существующему значению этого поля в таблице Customer.

Вы можете задать систему ссылок только между однотипными полями (рис. 3.6). Например, вы можете определить систему ссылок между таблицами Customer и Orders по полю Customer No. Аналогично, вы можете задать систему ссылок между таблицами Orders и Lineitem по полю Order No. В обоих случаях значения в соответствующих полях связанных между собой записей одни и те же (имена полей не имеют значения, если их типы и размеры идентичны.).

Помните, что вы можете установить систему ссылок только между таблицами одного каталога При использовании системы ссылок Paradox проверяет правильность значения перед занесением его в таблицу. Если вы задали систему ссылок между Customer и Orders по полю Customer No, а затем ввели значение в это поле в таблице Orders, Paradox найдет это поле в таблице Customer и:

- Занесет новое значение в таблицу Orders, если оно существует в таблице Customer.
- Отвергнет как недопустимое, если его нет в таблице Customer.

Обратите внимание, что если система ссылок определена для таблицы уже содержащей данные, Paradox поместит записи, не отвечающие вышеупомянутому требованию во временную таблицу Keyviol (ошибка) в вашем личном каталоге.

Чтобы задать систему ссылок сделайте следующее:

1. Выберите пункт Referential Integrity из списка Table Properties в диалоговом окне Create TaЬIe или Restructure TaЬIe. При этом станет доступна клавиша Define.
2. Нажмите Define для входа в диалоговое окно Referential Integrity.
3. Выберите материнскую таблицу в списке TaЬIe. В рамке Parent's Key на диаграмме системы ссылок появится поле ключа таблицы. Помните, что, если материнская таблица не имеет ключа или с ним имеются некоторые сложности, в поле сообщений появится соответствующее предупреждение. Список Fields содержит все поля дочерней таблицы. (BLOB-поля в этом списке неактивны. По этим полям вы не можете определить систему ссылок.)
4. Двойным щелчком выберите поле дочерней таблицы в списке Fields (или выберите его клавишей Tab и нажмите кнопку - стрелку Add Field. Имя поля появится в рамке Child Fields на диаграмме системы ссылок. Если вы выберите поле с типом, не идентичным типу поля ключа материнской таблицы, Paradox выдаст предупреждение и не поместит поле на диаграмму. Если вы ошиблись и добавили неверное поле, нажмите кнопку-стрелку Remove Field или ALT+R.
5. Выберите желаемый способ обновления. (См. «Варианты способов обновления» в следующем разделе.)
6. Определите, будете ли вы задавать строгую систему ссылок. (См. «Использование строгой системы

ссылок» ниже в этом разделе.)

Нажмите OK для задания имени и сохранения системы ссылок. Вы можете установить систему ссылок со сложным ключом.

Если материнская таблица имеет сложный ключ, добавьте поля из списка Fields до соответствия одному, нескольким или всем полям этого ключа.

Варианты способов обновления

Paradox предлагает два способа обновления таблиц, использующих систему ссылок. Вы должны задать один из них при определении системы ссылок.

- Cascade: Все изменения значения ключа материнской таблицы автоматически происходят и в дочерней таблице. Этот способ используется по умолчанию. Помните, что при этом Paradox блокирует как материнскую, так и все ее дочерние записи. Если блокировка отвергается (уже заблокировано другим пользователем), Paradox не в состоянии произвести обновление.
- Prohibit Вы не можете изменить значение ключа материнской таблицы, если в дочерней таблице имеются записи, соответствующие этому значению. Например, если в поле Customer No таблицы Orders имеется значение 350, Paradox запретит вам менять это значение поля в таблице Customer (вы можете изменить его в Customer только при условии, что сначала были удалены или изменены все записи в Orders, содержащие это значение). Тем не менее, если ни в одной записи дочерней таблицы это значение не встречается, Paradox разрешает изменения в материнской таблице.

Использование строгой системы ссылок

Paradox for Windows - первая версия Pa/adox, полностью использующая воз-можности системы ссылок. Выбор опции Strict Referential Integrity (строгая система ссылок) позволяют вам контролировать работу ранних версий Paradox с таблицами, для которых определена система ссылок. Предположим, вы используете более раннюю версию Paradox для доступа к таблице Paradox for Windows, ко-торая использует систему ссылок. Вы можете внести данные, нарушающие взаимосвязь таблиц, т.к. ваша версия Paradox не поддерживает систему ссылок. Во избежание такой ситуации включите опцию Strict Referential Integrity диалогового окна Referential Integrity.

Сохранение системы ссылок

Когда вы зададите систему ссылок, нажмите OK. Вы увидите диалоговое окно Save Referential Integrity As

Введите имя, которое вы хотите присвоить системе ссылок. Имя может иметь до 31 печатного символа и не требует расширения. После нажатия кнопки OK окно закрывается, а имя появляется в списке ниже клавиши Define в диалоговом окне Create Table (или Restructure Table).

Помните, что при сохранении Paradox - таблицы параметры системы ссылок записываются в файле с именем таблицы и расширением .VAL .

Когда вы сохраняете систему ссылок, Paradox проверяет, определён ли индекс по полям системы ссылок. Если нет, то Paradox сам создает его, присваивая ему имя поля (если область была определена на основе одного поля) или же имя, которое вы дали системе ссылок (если она была задана для нескольких полей). Индекс появится в списке вторичных индексов, когда вы выберете Secondary Indexes из меню Table Properties в диалоговом окне Create Table (или Restructure Table). Когда вы удаляете систему ссылок, Paradox не удаляет автоматически индекс. Вам надо самим позаботиться об этом.

Изменение или удаление системы ссылок

Выберите имя любой системы ссылок из списка в диалоговом окне Create Table (или Restructure Table), если необходимо ее измененить или удалить. Затем:

- Нажмите Modify для входа в диалоговое окно Referential Integrity, чтобы отредактировать выбранную систему ссылок. При этом Paradox должен блокировать все таблицы, входящие в нее. Вы можете изменить:
    - Имя системы ссылок (сохраните ее под другим названием)
    - Способ обновления

- Установку строгой системы ссылок
- Нажмите Erase для удаления выделенной системы ссылок.

Создание самоссылающейся системы ссылок

Вы можете создать систему ссылок для таблицы так, что одно поле будет ссылаться на поле ключа этой же таблицы. Например, вы имеете таблицу со списком служащих. Ключевое поле для нее - Employee ID (идентификатор служащего). Имеется также поле руководители Supervisor. Но руководители - тоже служащие. Вы можете создать такую систему ссылок, чтобы данные, вводимые в поле Supervisor, записывались и в поле Employee ID. При создании самоссылающейся системы используйте способ обновления Prohibit.

Помните, что нельзя задавать круговых ссылок, то есть вы не можете создать систему ссылок с полем, ссылающимся на себя.

Установка пароля доступа к данным

Этот раздел относится только к Paradox-таблицам. Очень важно быть абсолютно уверенным, что созданные вами таблицы защищены от нежелательного доступа к ним. Вы можете не только задать пароль для таблицы в целом, но и определить права доступа к ней и ее отдельным полям.

Если вы установили парольную защиту, то доступ к вашей таблице получат только пользователи, знающие пароль, поэтому постарайтесь не забыть его.

При попытке пользователя открыть защищенную паролем таблицу, Paradox попросит его ввести пароль (если он до сих пор этого не сделал, т.к. пароль сохраняется на весь сеанс работы).

Чтобы установить пароль для таблицы:

1. Выберите опцию Password Security на панели Table Properties.При этом станет доступна кнопка Define.
2. Нажмите Define для входа в диалоговое окно Password Security
3. Введите пароль в текстовом окошке Master Password Вместо введённых ва-ми символов вы увидите звездочки (*). Пароль может содержать от 1 до 31 символа, в том числе пробелы.
4. Подтвердите пароль, введя его еще раз в текстовом окошке Verify Master Password. Вы вновь увидите лишь звездочки.
5. Если содержимое этих двух текстовых окошек окажется различным (регистр также принимается во внимание), то вы получите сообщение об ошибке и вам придётся вновь заполнить оба окошка. Нажмите OK для выхода из этого диалогового окна. Вы вновь окажетесь в диалоговом окне Create Table. Paradox сохраняет пароль, когда вы выходите из окна Password Security или Auxiliary Passwords.

Если вам нужна дополнительная защита, нажмите Auxiliary Passwords в диалоговом окне Password Security. Перед вами появится окно Auxiliary Passwords

Это диалоговое окно используется для задания определённых прав доступа к таблицам и полям по дополнительным паролям.

Имейте в виду, вы можете вернуться к диалоговому окну парольной защиты Password Security и изменить основной пароль доступа к таблицам, нажав кнопку Cancel или клавишу Esc. Если вы сделаете это, любые определённые вами ранее дополнительные пароли будут утеряны.

Основной пароль обеспечивает все права доступа на все таблицы и их поля. Используя дополнительные пароли, вы можете быть более избирательным при задании прав доступа для каждого пользователя.

Ниже представлены типы прав доступа:

- **All :** Дает пользователю все права доступа к любой табличной функции, включая возможность изменения структуры и удаления таблиц.
- **Insert & Delete :** Позволяет пользователю удалять записи или очищать таблицу, не удаляя её.
- **Data Entry :** Предоставляет пользователю возможность вставлять записи в таблицу, но налагается запрет на удаление записей, изменение структуры или очистку таблицы.
- **Update :** Дает право пользователю просматривать таблицу и изменять неключевые поля. Нельзя, однако, вставлять или удалять записи, изменять ключевые поля.
- **Read Only** : Пользователь может только осуществлять просмотр таблицы без каких-либо изменений в ней.

В дополнении к правам доступа, определённым одновременно для всех полей таблицы, вы можете определить права к отдельным полям. По умолчанию, право доступа в списке Field Rights установлено в All (все права).

Чтобы выбрать другую опцию, необходимо дважды щелкнуть на выбранном поле (или нажать кнопку Field Rights). Выполняя двойной щелчок кнопки мыши, вы каждый раз циклически переходите в следующие режимы: Read Only (только чтение), None (нет прав доступа) и опять к ALL (все права).

Пример. Установка дополнительных паролей

Для определения дополнительного пароля необходимо :

1. Набрать с клавиатуры пароль в текстовом окошке Current Password
2. Выбрать уровень парольной защиты из панели прав доступа к таблице

Таблица. 3.4 Права доступа к таблицам и полям

Права доступа к таблице	Права доступа к полям		
	ALL	Read Only	None
ALL	+		
Insert & Delete	+	+	+
Data Entry	+	+	+
Update	+	+	+
Read Only	+	+	+

3. Присвоить права доступа (All, Read Only, None) паролям.
4. Нажать кнопку Add для занесения пароля в список дополнительных паролей
5. Повторить весь процесс для определения всех требуемых дополнительных па-ролей
6. Нажать О К для сохранения дополнительных паролей и закрытия диалогового окна (это действие сохраняет также и основной пароль)

Для удаления пароля выберите его в списке паролей и нажмите кнопку Delete. Чтобы изменить права доступа по данному паролю, нажмите кнопку Change. Когда необходимые изменения будут сделаны, нажмите кнопку Accept для их сохранения или кнопку Revert для возврата в исходное состояние.

Выбор драйвера национального языка

Paradox поддерживает различные национальные языковые драйверы клавиатуры и экрана, в том числе и драйвер кириллицы.

Драйвер, поддерживающий язык таблицы, и набор кодов национальной клавиатуры определяет порядок сортировки в таблице и допустимый набор символов. Выбрать драйвер языка для Paradox и dBASE можно из утилиты конфигурации (IDAPY Configuration Utility в папке ПУСК / Программы/Paradox…). Вы можете заменить текущий язык таблицы при создании новой таблицы, выбрав Table Language из списка Table Properties в диалоговом окне Create Table. Затем, нажав кнопку Modify, выбрать нужный драйвер.

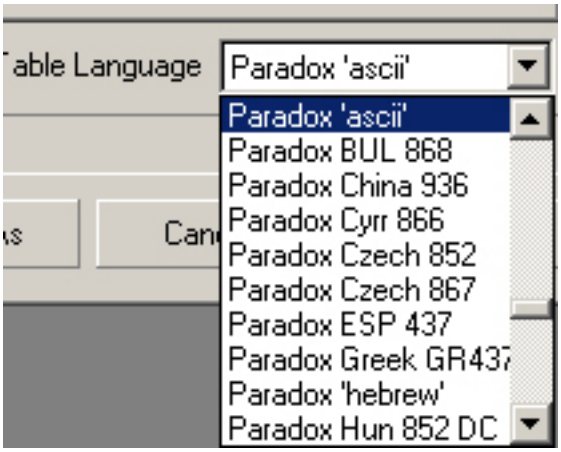


Рис.3.7 Изменение языкового драйвера при создании таблицы

Окно изменения языкового драйвера показано на рис.3,7 На рис.3.8 показано окно, когда вы меняете драйвер в поле открытой таблицы, нажав в поле, где стоит курсор Правая кн.мыши / Font / Typefase и выберете тип шрифта.

Для пользователей Borland Delphi 7 изменение языкового драйвера для отдельного поля в режиме Database Desktop в существующей таблице закрыто как и многие другие режимы. Однако, таблицы созданные и рабjтающие в Paradox полностью совместимы, т.к. они созданы на одной платформе BDE и вообще в Delphi вставлен просто кусок из Paradox Borland.

Поэтому, при работе с Delphi и обработке табличных данных удобнее и проще все разработки таблиц производить в Paradox а не в Database Desktop Delphi.

Часто при выполнении запросов, в работе с различными операционными системами или при другом изменении языкового драйвера операционной системы, некоторые надписи изменяют свой стиль. Отследить все изменения, которые производятся в операционной системе и учесть их в разработке практически невозможно. Исходя из этого, при написании программы желательно действовать по следующей схеме:

- Выбрать операционную систему со стандартными установками
- Установить параметры BDE
- При работе в программе для полей, в которых надо установить националь-ный драйвер, этот драйвер устанавливается
- При создании таблиц, которые поддерживает Paradox для других языков, после создания этой таблицы установить драйвер для всей таблицы (например, если вы создает отчеты для матричного принтера на Paradox for DOS) и потом устанавливать отдельно для полей.

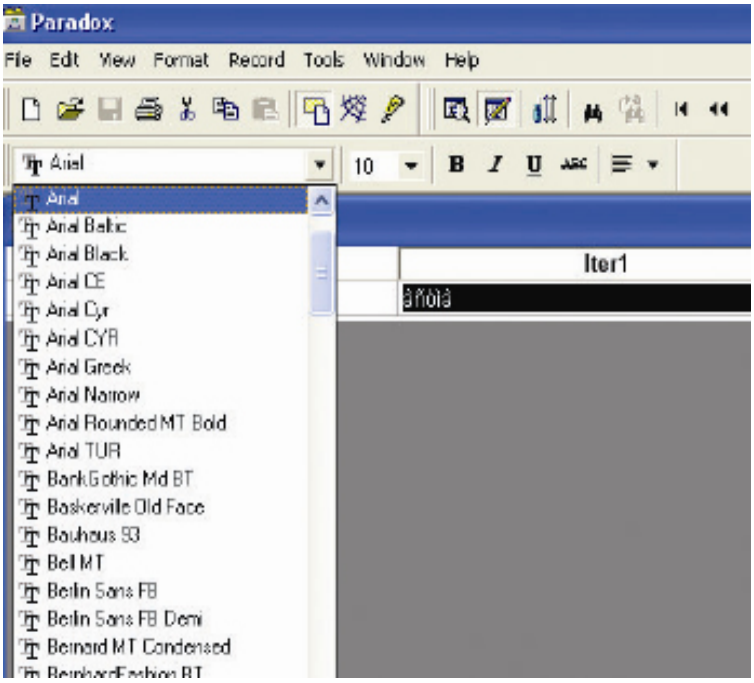


Рис.3.8 Изменение языкового драйвера при просмотре таблицы

Разработка dBASE-таблицы

Если вы выберете режим создания dBASE-таблицы из диалогового окна Table Type, на экране появляется диалоговое окно Create Table (рис. 3.9), в котором вам необходимо сделать следующее:

- Ввести имена полей таблицы
- Определить типы и размеры полей
- Создать индексы для полей или индексные выражения.

Следует отметить, что процесс создания dBASE - таблиц отличается от создания Paradox - таблиц, так как проверка корректности данных, система ссылок и таблицы-справочники не применимы к dBASE-таблицам.

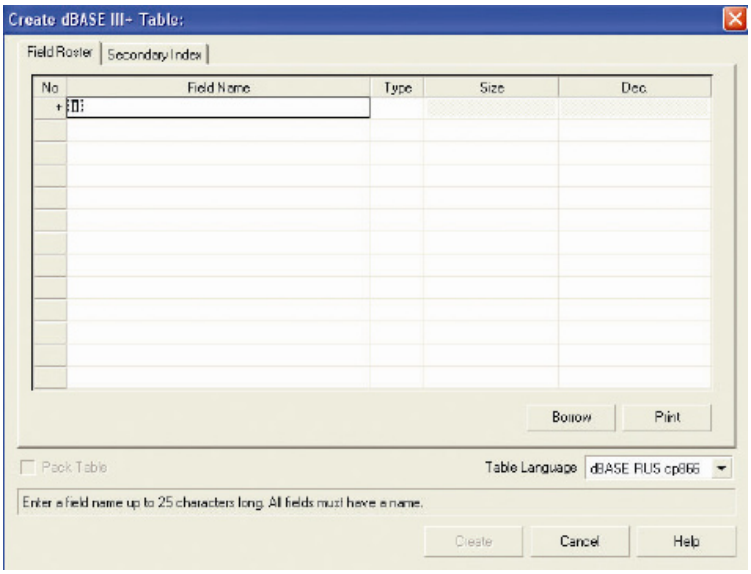


Рис. 3.9 Диалоговое окно Create dbase Table

Определение полей таблицы

Используйте панель Field Roster (перечень полей) в диалоговом окне Great Table для определения полей новой таблицы. Вы можете использовать мышь, клавиши управления курсором или Enter, Tab, Shift+Tab для перемещения между столбцами перечня. При перемещении окно системных сообщений поможет вам выбрать допустимые варианты ввода. Вертикальная линейка прокрутки появится в списке полей, если вы введете больше полей, чем помещается на экране.

Имена полей

Вводите имена полей в столбце Field Name панели Field Roster, придерживаясь следующих правил:

- Имена полей могут содержать в себе буквы (лат.), цифры и символы подчёркивания.
- Первым символом имени поля должна быть буква (желательно латинская).
- Знаки препинания (пунктуации), пробелы и другие специальные символы недопустимы.
- Каждое имя поля должно быть уникальным (вы не можете сделать имя уникальным путем добавления пробелов в конце имени).

Типы и размеры полей

Тип поля определяет класс данных, которые вы можете вводить в конкретное поле. Для присвоения dBASE-полю (рис. 3.14) какого-либо типа, поместите курсор мыши в столбец Type перечня полей и воспользуйтесь одним из трех вариантов:

- Введите с клавиатуры соответствующий типу поля символ (см. ниже в таблице).
- Проинспектируйте столбец перечня и в меню типов полей сделайте выбор нужного вам.
- Для активизации меню Type можно также нажать пробел на клавиатуре, а затем выбрать желаемый тип.

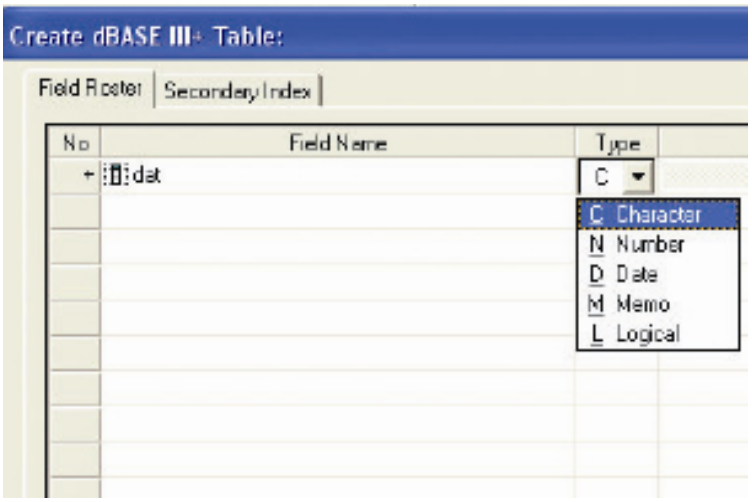


Рис. 3.10 Типы dBASE-полей

Поле Dec перечня полей используется для установки количества позиций для численных полей в диапазоне от нуля до заданной величины.

Формат полей должен соответствовать тому языку, для которого создается таблица.

Вставка полей и удаление полей

Для вставки поля между уже существующими полями в перечне, выберите поле, над которым вы хотите поместить вставляемое поле, и нажмите клавишу Ins. Paradox вставит пустую строку.

Чтобы удалить поле из перечня, выберите его и нажмите Ctrl+Del. Paradox удалит всю строку.

Заимствование структуры существующей dBASE-таблицы

Если вы хотите заимствовать уже существующую структуру dBASE-таблицы, щелкните кнопку Borrow (рис.3.13). Вы увидите диалоговое окно Borrow Table Strusture. Когда вы заимствуете dBASE-таблицу, доступной является только опция Indexes. Выберите таблицу, структуру которой вы хотите позаимствовать, и установите опцию Indexes, если вы желаете позаимствовать кроме структуры еще и табличные индексы. После нажатия кнопки Borrow Paradox закроет окно Borrow Table Structure и поместит заимствованную структуру в перечень полей создаваемой вами таблицы.

Вы можете заимствовать структуру таблицы только в том случае, если перечень полей в вашей новой таблице пуст.

Редактирование имени поля

Если после того, как вы ввели имя поля, вы решите изменить его, укажите курсором мыши на имя и сделайте двойной щелчок или нажмите клавишу F2 для того, чтобы поместить текстовый курсор внутри поля. Далее используйте стандартные приёмы редактирования с помощью клавиш Del и Backspace. Для замены имени поля целиком, выберите его и просто начните вводить новое имя.

Блокирование записей

В многопользовательском режиме работы каждый пользователь может запретить доступ к текущей записи в общей таблице. Это мера защиты своих записей от неумышленного вмешательстве других пользователей в личную рабочую область.

Блокировка записей работает независимо от того, видите вы их или нет. Чтобы иметь возможность просматривать информацию о блокировках, dBASE - таблицы предоставляют опцию Record Lock. Если ее включить в окне Create Table, Paradox добавит в таблицу специальное скрытое поле Record Lock. Вы сможете увидеть его в том случае, если попытаетесь войти в заблокированную кем-либо запись.

В этом случае Paradox отобразит на экране информацию из поля Record Lock автоматически. Информация, которую вы увидите, зависит от формата изображения, определённого вами. Размер Record Lock поля может быть от 8 до 24 символов (по умолчанию - 16):

- Первые два символа определяют, была ли запись изменена.
- Следующие три символа информируют о времени блокирования.
- Еще три символа отображают дату блокирования.
- Оставшиеся 16 символов предоставлены для вывода имени пользователя, установившего блокировку.

Размер поля в 16 символов (по умолчанию) отображает статус записи, время и дату установки запрета, а также первые 8 символов имени пользователя, установившего запрет.

Создание индексов

Создать индексы dBASE-таблиц можно как из диалогового окна Create Table, так и из диалогового окна Restructure Table.

Щелкните мышью кнопку Define (рис.3.13) для вызова диалогового окна Define Index (это окно для dBASE-таблицы отличается от окна определения индексов Paradox-таблицы). Сделайте выбор поля из списка Field List, для которого вы хотите создать индекс. Выбранное поле появится в текстовом окошке Indexed Field. Включите на панели Options необходимые опции.

Уникальные dBASE-индексы

Когда вы отмечаете опцию Unique в диалоговом окне Define Index, вы, тем самым, определяете, что каждое значение в индексе должно быть уникальным. Когда Paradox создаёт unique-индекс для dBASE-таблицы, он включает в него первое встретившееся из дублированных значение поля и игнорирует остальные. Это означает, что дублированные значения поля могут присутствовать в таблице, но отсутствуют в индексе.

Данную опцию полезно включить, если необходимо иметь неповторяющиеся значения индекса. Это пригодится для составления модели данных при разработке формы или отчета аналогично Paradox-таблицам. Paradox может интерпретировать уникальный поддерживаемый dBASE индекс, как первичный ключ для Paradox.

Поддерживаемые (maintained) и неподдерживаемые (non-maintained) dBASE-индексы

Поддерживаемый индекс изменяется (обновляется) при любом изменении таблицы. Поддерживаемые индексы применимы только для dBASE IV - таблиц. Paradox хранит поддерживаемый индекс как часть .MDX-файла с именем исходной таблицы. Когда вы сохраняете поддерживаемый индекс, то Paradox предлагает присвоить имя (tag) этому индексу. MDX-файл может одновременно содержать несколько поддерживаемых индексов.

Неподдерживаемые индексы хранятся в .NDX -индексных файлах. Неподдерживаемые индекс не модифицируется автоматически с изменением данных в таблице.

Вы должны использовать диалоговое окно Order / Range, чтобы открывать неподдерживаемый индекс каждый раз при внесении изменений в таблицу. Вы можете одновременно иметь только один открытый неподдерживаемый индекс в таблице. Вы не имеете возможности реструктурировать или удалить данный индекс из Paradox.

Будьте внимательны: выполнение операций сложения, вычитания или очистки приводят к повреждению индекса. Если вы выполните сортировку в таблице без предварительного открытия неподдерживаемого индекса, то эта операция также приведет к повреждению индекса.

Создание убывающих dBASE-индексов

Если вы установили опцию Descending в диалоговом окне Define Index , Paradox создаст индекс в убывающем порядке (от Z к A или от Я к А). Если вы установили Descending и попытаетесь связать данную таблицу с другой, использую-щей индекс по возрастанию (от А к Z или от А к Я), то у вас ничего не получится.

Создание вычисляемых индексов

Вычисляемый индекс, т.е. индекс по вычисляемому полю, создается так же, как и индекс по одиночному поля. Например, вы можете создать вычисляемый индекс такого вида: FirstName + LastName, где слагаемые являются именами полей (все поля, включаемые в вычисляемый индекс, должны быть со-вместимы по типам). Для создания вычисляемого индекса нажмите кнопку Expression Index. Наименование кнопки изменится на Index Field, а курсор переместится в текстовое окошко Expression Index. Введите выражение, которое вы хотите использовать для индексирования. Используйте кнопку Index Field для возврата к списку полей. Когда вы вводите выражение, вам не нужно набирать имена полей на клавиатуре. Поместите текстовый курсор в текстовое окошко и нажмите клавишу мыши над полем в списке полей. Paradox автоматически скопирует имя поля в текстовое окошко.

Создание условий отбора

Условие отбора (subset condition), иногда называемое фильтром, это выражение, проверяющее на соответствие по критерию истина - ложь. Это значит, что Paradox создаёт индекс, ссылающийся только поля, значения в которых отвечают требованиям фильтрации. Например, если вы указали условие State=CA, вы сообщаете Paradox о том , что необходимо создать индекс только по тем значениям поля State, которые совпадают с «CA». Введите условие отбора в текстовом окошке Subset Condition Expression. Условие отбора вводится точно так же, как и вычисляемые индексы, описанные выше.

Сохранение индекса

Нажмите OK для сохранения индекса. Вы увидите диалоговое окно Save Index As. Если вы определили неподдерживаемый индекс, то становится доступным текстовое окошко Index File Name.

- Если вы задали индекс по одному полю, Paradox использует имя поля как имя файла и автоматически присваивает этому типу файлу расширение .NDX. Вы можете изменить имя файла, но должны использовать расширение .NDX.
- Если вы задали неподдерживаемый вычисляемый индекс, вы должны ввести имя файла (не более 8 символов). Paradox также присваивает этому файлу расширение .NDX.

Если вы определили поддерживаемый индекс по одному полю либо по выражению, то вам будет доступно текстовое окошко Index Tag Name, в которое вы должны ввести имя (tag) индекса. Это имя появится в диалоговом окне Create Table (или Restructure Table) под кнопкой Define. Для сохранения индекса Paradox создаст файл, используя имя таблицы и расширение .MDX.

Сохранение новой таблицы

Если вы закончили описание структуры таблицы, нажмите мышью кнопку Save as для сохранения созданной таблицы. Если вы не завершили работу или ввели ошибочные или некорректные параметры, Paradox сообщит о возникших ошибках в специальном окошке в нижней части панели Field Roster. При выборе Save as появится диалоговое окно Save Table As.

Введите имя таблицы в текстовом окошке New Table Name. Paradox распо-знает тип файла, который вы выбираете по информации из текстового окошка Table Type. По умолчанию Paradox сохраняет таблицу в рабочем каталоге. Используйте список Path, если вы хотите сохранить таблицу в каком-либо другом каталоге. Если вы хотите сохранить таблицу в каталоге, не представленном в списке, введите полный путь с именем таблицы в текстовое окошко New Table Name.

Реструктурирование Paradox-таблиц

По мере использования ранее созданных таблиц возникает потребность добавить, удалить или поменять местами некоторые поля, т.е. реструктурировать таблицу. Paradox предоставляет такую возможность. Процесс реструктурирования (изменения структуры) таблицы очень похож на процесс создания ее в первый раз. Если вы хотите только переименовать таблицу, вы можете использовать команду Tools / Utilities / Rename.

Вы можете изменить структуру таблицы несколькими путями:

- В окне Browser или Folder щелкнуть правой клавишей мыши на табличной иконке и выбрать из меню Restructure.
- Из меню окна Table выбрать пункт Table / Restructure.
- Из меню Desktop выбрать пункт Tools/ Utilities / Restructure, а затем вы-брать таблицу для изменения.

Любой из этих способов открывает для выбранной таблицы диалоговое окно Restructure Table, описывающее ее структуру. Внешний вид и работа с окном Restructure Table полностью аналогична работе в диалоговом окне Create Table.

В результате реструктурирования таблицы иногда появляются временные таблицы, такие как Problems , которые Paradox использует для хранения записей, не совместимых с новой структурой. Paradox нумерует временные таблицы последовательно от 1 до 99 и хранит их в вашем личном каталоге. Например, если вы реструктурировали таблицу дважды, и дважды это приводило к потере данных, Paradox создает две таблицы с именами Problems и Problem 1.

Основные правила реструктурирования

Когда вы перестраиваете таблицу, вы часто вносите изменения, которые могут привести к потере данных. Такие изменения, как уменьшение размера поля, установка контроля корректности данных или изменение типов полей, мо-гут привести к искажению данных. В любом из этих случаев Paradox откроет предупреждающее окно Restructure Warning перед тем, как закрыть диалоговое окно Restructure Table.

Ниже представлены отличия процесса реструктурирования таблицы от процесса ее создания:

- Вы не можете изменить тип таблицы. Например, нет возможности преобразовать Paradox-таблицу в dBASE-таблицу (вы не можете копировать таблицу одного типа в таблицу другого типа).
- Если вы изменяете структуру таблицы, созданной в предыдущей версии Paradox, появится предупреждающее окно, в котором Paradox запросит подтверждение на конвертирование этой таблицы в таблицу Paradox for Windows.
- Если вы добавляете первичный ключ в таблицу, которая ранее не имела ключа или имела другой ключ, могут возникнуть конфликты ключей (key violations). Это значит, что в таблицу могут уже быть введены данные, которые нарушают правила, установленные новым ключом. Paradox выделяет записи с ключевыми рассогласованиями в специальную временную таблицу Keyviol и помещает ее в ваш личный каталог каталог. Если уже существует Keyviol -таблица, Paradox добавляет порядковый номер к вновь создаваемой (Keyviol 1, Keyviol2 и т.д.). Paradox может создать до 100 временных таблиц с одним и тем же именем (первая не нумеруется, а последняя имеет номер 99). Paradox удаляет записи с конфликтующими ключами из вашей таблицы. Вы можете изменить записи в Keyviol в соответствии с требованиями ключа и затем вернуть их обратно в вашу таблицу, используя команду Tools / Utilities / Add.
- Если вы изменили тип полей и Paradox не может конвертировать некоторые данные из поля данного типа в новый тип, Paradox попросит вас подтвердить изменения. Если вы сделаете это, Paradox переместит записи, содержащие данные, которые были неконвертируемы, в специальную временную таблицу, называемую Problems. Вы можете изменить записи в Problems в соответствии с новой структурой таблицы и затем вернуть их в вашу таблицу, используя команду Tools / Utilities / Add.
- Если вы уменьшаете («обрезаете») размер поля, то Paradox запросит подтверждение на преобразование сущзствующих данных в диалоговом окне Restructure Warning. Если вы выполнили такое преобразование данных, то Paradox переместит записи, содержащие данные, которые не умещаются в новый размер поля, в таблицу Problems.
- Если вы изменяете или добавляете правила корректности значений (validity checks) к уже существующим в таблице данным, то все неподчиняющиеся этим условиям данные Paradox помещает в таблицу Keyviol. Вы можете изменить записи в Keyviol и затем вернуть их обратно в таблицу, используя команду File futilities! Add.
- Если вы изменяете драйвер языка таблицы в процессе её реструктурирования, то рискуете потерять множество специальных символов, которые могут в ней присутствовать.

Сужение поля

Когда вы сужаете поле, которое уже содержит данные, то можете потерять часть информации в нем. В этом случае Paradox откроет диалоговое окно Restructure Warning, которое позволяет вам решить, будут ли «обрезаны» данные, выходящие за пределы нового размера поля, или сохранены в таблице Problems.

Добавление и удаление полей в существующей таблице

Добавление полей в процессе реструктурирования существующей таблицы аналогично добавлению полей в новой таблице.

Когда вы добавляете поля в существующую таблицу, Paradox не вводит автоматически эти поля в формы, отчёты и запросы, связанные с таблицей. Если вы хотите, чтобы новые поля присутствовали в этих документах, вы должны последовательно добавить их в каждый из них.

Удаление полей существующей таблицы производится аналогично удалению поля из новой таблицы. Однако существуют ряд отличий, на которые следует обратить внимание:

- Удаление поля обычно приводит к потере данных (за исключением случая, когда поле пустое). Paradox предупредит вас о потере данных и запросит подтверждение на удаление.
- Если удаляемое из таблицы поле присутствует в каком-либо разработанном документе (форме, отчете), то конструкционный объект, содержащий его, теряет свое определение. В следующий раз, когда вы откроете документ, то будете вынуждены либо переопределить его либо удалить совсем. Если удаленное поле присутствует в запросе, то запрос не откроется.

Редактирование имени поля

Для изменения имени поля таблицы, войдите в него, дважды щелкнув на нем мышью или нажав клавишу F2, а далее используйте стандартные средства редактирования текста. Для полной замены существующего имени, отметьте поле и напечатайте новое имя поверх старого. Если вы редактируете имя поля существующей таблицы и на это поле имеется ссылка в каком-либо документе, Paradox попытается согласовать изменения при следующем вашем обращении к данному документу.

Преобразование неключевого поля в ключевое поле

Когда вы преобразовываете неключевое поле в ключевое, помните, что поля с ключами должны располагаться последовательно друг за другом и находиться в самом верху перечня полей (если это необходимо, вы можете перемещать поля, как описывалось выше).

Изменение типов полей в Paradox-таблицах

Действия по изменению типа поля существующей таблицы идентичны определению типа поля во время создания таблицы: вы просто переписываете существующий символ типа поля на новый. Однако следует помнить, что изменение типа поля существующей таблицы может привести к потере данных или их порче. В процессе преобразования Paradox запросит у вас подтверждение на изменение типа данных. Изменения типа полей можно производить только для определенных типов полей, которые представлены ниже в таблице.

Таблица 3.6 Допустимые преобразования типов полей Paradox-таблиц

	A	N	$	D	S	M	F	B	G	0
A	+	P	P	P	P	+				
N	+	+	+		P					
$	+	+	+		+					
D	+			+						
S	+	+	+		+					
M	+					+	+	+		
F						+	+	+		
B								+		
G								+	+	
O								+		+

\+ - означает, что Paradox разрешает данное преобразование данных, однако это может привести к их потери. Если Paradox вынужден «обрезать» данные, он запросит разрешение на преобразование в диалоговом окне Restructure Warning.

P - означает, что преобразования разрешены в Paradox, однако они могут породить таблицу Problems.

В таблице показаны совместимые типы полей. Совместимость полей относится также и к другим операциям с таблицами, таким как Add, Subtract, Copy и Query.

Преобразования алфавитно-цифрового поля

Результаты преобразования других типов полей в алфавитно-цифровое поле могут быть различными. Все форматы и особенности, присущие другим полям, теряются.

Если вы преобразовываете какой-либо тип поля в алфавитно-цифровой, вы должны определить размер поля. Если уже существующие в поле данные содержат больше символов, чем вновь определенный вами размер алфавитно-цифрового поля, то вы можете «обрезать» их или переместить в таблицу

Problems. Для преобразования алфавитно-цифрового поля в поле типа даты, числовое или денежное и наоборот вы должны удостовериться, что установки в Control Panel системы Windows и файле конфигурации ODAPI.CFG идентичны.

Преобразования числовых и денежных полей

Вы можете преобразовывать числовые, денежные и поля типа короткое число из одного в другой тип без потери данных за исключением тех случаев, когда значения слишком велики для поля типа короткое число или включают в себя десятичные знаки. В этом случае вы можете либо «обрезать» данные либо переместить их в таблицу Problems.

Преобразование поле даты (date)

Следующая таблица показывает, какие текстовые строки могут быть преобразованы в даты.

Таблица 3.6 Преобразование текстовых строк в даты

Строки, которые могут быть преобразованы в дату	Строка, которые не могут быть преобразованы в дату
5/05/1812	5 мая 1812
3/12/93	12 марта 1993
9-May-1945	День Победы 1945
11-Nov-18	Armistice Day
1.01.1999	New Year's Day, the year 1999
14.10.61	My birthday

Если вы настраиваете формат дат, используя Configuration Utility, то значения дат автоматически преобразовываются в соответствии с вашими новыми установками.

Реструктурирование таблиц, связанных системой ссылок

Когда вы перестраиваете основную таблицу, которая находится во взаимодействии с совокупностью других таблиц, на вас могут быть наложены некоторые ограничения по осуществлению операций реструктурирования.

Для того, чтобы просмотреть основную таблицу, связанную системой ссылок с другими таблицами, нажмите кнопку Dependent Tables на панели Table Properties в диалоговом окне Restructure Table. Paradox представит список всех дочерних таблиц, которые зависят от редактируемой таблицы.

Помните, что при реструктурировании родительской таблицы вы не можете выполнять такие операции, которые приводят к удалению записей из таблицы. Если вы удаляете данные из родительской таблицы, то вы рискуете оставить несвязанными данные (т.е. получить «висячие» строки) в дочерних таблицах.

Это является серьёзным нарушением целостности базы данных. Каждая запись в дочерней таблице должна иметь корректную связь с записью в родительской таблице.

Для корректного реструктурированию таблиц, которые связаны с другими таблицами посредством системы ссылок, необходимо придерживаться следующих правил:

- Если вы изменяете размер поля в основной таблице, то вы должны «обрезать» данные, которые не помещаются в поле нового размера, а не сохранять их в таблице Problems.
- Не изменяйте определение ключа основной таблицы или определение заимствованного ключа дочерней таблицы таким способом, чтобы конфликтующие записи сохранялись в таблице Keyvoil.
- Вы можете изменять имена полей, но не изменяйте типы и размеры тех полей, которые являются частью системы ссылок к другим таблицам.
- вы можете добавить правила контроля корректности данных в обе таблицы, но вы не можете применить их к существующим данным (используйте для этого диалоговое окно Restructure Warning). Исключением из этого правила является установка контроля корректности типа «значение по умолчанию» при создании нового поля таблицы.
- Для того чтобы сделать родительскую таблицу дочерней таблицей другой таблицы, данная таблица и все ее дочерние таблицы должны быть пусты. Например, если таблица Orders является родительской таблицей для таблицы Stock, то вы не можете сделать Orders дочерней для таблицы Customer до тех пор, пока обе эти таблицы не будут пусты.
- Если вы связали посредством системы ссылок более чем две таблицы, содержащие данные, то первой вы должны определить связь с той, которая не имеет родительской таблицы. Например, для

создания системы ссылок между таблицами Customer, Orders, Lineitem и Stock, вы должны:

1. Сначала создать связь от Orders к Customer.
2. Затем создать связь от Lineitem к Orders.
3. И, наконец, создать связь от Stock к Lineitem.

- Чтобы создать циклическую систему ссылок (от A к B, от B к C и обратно от C к A), все эти таблицы должны быть пусты.

Реструктурирование dBASE-таблиц

Вы можете реструктурировать dBASE-таблицы аналогично Paradox-таблицам. Выберите пункт меню File (Utilities (Restructure, либо проинспектируйте иконку таблицы в окне Browser или Folder, а затем выберите пуню-меню Restructure, либо выберите пункт меню Table / Restructure окна Table. Чтобы реструктурировать dBASE III+ -таблицы, Paradox должен сначала преобразовать их в dBASE IV -таблицы.

Упаковка таблиц

Если вы удаляете записи из dBASE-таблицы (на самом деле удаления этих записей из таблицы не производится, и их можно просмотреть, установив опцию Table / Show Deleted в окне TaЬIe), то вам может понадобиться их физическое удаление из таблицы. Paradox предоставляет вам возможность выполнить эту операцию во время ее переструктурирования. Это называется упаковкой таблиц. Для упаковки таблицы следует включить опцию Pack Table диалогового окна Restructure Table, и если затем вы нажмёте Save, Paradox удалит записи из таблицы физически. Если вы нажмёте Save As, Paradox удалит записи из вновьсозданной в результате реструктурирования таблицы, а исходную таблицу оставит без изменений.

Изменение типов dBASE-полей

Возможные варианты преобразования типов dBASE-полей приведены в следую-щей таблице.
Таблица 3.7 Преобразования типов полей в dBASE-таблицах

	C	F	N	D	L	M
C	+	P	P	P	+	+ .
F	+	+	+		+	
N	+	+	+		+	
0	+			+		
L	+	+	+		+	
M	+					+

+-означает, что Paradox разрешает преобразовывать данные, однако это может привести к их потере. Если Paradox вынужден «обрезать» данные, он запросит разрешение на преобразование в диалоговом окне Restructure Warning.
P- означает, что преобразования разрешены в Paradox, однако они могут породить таблицу Problems.

Преобразование числового поля в символьное

Вы можете преобразовывать данные из числового поля и числового поля с плавающей точкой в символьное без потери данных, если последнее имеет достаточный размер. Однако, вы не сможете производить операции вычисления над числовыми данными, сохраненными в символьном поле.

Преобразование символьного поля в числовое

Вы можете преобразовать символьное поле в поле числовое и в числовое с плавающей точкой и при этом получить следующие результаты

- Если данные в символьном поле состоят из цифр, то Paradox преобразует их в числовые или в числовые с плавающей точкой без потери данных.
- Если данные в символьном поле состоят из текста и цифр и начинаются с цифр, тогда Paradox преобразует цифры в числа или в числа с плавающей точкой, а весь текст удалит.
- Если данные в символьном поле состоят из текста и цифр и начинаются с текста, тогда Paradox присвоит числовому полю или полю с плавающей точкой значение нуль.

Преобразование логического поля в символьное

Paradox преобразует логические значения типа True и False в символьные величины. В результате получается только один символ (буква T или F).

Преобразование символьного поля в логическое

Paradox преобразует символы T,t,Y и y в логическую истину, а все остальные в логическую ложь

Преобразование поля даты в символьное

Вы можете преобразовать значение даты в символьное значение, которое будет представлено в том формате, который задан в утилите конфигурации Configuration Utility.

Преобразование символьного поля в поле даты

Данное преобразование совпадает с аналогичным для Paradox-таблиц, рассмотренным выше.

Сохранение преобразованной таблицы

Вы можете сохранить преобразованную таблицу двумя командами:

- **Save** : Таблица с новой структурой записывается на место старой. При этом Paradox предупредит вас о потере данных в старой таблице
- **Save As** ; Создается новая таблица с новой структурой. Включите опцию Add Data To New Table на панели Options в диалоговом окне Save Table as для сохранения данных из старой таблицы в новой.

После того, как диалоговое окно Restructure Table будет закрыто, Paradox откроет окно реструктурированной таблицы, а также все Keyviol, Problems или другие временные таблицы, созданные в процессе преобразования.

Глава 4. Просмотр данных

В данной главе рассматривается вывод данных на экран в виде таблиц, форм и отчетов. Вы узнаете, как производить следующие операции:

- Настраивать таблицы для просмотра
- Изменять свойства таблиц и форм
- Использовать меню команд и SpeedBar для работы с таблицами и формами
- Предварительно просматривать на экране отчёты, подготовленные для пе-чати

Способы просмотра данных

Paradox предоставляет несколько способов просмотра данных:

- Использовать окно **Table** для просмотра данных в таблице, перемещаясь по столбцам и строкам. Вы можете использовать как стандартный формат таблиц, так и разработанный вами самими.
- Использовать окно **Form** для отображения табличных записей не табличным способом. Данный режим является очень гибким. Вы можете просматривать сразу все или только несколько полей одной таблицы, а также любую комбинацию полей из нескольких таблиц.
- Использовать окно **Report** для предварительного просмотра отчёта на экране перед его распечаткой.
- Использовать окно **Chart** для предварительного просмотра графика на экране перед его распечаткой
- Paradox позволяет отображать одновременно данные в различных режимах и в отдельных окнах, причём количество открытых окон может быть практически не ограничено.

Использование таблиц

Данный раздел рассматривает способы представления таблиц на экране, приемы работы в окне Table и возможности изменения изображения таблиц, а также распечатывание табличных данных. В зависимости от ситуации у вас может возникнуть желание поразному отображать данные. Paradox предоставляет практически неограниченное число способов просмотра данных, находящихся в таблицах.

Чтобы открыть на экране таблицу, необходимо выбрать команду File / Open / Table из меню Paradox Desktop. Вы увидите диалоговое окно Open Table. Вам остается выбрать с его помощью нужную таблицу. Если на Desktop или в окне Folder есть иконка нужной вам таблицы, вы можете дважды щелкнуть её мышью, и таблица будет открыта непосредственно, без использования диалогового окна Opel Table.

Paradox открывает таблицу в окне Table. При этом меню и SpeedBar отображают операции, которые вы можете использовать для работы с таблицей.

Заметьте, что команды ввода данных не будут активны до тех пор, пока вы не включите режим редактирования.

Строки и столбцы, записи и поля

Рис.4.1 положение поля и записи в таблице.

Paradox -таблицы состоят из строк и столбцов. Каждая строка называется записью и содержит всю имеющуюся информацию о каком-либо одном объекте. Каждый столбец называется полем и содержит какой – либо один элемент информации, составляющей запись.

Поля Paradox – таблицы могут содержать различные виды информации. Paradox предусматривает для каждого из них определённый тип поля, который определяет, какой вид информации может храниться в данном поле (типы полей рассмотрены в гл 1)

Перемещение по таблице

Для перемещения по записям таблицы используйте меню Record или иконки на SpeedBar. Меню Record представляет следующие команды перемещения:

First – переход к первой записи

- Last – переход к последней записи
- Next – переход к следующей записи
- Previous – переход к предыдущей записи
- Next Set – переход к следующей группе записей
- Previous Set – переход к предыдущей группе записей

Для перемещения по таблице можно использовать клавиатуру. Для перехода от одного поля к другому служат клавиши Home, End и клавиши управления курсором.

Использование линейки прокрутки (Scroll bars)

Перемещение от записи к записи вы можете, щелкая мышью стрелки вверх или вниз, находящиеся на вертикальной линейке прокрутки (scroll bar). Аналогично, для перемещения по полям записи можно использовать горизонтальную линейку прокрутки.

Использование блокировок прокрутки (Scroll lock)

Если вы хотите зафиксировать на экране один или несколько столбцов во время горизонтального перемещения по таблице, можно использовать блокировку прокрутки (обозначается на экране треугольником и находится в нижнем левом углу таблицы) справа от того столбца, которым заканчивается группа. Для этого необходимо отбуксировать мышью маркер блокировки и установить его на правой вертикальной линии поля, которое вы хотите зафиксировать. Теперь все поля слева от маркера будут при перемещении постоянно находиться на экране.

Использование SpeedBar окна Table

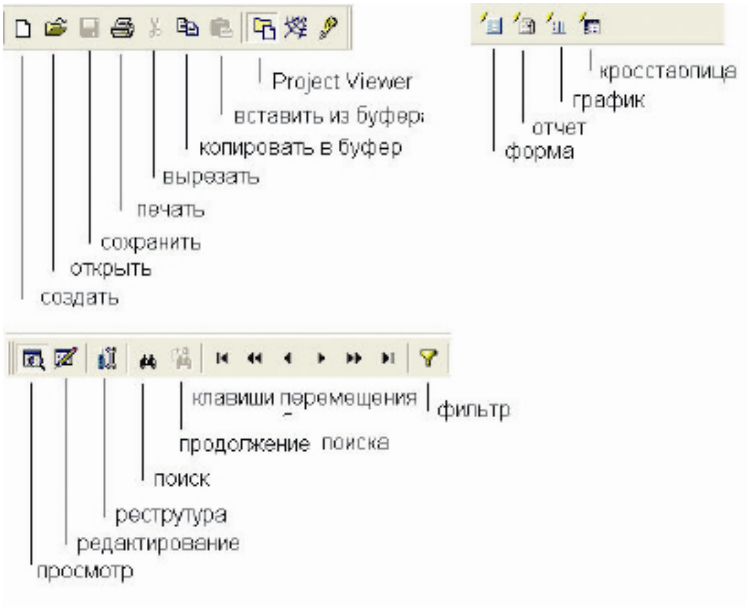


Рис.4.2 SpeedBar окна Table

Cut, Copy, Past (вырезать, копировать в буфер, вставить из буфера) используются для перемещения данных в Windows Clipboard и из него.

Print для распечатки текущего отчета. Если Вы не указали, какой отчет в данный момент является текущим, Paradox распечатает по умолчанию отчет стандартной формы.

Кнопки First, Last, Previous, Next, Previous set, Next set (кнопки перемещения) предназначены для перемещения к первой, последней, предыдущей, следующей, предыдущей группе и следующей группе записей соответственно.

Используйте кнопки со стрелками для перемещения по окну Table и кнопки Locate для поиска полей или записей таблицы по заданным значениям.

Кнопка Field View предназначена для входа в режим просмотра полей и выхода из него, а кнопка Edit Data - для входа и выхода из режима редактирования (нажатая кнопка означает, что вы находитесь в данном режиме).

Graf – выводит график для данных полей, где стоит курсор. Report, Form - соответственно отчет и экранная форма. Edit data – включение – выключение режима редактирования

Изменение способа отображения

Вы можете изменить свойства, т.е. визуальное представление и параметры таблицы, которую вы просматриваете. Paradox предоставляет вам возможность перемещать и изменять ширину столбцов, варьировать изображение линий сетки, а также контролировать отображение и формат данных.

Вы можете изменить формат таблицы двумя способами:

- Проинспектировать нужный элемент таблицы.
- Пометить область, которую вы хотите изменить, после чего выбрать соответствующую команду из пункта меню Properties

Непосредственные манипуляции с таблицей

Непосредственные манипуляции представляют собой работу с изображением таблицы при помощи мыши:

- вы можете изменить форму, размер и положение на экране любого объекта. Рисунки иллюстрируют, каким образом изменяется форма курсора мыши во время прохождения тех областей в таблице, где вы можете использовать мышь для изменения размеров столбцов или их перемещения, а также изменения высоты строк и заголовка таблицы.

Манипуляции со столбцами

Для перемещения столбца, установите курсор мыши на заголовок столбца и нажмите левую клавишу мыши. Если курсор находится в правильном положении, его форма изменяется. Затем отбуксируйте столбец в новую позицию.

Если вам нужно изменить местоположение столбца, вы можете выбрать его (сделать его текущим) и нажать комбинацию клавиш Ctrl+R. Paradox переместит столбец в конец таблицы, а все находящиеся за ним столбцы передвинутся на одну позицию влево.

Вы можете изменить размеры столбца, поместив курсор на его правой вертикальной линии в области заголовка или верхней строки данных (при этом курсор изменит свою форму на двухстороннюю стрелку), и отбуксировать вертикальную линию влево или вправо для увеличения или уменьшения ширины столбца.

Манипуляции со строками

Аналогично, вы можете изменить высоту строк таблицы, если отбуксируете линию, которая находится под номером первой записи в таблице.

Инспектирование и изменение свойств объектов

Чтобы изменить свойства какого-либо элемента таблицы, вы должны его проинспектировать, чтобы на экране появилось меню выбора свойств.

Инспектирование при помощи мыши осуществляется щелчком правой клавиши над объектом. Вы можете изменять отдельные поля (столбцы) таблицы, сетку таблицы или заголовки столбцов.

Вы также можете инспектировать свойства областей таблицы, используя следующие команды меню или определённые комбинации клавиш на клавиатуре:

- Для изменения свойств текущего поля нужно нажать клавишу F6, ли-бо выбрать пункт меню Properties / Data.
- Для изменения свойств сетки нужно нажать клавиши Ctrl+G или выбрать пункт меню Properties / Grid.

Все поля и все заголовки имеют в меню своих свойств пункты Alignment (выравнивание), Color (цвет) и Font (шрифт).

Учтите, что если ваша таблица не содержит данных, вы должны сна-чала войти в режим редактирования, и только после этого вы сможете изменять свойства столбцов. Для этого нажмите F9 или щелкните кнопку Edit Data на SpeedBar (более подробно вход в режим редактирования и выход из него описаны ниже в этой главе).

Установка режима выравнивания

В режиме редактирования активизируется окно работы с текстом, где можно задать шрифт, выполнить выравнивание (определяет расположение данных в поле или текста в заголовке). Текст и данные могут быть выравнены по горизонтали (по левому или правому краю столбца либо центрированы) и по вертикали (по верху, центру или низу строки).

Выбор цвета

Вы можете изменить цвет любой части таблицы: фона таблицы, линий сетки, конкретного поля, фона и символов столбца, а также фона и символов заголовка. Если вы выберете в меню свойств

(properties) объекта цвет (шрифта, поля, таблицы) , надо нажать клавишу «применить» и инспектируемая вами часть таблицы изменит свой цвет на новый

Выбор шрифта

Вы можете изменять внешний вид текста в полях и заголовках, инспектируя текст и выбирая пункт Font из меню его свойств (рис. 4.3 вид слева). Вы можете выбрать по своему усмотрению начертание, размер, стиль и цвет символов шрифта. Для получения меню доступных видов начертания шрифта нужно выбрать пункт Typeface. Перечень доступных для вас видов начертания определяет-ся наличием установленных в Windows шрифтов. Как правило, стандартный перечень включает в себя типы Helvetica, Times Roman, Courier и System, но может быть и намного шире. Щелкните мышью желаемый шрифт в меню, и Paradox будет использовать его в выбранной области таблицы. Для изменения размера символов текста выберите пункт Size. Paradox покажет меню с доступными размерами. Укажите в нем желаемый размер для использования в выбранном тексте. Для изменения стиля текста выберите пункт Style, и вы увидите доступные типы стилей текста

- Normal - снимает все атрибуты текста
- Bold - жирный текст
- Italic – курсив
- Strikeout - текст, перечеркнутый горизонтальной линией
- Underline - подчёркивание

Для изменения цвета выбранного вами текста выберите пункт Color.. Для изменения шрифта выберите поле, которое вы хотите изменить, а затем желаемый вариант из палитры. Выбранный текст изменится в соответствии с выбранным вариантом.

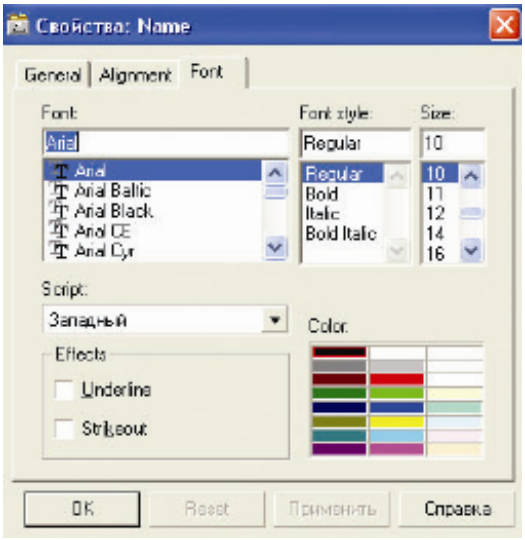


Рис.4.3 окно изменения шрифта в Properties

Пример. Изменение свойств таблицы

Предположим, вы хотите изменить внешний вид таблицы Customer.

1. Нажмите кнопку Open Table на SpeedBar, затем выберите файл Customer.db из появившегося меню.
2. Листайте таблицу, используя горизонтальную линейку прокрутки окна Table, пока не увидите поле с названием First Contact.
3. Установите курсор в область заголовка First Contact. Курсор примет вид треугольника, показывая, что вы можете перемещать поле. Отбуксируйте поле First Contact в положение справа от поля Customer No.
4. Проинспектируйте поле Customer No. Выберите Alignment из появившегося меню и затем Center. Записи в поле Customer No. будут отцентрированы.
5. Проинспектируйте первый столбец в области заголовка таблицы. Выберите пункт Color и измените фон поля на белый.
6. Для сохранения текущих свойств таблицы дайте команду Properties / View / Properties Restore.

Изменение свойств в соответствии с диапазоном данных

Вы можете изменить атрибуты всех данных в поле, которые удовлетворяют определенным требованиям. Предположим, что вы хотите выделить белым фоном все значения в поле Qty таблицы Lineitem, которые меньше 5 .

Вы можете сделать это, используя пункт Data Dependent в меню свойств поля Qty. Поля типа алфавитно-цифровое, числовое, короткое число, дата, денежное, а также поля dBASE -таблиц типа символьное, число с плавающей точкой, дата и логическое имеют в меню своих свойств пункт Data Dependent. Используйте его для определения диапазона величин, который должен быть выделен. Для определения диапазона данных необходимо проинспектировать поле и выбрать Data Dependent из его меню. Вы увидите диалоговое окно Data Dependent Properties. Все ранее определённые вами диапазоны появятся в списке Range. Вы можете задать любое количество различных диапазонов

Пример. Определение свойств для диапазона значений показан на рис.4,4

1. Откройте таблицу Lineitem (c:\Program Files\Corel\WordPerfect Office 2000\ Paradox\ FwSample\lineitem) и проинспектируйте запись в поле Qty. Из меню записи выберите Data Dependent, и Paradox откроет диалоговое окно Data Dependent Properties.
2. Выберите режим New Range.
3. Введите границы диапазона на панели Range includes Values. Включите опцию >=, затем введите 0 в верхнем текстовом поле. Это установка начала диапазона. Она означает «больше или равно 0». Затем выберите опцию <= и введите 5 в нижнем текстовом поле. Это установка конца диапазона, и она означав «меньше или равно 5». Слово And на панели Range Includes Values поможет вам определить диапазон как «больше или равно 0 и меньше или равно 5».

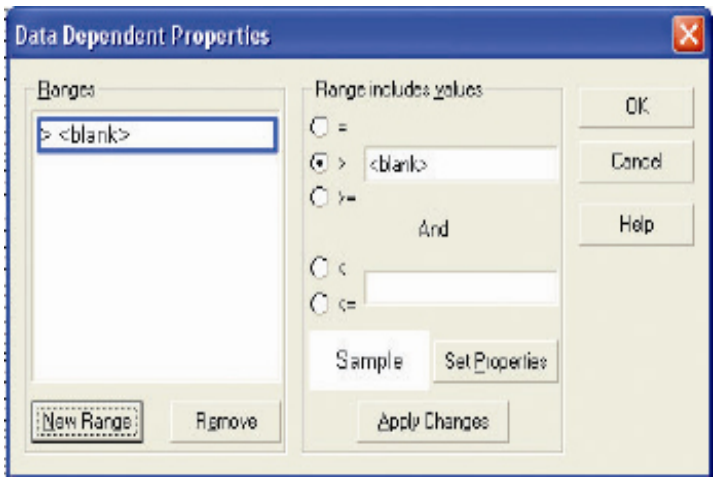


Рис.4.4 окно Data Dependent Properties

4. После того, как вы задали диапазон, необходимо определить атрибуты попадающих в него величин. Для этого проинспектируйте область Sample или нажмите кнопку Set Properties. Бы увидите меню Font Samples, в котором можно установить цвет фона и символов, а также стиль, размер и вид начертания. Установите атрибуты черные символы на белом фоне. Нажмите Apply Changes для сохранения диапазона с выбранными атрибутами. После этого данный диапазон появится в списке Ranges.
5. Нажмите OK для возврата в окно Table. Данные, значения которых попадают в заданный вами диапазон, будут изображены черными символами на белом фоне.

Диапазон, задаваемый вами в диалоговом окне Data Dependent Properties (рис. 4.4), не обязательно должен быть числовым. Вы можете установить диапазон дат или указать последовательность текстовых символов. Например, вы можете задать один диапазон, чтобы все значения в поле States таблицы Customer, которые равны CA, изображались желтым курсивом, и второй - чтобы все даты, равные 1991 , должны были быть написаны подчёркнутыми голубыми символами.

Для того, чтобы применить определённые свойства к заданному вами диапазону, необходимо нажать кнопку OK. Paradox закроет диалоговое окно Data Dependent Properties, найдёт в таблице нужные величины и изменит их атрибуты.

Заметьте, что свойства диапазона значений замещают свойства, заданные вами для всего столбца. Если, например, вы выбрали для столбца голубой цвет фона, то он не будет применён к данным, попадающим в пределы определённого вами диапазона. Эти записи будут использовать цвет, который вы указали для диапазона.

Изменение сетки

Сеткой называется рисунок из линий, разделяющих столбцы и (при желании) строки таблицы. Вы можете изменить цвет, стиль и количество отображаемых линий и установить атрибуты для выделения текущей записи. Режим вызывается через Properties/Grid

Изменение фона сетки

Фон сетки - это все пространство в окне Table, которое не занято самой таблицей. Для изменения фона сетки проинспектируйте его и из появившегося меню выберите пункт Color. Вы увидите палитру Color, которая была описана выше в этой главе.

Вы можете изменить фон отдельного столбца, проинспектировав его и выбрав пункт меню Color. Вы также можете изменить цвет всех столбцов одновременно, нажав Shift+F6 на клавиатуре. При этом появится меню All. Paradox применит все изменения, сделанные вами с помощью этого меню, ко всем столбцам.

Изменение линий сетки

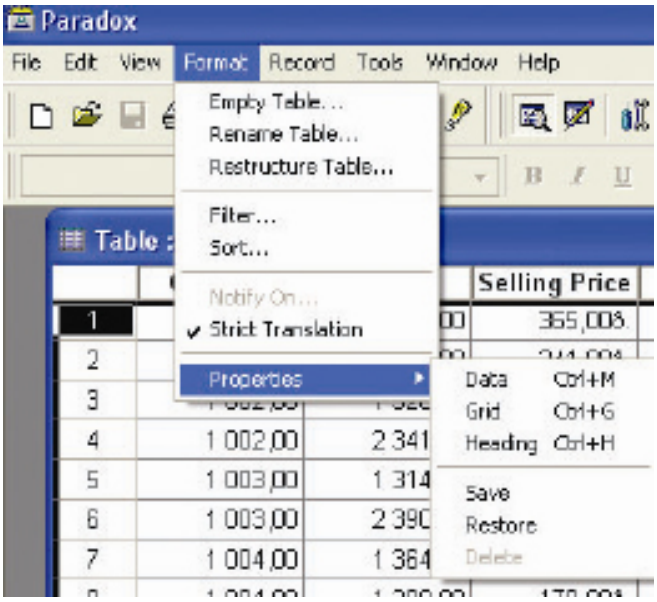


Рис.4.5 Меню свойств сетки

Вы можете придать сетке практически любой внешний вид. Проинспектируйте ее и выберите пункт Grid Lines

Paradox предоставит возможность указать, какие линий сетки следует выводить на экран. Вы можете выбрать :

- Heading lines для отображения линий в заголовке
- Column Lines для отображения вертикальных линий, разделяющих колонки
- Row Lines для отображения горизонтальных линий между записями в таблице

Кроме того, Paradox позволяет задать следующие параметры линий сетки:

- Line Style - один из пяти различных типов линий, включая различные стили штриховых линий. Вам будет предоставлена палитра Line Style. Выберите в ней желаемый стиль, и Paradox соответственно изменит все линии сетки.
- Color - цвет линий.
- Spacing - количество линий между каждым столбцом или каждой строкой. Вы можете задать одинарную, двойную или тройную линию.

Изменение способа выделения текущей записи

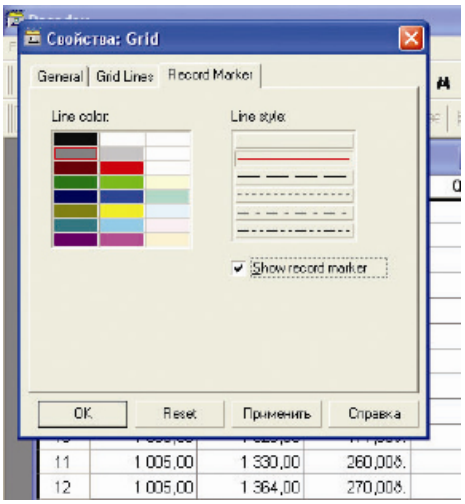


Рис.4.6 Record Marker

Вы можете потребовать выделять текущую запись специальным маркером. Маркер (Current Record Marker) - это горизонтальная линия, расположенная под текущей записью. Проинспектируйте сетку таблицы и выберите из появившегося меню пункт Current Record Marker. В открывшемся меню вы можете выбрать сле-дующие пункты (на рис. 4.6 показан именно этот режим):

- Show - показывает или убирает маркер с экрана.
- Line Style - позволяет выбрать тип линии для изображения маркера.
- Color - показывает палитру цветов, в которой вы можете выбрать цвет маркера.

Быстрый просмотр объектов

Вы можете использовать команды Table / Quick или кнопки Quick на SpeedBar для просмотра форм, графиков, кросстаблиц или отчётов, при этом, изображение документа зависит от того, был ли ранее определён текущий (preferred) объект при помощи команды Properties / Preferred. Если нет, Paradox использует стандартную форму для представления объекта данного типа.

Учтите, что когда вы просматриваете данные вашей таблицы в альтернативных форматах (таких, как форма или отчет), свойства, заданные вами в окне Table, не используются. Внешний вид форм и отчетов настраивается отдельно.

Когда вы выбираете пункты Preferred / Form, Graph или Crosstab, появляется диалоговое окно со словом <Forms> в списке Type (графики и кросстаблицы являются объектами, которые могут быть помещены в формы). Выберите желаемую форму. Модель представления данных выбранной формы должна содержать таблицу, которую вы хотите просматривать в виде этой формы.

Вместо использования команд Properties / Preferred, вы можете проинспектировать кнопки Quick Form, Quick Report, Quick Graph или Quick Crosstab на SpeedBar.

Отчет, который вы выбрали в качестве текущего, может быть многотабличным. В этом случае данная таблица должна быть главной (master table) в модели данных отчёта.

Существует четыре вида объектов, использующихся для быстрого просмотра :

- Выберите Table / Quick Form для вызова текущей (или используемой по умолчанию) формы данной таблицы. Окно Form откроется поверх открытого окна Table. В окне Form вы можете использовать команду Table View или аналогичную кнопку на SpeedBar, чтобы вернуться к просмотру таблицы, либо вы можете просто щелкнуть мышью в любом месте окна Table, и оно станет активным.
- Выберите Table / Quick Report для просмотра на экране текущего (или используемого по умолчанию) отчета.
- Выберите Table / Quick Graph для просмотра текущего (или используемой по умолчанию) графика.
- Выберите Table / Quick Crosstab для просмотра текущей (или используемой по умолчанию) кросстаблицы.

Сохранение свойств таблицы

Для сохранения всех сделанных вами изменений свойств таблиц, включая изменения атрибутов отдельных полей, выберите пункт Properties / View Properties / Save. Табличные данные Paradox сохраняет при их вводе, поэтому команды Fife / Save и File / Save As становятся временно недоступными в окне Table.

Paradox сохраняет заданные вами свойства в файлах с расширением .TV (атрибуты для dBASE-таблиц сохраняются в файлах с расширением .TVF). На-пример, свойства, которые вы определили для таблицы Customer, будут сохранены в файле Customer.tv.

Если вы попытаетесь закрыть окно Table без сохранения изменённых свойств, Paradox спросит вас о необходимости сохранить их.

Если вы уже модифицировали свойства таблицы, а затем изменили свое решение, вы можете выбрать пункт Properties / View Properties / Restore, и Paradox восстановит все первоначальные (или сохраненные перед этим) свойства. Файл с расширением .TV (или .TVF) можно удалить, выбрав Properties / View Properties / Delete. В этом случае для табличных атрибутов Paradox будет использовать уста-новки по умолчанию.

Определение свойств таблицы, используемых по умолчанию

Предположим, что вам часто приходится просматривать числовые поля в формате General, или поля типа дата, выравненные по левому краю, или текст, отображенный голубым цветом. Paradox предоставляет вам возможность установить атрибуты для поля каждого типа и сохранить их в файле текущих свойств.

Первым из способов задать текущие свойства для каждого типа поля является создание в личном каталоге таблицы с именем Default, в которой будет по одному полю каждого типа.

Откройте таблицу Default в окне Table, включите режим редактирования и проинспектируйте каждое поле, устанавливая нужные свойства, а затем сохраните их в файле DEFAULT. TV командой Properties / View Properties / Save.

Всякий раз, когда вы будете работать с таблицей, не имеющей своего собственного файла с расширением .TV, Paradox применит к ней установки из DEFAULT.TV. Специально определенные для таблиц файлы с расширением .TV, замещают установки файла DEFAULT.TV.

Другой способ создания файла DEFAULT.TV состоит в использовании уже настроенной по вашему желанию таблицы и копировании ее в файл DEFAULT.DB в личном каталоге. Paradox, затем использует ее .TV -файл при определении текущих свойств.

Чтобы сэкономить место на диске, вы можете удалить файл DEFAULT.DB, равно как и любые другие файлы с именем DEFAULT (например, с расширением .PX или .VAL), которые были скопированы вместе с таблицей. Единственное, что необходимо оставить файл DEFAULT.TV.

Файлы текущих свойств для dBASE -таблиц создаются аналогично, только они будут иметь расширение .TVF . Файл текущих свойств по умолчанию для dBASE -таблиц должен называться DEFAULT.TVF.

Удаление данных из таблиц

Для удаления всех данных из просматриваемой таблицы следует выбрать пункт меню Table / Empty. Paradox предупредит вас о том, что все данные из таблицы будут удалены и предложит подтвердить удаление или отменить данную операцию.

Если Paradox удалил из таблицы данные, вы уже не сможете восстановить их. Поэтому, перед тем, как производить операцию удаления, рекомендуется создавать резервные таблицы с данными, которые вам еще могут понадобиться.

Просмотр структуры таблицы

Иногда бывает необходимо просмотреть структуру таблицы - порядок следования и типы ее полей, определенные вами ранее способы контроля значений (validity checks) в каждом поле или используемые при вводе данных таблицы-справочники (lookup table).

Для этого выберите пункт Tools / utiliters / Info Structure, и на экране появит-ся окно Structure Information. Данное диалоговое окно похоже на окна Create Table и Restructure Table, однако здесь вы не можете делать какие-либо изменения в структуре таблицы. При перемещении по перечню полей таблицы, Paradox показывает способ контроля значений текущего поля. Если вы захотите получить информацию об используемых таблицах-справочниках, вторичных индексах, драйвере языка таблицы или системе ссылок между таблицами (referential integrity), сделайте соответствующий выбор из списка Table Properties.

Переименование таблицы

Чтобы переименовать таблицу, выберите пункт меню Tools / utiliters / Rename. Вы увидите диалоговое окно Rename. Сначала надо будет выбрать объект, который хотим переименовать, а потом ввести новое имя для него. Введите новое имя в текстовое поле окна.

Изменение структуры таблицы

Чтобы изменить структуру таблицы, выберите пункт меню Tools / utiliters / Restructure (или кнопку на панели инструментов). Вы увидите диалоговое окно Restructure просматриваемой таблицы. Процесс реструктурирования таблиц подробно рассматривается в Главе 3.

Сортировка таблицы

Когда вы сортируете таблицу, вы указываете Paradox, чтобы он в соответствии с вашим требованием изменил порядок расположения записей в таблице.

Сортировка таблицы, имеющей ключ

Если таблица имеет ключ, то Paradox всегда сортирует записи по значениям в ключевом поле (или полях).

- Если вы хотите посмотреть данные таблицы в каком-либо другом порядке без реального изменения местоположения записей, можно использовать команду Table/Order/Range.
- Если вы хотите изменить реальный порядок записей в таблице, вы можете отсортировать таблицу, имеющую ключ, в новую таблицу.

Нельзя изменить установленный ключом порядок сортировки записей. Однако, вы можете использовать вторичные индексы для изменения порядка просмотра таблицы, имеющей ключ, при этом физическое расположение записей в таблице не изменяется. Для такого изменения порядка записей при просмотре таблицы следует использовать диалоговое окно Order / Range (но не окно Sort Table) (см. «Просмотр данных в различных порядках и в различных диапазонах» ниже в данной главе).

Если же вы хотите изменить физический порядок записей в таблице с ключом, необходимо произвести пересортировку и сохранить записи в другой таблице. Новая таблица, созданная операцией сортировки, не будет иметь ключа, а исходная таблица останется без изменений.

Сортировка таблиц, не имеющих ключа

Если таблица не имеет ключа, то записи в ней располагаются в том порядке, в котором их ввели.

Если вы выполняете сортировку таблицы, не имеющей ключа, то в ней изменяется реальное расположение записей. Вы должны сообщить Paradox, по какому полю таблицы вы хотите произвести

сортировку, после чего Paradox выполнит ее на основе значений, находящихся в указанном поле Вы также можете по своему усмотрению отсортировать саму таблицу или создать новую таблицу с отсортированными данными, оставив оригинал без изменений.

Использование пункта меню Sort

Чтобы отсортировать таблицу, вы можете:

- Выбрать пункт Tools / Utilities / Sort и указать в диалоговом окне Select File таблицу, в которой вы хотите произвести сортировку записей
- Проинспектировать иконку таблицы в окне Folder и выбрать пункт Sort
- Любой из этих способов откроет диалоговое окно Sort Table (рис. 4.7).

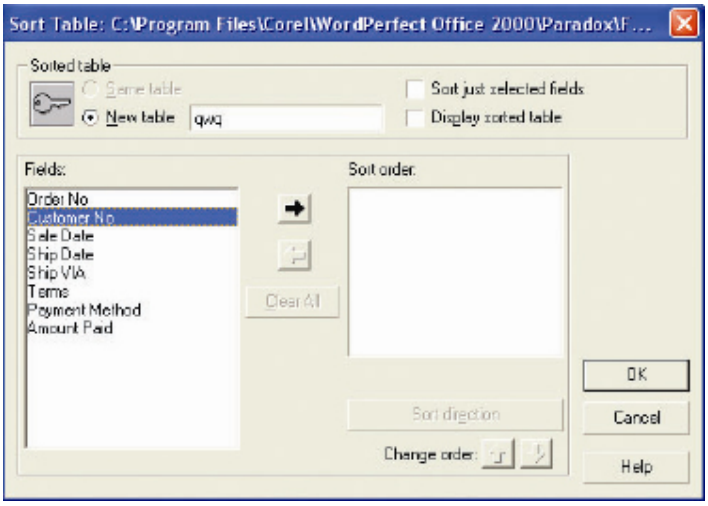


Рис 4.7 Диалоговое окно Sort Table

Определение порядка сортировки

Вы определяете порядок сортировки записей в таблице посредством выбора полей из списка Fields и добавлением их в список Sort Order. Когда Paradox выполняет операцию сортировки, он основывается на значениях первого поля, указанного в списке Sort Order, затем на значениях второго поля и так далее.

У вас нет необходимости помещать все поля из списка Fields в список Sort Order. Paradox добавит все поля, которые вы не указали явно, в конец списка Sort Order перед тем, как выполнить сортировку (если только вы не установили опцию Sort Just Selected Fields). В любом случае, Paradox включит все поля в результи-рующую таблицу - либо в оригинал либо в новую таблицу.

Учтите, если вы не включили ни одного поля в список Sort Order, Paradox от-сортирует таблицу в порядке полей, указанном в списке Fields. Если вы используете опцию Sort Just Selected Fields, вы обязаны поместить в список Sort Order по меньшей мере одно поле.

Существует несколько типов полей, которые нельзя сортировать. Paradox не может сортировать BLOB-поля в Paradox-таблицах, а также мемо- и логические поля в dBASE-таблицах. Эти типы полей отображаются в списке Fields, но недоступны для помещения их в список Sort Order.

Добавление полей в список Sort Order

Чтобы добавить поле в список Sort Order, вы должны выбрать его в списке Fields и нажать мышью кнопку-стрелку Add Field либо нажать Alt+A на клавиатуре. Поле немедленно появиться в списке Sort Order под выделенным в нем полем. Имя уже внесенного в список Sort Order поля останется в списке Fields, однако станет недоступным.

Для того, чтобы поместить поле в начало списка Sort Order, следует:

- Выбрать первое поле в списке Sort Order
- Добавить в список желаемое поле (оно станет вторым в списке)
- Щелкнуть мышью кнопку-стрелку вверх Change Order для перемещения поля в первую позицию

Для добавления в список Sort Order нескольких следующих друг за другом в списка Fields полей, необходимо нажать левую клавишу мыши, указывая курсором на начальное поле в группе, переместить курсор на конечное поле в группе и отпустить клавишу (или использовать на клавиатуре комбинацию Shift+стрелка вниз, чтобы отметить все необходимые поля). Затем нажмите кнопку-стрелку Add Fields или нажмите комбинацию Alt+A на клавиатуре, и выбранная вами группа полей будет помещена в список Sort Order. Если в группе окажутся поля, по которым нельзя производить сортировку, либо уже указанные в списке Sort Order, Paradox проигнорирует их.

Удаление отмеченных полей из списка Sort Order

Для удаления одного поля, отметьте его в списке Sort Order и используйте кнопку-стрелку Remove Field (или нажмите комбинацию Alt+R на клавиатуре). Поле вернется в список Fields. Для

удаления группы полей отметьте их и затем вы-полните те же действия. Кнопка-стрелка Remove Field является доступной только тогда, когда в списке Sort Order уже имеется одно или несколько полей.

Удаление всех полей из списка Sort Order

Для того, чтобы удалить все поля из списка Sort Order и сделать их снова доступными в списке Fields, нажмите .мышью кнопку команды Clear All (или нажмите комбинацию Alt+C на клавиатуре). Команда Clear All доступна, если в списке Sort Order имеется хотя бы одно поле.

Перестановка полей в списке Sort Order

Для того, чтобы передвинуть поле или группу полей в различные позиции списка Sort Order, отметьте нужное поле (поля), а затем нажмите кнопку-стрелку вверх Change Order (или кнопку-стрелку вниз соответственно).

Заметьте, что кнопки-стрелки Change Order доступны только тогда, когда в списке Sort Order присутствует два или более поля.

Установка порядка сортировки

Перед каждым полем в списке Sort Order специальный индикатор отображает порядок сортировки значений: 123... -означает по возрастанию, 321... - по убыванию. По умолчанию действует порядок сортировки по возрастанию значений.

Для изменения порядка сортировки поля необходимо дважды щелкнуть мышью индикатор или отметить поле и нажать кнопку Sort Direction.

Опция Same Table/New Table

Paradox предоставляет вам выбор способа сохранения отсортированных данных:

- Установите опцию Same Table, если вы хотите сохранить результаты сортировки непосредственно в самой сортируемой таблице. При этом необходимо помнить следующее:
    1. Опция Same Table доступна только в том случае, когда вы сортируете таблицу без ключа, поскольку сортировка таблицы, имеющей ключ, методом Same Table неизбежно приведет к конфликту первичных индексов.
    2. Перед проведением сортировки вы обязаны закрыть все окна, которые содержат данные из таблицы. Это относится к окну самой таблицы, а также ко всем формам, отчетам и другим документам, которые используют поля этой таблицы
- Установите опцию New Table, если вы хотите, чтобы результаты сортировки сохранились во вновь созданной таблице. Имя новой таблицы напечатайте в текстовом окошке New Table. При этом:
    1. Если вы введете имя уже существующей таблицы, Paradox предложит вам подтвердить замену данных в ней.
    2. Если вы замещаете данные в уже существующей таблице, вы обязаны закрыть все окна, которые содержат данные из этой таблицы, перед тем, как выполнить сортировку.

Опция Sort Just Selected Fields

Если включить опцию Sort Just Selected Fields, Paradox произведет сортировку только по тем полям, которые находятся в списке Sort Order. При этом в ре-зультирующую таблицу включаются все поля исходной таблицы, но они не сортируются, если не включены в список Sort Order.

Если вы установили опцию Sort Just Selected Fields и в таблице оказались две или более записи с идентичными значениями в сортируемом поле, Paradox оставит эти записи не отсортированными между собой.

Если вы не используете опцию Sort Just Selected Fields, Paradox выполнит сортировку сначала по полям из списка Sort Order, и затем (если существует две или более записи с одинаковыми значениями в сортируемых полях) по полям из списка Field List.

Опция Display Sorted Table

Если вы установили опцию Display Sorted Table и выполняете сортировку, Paradox открывает сортируемую таблицу после закрытия диалогового окна Sort Table.

Выполнение сортировки. Сортировка в сети

Когда вы закончите определять порядок сортировки, нажмите мышью кнопку ОК.

Если вы сортируете таблицы в многопользовательской среде, Paradox автоматически блокирует сортируемую таблицу. Это не позволяет другим пользователям модифицировать ее содержание или структуру. Если другой пользователь заблокировал таблицу, вы не можете начать ее сортировку до тех пор, она не будет разблокирована.

Если вы производите сортировку с помещением данных в новую таблицу, Paradox автоматически блокирует эту таблицу так же, как и таблицу-оригинал.

Использование форм

Используя формы Paradox, вы можете отображать данные из таблиц любым желаемым способом. Вы можете просматривать именно то, что вы хотите, и именно тем способом, который вам нужен.

Вызов формы

Для того чтобы открыть форму из Desktop, выберите пункт меню File / Open / Form, и вы увидите диалоговое окно Open Document, которое используется для выбора нужной формы. Из диалогового окна Open Document вы можете выбрать форму и открыть ее либо для просмотра данных либо в режиме разработки формы. Данный раздел рассматривает режим просмотра данных в форме.

Открытие формы текущей таблицы

Из окна Table вы можете открыть форму тремя способами:

1. Нажать мышью кнопку Quick Form на SpeedBar
2. Нажать F7 на клавиатуре.

Если вы не определили текущей формы для таблицы (см. раздел «Быстрый просмотр объектов», помещенный ранее в этой главе), Paradox откроет для вас по умолчанию стандартную форму.

Заметьте, если вы открыли форму (текущую для данной таблицы или использующуюся по умолчанию) из окна таблицы, Paradox сделает текущей ту запись в форме, которая до настоящего момента была активна в таблице. Например, если вы находились в таблице на записи номер 1, то форма откроется именно на записи номер 1.

Использование SpeedBar для окна Form

Просмотр таблицы

Вы в любой момент можете открыть таблицу, для которой построена форма. Если вы открыли форму из окна Table, выберите пункт меню Form / Table View для активизации окна таблицы (или его открытия, если оно было закрыто). Если вы открыли форму из Desktop, то выбрав Form / Table View (либо нажав мышью кнопку Table View на SpeedBar или клавишу F7 на клавиатуре) вы откроете исходную таблицу однотабличной формы или главную таблицу многотабличной формы.

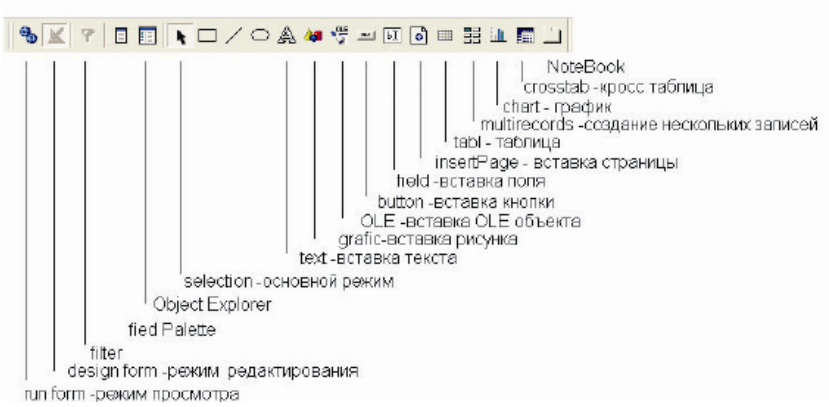


Рис.4.8 Использование SpeedBar для окна Form

Перемещение по полям формы

Вы можете перейти в любое поле, щелкнув его мышью. Если вы предпочитаете работать с клавиатурой, пользуйтесь клавишей Tab для перемещения по полям однотабличной формы. Клавиша Tab задаёт порядок перемещения слева направо и сверху вниз. Когда вы достигнете последнего (самого правого внизу) поля, вы можете нажать Tab для возвращения в первое (самое левое вверху) поле формы.

Клавиша Tab предназначена, в основном, для работы в простых формах . Если форма становит-

ся более сложной, с большим количеством объектов в ней, работа клавиши Tab приводит к ошибкам (однако, вы всегда можете использовать мышь для перемещения к любому объекту по вашему усмотрению). Использование комбинации Shift+TaЬ изменяет порядок перемещения при помощи клавиши Tab на противоположный - справа налево и снизу вверх.

Если вы выключите для какого-либо поля опцию Tab Stop, то вы не сможете использовать клавишу Tab или кнопки управления на SpeedBar для перехода к нему. Данное поле будет просто пропускаться. Для включения или выключения Tab Stop поля проинспектируйте его в окне Form Design.

Если вы работаете с многотабличной формой, вы можете использовать клавиши F3 и F4 для перехода от области главной (master) таблицы к детальным (de-tail) и обратно.

Перемещение по записям

Для осуществления перехода от записи к записи в форме вы можете:

1. Выбрать команды меню First (первая), Last (последняя), Next (следующая), Next Set (следующий набор), Previous Set (предыдущий набор)
2. Нажать мышью кнопки перемещения на SpeedBar
3. Нажать PgUp или PgDn на клавиатуре

Перемещение по страницам

Если вы используете многостраничную форму, выберите пункт меню Form / Page для перемещения по страницам: вы сможете быстро перейти к первой, последней, следующей или предыдущей странице. Если вы знаете номер необходимой вам страницы, выберите Form / Page / Go to для вызова диалогового окна Go To Page (рис. 4.14). Вы можете ввести номера страниц и нажать кнопку OK.

Иногда при переходе от поля к полю происходит мерцание экрана. Это особенно заметно, если форма имеет темный цвет фона. Для подавления этого эффекта выберите пункт меню Properties / Designer и включите опцию Flicker-Free Drawn. Включение опции Flicker-Free Drawn несколько уменьшит мерцание, однако на некоторых типах видеоадаптеров может привести к замедлению перемещения между полями. Попробуйте поработать с вашей формой при включенной и при выключенной опции Flicker-Free Drawn, и вы сможете определить приемлемый для вас режим.

Масштабирование формы

Выберите пункт View / Zoom, если хотите изменить масштаб формы. Коман-да Zoom in увеличивает масштаб окна Form. Вы можете его увеличить до 400% от первоначального размера вашего документа. Zoom out уменьшает картинку экрана. Вы можете сжать ее до 25% от исходного размера.

В дополнение к этому Paradox предоставляет вам выбор трех автоматических средств масштабирования:

- Fit Width - пропорционально изменяет размеры формы так, чтобы она полностью могла поместиться в окне по ширине.
- Fit Height - пропорционально изменяет размеры формы так, чтобы она полностью могла поместиться в окне по высоте.
- Best Fit - пропорционально изменяет размеры формы так, чтобы она полностью поместилась в окне и по высоте и по ширине. Эти условия продолжают действовать, когда вы изменяете размеры окна Form, при необходи-мости растягивая или сужая форму.

Сохранение настройки окна Form

Вы можете сохранить настройку окна Form, с которым работаете в настоящий момент, путем выбора пункта меню Properties / Form Option / Save Defaults. Paradox сохранит установки, перечисленные в меню Properties в качестве параметров, устанавливаемых по умолчанию для всех окон Form

Работа с данными в таблицах или формах

За некоторыми исключениями операции, описанные в данном разделе, работают одинаково и в таблицах и в формах.

Режим Field View (просмотр поля)

Когда вы перемещаетесь по полям таблицы или формы, Paradox выделяет цветом все поле полностью. Если же вы хотите установить текстовый курсор внутри поля, необходимо включить режим Field View. При этом Paradox поместит курсор в конец содержимого поля, и вы сможете перемещать его в пределах этого поля.

Вы можете войти в режим Field View следующими способами:

- Отметить поле и выбрать пункт меню Table / Field View или Form / Field View.
- Отметить поле и нажать кнопку Field View на SpeedBar,
- Отметить поле и нажать F2 на клавиатуре.
- Дважды щелкнуть мышью в любом месте нужного вам поля.

Вы можете выйти из режима Field View посредством команды Table / Field View, нажав иконку Field View на SpeedBar, отметив другое поле или нажатием клавиши F2. Кроме того, нажатие Enter, Tab или Alt+клавиша управления курсором даст вам возможность покинуть Field View и перейти к следующему полю.

Paradox имеет три режима просмотра полей:

- Field View - дает возможность перемещаться внутри поля от символа к символу.
- Persistent Field View - позволяет перемещаться к другому полю, не отменяя режим просмотра поля. Для включения данного режима нажмите Ctrl+F2.
- Memo View - режим просмотра мемо- и форматированных мемо-полей, ко-торый предоставляет вам некоторые возможности текстового редактора (например, можно ввести пустую строку , символ табуляции и т.д.). Данный режим включается нажатием Shift+F2.

Если при просмотре таблицы вы войдете в режим Field View, находясь в мемо-, форматированном мемо-, графическом или OLE-поле, Paradox покажет данное поле в отдельном окне. При просмотре формы, Paradox не будет показывать эти поля в отдельных окнах.

Выбор поля

Когда вы делаете какое-либо поле текущим (перемещаетесь к полю), Paradox выделяет его цветом. Это означает, что поле выбрано. Если вы напечатаете что-либо в выбранном поле, то замените его содержимое на те данные, которые вы ввели. Вы также можете выбрать несколько полей одновременно или только часть данных поля.

Учтите, что вы можете выбрать группу полей только в таблице, но не в форме. Несколько полей, находящихся в соседних строках и столбцах, можно выбрать, просто окружив их рамочкой

Чтобы отметить группу полей при помощи мыши, нажмите клавишу мыши над полем, с которого вы хотите начать выделение (не входя в режим Field View), и, перемещая мышь, раздвигайте границы прямоугольной рамочки вокруг необходимых вам полей. Курсор мыши при этом выглядит как две пересекающиеся двунаправленные стрелки.

Чтобы выделить группу полей при помощи клавиатуры, выберите поле, с ко-торого вы хотите начать выделение (не входя в Field View), и, удерживая в нажатом положении клавишу Shift, раздвигайте при помощи клавиш управления курсором границы рамочки так, чтобы необходимые вам поля оказались внутри нее.

Если вы хотите Отметить все поля в таблице, дайте команду Edit / Select All. Paradox нарисует рамку вокруг всей таблицы.

Копирование данных

Если вы хотите скопировать данные в Clipboard, выберите необходимые данные и дайте команду Edit / Copy. Помещенные в Clipboard данные можно перенести в другие поля или другие Windows-программы.

- Чтобы скопировать поле целиком, сделайте его текущим и выберите пункт меню Edit / Copy.
- Для копирования части поля, включите режим Field View и выберите данные, которые вы хотите копировать, затем дайте команду Edit / Copy.
- Находясь в окне Table, вы можете копировать сразу более, чем одно поле. Выбранные для копирования данные выделяются на экране рамкой.

- Чтобы скопировать целый столбец, дважды щелкните его заголовок, затем выберите пункт меню Edit /Copy.
- Для копирования целой строки, щелкните дважды номер записи, а затем выберите Edit / Copy (учтите, что если запись активна, то двойной щелчок приведет к включению режима Field View,).
- Для копирования группы полей следует дать команду Edit /Select All и затем Edit /Copy (это копирует всю таблицу в
- Clipboard) или выделить мышью необходимые поля и дать команду Edit / Copy.

Имейте в виду, что группу полей можно скопировать только из таблицы, но не из формы. Однако обратно в таблицу из Clipboard группу полей поместить нельзя. Вы можете, однако, вставить их в любую другую подходящую Windows - программу (например Quattro Pro for Windows).

Просмотр данных в различных порядках и в различных диапазонах

Предположим, что вы хотите просмотреть таблицу, имеющую ключ, в ином порядке, нежели в том, который задается первичным индексом. Для этого вы можете использовать вторичные индексы. Вы также можете задать диапазон значений индекса и сообщить Paradox, что вы хотите просмотреть записи, соответст-вующие поля которых находятся в определенном диапазоне.

Для просмотра таблицы в соответствии с вторичным индексом, выберите нужный индекс из списка Index List. Имя индексного поля появится на панели Field Values. Нажмите OK, и Paradox отобразит записи таблицы в новом порядке.

Учтите, что если вы хотите просмотреть данные в порядке возрастания зна-чения индекса, а не убывания, или с учетом верхнего регистра символов, вам придется заранее создать такой индекс.
Чтобы задать диапазон значений индекса, который вы хотите посмотреть, сначала выберите из списка нужный индекс. Если тип индексного поля допускает задание диапазона значений, то имя поля появится рядом с текстовым окошком в панели Field Values.

Индекс, выбранный вами, определяет порядок просмотра таблицы. Он группирует близкие по значению данные вместе, поэтому Paradox может найти их достаточно быстро. Когда вы задаете диапазон, вы сообщаете Paradox, какая группа значений необходима вам для просмотра.

Задание диапазона точным равенством

Вы можете задать диапазон точным равенством, то есть Paradox должен отобразить только те записи, значения индексов которых совпадают с определенной вами величиной.

Желаемую величину следует ввести в текстовое окошко панели .Field Values. Например, если у вас есть индекс поля Country в таблице Customer, и вы вводите «Canada» в качестве величины для сравнения, Paradox отобразит только те записи таблицы, в поле Country которых стоит «Canada».
Пример. Поиск записей по точному совпадению значения
Предположим, вы хотите увидеть только канадских заказчиков.

1. Откройте таблицу Customer.
2. Выберите пункт меню Table / Restructure и создайте вторичный
3. индекс поля Country.
4. Выберите пункт Table /Order/Range.
5. Выберите Country в качестве используемого для просмотра индекса и напечатайте «Canada» в текстовом окошке панели Field Values.
6. Нажмите OK.

Paradox покажет только те записи, у которых в поле Country аходится значение «Canada».

Задание диапазона значений

Если вы хотите увидеть все записи, попадающие в какой-либо диапазон, вы можете использовать диалоговое окно Set Range. Для задания диапазона значений одного из полей, установите щелчком мыши курсор в окошко рядом с нужным полем и нажмите кнопку Set Range.

Учтите, если вы нажмете кнопку Set Range без предварительной установки курсора в текстовое окошко, Paradox автоматически выберет последнее поле, для которого вы задали диапазон.

Когда вы щелкните кнопку Set Range, под текстовым окошком панели Field Values появится еще одно. В первое следует ввести верхнюю границу диапазона, во второе - нижнюю. Например, если у вас есть индекс для поля Sale Date таблицы Orders, и вы хотите увидеть все записи, относя-

щиеся к 1991 году, вы можете ввести 1/1/91 в верхнем текстовом окошке и 12/31/91 - в нижнем. Теперь при про-смотре таблицы в ней будут отражены только те записи, которые относятся к 1991 году.

Учтите, что Paradox при сравнении величин считает, что пустое значение совпадает с любым непустым, поэтому задание пустых значений в качестве границ диапазона допускается только в последнем поле составного индекса.

Для алфавитно-цифровых полей можно задавать диапазон путем сравнения начальной части поля с определенными значениями. Предположим, что вы разделили ответственность за контакты с клиентами среди своих служащих в алфавитном порядке. Один из служащих отвечает за работу с клиентами, чьи фамилии начинаются с букв от А до К. Для просмотра этого диапазона клиентов вы должны:

1. Выбрать индексное поле Name, чтобы отсортировать записи таблицы Customer в алфавитном порядке по фамилиям.Ввести букву А в верхнее текстовое окошко, чтобы сообщить Paradox, что вы хотите начать диапазон с фамилий на букву А.
2. Ввести букву К в нижнее текстовое окошко. Это сообщит Paradox, что вы хотите закончить диапазон фамилиями на букву К.
3. Включить опцию Match Partial Strings (она недоступна до тех пор, пока не установлена опция Set Range). Когда вы нажмете ОК., Paradox отобразит все записи с фамилиями тех клиентов, которые попали в заданный диапазон.

Диапазон значений составного индекса

Если вы выбрали из списка Index List составной индекс, все его поля появятся на панели Field Values (в порядке их следования в таблице).

При задании диапазона по составному индексу следует помнить:

Вы не обязаны задавать диапазон значений для каждого поля, однако вы не можете и пропустить поле. Например, если индекс состоит из трех полей, вы можете:

- Задать диапазон только для первого поля, но не для второго или третьего
- Задать диапазон для первого и второго поля

Однако, вы не можете:

- Задать диапазон для первого и третьего поля, пропустив второе

В случае составного индекса можно сочетать точное совпадение и диапазон значений, но диапазон можно задать только для одного последнего поля. В примере индекса из трех полей вы можете:

- Задать точное совпадение значений для первого и второго поля и диапазон для третьего
- Задать точное совпадение для первого поля, диапазон для второго
- Задать диапазон значений только для первого поля

Но Вы не можете:

- Задать диапазон значений для первого поля и точные совпадения для второго и третьего полей

Предположим, что вы сделали необходимые установки в окне Order/Range для таблицы в окне Table, а затем нажали кнопку Quick Form на SpeedBar, чтобы открыть текущую форму таблицы. Даже если вы ранее установили другой порядок просмотра данных в этой форме, Paradox будет использовать табличные установки в обоих окнах, потому что таблица была открыта первой. Аналогично, если вы первой откроете форму, а затем нажмёте кнопку Table View на SpeedBar для открытия окна Table, таблица будет использовать установки формы. Иными словами. Paradox использует установки порядка просмотра данных, определённые для окна, открытого первым.

Вы можете сохранить установки порядка и диапазона просмотра вместе с формой (однако, нельзя сохранить их вместе с таблицей). Для этого просто задайте желаемые установки, а затем сохраните форму либо из окна Form либо из окна Form Design

Использование окна Order / Range в dBASE-таблицах

Когда вы открываете диалоговое окно Order/Range для dBASE-таблиц. список Index List показывает все имена (tags) поддерживаемых (maintained) индексов (хранящихся в файле с расширением MDX и именем таким же. как и имя таблицы. Выберите имя индекса и используйте диалоговое окно так же, как и в случае Paradox-таблиц.

Установите опцию NO INDEX, если вы хотите просмотреть таблицу в неупорядоченном виде.

Если вы хотите использовать другой индекс (файл с расширением .NDX или индекс из другого .MDX-файла), введите его имя файла (включая расширение) в текстовое окошко Select dBASE Index File, и далее пользуйтесь им, как и любым другим индексом.

Когда вы изменяете порядок просмотра dBASE-таблицы, номера записей, которые показывают физическое расположение каждой записи в таблице, выводятся не по порядку.

Учтите, что нельзя использовать составной индекс для dBASE-таблиц, чтобы задать диапазон отображаемых записей. Однако, вы можете использовать ин-дексные выражения.

Кроме того, если вы использовали окно Order/Range для dBASE-таблиц, маркер вертикальной прокрутки будет всегда находится посередине линейки. Вы можете его передвигать, чтобы перемещаться по таблице, но после каждой такой операции он будет возвращаться в исходное положение.

Просмотр удаленных записей в dBASE-таблицах

Когда вы удаляете запись из dBASE -таблицы, на самом деле Paradox не стирает ее, он просто скрывает ее или отмечает как удаленную (в зависимости от вашего желания). Если вы хотите просмотреть удаленные данные, нужно дать команду Table / Show Deleted или Form / Show Deleted.

В окне Table Paradox отображает удаленные записи, помечая их квадратом в столбце номера записи. Если вы просматриваете удалённые записи в окне Form, в поле системных сообщений (status bar) вслед за именем таблицы появляется надпись «Record deleted».

Вы можете восстановить удалённые dBASE-записи. Для этого воспользуйтесь командой Records / Undelete или нажмите Ctrl+Del, если находитесь в поле удаленной записи в режиме Edit с включенной опцией Show Deleted.

Опция Show Deleted и команда Undelete доступны только в dBASE - таблицах. Когда вы удаляете запись из Paradox - таблицы, она удаляется необратимо и не может быть восстановлена. Для необратимого удаления dBASE-записей, войдите в режим изменения структуры таблицы (см. Главу 3) и включите опцию Pack Table.

Обратите внимание, что линейка вертикальной прокрутки «не замечает», что часть записей «удалена» даже при выключенной опции Show Deleted.

Поиск информации

Для поиска записей, полей и значений в таблицах и формах используются команды Locate меню Records.

Поиск полей

Дайте команду Record /Locate/ Field для перемещения к конкретному полю таблицы. Вы увидите диалоговое окно Locate Field.

Список Fields показывает все поля таблицы. Выберите нужное и нажмите OK (кнопка OK недоступна, пока вы не укажете поле).

Такой способ перемещения особенно полезен, если вы работаете с большой таблицей, состоящей из множества полей. Иногда он работает быстрее и точнее, нежели использование линейки прокрутки или многократные нажатия клавиши Tab. Когда вы впервые используете диалоговое окно Locate Field, Paradox помечает текущее поле таблицы. При последующих использованиях Paradox по умолчанию предлагает поле, которое вы искали в прошлый раз.

Поиск записей по значению

Для поиска записей с конкретным значением в каком-либо поле выберите пункт меню Record /Locate Value. Вы увидите диалоговое окно Locate Value (рис. 4,9.).

Выберите из списка Fields то поле, которое должно содержать искомую величину, и введите ее в текстовое окошко Value. После того, как нажмете OK, Paradox переместит вас к первому вхождению искомой величины в таблице. Выберите Record / Locate Next для поиска следующего вхождения. Если вы нашли все величины, Paradox отобразит сообщение на экране в центре строки системных сообщений.

Включите Case Sensitive, если вы хотите, чтобы Paradox производил поиск с учетом верхнего и нижнего регистра напи-

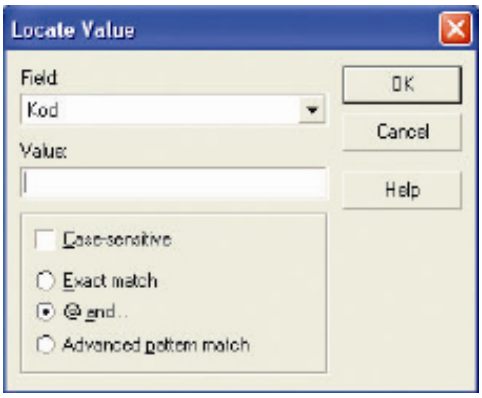


Рис. 4.9 Диалоговое окно Locate Value

сания букв.

Пример. Поиск записей по значению

Предположим, что вы хотите найти все записи, касающиеся канадских клиентов.

1. Откройте таблицу Customer.
2. Нажмите кнопку Locate Field Value на SpeedBar.
3. Выберите поле Country из списка в панели Fields.
4. Введите Canada в текстовое окошко Value .
5. Включите опцию Exact Match.
6. Нажмите ОК. Paradox найдет первую запись со значением Canada в поле Country.
7. Для поиска следующей такой записи, щелкните кнопку Locate Next на Speed-Bar.

Поиск по простым шаблонам

Вы можете производить поиск значений по несложным шаблонам. По умолчанию Paradox допускает использование символов @ и .. в качестве универсальных операторов для составления шаблонов.

- @ означает любой одиночный символ
- .. означает любое значение

Если вы не хотите пользоваться шаблоном, включите опцию Exact Match, и Paradox будет искать только те значения, которые точно совпадают с тем, которое вы ввели в текстовое окошко Value.

Сравнение по расширенным шаблонам

Если вы хотите использовать для построения шаблонов расширенный набор специальных операторов, включите опцию Advanced Pattern Match. Список расширенного набора операторов приведен в Таблице 4-1.

Таблица 4.1 Универсальные операторы, применяемые для построения шаблонов

Оператор	Действие
^	начало поля
$	конец поля
*	любое количество повторений символа, стоящего перед * или его отсутствие
+	одно или более повторение символа, стоящего перед +
?	одно или ни одного повторения символа, стоящего перед ?
|	сравнивает символы как перед так и после 1
[ ]	сравнивает все символы внутри []
[^]	сравнивает все символы вне []
( )	группирует содержащиеся символы
\	использует следующий универсальный оператор как обычный символ
\r	возврат каретки
\n	перевод строки
\t	табулятор
\f	перевод страницы (прогон)

Вы можете комбинировать перечисленные операторы для создания сложных критериев поиска.

Таблица 4.2 Примеры шаблонов

Шаблон	Результат поиска
..red..	Red paper, Color of red,
Red Glass	
red..	Red paper, Red Glass
^red..	Red paper, Red Glass
^red	Red paper, Red Glass
^red|^col	Red paper, Color of red, Red Glass
^(red | col)	Red paper, Color of red, Red Glass

abc	abc
abcabc+	abc, abcc, abccc (любое количество c)
*	ab, abc, abcc, abccc (нет c или любое их кол-во)
abc?	ab, abc (нет c или одно c)
[abc]	a, Ь, или c
[^abc]	все что исключает a, b или c
[a-z]	все символы в диапазоне a-z
[]/*^	символы],/,\,* или «
(abc)	abc
(abc)+	abc, abcabc, abcabcabc
Al(an)	или a или an
c..k	самая длинная в пределах возможного строка, которая начинается с (c) и заканчивается на (к)
c[^]k	одно слово, которое начинается с (c) и заканчивает-ся на(К)

Изменение способа отображения данных

Вы можете изменить вид данных в таблице или в форме путем изменения свойств полей. Набор свойств в меню поля различается в зависимости от типа поля. Свойства алфавитно-цифрового поля отличны от свойств числового поля, свойства числового отличны от свойств поля даты и т. д.

Изменение свойств поля не влияет на сами данные или на способ их хранения. Это просто еще одно мощное и гибкое средство просмотра данных.

Изменение формата чисел

Формат вывода на экран чисел можно изменить, проинспектировав числовое поле и выбрав пункт меню Number Format. Инспектировать поля можно либо в окне Table либо в окне Form Design. При этом на экране появляется меню возмож-ных числовых форматов (использующийся в данный момент формат помечен галочкой).

Чтобы изменить формат инспектируемого поля выберите один из предлагаемых пунктов меню.

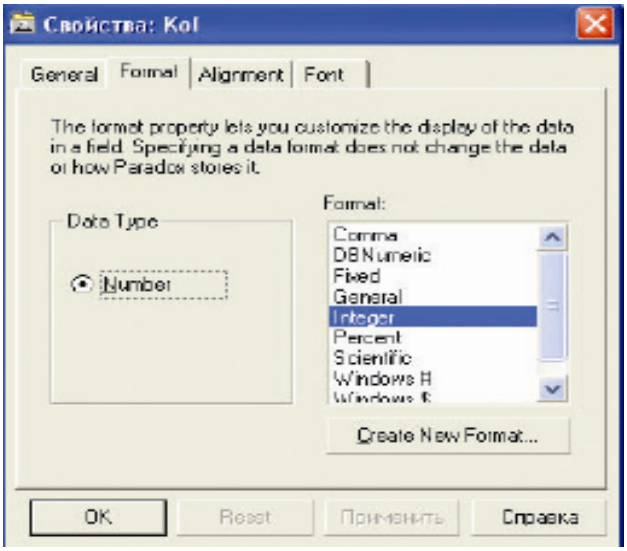


Рис.4.10 Меню Number Format

- Windows # является форматом для числовых полей Paradox по умолчанию. Paradox использует формат, который вы определите из Control Panel системы Windows.
- Windows $ использует символ валюты и формат, который вы определили в Control Panel.
- Fixed отображает числа с двумя десятичными знаками. Нули после десятичной точки отображаются всегда, а разделители периодов (тысяч, миллионов и т.д.) не используются. Перед отрицательными числами стоит знак минус (-).
- Scientific отображает числа в экспонентной форме (с двумя десятичными знаками после запятой), т.е. в виде числа от 1 до 10, помноженного на степень числа 10. Отрицательным числам предшествует знак минус (-). Любой формат использует экспонентную форму для отображения очень больших чисел. Формат Scientific использует экспонентную форму всегда.
- General отображает числа, имеющие до двух десятичных знаков, если число имеет десятичную точку. Нули после запятой и разделители периодов не отображаются. Отрицательным числам предшествует минус (-).
- Comma отображает число с двумя десятичными знаками. Нули после десятичной точки отображаются. Используется разделитель периодов. Отрицательные числа отображаются в скобках.

- Percent отображает числа с последующим знаком процента (%). Например, число .5 отображается как 50%. Разделитель порядков не используется. Отрицательным числам предшествует знак минус (•).
- Integer отображает только целые числа. Знаки после десятичной точки округляются, после того, как вы переходите в формат Integer. Если вы переходите в формат, который отображает десятичные знаки, они вновь отображаются. Разделитель порядков не используется. Отрицательным числам предшествует знак минус(-).

Вы можете задать свой собственный формат чисел посредством выбора мно-готочия (...) в верхней части меню числовых форматов. Вы увидите диалоговое окно Select Number Format. Придумайте название своему собственному формату и воспользуйтесь многочисленными опциями диалогового окна для его разработки.

Учтите, что вы можете создавать, удалять или модифицировать только вами разработанные форматы, но не форматы, предлагаемые Paradox.

Чтобы разработать свой собственный формат, нажмите мышью кнопку ко-манды Create. Имя диалогового окна изменится на Create Number Format и на панели Format появятся доступные опции.

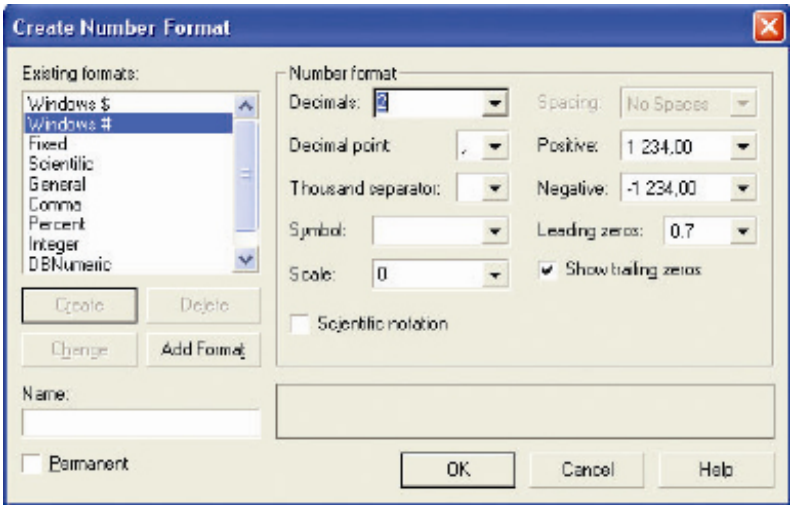


Рис.4.11 диалоговое окно Select Number Format

В текстовое окошко Name введите название формата и с помощью раскрывающихся списков панели Format задайте его свойства.

Если вы хотите использовать стандартный формат, определяемый в Control Panel, для задания значений какой – либо опции, вы можете проинспектировать текстовое окошко нужной опции. Вам будет представлено меню значений по умолчанию. Это поможет ускорить разработку вашего собственного формата.

Не забудьте, что форматы чисел и валют задаются в Control Panel системы Windows.

Для задания формата чисел используются следующие опции:

- Decimal определяет количество отображаемых десятичных знаков. Вы имеете возможность вывести до 15 десятичных знаков.
- Decimal Point определяет символ точки или запятой в качестве разделителя десятичных знаков.
- Thousand Separator позволяет определить разделитель периодов (обычно группа по три цифры), а также выбрать символ точки (.) или запятой (,) в качестве разделителя.
- Symbol определяет символы, отображаемые вместе с числом. Набор доступных символов включает в себя $, inch, Ib, kg, cm, mi и DM. Вы также можете задать свой собственный символ, введя его в текстовое окошко Symbol.
- Spacing определяет количество пробелов между числом и предшествующим символом. Вы можете указать, что хотите установить пробелы между символом и числом для всех положительных чисел, для всех отрицательных чисел или для всех чисел вообще. Заметьте, что опция Spacing недоступна до тех пор, пока вы не выберите символ опцией Symbol.
- Positive определяет использование знака плюс (+} перед положительными числами. Вы можете выбрать один из нескольких вариантов.Negative задает способ выделения отрицательных чисел знаком минус (-) или круглыми скобками. Раскрывающийся список содержит несколько вариантов.
- Leading Zeros задает количество цифр до десятичного разделителя. Например, если вы вводите число 123 в поле, которое имеет значение опции Leading Zeros 4 , Paradox отобразит число 0123 , как только вы покинете по-ле. Если вы введете число 01234 в поле, которое имеет значение опции Leading Zeros менее 5 , Paradox отобразит число 1234. Если же установить значение опции Leading

Zeros равное 5 , то Paradox дополнит число нулями слева до пятизначного поля. Эта опция полезна, если вы собираетесь хранить почтовые zip - код в числовом поле.

- Scale позволяет умножать числа на степень десяти. Если, например, вы введете 3 в текстовое окошко Scale, то в панели примеров появится образец числа, умноженного на 10 в степени 3. Вы также можете ввести отрицательную величину для того, чтобы производить деление на 10 в нужной степени.
- Scientific Notation указывает, что числа следует выводить в экспонентной форме.
- Show Trailing Zeros указывает на необходимость выводить нули в конце числа. Это означает, что числа, не имеющие десятичных знаков, будут отображаться с таким количеством нулей после десятичной точки, которое задано опцией Decimal.

Когда вы определили формат и присвоили ему имя, нажмите мышью кнопку Add Format для того, чтобы добавить его к списку Existing Formats. Теперь имя формата будет появляться в меню Number Format при инспектировании поля.

Если включить опцию Permanent, ваш формат сохранится в файле PDXWIN.INI . В противном случае он будет существовать лишь в течение сеанса работы с Paradox.

Учтите, что собственный формат можно разработать для полей различных типов, но каждый формат должен иметь свое уникальное имя независимо от типа поля. Удалить любой разработанный вами формат можно, выбрав его из списка Existing Formats и нажав мышью кнопку Delete.

Чтобы модифицировать разработанный вами формат, следует выбрать его из списка Existing Formats и нажать мышью кнопку Change. Сделав необходимые изменения, нажмите кнопку Accept, и они будут сохранены.

Изменение формата денежного поля

Так же, как и числовой, формат денежного поля можно изменить, выбрав в меню Number Formats один из предлагаемых стандартных форматов либо разработав свой собственный при помощи диалогового окна Select Number Format.

Изменение формата дат

Поля типа дата имеют свойство Date Format. Выберите его в меню, чтобы изменить способ представления дат.

Вам будет предложен список возможных форматов для выбора нужного вам. Выберите его для изменения способа представления дат Paradox.

- Windows Short - краткий формат дат, который вы определяете в диалоговом окне International в Control Panel системы WinowsParadox for Windows
- Windows Long - полный формат дат, который вы определяете в диалоговом окне International в Control Panel системы Windows.
- m/dd/yy отображает даты в формате, который использует по две цифры для числа, месяца и года, разделяя их знаком (/).

Для каждого из этих форматов два знака yy предполагают, что год относится к текущему столетию.

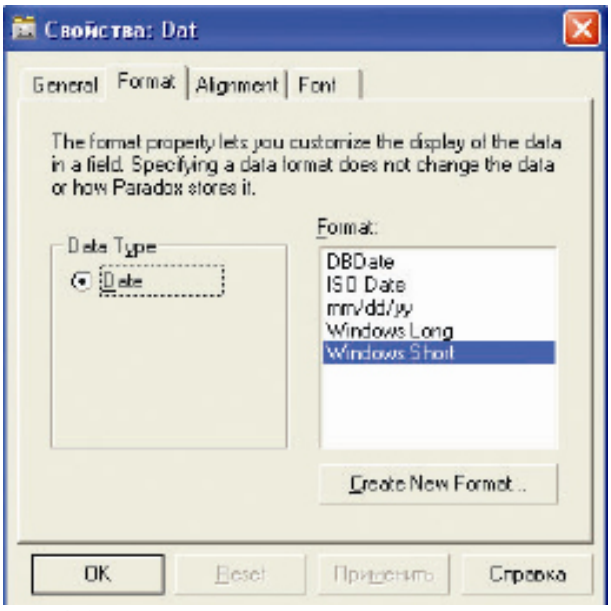


Рис.4.12 Меню Date Format

Чтобы иметь возможность использовать даты, которые вы-ходят за пределы двадцатого столетия, вам придётся определить в формате все цифры года.

Вы можете разработать свои собственные форматы дат, войдя в пункт (...) в верхней части меню формата дат.

- Day определяет, хотите ли вы отображать число с нулем перед ним или без нуля.
- Month задает способ отображения месяца либо его названием либо его номером.
- Year задает способ отображения года либо четырьмя цифрами либо только двумя последними.
- Order определяет порядок следования компонентов даты: дня недели (%H), числа (%D), месяца

(%M), года (%Y). Если вы не хотите отображать какой-либо из компонентов, удалите его оператор.

- Case определяет, словами или числами отображать месяц и день недели. Вы можете выбрать следующие варианты:
- Lower - слова печатаются маленькими буквами
- Upper - слова печатаются заглавными буквами
- Mixed - первые буквы слов заглавные, остальные – маленькие

Если вы определили формат и присвоили ему нажмите мышью кнопку Add Format, чтобы включить его в список Existing Formats.

Изменение формата времени

В формах вы можете помещать специальные которое содержат данные о самой форме или таблице, которой была создана форма. Например, поле Now отображает в форме текущее время. Вы можете изменить формат представления времени инспектированием поля Now и выбором из его меню пункта Format / Time Format. Вы получите меню доступных форматов отображения времени. Выберите из него нужный вам.

- Windows Time использует формат времени, определенный в диалоговом окне International в Control Panel системы Windows.
- hh:mm:ss am выводит по две цифры для отображения часов, минут и секунд, разделяя их двоеточием, и добавляет «am» (до полудня) и «pm» (после полудня).

Чтобы разработать свой собственный формат представления времени, выберите пункт (...) в верхней части меню форматов. Вы увидите диалоговое окно Select Time Format.

Для создания собственного формата, щелкните мышью кнопку Create. Имя диалогового окна изменится на Create Time Format, и панель Format отобразит доступные опции для создания желаемого формата.

Опции формата представления времени:

- Hour, Minute или Second под надписью Leading Zero определяют, следует ли ставить 0 перед числом из одного знака.
- 12 Hour или 24 Hour определяют 12- или 24-часовой форматы соответственно. Если вы установили 12 Hour, вы можете указать, чтобы Paradox отображал время с AM и PM.
- Order определяет последовательность отображения часов (%H), минут (%M), секунд (%S) и индикатора am/pm (%N). Удалите соответствующий оператор, если вы не хотите выводить на экран какой-либо из компонентов.

Когда вы определите формат времени и присвойте ему имя, нажмите мышью кнопку Add Format для внесения его в список Existing Formats.

Если вы установили Permanent, то Paradox сохранит определённый вами формат времени в файле инициализации PDXWIN.INI Paradox, иначе ваш формат будет доступен только в текущем сеансе работы с Paradox.

Задание комплексного (timestamp) формата времени

Paradox позволяет размещать в формах неопределенные (undefined) поля, с помощью которых можно отображать значения переменных типа время в комплексном (timestamp) формате, включающего в себя дату и время суток. Данный формат можно изменить, проинспектировав соответствующее поле в окне Form Design и выбрав пункт Format /Timestamp Format. Вы увидите меню доступных форматов.

- Win DateStamp использует форматы дат и времени, которые вы определили в Control Panel / International. h:m:s am m/d/y отображает часы, минуты и секунды, разделённые двоеточием, за ними am или pm и месяц, день и год.

Вы можете определить свой собственный timestamp формат посредством выбора пункта (...) в верхней части меню. Вы увидите диалоговое окно Select Timestamp Format.

Для создания собственного timestamp формата нажмите кнопку Create. Имя диалогового окна изменится на Create Timestamp Format и панели Date Format и Time Formal отобразят доступные опции для создания формата. Процесс разработки формата аналогичен рассмотренному выше.

Изменение логического формата

Объекты форм типа неопределеное (undefined) поле и логические dBASE-поля имеют свойство Logical Format, который дает возможность выбора различных представлений значений «истина» и «ложь». Проинспектировав поле, вы получите список форматов логических величин, таких как True/False (Истина/Ложь) или Male/Female (Мужской/Женский)). Чтобы задать свой собственный формат, следует выбрать пункт (...). Процесс разработки нового формата при помощи диалоговых окон Select Logical Format и Create Logical Format аналогичен рассмотренным выше.

Для удаления созданного вами формата выберите его из списка Existing For-mats и нажать кнопку Delete. Вы можете удалить только те форматы, которые создали сами. Кнопка Delete недоступна, когда вы выбираете существующий формат, который не может быть удален.

Для изменения созданного вами формата выберите его из списка Existing Formats и нажмите кнопку Change. Paradox отобразит параметры формата в панели Format Произведите желаемые изменения, а затем нажмите Accept для сохранения изменений и возвращения формата обратно в список Existing Format.

Управление выводом на экран memo- и форматированных memo-полей

Paradox хранит memo- и форматированные memo-поля в специальных файлах (с расширением .MB для Paradox-таблиц и с расширением .DBF для dBASE-таблиц), а не в самих таблицах. Таблица содержит только часть поля (она для Paradox-таблиц определяется в диалоговом окне Create Table) и ссылку на .MB-файл или .DBF-файл. Paradox ищет содержимое этих полей каждый раз, когда их нужно отобразить на экране.

Поэтому, в зависимости от быстродействия вашего компьютера и размера memo-полей вывод на экран может оказаться значительно замедлен. Для повышения скорости работы Paradox предоставляет вам возможность избегать появления на экране mem-полей до тех пор, пока вы действительно не захотите их увидеть. Для управления выводом на экран memo- и форматированных memo-полей используйте их свойство Complete Display.

Если вы работаете в окне Table, проинспектируйте memo-поле (или форматированное memo-поле) и включите опцию Complete Display, чтобы полностью видеть все поля записей, либо выключите ее, чтобы видеть memo-поле целиком только в том случае, если оно является текущим.

Если вы установили опцию Complete Display, вы будете видеть полное содержание всех memo-полей всех видимых на экране записей.

Если вы отключите опцию Complete Display для Paradox-таблицы, то Paradox будет отображать только те части memo-полей, которые хранятся непосредственно в самой таблице (без содержимого .MB-файла). Выводимая на экран часть memo-поля будет заканчиваться многоточием (...). Чтобы увидеть memo-поле целиком, необходимо сделать его текущим.

Если вы отключите опцию Complete Display для dBASE-таблицы, то наличие данных в memo-поле будет обозначаться специальным маркером. Чтобы увидеть memo-поле целиком, необходимо сделать его текущим.

Для более быстрого перемещения по записям таблицы рекомендуется выключить опцию Complete Display.

Если вы работаете в окне Form, за вывод на экран memo-полей отвечает свойство Run Time\ Complete Display. Его можно изменить только в окне Form Design.

Управление выводом графических и OLE полей

В дополнение к свойствам Alignment, Color, Font и Complete Display, графические и OLE-поля имеют еще одно -масштабирование (Magnification).

Используя масштабирование вы можете сжимать изображение графического или OLE-объекта до 25% или 50% от его оригинального размера или увеличивать его до 200% или 400%. По умолчанию графические и OLE-объекты изображаются в натуральную величину.

Одним из пунктов меню масштабирования является опция Best Fit, которая сжимает график или OLE-объект до размера поля, сохраняя при этом его пропорции.

Для повышения скорости вывода на экран изображать графические и OLE-объекты следует в натуральную величину.

При работе с формами свойство масштабирования (Magnification) доступно только в окне Form Design.

Предварительный просмотр отчета

Обычно, отчётами в Paradox считаются документы, предназначенные для печати, а на экране данные просматриваются в виде таблиц или форм. Однако, иногда бывает нужно просмотреть отчет на экране перед тем, как его распечатать на принтере. Для этого в Paradox имеется специальное окно Report

Заметьте, что в отчете вы не можете осуществлять ввод данных или их редактирование. Окно Report является исключительно средством просмотра.

В данном разделе обсуждается только просмотр данных в окне Report. Сам отчет вы можете настроить необходимым образом в окне Report Design.

Окно Report

Чтобы открыть окно Report для просмотра текущего для данной таблицы от-чета, щелкните кнопку Quick Report на SpeedBar или выберите пункт меню Table / Quick Report. Если вы не определили для данной таблицы текущий отчет, Paradox выведет на экран по умолчанию стандартный отчет.

Вы, также можете произвести предварительный просмотр разработанного вами ранее отчёта, посредством выбора пункта меню File / Open / Report. На экране появится диалоговое окно Open Document, описанное ранее в Главе 2.

Выберите из списка нужный вам отчет и включите опцию View Data на панели Open Mode. После того, как вы нажмете кнопку OK, в окне Report появится отчет.

Для просмотра окна Report пользуйтесь линейками прокрутки. Для перемещения к другим страницам используйте кнопки SpeedBar или меню Page.

Так как в окне Report вы можете только просматривать или распечатывать ваши данные, число доступных команд меню ограничено:

- Пункты File и Window доступны во всех окнах Paradox.
- С помощью меню Report вы можете перейти в окно Report Design и распечатать отчет, используя команду Print.
- Пункт Раде поможет вам переместиться к нужной странице отчете.
- Меню Properties используется для вызова диалогового окна Desktop Properties, рассмотренного в Главе 2, команд Zoom (распахнуть окно) и Save Properties (сохранить свойства).

Использование SpeedBar в окне Report

Как и в меню, ваш выбор на SpeedBar ограничен. Design - переход к конструированию отчета.

- Print - распечатка отчета.
- First page, Next page, Previous page, Last page переход к первой, следующей, предыдущей и последней странице отчета соответственно.
- Go to page переход к заданной странице отчета.

Использование фильтра

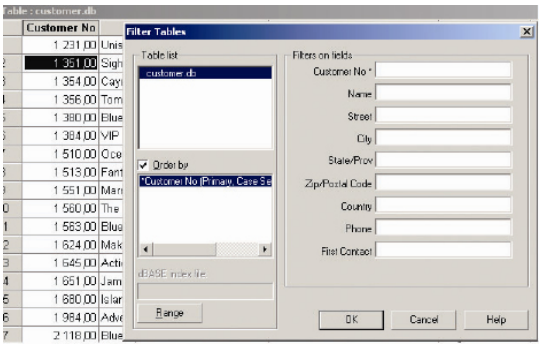


Рис.4.13 Окно фильтра, вызванного из SpeedBar

рис.4.14 вызов фильтра для конкретного поля

Часто при работе с таблицами для наглядности надо выделить группу записей, которые останутся на экране. В этих целях используют фильтр. Его можно установить двумя путями. Первый использует SpeedBar. После вызова фильтра (кнопка с воронкой) на экране появится окно, изображённое на рис.4.13

В правой части показаны наименования всех полей таблицы. Фильтр можно установить по любому количеству полей. Если надо установить диапазон данных для какого-либо поля, достаточно через запятую указать верхнее и нижнее значе-ние. Например, для поля Customer No записали: >1354,<1560

и в результате на экране будут записи, удовлетворяющие этому условию.

Второй способ позволяет вызвать фильтр не только при наличии на экране SpeedBar, но и в реальной работе, где на экране только форма (режим Dialog box) и больше ничего нет. В этом случае надо стать на поле, по которому устанавливается фильтр и нажать правую кнопку мыши (результат на рис.4.14).

При выборе «Filter…» появится окно, куда и надо ввести фильтр. Для отмены фильтра в обеих случаях надо удалить его значение.

Глава 5. Ввод и редактирование данных

В данной главе рассматриваются режим редактирования данных (Edit mode), с помощью которого вы сможете выполнять следующие операции:

- Вставлять и удалять записи
- Работать в режимах просмотра поля Field View, Persistent Field View и Memo View
- Вырезать, копировать и вставлять данные из Clipboard и других файлов
- Отменять сделанные изменения
- Производить поиск данных в режиме Find and Replace
- Вводить графики и OLE-объекты в ваши таблицы, редактировать специальные поля (такие как memo, форматированные memo) и поля с контролем правильности данных (validity checks)
- Блокировать записи
- Использовать при вводе проверку данных по таблице-справочнику

Включение режима редактирования

Вы можете редактировать данные, открыв таблицу или форму и войдя в режим редактирования. Для этого:

1. Откройте таблицу или форму посредством выбора пункта File /Open, как описывалось в Главе 4.
2. Включите режим редактирования, используя один из следующих способов:
   - Выберите пункт меню Table / Edit Data или Form / Edit Data, uли нажмите кнопку Edit Data на SpeedBar
   - Нажмите клавишу F9 Внесите необходимые изменения.Выйдите из режима Edit, используя кнопку Edit Data на SpeedBar, команду Table / End Edit или клавишу F9.

Когда вы войдёте в режим редактирования, кнопка Edit Data на SpeedBar будет нажата, а в строке системных сообщений (status bar) появится соответствующая надпись.

Таблица В. 1 Клавиши, используемые при вводе и редактировании данных

Клавиша	**Действие**
Ctrl+Backspase	удаляет слово слева от курсора ввода или текущую отмеченную область
Ctrl+D	копирует в текущее поле содержимое соответствующего поля предыдущей записи
Esc	отмена сделанных при редактировании изменений содержимого поля
Alt+Backspace	отмена сделанных при редактировании изменений всех полей записи
Spacebar (Пробел)	вводит текущую дату в поле дат (если дата состоит из нескольких компонентов, необходимо нажать пробел несколько раз!
Двойной щелчок мышью	в режиме Field View, отмечает слово в алфавитно-цифровом, memo, форматированном memo, (или dBASE-символьном или memo) поле
Home	перемещает в самое левое поле текущей записи
Ctrl+Home	перемещает в самое левое поле первой записи таблицы.
End	перемещает в самое правое поле текущей записи
Ctrl-fEnd	перемещает в самое правое поле последней записи таблицы

Если вы находитесь в режиме редактирования, вы можете выбрать (сделать текущим) любое поле таблицы и начать вводить данные (этим вы полностью перезапишете содержимое поля).Заметьте, что ввод данных в memo, форматированные memo, графические и OLE-поля может отличаться от ввода данных в поля других типов. Эти типы полей рассматриваются в разделе «Ввод специальных типов данных» данной главы.

Если вам необходимо установить курсор в какую-либо конкретную точку поля (например, чтобы исправить орфографическую ошибку или опечатку), вы должны войти в режим просмотра поля

Field View (он рассматривается в разделе «Использование режима Field View»).

Большинство операций, рассматриваемых в данной главе, применимы как к таблицам так и к формам. Однако, некоторые различия все же существуют

Вставка и удаление записей

Вы можете вставлять новые записи или удалять существующие, работая как с таблицей, так и с формой. Выберите пункт меню Record / Insert (или нажмите Ins), если хотите вставить пустую запись перед текущей.

Если таблица имеет ключ, вы можете вводить данные в любом месте, а Paradox потом автоматически переместит новую запись в нужное место. Если это место находится вне экрана, вам может показаться, что запись исчезла. Однако, если вы посмотрите на счетчик записей в строке системной информации (status bar), то убедитесь, что запись действительно добавлена. Пока Paradox не сохра-нит новую запись, текущей остается та же запись, которая была до нажатия клавиши Ins.

Запись, вставленная в таблицу, не имеющую ключей, будет сохранена в том месте, где вы ее вводили Во время работы в форме, содержащей одну запись, вставка новой записи выглядит, как очистка экрана. Когда вы нажимаете клавишу Ins или выбираете Record / Insert, запись на экране не будет содержать никаких данных. Это происходит потому, что Paradox сразу перемещает вас к только что вставленной записи. Помните, что Paradox всегда вставляет новую запись перед отмеченной текущей записью.

Для удаления текущей записи таблицы или формы следует выбрать пункт меню Record / Delete или нажать комбинацию Ctrl+Del. Помните, что в Paradox-таблице вы не можете восстановить удалённую запись, так что перед удалением еще раз удостоверьтесь в том, что вы действительно хотите удалить именно эту запись.

При работе с dBASE-таблицей физического удаление данных не происходит, и вы даже можете воспользоваться опцией Show Delete для просмотра уда-ленных записей.

Использование режима Field View

При вводе данных Paradox полностью заменяет содержимое поля. Например, когда вы перемещаетесь в поле, содержащее данные, Paradox помечает все данные в поле, если вы начинаете вводить информацию в поле, они теряются. Для того, чтобы перемещать текстовый курсор внутри поля, выделять фрагменты поля, добавлять в него информацию или редактировать, существует режим просмотра поля Field View.

Paradox имеет три типа режима Field View:

1. Field View
2. Persistent Field View
3. Memo View

Режим Field View

Когда вы входите в режим Field View, Paradox располагает тексто-вый курсор внутри текущего поля. Для перемещения по полю используются клавиши управления курсором. Чтобы включить Field View, переместитесь к нужному полю и выполните одно из следующих действий:

- Выберите пункт меню Table / Field View или Form / Field View
- Нажмите кнопку Field View на SpeedBar
- Нажмите клавишу F2
- Дважды щелкните мышью на неотмеченном поле
- Курсор расположится в той точке поля, где он находился курсор мыши в момент нажатия клавиши мыши (если поле уже отмечено, вы должны нажать кнопку мыши только один раз).

Если режим Field View не включен, вы можете использовать клавиатуру для перемещения в пределах таблицы (от столбца к столбцу или от строки к строке), а также в пределах формы (от поля к полю и от записи к записи).

Таблица 5.2 Использование клавиш управления курсором

Клавиша	Действие в таблице	Действие в форме
переход на одну колонку влево	переход влево или вверх на одно поле	

переход на одну колонку вправо	переход вправо или вниз на одно поле	
переход вверх на одну строку	переход вверх на одно поле	
переход вниз на одну строку	переход вниз на одно поле	
PgUp	переход вверх на один экран	переход вверх на одну запись
PgDn	переход вниз на один экран	переход вниз на одну запись
Home	переход в первое поле строки	переход в первое поле записи
End	переход в последнее попе строки	переход в последнее поле записи

В режиме Field View, вы можете комбинировать клавишу Alt с клавишами, указанными в таблице 5-2, для получения тех же результатов. Например, действие комбинации клавиш Alt+Home в режиме Field View полностью аналогично действию клавиши Home вне режима Field View.

Учтите, что если вы не находитесь в Field View, то нажатие клавиши Alt в комбинации с вышеуказанными клавишами не изменит их поведения.

Кроме того, существует набор клавиш, которые работают независимо от режима Field View (см. таблицу 5.3).

Таблица 5.3 Использование стандартных клавиш

Клавиша	**Действие в таблице**	**Действие в форме**
Ctrl+Home	переход в первое поле первой записи	переход в первое поле первой записи
Ctrl+End	переход в последнее поле последней записи	переход в последнее поле последней записи
Tab	переход в следующее поле	переход в следующее поле
Enter	переход в следующее поле	переход в следующее поле
Ctrl+Del	удаляет запись	удаляет запись
Ins	вставляет запись	вставляет запись

Если вы включили режим Field View и покидаете поле (посредством выбора другого поля мышью либо нажатием клавиш Enter, Tab или Alt в комбинации с клавишами управления курсором), Paradox выходит из режима Field View. Если вы хотите переместиться к другому полю и остаться в режиме Field View, используйте режим Persistent Field View (рассматривается ниже). Если вы хотите остаться в отмеченном поле и выйти из Field View, можно использовать любую из тех же команд, кнопок SpeedBar и клавиш, которыми режим Field View включается.

Режим Persistent Field View

Предположим, что вы хотите переместиться от одного поля к другому, не выходя из режима Field View. Для этого в Paradox существует режим Persistent Field View. Чтобы включить его, нажмите Ctrl+F2. Вы увидите слово Persist в поле системных сообщений. Вы можете использовать клавиши Home, End и клавиши управления так же, как вы делаете это в стандартном Field View, т.е. для перемещения в пределах поля. Для перемещения от одного поля к другому пользуйтесь клавишами Tab, Enter или Alt в комбинации с клавишами управления курсором. При этом вы останетесь в режиме Field View.

Чтобы выйти из Persistent Field View, нажмите Ctrl+F2. Теперь вы сможете использовать клавиши управления курсором для перемещения между полями.

Режим Memo View

Для редактирования memo и форматированных memo-полей Paradox предоставляет режим Memo View, который обладает некоторыми возможностями обработки текста и более широкими функциональными возможностями клавиатуры. Например, вы можете вставлять в текст символы конца строки и табуляции. Режим Memo View описывается ниже в разделе «Редактирование memo и форматированных memo-полей» данной главы

Вырезание, копирование и вставка данных с помощью Clipboard

Кроме того, что вы можете вручную вводить данные в поля, существует возможность вырезать или копировать данные из одного поля и вставлять их в другое поле или даже в Другие Windows-приложения. Вырезанные или скопированные данные остаются в Clipboard до тех пор, пока вы не очистите Clipboard или не выйдете из Windows. Clipboard обеспечивает временное хранение данных до их перемещения в другие области.

Вы можете производить с данными следующие действия:

- Дайте команду Edit / Cut, если хотите удалить отмеченные данные из таблицы (или формы) и поместить их в Clipboard. (Чтобы просто удалить выделенные в окне Table данные, выберите пункт Edit/Delete или нажмите клавишу Del.),
- Выберите пункт меню Edit / Copy для копирования отмеченных данных поля из таблицы (или формы) в Clipboard. Если вы работаете с таблицей, вы можете копировать данные из нескольких полей одновременно. Информация о выделении данных одновременно в нескольких полях таблицы изложена в Главе 4.
- Выберите пункт Edit / Paste, если хотите вставить содержимое Clipboard в отмеченное поле (в пределах одной операции вы можете вставить данные только в одну таблицу).

Учтите, что вы можете вставлять в поле только корректные данные (нельзя, например, вставить графические данные в текстовое поле).

Копирование в файлы и вставка данных из файлов

Вы можете копировать данные из поля во внешние файлы, а также вставлять данные из файлов в поле.

Paradox предоставляет вам возможность копировать данные из поля во внешние файлы. При работе с таблицей, вы можете скопировать данные графических, двоичных, memo и форматированных memo-полей в файлы других (от-личных от Paradox) форматов без использования команды Export.

Если вы работаете с формой, можно копировать данные из полей любых типов во внешние файлы без использования команды Export.

Например, для того, чтобы скопировать графические данные в .BMP-файл, выберите пункт меню Edit (Copy To. Вы увидите диалоговое окно Copy To Graphic File, с помощью которого вы можете выбрать имя существующего графического файла (содержимое поля перезапишет файл) или ввести имя нового файла в текстовом окошке New File Name .

Нажмите мышью кнопку OK, и Paradox разместит копию данных графического поля в определенном вами файле, а оригинал останется в вашей таблице или форме.

Учтите, что Paradox может копировать графические данные только в файлы .BMP-формата. Если вы работаете с двоичным полем в окне Table, вы можете использовать команду Copy To для копирования поля во внешний файл. При этом на экране появляется диалоговое окно Copy To File, которое выглядит так же, как диалоговое окно Copy To Graphic File, за исключением списка Type, показывающего расширения файлов, которые вы можете использовать.

Если вы работаете с формой, для копирования строк текста (поля любого типа, включая memo, числовые или даты) в файл следует дать команду Edit / Copy To. Paradox может копировать текстовые данные в файлы форматов .TXT или .PXT.

Находясь в окне Table, вы также можете скопировать в текстовый файл только memo и форматированные memo-поля. Чтобы пункт меню Edit/Copy To был доступен, должен быть включен режим Memo View.

Для копирования данных memo-поля в файл вы должны:

1. Выбрать нужное поле.
2. Войти в режим Memo View (нажать мышью кнопку Field View на SpeedBar или комбинацию кла-

виш Shift+F2). Все содержимое поля будет отмечено. (Конечно, вы можете отметить часть memo-поля, по своему усмотрению, и копировать в файл только ее).

3. Дайте команду Edit Copy To, на экране появится диалоговое окно Copy To File В нем вы можете воспользоваться раскрывающимся списком Path для выбора каталога, в которой хотите поместить файл, или использовать кнопку Browse для просмотра других каталогов.
4. Выполните одно из следующих действий:
   - Выберите имя существующего файла (содержимое поле перезапишет дан-ные в файле)
   - Создайте новый файл путем ввода его имени в текстовое окошко New File Name
5. По умолчанию Paradox копирует данные в файл, который находится в рабочем каталоге. Для сохранения файла в другом каталоге используйте одну из следующих возможностей:
   - Введите полное наименование каталога в текстовое окошко New File Name
   - Выберите нужный каталог из списка Path
   - Воспользуйтесь кнопкой Browse для просмотра и выбора каталога
6. Нажмите OK.

Paradox создаст новый файл с указанным вами именем и разместит в нем со-держимое отмеченного поля (если вы указали имя существующего файла, Paradox заменит его содержимое на данные из отмеченного поля).

Вы можете вставлять данные из внешних файлов в поля Paradox-таблиц. Для этого, находясь в режиме редактирования, выберите пункт меню Edit /Paste From. На экране появится диалоговое окно Paste From.

Тип файлов, отображенных в окне Paste From, зависит от выбранного вами типа поля или объекта. Если вы выбрали графическое поле, то Paradox откроет диалоговое окно Paste From Graphic File со словом <Graphic> в списке Type. Вы можете вставлять в графическое поле или объект данные из файлов, имеющих форматы .BMP, .PCX, .TIP, .GIF и .EPS.

Обратите внимание, что при импорте данных в графическое поле или объ-ект из файлов, имеющих форматы .PCX, .TIF, .GIF, и .EPS, Paradox сначала преобразовывает их в .BMP-формат.

Если вы выбрали в таблице memo или форматированное memo-поле, Paradox откроет диалоговое окно Paste From File со словом <Text> в списке Type. Вы можете вставить текст из файлов, имеющих форматы .PXT, .TXT и .RTF (если вы используете форму, то можете вставить текст в любой тип поля, включая графический и OLE).

После того, как вы выберете файл и нажмёте OK, Paradox поместит содержимое файла в текущее поле или объект.

Использование команды отмены Undo

Если вы отредактируете запись и дадите команду Edit / Undo перед тем, как выйти из редактируемой записи, Paradox проигнорирует все сделанные в записи изменения и выдаст сообщение о том, что они отменены. Так как Paradox сохраняет изменения сразу после выхода из редактируемой записи, вы должны использовать Undo до выхода из нее. Для того, чтобы отменить сделанные изменения в текущем поле, вы можете нажать Esc перед выходом из него: Paradox восстановит первоначальное содержание поля. Однако помните, что вы не можете использовать Edit / Undo для восстановления удаленных записей. Если вы однажды удалили запись из Paradox-таблицы, то уже не существует способа вернуть ее обратно, за исключением ее повторного ввода.

Замена данных

Paradox обеспечивает два средства быстрой замены данных существующе-го поля:

- Команда Record / Locate and Replace для замены всего содержимого поля
- Команда Edit / Search Text для замены последовательности символов в memo-поле

Для замены данных поля вы можете также использовать запрос CHANGETO .

Использование команды Locate and Replace

Пункт Record / Locate and Replace используется для поиска записи с конкретным значением в поле и замены его на другое. Для замены поля во всей таблице, вам придется повторять команду Locate and Replace для каждой записи. Поэтому для замены данных во всей таблице лучше использовать запрос CHANGETO

Предположим, что вы хотите заменить слово «Unisco» на фразу «Unisco Ltd» в поле Company

таблицы Contacts. Для этого:

- Откройте таблицу Contacts, войдите в режим Edit и перейдите в поле Company.
- Выберите пункт Record / Locate and Replace или нажмите Shift+Ctrl+Z. Вы увидите диалоговое окно Locate and Replace.
- Напечатайте «Unisco» в текстовом окошке Value. Напечатайте «Unisco Ltd» в текстовом окошке Replace With.
- Выберите опцию Exact Match для того, чтобы при поиске выполнялось точное сравнение с величиной в текстовом окошке Value.
- Нажмите ОК.
- Paradox переместит вас к первому местонахождению величины и запросит подтверждение на изменение данных (Yes -произвести замену и искать следующее вхождение, No -отказаться от замены и перейти к дальнейшему поиску, Cancel -прекратить поиск). Paradox будет запрашивать у вас разрешение на каждое изменение.

В каждой записи, которая содержит значение, введенное вами в окошко Value, Paradox заменит на значение в окошке Replace With.

Использование Search & Replace в memo-полях

В memo и форматированных memo-полях вы можете использовать команду Edit / Search Text для поиска фрагментов текста и (необязательно) замены их на другой текст. Для этого используется диалоговое окно Search & Replace.

Предположим, что вы хотите найти слово «Air Regulators» в поле Equipment Class таблицы Stock. Для этого:

1. Откройте таблицу Stock, войдите в режим Edit и переместитесь к первой записи поля Equipment Class. Нажмите Shift+F2, чтобы включить режим Memo View.
2. выберите пункт Edit/Search Text и вы увидите диалоговое окно Search & Replace
3. Ведите ”air Regulators” в текстовом окне Search for
4. Нажать мышью кнопку Search и Paradox найдет первое вхождение искомого слова, выделит его цветом и выведет “Match found” в поле системных сообщений
5. Нажмите Search еще раз, чтобы переместиться к следующему вхождению. Если в пределах текстового объекта вхождений больше нет, будет выдано сообщение “No match found”.

Вы можете использовать то же диалоговое окно для замены какого-либо фрагмента текста на другой (процедура замены аналогична процедуре поиска с той разницей, что заполняются два текстовых окошка : одно - Search For - для ввода искомого текстового фрагмента, а другое - Replace With -для ввода заменяющего текста, после чего сначала следует нажать кнопку Search, а когда Paradox найдёт искомый текст, использовать кнопку Replace для пошаговой замены или кнопку Replace Alt для замены всех вхождений).

Если Paradox нашел поле, которое вы не хотите заменять, нажмите Search для выхода из него без изменения и поиска следующего вхождения.

Чтобы удалить какие-либо данные, оставьте окошко Replace With незаполненным. Когда вы нажимаете Replace, Paradox удаляет найденные фрагменты текста, путем замены на пустой фрагмент.

Редактирование специальных типов данных

Некоторые типы полей Paradox требуют специальных методов ввода данных. Например, если вам нужно ввести картинку в графическое поле, то нельзя просто напечатать ее имя точно так же имеются определенные правила и соглашения, которые контролируют способ ввода и редактирования данных в полях типа OLE, memo и форматированных memo.

Редактирование memo и форматированных memo-полей

Ввод данных в memo и форматированные memo-поля осуществляется таким же образом, как и в алфавитно-цифровые. Однако при вводе данных в memo и форматированные memo-поля Paradox практически не ограничивает на объем вводимых данных. Во время редактирования memo и форматированных memo-полей вы можете использовать режим Memo View, который предоставляет некоторые возможности по обработке текста и повышает функциональные возможности клавиатуры по сравнению с режимом Field View. Например.

- В режиме Field View клавиши Enter и Tab служат для выхода из поля.

- В режиме Memo View клавиша Enter разбивает строку на две, а клавиша Tab устанавливает символ табуляции

Как включить режим Memo View, зависит от того, с чем вы работаете - с таблицей или формой.

Включение Memo View из таблицы

Находясь в таблице, вы можете войти в режим Memo View нажав Shift+F2 или используя способы включения Field View. При этом Paradox заполнит все окно Table текущим memo-полем. Tab, Enter и клавиши управления курсором позволяют перемещаться в пределах memo-поля. В поле сообщений выводится подсказка о том, что вы находитесь в режиме Memo View, а также как из него выйти. Текст пролистывается вверх, если вы достигли нижнего края окна, и автоматически форматируется правой границей окна. При изменении размеров окна изменяется и формат текста.

Если вы закончили процесс редактирования, то можете выйти из Memo View следующим образом:

- Нажать F2 или Shift+F2
- Выбрать пункт меню Table / Reid View
- Нажать кнопку Field View на SpeedBar
- Закрыть окно Table

Учтите, что при выходе из режима Memo View Paradox сохраняет измене-ния, сделанные в memo-поле. Если необходимо отменить произведённые изменения, нажмите Esc.

Когда вы возвращаетесь в таблицу, размер видимой части memo-поля зависит от ширины содержащего его столбца. Вы можете варьировать ширину столбца, передвигая мышью линию сетки таблицы в области заголовка, но изменить размер поля можно только путем изменения структуры таблицы.

Включение Memo View из формы

Работая с формой, вы можете включить как режим Field View, так и режим Memo View:

- В режиме Field View вы можете пользоваться клавишами Enter и Tab для перемещения к другому полю формы. Этот способ удобен, если вы хотите отредактировать только один параграф или внести совсем небольшие изменения.
- В режиме Memo View (нажмите Shift+FSj вы можете разбивать строки клавишей Enter и ставить символы табуляции клавишей Tab. Чтобы снова использовать эти клавиши для перемещения по полям, необходимо выйти из Memo View еще раз).

Во время ввода данных в memo-поле формы вы не можете изменять размер поля. Чтобы сделать это, необходимо открыть окно Form Design, щелкнув кнопку Design на SpeedBar.

При вводе данных, когда вы достигаете правой границы окна, текст автоматически переносится на следующую строку. Этого не происходит, если выключена опция Word Wrap данного текстового объекта. В этом случае необходимо открыть окно Form Design и проинспектировать данный объект.

Инспектируя объект, вы также можете задать для поля горизонтальную и вертикальную линейку прокрутки, чтобы имея небольшое по размерам поле, просматривать все его содержимое

Форматирование текста

Для того, чтобы отформатировать текст в форматированном memo-поле, выделите нужный блок текста, проинспектируйте его и из меню выберите необходимые пункты. Для форматирования всего содержимого memo-поля, вы должны выделить его полностью: Paradox форматирует только выделенные участки текста.

Для установки интервала между строками в форматированном memo-поле войдите в пункт Line Spacing .

Вы можете выбрать интервал в 1 (по умолчанию), 1.5, 2, 2.5 или 3 строки между строками текста. Чтобы изменить начертание, размер, стиль или цвет текста, войдите в пункт Font (на экране появится палитра Font, работа с которой описывалась выше).

Ввод графических изображений

Данные в графическом поле могут быть любыми картинками или графиками, которые отсканированы или созданы в графических редакторах. Paradox предоставляет вам два способа ввода графического изображения в поле:

- Использованием команд Cut, Copy и Paste и при посредстве Clipboard
- Командой Paste From

Учтите, что Paradox не имеет возможностей для редактирования графических изображений. Для того, чтобы ввести графическое изображение в графическое поле, вы должны войти в режим редактирования и вставить графическое изображение из Clipboard или файла.

Когда вы входите в режим Field View в графическом поле, Paradox заполняет все окно Table графическим изображением. Ниже приведена последовательность действий по вводу графического изображения в поле:

1. Откройте графический файл в той среде, в которой он был создан.
2. Выделите необходимый фрагмент рисунка и скопируйте его в Clipboard.
3. Откройте окно Table или Form для размещения в нем графического изо-бражения.
4. Войдите в режим редактирования.
5. Выберите нужное графическое поле.
6. Дайте команду Edit / Paste.Paradox перенесет изображение из Clipboard в графическое поле.

Paradox предоставляет вам возможность вводить .BMP, .PCX, .TIP, .GIF или .EPS файлы непосредственно в графическое поле без дополнительного открытия соответствующих графических приложений. Просто используйте команду Edit / Paste From, например:

1. Выберите нужное графическое поле.
2. Включите режим редактирования.
3. Выберите пункт меню Edit / Paste From.
4. Выберите при помощи залогового окна Paste From Graphic File нужный гра-фический файл.
5. Нажмите OK и Paradox поместит графическое изображение в текущее поле.

Следует заметить, что при вводе изображений Paradox преобразовывает графику в .BMP формат. Если графическое изображение вводится в таблицу, вы можете не увидеть его целиком. Отрегулируйте размеры столбца так, чтобы увидеть максимальную часть рисунка. Чтобы посмотреть графическое изображение полностью, вы можете войти в режим Field View. Для ускорения перемещения по записям следует проинспектировать графическое поле и выключить опцию Complete Display (при этом на экран будет выводится графическое поле только текущей записи).

Если графическое поле формы имеет неподходящие размеры для вывода изображения, вам придется модифицировать форму в окне Form Design или про-инспектировать поле и включить опцию Magnification / Best Fit. Работая с формой вы также можете варьировать опцию Complete Display.

Использование технологии OLE

Аббревиатура OLE означает «связывание и встраивание объектов» (object linking and embeding). Вы можете использовать OLE-поля для того, чтобы виртуально хранить любой вид данных - от графических до текстовых и вычисляемых. Преимущество использования OLE-полей заключается в том, что если вы однажды разместили OLE-данные, они сохраняют связь со средой, в которой были созданы. Вы всегда можете перейти в эту среду или файл из OLE-объекта, который вы разместили в Paradox таблице или форме. Изменения, которые вы сделаете там с оригиналом OLE-объекта, отразятся в вашей Paradox-таблице или форме

Редактирование полей с контролем корректности данных

Проверка корректности накладывает на вводимые данные ограничения с тем, чтобы гарантировать, что они удовлетворяют определенным требованиям. Если установлен контроль корректности, то вы не сможете поместить в таблицу запись или выйти из нее, если все поля записи не отвечают поставленным условиям. Если вы введете некорректные данные, Paradox заблокирует выход из записи. Вы должны будете либо ввести корректные данные либо восстановить первоначальное содержание записи. Таблица 5.4 описывает типы контроля корректности данных и то, как они влияют на ввод данных в поле.

Таблица 5.4 Типы контроля корректности данных

Тип контроля	Описание
Required field	Вы не можете выйти из записи до тех пор, пока не введете величину. Это гарантирует, что запись будет иметь значение в каждом поле.

Minimum value	Paradox запрещает ввод величины меньшей, чем минимальное значение.
Maximum value	Paradox запрещает ввод величины большей, чем максимальное значение.
Default value	Paradox автоматически вводит заданное значение в конкретное поле когда вы вставляете новую запись. Для того, чтобы ввести величину, отличную от заданной, выберите нужное поле и введите величину по своему усмотрению. Для того, чтобы ввести пустую запись, выберите поле и нажмите Backspace или Del.
Picture	Шаблоны - используются Paradox для форматирования вводимых в поле данных. Обычный шаблон выглядит так: ##.##.##. Это шаблон для ввода даты (01.12.91). Если вы определили этот шаблон для данного поля, то можете вводить числа без точек - 011291. Paradox отформатирует числа в соответствии с шаблоном. Проверка корректности данных посредством шаблона помогает при редактировании (автоматическое форматирование данных), а также устанавливает правила, которые гарантируют, что вводимые данные будут соответствовать установленным вами требованиям для данного поля.

Блокирование записей

Блокирование записей - очень важная и необходимая операция для сохранения целостности базы данных в многопользовательской среде. Если вы работаете в сети (локальной или глобальной) и заблокировали запись, другие пользователи могут ее просматривать, однако не могут отредактировать или удалить. Paradox автоматически блокирует запись, когда вы начинаете ее редактирование и снимает блокировку, когда вы выходите из нее. Поле системных сообщений (status bar) постоянно информирует вас об этом.

Вы также можете заблокировать запись вручную перед тем, как начнёте ее редактирование. Отметьте запись и затем выберите пункт Record /Lock (либо нажмите F5 либо Ctrl+L). В поле сообщений вы увидите информацию о том, что данная запись заблокирована. Блокировка записи исключает возможность для других пользователей установить свою собственную блокировку. По этой же причине она также запрещает доступ к записи в таблице другим пользователям. После того, как вы заблокировали запись, команда Lock в меню заменится на команду Unlock. Выберите Unlock, если хотите освободить запись для доступа к ней других пользователей без выхода из нее (вы должны разблокировать записи, чтобы другие пользователи смогли ее редактировать

Использование таблицы-справочника

Данный режим позволит вам вводить только те данные, которые уже суще-ствуют в другой таблице - таблице-справочнике (lookup table). Например, вы можете задать для поля Customer No (номера клиентов) таблицы Orders (Счета - заказы) таблицу-справочник Customer (клиенты), чтобы случайно не принять заказ от несуществующего клиента.

Режимы использования таблицы-справочника

Paradox может использовать таблицу-справочник в двух режимах:

- Just Current Field : данные в текущем поле - единственное, что Paradox проверяет по таблице-справочнику.
- All Corresponding Fields : Paradox проверяет поле, для которого задан справочник, и переносит из него в вашу таблицу еще несколько полей (Paradox выбирает совпадающие имена полей).
- Видите вы таблицу-справочник на экране при вводе данных или нет зависит от другого режима ее использования:
- Help and Fill: разрешает просматривать таблицу-справочник из таблицы, которую вы редактируете.
- Fill No Help: запрещает просматривать таблицу-справочник из редактируемой таблицы, однако, вы можете просмотреть данную таблицу, открыв ее в собственном окне Table.

Режим Just Current Field

Предположим, вы редактируете таблицу Order, в которой полю Customer No задана таблица-справочник Customer (ее первое поле также называется Customer No) в режиме Just Current Field. Если вы заполняете поле Customer No, то должны ввести значение, которое уже имеется в поле Customer No таблицы Customer. Если вы ввели некорректные данные (номер клиента, который не существует в таб-лице Customer), Paradox выдаст соответствующее сообщение и не разрешит вам выйти из записи до тех пор, пока вы не введете корректные данные.

Если справочник используется в режиме Help and Fill, процедура ввода данных выглядит следующим образом:

- Переместитесь в поле Customer No. в таблице Order. Поле системных сообщений проинформирует вас о том, что можно нажать Ctrl+Spacebar для просмотра таблицы-справочника.
- Нажмите Ctrl+Spacebar или выберите пункт меню Record /Lookup Help. вы увидите таблицу-справочник (Customer) в диалоговом окне в верхней части редактируемой таблицы. Если в редактируемой таблице уже набрана корректная запись, то текущее положение курсора покажет эту запись в таблице-справочнике. Например, если вы ввели 1007 и после этого нажали Ctrl+Spacebar, курсор остановится на значении 1007 в таблице Customer.
- Отметьте в таблице-справочнике нужную запись.
- Нажмите ОК и значение из справочника перенесётся в текущее поле редактируемой таблицы.

Если используется режим Fill No Help, то вы сами должны ввести корректные значения (только те номера клиентов, которые существуют в Customer). Paradox будет проверять их существование в справочнике, но не предложит явно воспользоваться им.

Режим All Corresponding Fields

Предположим, вы редактируете таблицу Order, в которой полю Customer No задана таблица-справочник Customer (ее первое поле также называется Customer No) в режиме All Corresponding Fields. Обе таблицы имеют поле Name, которое содержит имена клиентов. Когда вы редактируете Order и вводите поле Customer No, Paradox проверяет его в справочнике и значения всех соответствующих полей (таких как Name) копирует из справочника в таблицу Order. Если вы ввели некорректные данные (номер клиента, который не существует в таблице Customer) Paradox выдаст соответствующее сообщение и не разрешит вам выйти из записи до тех пор пока, вы не введете корректные данные.

Процедуры использования опций Help and Fill и Fill No Help аналогичны приведённым выше.

Глава 6. Запросы

В данной главе рассказывается о способах обращения к данным, содержащимся в базе данных. Вы научитесь извлекать из ваших таблиц нужную вам информацию и получать ее в нужном вам виде.

Информация описанная ниже полностью подходит для Borland Delphi Data-Base Desktop. T.e. вы можете разрабатывать запросы по методике, описанной ниже и потом использовать их в Delphi. римеры этой главы используют таблицы, специально разработанные для учебных целей и размещённые в каталоге C:\Program Files\Corel\WordPerfect Office 2000\Paradox\FwSample\ (если, конечно, при инсталляции Paradox вы не изменили имя, присваиваемое корневому каталогу по умолчанию, и не забыли инсталлировать учебные таблицы). Поэтому для работы с учебными таблицами выберите в качестве рабочего каталог указанный каталог.

Запросом называется некоторая совокупность действий, выполняемых системой Paradox, с помощью которых пользователь может извлечь необходимую информацию из своих таблиц. Запросом может быть как простейший поиск информации по значению в какой-либо одной таблице, так и сложный процесс преобразования и представления в определённом виде взаимосвязанных между собой данных из нескольких таблиц.

- Составляя запрос, вы можете указать в нем:
- Интересующие вас таблицы
- Поля, из которых должен состоять ответ
- Интересующие вас записи
- Необходимые преобразования данных

Вы можете также использовать осуществления следующих операций:

- Вставка новых записей
- Удаление записей
- Изменение значений
- Создание новых полей

С помощью запросов вы можете решать некоторые задачи, типа:

- Насколько вырастет общий объем продаж вашей продукции, если объем продаж одного из видов продукции возрастёт на 10%?
- Каковы будут транспортные издержки, если цены на авиаперевозки увеличатся на 10%?

Как работает запрос

В Paradox используется метод, называемый запрос по образцу (query by example - QBE). При составлении запроса вы задаёте Paradox некоторый образец, в соответствии с которым он должен представить вам результат. Это даст вам возможность сосредоточить внимание на том, что именно (а не как) вы хотите получить.

Table : customer.db

	Customer No	Name	Street
1	1 231,00	Unisco	PO Box Z-547
2	1 351,00	Sight Diver	1 Neptune Lane
3	1 354,00	Cayman Divers World Unlimited	PO Box 541
4	1 356,00	Tom Sawyer Diving Centre	632-1 Third Frydenhoj
5	1 380,00	Blue Jack Aqua Center	23-738 Paddington Lane
6	1 384,00	VIP Divers Club	32 Main St.
7	1 510,00	Ocean Paradise	PO Box 8745
8	1 513,00	Fantastique Aquatica	Z32 999 #12A-77 A.A.
9	1 551,00	Marmot Divers Club	872 Queen St.

Рис. 6.1 исходная таблица

Предположим, у вас есть таблица Customer с данными по вашим клиентам (рис. 6.1). В ней приведены названия и адреса магазинов - заказчиков вашей фирмы.
Если вас интересуют лишь названия и телефонные номера магазинов, вы можете попросить Paradox показать вам только поля Name и Phone. Для этого откройте окно Query и задайте образец желаемой таблицы (рис. 6.2).

Query : <Untitled>

customer.db	Customer N	Name	Street	City	State/Prov	Zip/Postal C	Country	Phone
		✓						✓

Рис. 6.2 Запрос

Table : :PRIV:ANSWER.DB

	Name	Phone
1	Action Club	813-555-6732
2	Action Diver Supply	809-555-6917
3	Adventure Undersea	501-4-20013
4	American SCUBA Supply	213-555-6119
5	Aquatic Drama	613-555-3463
6	Atlantis SCUBA Center	207-555-0107
7	Blue Glass Happiness	213-555-1984
8	Blue Jack Aqua Center	808-555-8904
9	Blue Sports	503-555-0393
10	Blue Sports Club	813-555-9775
11	Catamaran Dive Club	213-555-0422
12	Cayman Divers World Unlimited	809-555-8575
13	Central Underwater Supplies	27-11-4432458
14	Davy Jones' Locker	804-555-2692
15	Diver's Grotto	213-555-1909

Рис. 6.3 Результат запроса

Результат выполнения этого запроса появится во временной таблице с названием Answer в личном каталоге (рис. 6.3).

Как можно видеть, ответ соответствует образцу, заданному вами в запросе, т.е. в нем присутствуют только отмеченные вами поля.

Таблица Answer

Paradox переписывает таблицу Answer при каждом вашем запросе и уничтожает ее по окончании сеанса работы. Поэтому, чтобы сохранить содержимое таблицы Answer, вы должны присвоить ей другое имя. Сделать это можно как до, так и после выполнения вашего запроса:

- Перед выполнением запроса (после того, как он полностью сформирован):
  1. Дайте команду Properties / Answer Table / Options из окна Query (или нажмите кнопку Answer Table Properties на SpeedBar).
  2. В текстовом окошке Answer Name диалогового окна Answer Table Proper-ties измените имя ANSWER.DB на какое-либо другое, отвечающее требованиям DOS и имеющее расширение .DB (или .DBF для dBASE-таблицы).
  3. Нажмите OK.
- После выполнения запроса:
  1. Выберите пункт меню Table /Rename из окна Table с таблицей Answer.
  2. В диалоговом окне Rename введите новое имя файла.
  3. Нажмите OK.

  либо:
  1. Выберите пункт меню Tools / Utilities / Rename.
  2. Выберите :PRIV:ANSWER.DB из списка Tables окна Table Rename. Paradox поместит имя :PRIV:ANSWER в текстовое окошко Table.
  3. Введите нужное имя файла в текстовое окошко New Name.
  4. Нажмите OK.

Paradox помещает все временные таблицы, включая Answer, в ваш личный каталог. Если вы не определили свой личный каталог и являетесь сетевым пользователем Paradox, установленного на файл-сервере, все ваши временные объекты, в том числе и таблица Answer, будут размещаться во временном подкаталоге системы Windows.

Пример. Типовой запрос

Предположим, вы хотите получить список имен клиентов с номерами их телефонов.

Запустите Paradox и выберите в качестве рабочего подкаталога FWSAMPLE. (Все команды, средства и приемы, используемые в данном примере, подробно обсуж-даются в последующих разделах этой главы.

1. Дайте команду File / New /Query. На экране появится окно Select File (см. Главу 2).
2. В окне Select File отметьте CUSTOMER.DB и нажмите OK (или выберите это поле двойным щелчком левой клавиши мыши). На экране появится окно Query с образцом запроса по таблице Customer.
3. Щелкните мышью включатель поля Name или переместитесь к нему и нажмите F6.
4. С помощью линейки прокрутки переместитесь к полю Phone и аналогичным образом включите его в запрос.
5. Выполните запрос, нажав мышью кнопку Run Query на SpeedBar или выбрав пункт меню Query /

Run. На рисунке активирована таблица Answer и поэтому на SpeedBar окно для таблиц.

6. После выполнения запроса Paradox открывает окно с таблицей Answer поверх окна Query. (рис.6,1, 6,2, 6,3)

Окно Query

С помощью команды File / New / Query создайте новый запрос. При этом откроется пустое окно Query, а поверх него - диалоговое окно Select File. Вы имеется возможность:
Нажав Cancel, закрыть окна Select File и Query.

- Выбрать из списка таблиц текущего рабочего каталога одну (двойным щелчком левой клавиши мыши) или несколько таблиц (Sift+щелчок мыши, если имена файлов расположены в списке подряд, и Ctrl+щелчок - в остальных случаях),
- Изменить рабочий и личный каталоги с помощью списка Path.
- Открыть окно Browser, нажав кнопку Browse. Оно может помочь вам при выборе каталога, содержащего нужные таблицы, или при выборе какой-либо отдельной таблицы.
- Ввести имя интересующей вас таблицы в текстовое окошко File Name. (Если эта таблица находится не в текущем каталоге, укажите ее полное имя, включая имя каталога.)
- Выбрать другие (отличные от таблиц) типы файлов с помощью списка Type. Когда вы указываете в нем тип файла, отличный от таблицы (расширения .DB и .DBF), в списке Select File появляются имена файлов данного типа (с расширениями .QBE для запросов, .FSL - для форм, .RSL или .RDL - для отчетов). После этого в текстовом окошке File Name вы можете задать имя нужного файла (соответствующее выбранному типу). Обратите внимание, что в отличие от таблиц из списка Select File вы можете выбрать только од-ну форму, отчет или запрос.

После выбора нужных файлов и нажатия ОК окно Select File закроется, и Paradox поместит в окно Query образцы запросов по всем объектам, находящихся в выбранных вами файлах. Следуйте следующим правилам при размещении образцов запросов в окне Query.

- В окне Query нельзя размещать более одного образца запроса по одной таблице.
- Если в окне Select File вы выбрали более одной таблицы (несколько файлов с расширением .DB или .DBF), то образцы запросов по этим таблицам появятся в окне Query в том же порядке, в каком имена файлов следовали в окне Select File.
- Если файлы формы, отчёта или запроса содержат более одной таблицы (например, в случае, если эти объекты были созданы на основе нескольких связанных между собой таблиц), то Paradox размещает сначала образец запроса по главной таблице, а затем - по связанным.
- Если вы хотите добавить некоторые образцы запросов (командой Query / Add Table или с помощью кнопки Add Table на SpeedBar) к уже имеющимся в окне Query, то выбирать вы можете лишь из файлов таблиц в окне Select File; файлы же форм, отчетов и запросов будут вам недоступны.

Запросы по dBASE-таблицам

Вы можете составлять и выполнять запросы по dBASE-таблицам точно так же, как и по Paradox-таблицам. В окне Select File вам доступны все файлы с расширением .DB (Paradox-таблицы) и .DBF (dBASE-таблицы) вашего рабочего каталога. При задании запроса по dBASE-таблице Paradox помещает в окно Query образец запроса точно так же, как в случае Paradox-таблицы.

Однако, различия между типами полей в dBASE- и Paradox-таблицах препятствуют использованию некоторых операторов запросов Paradox при работе с dBASE-таблицами.

Связанные многотабличные объекты

При выборе многотабличного запроса, формы или отчета Paradox помещает в окно Query образцы запросов по всем таблицам, автоматически связывая их в соответствии со связями внутри этих многотабличных объектов. Связь образцов осуществляется путем помещения элемент - примеров в общие (связывающие) поля (см. раздел «Использование элемент - примеров» ниже в этой главе).

Объекты с парольной защитой

При выборе в окне Select File защищённого паролем объекта Paradox попросит вас ввести пароль и только после того, как вы сделаете это и нажмёте ОК, разместит в окне Query образцы запросов.

Если вы потом удалите из памяти Paradox с помощью команды tools /Utilites / Passwords все вве-

денные вами пароли, вам придется ввести защищающий объект пароль вновь, чтобы запрос по этому объекту был принят к исполнению.

Образец запроса

Образец запроса имеет вид той таблицы, к которой он относится. В нем при-сутствуют те же поля и в том же порядке, что и в соответствующей таблице; отсутствуют лишь ее данные. Когда вы вносите в таблицу какие-либо изменения (например, изменяете порядок следования столбцов или вид текста в их заголов-ках), они не отражаются автоматически на образце, однако, вы можете сами изменить порядок следования столбцов в образце.

Перемещение между полями образца запроса и ввод данных осуществляются так же, как и при работе с таблицами в режиме редактирования. Вы можете вставлять и удалять строки в образце так, как если бы это были записи в таблице. На рисунке 6.4 изображено окно Query с образцом запроса.

Таблица 6.1 Операции, доступные при работе с образцом запроса

Операция	Работа с мышью	Работа с клавиатурой
Связывание таблиц с помощью элемент -примеров	Нажмите кнопку Joint Tables на SpeedBar и щелкните общие поля об-разцов запросов	Нажмите F5 и введите в запрос элемент-пример
Включение поля в запрос значком V	Щелкните мышью включатель поля V	Нажмите F6 - проявится V
Включение поля в запрос значком V+	Нажмите мышью включатель поля и, удерживая клавишу мыши, выберите в появившемся меню значок V+	Нажимайте комбинацию Shift+F6 пока не появится значок V+
Включение поля в запрос значком V ↓	Нажмите мышью включатель поля и, удерживая клавишу мыши, выберите в появившемся меню значок V ↓	Нажимайте комбинацию Shift+F6 пока не появится значок V ↓
Включение поля в запрос значком V G	Нажмите мышью включатель поля и, удерживая клавишу мыши, выберите в появившемся меню значок V G	Нажимайте комбинацию Shift+F6 пока не пояится значок V G
Вызов меню операций запроса в крайнем левом столбце образца запроса	Установите курсор мыши под именем таблицы нажмите левую клавишу	Нажмите Spacebar (пробел), находясь в крайнем левом столбце образца запроса под именем таблицы
Выбор в меню операций запроса		Вызвав меню, нажмите первую букву пункта меню
Удаление введённых в текущее поле запроса условий его выбора		Нажмите Esc
Удаление всех значков и условий выбора в текущей строке запроса		Нажмите Ctrl+Del, находясь в любом поле очищаемой или удаляемой

SpeedBar окна Query

На SpeedBar окна Query расположены кнопки, которые предназначены для работы с запросами (рис. 6.4).

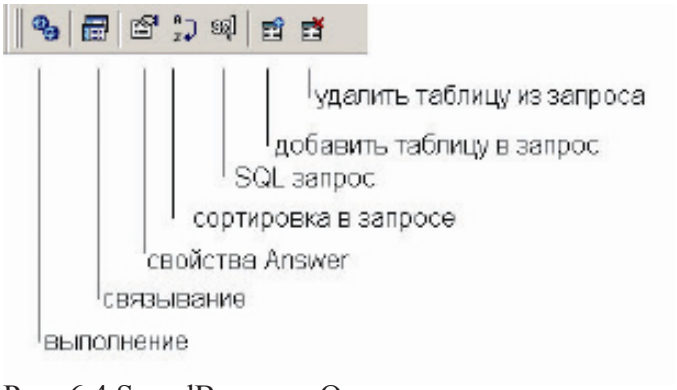


Рис. 6.4 SpeedBar окна Query

Операции Cut, Copy, Paste

Для работы в окне Query Paradox предоставляет следующие вспомогательные операции (см.выше), использующие Windows Clipboard: удалить и поместить в Clipboard (Cut), скопировать в Clipboard (Copy) и вставить из Clipboard (Paste). С их помощью можно манипулировать данными в образцах запросов, перенося их из одного поля в другое.

Помимо команды меню Edit / Cut и кнопки Cut на SpeedBar для удаления выделенного фрагмента вы можете пользоваться командой Edit / Delete, При этом удаляемая информация не помещается в Windows Clipboard.

Использование Paste Link

Если у вас возникла необходимость использовать механизм DDE (Dynamic Data Exchange - динамический обмен данными) для установки связи между об-разцом запроса и данными из другой Windows-программы, воспользуйтесь пунктом меню Edit / Past Link. (Более подробно вопросы, связанные с DDE, рассмотрены в Главе 14. Там же объясняется назначение пункта меню Query / Wait for DDE.)

Выполнение запроса

Выберите пункт меню Query / Run, нажмите клавишу F8 или кнопку Run Query на SpeedBar и ваш запрос будет принят к выполнению. При появлении каких-либо проблем Paradox немедленно выдает сообщение об ошибке и просит вас устранить ее.

Добавление и удаление образцов запросов

Для того чтобы ввести в запрос дополнительные образцы, выберите пункт меню Query / Add Table или нажмите на SpeedBar кнопку Add Table. На экране появится диалоговое окно Select File. Если в окне Query у вас уже есть по крайней мере один образец запроса, то в окне Select File вы сможете выбрать лишь табличный файл (но не форму, отчет или запрос).

Предположим, вы добавили по ошибке не тот образец запроса в окно Query или же хотите составить запрос по другой таблице. В этом случае у вас нет необходимости открывать новое окно Query, вы можете просто удалить нежелательные образцы. Для этого служат команда меню Query / Remove Table и кнопка Remove Table (удаление) на SpeedBar, вызывающие диалоговое окно Remove Table.

В этом диалоговом окне представляются все таблицы, образцы запросов по которым присутствуют в данный момент в окне Query. Отметьте ненужные таблицы и нажмите ОК. (Для выделения одного файла, последовательной группы последовательных файлов или нескольких, расположенных не подряд, используйте соответственно двойной щелчок левой клавиши мыши, Ctrl+щелчок и Shift+щелчок.)

Связывание таблиц

вы должны поместить в окно Query образцы запросов по всем необходимым таблицам и связать их, используя их общие поля, а, именно, помещая в них соответствующие элемент - примеры.

Ниже в этой главе будет рассказано об автоматическом и «ручном» способах размещения элемент - примеров.

Редактирование условий выбора

В случае необходимости отредактировать условия выбора, которые вы задали в образце запроса, вы можете воспользоваться как клавишей Backspace, так и режимом Field View. Для этого нужное поле сделайте текущим и выберите пункт меню Query / Field View либо нажмите клавишу F2 на клавиатуре или кнопку Field View на SpeedBar.

Режим Field View включается также щелчком мыши в текущем поле.

Изменение структуры таблицы Answer

Пока вы сами не пожелаете изменить структуру таблицы Answer, она будет в точности соответствовать образцу вашего запроса: первым полем является крайнее левое поле, отмеченное вами в первом образце запроса; вторым -следующее (слева направо) поле, отмеченное в первом образце и т.д. - по всем полям всех образцов запроса.

Если таблица Answer содержит поля с повторяющимися именами (из двух или более таблиц), то Paradox оставит имя первого из этих полей без изменения, а остальные назовёт имя_1, имя_2 и т.д.

Новые вычисляемые поля помещаются в конец таблицы Answer и получают имена в соответствии с формулой, по которой производятся вычисления. Все поля таблицы также могут быть переименованы с помощью оператора AS (см. разделы «Переименование полей таблицы Answer с помощью оператора AS» и «Вычисления в запросах» ниже в этой главе).

Изменение свойств таблицы Answer

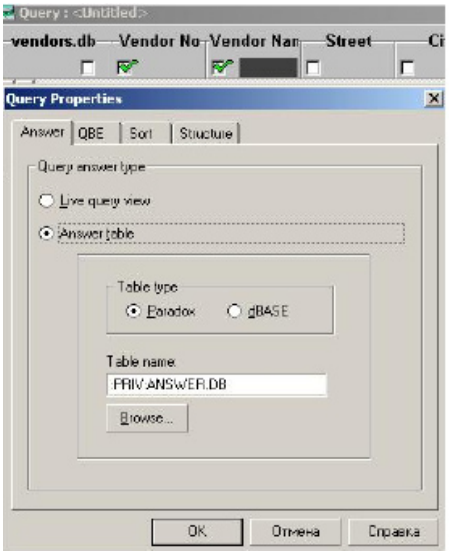


Рис.6.5 Свойства Answer

Вы можете изменить некоторые параметры таблицы Answer перед тем, как начать выполнение запроса (запрос должен быть сформирован). Выберите пункт меню Properties / Answer Table / Options или нажмите на SpeedBar кнопку Answer Table Properties. На экране появится диалоговое окно Answer Table Properties (рис.6.5).

Помните, что к этому моменту вся работа в окне Query должна быть завершена. Делать изменения в таблице Answer рекомендуется непосредственно перед выполнением запроса.

Находясь в окне Answer TaЬIe Properties, вы можете:

- Изменить каталог, котором располагается таблица Answer, введя в текстовое окошко Name полное имя файла таблицы, включая имя нового каталога, заменяющее псевдоним вашего личного каталога. Изменение каталога приведёт к сохранению таблицы Answer по окончании сеанса работы с Paradox. Помните, что при сохранении таблицы Answer в каталоге, уже содер-жащем таблицу с этим именем, старая будет без предупреждения перезаписана.
- Дать таблице Answer новое имя перед выполнением запроса, введя его в текстовое окошке Answer Name вместо значения по умолчанию ANSWER.DB. Это также приведёт к сохранению таблицы по окончании сеанса работы с Paradox. Помните, однако, что при сохранении таблицы под новым именем в каталоге, уже содержащем таблицу с таким же именем, старая будет перезаписана без предупреждения.
- Выбрать тип таблицы Answer - Paradox- или dBASE-таблица, включив соответствующую опцию на панели Answer TaЬIe Type.
- Изменить порядок следования полей в таблице, воспользовавшись образцом таблицы Answer на панели Image of Answer TaЬIe. При этом вам будут доступны все функции работы с таблицей, как если бы вы работали в ее собственном окне TaЬIe (см. Главу 4). Различие состоит лишь в том, что вы можете делать это перед выполнением запроса и, следовательно, до фактического создания таблицы.
- После внесения всех желаемых изменений нажмите OK для возврата в окно Query. При выполнении запроса все ваши изменения будут учтены. Сохранение характеристик таблицы Answer происходит при сохранении запроса.

Сортировка в таблице Answer

Перед выполнением запроса вы можете указать способ сортировки записей в таблице Answer. Для этого выберите пункт меню Properties / Answer Table / Sort. На экране появится диалоговое окно Sort Answer (рис. 6.8).

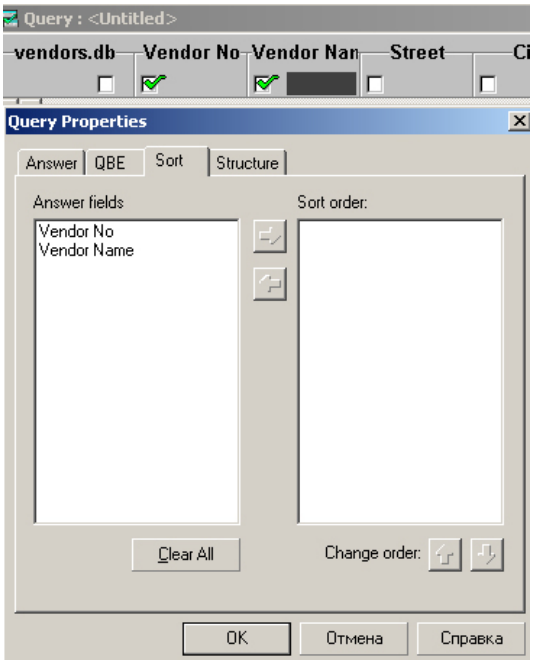


Рис. 6..6 Диалоговое окно Sort Answer

Для задания полей, по значениям которых вы хотите отсортировать записи в таблице Answer, выделите в списке Answer fields нужные поля и воспользуйтесь кнопкой-стрелкой Add Fields для переноса их в список Sort order. Вносите поля в этот список в том порядке, в котором должна быть отсортирована таблица Answer. Поля в списке Sort order можно удалять, а также изменять порядок их следования. Для этого отметьте нужное поле и воспользуйтесь кнопками-стрелками Remove Field или Change Order. Нажмите OK, и таблица Answer после выполнения запроса будет отсортирована в соответствии со списком Sort by.

Таблица 6.2 суммирует рассмотренные выше операции и команды, доступные в окне Query.
таблица 6.2 Выполнение операций в окне Query

Операция	Использование системы меню	Работа с мышью	Работа с клавиа-турой
Удаление выделенного текста и помещение его в Clipboard	Команда Edit / Cut	Кнопка Cut на SpeedBar	Комбинация Shift + Del
Копирование выделенного текста в Clipboard	Команда Edit / Copy	Кнопка Copy на SpeedBar	Комбинация Ctrl + Ins
Вставка текста из Clipboard	Команда Edit / Paste	Кнопка Paste на SpeedBar	Комбинация Shift + Ins
Удаление выделенного текста	Команда Edit / Delete		Клавиша Del
Выполнение запроса	Команда Query / Run	Кнопка Run на SpeedBar	Клавиша F8
Добавление таблицы в окно Query	Команда Query/ Add Table	Кнопка Add Table на SpeedBar	
Удаление таблицы из окна Query	Команда Query / Remove Table	Кнопка Remove Ta-ble на SpeedBar	
Включение / выключение режима Field View	Команда Query / Field View	Кнопка Field View на SpeedBar или щел-чок в текущем поле запроса	Клавиша F2
Изменение свойств таблицы Answer	Выберите пункт меню Properties / Answer Table	Кнопка Answer Ta-ble Properties на SpeedBar	

Сохранение и восстановление параметров запроса

Вы можете определить некоторые параметры запроса, которые будут действовать во всех выполняемых запросах и сохраняться после выхода из Paradox

- Опции выполнения запроса в локальной сети
- Способ отображения в окне Query нескольких образцов запросов

Эти параметры включаются с помощью пунктов Restart Options, Tile Tables, Cascade Tables меню Properties, а сохраняются и восстанавливаются командами Query Options / Save as Default и Query Options / Restore Default.

Опции выполнения запроса в локальной сети

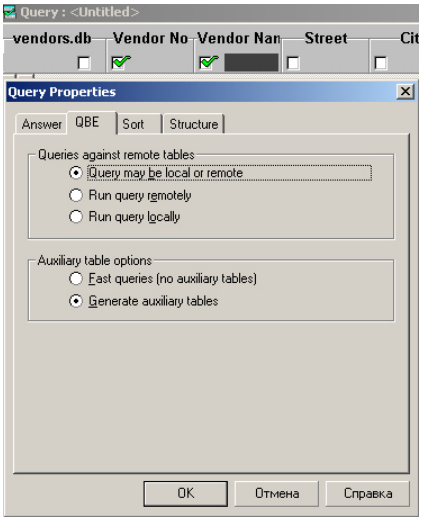


Рис. 6.7 Диалоговое окно Query Options

При работе в локальной сети таблицы Paradox могут подвергаться одновре-менным изменениям со стороны нескольких пользователей. Будучи одним из них, вы можете указать, следует ли вашей таблице Answer учитывать изменения, производимые другими пользователями во время выполнения вашего запроса.

Первоначальная установка Paradox такова, что он повторно выполняет за-прос при обнаружении изменений в оригинальных таблицах, произошедших в процессе работы запроса. Однако, вы можете указать другой способ обработки подобных ситуаций, вызвав командой Properties / Restart Options диалоговое окно Query Options (рис. 6.7):

- Query may be local or remote: запрос может выполняться локально или по сети
- Run query locale: на время выполнения запроса все использующиеся им таблицы блокируются. Если в данный момент какой-либо другой пользователь работает с нужными вам таблицами, запрос не будет принят к выполнению.
- Run query remotely: сетевое выполнение
- Fast query (no auxiliary tables): при выполнении запроса вспомогательные таблицы не создаются (answer, delete…)
- Generate auxiliary tables: вспомогательные таблицы создаются

Установленная в данном окне опция действует для всех выполняемых вами запросов на протяжении всего сеанса работы с Paradox

Способы отображение в окне Query нескольких образцов запросов

Вы можете задать способ расположения образцов запросов в окне Query.

- Window /Tile Tables: образцы располагаются последовательно (без наложения).
- Window / Cascade Tables: образцы располагаются один поверх другого.

Сделанный вами выбор режима отображения действует на протяжении всего сеанса работы с Paradox

Сохранение запроса

Чтобы сохранить запрос для последующего использования, выберите пункт меню File / Save или File / Save As. При попытке закрыть окно Query без сохранения Paradox сам предложит вам сохранить запрос и в этом случае необходимо ввести имя.

При сохранении запрос становится одним из объектов Paradox. Вы можете открыть его и поместить в окно Folder в виде иконки. Вы можете также разрабатывать формы и отчёты на основе сохранённых запросов вместо создаваемых ими таблиц Answer (см. Главу 9).

Составление запроса

От того, как составлен запрос, зависит результат его выполнения. Вы можете выбирать поля, группы записей, производить вычисления, связывать таблицы, изменять данные в ваших таблицах. Однако, прежде всего необходимо усвоить некоторые основные правила.

Форматы чисел в запросах

Когда вы вводите число в числовое или денежное поле:

- Не вводите символ $
- Не ставьте скобок для обозначения отрицательных значений (используйте знак минус -)
- Не разделяйте периоды числа запятыми (стандарт США) или точками (международный стандарт),

если используете операторы поиска по шаблону **..** или @

Paradox различает разделители периодов и десятичных знаков на основании указанного вами числового стандарта (американского или международного), учитывая положение разделителя и общий контекст. Неоднозначная интерпретация может возникнуть, например, если запятая в данной позиции могла бы означать оператор AND или точка может быть воспринята, как часть оператора .. .

В случае, если значение точки или запятой не однозначно, вы должны выделить этот символ кавычками или пробелами. Обычно, подобные ситуации происходят, когда вы используете шаблон с операторами **..** или @. В таких случаях лучше вообще не вводить разрядные разделители.

Если у вас установлен американский числовой формат:

- Paradox воспринимает точку в числовом поле как десятичный разделитель.
- Paradox интерпретирует первые две подряд стоящие точки как оператор .. . Три точки подряд будут считаться последовательно введёнными оператором **..** и десятичным разделителем, соответственно. Если же вы хотите в качестве десятичного разделителя указать первую из трех точек, то заключите ее в кавычки.
- Paradox воспринимает запятую в числовом поле в качестве разделителя периодов, если вы задаете поиск по точному совпадению и если запятая занимает «правильную» позицию. Если вы хотите, чтобы запятая воспринималась как оператор AND, а ее позиция дает повод сомневаться в однозначности такой интерпретации, то введите после запятой пробел или любой другой нечисловой символ (например, оператор сравнения, но не @ или часть оператора **..**).

Если у вас установлен международный числовой формат:

- Paradox воспринимает первую запятую в числовом поле (и расположенную между цифрами) в качестве десятичного разделителя.
- В числовом поле Paradox интерпретирует запятую в качестве оператора AND, если за ней следует пробел или какой-либо другой нечисловой символ (например, оператор сравнения, но не @ или часть оператора **..**).
- Paradox воспринимает точку в числовом поле как разделитель периодов, если вы используете поиск по точному совпадению и если точка занимает «правильную» позицию.

Использование кавычек

Рис. 6.8 Условия выбора алфавитно-цифрового поля, содержащие кавычки

Если вы хотите ввести алфавитно-цифровую величину, содержащую зарезервированные слова или символы Paradox, заключите их в кавычки (« »). В таком случае эти слова или символы будут восприняты как обычные символьные значения. В таблице 6.22 в конце этой главы приведены все зарезервированные слова и символы Paradox.

Нет необходимости применять кавычки для выделения пробелов между словами, входящими в какое-либо одно значение, однако они нужны при использовании символов, имеющих в Paradox специальное назначение, таких как точки, звездочки и т.д.

В случае, когда кавычки сами являются частью символьного значения, перед ними должен ставиться обратный слэш (\) (рис. 6.8).

Если вводимое вами значение содержит обратный слэш, то он должен быть удвоен (\\).

Включение в запрос полей

Рис. 6.9 значки, которые можно поместить в поле.

При составлении запроса вам необходимо указать поля, которые должны быть включены в таблицу Answer. Для этого их нужно отметить одним из следующих способов:

- Щелкните мышью включатели нужных полей
- В текущем поле нажмите клавишу F6

Все они имеют свое специальное назначение, описываемое в

последующих разделах. Чтобы выбрать один из этих значков:

- Нажмите на поле правую кнопку мыши и выберите нужный значок. При выполнении этой операции в крайнем левом поле указанный значёк установится для всей таблицы.
- Сделайте текущим нужное поле и нажимайте Shift+F6, пока не появится нужный значок.

Для удаления значка щелкните мышью соответствующий включатель или, на-ходясь в нужном поле, нажмите F6.

Значок V

только уникальные (неповторяющиеся) значения поля. При этом они будут отсортированы в порядке возрастания - от A до Z

Пример. Использование значка

Предположим, вы хотите увидеть (в алфавитном порядке) значения поля City таблицы Customer.

1. Откройте окно Query с запросом CUSTOMER.DB.
2. Удалите все условия выбора и значки, нажав Ctrl+Del, в данном случае - в полях Name и Phone.
3. Отметьте этим значком поле City одним из указанных выше способов.
4. Нажмите на SpeedBar кнопку Run Query или выберите пункт меню Query / Run, чтобы выполнить запрос.

Значок V+

Этот значок используется при необходимости получить все значения поля, включая повторяющиеся. При этом поля появляются в таблице Answer в таком же порядке, как и в таблице-оригинале (без сортировки). Если вы используете в каком-либо поле образца запроса этот значок, правила сортировки, задаваемые другими значками в остальных полях, перестают выполняться. Причина этого заключается в том, что Paradox не может одновременно выполнять полностью взаимоисключающие условия.

Помните, что несмотря на возможность ставить любые типы значков в BLOB-полях, обрабатываться эти поля будут, как помеченные значком +, поскольку Paradox не может ни сортировать эти поля, ни устанавливать их уникальность.

Пример. Использование значка

Предположим, вы хотите увидеть все значения поля City из таблицы CUSTOMER, включая повторяющиеся.

1. Откройте окно Query с образцом запроса CUSTOMER.DB, как в предыдущем примере.
2. Отметьте поле City этим значком.
3. Нажмите на SpeedBar кнопку Run Query или выберите пункт меню Query / Run, чтобы выполнить запрос.

Значок V ↓

Используйте данный значок для сортировки значений в убывающем порядке (от Z до A).

Пример. Использование значка

1. Откройте окно Query с образцом запроса CUSTOMER.DB, как в предыдущем примере
2. Отметьте поле City этим значком
3. Нажмите на SpeedBar кнопку Run Query или выберите пункт меню Query {Run, чтобы выполнить запрос.

Значок V G

Значок этого типа используется для задания группы записей в SET-запросах. Он позволяет группировать записи по значениям полей, не включая сами поля в таблицу Answer (см. Главу 7).

Включение всех полей

Если в таблицу Answer надо включить все поля оригинальной таблицы, воспользуйтесь включателем, расположенным под именем образца запроса (крайний левый столбец таблицы запроса). Выбранный вами значок появится во всех полях образца запроса.

Переименование полей таблицы Answer с помощью оператора AS

Поля, отмеченные вами в образце запроса, включаются в таблицу Answer под своими оригинальными именами. Если в таблицу Answer включены поля с совпадающими именами, то Paradox

переименует их как имя_1, имя_2 и т.д.

Вы можете создавать в таблице Answer поля, отсутствующие в исходных таблицах. Так, например, с помощью оператора CALC можно ввести в конец таблицы Answer новое вычисляемое поле. Имя ему Paradox присваивает автоматически, в соответствии с формулой, по которой производятся вычисления.

Если вы хотите переименовать какое-либо поле в таблице Answer, воспользуйтесь оператором AS (рис. 6.10). Введите условия запроса (если таковые имеются) в поле, затем введите оператор AS и новое имя. В случае вычисляемого поля введите as и новое имя сразу за оператором CALC и формулой.

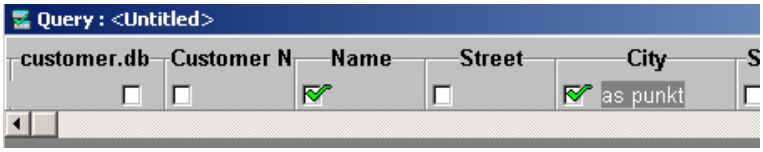


Рис. 6.10 Переименование поля в таблице Answer с помощью оператора AS

Помните, что оператор AS изменяет имена полей только в таблице Answer, но не в исходных таблицах.

Таблица 6.3 Использование значков и оператора AS с типами полей Paradox-таблиц

Оператор	A	N	$	D	S	M	F	B	G	O
V	+	+	+	+	+	+*	+*	+*	+*	+*
V+	+	+	+	+	+	+	+	+	+	+
V ↓	+	+	+	+	+	+*	+*	+*	+*	+*
V G	+	+	+	+	+					
AS	+	+	+	+	+	+	+	+	+	+

Таблица 6.4 Использование значков и оператора AS с типами полей dBASE-таблиц

Оператор	A	N	$	D	S	M
V	+	+	+	+	+	+*
V+	+	+	+	+	+	+
V ↓	+	+	+	+	+	+*
V G	+	+	+	+	+	
AS	+	+	+	+	+	+

* Вы имеете возможность поставить значки V или V ↓ в BLOB-полях, но они будут интерпретированы как V+, т.к. Paradox не может установить идентичность их значений и отсортировать.

Селекция записей

В большинстве случаев в результате запроса вы хотите получить только те записи, которые удовлетворяют определённым условиям. Эти условия задаются в полях образца запроса. В таблицу Answer попадут лишь те записи, значения полей которых удовлетворяют введённым вами условиям выбора

Точные совпадения

Если вы хотите увидеть записи с определенным значением какого-либо поля, просто введите это значение в соответствующем поле образца запроса.

Пример. Селекция по точному совпадению

1. Предположим, вы хотите увидеть магазины, которые находятся в США
2. Откройте окно Query с запросом .CUSTOMER. DB.
3. Включите поля Name и Country. Введите U.S.A. в поле Country.
4. Выполните запрос.

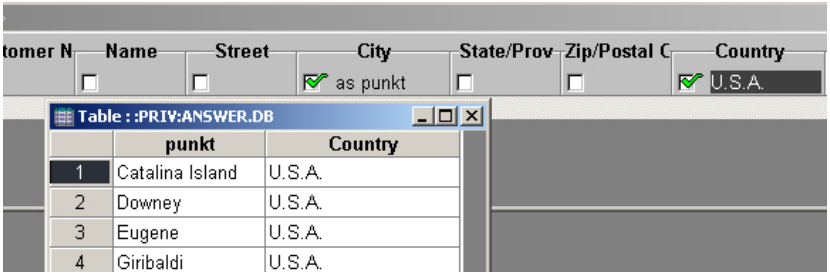


рис.6.11 запрос с определенным условием

Помните, что при задании поиска по точному совпадению данные в полях проверяются как по значению, так и по регистру. Так, если вы зададите поиск записей со значениями tools вместо Tools, ни одна запись со значением Tools в поле Equipment Class не попадёт в таблицу Answer.

Вы можете задавать поиск одновременно по нескольким полям (за исключением BLOB-полей). При вводе условий выбора в полях типа memo или форматированное memo вы должны использовать оператор .. (см. раздел «Поиск по шаблону» ниже в этой главе.).

Оператор LIKE

Оператор LIKE можно использовать при поиске алфавитно-цифровых значений, содержащих типографские опечатки или допускающих различное написание, как, например, Cafilornia вместо California или grey вместо gray.

Оператор LIKE ставится перед значением, которое вы хотите найти. При этом регистр, в котором оно записано, не имеет значения, как, впрочем, и регистр самого ключевого слова LIKE. На рисунке 6.12 показано использование оператора LIKE для поиска схожих алфавитно-цифровых значений.

Помните, что при работе с BLOB-полями оператор LIKE неприменим так же, как и при поиске по шаблону.

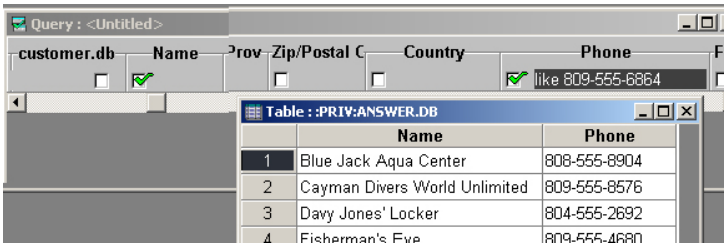


Рис. 6.12 Использование оператора LIKE

При использовании оператора LIKE имейте ввиду следующее:

- Первый символ в вашем образце должен соответствовать первому символу искомого значения (хотя регистр не имеет значения). Например, like Kalifornia не совпадает с California, a like California - совпадает.
- Если пример содержит от половины до двух третей или больше символов искомого значения, то вероятно, что поиск будет успешным, например: like Ion, like Idn, like Ind, like loo укажут на величину London; в то время хек like lo и tike In - нет.

Часто, когда запрос не дает ожидаемых результатов, использование оператора LIKE помогает обнаружить ошибки в написании алфавитно-цифровых значений.

Оператор NOT

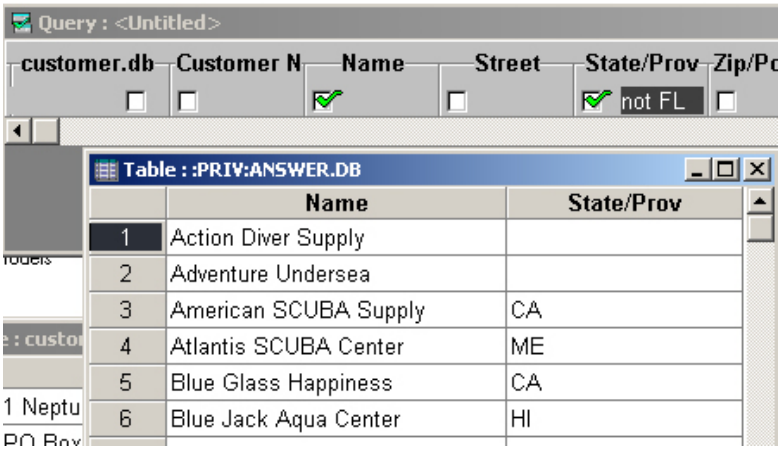


Рис. 6.13 Использование оператора NOT.

Позволяет выбрать записи, не содержащие определенных значений в некоторых полях. Для этого используется оператор NOT. Он ставится перед значением, появления которого в полях таблицы Answer вы хотите избежать.

Оператор NOT может стоять перед точными значениями, диапазонами, шаблонами и другими операторами (о диапазонах и шаблонах рассказывается ниже в этой главе). Если вы используете поиск по точному совпадению, то должны ввести это значение так, как оно встречается в исходной таблице (обращая внимание на написание и регистр). Регистр же самого ключевого слова NOT не имеет значения. Пример запроса с оператором NOT показан на рисунке 6.13

Оператор BLANK

Оператор BLANK используется при поиске записей, не содержащих никаких значений в определённых полях. В некоторых случаях вам может понадобиться найти подобные записи, чтобы внести в них данные, которые ранее были не известны.

Чтобы включить в таблицу Answer записи, не содержащие значений в определённом поле, введите в это поле запроса оператор BLANK (рис. 6.14).

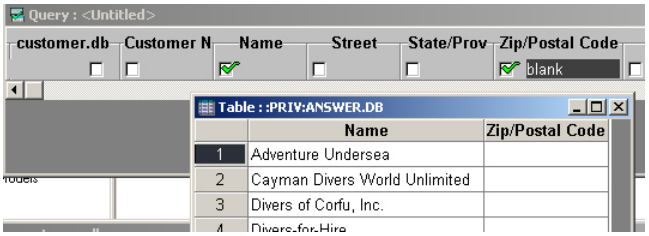


Рис. 6.14 Использование оператора BLANK

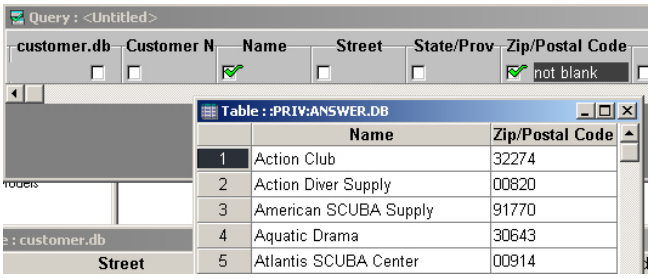


Рис. 6.15 Совместное использование операторов NOT и BLANK

Совместное использование операторов NOT и BLANK

Совмещая операторы NOT и BLANK, вы можете получить записи, содержащие непустые значения в определенном поле (рис. 6.15).

Поиск по шаблону

В Paradox имеется два оператора, которые вы можете использовать для поиска значений по шаблону. Дополняя оператор LIKE, они существенно расширяют ваши возможности.

Помните, что при использовании операторов шаблона с датами и числами вы должны придерживаться некоторых специальных правил (см. разделы «Использование операторов шаблона при работе с датами» и «Форматы чисел в запросах» в этой главе).

Оператор @

Оператор @ заменяет любой символ (букву или цифру). При задании шаблона вы можете использовать любое количество этих операторов. Если вам известно количество символов в искомой величине, вы можете заменить их таким же количеством символов @ вместо того, чтобы использовать оператор **..** . Как и в случае оператора LIKE, регистр написания остальных букв, введённых с оператором @, не имеет значения.

Обратите внимание, что данный оператор не может использоваться один без каких-либо других символов при работе с полями типа memo или форматированное memo. В этом случае вам придётся кроме оператора @, представляющего отдельные символы, воспользоваться также оператором .. .

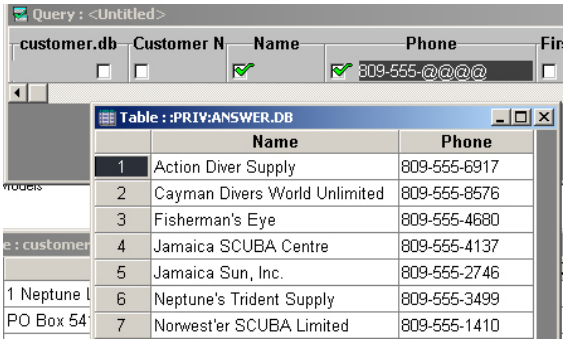


Рис. 6.16 Использование оператора @

Возможные результаты применения оператора представлены в табл 6.5.

Таблица 6.5 Применение оператора @

Шаблон	Найденные значения
m@@e	Mike, more, made
wom@n	woman, women
S@@@@	Smith, Smyth, scent
19@2	1932, 1952, 1992

Оператор ..

Оператор .. заменяет собой последовательность символов любой длины, включая пробелы. Регистр букв при этом не имеет значения. Ставится спереди или сзади шаблона и указывает, что информация с этой стороны значения не имеет.

Пример. Применение оператора

Предположим, что вы хотите найти магазины, в названиях которых есть слово DIVE. Если бы вы применили оператор LIKE и напечатали like dive в поле NAME запроса Customer, то вы получили бы только названия, начинающиеся со слова DIVE, причем те, в которых это слово составляет не менее половины всех букв. Вам же надо найти магазины, в названиях которых слово DIVE может стоять где угодно.

1. Откройте окно Query с запросом CUSTOMER.DB.
2. Включите поле NAME
3. Введите в этом поле ..dive..

4. Выполните запрос.

При работе с полями типа memo или форматированное memo поиск по шаблону с использованием оператора .. -единственно возможный вариант. Поиск по точному совпадению такого поля потребовал бы ввода в образец запроса всего значения поля. Поэтому Paradox запрещает подобные действия. Тем не менее, вы можете применять оператор @ в полях типа memo или форматированное memo, совмещая его, однако, с оператором .. .

В таблице 6.6 приведены возможные результаты применения оператора ..

Таблица 6.6 Применение оператора ..

Шаблон	Найденные значения
G..	Grant, gigantic, Georgia
g..t	Grant, gross weight
..T	hat, Elm st
..e..s	Thomas Edward Willis, ros-es
7..5	7485, 70005
6/../71	6/01/71, 6/30/71

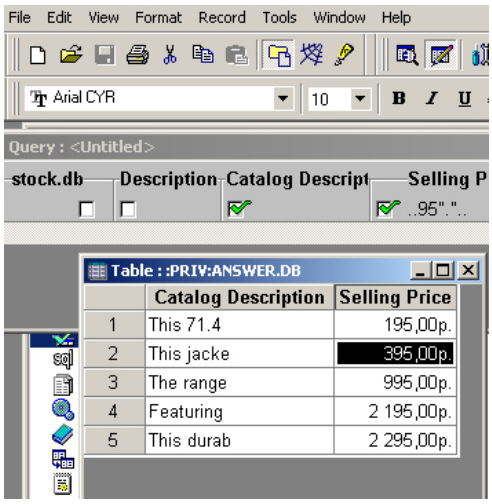


Рис. 6.17 Использование оператора .. в числовом поле

Особенности применения оператора .. при задании шаблона в поле числового типа были рассмотрены ранее в разделе «Форматы чисел в запросах». Рисунок 6.17 иллюстрирует запрос с использованием оператора .. и десятичного разделителя (в стандарте США) в числовом поле.

Рассмотрим результат выполнения запроса. В исходной таблице stock.db поле Selling Price имеет денежный формат. В таблице Answer это поле тоже имеет денежный формат. В запросе десятичный разделитель – точка, а в Answer - запятая. Это обусловлено тем, что для поля Selling Price таблицы Answer выбран шрифт Arial CYR, а запрос использует другую кодировку.

Использование операторов шаблона при работе с датами

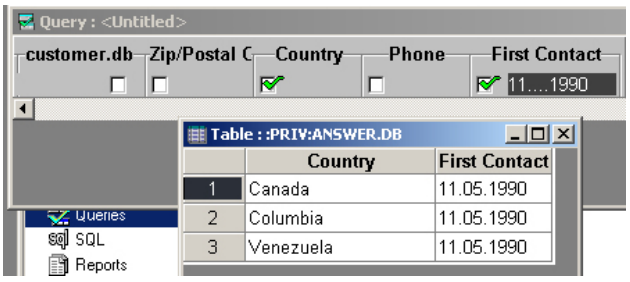


Рис. 6.18 Использование оператора .. в поле типа дата

При задании поиска по точному совпадению применительно к датам вы мо-жете использовать как соответствующие форматы Paradox, так и ваш собственный. Однако, когда вы задаете поиск по шаблону, заданный вами образец должен соответствовать формату даты, установленному в Borland Database Engine и в файле WIN.INI. (В обоих случаях форматы должны быть одинаковы.) Возможные форматы дат рассматривались в Главе 4. На рис 6.18 первая и последняя четвертая точка относятся к дате, а две средних - шаблон.

Оператор TODAY

Оператор TODAY используется в полях типа дата для обозначения текущей даты (в соответствии с системным календарем вашего компьютера). Данный опе-ратор особенно полезен при работе с арифметическими операторами Paradox (см. раздел «Использование арифметических выражений» ниже в этой главе).

Таблица 6.7 Типы полей Paradox-таблиц, допускающие использование поиска по точному совпадению и по шаблону

Оператор	A	N	$	D	S	M	F	B	G	0
Точное совпадение	+		+	+	+					
LIKE	+	+1	+1	+1	+1					
NOT	+	+	+	+	+	+	+	+	+	+
BLANK	+	+	+	+	+	+	+	+	+	+
..	+	+2	+	+	+	+	+			
@	+	+2	+	+	+	+3	+3			
TODAY				+						

1. Хотя вы можете использовать оператор LIKE в полях типа числовое и дата, операторы замены & и .. дают лучшие результаты.
2. Операторам замены в числовых полях Paradox ставит в соответствие только значащие цифры. Например, шаблону @@@,@ будет соответствовать значение 400.70, т.к. последний нуль не является значащим. В то же время это значение не будет отвечать шаблону @@<д>.@@ по той же причине.
3. В полях типа memo или форматированное memo вы можете использовать оператор @ только совместно с оператором ..

Таблиц» 6.8 Типы полей dBASE-ттблиц, допускающие испольяотани» поиска по точному совпадению и по шаблону

Оператор	C	F	N	D	L	M
Точное совпадение	+		+	+	+1	
LIKE	+2	+ 2	+2	+г	+3	
NOT	+	+	+	+	+	+
BLANK	+	+	+	+	+	+
..	+	+4	+4	+		+
@	+	+	+	+		+5
TODAY				+		

1. Точные совпадения, заданные для логических полей, могут представлять собой либо символы T и F либо целые слова True и False, причём регистр не имеет значения.
2. Хотя вы можете пользоваться оператором LIKE в полях типов числовое и дата, операторы шаблона .. и @ дают лучшие результаты.
3. Оператор LIKE не используется в логических полях dBASE-таблиц, т.к. логические величины представляют собой либо первые буквы Тир, либо целые слова True и False и не могут быть какими-либо частями этих слов (например, LIKE Tr или LIKE Fal).
4. В числовом поле dBASE-таблицы справа от десятичной точки могут распо-лагаться дополнительные нули, о которых вы можете не знать. Это происходит, когда в дробной части вы вводите большее количество цифр, чем может быть отображено в поле (в соответствии с заданным вами форматом отображения). В связи с этим рекомендуется ставить оператор .. в конец числового шаблона, даже если вы хотите сравнить только последние цифры. Например, шаблон ...95.. будет соответствовать всем числам, оканчивающимся на .95, независимо от количества десятичных знаков, определённого вами для данного поля; с другой стороны, шаблон ...95 подойдёт не ко всем значениям, особенно, если вы задали отображение меньшего количества десятичных знаков, чем определено форматом поля.
5. В полях типа memo вы можете использовать оператор @ только совместно с оператором .. .

Задание диапазонов

Оператор AND часто служит одним из основных компонентов при задании диапазона значений. Другим ключевым компонентом является оператор сравнения (иногда называемый оператором диапазона). В таблице 6.9 показаны операторы сравнения, которые можно использовать в запросах Paradox.

Операторы сравнения ставятся перед значениями, ограничивающими диапазон. Для задания ограниченных диапазонов вы можете комбинировать операторы сравнения, разделяя их запятыми. При работе с числовыми полями, когда есть возможность неоднозначной интерпретации, ставьте пробел после запятой, чтобы она была воспринята именно как оператор OR (см. раздел «Форматы чисел в запросах» выше в этой главе).

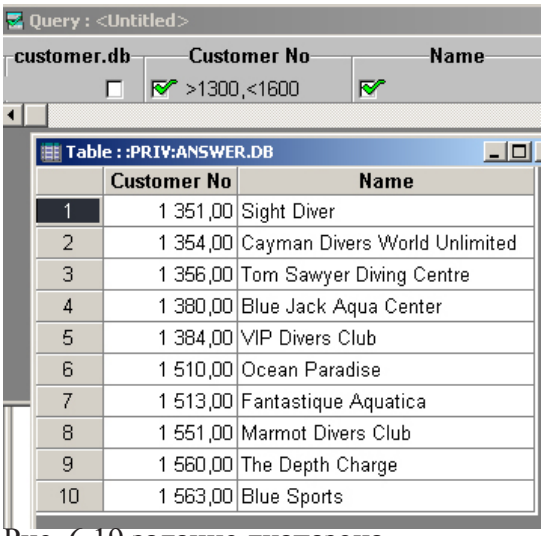


Рис. 6.19 задание диапазона

Таблица 6.9 Операторы сравнения

Оператор	A	N	$	D	S	M
=*	+	+	+	+	+	+
>	+	+	+	+	+	
<	+	+	+	+	+	
>=	+	+	+	+	+	
<=	+	+	+	+	+	

Таблица 6.10 полей Paradox – таблиц, допускающие применение операторов сравнения

Оператор	A	N	$	D	S	M	F	B	G	O
=*	+	+	+	+	+	+	+	+	+	+
>	+	+	+	+	+					
<	+	+	+	+	+					
>=	+	+	+	+	+					
<=	+	+	+	+	+					

Таблица 6.11 Типы полой dBASE-таблнц, допускающие применение операторов сравнения

Оператор	Значение
=	Равно (необязательный оператор)
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно

Условие OR

При необходимости выбора записей, удовлетворяющих одному из двух (или нескольких) условий, вы можете воспользоваться другим оператором Paradox - логическим ИЛИ (OR). Если значения-аргументы оператора OR относятся к одному полю, то введите их в этом поле, разделяя их ключевым словом OR..

Оператор служит для реализации условия «ИЛИ». На рис.6.20 показан при-мер использования этого оператора.

Ранее в разделе «Использование кавычек» обсуждалась необходимость заключать в них все значения, совпадающие по написанию с зарезервированными символами и словами Paradox, когда они должны интерпретироваться как обыч-ные алфавитно-цифровые последовательности

Это правило действует и в отношении оператора OR

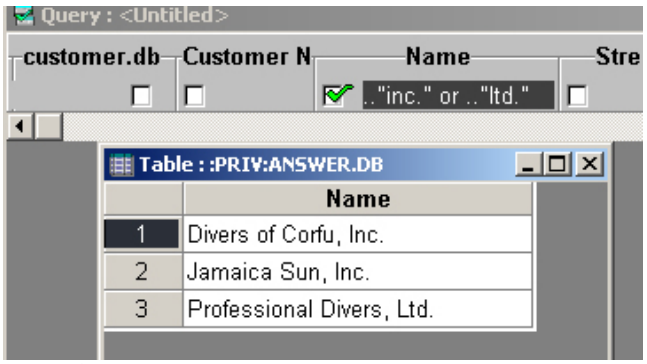


Рис. 6.20 использование оператора OR

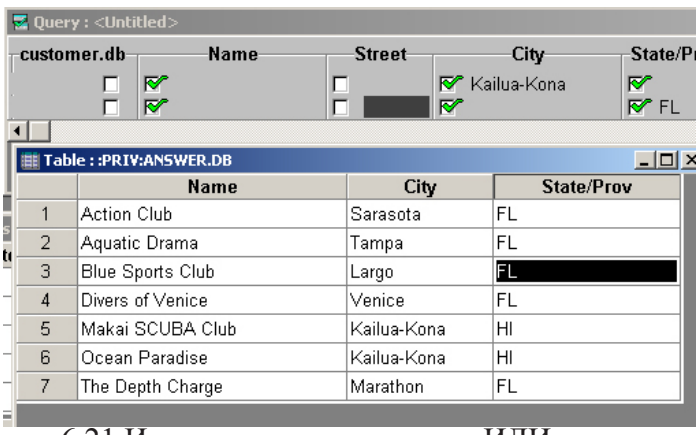


рис. 6,21 Использование оператора ИЛИ в разных полях.

При задании условия OR в разных строках образца запроса помните, что во всех строках должны быть включены одни и те же поля.

Пример. Задание условий OR в разных полях

Предположим, вы хотите получить список всех магазинов из таблицы Customer, расположенных или в городе KaiLua-Kona, или в штате FL

1. Откройте окно Query с запросом CUSTOMER.DB.
2. Включите поля Name, City и State/Prov.
3. Введите San Jose в поле City.
4. Нажмите клавишу F4 для вывода на экран второй строки образца запроса.
5. Включите на строке те же поля Name, City и State/Prov.
6. Введите на второй линии FL в поле State/Prov.
7. Выполните запрос.

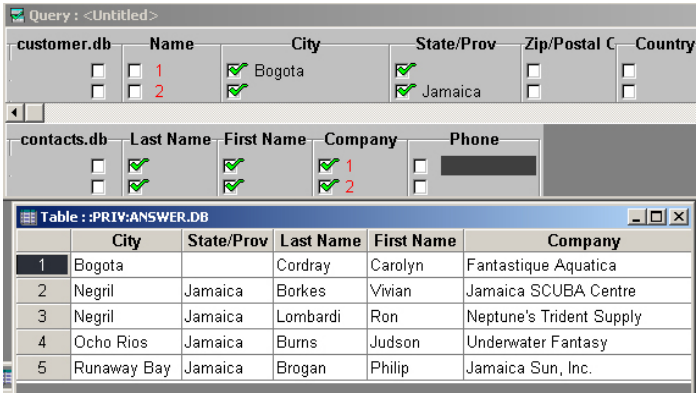
О том, как составлять многотабличные запросы, будет рассказано ниже в этой главе. Следующий же пример показывает, как задавать условие OR в разных полях и объединять таблицы. При составлении подобных запросов следует остерегаться случайного «самосвязывания» таблиц

Пример. Задание условия OR в многотабличном запросе

Предположим, необходимо получить перечень всех магазинов, расположенных либо в городе Nassau (Багамы), либо на Ямайке. Вы хотите знать также информацию по контактам с ними, для чего необходимо связать таблицу Customer с таблицей Contacts.

Откройте окно Query с запросами CUSTOMER.DB и CONTACTS.DB. воспользуйтесь кнопкой Join (или после нажатия клавиши F5 ввести любые символы для связки – в результате метка для связывания таблиц будет выделена красным цветом) Tables на SpeedBar для помещения элемент - примеров в поля Name и Company запросов CUSTOMER.DB и CONTACTS.DB.

1. Включите поля City и State/Prov в запросе CUSTOMER.DB.
2. Введите Nassau в поле City.
3. Нажмите 4 для вывода второй строки запроса CUSTOMER.DB.
4. Включите во второй строке те же поля, что и в первой.
5. Введите Jamaica в поле State/Prov второй строки.
6. Включите поля Last Name, First Name и Company в запросе CONTACTS. DB.
7. Нажмите 4 для вывода второй строки запроса CONTACTS.DB.
8. Включите в ней те же поля, что и на первой строке.
9. Вновь воспользуйтесь кнопкой Join Tables на SpeedBar для помещения элемент - примеров на второй строке в поля Name и Company таблиц CUSTOMER.DB и CONTACTS.DB.
10. Выполните запрос.



1 и 2 –элемент примера (связка таблиц). В полях, по которым производится связка информация должна быть идентична. Устанавливается после нажатия кл. F5 и выделена красным цветом.

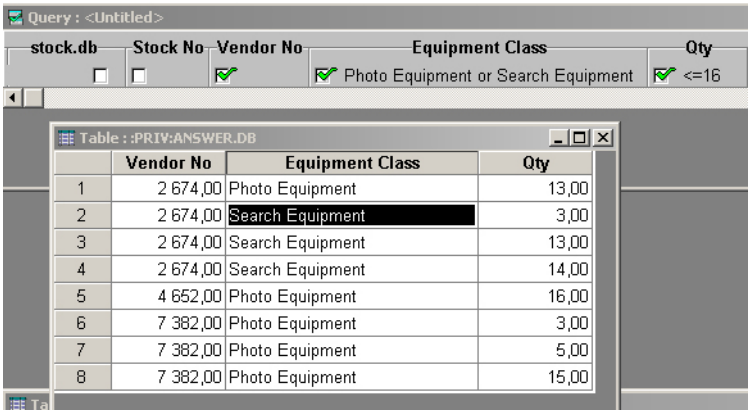
рис. 6.22 Оператор OR в многотабличном запросе

Комбинирование операторов сравнения и OR

Подобно разным типам оператора OR, вы можете комбинировать операторы AND и OR в одном запросе.

Пример. Использование комбинации сравнения и OR

Предположим, вам нужно узнать, продукция каких ваших поставщиков имеется на складе в недостающем количестве (вас) интересуют наименования Direct Sighting Compasses и Navigation Compasses, количество единиц которых на складе должно быть не менее 16).



1. Откройте окно Query с запросом STOCK. DB.
2. В поле Description введите Direct Sighting Compasses or Navigation Com-passes.
3. В поле Qty введите <=16.
4. Выполните запрос.

Таблица Answer будет содержать все записи из таблицы Stock со значениями поля Description либо Direct Sighting Compasses, либо Navigation Compasses и значениями поля Qty, которые не превышают 16.

Рис.6.23 Использование операторов сравнения и OR

Таблица 6.21 Типы полей Paradox-таблиц, допускающие применение операторов OR

Оператор	A	N	$	D	S	M	F	B	G	0
OR	+	+	+	+	+	+*	+*	+*	+*	+*

Таблица 6.22 Типы полей dBASE-таблиц, допускающие применение операторов AND (,) и OR

Оператор	C	F	N	D	L	M
OR	+	+	+	+	+	+*

* Вы можете использовать операторы OR в BLOB-полях, если условия выборки, заданные в этих полях, корректны для данного типа. Для полей типа memo и форматированное memo правильными условиями выборки могут являться шаблоны с операторами ... NOT.., BLANK, NOT BLANK. В бинарных, графических и OLE-полях можно применять операторы BLANK и NOT BLANK. Все условия выбора можно комбинировать с помощью оператора AND (,). С оператором OR можно применять только операторы BLANK и NOT BLANK.

Использование элемент - примеров

Элемент-пример применяется в запросе для представления некоторого конкретного значения. В простых запросах вы можете применять элемент - примеры в сочетании с зарезервированными словами и арифметическими операторами, чтобы производить над значениями полей определённые вычисления. В много-табличных же запросах элемент - примеры используются для связывания таблиц по общим полям. Элемент - примеры применимы в любых типах полей кроме BLOB-полей.

При вводе в запрос элемент - примеров вам необходимо помнить о том, что они:

- Могут содержать любые буквенные (от A до Z) или цифровые (от 1 до 9) символы
- Не должны включать в себя пробелы
- Не должны быть зарезервированными символами или словами

Вы можете сами создать элемент - примеры или же позволить Paradox сделать это за вас. Если вы пользуетесь цветным монитором, то элемент - примеры будут выделены цветом (например, красным), отличающим их от остального текста. В случае же монохромного монитора выделение производится подсвечиванием. В данном руководстве для выделения используется прямоугольная рамка вокруг элемент-примера.

Задание элемент-примеров

Когда вы вводите элемент-пример в одно поле однотабличного запроса, вы можете сделать это вручную и дать ему значащее имя:

1. Выберите нужное поле.
2. Нажмите F5.
3. Введите символы, составляющие элемент-пример.

При нажатии Spacebar или вводе запрещённого для элемент-примеров символа (например, запятой, тире, символа подчёркивания) Paradox воспримет ваши действия как сигнал к окончанию формирования элемента: последующие символы будут изображаться обычным цветом (или с обычной яркостью в случае монохромного монитора). Сигналом к окончанию формирования элемент-примера является также переход к другому полю, строке или образцу запроса.

Элемент-примеры в качестве значения

Элемент-пример может быть применен в условиях выбора, использующих какое-либо уже хранящееся в таблице значение. При этом он заменяет извлекаемое каждый раз для каждой обрабатываемой запросом записи конкретное значение.

Пример. Использование элемент-примера в качестве значения

Предположим, вы хотите знать, какие магазины расположены в том же городе, что и магазин VIP Divers Club. Вместо того, чтобы сначала выяснять, что это за город, а затем искать в нем интересующие вас магазины, можно провести весь поиск за один прием.

1. Откройте окно Query с запросом CUSTOMER. DB.
2. В поле Name введите VIP Divers Club.
3. В поле City нажмите F5 и введите city в качестве элемент - примера для замены настоящего названия города, где расположен магазин VIP Divers Club.
4. Нажмите клавишу i для вывода второй строки запроса.
5. Включите во второй строке поля Customer No, Name и City.
6. В поле City второй строки снова нажмите F5 и введите city для получения всех записей с теми же значениями поля City, что и в записи с VIP Divers Club.
7. Выполните запрос.

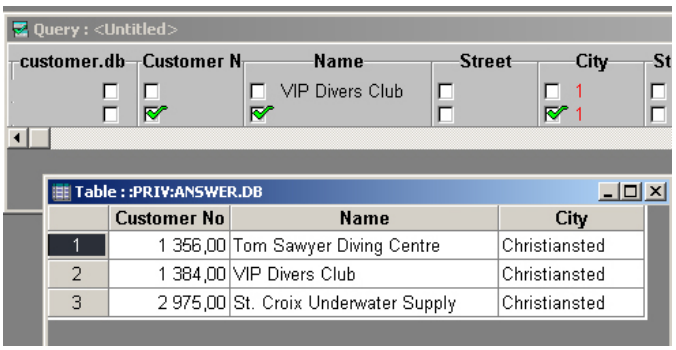


Рис.6,24 Использование элемент - примера в качестве значения

Вторая строка запроса обращается к записи со значением VIP Divers Club в поле Name таблицы Customer. Значение поля City в этой записи заменено эле-мент - примером. Этот же элемент-пример (точнее, его значение) используется в первой строке для получения всех записей с совпадающими значениями поля City.

Использование элемент - примера при задании диапазонов значений

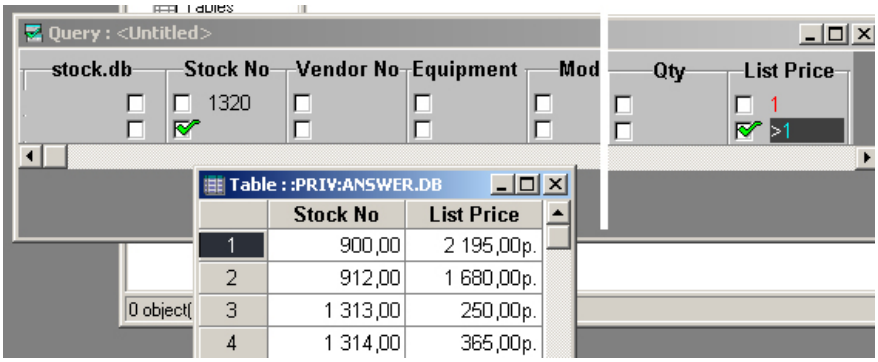


Рис. 6.25 Использование элемент - примера при задании диапазонов значений

Вы можете также пользоваться в запросах элемент - примерами для поиска значений из некоторого диапазона.

Пример. Использование элемент - примера в диапазонах

Предположим, вы хотите получить список товаров со стоимостью выше, чем стоимость товара под номером 1320 (Air Regulator model G-200B).

1. Откройте окно Query с запросом STOCK. Об.
2. В поле Stock No введите 132C.
3. Нажмите F5 в поле List Price и введите cost в качестве элемент - примера для замены значения этого поля в записях, содержащих код изделия 1320.
4. Нажмите клавишу 4 для вывода второй строки запроса.
5. Включите поля Stock No и List Price второй строки.
6. На второй строке в поле List Price введите >.
7. Вновь нажмите F5 и введите cost Выражение > cost будет указывать на записи со значениями в поле List Price, превышающими данное значение в записи с кодом изделия 1320. (Пробел между > и cost введен лишь для удобства чтения.)

8. Выполните запрос.

Использование элемент - примера в выражениях с данными типа дата

Вы можете вводить элемент - примеры в выражения, содержащие значения типа дата. Следующий пример демонстрирует использование элемент - примера в арифметическом выражении с оператором <.

Пример. Использование элемент - примера в выражениях с данными типа дата

Предположим, вам нужен список заказов, которые были выполнены менее, чем через 30 дней после заказа номер 1010.

1. Откройте окно Query с запросом ORDERS.DB.
2. В поле Order No введите 1010.
3. В поле Chip Date нажмите F5 и введите data в качестве элемент - примера.
4. Нажмите клавишу 4- для вывода второй строки запроса.
5. Во второй строке включите поля Order No и Ship Data.
6. Во второй строке в поле Chip Data введите < , вновь нажмите F5 и напеча-тайте data
7. Введите пробел, завершив создание элемент - примера. Затем введите +30. Выражение <data+30 укажет на заказы, выполненные не позднее 30 дней после заказа 1010. (Пробелы между оператором <, элемент - примером data, знаком + и числом 30 введены лишь для удобства восприятия. Знак +, напечатанный сразу после элемент-примера, также означал бы
8. окончание элемент-примера).
9. Выполните запрос.

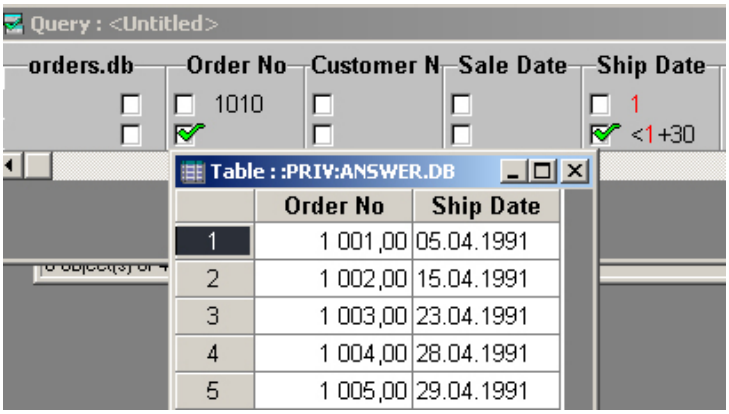


Рис. 6.26 Использование элемент - примера в выражениях с данными типа дата

Использование операторов LIKE и NOT с элемент-примерами

Следующий пример иллюстрирует применение элемент - примеров совмест-но с операторами LIKE и NOT.

Пример. Использование операторов LIKE и NOT с элемент - примерами

Предположим, вам нужно извлечь из таблицы Contacts записи с немного различающимся написанием значения в поле Last Name. Вы можете искать такие записи с помощью оператора LIKE, найдя сначала каким-либо образом одну них. Однако, у вас есть возможность провести весь поиск за один приём за счет совместного использования операторов LIKE и NOT с элемент - примерами.

1. Откройте окно Query с запросом CONTACTS.DB.
2. В поле Last Name нажмите F5 и введите элемент-пример name.
3. Нажмите клавишу 4 для вывода второй строки запроса.
4. Во второй строке включите поля Last Name и First Name.
5. В поле Last Name второй строки введите like и пробел.
6. Вновь нажмите F5 и введите элемент-пример name.
7. Введите запятую, завершая формирование элемент - примера затем введи-те оператор not и пробел.
8. Вновь нажмите F5 и введите элемент-пример name.
9. Выражение like name, not name укажет на записи, значения в полях Last Name которых будут схожи, но не идентичны. Они должны иметь не менее по-

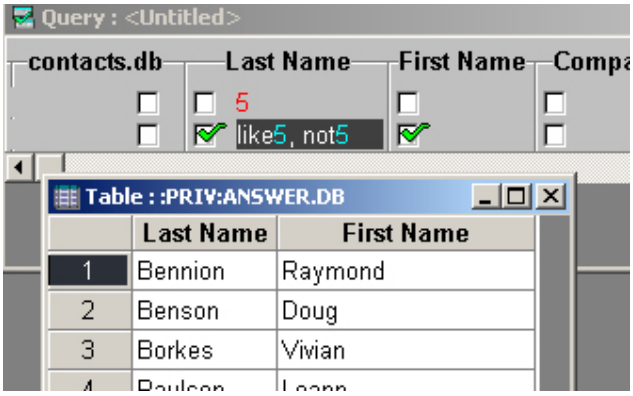


Рис. 6.27 Использование операторов LIKE и NOT с элемент - примерами

ловины одинаковых букв в одной и той же последовательности. (Пробелы по обе стороны от запятой введены лишь для удобства восприятия.)

10. Выполните запрос.

Использование элемент - примеров в запросах по нескольким таблицам

В одном запросе вы имеете возможность обратиться к нескольким (максимальное количество - 24) таблицам (см. раздел «Добавление и удаление образцов запросов» ранее в этой главе).

Чтобы составить запрос по нескольким таблицам, вы должны связать образцы запросов между собой с помощью элемент - примеров. Связь таблиц в запросе осуществляется по их общим полям, т.е. полям, присутствующим в каждой из связываемых таблиц и содержащим однородную информацию. Например, таблицы Customer и Orders содержат поле Customer No с идентификационными номерами клиентов. Так как данное поле в обеих таблицах содержит однородную информацию (имя поля не имеет значения), то оно может быть использовано для связывания этих таблиц.

Использование SpeedBar для задания элемент - примеров

Вместо «ручного» способа ввода элемент - примеров, для связи нескольких таблиц вы можете применить более удобное средство - кнопку Join Tables на SpeedBar.

Нажав кнопку Join Tables, поместите курсор мыши в нужное поле и щелкните левой клавишей. Элемент-пример сформируется в этом поле автоматически. Затем переместите курсор в соответствующее поле другого образца запроса, который должен быть связан с первым, и также щелкните его мышью. В этом поле появится такой же элемент-пример, а кнопка Join Tables выключится. Первая пара элемент - примеров получит имя EGO/, вторая - EG02 и т.д.

Поместите элемент - примеры в общие поля каждой пары связываемых таблиц. Чтобы произошло связывание, поля должны быть совместимых типов (но не обязательно одинаковых: например, числовое и денежное, а также memo и форматированное memo являются вполне совместимыми) и должны содержать совместимые данные.

Пример. Использование SpeedBar для задания элемент - примеров.

Предположим, вам необходимо узнать, какие магазины разместили у вас заказы. Однако, таблица Orders показывает только идентификационный номер клиента, но не название его магазина: оно хранится в таблице Customer, Возникает задача связать таблицы Orders и Customer no их общему полю Customer No, чтобы извлечь из них информацию как о размещённых заказах, так и о названиях магазинов заказчиков.

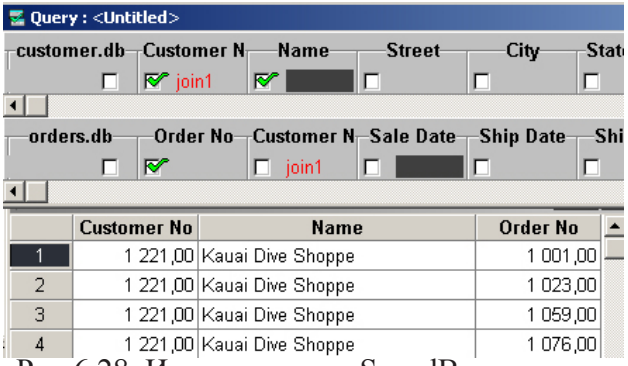


Рис.6,28 Использование SpeedBar для задания элемент - примеров

1. Откройте окно Query с запросами CUSTOMER.DB и ORDERS.DB.
2. В запросе CUSTOMER.DB включите поля Customer No и Name.
3. В запросе ORDERS.DB включите поле Order No. (Если вы включите в ней поле Customer No, то в таблице Answer появятся два таких поля - из таблиц Customer и Orders. В этом случае полю из таблицы Orders, как второму образцу запроса, будет присвоено имя Customer No_1. Значения обоих полей будут одинаковы.)
4. Нажмите на SpeedBar кнопку Join Tables.
5. 0 запросе CUSTOMER.DB щелкните поле Customer No. В нем появятся символы элемент-примера EG01 (отмеченные другим цветом или подсвеченные).
6. В запросе ORDERS.DB щелкните поле Customer No. В нем также появится элемент-пример.
7. Выполните запрос

Использование многотабличных документов для задания элемент-примеров

Paradox позволяет связать таблицы в запросе автоматически, используя уже существующие многотабличные запросы, формы или отчеты (см. Главу 9). Заданные в них связи между таблицами переносятся в составляемый запрос вместе с самими таблицами. Для этих же целей можно использовать ранее составленный многотабличный запрос, если новый запрос составляется по тем же таблицам.

Пример. Использование многотабличной формы для связывания таблиц в запросе

Предположим, у вас есть многотабличная форма, связывающая таблицы Customer и Orders по полю Customer No и таблицы Orders и Lineitem - по общему полю Order No (рис.6,30)

1. Откройте окно new/Query.
2. В окне Select File из списка Type выберите пункт Forms. Paradox заменит список таблиц на список доступных форм.
3. Выберите SUMMARY.FSL.

Paradox вводит в окно Query запросы по таблицам Customer, Orders и Lineitem и связывает их так, как они связаны в форме Summary, помещая соответствующие пары элемент-примеров в общих полях.

Paradox автоматически включает первое основное связующее поле Customer No в образце CUSTOMER.DB, поскольку оно является ключевым полем главной таблицы (Customer) формы.

Paradox также автоматически помещает оператор включения I после элемент - примера в поле Customer No запроса CUSTOMER.DB, т.к. связь между таблицами Customer, Orders и Lineitem в форме Summary является внешней связью (см. Главу 7).

4. В запросе CUSTOMER.DB включите поле Name.
5. В запросе LINEITEM.DB включите поля Stock No, Qty и Total.
6. Выполните запрос.

Таблица Answer покажет вам всех клиентов, а также какой товар, в каком ко-личестве и на какую сумму был ими приобретен. Обратите внимание, что одна из таблиц (ORDERS.DB) не выполняет никаких функций, кроме связывания двух других таблиц - CUSTOMER.DB и LINEITEM.DB (в таблице Answer не присутствует ни одного поля из таблицы ORDER&DB).

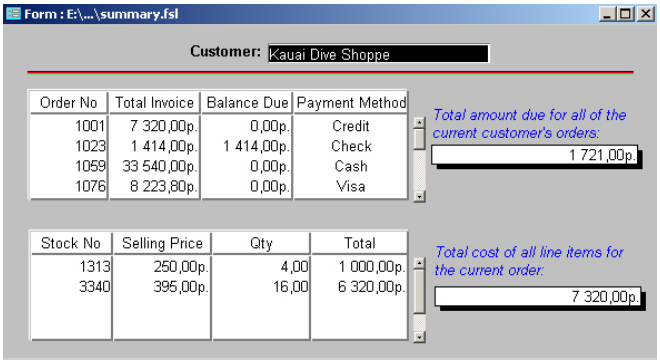


Рис. 6.29 форма summary.fsl

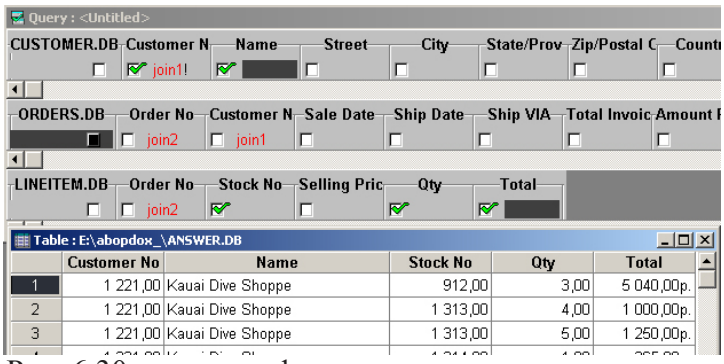


Рис. 6.30 запрос по форме и результат

Использование элемент - примеров в условиях выбора

В многотабличном запросе вы можете задать любое количество условий выбора и в любом образце запроса. Единственное требование к многотабличному запросу - чтобы все таблицы в окне Query были связаны с помощью элемент - примеров.

Пример. Использование условий выбора и элемент - примеров

Предположим, вы хотите получить список магазинов за пределами Калифор-нии, разместивших заказы на товары стоимостью от $500.00 до $1,500.00 и получивших свои заказы через транспортные агентства Federal Express или Emery.

1. Откройте окно Query со связанными запросами CUSTOMER.DB, ORDERS.DB и LINEITEM.DB. (см. предыдущий пример).
2. Удалите оператор ! после элемент - примера в поле Customer No запроса CUSTOMER.DB.
3. В поле Name запроса CUSTOMER.DB введите as Shop для переименования этого поля в таблице Answer.

4. Включите в запросе CUSTOMER.DB поле State/Prov.
5. В поле State/Prov запроса CUSTOMER.DB введите not CA, чтобы указать на магазины за пределами Калифорнии.
6. Включите в запросе ORDERS.DB поле Order No.
7. Включите там же поле Ship VIA.
8. В поле Ship VIA введите FedEx or Emery .
9. В запросе LINEITEM.DB выключите поля Stock No, Qty и Total.
10. В таблице LINBTEM.DB включите поле Selling Price. (Если вы включите еще и поле Order No, то в таблице Answer появятся два таких поля - из таблиц Orders и Lineitem. В этом случае полю из таблицы Lineitem (третьего образ-ца запроса) будет присвоено имя Order No_1. Значение обоих полей будут одинаковы.)
11. В запросе LINEITEM.DB в поле Selling Price введите >=500, <=1,500 (вы должны обязательно ввести пробел после оператора AND (,)).
12. Выполните запрос.

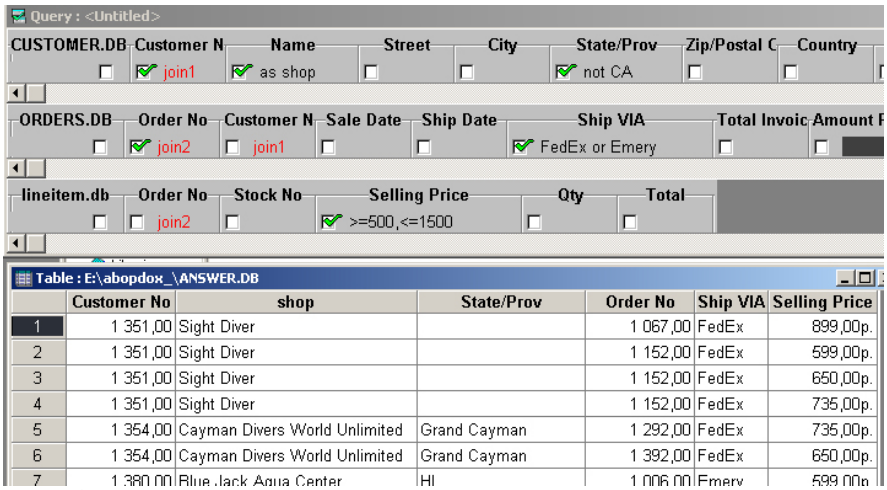


Рис. 6.31 Использование элемент - примеров в условиях выбора

Процедуры составления запросов по нескольким таблицам и по одной таблице весьма схожи. Как и в случае одной таблицы, если вы хотите задать несколько альтернативных условий в одном поле, используйте оператор OR. Если же речь идет о разных полях, введите условия на разных строках запроса.

Помните, что к элемент-примерам оператор OR неприменим. Выражение Qty or Price, где Qty и Price -элемент-примеры, не является логическим вопросом и приводит к сообщению об ошибке.

Арифметические выражения в запросах

Арифметические операторы применяются для составления арифметических выражений со значениями, содержащимися в полях запроса. Допустимые арифметические операторы перечислены в таблице 6.14.

Таблица 6.14 Арифметические операторы

Оператор	Значение
+	Сложение (или значений)
-	Вычитание
*	Умножение
/	Деление
( )	Группирующие скобки

В числовых полях Paradox (числовое, короткое числовое, денежное) и dBASE (числовое и с плавающей точкой) можно использовать любые арифметические операторы. Оператор сложения + можно также использовать в алфавитно-цифровых полях для слияния значений. Используя арифметические операторы с полями типа дата, можно выполнять, например, следующие операции:

- Прибавить некоторое количество дней к дате
- Вычесть некоторое количество дней
- Вычислить количество дней между двумя датами

Таблицы 6.15 и 6.16 демонстрируют возможности применение арифметических операторов в полях Paradox- и dBASE-таблиц, а в таблице 6.17 приведены некоторые примеры арифметических операций над датами с использованием оператора TODAY.

Таблица 6.15 Использование арифметических операторов с различными типами полей Paradox-таблиц

Оператор	A	N	$	D	S
+	+	+	+	+	+
-		+	+	+	+
*		+	+		+
/		+	+		+
( )	+	+	+	+	+

Таблица 6.16 Использование арифметических операторов с различными типами полей dBASE-таблиц

Оператор	C	F	N	D	L	M
+	+	+	+	+		
-		+	+	+		
*		+	+			
/		+	+			
( )	+	+	+	+		

Таблица 6.17 Арифметические действия над датами с использованием оператора TODAY

Выражение	Значение
<TODAY	Поиск дат, предшествующих сегодняшнему дню
<TODAY-90	Поиск дат, предшествующих сегодняшнему дню минус 90 дней
TODAY+30	Поиск дат, следующих на 30 дней дальше сегодняшнего

Вычисления в запросах

Помимо возможности извлечь нужные данные из таблиц вы можете также производить над ними вычисления. Использование оператора CALC позволяет:

- Составлять и редактировать математические выражения
- Комбинировать значения из нескольких полей из одной или нескольких таблиц
- Комбинировать значения полей с константами
- Создавать новые поля и помещать в них вычисленные значения

Если вы в своем запросе используете оператор CALC, в таблице Answer по-является дополнительное поле (по умолчанию - в конце таблицы), содержащее результаты вычислений. Paradox автоматически присваивает этому полю имя в соответствии с формулой, по которой производились вычисления, однако, вы можете изменить его с помощью оператора AS.

В условиях выбора вы можете определить те записи, над полями которых вы хотите произвести вычисления, а элемент-примерами заменить значения этих полей в формуле. Вы можете написать арифметическое выражение в любом поле запроса.

Следует заметить, что нет необходимости включать в запросе поле, в которое вы вводите CALC-выражение, т.к. его наличие заставляет Paradox автоматически создавать новое поле в таблице Answer.

Вычисление новых числовых значений

CALC-выражение может содержать:

- Константы (например, 154 или 12/91).
- Элемент-примеры
- Арифметические операторы (например, +, -, *, / и ())

Используйте скобки 0 для группирования операторов и установки последова-тельности выполнения действий. В выражениях без скобок сначала выполняются умножение и деление, а потом - сложение и вычитание. Действия с одинаковым приоритетом выполняются слева направо.

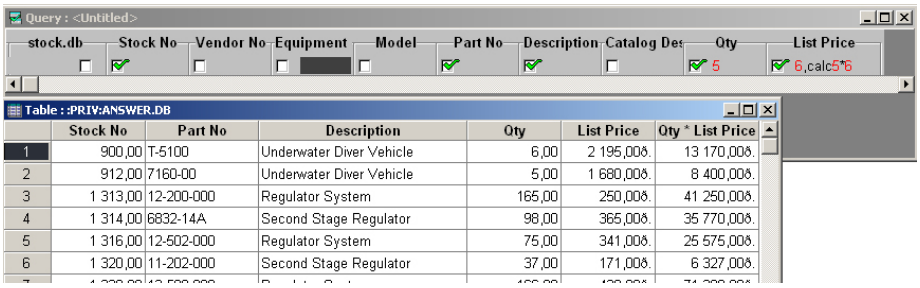
Пример. Вычисление новых числовых значений с помощью оператора CALC

Предположим, вам необходимо узнать общую стоимость каждого вида товара, хранящегося на складе. Для этого вам надо в таблице Stock перемножить зна-чения полей Qty (количество) и List Price (цена).

1. Откройте окно Query с запросом STOCK.DB.
2. Включите поля Stock No, Part No, Description, Qty и List Price.
3. В поле Qty нажмите F5 и введите элемент-пример Qty.
4. В поле List Price нажмите F5 и введите элемент-пример ListPrice
5. Теперь, после замены элемент - примерами тех значений, с которыми вы собираетесь работать, в

любом поле запроса можно ввести CALC-выражение Из соображений наглядности (и так как вы уже находитесь в этом поле) введите CALC-выражение в поле List Price.

6. После элемент - примера ListPrice поставьте запятую, чтобы закончить формирование элемент - примера и отделить его от CALC-выражения.
7. Введите calc и пробел.
8. Нажмите F5 и введите Qty, пробел (конец элемент - примера), *(оператор умножения) и еще один пробел. (Пробелы также делают запись более наглядной.)
9. Нажмите F5 и введите ListPrice.
10. Выполните запрос.



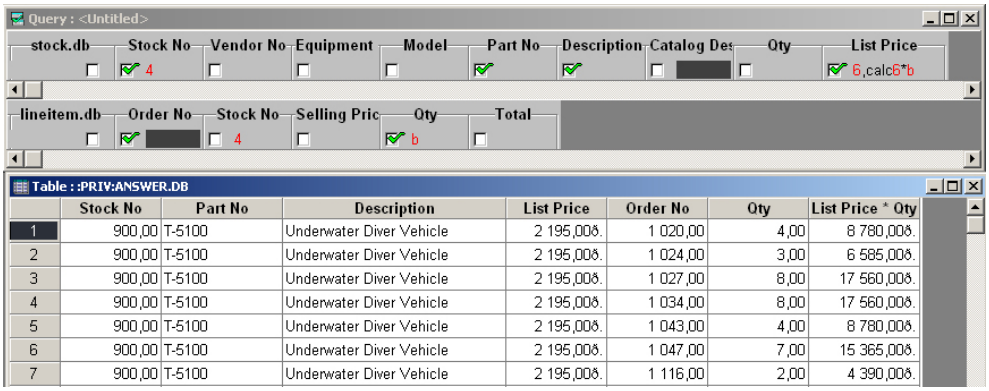
	Stock No	Part No	Description	Qty	List Price	Qty * List Price
1	900,00	T-5100	Underwater Diver Vehicle	6,00	2 195,00$.	13 170,00$.
2	912,00	7160-00	Underwater Diver Vehicle	5,00	1 680,00$.	8 400,00$.
3	1 313,00	12-200-000	Regulator System	165,00	250,00$.	41 250,00$.
4	1 314,00	6832-14A	Second Stage Regulator	98,00	365,00$.	35 770,00$.
5	1 316,00	12-502-000	Regulator System	75,00	341,00$.	25 575,00$.
6	1 320,00	11-202-000	Second Stage Regulator	37,00	171,00$.	6 327,00$.

рис. 6.32 Вычисление новых числовых значений с помощью оператора CALC

Предположим, вы хотите узнать общую стоимость размещенных заказов, исходя из значений поля List Price (таблица STOCK.DB), а не поля Selling Price (таб-лица UNEFTEM.DB). Вы должны перемножить значения полей List Price и Qty для каждого вида товара.

Количество заказанного товара (Qty) может быть получено связыванием таблиц Stock и Lineitem, значения же List Price находятся в таблице Stock.

1. Откройте окно Query с запросом STOCK.DB. (см. предыдущий пример)
2. Добавьте в окно Query запрос LINEITEM.DB.
3. Выключите поле Qty запроса STOCK.DB и удалите из него элемент-пример
4. Включите в запросе LINEITEM.DB поля Order No и Qty.
5. В поле Qty запроса LINEITEM.DB (но не в поле Qty запроса STOCK.DB) нажмите F5 и введите элемент-пример Qty.
6. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать запросы STOCK.DB и LINEITEM.DB по полю Stock No.
7. Выполните запрос



	Stock No	Part No	Description	List Price	Order No	Qty	List Price * Qty
1	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 020,00	4,00	8 780,00$.
2	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 024,00	3,00	6 585,00$.
3	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 027,00	8,00	17 560,00$.
4	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 034,00	8,00	17 560,00$.
5	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 043,00	4,00	8 780,00$.
6	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 047,00	7,00	15 365,00$.
7	900,00	T-5100	Underwater Diver Vehicle	2 195,00$.	1 116,00	2,00	4 390,00$.

рис. 6.33 Вычисления над числовыми значениями из различных таблиц

Использование оператора CALC с алфавитно-цифровыми значениями

С помощью операторов CALC и + вы можете производить различные операции с алфавитно-цифровыми значениями (но не значениями BLOB-полей). Например, вы можете взять значения из нескольких полей и соединить их, поместив в одно (также алфавитно-цифровое) поле.

Пример. Объединение алфавитно-цифровых значений с помощью оператора CALC

Предположим, вы хотите объединить данные из полей City, State и Zip табли-цы Customer в одно поле Address (полный адрес).

1. Откройте окно Query с запросом CUSTOMER.DB.
2. Включите поле Name.
3. В поле City введите элемент-пример City.
4. В поле State/Prov введите элемент-пример StateProv.
5. В поле Zip/Postal Code введите элемент-пример Zip.

Теперь, после замены элемент-примерами значений, с которыми вы собираетесь работать, вы можете ввести CALC-выражение в любом поле запроса.

6. В поле Country введите calc и пробел.
7. Нажмите F5 и введите City.
8. Затем введите +»,»+. (В кавычках после запятой должен быть пробел.)
9. Вновь нажмите F5 и введите StateProv.
10. Затем снова +» «+. (Между кавычками должен стоять один или два пробела.)
11. Вновь нажмите F5 и введите Zip.
12. На SpeedBar нажмите кнопку Answer Table Properties или выберите пункт меню Properties / Answer Table / Options.
13. В окне Answer TaЬIe Properties замените в текстовом окошке Answer Name ANSWER. DB на ADDRESS. DB, чтобы сохранять результаты запроса в новой таблице Address вместо временной таблицы Answer.
14. Выполните запрос.

Так как адреса магазинов за пределами США не соответствуют формату City-State-Zip, то вы можете изменить запрос, включив в него поле Country специально для таких магазинов.

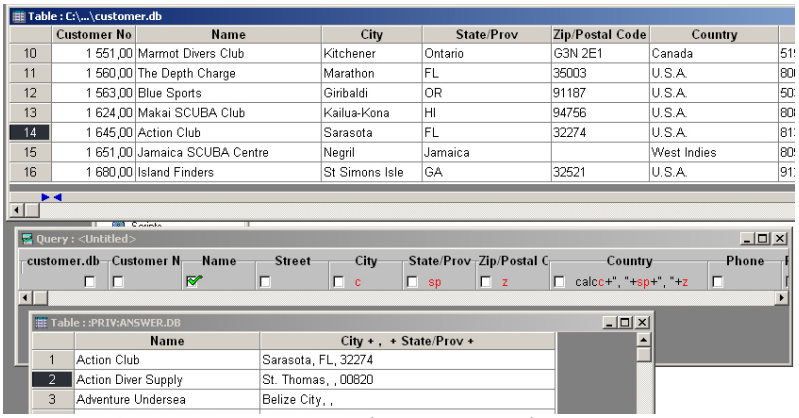


Рис. 6.34 Объединение алфавитно-цифровых значений с помощью оператора CALC

Создание в таблице Answer нового поля с постоянным значением

Дополнительное поле в таблице Answer может быть создано не только для отображения результатов вычислений, но и для хранения каких-либо постоянных значений - числовых, алфавитно-цифровых или дат. Чтобы задать поле с число-вой константой или датой, в любом поле запроса введите calc, пробел и значение константы. Для задания поля с алфавитно-цифровой константой введите calc с, пробел и константу в нужном регистре, заключенную в кавычки.

Новое поле таблицы Answer будет соответствовать имени вашей константы (вы также можете изменить имя поля с помощью оператора AS.) Если новое поле алфавитно-цифрового типа, то его размер установится в соответствии с количеством символов в константе.

Так как пробелы также являются константами, вы можете, введя оператор BLANK после оператора CALC, создать новое поле, оставив его пустым. В этом случае вы должны ввести CALC-выражение в том поле, типу которого должно соответствовать новое поле таблицы Answer. Такими полями могут являться числовые, денежные, типа дата или алфавитно-цифровые.

Пример. Создание нового поля таблицы Answer, содержащего константу

Предположим, вам необходимо «обзвонить» всех клиентов магазинов для вы-яснения спроса на товары. При этом желательно следить за тем, чтобы не позво-нить дважды одному и тому же клиенту. Для этого на основе данных таблицы Contacts вы решили создать новую таблицу Calls. В этой таблице вы хотите совместить поля Last Name и First Name таблицы Contacts, а также создать новое поле с алфавитно-цифровой константой Not called yet

1. Откройте окно Query с запросом CONTACTS.DB
2. Включите поля Company и Phone.
3. В поле Last Name введите элемент-пример LastName. В поле First Name введите элемент-пример FirstName.
4. В любом поле запроса введите формулу calc LastName + «,»+FirstName , где Last Name и First Name - элемент-примеры. Затем присвойте новому полю имя People to call, введя после формулы выражение as People to call.
5. CALC-выражение и оператор AS должны находится в одном поле. Если вы вводите их
6. в поля Last Name или First Name, которые уже содержат элемент-примеры, не забудьте

7. отделить элемент-пример от CALC-выражения запятой
8. Введите выражение calc «Not called yet» в любом поле таблицы запроса. Это приведёт к созданию в таблице Answer нового алфавитно-цифрового поля с именем Not called yet и таким же значением во всех записях. Поле будет иметь тип A14, где 14 - длина (в символах), необходимая для размещения константы Not called yet (Если вы вводите CALC-выражение в поле, уже содержащее некоторые условия выбора, отделяйте это выражение от условии запятой.)
9. Выберите пункт меню Properties / Answer Table / Options или нажмите на SpeedBar кнопку Answer Table Properties.
10. В текстовом окошке Answer Name окна Answer TaЫe Properties замените ANSWER.DB на CALLS.DB и нажмите OK
11. Выполните запрос.

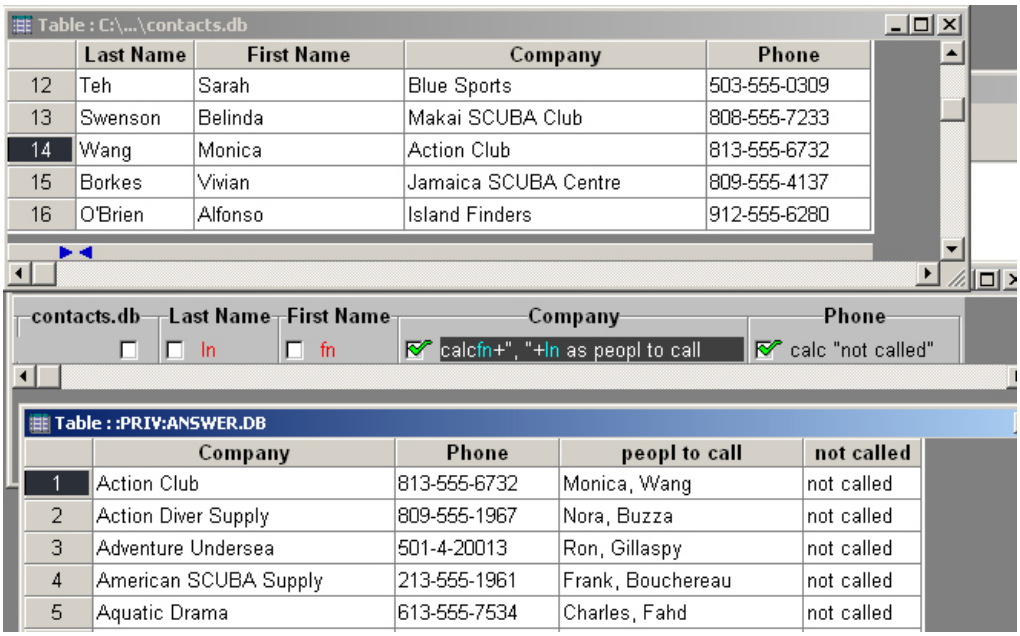


Рис. 6.35 Создание нового поля таблицы Answer, содержащего константу

Таблица 6.18 Типы Paradox-полей, допускающие использование элемент -примеров и оператора CALC

Операция	A	N	$	D	S
Элемент-пример	+	+	+	+	+
Вычисления над значениями полей с помощью оператора CALC	+*	+	+	+**	+

Таблица 6.19 Типы dBASE-полей, допускающие использование элемент-примеров и оператора CALC

Операция	C	F	N	D	L
Элемент-пример	+	+	+	+	+
Вычисления над значениями полей с помощью оператора CALC	+*	+	+	+**	

* Вы можете только объединить значения (+ совместно с оператором CALC) и помещать результат в новое алфавитно-цифровое или символьное поле.

** Вы можете только складывать и вычитать даты (+ и -совместно с операто-ром CALC), чтобы получить новое поле типа дата.

Изменение таблиц с помощью запросов

В запросах Paradox можно использовать следующие зарезервированные слова:

- INSERT: вставить записи в таблицу
- DELETE: удалить записи из таблицы
- CHANGETO: изменить определенные значения

INSERT-запрос

INSERT-запрос позволяет вставлять в таблицу-приемник записи из нескольких таблиц-источников. При этом таблица-приемник и источник(и) могут быть разных типов, например, Paradox и dBASE.

Чтобы составить INSERT-запрос, выполните следующие действия:

1. Введите в окно Query таблицы-источники и таблицу-приемник. (Если вы хотите вставить записи в новую таблицу, вы должны заранее создать ее.)
2. Свяжите таблицы элемент - примерами.
3. Для каждой таблицы-источника введите в нужных полях условия выбора.
4. В таблице-приемнике поместите слово Insert в крайней левой колонке (под именем таблицы), используя один из следующих способов:
    - установите курсор мыши под именем таблицы, нажмите левую клавишу, и, удерживая ее, выберите из меню операций запроса пункт Insert
    - Нажмите Spacebar из появившегося меню операций запроса выберите пункт Insert
    - Нажмите на клавиатуре клавишу i

    В той строке запроса, где находится оператор INSERT, не включайте ни од-ного поля, иначе Paradox фиксирует ошибку.
5. Выполните запрос.

Paradox вставляет записи таблицы-источника, заданные вашими условиями выбора, в таблицу-приемник.

Помните, что в незаполненные (не содержащие элемент - примеров) поля таблицы-приемника никакие значения из таблиц-источников внесены не будут. Так как BLOB-поля Paradox и memo поля dBASE не допускают применения элемент - примеров, то INSERT-запрос с такими полями не работает.

Таблица Inserted

INSERT-запрос создает в вашем личном каталоге временную таблицу Inserted. Она переписывается при каждом выполнении запроса и удаляется по окончании сеанса работы с Paradox. Чтобы сохранить содержащуюся в ней информацию, переименуйте ее.

Вместе с таблицей Inserted вы также можете сформировать таблицу Answer. Для этого включите какое-либо поле в отдельной строке таблицы-приемника. Если в этой строке вы введете еще и условия выбора, то записи в таблице Answer будут, разумеется, им удовлетворять, хотя ценность подобной таблицы окажется небольшой: информация, содержащаяся в ней, не имеет никакого отношения к операции Insert (см. раздел «Порядок операций в многофункциональных запросах» ниже в этой главе).

Вы можете использовать таблицу Inserted вместе с Delete-запросом (см. раздел «DELETE-запрос»), чтобы удалить из таблицы-приемника записи, введенные INSERT-запросом.

Таблица Errins

Если записи, вставляемые в таблицу-приемник, конфликтуют с ее системой ссылок или правилами проверки корректности данных (при выполнении INSERT-запроса не учитываются только правила соответствия данных шаблону), Paradox помещает такие записи во временную таблицу Errins. Нарушение системы ссылок может произойти, например, при попытке вставить в дочернюю таблицу запись, значение ключевого поля которой отсутствует в родительской таблице.

В любом случае Paradox создаёт таблицу Inserted, содержащую все записи, которые предназначались вами для вставки, и, при необходимости, таблицу Errins с записями, которые не удалось вставить в таблицу-приемник.

Пример: INSERT-запрос

Предположим, вы хотите выяснить, станут ли для вас дешевле международные телефонные переговоры, если вы воспользуетесь услугами другой телефон-ной компании. Однако, перед этим вы хотите узнать, сколько магазинов ваших клиентов расположены за пределами США.

Данный пример демонстрирует действие INSERT-запроса, который помещает всех зарубежных клиентов в новую таблицу Phoncall. В данном случае вы можете достичь результатов быстрее, используя значок **V** полях Name и Phone таблицы CUSTOMER. DB и сохраняя таблицу Answer под

именем Phoncall. Однако, применение V не всегда является более эффективной альтернативой INSERT-запросу.

Данный пример просто показывает, как может работать операция insert в составе более сложного запроса.

Откройте окно Query с запросом CUSTOMER. DB.

Создайте новую таблицу Phoncall (команда File / New) Table,) с двумя алфавитно-цифровыми полями Name и Phone . Присвойте им типы A3O и A15 соответственно – по аналогии с полями Name и Phone таблицы Customer. (Вы можете быстро создать таблицу Phoncall, позаимствовав для нее структуру таблицы Customer и уничтожив ненужные и переименовав нужные поля.)

1. Введите в окно Query запрос PHONCALL. DB.
2. В поле Name запроса CUSTOMER. DB введите элемент-пример name.
3. В поле Country запроса CUSTOMER.DB введите not U.S.A. .
4. В поле Phone запроса CUSTOMER.DB введите элемент-пример phone.
5. В запроса PHONCALL.DB установите операцию Insert.
6. В поле Client Name запроса PHONCALL.DB введите элемент- пример name.
7. В поле Phone Number запроса PHONCALL.DB введите элемент- пример phone.
8. Дайте команду File / Open / Table В окне Select File выберите PHONCALL.DB. Так как Phoncall была до этого пуста, ее записи будут в точности соответство-вать записям таблицы Inserted

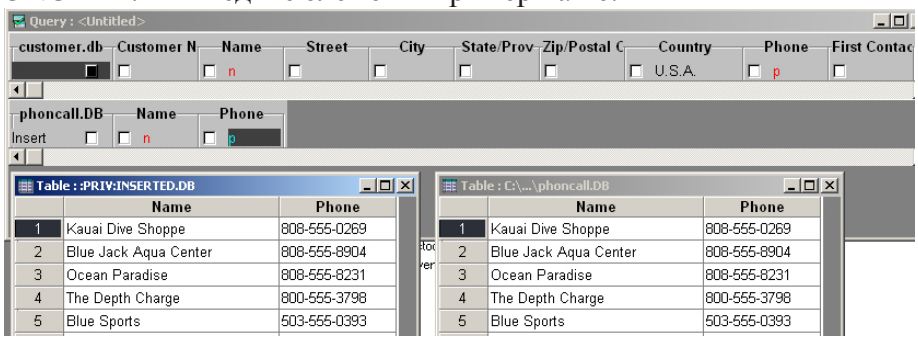


Рис. 6.36 INSERT-запрос

Пример. Вставка фрагментов текста с помощью INSERT-запроса

Предположим, вы хотите вставить некоторую запись в таблицу Contacts.

1. Откройте окно Query с запросом CONTACTS. DB.
2. В крайней левой колонке таблицы установите операцию Insert.
3. В поле Last Name введите Tot.
4. В поле First Name введите Lari.
5. В поле Company введите Shop 1.
6. В поле Phone введите 111-22-56.
7. Выполните запрос. Перед вами появится таблица Inserted.
8. С помощью команды File / Open / Table откройте таблицу CONTACTS. DB.

В конце таблицы Contacts вы увидите вставленную запись

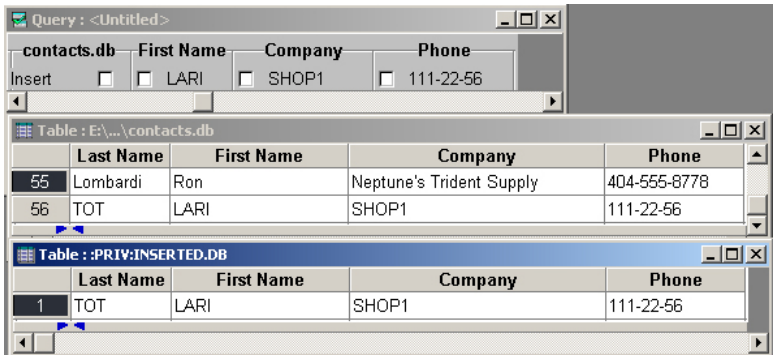


Рис. 6.37 Вставка фрагментов текста с помощью INSERT-запроса

DELETE-запрос

DELETE-запрос применяется для удаления из таблицы определённых записей. Им удобно пользоваться, когда ненужные записи имеют какие-либо общие элементы, которые вы можете описать в условиях выбора.

DELETE-запрос удаляет из таблицы целые записи, а не отдельные значения в их полях (см. «CHANGETO-запрос» ниже в этой главе).

Чтобы составить DELETE-запрос, выполните следующие действия:

1. Поместите в окно Query таблицу с удаляемыми записями и таблицы (если нужно), которые вы хотите связать с первой таблицей и использовать для задания условий исключения из нее записей.

2. В таблице с удаляемыми записями поместите слово Delete в крайней левой колонке (под именем таблицы), используя один из следующих способов:
   - Установите курсор мыши под именем таблицы, нажмите левую клавишу, и, удерживая ее, выберите из меню операций запроса пункт Delete.
   - Нажмите Spacebar и из появившегося меню операций запроса выберите пункт Delete.
   - Нажмите на клавиатуре клавишу d.

   В той строке запроса, где находится оператор Delete, не включайте ни одного поля, иначе Paradox фиксирует ошибку.
3. Введите условия выбора ненужных записей. Вы можете сделать это в нескольких полях одного запроса или в полях нескольких запросов, связанных элемент-примерами.
4. Учтите, если вы не укажете никаких условий выбора, Paradox удалит все записи таблицы.
5. Выполните запрос.

Paradox удаляет из таблицы, в которой установлена операция Delete, все записи, удовлетворяющие введенным вами условиям выбора.

Таблица Deleted

Delete-запрос создает в вашем личном каталоге временную таблицу Deleted, содержащую удаленные записи. Эта таблица перезаписывается при каждом выполнении DELETE-запроса и уничтожается по окончании сеанса работы с Paradox. Вы можете сохранить эту таблицу под другим именем с помощью коман-ды tools/ Utilities /Rename.

Наличие таблицы Deleted не лишает вас возможности создать также таблицу Answer: включите для этого нужные поля в отдельной строке запроса. Если вы к тому же введете условия выбора, то записи в таблице Answer будут, разумеется, удовлетворять этим условиям, но ценность подобной таблицы окажется небольшой: информация, содержащаяся в ней, не имеет никакого отношения к операции Delete (см. «Порядок операций в многофункциональных запросах» ниже в этой главе).

Вы можете восстановить удаленные записи, применив INSERT-запрос с таблицей Deleted в качестве источника. Если таблица не имеет ключа, то возвращенные записи независимо от их прежних позиций будут помещены в конец таблицы. Восстановление можно также произвести командой tools / Utilities / Add. Указанные способы восстановления - единственные, доступные в случае Paradox-таблиц (при работе с dBASE-таблицами вы можете, просматривая таблицу в режиме редактирования, включить опцию Record / Show Deleted и восстановить (по одной) все удаленные записи с помощью команды Record (Undelete.)

Таблица Errdel

При попытке удалить записи, отсутствие которых привело бы к нарушению системы ссылок, Paradox не производит удаления, а лишь помещает копии этих записей во временную таблицу Errdel. Нарушение системы ссылок может произойти, например, при попытке удалить из родительской таблицы запись, имеющую связи с записями, находящимися в дочерних таблицах. В этом случае удаление записи из родительской таблицы привело бы к потере связанных с ней записей в дочерних таблицах.

В любом случае Paradox создает таблицу Deleted со всеми записями, предназначенными для удаления, и, при необходимости, таблицу Errdel с записями, удаление которых привело бы к указанным выше конфликтам.

Пример. Удаление записи DELETE-эапросом

Предположим, один из магазинов вашего клиента прекратил свою деятельность и вы хотите исключить его из таблицы Contacts

1. Откройте окно Query с запросом CONTACTS.DB
2. В крайней левой колонке запроса установите операцию Delete

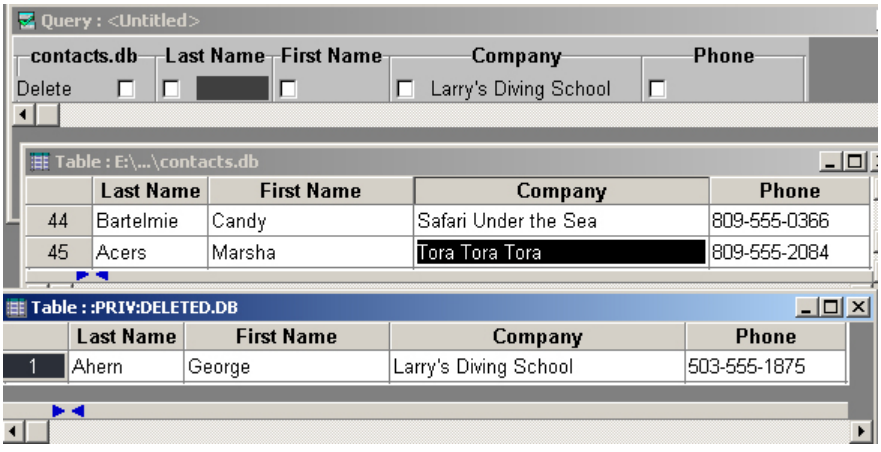


Рис. 6.38 Удаление записи DELETE-эапросом

3. В поле Company введите название магазина Larry's Diving School
4. Выполните запрос

Пример. Восстановление INSERT-запросом записи, удаленной DELETE-запросом

Предположим, вы передумали и, после удаления записи Larry's Diving School (см. предыдущий пример), решили указать другое лицо для контакта с клиентами закрывшегося магазина.

Простейшим способом восстановить удалённую запись была бы команда File / Utilities / Add. Однако, данный пример показывает иной способ. (В каждом конкретном случае следует выбирать способ, исходя из сложности операции, которую необходимо проделать, однако, всегда рекомендуется делать резервные копии таблиц на каждом шаге их преобразования.)

1. Откройте окно Query с запросом из предыдущего примера.
2. Удалите все прежние условия выбора в запросе CONTACTS. DB , нажав Ctrl+Del в любом его поле.
3. Добавьте в окно Query запрос DELETED. DB.
4. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать элемент -примерами запросы CONTACTS.DB и DELETED.DB по общему полю. .
   В крайней левой колонке запроса установите операцию Insert.
5. Выполните запрос

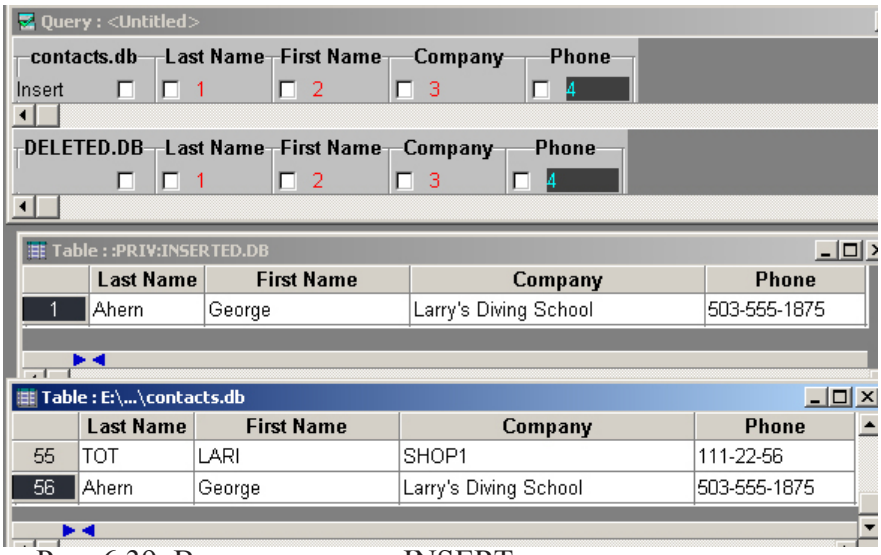


Рис. 6.39 Восстановление INSERT-запросом записи

CHANGETO-запрос

CHANGETO -запрос позволяет изменять значения полей в таблице, удовлетворяющих заданным вами условиям (это табличный аналог операции Search and Replace). CHANGETO- запрос бывает крайне полезен при однообразном изменении большого количества значений.

Чтобы изменить значения поля:

1. Введите значение, которое вы хотите изменить в том поле запроса, где оно встречается.
2. После изменяемого значения поставьте запятую
3. Затем введите changeto и пробел (регистр оператора CHANGETO не имеет значения).
4. Затем, после пробела введите новое значение. Вы можете повторить данную процедуру в других полях. Оператор CHANGETO должен находиться на одной строке с условиями выбора.
5. В той строке запроса, где находится оператор Delete, не включайте ни од-ного поля, иначе Paradox фиксирует ошибку.
6. Выполните запрос.

Таблица Changed

CHANGETO-запрос создает в вашем личном каталоге временную таблицу Changed с исходными оригиналами записей, изменённых в процессе выполнения запроса. Эта таблица перезаписывается при каждом исполнении CHANGETO-запроса и удаляется по окончании сеанса работы с Paradox. Вы можете сохранить ее под другим именем, воспользовавшись командой File / Utilites / Rename.

Чтобы в результате запроса получить еще и таблицу Answer, включите нуж-ные поля в отдельной строке запроса. Если вы зададите при этом некоторые условия выбора, записи в таблице Answer будут, разумеется, соответствовать этим условиям, однако ценность такой таблицы окажется небольшой: информация, содержащаяся в ней, не имеет никакого отношения к операции CHANGETO (см. раздел «Порядок операций в многофункциональных запросах» далее в этой главе).

С помощью таблицы Changed вы можете проверить правильность внесенных изменений. Если

вы изменили случайно не те записи, вы можете восстановить исходные значения следующим образом:

1. Выполните DELETE-запрос по таблице с неверно изменёнными записями. В качестве условий выбора введите новые значения. Неверные записи будут удалены.
2. Выполните INSERT-запрос, вставляя в исходную таблицу записи из таблицы Changed. (Если исходная таблица не имеет ключа, возвращённые записи окажутся в самом ее конце.)

Таблица Errchng

При попытке изменить с помощью CHANGETO-запроса значений ключевых полей (первичных или вторичных), которая привела бы к нарушению системы ссылок, Paradox помещает копии конфликтующих записей во временную таблицу Errchng в вашем личном каталоге. Такое нарушение может произойти, например, при изменении значений зависимых ключевых полей дочерней таблицы на значения, не содержащиеся в родительской таблице.

В любом случае Paradox создаёт таблицу Changed со всеми записями, которые предназначались для внесение в них изменений, и, при необходимости, таблицу Errchng с записями, которые не удалось изменить.

Пример. Поиск и изменение значений с помощью CHANGETO-запроса

Предположим, вы узнали, что George Ahern (лицо, через которое вы намеревались контактировать с бывшими клиентами закрывшегося магазина Larry's Diving School) нашёл работу в городе Savannah (штат Джорджия) в магазине The Human Gill Dive Shop. Вы хотите установить связь с этим человеком, чтобы попытаться получить заказ от его нового хозяина. Так как George Ahem работает теперь в другой компании и имеет другой номер телефона, вам необходимо внести соответствующие изменения в таблицу Contacts.

1. Откройте окно Query с запросом CONTACTS.DB.
2. В поле Last Name введите Ahern.
3. В поле First Name введите George.
4. В поле Company введите элемент-пример company.
5. В поле Phone введите элемент-пример phone.
6. В этом же поле поставьте запятую и введите оператор changeto, затем про-бел и 404-555-1451 - новый номер телефона господина George Ahem.
7. Выполните запрос.
8. Откройте таблицу Contacts.

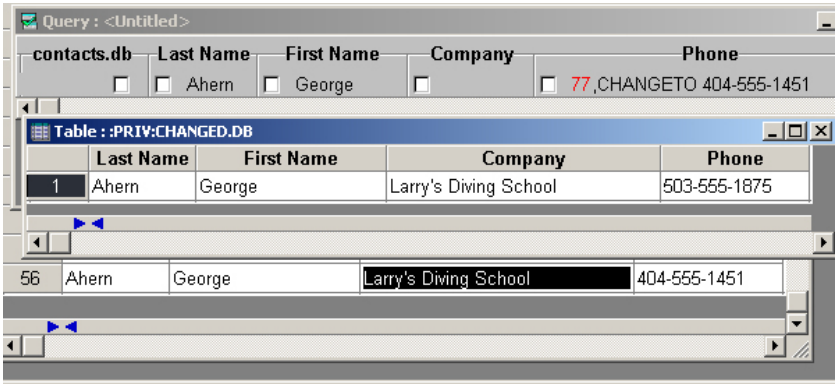


Рис. 6.40 Поиск и изменение значений с помощью CHANGETO-запроса

Восстановление таблицы после CHANGETO-запроса

Предположим, что после изменения записи George Ahern в таблице Contacts (предыдущий пример) выяснилось, что магазин Lyre's Diving School всеже продолжит свою работу. Более того, администрация этого магазина предложила George Ahem возвратиться на прежнее место работы. В этой связи вам надо восстановить запись с «координатами» господина George Ahern в ее прежнем виде.

В данном случае простейшим способом восстановить прежнюю информацию были бы действия, в точности противоположные предпринятым в предыдущем примере: вы могли бы составить CHANGETO-запрос, указав в полях Company и Phone в качестве условий CHANGETO прежние их значения. Однако, в случае, когда вы произвели достаточно большое количество изменений, удобнее пользоваться запросами DELETE и INSERT.

1. Откройте окно Query с запросом из предыдущего примера.
2. Удалите условия CHANGETO из полей Company и Phone, оставив содержание полей Last Name и First Name - Ahern и George, соответственно.
3. В крайней левой колонке запроса установите операцию Delete.

4. Выполните запрос.
5. Вернитесь в окно Query и добавьте в него таблицу CHANGED. DB.
6. Удалите запись George Ahern, нажав Ctrl+Del.
7. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать элемент-примерами таблицы CONTACTS.DB и CHANGED.DB.
8. В крайней левой колонке запроса установите операцию Insert.
9. Выполните запрос. Запись George Ahern будет восстановлена в таблице Contacts в ее прежнем виде.

Использование CHENGETO - запроса с элемент - примерами

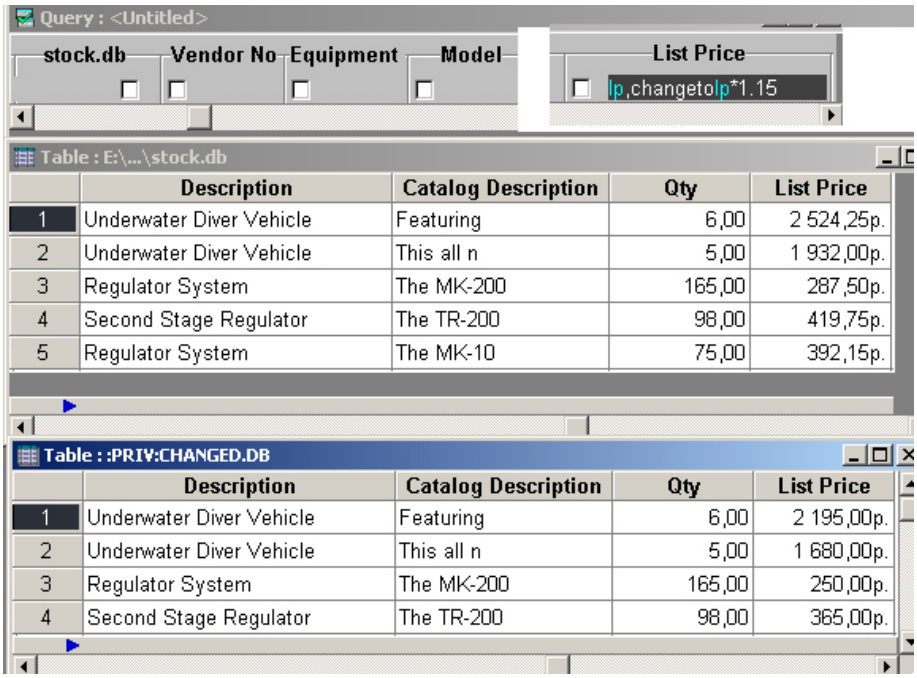


Рис. 6.41 Использование CHENGETO - запроса с элемент - примерами

Элемент - примеры могут быть использованы в CHANGETO-запросах, чтобы производить вычислений над значениями некоторых полей и заменять их на новые, вычисленные значения. (Если вы производите вычисления с использованием оператора CALC, то оригинальные значения остаются без изменений, а вычисленные значения помещаются в новое поле таблицы Answer.)

Пример (рис.6,41). Использование CHANCETO-запроса для вычисления и замены значений

Предположим, вы хотите увеличить цену всех ваших товаров на 15%.

1. Откройте окно Query с запросом STOCK.DB.
2. В поле List Prise введите элемент-пример ListPrise.
3. Поставьте запятую для отделения элемент-примера, введите changeto и поставьте пробел.
4. Введите еще один элемент-пример ListPrise.
5. Поставьте пробел для отделения элемент-примера и введите *1.15.
6. Выполните запрос.
7. Откройте таблицу Stock

Многотабличный CHANGETO-запрос

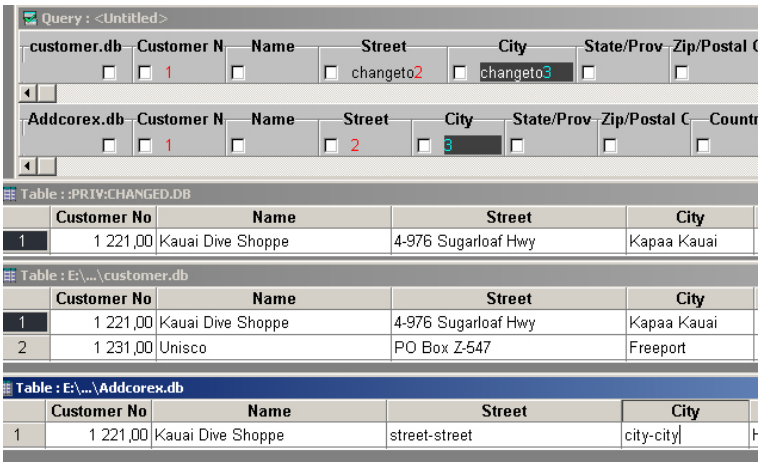


Рис. 6.42 Использование запроса CHANGETO для приведения в соответствие записей двух таблиц

Вы можете использовать CHANGETO-запрос для изменения записей одной таблицы так, чтобы они пришли в соответствие с системой ссылок другой таблицы.

Пример. Использование запроса CHANGETO для приведения в соответствие записей двух таблиц

Предположим, вы создаёте таблицу Addcorex для внесения уточнений в адреса магазинов - ваших заказчиков. Вы создаёте эту таблицу с системой ссылок на таблицу Customer по ключевому полю Customer No.

После ввода правильных адресов в таблицу Addcorex вы собираетесь заменить

на них прежние (невер-ные) значения в таблице Customer

1. Откройте окно Query с запросом CUSTOMER.DB.
2. С помощью команды File / New / Table создайте новую таблицу Addcorex со структурой, аналогичной таблице Customer. Для максимально аккуратного задания системы ссылок сделайте Customer таблицей-справочником для поля Customer No таблицы Addcorex и следите за тем, чтобы в этом поле не появились идентификационные номера заказчиков, которых нет в таблице Customer.
3. После создания таблицы Addcoreix введите в нее какие-либо пробные данные, например, новый адрес какого-нибудь магазина, расположенного в США. Пусть единственным изменением в отом адресе будет номер улицы. Вы можете затем восстановить информацию в первоначальном виде с помощью DELETE- и INSERT-эапросов, как было показано в одном из предыдущих примеров.
4. Добавоте в окно запрос ADDCOREX.DB.
5. С помощью кнопки Join Tables на SpeedBar свяжите элемент примерами оба запроса по полю Customer No.
6. С помощью кнопки Join Tables разместите пары элемент -примеров в полях Street, City обоих запросов.
7. Во всех перечисленных полях запроса CUSTOMER.DB введите changeto и пробел перед каждым элемент-примером.
8. Выполните запрос.

   Откройте таблицу Customer и убедитесь, что адрес заказчика в таблице изменился.

Порядок операций в многофункциональных запросах

Вы можете составлять запросы, изменяющие несколько таблиц одновременно. Если в окне Query открыто несколько образцов запроса, единственным и основным требованием для его выполнимости является связывание всех запросов посредством элемент-примеров.

Вы можете, например, создать запрос, удаляющий записи из одной таблицы, вставляющий записи в другую и изменяющий значения в третьей таблице. Вы можете также выполнить запрос, производящий все перечисленные действия в одной таблице.

Если вы собираетесь создать такой многофункциональный запрос, вам необходимо знать о порядке выполнения данных операций:

- Сначала отбираются записи, удовлетворяющие условиям выбора.
- Далее выполняются операции Insert в порядке их нахождения программой, т.е. сначала находящиеся в первом образце, затем - во втором и т.д.
- После этого (также в порядке нахождения) выполняются операции CHANGETO.
- Далее следуют операции Delete (в том же порядке).
- В заключение формируются все временные таблицы, в том числе таблица Answer, если вы включили какие-либо поля.
- Так как операции Delete выполняются после операций Insert, вы можете создать запрос, уничтожающий результаты своих же изменений (например, запрос сначала вставляет записи, а затем удаляет их из той же таблицы). Вообще, вы можете создавать довольно запутанные запросы, совмещая в них множество различных операций, однако, учтите, что восстановить данные при этом в их первоначальном виде может оказаться затруднительно.

Таблица 6.20 Типы Paradox-полей, допускающие использование запросов INSERT, DELETE, CHANGETO

Операция	A	N	$	D	S	M
INSERT	+	+	+	+	+	
DELETE		+	+	+	+	
CHANGETO	+	+	+	+	+	

Таблица 6.21 Типы dBASE-полвй, допускающие использование запросов INSERT, DELETE, CHANGETO

Операция	C	F	N	D	L	M
INSERT	+	+	+	+	+	
DELETE		+	+	+	+	
CHANGETO	+	+	+	+	+	

В таблице 6.22 приведены все операторы запросов Paradox. Вопросы, связанные с применением некоторых из них, рассматриваются в Главе 7.

Таблица 6.22 Операторы запроса

Оператор	Назначение
Зарезервированные символы	
V	Включение в таблицу Answer только неповторяющихся значений поля
V+	Включение в таблицу Answer всех (включая повторяющиеся) значений поля
V↓	Включение в таблицу Answer поля в порядке убывания значений
VG	Определение набора записей для групповых операций
Зарезервированные слова	
CALC	Включение в таблицу Answer вычисляемого поля
INSERT	Вставка записей с определёнными значениями и формирование временной таблицы
DELETE	Удаление записей с определёнными значениями и формирование временной таблицы
CHANGETO	Изменение определённых значений и формирование временной таблицы
SET	Определение набора записей для операций сравнения групп с набором
Арифметические операторы	
+	Сложение или объединение алфавитно-цифровых значений
-	Вычитание
*	Умножение
/	Деление
0	Группирование действий
Операторы сравнения	
=	Равно (не обязательный оператор)
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно
Операторы замены	
..	Последовательность любых символов
@	Любой символ
Специальные операторы	
LIKE	Подобно ...
NOT	Отлично от ...
BLANK	Пустое значение
TODAY	Текущая дата
OR	Логическое ИЛИ
,	Логическое И
AS	Переименование поля в таблице Answer
!	Включение записи в таблицу Answer, несмотря на отсутствие соответствующей записи в связанной таблице
Статистические операторы	
AVERAGE	Среднее значение
COUNT	Количество значений
MAX	Максимальное значение
MIN	Минимальное значение
SUM	Сумма значений

ALL	Статистические вычисления производятся над всеми записями группы, в т.ч. над повторяющимися значениями
UNIQUE	Статистические вычисления производятся только над неповторяющимися значениями в группах записей
Групповые операторы сравнения	
ONLY	Выбор групп, все записи которых принадлежат определённому набору
NO	Выбор групп, ни одна запись которых не принадлежит определённому набору
EVERY	Выбор групп, в которых присутствуют все записи определённого набора
EXACTLY	Выбор групп, совпадающих определённым набором записей

Глава 7. Сложные запросы

В Главе 6 обсуждались запросы к одиночным записям таблицы. Данная глава рассматривает запросы, работающие с группами и наборами записей. Кроме этого, вы узнаете, как при выполнении запросов устанавливать между таблицами включающую связь.

Выполнение запросов к группам записей

Используя возможности групповых запросов Paradox, вы можете:

- Выбирать из базы данных записи по групповым статистическим характеристикам (например, найти виды товаров, на которые поступило два и более заказов)
- Производить статистические вычисления внутри групп записей (например, вычислить средние значения сумм, на которые сделаны заказы, по каждой стране)
- Сравнивать характеристики группы записей с отдельными записями (например, кто из зарубежных клиентов разместил заказов больше, чем любой московский)

Чтобы отвечать на подобные вопросы, необходимо анализировать одновременно данные нескольких записей таблицы, предварительно сгруппировав их нужным образом.

Статистические операторы

Статистические (summary) операторы производят специальные вычисления над группами записей, которые вы задаёте, отмечая в запросе определённое поле (или поля). В Paradox существует пять статистических операторов:

- AVERAGE - усредняет значения в группе
- COUNT - вычисляет количество значений в группе
- MAX - определяет максимальное значение внутри группы
- MIN - определяет минимальное значение
- SUM - суммирует значения внутри группы

Таблицы 7.1 и 7.3 показывают, как используются статистические операторы с полями различных типов Paradox- и dBASE-таблиц.

Модификаторы статистических операторов

По умолчанию все статистические операторы кроме COUNT производят вычисления над всеми значениями в группе (COUNT по умолчанию подсчитывает только неповторяющиеся значения). Однако, вы можете использовать модификаторы статистических операторов:

- ALL означает, что вычисления производятся над всеми значениями внутри группы, включая дублирующиеся. Если вы хотите подсчитать полное количество значений, вы должны использовать оператор COUNT ALL.
- UNIQUE означает, что статистическая операция производится только над уникальными (неповторяющимися) значениями. В таких случаях вам придётся использовать данный модификатор со всеми операторами кроме COUNT.

Таблицы 7.2 и 7.4 показывают, как используются модификаторы статистических операторов с полями различных типов Paradox- и dBASE-таблиц.

Таблица 7.1 Использование статистических операторов с различными типами полей Paradox-таблиц

Оператор	A	N	$	D	S
AVERAGE		+	+		+
COUNT	+	+	+	+	+
MAX	+	+	+	+	+
MIN	+	+	+	+	+
SUM		+	+		+

Таблица 7.2 Использование модификаторов статистических операторов с раз-личными типами полей Paradox-таблиц

Модификатор	A	N	$	D	S
ALL	+	+	+	+	+
UNIQUE	+	+	+	+	+

Таблица 7.3 Использование статистических операторов с различными типами полей dBASE-таблиц

Оператор	C	F	N	D	L
AVERAGE		+	+		
COUNT	+	+	+	+	+
MAX	+	+	+	+	+
MIN	+	+	+	+	+
SUM		+	+		

Таблица 7.4 Использование модификаторов статистических операторов с различными типами полей dBASE-таблиц

Модификатор	C	F	N	D	L
ALL	+	+	+	+	+
UNIQUE	+	+	+	+	+

Селекция групп записей

Рис.7 1 Значки, находящиеся в полях запроса

Чтобы определить группы записей, кроме отметки полей, включаемых в запрос, необходимо использовать статистические операторы. Значки, находящиеся в полях запроса, выполняют следующие функции :

- Группируют записи по значениям в отмеченных полях
- Включают поля в таблицу Answer (как в простых запросах)

Поскольку статистические операторы и их модификаторы являются зарезервированными словами Paradox, регистр (верхний или нижний), в котором они записаны, не имеет значения.

Селекция записей по количеству

Следующий пример демонстрирует, как оператор COUNT используется для подсчета внутри групп записей неповторяющихся значений.

Пример. Подсчет записей в группе

Предположим, вы хотите узнать, в каких странах находится три и более ваших клиентов.

1. Откройте окно Query с запросом к таблице CUSTOMER.DB.
2. Включите в запрос поле Country. Значок по F6 сгруппирует записи по значениям в этом поле и включит его в таблицу Answer.
3. Введите count>=3 в поле Customer No. Данное выражение укажет Paradox подсчитать количество клиентов в каждой группе (стране) и выбрать те группы, в которых оно больше или равно 3 Таблица Customer имеет ключ по полю Customer No, поэтому все номера клиентов уникальны.
4. Выполните запрос.

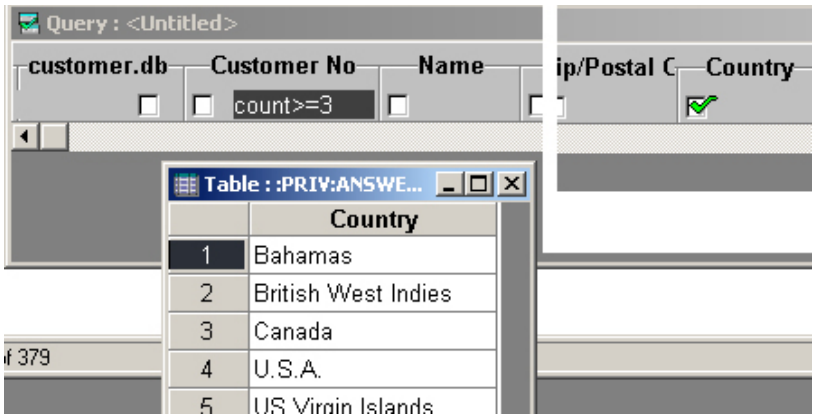


Рис. 7.2 Подсчет записей в группе

Селекция записей по сумме значений

Следующий пример демонстрирует, использование оператора SUM для подсчёта суммы значений внутри групп записей.

Пример. Суммирование в группе

Предположим, вы хотите узнать, кто из клиентов разместил у вас заказов на сумму более $5000.

1. Откройте окно Query с запросами к таблицам CUSTOMER.DB и ORDERS. DB.

2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать оба запроса по полю Customer No.
3. Включите в запросе CUSTOMER. DB поля Customer No и Name. Оба этих поля группируют записи абсолютно идентично, поскольку однозначно связаны между собой (просто вы включаете поле Name в таблицу Answer).
4. Включить (для наглядностит) и ввести sum>5000 в поле Balance Due запроса ORDERS.DB .
5. Выполните запрос.

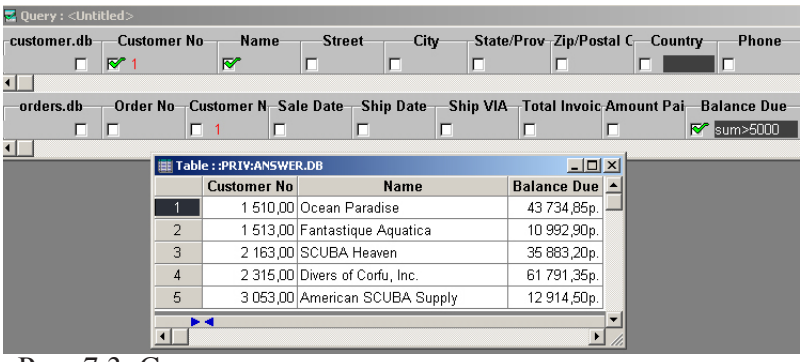


Рис. 7.3 Суммирование в группе

Селекция записей по среднему значению

Следующий пример демонстрирует, использование оператора SUM для усреднения значений внутри каждой группы записей.

Пример. Вычисление среднего значения записей в группе

Предположим, вы хотите узнать в каких штатах находятся ваши клиенты, которые делают заказы на среднюю сумму меньше $50000.

1. Откройте окно Query с запросами к таблицам CUSTOMER.DB и ORDERS.DB.
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать оба запроса по полю Customer No.
3. Включите в запросе CUSTOMER. DB поле State/Prov.
4. Введите average<50000 в поле Total Invoice запроса ORDERS. DB.
5. Выполните запрос.

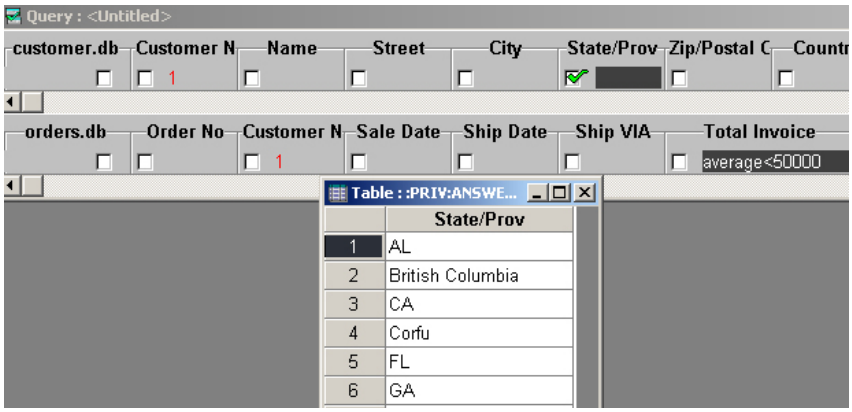


Рис. 7.4 Вычисление среднего значения записей в группе

Селекция записей по минимуму или максимуму

Следующий пример демонстрирует, как использовать оператор MAX для поиска максимального значения в группе записей. (Оператор MIN используется аналогично.)

Пример. Поиск максимального значения в группе

Предположим, вы хотите узнать, в каких странах есть ваши клиенты, максимальная задолженность среди которых составляет не больше $20000.

1. Откройте окно Query с запросами к таблицам CUSTOMER.DB и ORDERS. DB.
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать оба запроса по полю Customer No.
3. Включите в запросе CUSTOMER.DB поле Country.
4. Включите (для наглядности) и введите max< =20000 в поле Total Invoice запроса ORDERS.DB .
5. Выполните запрос.

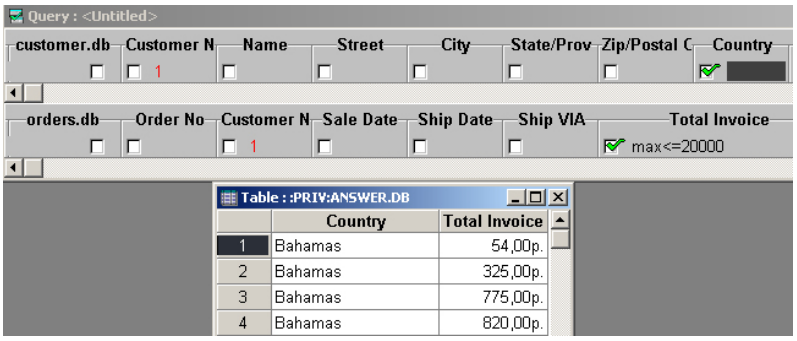


Рис. 7.5 Поиск максимального значения в группе

Вычисления над группами записей

В Главе 6 было описано, как с помощью запроса можно производить вычисления внутри отдельных записей таблицы и помещать полученное значение в новое поле. Помимо этого, вы можете рассчитывать статистические характеристики групп записей. Например, вы можете узнать :

- Сколько заказов поступило на каждый вид вашей продукции?
- Каков объем продаж каждому клиенту?
- Сколько ваших клиентов живет в каждой стране?
- Каковы цены самого дешёвого и самого дорогого из товаров, находящихся на вашем складе?

Чтобы вычислять статистические характеристики полей таблиц, вместе со статистическими операторами используется оператор CALC.

Как и все CALC-запросы, рассматриваемые в данном разделе групповые запросы добавляют в таблицу Answer новое вычисляемое поле. Paradox автоматически присваивает ему имя, исходя из наименования групповой операции, но вы также можете задать его явно при помощи оператора AS.

Пример. Использование CALC SUM в запросе

Предположим, вы хотите узнать количество изделий каждого вида из находящихся в данный момент на вашем складе.

1. Откройте окно Query с запросом к таблице STOCK. DB.
2. Включите в запрос поле Equipment Class.
3. Введите calc sum в поле Qty.
4. Выполните запрос

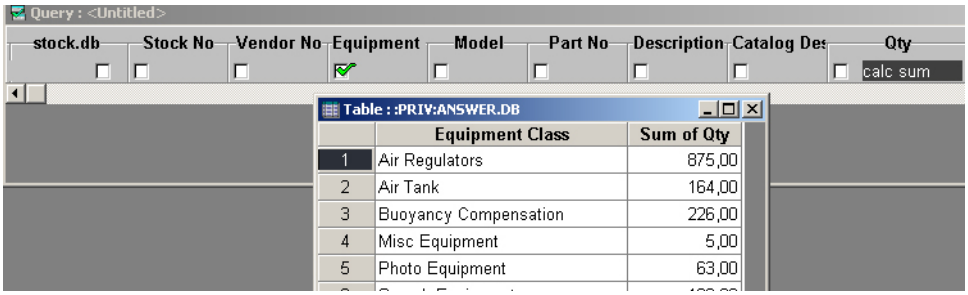


Рис. 7.6 Использование CALC SUM в запросе

Группирование записей по нескольким полям

Чтобы при выполнении запроса записи таблицы группировались по нескольким полям, все эти поля следует включить в запрос. Если некоторые из них однозначно связаны между собой, то соответствующие им группы записей просто совпадают друг с другом.

Пример. Группирование записей по нескольким полям

Предположим, вас интересует связь между способом оплаты товаров вашими клиентами и предпочитаемым ими способом доставки.

1. Откройте окно Query с запросом к таблице ORDERS. Об.
2. Включите в запрос поля Payment Method и Ship VIA.
3. Введите calc sum as Group Total в поле Total Invoice, чтобы включить в таблицу Answer вычисляемое поле Group Total (оно будет содержать выплаченные по счетам суммы для каждой возможной комбинации способа оплаты и способа доставки).
4. Выполните запрос.

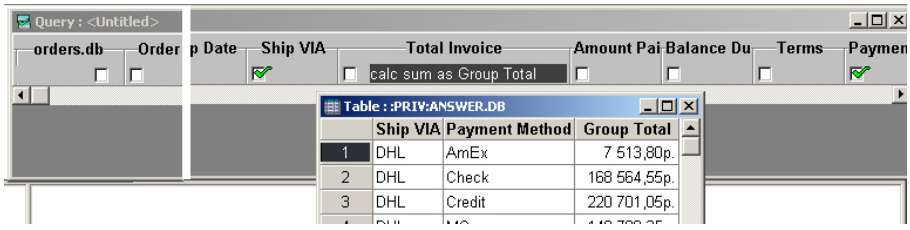


Рис. 7.7 Группирование записей по нескольким полям

Групповые вычисления по всей таблице

Выше было показано, каким образом включённые в запрос поля группируют записи таблицы. Если же вы не включите в запрос ни одного поля, статистические операторы будут работать с целой таблицей, как с одной группой.

Пример. Групповой запрос к целой таблице

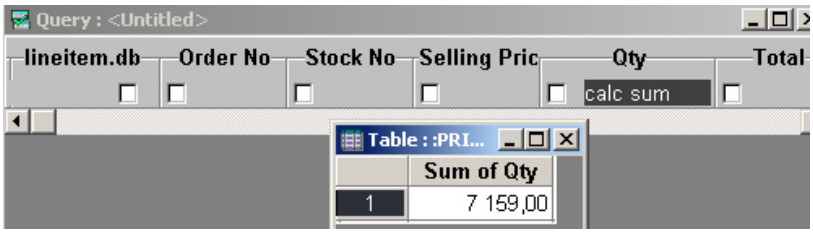


рис. 7.8 Групповой запрос к целой таблице

Предположим, вы хотите узнать полное количество заказанных единиц товара, независимо от их вида, цены и заказчика.

1. Откройте окно Query с запросом к таблице LINEITEM. DB.
2. Введите calc sum в поле Qty.
3. Выполните запрос.

В результате вы получите сумму значений, находящихся в поле Qty, вычисленную по всем записям

Статистические вычисления над не группирующими полями

Чтобы в результате запроса получить статистически обработанные значения и при этом по данному полю не производить группирования записей, в этом поле кроме статистического оператора необходимо дополнительно использовать оператор CALC. Оператор CALC заставит Paradox включить в таблицу Answer новое вычисляемое поле, которое будет содержать значения, отвечающие условию, задаваемому статистическим оператором.

Пример. Получение статистических характеристик

Предположим, вы хотите узнать, какие виды товаров впервые были проданы после 1 июня 1990г., а также даты соответствующих заказов.

1. Откройте окно Query с запросами к таблицам ORDERS.DB и LINEITEM.DB .
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать оба запроса по полю ORDER No.
3. Включите в запросе LINEITEM.DB поле Stock No, чтобы включить его в таблицу Answer и сгруппировать по нему записи.
4. Введите min>01.01.90, calc min в поле Sale Date запроса ORDERS. DB . Если данное поле включить в запрос, по нему будет производиться группирование записей так же, как и по полю Stock No таблицы LINETEM.DB . Используемый вместо этого оператор calc min заставит Paradox создать в таблице Answer дополнительное поле Min of Sale Date, ко-торое будет содержать даты продаж, удовлетворяющие условию min>01.01.90, не нарушая при этом необходимого порядка группирования записей при выполнении запросе.
5. Выполните запрос.

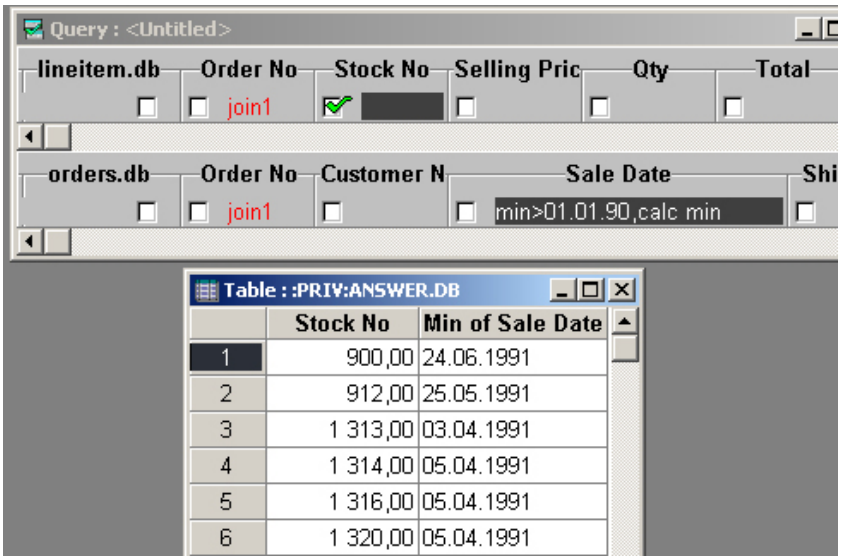


Рис. 7.9 Получение статистических характеристик

Подсчет неповторяющихся значений

По умолчанию оператор CALC COUNT подсчитывает только неповторяющиеся значения. Однако, для BLOB-полей Paradox- и dBASE-таблиц Paradox не может установить уникальность их значений. Поэтому в полях данных типов оператор CALC COUNT подсчитает количество всех значений, даже если вы используете модификатор UNIQUE.

Пример. Использование оператора CALC COUNT

Предположим, вы хотите узнать полное количество клиентов, разместивших заказы в вашей фирме.

1. Откройте окно Query с запросом к таблице ORDERS. DB.
2. Введите calc count в поле Customer No.
3. Выполните запрос.

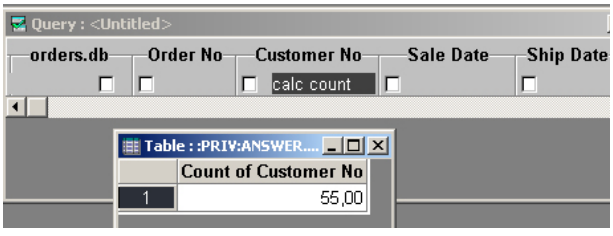


Рис. 7.10 Использование оператора CALC COUNT

В результате вы получите количество различных значений, находящихся в поле Customer No таблицы ORDERS.DB.

Подсчет всех значений

Чтобы оператор COUNT обрабатывал повторяющиеся значения, вы можете просто добавить слово all после calc count.

Пример. Использование оператора CALC COUNT ALL

Предположим, вы хотите узнать полное количество заказов, размещенных в вашей фирме.

1. Откройте окно Query с запросом к таблице ORDERS.DB.
2. Введите calc count all в поле Customer No.
3. Выполните запрос.

В результате вы фактически получите количество записей в таблице ORDERS.DB (конечно, при условии, что для всех зака-зов указан кодовый номер клиента-заказчика).

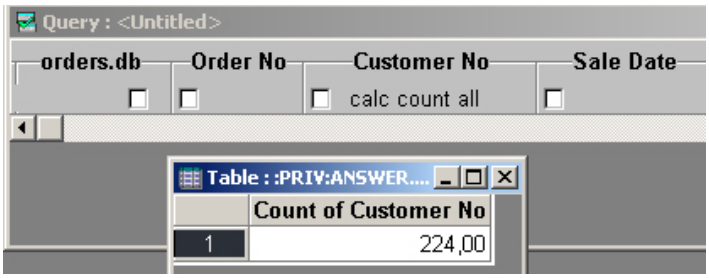


Рис. 7.11 Использование оператора CALC COUNT ALL

Оператор ONLY

Оператор ONLY не относится к статистическим, поскольку не производит никаких вычислений, однако, в остальном действует аналогично: он объединяет в группы записи, содержащие одно определенное значение.

Пример. Использование оператора ONLY

Предположим, вы хотите узнать, кто из ваших клиентов заказал товар класса «Small Instruments» (мелкий инвентарь).

1. Откройте окно Query с запросами к таблицам ORDERS.DB, LINEITEM.DB и STOCK. DB .
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать запросы к ORDERS.DB и LINEITEM. DB по полю Order No.
3. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать запросы к LINEITEM.DB и STOCK. DB по полю Stock No.
4. Включите в запросе ORDERS.DB поле Customer No, чтобы включить его в таблицу Answer и сгруппировать по нему записи.
5. Введите only Small Instruments в поле Equipment Class запроса STOCK. DB .
6. Выполните запрос .

В таблице Answer вы получите номера клиентов, заказавших товары «Small Instruments» и больше ничего

Таблиц» 7.5 Типы полей Paradox-таблиц, допускающих использование операто-ра ONLY

Оператор	A	N	$	D	S
ONLY	+	+	+	+	+

Таблица 7.6 Типы полей dBASE-таблиц, допускающих использование оператора ONLY

Оператор	C	F	N	O	L
ONLY	+	+	+	+	+

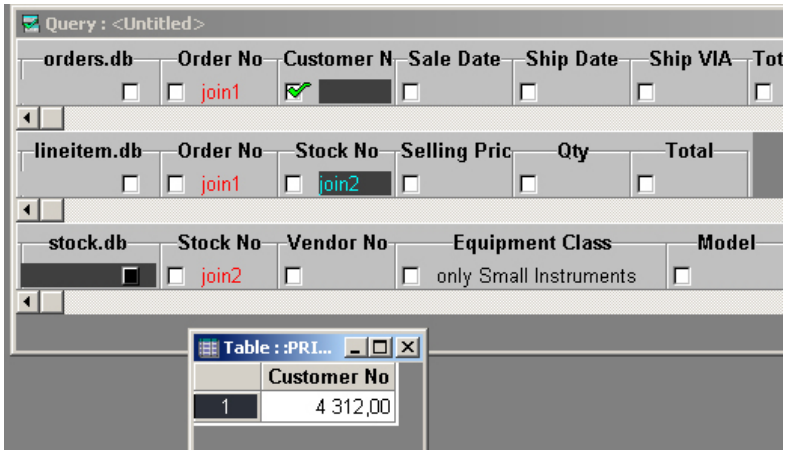


Рис. 7.12 Использование оператора ONLY

Использование наборов записей

Вообще говоря, набором является любая совокупность объектов. В Paradox термин набор (set) обозначает определённую группу записей, к которой вы хотите задать «дополнительные вопросы».

Например, таблица Orders представляет собой полный список заказов, сделанных вашими кли-

ентами. Из него вы можете составить наборы по различным классам заказанных товаров (инструменты, средства передвижения и т.п.) Так, с помощью оператора SET вы можете определить наборы заказов, которые:

- Касаются только товаров класса «Small Instruments» (мелкий инвентарь)
- Не содержат ни одного пункта, относящегося к классу «Small Instruments»
- Содержат все товары класса «Small Instruments»
- Содержат все товары класса «Small Instruments» и никаких больше

Следующие рисунки иллюстрируют перечисленные наборы. Пунктирной линией выделены товары одного класса Small Instruments таблицы stock

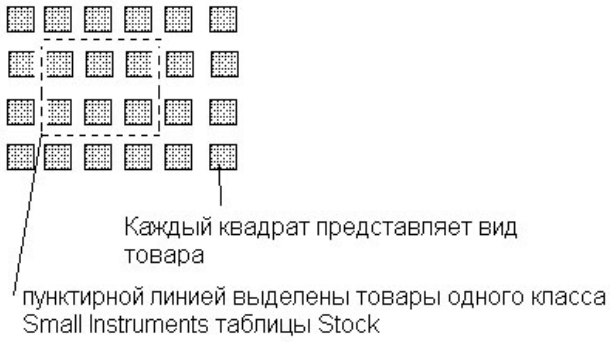


Рис.7.13а перечень наборов

Рис.7.13б перечень наборов в этот заказ не входят товары Small Instruments

Рис.7.13в перечень наборов

Если в запросе определен набор, вы можете сравнивать его с другими записями, а также с группами записей. Для этого используются операторы срав-нения ONLY, NO, EVERY и EXACTLY.

Оператор определения набора SET служит, во-первых, для того, чтобы одним запросом выполнять такие операции, которые, обычно, реализуются двумя и более запросами, и вовторых, для вычисления статистических характеристик групп записей.

Разработка SET-запроса

Каждый SET-запрос состоит из следующих компонентов:

- Одной или нескольких строк, определяющих набор
- Одной или нескольких строк, которые задают, каким образом другие записи таблицы сравниваются с набором
- Одной или нескольких строк, которые отображают необходимую информацию (необязательный компонент)

Определение набора

Определение набора записей представляет собой выбор записей, которые следует использовать для построения таблицы Answer. Фактически, вы должны создать запрос внутри запроса.

Ниже приведены основные этапы определения набора:

1. Введите в запрос (запросы) условия отбора записей и элемент-примеры, если необходимо связать между собой записи из разных таблиц.
2. Используя меню операций запроса, установите в первых полях строк, определяющих набор, оператор SET.
3. В поля, которые в обычном случае вы бы включили в запрос при помощи значков F6 поместите элемент-примеры (строки, определяющие набор, не должны содержать значков F6 и статистических операторов).

Определение групп записей для сравнения с набором

Если вы хотите получить номера заказов на класс товаров, определяемый заданным набором, добавьте в окно Query запрос к таблице Lineitem (пункты заказов) и установите значок V в поле Order No (номер заказа). Затем в поле Stock No (складской код товара) таблицы Lineitem введите элемент-пример. Связка по Stok No равносильна в данном случае оператору ONLY. Данный запрос означает следующее:

- В таблице Stock выделен класс товара «Small Instruments»
- Записи в таблице Lineitem сгруппированы по значениям поля Order No
- В таблицу Answer включено поле Order No таблицы Lineitem

В результате выполнения данного запроса таблица Answer содержит номе-ра заказов, в которых перечислены продукты, относящиеся исключительно к классу «Small Instruments

Операторы NO, EVERY и EXACTLY используются в запросах аналогично оператору ONLY.

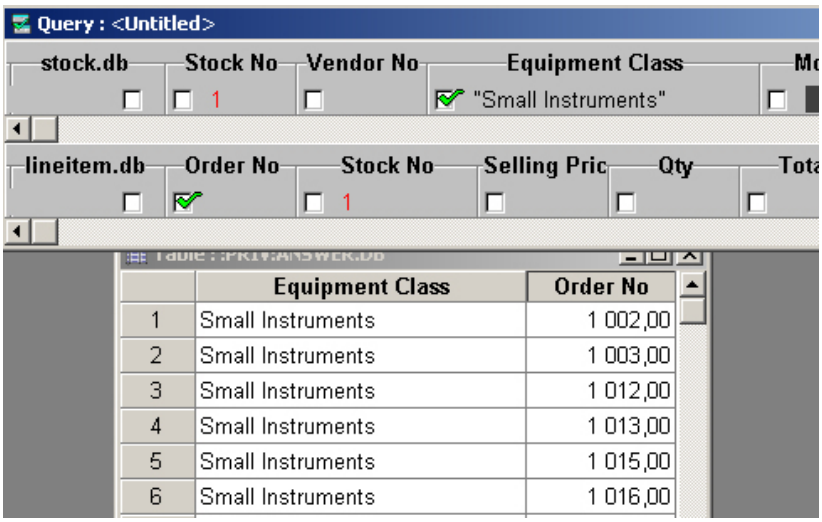


Рис. 7.14 Определение групп записей для сравнения с набором

Использование значка VG для группирования записей по значению

Иногда бывает необходимо сгруппировать записи по значению какого-либо из ее полей, но само его в таблицу Answer не включать. Для этого данное поле следует отметить значком VG (GroupBy check). Этот значок используется только в SET-запросах. Типы полей, которые можно отмечать значком VG, приведены в таблицах 6.3 и 6.4 Главы 6.

Оператор ONLY указывает Paradox, что отбирать следует только те значения, которые присутствуют в определённом вами наборе. Следующий пример демонстрирует запрос, аналогичный предыдущему, но содержащий дополнительно таблицу Orders. В результате выполнения обоих запросов получается одна и та же таблица Answer, но в них различными способами задается группирование записей по номерам заказов.

Таблица Orders является «родительской» по отношению к таблице Lineitem, они связаны по полю Order No, поэтому таблица Lineitem не может содержать заказов, которые не присутствуют в таблице Orders. Но если бы такие записи в таблице существовали, то запрос из предыдущего примера обнаружил бы их и поместил бы в таблицу Answer, а запрос из следующего примера - нет.

Пример. Использование оператора ONLY

Предположим, вы хотите узнать номера заказов, в которых все наименования товаров относятся к классу «Small Instruments».

1. Откройте окно Query с запросами к таблицам LINEITEM.DB, ORDERS.DB и STOCK.DB .
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать запросы к ORDERS.DB и LINEITEM.DB по полю Order No.
3. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать запросы к LINEITEM.DB и STOCK.DB по полю Stock No.
4. В левом столбце запроса STOCK.DB введите оператор SET, нажав клавишу S или вызвав щелчком мыши меню операций запроса и сделав в нем соответствующий выбор.
5. Введите Small Instruments в поле Equipment Class запроса STOCK.DB.
6. Отметьте значком F6 поле Order No в запросе ORDERS.DB: вы группируете записи по значениям в этом поле и включаете его в таблицу Answer

7. Отметьте значком поле Order No в запросе LINEITEN.DB: вы группируете записи по значениям в этом поле, но не включаете его в таблицу Answer.
8. Введите only перед элемент-примером в поле Stock No запроса LINEITEM.DB, чтобы Paradox отбирал только те заказы, в которых перечислены складские коды товаров, относящихся исключительно к классу «Small Instruments». (Если из этого запроса удалить операторы SET и ONLY, вы получите номера заказов, в которых присутствуют товары класса «Small Instruments» в комбинации с другими классами.)
9. Выполните запрос

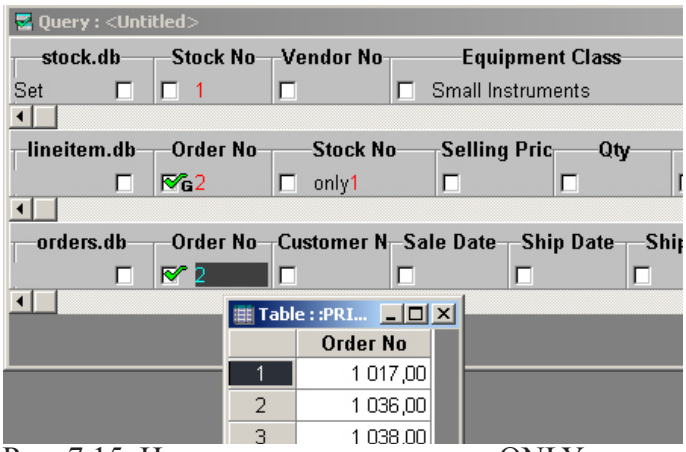


Рис. 7.15 Использование оператора ONLY

Оператор NO

Оператор NO указывает Paradox, что отбирать следует только те группы, в которых ни одна запись не содержит ни одного значения, присутствующего в определённом вами наборе.

Пример. Использование оператора NO

Предположим, вы хотите узнать номера заказов, в которых все товары дороже $50

1. Составьте запрос, рассмотренный в предыдущем примере.
2. Удалите условие отбора Small Instruments из поля Equipment Class запроса STOCK.DB.
3. Введите >50 в поле List Price запроса STOCK.DB, чтобы определить набор складских кодов товаров, цена которых превышает $50.
4. Замените оператор ONLY перед элемент-примером в поле Stock No запроса LINEITEM.DB на N0, чтобы Paradox отбирал только те заказы, в которых отсутствуют пункты ценой меньше или равной $50.
5. Выполните запрос.

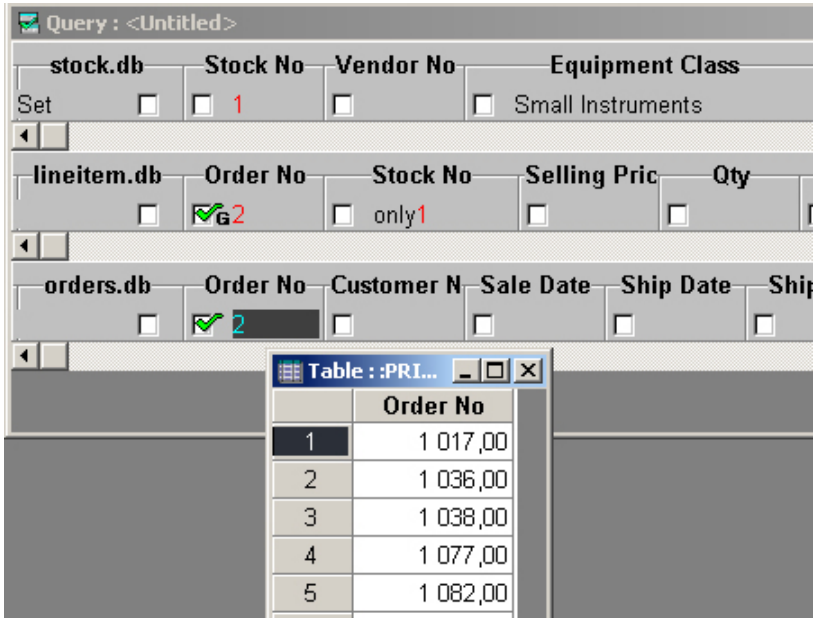


Рис. 7.16 Использование оператора NO

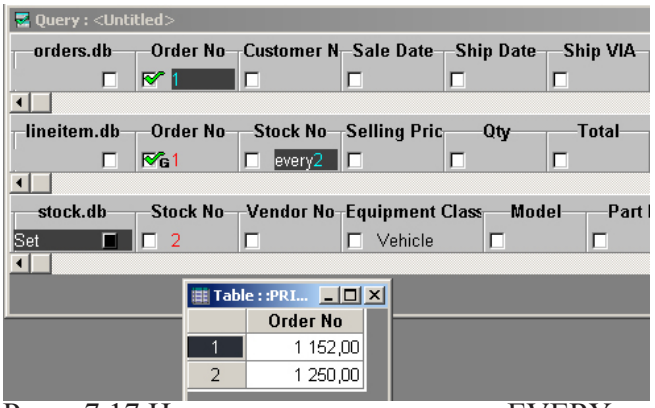
Оператор EVERY

Оператор EVERY отбирает только те группы, в которых каждая запись соответствует всему определенному вами набору.

Пример. Использование оператора EVERY

Предположим, вы хотите узнать номера заказов, в которых все товары относятся к классу «Vehicle».

1. Составьте запрос, рассмотренный в предыдущем примере.
2. Удалите >50 из поля List Price запроса STOCK.DB.
3. Введите vehicle в поле Equipment Class запроса STOCK.DB, чтобы определить набор складских кодов из товаров, относящихся к классу «Vehicle».



Ррис. 7.17 Использование оператора EVERY

4. Замените оператор NO перед элемент-примером в поле Stock No запроса LINEITEM.DB на EVERY, чтобы Paradox отбирал только те заказы, все пункты которых касаются товаров класса «Vehicle».
5. Выполните запрос

Оператор EXACTLY

Оператор EXACTLY отбирает только те группы, которые совпадают с определённым вами набором.

Пример. Использование оператора EXACTLY

Предположим, вы хотите узнать, заказал ли кто-либо у вас сразу всю номенклатуру товаров класса «Vehicle», при этом больше ничем не интересуясь. В исходной таблице Orders отсутствуют такие заказы, но представьте себе, что по просьбе одного из клиентов вам пришлось несколько изменить сделанный им заказ: например, отредактируйте следующим образом одну из записей таблицы Lineitem, относящуюся к заказу номер 1363:

Поле	Старое значение	Новое значение
Stock No	1390	912
Selling Price	170.00	1680.00
Qty	8	1
Total	1360.00	1680.00

Затем составьте запрос:

1. Возьмите за основу запрос, рассмотренный в предыдущем примере.
2. Оставьте условие отбора vehicle в поле Equipment Class запроса STOCK. DB.
3. Замените оператор every перед элемент-примером в поле Stock No запроса LINEITEM.DB на exactly, чтобы Paradox отбирал только те заказы, в которых содержится полный перечень товаров класса «Vehicle» и более ничего.
4. Выполнитезапрос.

Как видно из результата – данных, отвечающих условию нет, что можно проверить просмотрев таблицы STOCK.DB, LINEITEM.DB

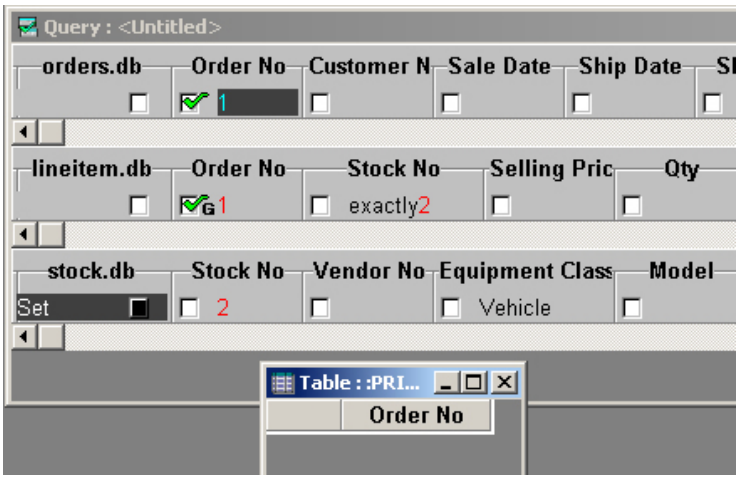


Рис. 7.18 Использование оператора EXACTLY

SET-запросы с несколькими наборами

Следующий пример демонстрирует поиск записей по запросу, основанный на сравнении с двумя наборами.

Пример. SET-запрос с двумя наборами

Предположим, вам необходимо связать имена клиентов со сделанными ими заказами (из рассмотренных выше примеров), а именно, вас интересуют клиенты, которые заказывают только товары класса «Small Instruments».

1. Откройте окно Query с запросами к таблицам CUSTOMER.DB, ORDERS.DB, LINEITEM.DB и STOCK.DB.
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать соответствующими элемент-примерами запросы CUSTOMER.DB
3. и ORDERS.DB по полю Customer No, запросы ORDERS.DB и LINEITEM.DB по полю Order No, а запросы LINEITEM.DB и STOCK. DB - по полю Stock No.
4. Введите Small Instruments в поле Equipment Class запроса STOCK.DB, а в крайнем левом поле установите оператор SET (вы выделили из товаров, находящихся на складе, изделия класса «Small Instruments»).
5. Введите only перед элемент-примером в поле Stock No и отметьте значком VG поле Order No запроса LINEITEM.DB (вы определили, как записи таблицы Lineitem должны сравниваться с набором,

выделенным из таблицы Stock).

6. Установите оператор SET в крайнем левом поле запроса LINEITEM.DB (теперь все отобранные записи объявляются, в свою очередь, набором).
7. Введите only перед элемент-примером в поле Order No и отметьте значком Vg поле Customer No запроса ORDERS.DB (вы определили, как записи таблицы Orders должны сравниваться с набором, выделенным из таблицы Lineitem).
8. Отметьте значком V поле Name запроса CASTOMER.DB, чтобы указать, что из таблицы Customer следует извлечь только тех клиентов, которые сделали заказы исключительно на товары из определённого вами набора.
9. Выполните запрос

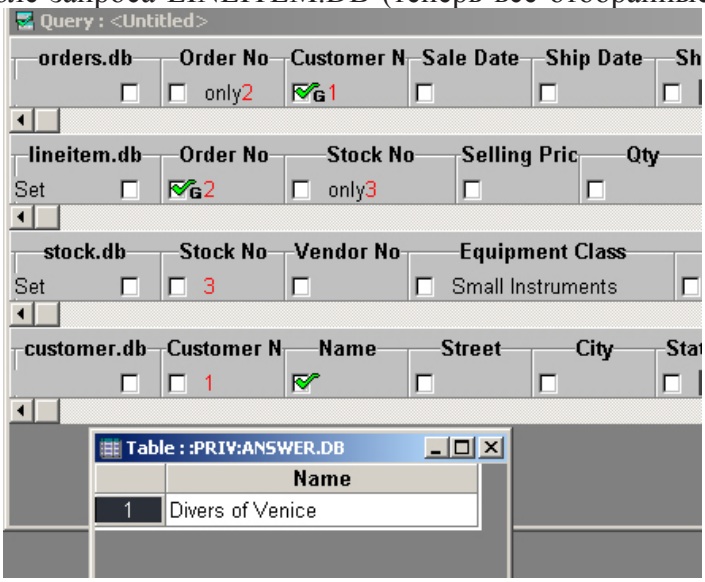


Рис. 7.19 SET-запросы с несколькими наборами

Включающие связи

В рассмотренных выше разделах данной главы для связи таблиц между собой в запросах использовались элемент-примеры. При этом из одной таблицы извлекались только те записи, которым находились соответствующие в другой таблице. Данный принцип работы запроса называется исключающей связью.

Если же вы хотите, чтобы в таблицу Answer включались еще и записи, которым не соответствует ни одна запись из связанной таблицы, в запросе необходимо использовать оператор включения (!). Такие запросы работают по принципу включающей связи. Оператор **!** ставится после элемент-примера и указывает на то, что из данной таблицы следует извлекать все записи независимо от того, соответствует ли им какая-либо запись из другой таблицы. Кроме того, можно добавить условия отбора, чтобы определить, какие именно записи из данной таблицы должны попасть в таблицу Answer. В данном разделе рассказывается о том, как:

- Использовать операторы ! для извлечения всех записей сразу из нескольких таблиц
- Комбинировать оператор ! с арифметическими выражениями
- Применять в одном запросе как включающую, так и исключающую связи

Связывание всех записей таблицы

Следующий пример демонстрирует запрос, работающий по принципу включающей связи.

Пример. Использование оператора **!**

Предположим, вы хотите знать, есть ли в таблице Customer клиенты, не сделавшие ни одного заказа. Если просто связать таблицы Customer и Orders элемент-примером по полю Customer No, вы получите клиентов, которым соответствует одна и более записей таблицы Orders. Если же после элемент-примера поставить оператор !, вы получите имена всех клиентов, втом числе и не сделавших ни одного заказа.

новая запись к п.2

Поле	Значение
Customer No	9999
Name	The Human Gill Dive Shop
Street	1 223 E. River St.
City	Savannah
State/Prov	GA
Zip/Postal	30541
Country	U.S.A.
Phone	404-555-1451
First Contact	10/29/93

1. Откройте окно Query с запросами к таблицам CUSTOMER.DB и ORDERS.DB.
2. Откройте таблицу Customer (например, командой File / Open / Table и добавьте в нее одну запись (переместитесь в конец таблицы, нажмите F9 и клавишу ins или кн. с иконкой «редактирование»). Заполните ее информацией о новом клиенте:

3. После ввода в таблицу нового клиента снова нажмите F9, чтобы выйти из режима редактирования. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать элемент-примерами оба запроса по полю Customer No.
4. Введите оператор ! после элемент-примера в поле Customer No запроса CUSTOMER.DB, чтобы включить в таблицу Answer всех клиентов, в том числе и не сделавших ни одного заказа.
5. Отметьте значком F6 поля Customer No и Name в запросе CUSTOMER.DB.
6. Отметьте значком F6 поле Order No в запросе ORDER.DB.
7. Выполните запрос и обратите внимание на последнюю запись таблицы Answer

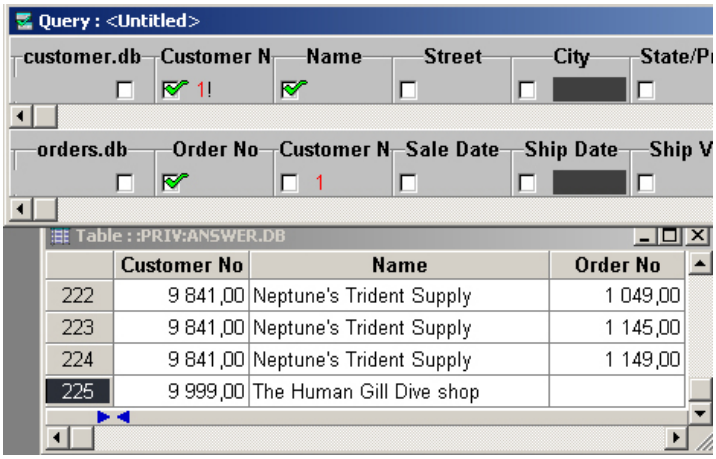


Рис. 7.20 Использование оператора !

Использование оператора ! в запросах, производящих вычисления

Следующий пример демонстрирует использование оператора ! в запросах, предназначенных для выполнения определённых вычислений.

Пример. Использование оператора ! в вычислительном запросе

Предположим, вас интересуют заказы, которые вы не можете выполнить за счет продукции, находящейся в данный момент на вашем складе. Точнее, вы хотите получить список заказов, объем которых превышает четвертую часть количества находящейся на складе продукции данного вида.

1. Откройте окно Query с запросами ORDERS.DB, LINEITEM.DB и STOCK.DB.
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать соответст-вующими элемент-примерами запросы ORDERS.DB и LINEITEM.DB по полю Order No, а запросы LINEITEM.DB и STOCK. DB - по полю Stock No.
3. Введите оператор ! после элемент-примера в поле Order No запроса OR-DERS.DB, чтобы в процессе выполнения запроса обрабатывались все заказы.
4. Отметьте значком F6 поля Stock No и Qty в запросе LINEITEM.DB.
5. В поле Qty запроса LINEITEM.DB введите элемент-пример qty.
6. В том же поле Qty запроса LINEITEM.DB после элемент-примера введите as Order Qty.
7. Отметьте значком F6 поле Qty в запросе ORDERS.DB.
8. В этом же поле введите <(, элемент-пример qty и окончание выражения: *4), as Stock Qty. Теперь объем каждого пункта всех заказов умножается на 4 и сравнивается со значением поля Qty соответствующей записи таб-лицы Stock.
9. Выполните запрос.

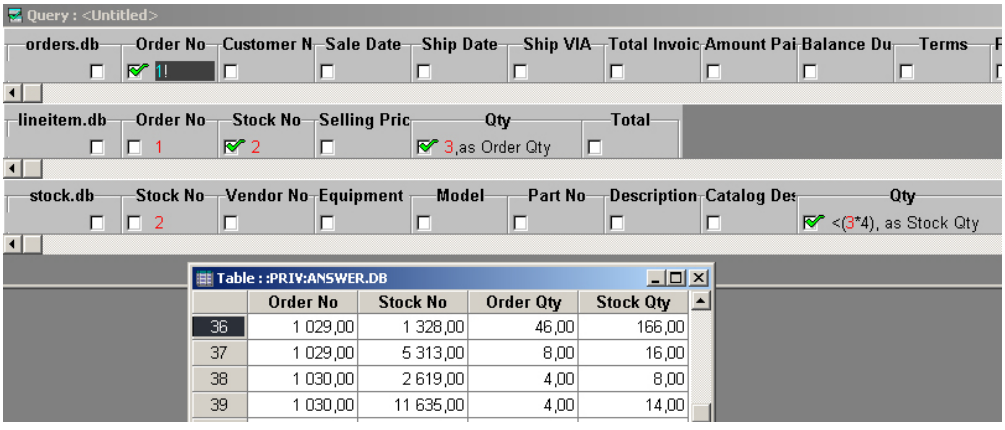


Рис. 7.21 Использование оператора ! в вычислительном запросе

Извлечение из таблицы записей со значениями, которых нет в другой таблице

В следующем примере демонстрируется использование включающей связи в сочетании с оператором COUNT и значками V+ для поиска в таблице записей с теми значениями, которые отсутствуют в другой таблице.

Пример. Извлечение из таблицы записей со значениями, которых нет в другой таблице

Предположим, вы хотите узнать, у каких поставщиков из занесённых вами в таблицу Vendors вы до сих пор ничего не купили. Иными словами, какие поставщики есть в таблице Vendors, но отсут-

ствуют в таблице Stock.

1. Откройте окно Query с запросами STOCK.DB и VENDORS.DB.
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать элемент -примером запрос по полю Vendor No.
3. Введите оператор ! после элемент - примера в поле Vendor No запроса VENDORS.DB.
4. Отметьте значком поля Vendor No и Vendor Name в запросе VENDORS. DB.
5. В поле Vendor No запроса STOCK. Об после элемент - примера поставьте запятую, пробел и введите выражение count=0.
6. Выполните запрос.

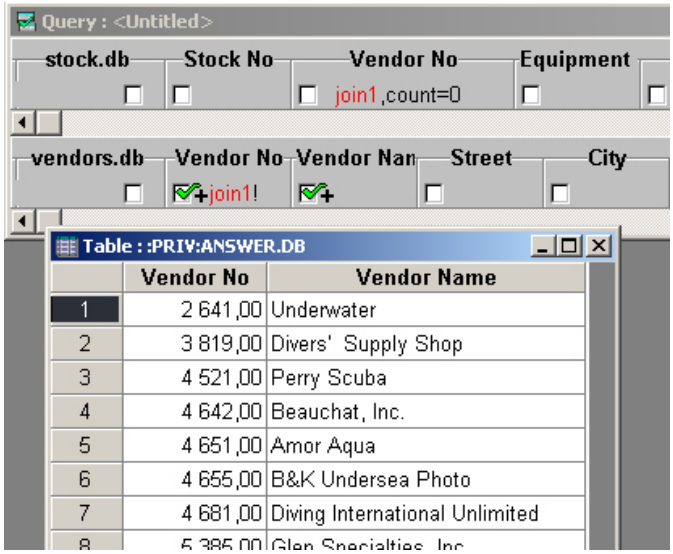


Рис. 7.22 Извлечение из таблицы записей со значениями, которых нет в другой таблице

Включающая и исключающая связи в одном запросе

Пример «Использование оператора ! в вычислительном запросе» демонстрирует, как в одном запросе работают между таблицами связи обоих типов - включающая и исключающая. Ниже приводится более сложный пример применения комбинации типов связи.

Пример. Использование включающей и исключающей связи в одном запросе

Предположим, вы недавно договорились с поставщиками, что не будете продавать вашим клиентам товары, если их поставщик находится в том же штате, что и покупатель. Теперь вы хотите узнать, как это соглашение повлияет на полный объем заказов, которые в данный момент размещены в вашей фирме.

1. Откройте окно Query с запросами VENDORS.DB, STOCK.DB, LINEITEM.DB, ORDERS.DB и CUSTOMER.OS.
2. Воспользуйтесь кнопкой Join Tables на SpeedBar, чтобы связать соответствующими элемент-примерами запросы VENDORS.DB и STOCK.DB по полю Vendor No, запросы STOCK.DB и LINEITEM.DB по полю Stock No, запросы LINEITEM.DB и ORDERS.DB по полю Order No, а запросы ORDERS.DB и CUSTOMER DB - по полю Customer No.
3. Введите оператор ! после элемент - примера в поле Vendor No запроса VENDOR.DB и отметьте это поле значком при нажатии F6, чтобы включить в таблицу Answer всех поставщиков, в том числе и тех, у которых вы ничего не купили.
4. Отметьте значком при нажатии F6 поле State/Prov запроса VENDORS. DB и введите в него элемент-пример state.
5. В это же поле после элемент - примера введите оператор !, чтобы запросом обрабатывались все штаты, и as Vendor State, чтобы переименовать это поле в таблице Answer на Vendor State.
6. Отметьте значками F6 поля Stock No и Description запроса STOCK.DB, чтобы включить их в таблицу Answer.
7. Введите calc sum as Dollars of Stake в поле Total запроса LINEITEM.DB, чтобы включить в таблицу Answer дополнительное вычисляемое поле. Оно

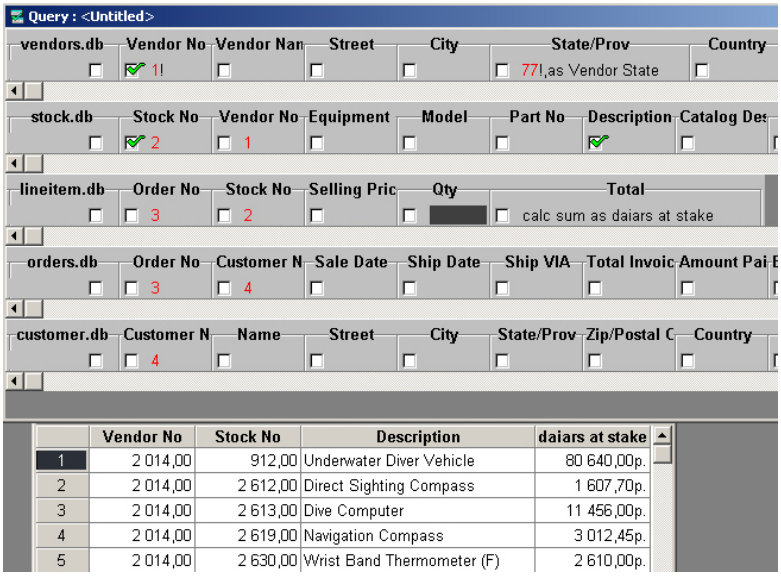


Рис. 7.23 Использование включающей и исключающей связи в одном запросе

будет содержать суммы, на которые сделаны заказы по типам изделий, отвечающим условию, что их поставщик и заказчик находятся в одном штате.

8. Введите элемент-пример state в поле State/Prov таблицы CUSTOMER.DB.
9. Выполните запрос.

В полях таблицы Answer содержится следующая информация:

- Поставщик, независимо от того, покупали ли вы у него товары.
- Штат, в котором находится поставщик и, соответственно, ваши клиенты (если таковые окажутся), заказавшие товар именно этого поставщика.
- Складской код товара данного поставщика, если на него имеется заказ от клиента, находящегося в одном штате с поставщиком, в противном случае поле остается незаполненным.
- Сумма, которую вы потеряете, поскольку по новому соглашению с поставщиками данный вид товара данному клиенту вы не имеете права продавать

Правила связывания таблиц

Две любые строки запроса вы можете связать только одним типом связи - включающей либо исключающей. Это правило объясняется просто: включающая запись выбирает из таблицы все записи, в то время как исключающая - только те, которым соответствуют записи из связанной таблицы.

Если вы используете оба типа связи, Paradox не будет знать, по какой из них начать обработку табли-цы (в зависимости от последовательности их использования будут получаться различные таблицы Answer).

Чтобы избежать этого конфликта, запомните, что оператор ! с одним элемент - примером можно использовать лишь однажды в строке и дважды в запросе. Иными словами, между двумя строками запроса может существовать только один тип связи - либо включающая либо исключающая.

В одном запросе можно использовать оба типа связывания таблиц, при этом первыми всегда обрабатываются наиболее исключающие:

- Сначала обрабатываются исключающие связи, которые не выбирают записи, которым нет соответствующих записей в связанной таблице.
- Далее обрабатываются асимметричные включающие связи, которые выбирают все записи из главной таблицы и только соответствующие им записи из связанной таблицы
- Последними обрабатываются симметричные включающие связи (только с главными таблицами), которые выбирают все записи из обеих таблиц.

Такой порядок обработки разных типов связи обеспечивает однозначность полученных результатов. Если вы хотите, чтобы связи обрабатывались в другом порядке, вам придётся разбить ваш запрос на несколько отдельных запросов.

Таблица 7.7 Типы полей Paradox-таблиц, допускающие использование операторов сравнения с набором и оператора !

Оператор	A	N	$	D	S
ONLY	+	+	+	+	+
NO	+	+	+	+	+
EVERY	+	+	+	+	+
EXACTLY	+	+	+	+	+
!	+	+	+	+	+

Таблица 7.8 Типы полей dBASE-таблиц, допускающие использование операторов сравнения с набором и оператора !

Оператор	C	F	N	D	L
ONLY	+	+	+	+	+
N0	+	+	+	+	+
EVERY	+	+	+	+	+
EXACTLY	+	+	+	+	+
!	+	+	+	+	+

Глава 8. Утилиты работы с объектами

Для работы с файлами объектов Paradox предоставляет набор операций общего на-значения (рис. 8.1). Утилиты работы с объектами можно вызвать, используя пункт меню Tools / Utilities

В некоторых версиях Paradox часть утилит может находиться в другом месте, но работают он одинаково во всех версиях. Они позволяют копировать, переименовывать или уничтожать любой объект Paradox. С их помощью можно также производить некоторые специальные действия с таблицами:

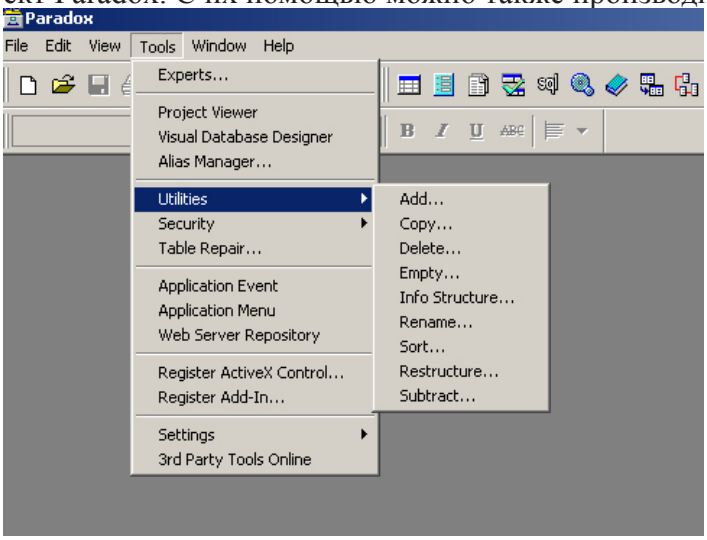


Рис. 8.1 Утилиты

- Добавлять записи из одной таблицы в другую
- Удалять записи, существующие в одной из таблиц, из другой таблицы
- Делать очистку таблицы (удалять все записи)
- Получать информацию о структуре таблицы
- Переименовывать
- Сортировать
- Назначать или отменять пароли
- Изменять структуру таблицы (см. Главу 3)

Добавление записей в таблицу

Утилита Add служит для добавления записей из одной таблицы в другую. Для ее вызова дайте команду File (Utilities) Add (или проинспектируйте иконку таблицы-источника, из которой выбираются добавляемые записи, и выберите пункт Add). Вам будет предоставлено диалоговое окно Add (рис. 8.2).

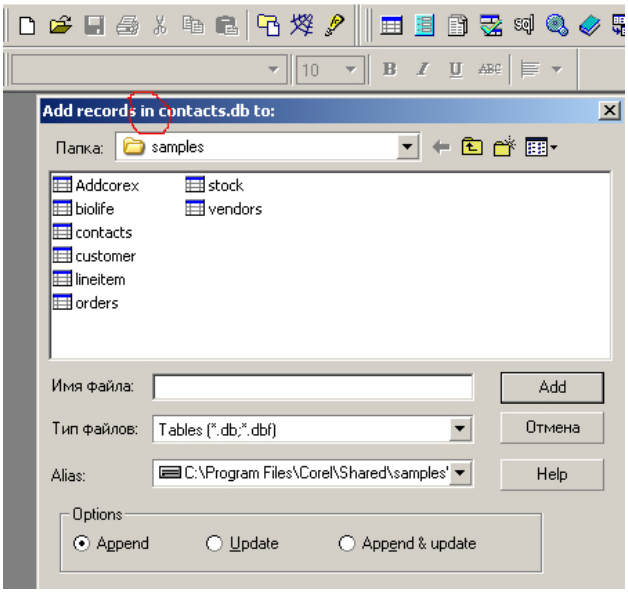


Рис. 8.2 Утилита ADD

После выбора режима сначала открывается окно (рис 8.2) где надо выбрать таблицу, куда надо вставлять (состояние выделено красным), а потом надо выбрать таблицу, которую надо добавить

1. Установите текстовый курсор в окошко Add Records From и выберите таблицу-источник. (Если вы открыли диалоговое окно Add, проинспектировав иконку, Paradox сам поместит имя таблицы-источника в это окошко).
2. Установите текстовый курсор в окошко To и выберите таблицу- приемник.
3. Включите опцию Append (добавление) или Update (обновление) (эти опции описываются ниже в этом разделе).
4. Нажмите OK.

Таблица-источник и таблица-приемник должны иметь поля совместимых (но не обя-зательно идентичных) типов. Совместимыми считаются типы, которые преобразуются один в другой с помощью операции реструктурирования таблицы. Например, числовой тип и тип денежного поля совместимы, а числовой и графический - нет. Полную информацию о совместимости типов полей в Paradox можно найти в таблице 3.6 Главы 3, о совместимости типов в dBASE - в таблице 3.7 той же главы.

При использовании операции Add необходимо помнить следующее:

- Добавлять запись из одной таблицы в другую можно только в том случае, если их структура совместима, т.е. порядок следования полей совместимых типов одинаков в обеих таблицах.
- Таблица-приемник может иметь больше полей, чем таблица-источник, если первые поля таблицы-

приемника совместимы со всеми полями таблицы-источника. В «лишние» поля Paradox помещает нулевые значения.

- Таблица-источник может иметь больше полей, чем таблица-приемник, если первые поля таблицы-источника совместимы с полями таблицы-приемника. Лишние поля таблицы-источника игнорируются.

Добавление записей в таблицу другого типа

Если вы добавляете записи из таблицы одного типа в таблицу другого типа, то необходимо предварительно убедиться в совместимости их полей. Таблица 8.1 показывает совместимость типов полей при добавлении из Paradox-таблицы в dBASE-таблицу, таблица 8.2 - при добавлении из dBASE-таблицы в Paradox.

Таблица 8.1 Совместимость полей при добавлении записей из Paradox-таблицы в dBASE-таблицу

	dBASE C	dBASE F	dBASE N	dBASE D	dBASE L	dBASE M
Paradox A	+	P	P	P	P	+
Paradox N	+	+	+		P	
Paradox $	+	+	+			
Paradox D	+			+		
Paradox S	+	+	+		P	
Paradox M						+
Paradox F						+
Paradox B						+
Paradox G						+
Paradox O						+

+ - совместимость

P - частичная совместимость: в результате преобразования может быть образована таблица Problems

Преобразование Paradox-полей в dBASE BLOB-поля

При добавлении данных из форматированного memo поля Paradox-таблицы в memo поле dBASE-таблицы Paradox преобразует форматированный текст в неформатированный.

Если данные поступают из поля графического типа, двоичного или OLE-поля, dBASE воспринимает их, но отобразить не может.

Таблица 8.2 Совместимость полей при добавлении записей из dBASE-таблицы в Paradox-таблицу

	Paradox									
	A	N	$	D	S	M	F	B	G	O
dBASE C	+	P	P	P	P					
dBASE F	+	P	P		P					
dBASE N	+	P	P		P					
dBASE D	+			+						
dBASE L	+									
dBASE M						+	+	+	+	+

+ - совместимость

P - частичная совместимость: в результате преобразования может быть образована таблица Problems

Преобразование dBASE-полей в BLOB-пола Paradox-таблиц

Вы можете добавлять данные из memo поля dBASE-таблицы в форматированное memo, графическое, OLE или бинарное поле Paradox-таблицы. Это допустимо, поскольку dBASE memo поле может хранить данные любого типа. При этом необходимо убедиться, что данные соответствуют типу поля Paradox - таблицы. Например, если в dBASE memo поле содержится графическое изображение, то при добавлении оно должно попасть в поле графического типа, чтобы Paradox смог отобразить данные. Paradox не интерпретирует данные в dBASE memo, пока не сделано добавление, поэтому вам необходимо самостоятельно убедиться в их совместимости с типом соответствующего поля Paradox-

таблицы.

Добавление записей в таблицы с ключом

Если таблица, в которую вы добавляете записи, имеет ключ, то добавляемые данные не должны вызывать конфликтов значений ключа. Данные, не удовлетворяющие этому требованию, помещаются во временную таблицу Keyviol в вашем личном каталоге. Таблица-источник никогда не изменяется при добавлении, независимо от того, имеет она ключ или нет.

Опции Append и Update

Опции Append и Update панели Options определяет режим операции добавления:

- Append: новые записи добавляются, не оказывая влияния на уже существующие:
    - Если таблица-приемник имеет ключ, Paradox вставляет новые записи в подходящие позиции. Записи, вызывающие конфликт ключа, помещаются в таблицу Keyviol в' вашем личном каталоге. (Вы можете отредактировать их и затем добавить в таблицу-приемник).
    - Если таблица-приемник не имеет ключа, Paradox добавляет новые записи после уже существующих.
- Update: происходит обновление записей таблицы-приемника.
    - Если запись в таблице-источнике не соответствует по значению ключа ни одной записи в таблице-приемнике, то она не добавляется.
    - Если обе записи имеют одинаковые значения ключа, то запись из таблицы-источника замещает запись в таблице-приемнике. Записи, удалённые из таблицы-приемника, Paradox помещает в таблицу Changed в вашем личном каталоге. Обратите внимание, что опция Update допустима только в том случае, если таблица-приемник имеет ключ.
- Append & Uodate: производится добавление новых записей и обновление существующих в соответствии с рассмотренными выше правилами. Таблица-приемник должна иметь ключ

Добавление записей при работе в локальной сети

При добавлении Paradox должен установить блокировку на чтение для таблицы-источника и блокировку на запись для таблицы-приемника. Это означает, что другие пользователи сети в этот момент не могут:

- Изменять данные и структуру любой из двух таблиц
- Выполнять операции, требующие блокировки на запись или полной блокировки для любой из таблиц

Аналогично, вы не можете сделать добавление записей, если другой пользователь установил блокировку на запись или полную блокировку хотя бы одной из нужных вам таблиц.

Если добавление производится для dBASE-таблиц, то Paradox устанавливает для них обеих блокировку на запись. Это делается потому, что у dBASE-таблиц блокировки чтения не существует.

Вычитание записей

Утилита Subtract служит для удаления записей, существующих в одной таблице - вычитаемом, из другой таблицы - уменьшаемого. Вычитать можно только из таблицы, имеющей ключ. Поскольку в dBASE-таблицах не поддерживаются ключи Paradox, опера-ция вычитания для них недопустима (в этом случае используйте DELETE-запрос).

При вычитании Paradox удаляет любую запись из таблицы-уменьшаемого, если зна-чение в ключевом поле этой записи совпадает с соответствующим полем таблицы - вычитаемого. Чтобы произвести вычитание записей, дайте команду Tools / Utilities / Subtract либо проинспектируйте иконку таблицы в окне Folder или Browser и выберите пункт меню Subtract. На экране появится диалоговое окно Subtract (рис. 8.3).

1. Введите имя таблицы - вычитаемого в текстовое окошко In.
2. Введите имя таблицы-уменьшаемого в текстовое окошко From
3. Нажмите ОК.

Paradox предложит подтвердить удаление записей из таблицы-уменьшаемого. Нажмите Yes для подтверждения или No для отмены. Если вы на-жали Yes, Paradox сравнит содержимое двух таблиц и

вычтет совпадающие по значению ключа записи.

При использовании операции Subtract необходимо помнить, что:

- Обе таблицы должны иметь совместимую структуру. Это означает, что порядок следования в них полей совместимых типов должен быть один и тот же. Информация о совместимости полей при реструктурировании Paradox- и dBASE-таблиц содержится в таблицах 3.6 и 3.7 Главы 3.
- Если между таблицами определена система ссылок и таблица-уменьшаемое является родительской таблицей, то операция вычитания не разрешается.

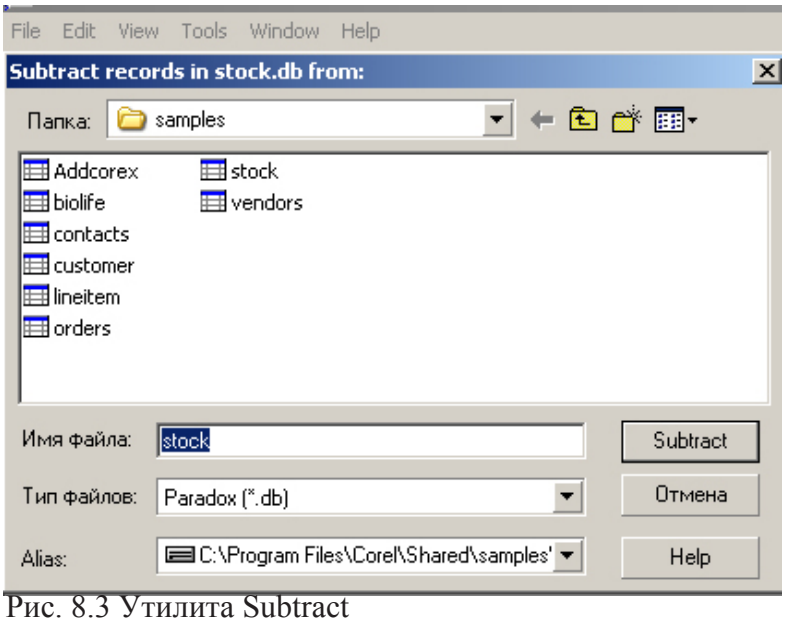


Рис. 8.3 Утилита Subtract

Вычитание при работе в локальной сети

При вычитании Paradox должен установить блокировку на чтение для таблицы-вычитаемого и блокировку на запись для таблицы-уменьшаемого. Это означает, что другие пользователи в этот момент не могут:

- Изменять данные и структуру любой из двух таблиц.
- Выполнять операции, требующие блокировки на запись или полной блокировки для любой из таблиц.

Аналогично, вы не можете производить вычитание, если другой пользователь уста-новил блокировку на запись или полную блокировку для хотя бы одной из нужных вам таблиц.

Копирование объектов

С помощью средств Paradox вы можете копировать таблицы, формы, отчеты, запросы, программы и библиотеки программ. При копировании таблицы Paradox дублирует как данные, так и ее структуру. Поэтому для копирования таблиц всегда используйте утилиту Paradox Copy.

Учтите, что команды копирования файлов DOS или Windows File Manager не копи-руют все связанные с таблицей файлы, содержащие, например, первичные и вторичные индексы, правила проверки корректности данных и данные BLOB-полей. Утилита Paradox Copy всегда, корректно копирует эти файлы.

Чтобы скопировать объект, дайте команду Tools / Utilities / Copy либо проинспектируйте иконку объекта в окне Folder или Browser и выберите пункт меню Copy. На кране появится диалоговое окно Copy (рис. 8.4). В списке File Name перечислены имена всех таблиц вашего рабочего и личного каталогов. Для просмотра имен объектов другого типа укажите нужный тип в раскрывающемся списке Type. Чтобы работать с объектами из других каталогов, используйте раскрывающийся список Path.

- При копировании таблиц Paradox также копирует:
- Ключи (первичные индексы)
- Вторичные индексы (кроме .NDX-файлов dBASE-таблиц)
- Систему ссылок на другие таблицы (см. «Копирование системы ссылок» далее в этом разделе)
- Свойства таблицы, которые были установлены

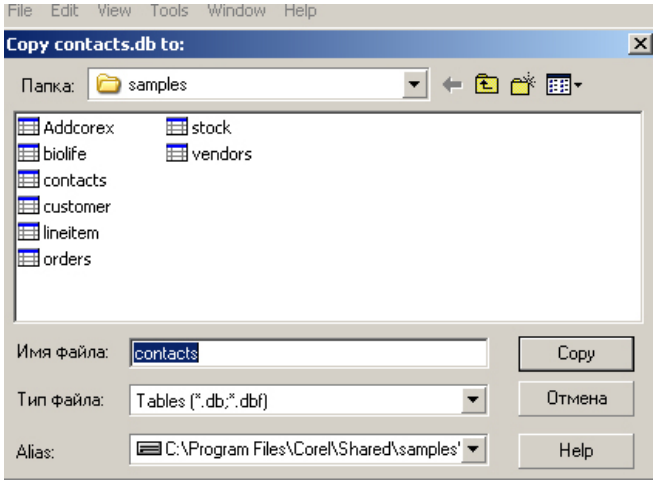


рис. 8.4 диалоговое окно Copy

в окне Table

Пример. Копирование формы из окна Folder

Чтобы скопировать, например, форму Vendors (Поставщики), произведите следующие действия:

1. Определите каталог SAMPLE в качестве рабочего и откройте окно Folder.
2. Проинспектируйте иконку VENDORS.FSL. Выберите пункт COPY. Paradox откроет диалоговое окно Copy.
3. Имя VENDORS появится в текстовом окошке New File Name. Замените его на NEWVEND.
4. Нажмите OK. Paradox создаст копию формы и назовет ее NEWVEND.FSL

Копирование в сети

При копировании в сети Paradox должен установить блокировку на чтение для таблицы-оригинала и полную блокировку для таблицы-копии. Это означает, что:

- Никакой другой пользователь не может изменять содержимое и структуру таблицы-оригинала.
- Если копирование производится в уже существующую таблицу, то на нее не должно быть наложено никаких блокировок.

Вы не можете произвести копирование, если другой пользователь установил блоки-ровку на запись или полную блокировку для таблицы-оригинала.

Копирование системы ссылок

Если вы определили для таблицы систему ссылок, это означает, что вы связали ее с другой таблицей. При копировании таблиц помните следующее:

- Paradox не копирует систему ссылок родительской таблицы.
- При копировании дочерней таблицы ее система ссылок копируется. Это означает, что таблица-копия также будет подчиняться условиям системы ссылок. Чтобы уда-лить ссылки, вы можете воспользоваться операцией реструктурирования таблицы.
- Обе таблицы, связанные системой ссылок, должны находиться в одном каталоге. Копирование дочерней таблицы в другой каталог разрывает ее систему ссылок.

Копирование в таблицу другого типа

Вы можете копировать Paradox-таблицу в dBASE-таблицу и наоборот. Для этого достаточно указать соответствующее расширение для файла таблицы в текстовом окошке New Table Name (.DB для Paradox и .DBF для dBASE). Например, если вы хотите скопировать Paradox-таблицу Customer в dBASE-таблицу Customer, введите CUSTOMER.DB в текстовое окошко From и CUSTOMER.DBF в текстовое окошко To.

При копировании Paradox автоматически изменяет тип полей. Таблица 8.3 показывает, как Paradox изменяет тип полей Paradox-таблицы при копировании в dBASE-таблицу.

Таблица 8.3 Преобразование полей при копировании Paradox-таблицы в dBASE-таблицу

Paradox - поле	dBASE - поле	Побочный эффект
Алфавитно-цифровое	Символьное	
Числовое	Числовое	Устанавливается формат 20.4
Денежное	Числовое	Устанавливается формат 20.4
Короткое число	Числовое	Устанавливается формат 6 (целое число)
Дата	Дата	
Memo	Memo	
Форматированное memo	Memo	Потеря формата
Графическое Memo	Memo	Данные не отображаются
OLE	Memo	Данные не отображаются
Бинарное	Memo	Данные не отображаются

Следует отметить, что если создаваемая при копировании dBASE-таблица не имеет индекса, вычисляемого по выражению, а также числовых полей с плавающей точкой и memo полей, то Paradox

создает таблицу типа dBASE III+. В остальных случаях создается таблица типа dBASE IV. Таблица 8.4 показывает, как производится копирование dBASE-таблицы в Paradox-таблицу.

Таблица 8.4 Преобразование полей при копировании из dBASE -таблицы в Paradox-таблицу

dBASE-поле	Paradox-поле	Побочный эффект
Символьное	Алфавитно-цифровое	
Числовое с пла-вающей точкой	Числовое	Удаляется размер
Числовое	Числовое	Удаляется размер
Логическое	Алфавитно-цифровое	Устанавливается размер 1 и сохраня-ется первый символ
Дата	Дата	
Memo	Memo	Устанавливается размер 1 *

* Предполагается, что данные в dBASE memo поле имеют текстовый вид. Если это не так, то вам следует скопировать информацию из dBASE memo в BLOB-поле Paradox подходящего типа, используя утилиту Add.

Удаление объектов

Средствами Paradox вы можете удалять таблицы, формы, отчёты, запросы, программы или библиотеки программ. Для удаления таблиц всегда используйте утилиту Paradox Delete. Команды удаления файлов DOS или Windows File Manager не удаляют все связанные с таблицей файлы, содержащие, например, первичные и вторичные индексы, правила проверки корректности данных и данные BLOB-полей. Утилита Paradox Delete всегда корректно удаляет нужные файлы.

Чтобы удалить объект, дайте команду Tools /Utilities / Delete.
Если вы дали команду File / … / Delete, Paradox откроет диалоговое окно Delete (рис. 8.5).

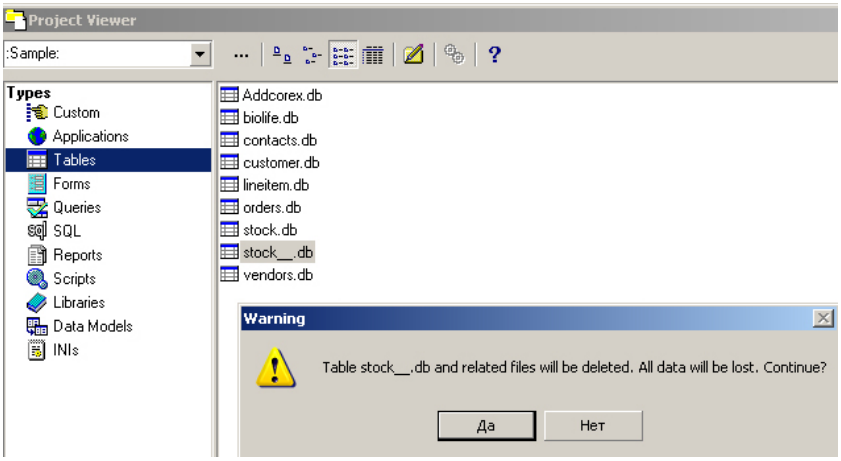


Рис. 8.5 диалоговое окно Delete

В списке File Name перечислены имена объектов только указанного каталога. Чтобы изменить тип отображаемых объектов, воспользуйтесь раскрывающимся списком Type. Чтобы работать с объектами из других каталогов, используйте список Path или кнопку Browse.

Если вы выбрали пункт Delete из меню иконки объекта, Paradox откроет диалоговое окно с запросом на подтверждение удаления. Нажмите Yes для подтверждения, No - для отмены.

Обратите внимание, что если между таблицами определена система ссылок, вы не можете удалить родительскую таблицу. Для этого сначала необходимо удалить из дочерней таблицы ссылки на нее либо очистить или удалить дочернюю таблицу.

Будьте внимательны при удалении объектов: удалённые объекты в Paradox восстановить невозможно. Если удаляется таблица, убедитесь, что она не используется в формах, отчётах или запросах (формы, отчеты и запросы не удаляются вместе с таблицей).

Удаление в сети

При удалении Paradox должен установить полную блокировку таблицы. Это означает, что:

- Ни один другой пользователь не может получить к ней доступ.
- Если для таблицы установлена любая блокировка, ее удаление невозможно.

Очистка таблиц

Утилита Empty удаляет все записи из таблицы, оставляя в неприкосновенности ее структуру (в том числе все ключи, индексы, правила проверки корректности и т.д.).

Очистить таблицу вы можете одним из следующих способов:

- Дайте команду Tools / Utilities / Empty. Paradox откроет диалоговое окно Empty (рис. 8.6).
- Откройте таблицу в окне Table и дайте команду Table / Empty. Paradox выведет запрос на подтверждение удаления. Нажмите Yes для подтверждения либо No для отмены команды очистки.

Учтите, если между таблицами определена система ссылок, вы не можете очистить родительскую таблицу. Для этого сначала необходимо удалить из дочерней таблицы ссылки на нее либо очистить или удалить дочернюю таблицу.

В списке File Name перечисляются имена всех таблиц рабочего и личного каталогов. Чтобы работать с файлами из других каталогов, используйте раскрывающийся список Path или кнопку Browse.

Введите имя очищаемой таблицы в текстовое окошко Empty Table. После нажатия кнопки OK появится диалоговое окно для подтверждение команды удаления. Нажмите Yes для очистки таблицы, No - для отмены.

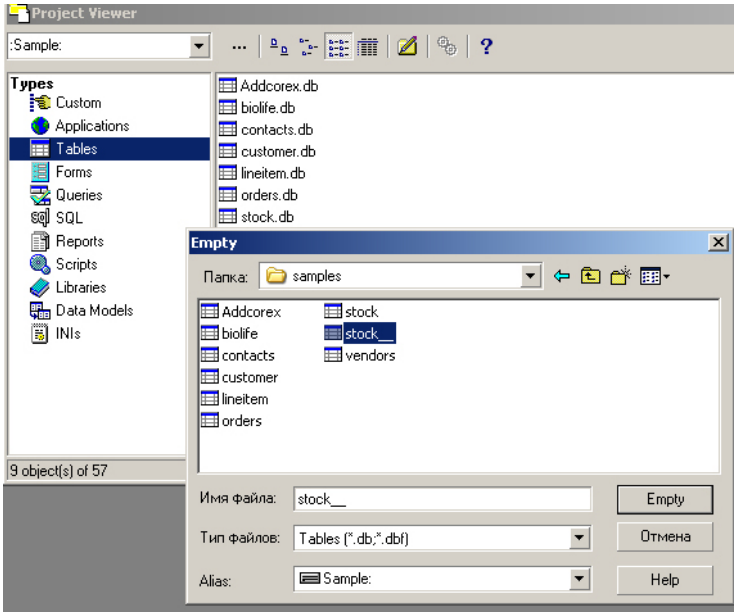


рис. 8.6 утилита Empty

Очистка таблиц в сети

При очистке Paradox должен установить полную блокировку таблицы. Это означает, что:

- Ни один другой пользователь не может получить к ней доступ.
- Если для таблицы установлена блокировка любого типа, ее очистить невоз-можно

Переименование объектов

Средствами Paradox вы можете переименовывать таблицы, формы, отчёты, запросы, программы и библиотеки. Для переименования таблиц всегда используйте утилиту Paradox Rename. Аналогичные команды DOS или Windows File Manager не переименовывают связанные с таблицей файлы, содержащие, например, первичные и вторичные индексы, правила проверки корректности данных и данные BLOB-полей. Утилита Paradox Rename всегда корректно переименовывает все требуемые файлы.

Для переименования объекта вы можете сделать одно из следующих действий:

- Откройте таблицу в окне Table и дайте команду Table /Rename. Paradox откроет диалоговое окно Rename (рис. 8.7). В текстовое окошко To следует ввести новое имя.
- В списке File Name окна Rename перечислены имена объектов рабочего и личного каталогов. Для смены типа отображаемых объектов служит раскрывающийся список Type. Чтобы работать с объектами из других каталогов, используйте список Path. Введите имя существующего объекта в текстовое окошко From и новое имя в текстовое окошко «To». Нажмите OK.
- При переименовании объекта необходимо помнить следующее:
- Нельзя переименовывать таблицу, изменяя ее тип: Paradox-таблица должна переименовываться в Paradox-таблицу, dBASE-

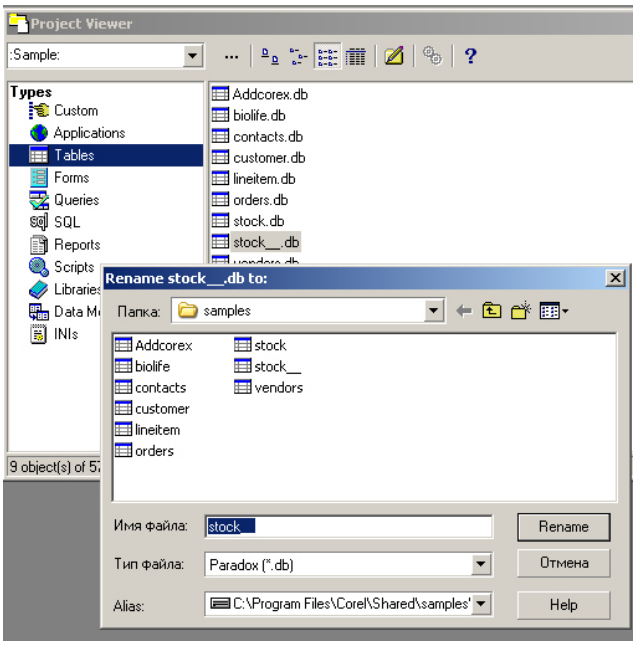


рис. 8.7 утилита Rename

таблица - в dBASE-таблицу. Чтобы изменить тип таблицы, вы должны скопировать ее (см. раздел «Копирование в таблицу другого типа « в этой главе).

- Если между таблицами задана система ссылок, вы не можете переименовать родительскую таблицу».
- При переименовании таблицы вы можете поместить ее в другой каталог. Для этого требуется ввести новое имя вместе с полным именем нужного каталога. Однако, нельзя перемещать в другой каталог объекты прочих типов (формы, отчеты и т.д.).

Будьте внимательны при переименовании таблиц. Переименованную таблицу нельзя найти по ссылающимся на нее документам: формы, отчеты и запросы будут ссылаться на таблицу со старым именем. Если вы откроете такой объект, Paradox запросит новое имя таблицы, с которой он попытается связать вызванную вами форму или отчет.

Переименование в сети

При переименовании Paradox устанавливает полную блокировку таблицы. Это означает, что:

- Ни один другой пользователь не может получить к ней доступ.
- Если для таблицы установлена блокировка любого типа, ее переименование невоз-можно.

Экспорт данных

Paradox предоставляет возможность экспортировать данные из таблиц во внешние файлы различных форматов. С помощью утилиты Export вы можете производить передачу данных из среды Paradox в другие прикладные программы. Экспортировать данные можно только во вновь создаваемые файлы.

Для экспорта данных из таблицы дайте команду File / Export либо проинспектируйте иконку таблицы в окне Folder или Browser и выберите пункт меню Export. Paradox откроет диалоговое окно Export (рис. 8.9). Форматы файлов, в которые Paradox может экспортировать данные, перечислены в списке Export File Type.

Экспорт в форматированный файл

Для экспорта данных в ASCII-файл, в котором длина строки определяется размером поля, используйте тип экспортного файла - Delimited Text.

В списке File Export выберите необходимую таблицу, а в списке Export File Type укажите тип Delimited Text. Нажмите OK. Paradox откроет диалоговое окно Delimited ASCII Export .

Следующим этапом будет предложено окно с указанием типов файлов для экспорта (рис.8,9):

По умолчанию поля в экспортируемом файле разделяются запятыми, а нечисловые значения заключаются в двойные кавычки. Записи разделяются символами возврата каретки и перехода на новую строку. Если требуется, чтобы файл имел другой формат,

Нажмите кнопку Options в окне Delimited ASCII Export. Paradox

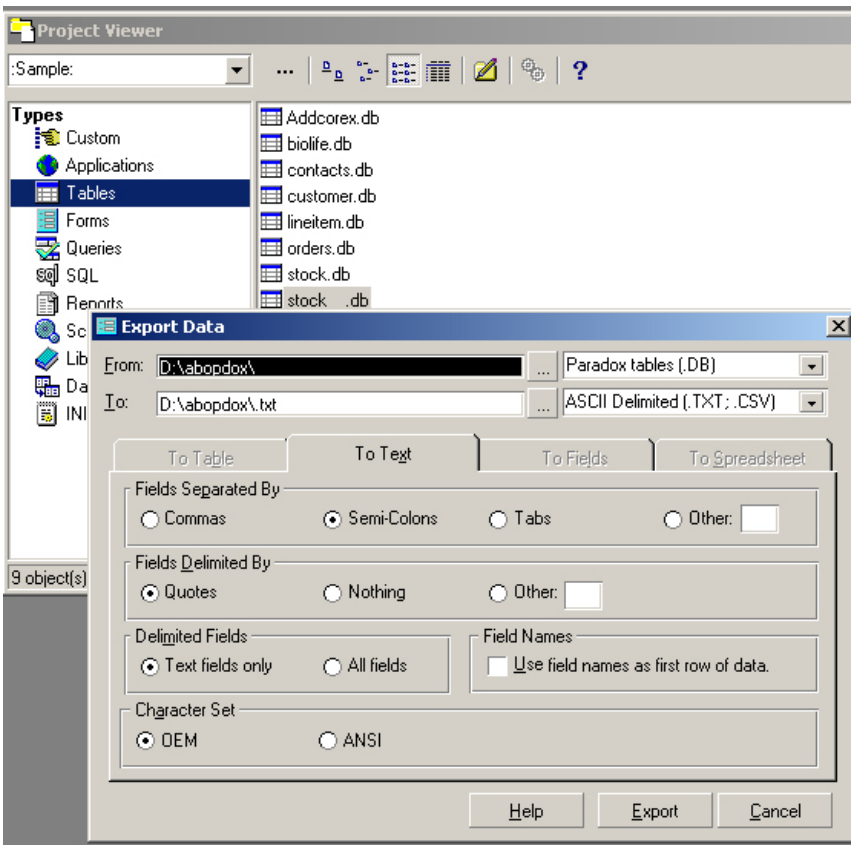


Рис. 8.8 экспорт данных в ASCII-файл

откроет диалоговое окно Text Options .

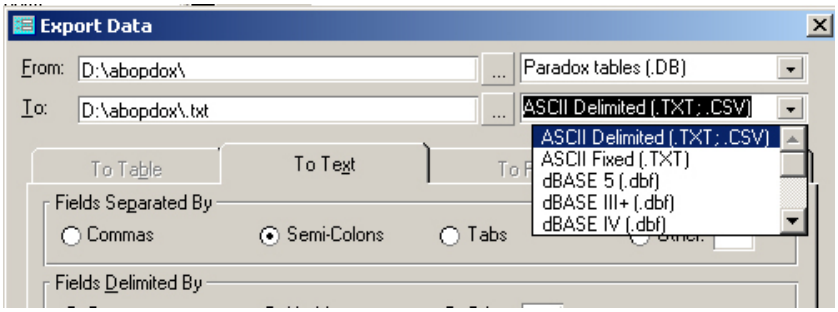


рис. 8.9 экспорт данных в ASCII-файл (продолжение)

- Панель Field Separated By используется для задания символов, разделяющих поля. Вы можете выбрать запятую, символ табуляции или другие символы. Для выбора других символов-разделителей включите опцию Other и введите их в текстовое окошко Other.
- Панель Field Delimited By служит для выбора символов, в которые заключается со-держимое нечисловых полей. Вы можете выбрать кавычки, любые другие символы либо вообще никаких (опция Nothing).
- С помощью панели Delimited Fields можно указать, в каких случаях содержимое полей заключается в кавычки, а в каких - в определенные вами символы. Для заключения в кавычки только текстовых (символьных или алфавитно-цифровых) по-лей включите опцию Text Field Only, для заключения в кавычки полей всех типов - All Fields.
- Учтите, что Paradox не экспортирует memo (Paradox или dBASE), форматированные memo, графические, OLE и бинарные поля в форматированный файл.
- Панель Character Set используется для выбора стандарта ANSI или OEM, в котором будет осуществляться экспорт.

После установки необходимых опций нажмите OK, чтобы вернуться в окно Delimited ASCII Export. На рисунке 8.10 показан результат экспорта таблицы Contacts .

```
"Pan";"Jerry";"Kauai Dive Shoppe";"808-555-0270"
"Dealy";"Devin";"Unisco";"809-555-7201"
"Cuenin";"Peggy";"Sight Diver";"357-6-876708"
"Minford";"Phil";"Cayman Divers World Unlimited";"809-555-3421"
"Montengro";"Tod";"Tom Sawyer Diving Centre";"809-555-7288"
"Rioux";"Rick";"Blue Jack Aqua Center";"808-555-8903"
"Klaner";"Tara";"VIP Divers Club";"809-555-6864"
"Bondogji";"Murray";"Ocean Paradise";"808-555-8231"
"Cordray";"Carolyn";"Fantastique Aquatica";"57-1-773421"
```

Рис. 8.10 Результат экспорта таблицы Contacts

Экспорт в файл с фиксированной длиной полей записи

Вы можете экспортировать данные из таблицы в ASCII-файл, в котором длина полей каждой записи одинакова.

Выберите таблицу в списке Table Export и укажите тип Fixed Length Text на панели Export File Type. Нажмите OK. Paradox откроет окно Fixed Length ASCII Export.

При экспорте в файл с фиксированной длиной полей записей Paradox создает временную таблицу EXPORT.DB в вашем личном каталоге. Эта таблица используется при спецификации формата создаваемого файла: для каждого поля вы указываете стартовую позицию и количество символов, в которые будут выведены его данные.

На панели Export Specification находятся кнопки следующих операций для работы с таблицей Export:

- Save: сохраняет установленные вами параметры таблицы. (В противном случае таблица Export будет удалена при выходе из Paradox или изменении личного каталога.)
- Load: загружает параметры последней ранее сохранённой Export- таблицы.
- Clear: удаляет параметры таблицы Export.

Экспорт в электронную таблицу

Вы можете экспортировать данные в различные электронные таблицы. Для этого укажите требуемую таблицу в окне Table Export и выберите требуемый формат файла электронной таблицы на панели Export File Type. Нажмите OK. Paradox откроет диалоговое окно Spreadsheet Export (рис. 8.11). Результат экспорта данных из таблицы Lineitem в электронную таблицу Excel показан на рисунке 8.12.

После выбора требуемого формата Paradox автоматически помещает необходимое расширение файла в текстовое окошко New File Name. В таблице 8.5 приведены расширения файлов различных электронных таблиц.

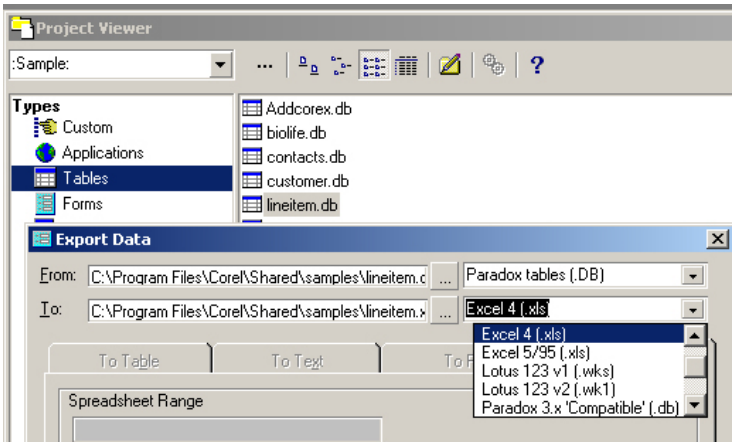
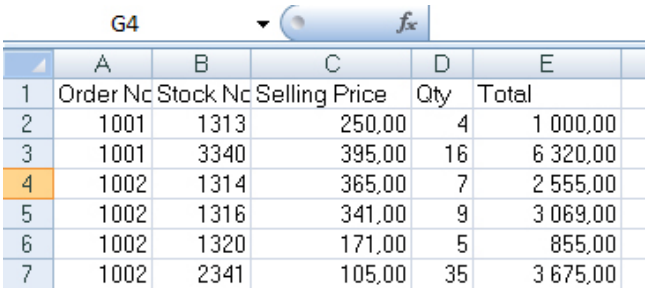


Рис. 8.11 диалоговое окно Export для электронных таблиц



G4

	A	B	C	D	E
1	Order Nc	Stock Nc	Selling Price	Qty	Total
2	1001	1313	250,00	4	1 000,00
3	1001	3340	395,00	16	6 320,00
4	1002	1314	365,00	7	2 555,00
5	1002	1316	341,00	9	3 069,00
6	1002	1320	171,00	5	855,00
7	1002	2341	105,00	35	3 675,00

Рис. 8.12 вид таблицы, экспортированной в Excel

Таблица 8.5 Расширения файлов электронных таблиц

Формат	Расширение
Quattro Pro Win	.WB1
Quattro Pro DOS	.WQ1
Quattro	.QKQ
Lotus 2.x	.WK1
Lotus 1 .A	.WKS
Excel	.XLS

При экспорте данных в электронную таблицу Paradox представляет каждую запись в виде строки, поле - в виде колонки. Если содержимое поля не помещается в отведенную колонку, то часть информации оказывается не отображённой.

Если данные не укладываются в заданный диапазон значений в электронной таблице, в ячейки электронной таблицы записывается значение ERROR.

Импорт данных

С помощью утилиты Import вы можете производить передачу данных из внешних файлов различных форматов в Paradox. Импортировать информацию можно только во вновь создаваемые таблицы.

Для импорта данных дайте команду Tools / Utilities / Import, Paradox откроет диалоговое окно File Import . Paradox позволяет импортировать данные из файлов следующих форматов:

- форматированный текстовый файл
- файл с фиксированной длиной строк
- Quattro Pro for Windows u Quattro Pro (DOS)
- Quattro -Lotus 2.x и 1.A u
- Excel 3.0/4.0

Для выбора необходимого формата используется раскрывающийся список Type. Имена всех файлов выбранного формата из рабочего каталога будут помещены в список файлов. Если требуемый

файл расположен в другом каталоге, вы можете либо ввести его полное имя в текстовое окошко File Name, либо воспользоваться списком Path или кноп-кой Browse. Выберите файл-источник и нажмите ОК. Вид импортированной информации будет зависит от формата файла-источника.

Импорт из электронной таблицы

Если файл-источник - электронная таблица, то на экране появится диалоговое окно Spreadsheet Import (рис. 8.13). При импорте дянных из электронной таблицы Paradox автоматически определяет тип каждой колонки данных (таблица 8.6 показывает, как это происходит).

Таблица 8.6 Преобразование типов при импорте из электронной таблицы

Данные в электронной табли-це	Paradox-поле	dBASE-поле
Метки	Алфавитно-цифровое	Символьное
Числа (целые)	Короткое целое	Число с плавающей точкой формата 5.0
Числа (не целые)	Числовое	Число с плавающей точкой формата 20.4
Числа в денежном формате	Денежное	Число с плавающей точкой формата 20.4
Числа в формате даты	Дата	Дата

Типы данных преобразуются в соответствии со следующими правилами:

- Если колонка содержит метку (текст), то ее содержимое приводится к алфавитно-цифровому типу (Paradox), либо символьному типу (dBASE).
- Если колонка содержит информацию числового типа и типа даты, то ее содержимое приводится к алфавитно-цифровому типу (Paradox), либо к символьному типу (dBASE).
- Если данные в колонке числового типа, но в денежном формате, то они приводятся в Paradox к денежному типу.
- Если колонка содержит одновременно информацию числового и денежного типов, то ее содержимое преобразуется к числовому типу.

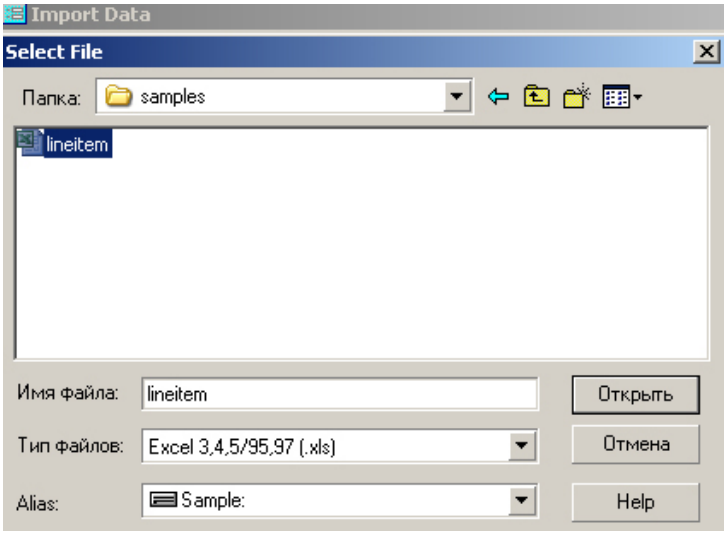


рис. 8.13 диалоговое окно Spreadsheet Import

Исходя из этих правил, Paradox часто импортирует числа из неотредактированных электронных таблиц в алфавитно-цифровые поля. Например, колонки в электронных таблицах часто содержат числа, разделённые дефисом. Поскольку дефис и цифры могут располагаться вместе только в полях алфавитно-цифрового типа Paradox-таблиц, то данные таких колонок приводятся, соответственно, к алфавитно-цифровому типу.

Помимо целой электронной таблицы вы можете импортировать отдельный ее блок. Для этого введите имя первой ячейки вводимого блока в текстовое окошко From Cell и имя последней ячейки - в текстовое окошко To Cell либо выберите именованный диапазон в списке Named Range (это возможно только в том случае, если именованный диапазон был определен при создании электронной таблицы).

Чтобы избежать проблем с преобразованием данных при импорте, сначала отредактируйте электронную таблицу следующим образом:

1. Удалите «лишние» символы (дефисы, звездочки и восклицательные знаки).
2. Убедитесь, что в каждой колонке содержится только один вид данных, и используется только одна опция форматирования.
3. Поместите заголовок колонки в верхний ряд выбранного диапазона. Paradox использует этот ряд

для генерации имен полей. (Если в электронной таблице нет заголовков колонок, выключите опцию Get Field Names From First Row в иалоговом окне Spreadsheet Import

Определение имен полей

При импорте Paradox использует первый (верхний) ряд, содержащий текстовую информацию, для определения имен полей. Если такой ряд отсутствует, Paradox именует поля FIELD001, FIELD002 и т.д. Если несколько полей имеют одно и то же имя, Paradox дополняет их номерами (например, Customer!, Customer2).

Импорт форматированного текста

Выберите файл-источник и установите тип Delimited Text в раскрывающемся списке Type. Paradox откроет диалоговое окно Delimited ASCII Import.

По умолчанию ожидается, что поля в файле-источнике разделены запятыми (изменить разделители можно в Fields Separated By), а содержимое текстовых полей взято в двойные кавычки (изменить можно в строке Fields Delimited By). Для переопределения этих установок нажмите кнопку Options. Paradox откроет диалоговое окно Text Options .

- Панель Fields Separated By служит для определения символов, разделяющих поля в файле-источнике.
- Панель Fields Delimited By служит для установки символов, в которые заключены данные в файле-источнике.
- Панель Delimited Fields определяет, все ли поля в файле - источнике заключены в двойные кавычки (или символы, определенные в панели Fields Delimited By) либо только числовые.
- Панель Character Set задает стандарт импорта ANSI или OEM.

При импорте данных из форматированного текстового файла Paradox сначала просматривает его и определяет типы и количество полей. Затем создается новая таблица с именем, заданным в текстовом окошке New Table Name, и производится импорт данных.

Импорт из файла с фиксированной длиной строк

Если вы выбрали тип файла Fixed Length Text из раскрывающегося списка Type в ок-не File Import, то Paradox откроет окно Fixed Length ASCII Import . Введите имя создаваемой таблицы в текстовое окошко New Table Name и задайте тип таблицы (Paradox, dBASE и т.д.).

При импорте данных из файла с фиксированной длиной записей Paradox создает временную таблицу IMPORT.DB в вашем личном каталоге. Она используется в диалоговом окне Fixed Length ASCII Import для определения имен и типов полей в новой таблице: для каждого поля вы указываете тип, стартовую позицию и количество символов строки файла, в которых находятся данные, импортируемые в данное поле.

На панели Import Specification находятся кнопки следующих операций для работа с таблицей Import:

- Save: сохраняет установленные вами параметры таблицы. (В противном случае таблица Import будет удалена при выходе из Paradox или изменении личного каталога.)
- Load: загружает параметры последней ранее сохранённой таблицы Import.
- Clear: удаляет параметры таблицы Import.

После определения структуры создаваемой таблицы, нажмите ОК. Paradox импортирует данные в таблицу с именем, заданным в текстовом окошке New Table Name окна Fixed Length ASCII Import.

Использование паролей

Вы можете назначить пароль доступа к таблице, используя диалоговое окно Create Table или Restructure Table (см. Главу 3). При попытке открыть такую таблицу Paradox запрашивает у вас пароль.

Предположим, что вы закрыли таблицу, а затем открываете ее снова. Если вы не вы-ходили из Paradox, то пароль не запрашивается, поскольку факт вашего доступа к таблице запоминается програм-

мой. При выходе из Paradox все пароли отменяются.

Если вы хотите отменить пароль, не выходя из Paradox, выберите пункт Tools / Security / Passwords. При этом появится диалоговое окно Enter Password(s) (рис. 8.14).

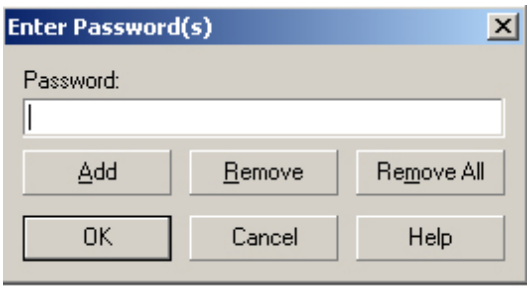


Рис. 8.14 диалоговое окно Enter Password(s)

Введите в текстовое окошко Password пароль, который вы хотите удалить из памяти Paradox (вводимые вами символы изображаются в виде звездочек (*)) и нажмите кнопку Remove.

После этого вам потребуется ввести его снова, прежде чем опять получите доступ к таблице. Чтобы удалить из памяти Paradox все пароли, нажмите кнопку Remove АИ. При этом все ранее открытые и затем закрытые таблицы окажутся вновь защищёнными (на таблицы, остающиеся открытыми в этот момент, отмена пароля не распространяется).

Получение информации о структуре таблице

Если необходимо узнать или вспомнить структуру таблицы, дайте команду Tools / Utilities / info Structure.

Paradox откроет диалоговое окно Structure Information (рис. 8.15), показывающее типы и размеры полей таблицы, все ключи, заданные правила проверки корректности данных, индексы, таблицы-справочники и информацию о системе ссылок.

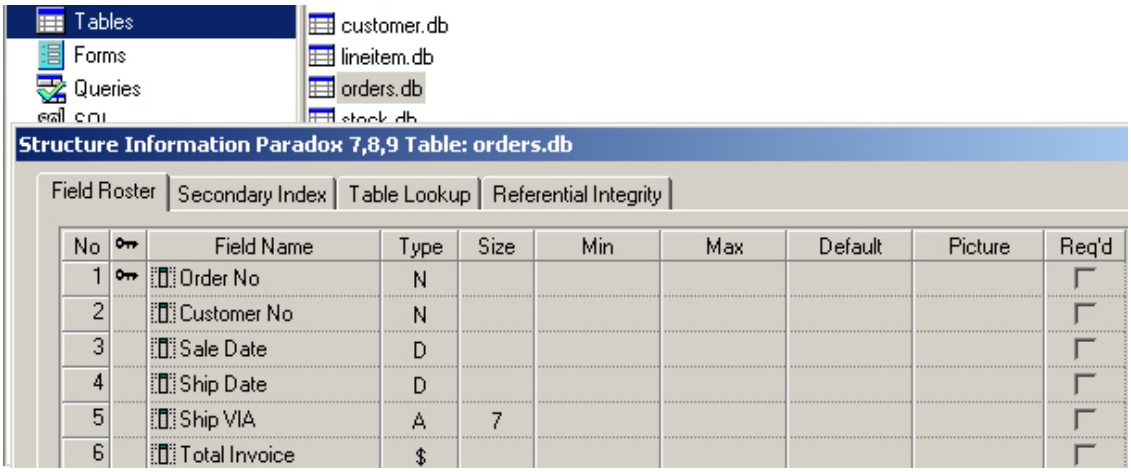


Рис. 8.15 диалоговое окно Structure Information

Глава 9. Разрабатываемые документы

Разрабатываемыми документами в Paradox называются формы и отчеты Разрабатываемые документы могут выводиться как на экран компьютера, так и распечатываться на принтере

В данной главе рассматривается первый этап разработки документа:

- Выбор таблицы (или таблиц), на основе которой создается документ
- Определение взаимосвязи между таблицами в многотабличном документе
- Выбор полей таблицы, которые вы хотите поместить в документ
- Выбор одного из предлагаемых Paradox чертежей в качестве исходного для вашего документа

Прежде чем начинать разработку документа, необходимо решить, что вы хотите создать форму или отчет.

Формы в Paradox используются для отображения данных из ваших таблиц - в самых различных форматах и в сочетании с графическими объектами. Формы разрабатываются, в основном, для использования на экране и, прежде всего, для облегчения процесса ввода данных.

Отчеты в Paradox служат для извлечения из базы данных необходимой информации. Они позволяют логически группировать данные и распечатывать их в виде, удобном для дальнейшего использования содержащейся в них информации.

Разработка модели данных

Моделью данных называется графическое представление взаимных связей между таблицами, на основе которых разрабатывается документ. Она предоставляет вам простой и наглядный способ указать Paradox, какие таблицы будут использованы вами в документе, и как они должны между собой взаимодействовать.

Разработка любого документа начинается с построения модели данных. Войдите в пункт меню File / New и выберите тип документа (форму или отчет). Paradox выведет на экран диалоговое окно Data Model (рис. 9.1). Это окно одинаково и для формы и для отчета.

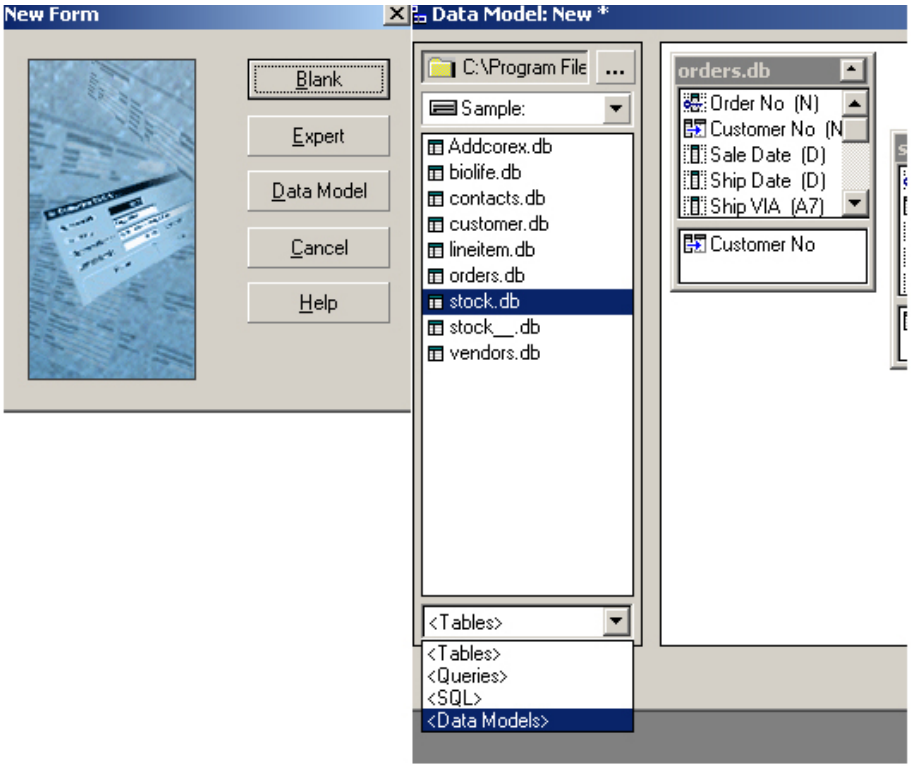


Рис. 9.1 Диалоговое окно Data Model

Когда вы выбираете из списка File Name таблицу и помещаете ее на панель модели данных, вы указываете Paradox, что в данном документе будут использоваться поля из данной таблицы.

Если щелкнуть мышью маркер раскрывающегося списка Type, вы увидите два пункта: <Tables> и <Queries>. Пункт <Queries> используется для разработки документа, служащего для представления информации, получаемой в результате выполнения запроса (см. раздел «Разработка документов на основе запросов» ниже в данной главе).

Бланк документа

Вы можете вообще не использовать ни одной таблицы в качестве основы документа. При этом Paradox создаст бланк документа, не связанный ни с какими данными.

Если вы, начиная разработку нового документа, не связали его ни с одной таблицей, это всегда можно будет сделать позднее. Для этого, находясь в окне разработки (Form Design или Report Design), щелкните мышью кнопку Data Model на SpeedBar и вы вернетесь в окно Data Model, в котором можете

добавить в модель данных любую таблицу.

Однотабличная модель данных

В основе однотабличных документов лежит простейшая модель данных - одна единственная таблица. Учтите, что некоторые возможности диалогового окна Data Model используются только при разработке многотабличных документов.

Пример. Построение однотабличной модели данных

- Дважды щелкните мышью (или один раз мышью и потом Enter) в списке File Name имя нужной вам таблицы.
- Оно появится в отдельном текстовом окошке справа.
- Нажмите мышью кнопку OK. Paradox закроет диалоговое окно Data Model.
- Если вы предпочитаете работать с клавиатурой, просто введите имя таблицы в текстовое окошко File Name и нажмите клавишу Enter.

Инспектирование таблиц в модели данных

Таблицы, находящиеся на панели модели данных, можно инспектировать. Для этого либо щелкните ее дважды правой клавишей мыши либо выберите ее при помощи Tab и клавиш управления курсором и нажмите F6. Вам будет предоставлено меню свойст таблицы (на рисунке 9.2 показано меню, относящееся к разработке формы). Если же вы создаете отчет, оно будет состоять из имен, типов и размеров полей.

- Пункт Fields содержит список имен, типов и размеров полей таблицы.
- Пункт Order / Range вызывает диалоговое окно Order / Range, рассмотренное в Главе 4.
- Пункт Read-Only служит для защиты таблицы от редактирования при работе с ней в данной форме (в других формах и в окне Table защита действовать не будет).
- Пункт Strict Translation служит для ограничения набора используемых символов с целью их более простого преобразования при переходе с одного языкового драйвера на другой.

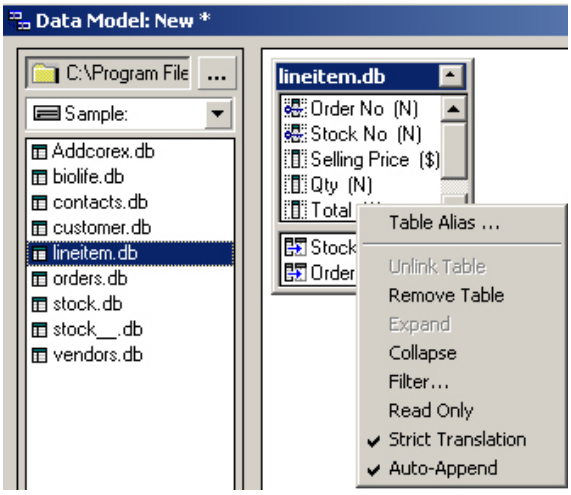


Рис. 9.2 Инспектирование таблицы в диалоговом окне Data Model

Разработка многотабличной модели данных

Paradox позволяет создавать документы, использующие данные одновременно из нескольких таблиц. При построении модели данных такого многотабличного документа вы:

- Во-первых, определяете для него набор таблиц (помещаете таблицы на панель модели данных).
- Во-вторых, определяете отношения между таблицами (связываете их).

Не забывайте, что из окна разработки документа (Form Design или Report Design) вы всегда можете вернуться в окно Data Model (с помощью кнопки на SpeedBar или командой Data Model), чтобы ввести или удалить из модели данных какую-либо таблицу, а также чтобы отредактировать связи между таблицами.

Ввод таблицы в модель данных

Все таблицы, данные из которых вы собираетесь использовать в разрабатываемом документе, должны быть размещены на панели модели данных окна Data Model.

Чтобы ввести таблицу в модель данных, щелкните дважды мышью ее имя в списке File Name либо щелкните его один раз, а затем нажмите мышью кнопку-стрелку Add Table (рис. 9.1). Проделайте это для каждой таблицы, которая должна быть помещена на панель модели данных.

Если вы предпочитаете работать с клавиатурой, с помощью клавиши Tab переместитесь к списку File Name, клавишами управления курсором выберите в списке нужную таблицу и нажмите Alt+A.

Повторите эту операцию для всех таблиц разрабатываемого документа.

Если вы планируете использовать в документе очень много таблиц, рекомендуется после ввода в модель данных очередной таблицы сразу связывать ее с таблицами, уже находящимися в модели. Это позволит вам избежать пролистывания (скроллинга) панели модели данных при просмотре всего списка содержащихся в ней таблиц.

Удаление таблиц из модели данных

Удалить таблицу из модели данных можно, выбрав ее на панели при помощи мыши и нажав кнопку-стрелку Remove Table либо комбинацию клавиш Alt+D.

Учтите, если таблица уже связана с другими таблицами модели данных, то для ее удаления необходимо сначала разорвать связь, воспользовавшись Unlink.

Связи между таблицами

Чтобы понять, как таблицы связываются между собой в документах, необходимо знать, как Paradox производит сортировку и поиск данных, основываясь на значениях ин-дексов (для получения информации о ключах, индексах и других способах связывания таблиц в базах данных обратитесь к Главе 3).

Таблицы, которые вы хотите связать между собой, должны иметь общее поле. Имена полей обеих таблиц не обязательно должны совпадать, но их тип и размеры обязаны быть идентичными.

Например, пусть ваша модель данных состоит из двух таблиц - Customer и Orders. Поле Customer No таблицы Orders содержит те же значения, что и аналогичное поле таблицы Customer. Проще и эффективнее хранить информацию о клиентах и о заказах в разных таблицах, иначе вам пришлось бы каждый раз вводить сведения о клиенте (адрес, телефон и т.д.) при вводе очередного заказа. Но иногда бывает необходимо одновременно видеть данные из обеих таблиц. В этом случае вы должны связать обе таблицы между собой. При этом Paradox для каждого значения поля Customer No таблицы Customer находит соответствующие значения в поле Customer No таблицы Orders, а позволяет ему это делать индекс.

Paradox использует индекс, чтобы «помнить», в каком месте таблицы находится нуж-ное значение. Когда вы задаёте по какому-либо полю вторичный индекс (см. Главу 3), Paradox просматривает все значения в этом поле и создает файл, в котором записывает позицию каждого значения в таблице. Впоследствии это позволит Paradox легко и быстро отыскать в таблице нужное вам значение. Связывая между собой две таблицы, вы указываете Paradox вычислить определённое значение в главной таблице (таблице, от которой идет связь) и отыскать соответствующие значения в связанной таблице (таблице, к которой идет связь). Это означает, что связанная таблица обязательно должна быть проиндексирована по полю, которое вы используете для связывания. Здесь может использоваться как первичный индекс (ключ), так и вторичный.

Рис. 9.3 (режим редактирования) иллюстрирует связывание таблиц Customer и Orders. Данная форма показывает, что каждой записи главной таблицы (Customer –верхнее поле) соответствует одна или несколько записей связанной таблицы (Orders – среднее поле).

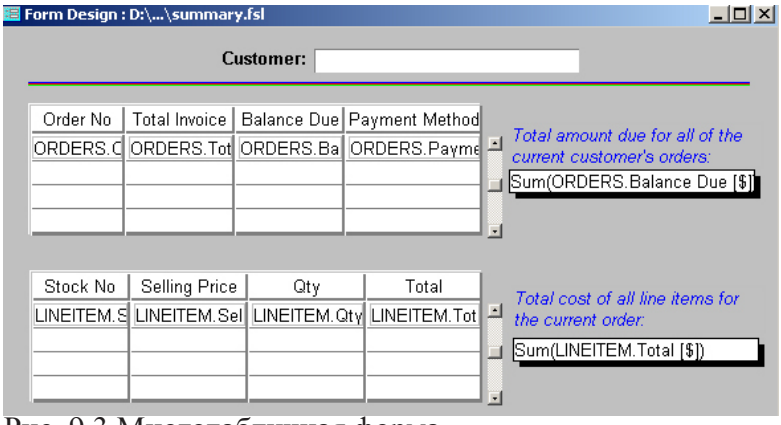


Рис. 9.3 Многотабличная форма

Типы связей

Вы можете создавать между таблицами как однозначные отношения, называемые также связью один-к-одному (1-1) или много-к-одному (M-1), так и многозначные отношения, или связь один-к-многим (1-M).

Однозначные отношения

Однозначными называются такие отношения между таблицами, при которых каждая запись одной таблицы связана не более, чем с одной записью другой таблицы. Например, таблицы Lineitem и

Stock связаны однозначным отношением: каждому пункту каждого заказа соответствует единственное наименование вида товара из имеющихся на вашем складе.

При наличии связи типа много-к-одному (M-1) нескольким записям главной таблицы может соответствовать одна запись связанной таблицы. Например, в таблице Lineitem перечислены все пункты заказов, сделанных вашими клиентами. Группа из нескольких пунктов, заказанных одновременно, относится к одному заказу, т.е. несколько записей таблицы Lineitem ссылаются на одну запись в таблице Orders.

Многозначные отношения

Многозначными называются такие отношения между таблицами, при которых каждой записи одной таблицы может соответствовать более одной записи другой таблицы Например, какой-либо клиент (одна запись таблицы Customer) мог сделать любое количество заказов - от нуля до неограниченного (несколько записей таблицы Orders). Данная связь относится к типу один-к-многим (1-M).

Связывание таблиц

Данный раздел описывает использование диалогового окна Define Link для связывания двух Paradox-таблиц. Информация по связыванию dBASE-таблиц изложена в разделе «Связывание dBASE-таблиц» ниже в данной главе. После того, как вы поместили на панель модели данных документа таблицы, которые должны быть связаны между собой, вы должны определить эту связь.

Помните, что нельзя связывать таблицы по полям типа memo, форматированное memo, а также по графическим, бинарным и OLE-полям.

Если вы ранее задали систему ссылок между таблицами (см. Главу 3), Paradox автоматически соответствующим образом свяжет эти таблицы и в модели данных документа.

Пример. Связывание таблиц, для которых определена система ссылок

Предположим, необходимо связать в модели данных документа таблицы Customer и Orders.

1. Откройте диалоговое окно Data Model.
2. Сделайте двойные щелчки мыши над CUSTOMER.DB и ORDERS.DB в списке File Name. Paradox поместит обе таблицы на панель модели данных.
3. Установите курсор мыши над именем таблицы Customer, находящемся на панели модели данных (при этом его изображение изменяется). Нажмите левую клавишу мыши и переместите курсор мыши на таблицу Orders. Отпустите клавишу мыши.
4. Paradox знает о наличии системы ссылок между двумя таблицами и автоматически связывает их по полю Customer No (рис. 9.4).

Если между двумя таблицами не определена система ссылок, для связывания их в модели данных вам придется использовать диалоговое окно Define Link.

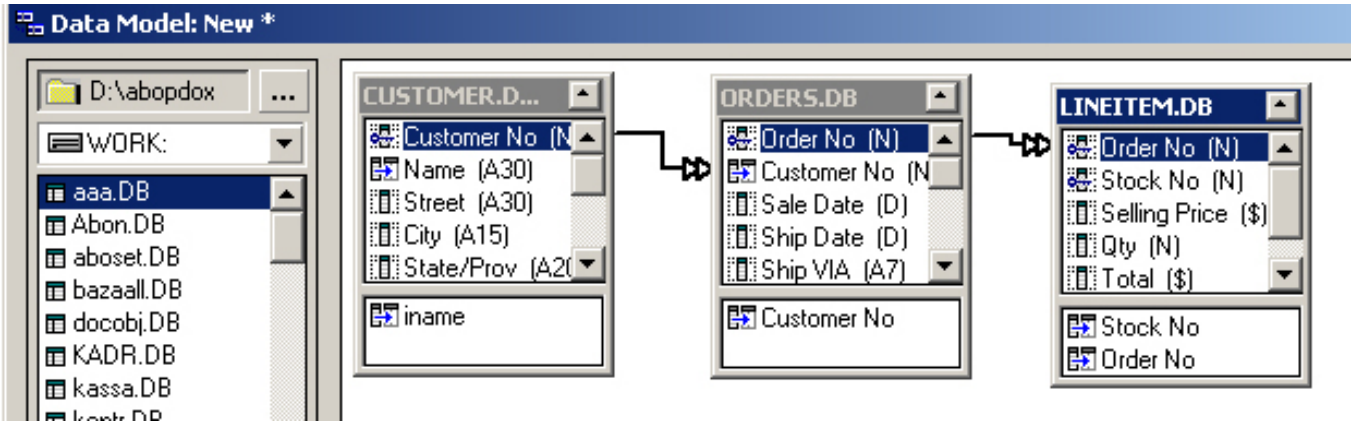


Рис. 9.4 Связывание таблиц, для которых определена система ссылок

Пример. Связывание таблиц, для которых система ссылок не определена

Поскольку для всех таблиц-примеров определена система ссылок, для выполнения данного примера вам придется создать новую таблицу: скопируйте таблицу Customer в Cust2 (копирование таблиц рассматривалось в Главе 8).

1. Откройте диалоговое окно Data Model.
2. Сделайте двойные щелчки мыши над CUST2.DB и ORDERS.DB в списке File Name. Paradox по-

местит обе таблицы на панель модели данных.

3. Установите курсор мыши над именем таблицы Cust2, находящемся на панели модели данных. Нажмите левую клавишу мыши и переместите курсор мыши на таблицу Orders. Отпустите клавишу мыши. Paradox откроет диалоговое окно Define Link (рис. 9.5)
4. На диаграмме связей под именем таблицы Cust2 находится имя поля Customer No, поскольку по умолчанию Paradox предлагает для создания связи между таблицами ключевое поле.
5. Под именем таблицы Orders находится имя поля Customer No. Paradox использует индексное поле связанной таблицы, полностью соответствующее ключевому полю главной таблицы. Между двумя полями проведена линия, обозначающая связь между таблицами.
6. Если вас устраивает предлагаемый способ связывания таблиц в документе, нажмите OK, и Paradox вернет вас в окно Data Model.

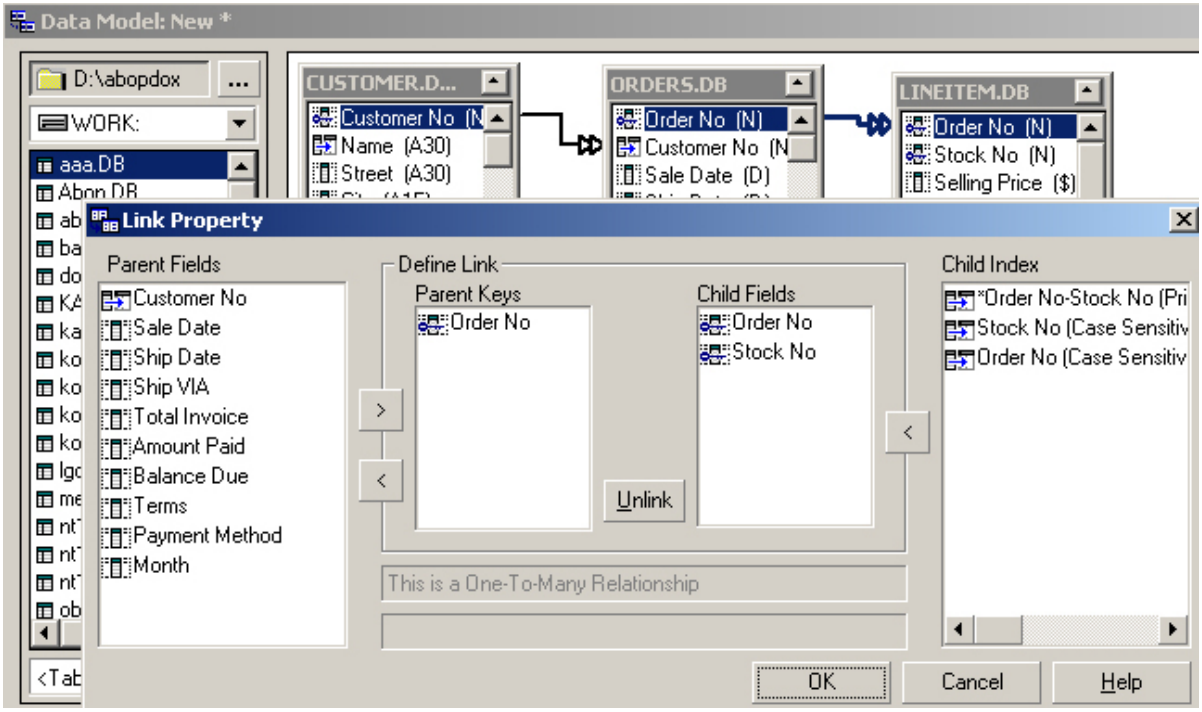


Рис. 9.5 Диалоговое окно Define Link Автоматическое связывание таблиц

В примере «Связывание таблиц, для которых определена система ссылок» Paradox автоматически без вашего участия связал две таблицы. Если в таблицах существует пара полей, по которым можно связать таблицы (например, ключевое поле главной таблицы имеет тот же тип и размер, что и проиндексированное поле связанной таблицы), Paradox открывает окно Define Link с уже готовой связью. Вы можете либо принять ее, нажав OK, либо нажать кнопку Unlink, чтобы разорвать автоматически созданную связь и самостоятельно определить новую.

Связывание таблиц вручную

В списке Fields диалогового окна Define Link перечисляются все поля главной таблицы. Выберите в нем поле, по которому необходимо организовать связь, и его имя появится на диаграмме связей под названием таблицы. Если в связанной таблице имеется индекс, соответствующий имени и типу выбранного вами поля, Paradox автоматически свяжет их друг с другом. Если такового не окажется, Paradox выберет первый индекс, соответствующий типу выбранного поля. Естественно, вы в любой момент можете разорвать предложенную вам связь.

Список Index окна Define Link содержит все ранее определённые поддерживаемые индексы связанной таблицы. Ключ таблицы (первичный индекс) отмечается ключём. Все поля составного ключа записываются через черточку и также отмечаются. Например, составной ключ таблицы Lineitem записывается в виде *Orders#-Customer#. Вторичные индексы таблицы перечисляются после ключа

Выберите нужный индекс, и перетащите его мышью в связываемую таблицу на соответствующее место (на рис 9,6 запись как бы накладывается). Если в главной таблице используется составной ключ, для каждого его поля вы должны указать соответствующий индекс связанной таблицы.

Графическое изображение связи

После того как вы задали соответствующие друг другу поле главной таблицы и индекс связанной таблицы, нажмите OK, и Paradox вернет вас в диалоговое окно Data Model. На его панели модели данных таблицы будут изображены связанными специальной стрелкой (рис. 9.6), которая показывает направление (от главной таблицы к связанной) и тип связи. Двойная стрелка между двумя расположенными рядом таблицами обозначает многозначное отношение между ними. Одинарная стрелка – связь типа один к одному или многие к одному.

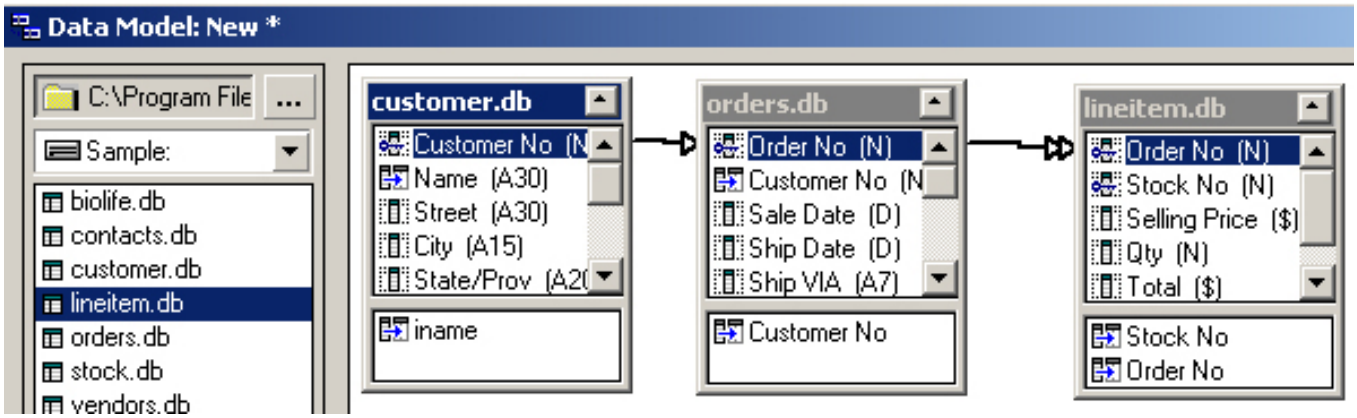


Рис. 9.6 Виды связей в модели данных

Изменение и удаление связи

Чтобы удалить связь, выберите на панели модели данных связанную таблицу и на-жмите кнопку Unlink.

Если вы хотите изменить способ связывания таблиц, произведите щелчок правой клавишей мыши над «наконечником» стрелки или выберите связанную таблицу и нажмите кнопку Link. Вы попадете в диалоговое окно Define Link, в котором кнопкой Unlink сможете разорвать существующую между таблицами связь и затем определить новую.

Связывание dBASE-таблиц

При связывании dBASE - таблиц диалоговое окно Define Link аналогично рассмот-ренному выше. В его списке Index перечисляются имена (tags) индексов связанной таблицы. dBASE-таблицы связываются только по поддерживаемым индексам. Поля, по которым осуществляется связывание (если не используются индексные выражения), должны быть схожих типов.

Рекомендуется в главной dBASE-таблице для связывания использовать уникальный индекс: для Paradox поддерживаемые уникальные индексы dBASE-таблиц абсолютно аналогичны ключам Paradox-таблиц. В этом случае ваша модель данных для dBASE-таблиц будет работать точно так же, как описанные выше модели данных Paradox-таблиц.

dBASE-таблицы можно связывать по комбинации полей, одиночному индексу из MDX-файла, а также по индексному выражению (см. таблицу 9.1).

Таблица 9.1 Комбинации связей между dBASE-таблицами

Главная таблица	Связанная таблица
Поле	Индексное выражение
Поле	Проиндексированное поле
Выражение (Master Expression)	Индексное выражение
Выражение (Master Expression)	Проиндексированное поле

Построение сложных моделей данных

Процесс ввода в модель данных новых таблиц и их связывания можно продолжать до тех пор, пока вы не получите необходимую модель (а также, пока у вас есть незадействованные индексы). На рисунке 9.7 (и рис.9.6) изображён пример модели данных для нескольких таблиц.

Это отношение установлено между полями Order No таблиц Orders и Lineitem Для таблицы Ordersr это поле является ключом, а для таблицы Lineitem вторичным индексом.

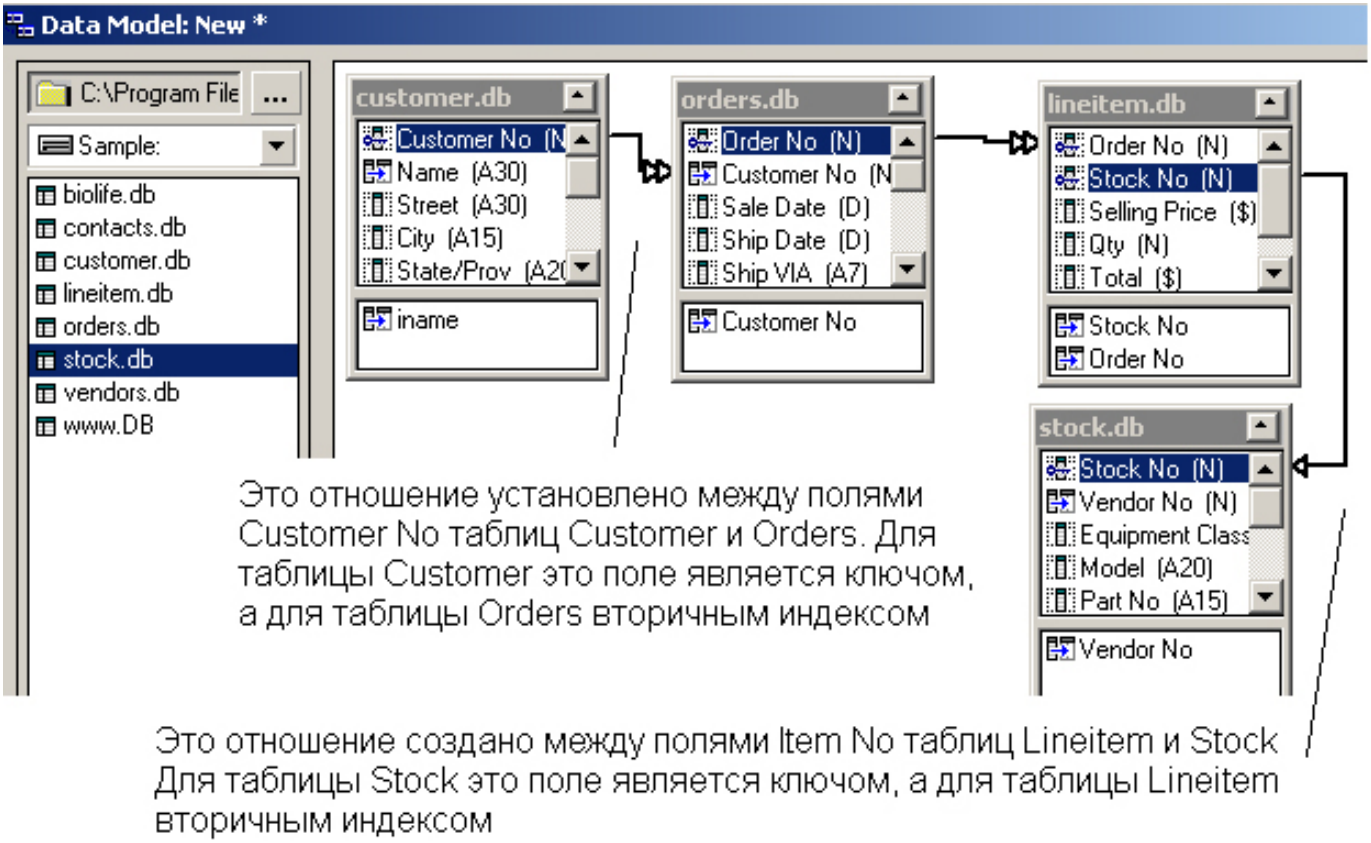


Рис. 9.7 Модель данных для нескольких таблиц

Данная модель логически связывает воедино данные из всех таблиц. В разработанной на ее основе форме (рис. 9.8) вы увидите все заказы, сделанные каждым клиентом, а также все пункты, из которых состоит каждый заказ.

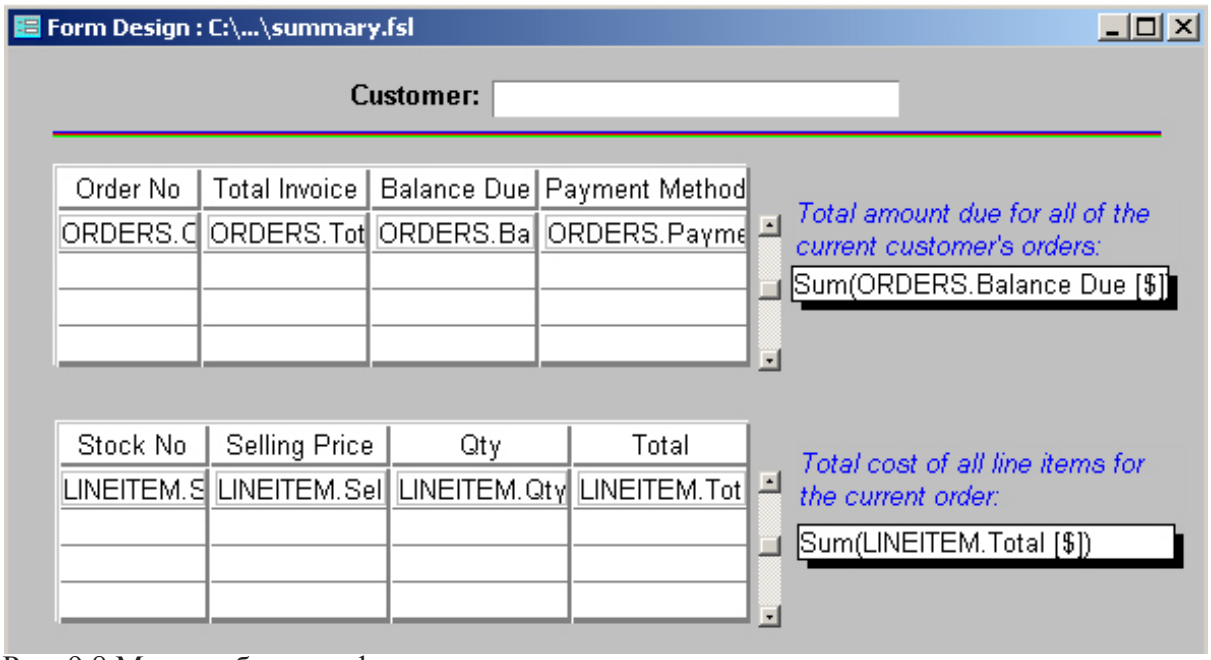


Рис. 9.8 Многотабличная форма

Обратите внимание, если между таблицами однозначное отношение, Paradox размещает их поля в одном объекте типа таблица. Такие расширенные табличные объекты существуют только в разрабатываемых документах (формах и отчетах). В имени каждого его поля присутствует название таблицы, из которого оно взято, и все производимые с его значением действия отражаются одновременно в самой таблице-источнике.

Выбор исходного чертежа документа

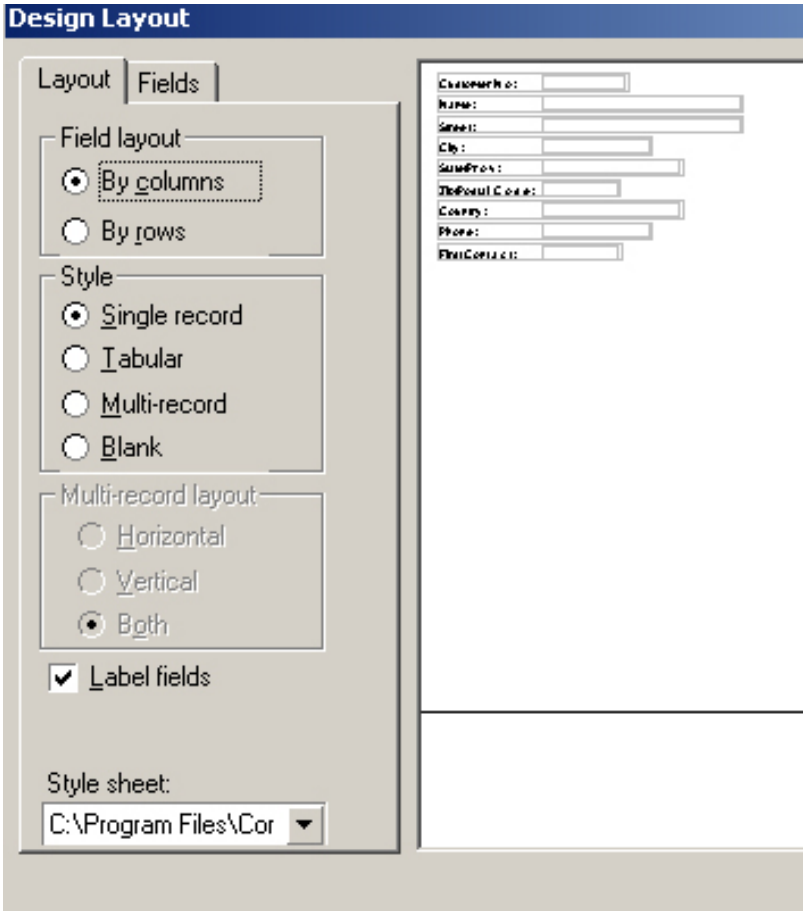


Рис. 9.9 Чертеж однотабличной формы, предлагаемый по умолчанию

Если вы в окне Data Model нажмёте кнопку OK, Paradox перенесёт вас в диалоговое окно Design Link. В просмотровом окошке изображаются поля модели данных. Например, на рисунке 9.9 показан исходный чертеж однотабличной формы, предлагаемый по умолчанию.

Окно Design Link представляет собой средство просмотра и выбора одного из типов исходного чертежа документа

Основные возможности окна Design Link

Для одно - и многотабличных документов в окне Design Link доступны различные наборы опций. Тип окна Design Link, который вы увидите, определяется разработанной вами моделью данных. Однако, независимо от типа документа вы можете:

- Выбрать поля, которые необходимо поместить в документ.
- Назначить или отменить присутствие меток полей
- Определить, для чего разрабатывается документ - для просмотра на экране либо для печати на принтере

Выбор полей

По умолчанию, когда вы начинаете разрабатывать новый документ, Paradox размещает на чертеже все поля из всех таблиц, включенных в модель данных, за исключением тех полей связанных таблиц, по которым установлена связь (чтобы не дублировать значение соответствующего поля главной таблицы).

Если вы нажмете кнопку Show Fields, на экране появится диалоговое окно Selected Fields (рис. 9.10). Оно позволяет вам решить, какую именно информацию необходимо разместить в документе. Возможно, после этого вам даже ничего не придется делать в ок-не разработки документа.

В левой части окна располагаются имена таблиц включенных в модель данных, в правой части находится список Select Fields, содержащий имена полей текущей таблицы, выбранных вами для размещения в документе.

Учтите, что в окне Select Fields доступны поля только главной и связанных с ней таблиц. Поля из несвязанных таблиц можно разместить в документе при помощи инструмента Field на SpeedBar окна разработк).

Чтобы ввести поле в список Select Fields, войдите в раскрывающийся список полей таблицы и щелкните мышью имя нужного вам поля. Для выбора одновременно нескольких полей используйте клавиши Ctrl или Shift в сочетании с мышью.

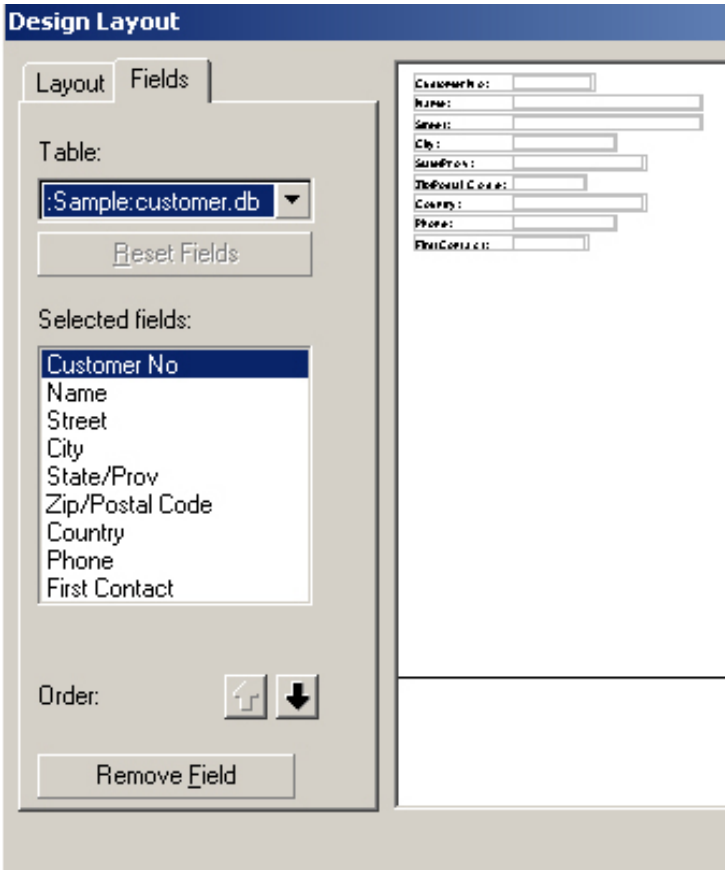


Рис. 9.10 Диалоговое окно Select Fields

Чтобы удалить поле из списка Select Fields, выберите его в списке и нажмите кнопку Remove Field. Удалённое поле снова станет доступным в раскрывающемся списке полей таблицы.

Чтобы изменить порядок следования полей в списке Select Fields (и, соответственно, по-рядок их появления в документе), выберите нужное поле и воспользуйтесь кнопками-стрелками Change Order

Имейте в виду, что все, что вы делаете в диалоговом окне Select Fields, впоследствии может быть изменено в окне разработки (в том числе, вы сможете вернуть в документ удалённые поля). Окно Select Fields предназначено для предварительных установок пара-метров чертежа документа.

Использование меток полей

По умолчанию, каждое поле в окне Design Link (как и в окне разработки) имеет метку - текстовый объект, содержащий имя поля. Чтобы удалить с чертежа документа все метки полей, используется опция Labeled Fields

Находясь в окне разработки, вы сможете проинспектировать любое поле и включить либо выключить его метку, а в окне Design Link вы просто решаете, будет ли включена либо выключена эта метка по умолчанию.

Опции Page Layout

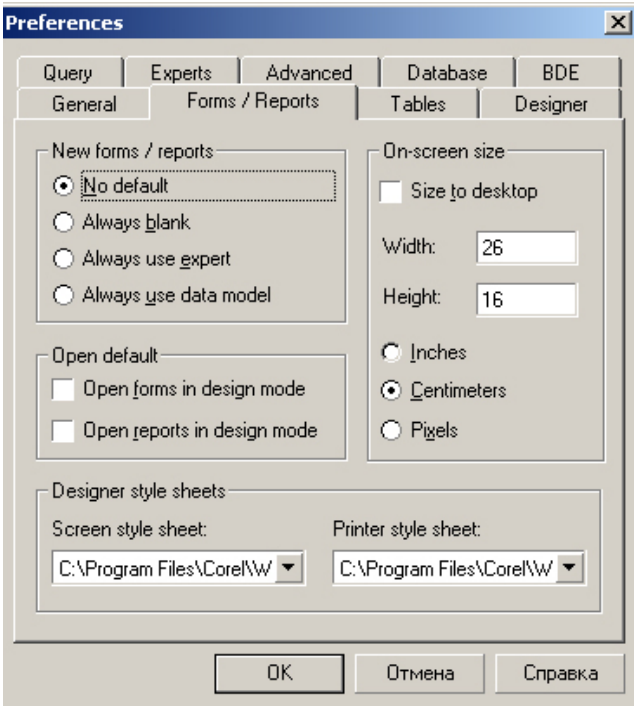


Рис 9.11 задание размеров формы

Размер окна формы задается для всех разрабатываемых форм в режиме: Tools / Settings / Preferences в режиме Forms / Reports .

Если метка стоит напротив Size to desktop, то первоначально форма будет формироваться на весь экран. Если задачу предполагается эксплуатировать с неопределенными мониторами, то эту метку надо убрать и задать размер самостоятельно. В дальнейшем его можно изменить.

Чертеж однотабличного документа

Диалоговое окно Design Link используется абсолютно одинаково как при разработке формы, так и при разработке отчёта. Единственное различие состоит в способе изображения документа.

В отчётах для отделения разных областей чертежа используются зоны (bands). Существуют зоны для заголовка и окончания (итога) всего отчёта, для верхнего и нижнего колонтитулов страницы, а также зона тела самого отчета. Все эти зоны изображаются в просмотровом поле окна Design Link. Разработанная вами модель данных документа определяет содержимое зоны записей (подробно зоны отчёта рассматриваются в Главе 12).

В формах нет зон, поэтому на чертеже документа в окне Design Link изображаются только поля таблиц, включенные в модель данных

Опции однотабличного чертежа

Панель Style диалогового окна Design Link сдержит четыре опции, соответствующие основным видам однотабличного чертежа (см. таблицу 9.2).

Таблица 9.2 Виды однотабличного чертежа

Опция	Вид чертежа
Single-Record	Одна запись таблицы
Multi- Record	Несколько записей в объекте
Tabular	В виде таблицы
Blank	Ни одной записи таблицы

По умолчанию для формы включена опция Single-Record, для отчета - опция Tabular.

Если в таблице есть длинные memo-поля, для отчета рекомендуется использовать опцию Single-Record, поскольку при формировании отчета каждая запись должна помещаться на одной странице, иначе Paradox фиксирует ошибку.

Однозаписный чертеж

При включенной опции Single-Record чертеж документа состоит из полей одной за-писи таблицы. Если вы включили опцию (рис.9,10) Single-Record или Multi-Record, панель Field Layout предоставит вам выбор способа расположения полей таблицы:

- By Columns: Paradox располагает все поля одно под другим вдоль левого края документа.
- By Rows: все поля располагаются в одну строку слева направо.

Многозаписный чертеж

Опция Multi-Record позволяет отображать одновременно несколько записей таблицы. В этом случае в окне Design Link вы увидите многозаписный объект. В его области, предназначенной для первой записи таблицы, вы должны определить расположение полей, а затем задать, сколько раз должна повториться эта область в чертеже документа по ширине и длине страницы. При этом панель Multi-Record Layout предоставляет три способа расположения повторяющихся областей:

- Horizontal: записи таблицы располагаются по ширине страницы (устанавливается по умолчанию для отчета).
- Vertical: записи располагаются по длине страницы.
- Both: записи располагаются как по ширине, так и по длине страницы (устанавливается по умолчанию для формы).

Имейте ввиду, что в окне Design Link вы задаете исходный вариант чертежа документа, который вы сможете уточнить и дополнить нужным образом, работая с чертежом в окне разработки.

Табулярный чертеж

При включенной опции Tabular изображается в виде таблицы, состоящей из тех же строк и столбцов, что и сама таблица, лежащая в основе модели данных документа

Пустой чертеж

Опция Blank панели Style служит для удаления с чертежа всех полей таблицы, на основе которой построена модель данных документа. Впоследствии, вы, конечно, сможете разместить на чертеже с помощью средств окна разработки документа.

Учтите, что выбор пустого чертежа для документа не то же самое, что разработка бланка документа. В первом случае документ уже связан с ' находящейся в модели данных таблицей, и ее поля можно разместить на чертеже документа, работая в окне разработки. В случае разработки бланка (в модели данных нет ни одной таблицы) на чертеже документа можно разместить только специальные поля и разработанные вами объекты. (Конечно, в любой момент вы можете нажать кнопку Data Model на SpeedBar, вернуться в диалоговое окно Data Model и ввести в пустую модель данных любую таблицу).

Чертеж многотабличного документа

Диалоговое окно Design Link разработки многотабличного документа аналогично однотабличному документу, где в исходном состоянии показана основная таблица. Если вас не устраивают предлагаемые способы расположения полей, вы можете включить опцию Blank, а затем, используя средства окон разработки Form Design или Report Design, разместить на чистом чертеже документа все необходимые поля в нужном вам порядке.

Перед тем как выбрать тип исходного чертежа документа, рекомендуется воспользоваться кнопкой Select Fields и выбрать те поля, которые необходимо включить в документ. Это будет особенно полезно при разработке многотабличного документа, т.к. может существенно сократить количество объектов, изображаемых в поле просмотра чертежа диалогового окна Design Link. Как и в случае однотабличных документов, диалоговое окно Design Link многотабличных документов различается для форм и отчётов. Но, в любом случае, порядок разработки документа один и тот же.

Чтобы записи связанной таблицы изображались в виде многозаписных объектов, не-обходимо включить опцию Record на панели Detail Table Style

Если на панели Number of Master Records включить опцию Many, Paradox будет ото-бражать одновременно несколько записей главной таблицы. Как они будут расположены в форме зависит от опции, установленной на панели Detail Table Style, и от того, включена ли опция Nested на панели Object Layout. В отчете записи главной таблицы изображаются в виде многозаписного объекта, а записи связанной таблицы могут располагаться в виде таблицы либо в виде многозаписного объекта (в обоих случаях они вложены в многозаписный объект главной записи).

Помните, что невложенный способ изображения записей связанной таблицы (опция Nested отключена) доступен только в формах.

Учтите, что в отчетах каждая запись (а также все относящиеся к ней записи связан-ной таблицы в случае многотабличного документа) должна размещаться в пределах одной страницы. Позаботиться об этом вы должны заранее -при разработке отчёта. Например, вы можете ограничить размер или количество связанных записей либо изменить («проинвертировать») модель данных, если в вашем документе много уровней вложения (модель данных типа 1-M-M.

Вполне возможно, что в окне Design Link вам окажется достаточно установить количество записей главной таблицы и задать способ изображения связанной. Но кроме этого, у вас имеется возможность более детальной настройки чертежа вашего документа.

В многотабличном документе вы можете задать расположение объектов (полей, таблиц и многозаписных) либо в виде столбцов (сверху вниз) либо в строку (слева направо). На всех предыдущих рисунках объекты располагались в виде столбцов (этот способ устанавливается по умолчанию). Если вы выключите опцию Fields Before Tables, связанная таблица будет располагаться перед записью главной таблицы

Вложение связанных записей

Если ваша модель данных типа 1->M и вы установили на панели Number of Master Records опцию Number, на панели Object Layout становится доступной опция Nested. Если она включена, Paradox помещает объект, содержащий записи связанной таблицы, «внутрь» многозаписного объекта главной таблицы

Если модель данных вашего документа типа 1-M-M, опция Nested доступна в любом случае, независимо от установок панели Number of Master Records.

Если вы разрабатываете отчет и включаете опцию Many на панели Number of Master Records (либо ваша модель данных типа 1-M-M), Paradox автоматически «вкладывает» связанные записи в

объект, содержащий запись главной таблицы (в диалоговом окне De-sign Link многотабличного отчета опция Nested вообще отсутствует).

При работе со связью между таблицами типа 1-M-M Paradox требует, чтобы обе связанные таблицы (в данном случае Orders и Lineitem) изображались внутри повторяющейся области многозаписного объекта (либо в виде таблицы либо, в свою очередь, в виде многозаписного объекта). Во всех случаях таблицу первого уровня вложения (в данном случае Orders) Paradox выводит в виде многозаписного объекта. Позднее, работая в окнах Form Design и Report Design, вы сможете переопределить способ ее изображения, например, на табличный

Широкий спектр чертежей, доступных в окне Design Link, является исходным материалом для разработки вашего документа. Дальнейшая его детализация производится в окнах Form Design и Report Design.

Возврат в окно Design Link

Из окна разработки документа вы можете вернуться в диалоговое окно Design Link командой Design Design Link. Если после этого, находясь в нем, вы сделаете какие-либо изменения, Paradox предупредит вас, что документ, над которым вы перед этим работали, будет заменён на новый исходный чертеж.

Учтите, что в окне Design Link вы увидите только те поля таблиц, которые на данный момент присутствуют в разрабатываемом документе. Естественно, вы можете изменить их набор.

Вернувшись в окно Design Link, вы, как обычно, можете открыть диалоговое окно Page Layout, но вместо этого его рекомендуется вызывать непосредственно из окна разработки командой Form (Page Layout (или Report) Page Layout).

Разработка документов на основе запросов

Вы можете разрабатывать формы и отчеты для запросов, так что Paradox сразу после выполнения запроса оформляет его результаты в виде документа. Например, вы можете разработать еженедельный отчет по заказам, сделанным вашими клиентами. Такой отчет создается непосредственно на основе сохраненного запроса:

Разработайте и сохраните запрос по таблице Orders, который извлекает из базы данных заказы, сделанные на текущей неделе (см. Главу 6).

1. Дайте команду File / New Report в окне Data Model из раскрывающегося списка Type выберите <Queries>. В списке File Name укажите имя сохранённого вами запроса и разработайте отчет на основе полей, включённых в таблицу Answer.
2. В списке Type вы снова можете выбрать пункт <Tables> и дополнить модель данных другими таблицами. В результате, таблица Answer станет главной таблицей документа, а остальные - либо связанными, либо останутся несвязанными.
3. Доработайте документ в окне Design Report.
4. Сохраните отчет.

Каждый раз при формировании отчёта (либо для печати, либо для предварительного просмотра на экране) Paradox сначала будет выполнять запрос. Таким образом, вы сможете получать текущую информацию, не повторяя каждый раз одну и ту же работу.

Свойства объекта в форме

Все элементы отчета являются контейнерами со своими свойствами. Для вызова свойств необходимо в режиме редактирования левой кнопкой выделить объект и правой вызвать меню свойств

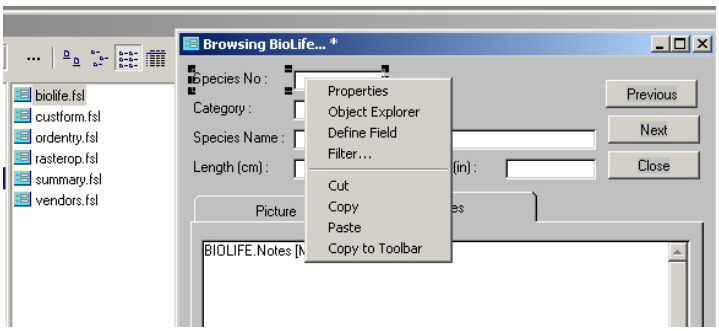


Рис.9.12 вызов меню свойств

Глава 10. Средства и приёмы разработки документов

В данной главе рассматриваются средства, использующиеся в окнах разработки документов Form Design и Report Design. В обоих окнах процесс создания документа состоит из одних и тех же операций: выбора объектов, инспектирования, использования инструментария, расположенного на SpeedBar, и широкого круга доступных приемов оформления чертежа документа.

Выбор объектов

Для выбора на чертеже документа объектов с целью проведения с ними определенных действий (перемещение, изменение размера, редактирование и т.п.) служит специальный инструмент SpeedBar Selection Arrow.

Когда вы в окне разработки выбираете какой-либо объект (щелкаете его мышью), вокруг него появляется рамка с темными квадратами - «ручками» (рис. 10.1).

Рис. 10.1 выделение объекта

Если курсор мыши находится над «точкой», он изменяет свой вид на стрелку, показывающую направление изменения размера объекта. Вы можете нажать клавишу мыши («схватить точку») и, отбуксировав сторону или вершину рамки, задать новый размер объекта. На рис. 10.1 показаны варианты буксировки точки.

Порядок выбора объектов

Вы можете задать порядок выбора вложенных объектов. Предположим, на чертеже вашего документа есть эллипс, помещенный внутрь прямоугольника. Какой объект будет выбран, если вы щелкнете мышью эллипс - прямоугольник или эллипс? По умолчанию Paradox первым выбирает внешний объект (даже если вы щелкаете мышью внутренний). Поэтому, чтобы выбрать эллипс, вам придется щелкнуть его еще раз. Аналогично, если у вас внутри эллипса находится поле, а сам эллипс содержится в прямоугольнике, и вы хотите выбрать поле, то после первого щелчка окажется выбранным прямоугольник, после второго - эллипс, и лишь после третьего - поле.

При желании, вы можете установить опцию Select From Inside в диалоговом окне Designer Properties, которое вызывается командой Properties / Designer. При этом будет первым выбираться тот объект, который вы щелкнули.

Если вы выбрали объект, содержащийся в каком - либо другом, то нажатие клавиши Esc приведет к выбору внешнего объекта.

Выбор нескольких объектов

Вы можете выбрать одновременно несколько объектов (это называется множественным выбором). Способ выбора зависит от взаимного расположения объектов:

- Чтобы выбрать изолированные друг от друга объекты (непересекающиеся), щелкните каждый из них мышью при нажатой клавише Shift
- Если необходимо выбрать несколько расположенных рядом объектов, следует нажать Shift и левую клавишу мыши и, перемещая мышь, изобразить вокруг объектов рамку. Все объекты, попавшие в нее, будут выбраны.
- Командой Edit / Select / All можно выбрать все объекты, находящиеся внутри текущего объекта. Если вы дадите эту команду, не выбрав предварительно ни одного объекта, Paradox выберет все объекты вашего документа.

Инспектирование объектов

Каждый объект вашего документа имеет свое меню свойств, определяющих его внешний вид (и, иногда, поведение). Меню объекта можно вызвать, проинспектировав его. На рисунке 10.2 показано меню объекта типа поле-объект.

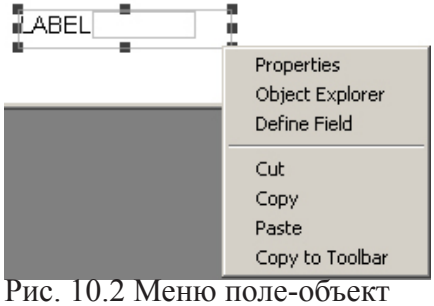


Рис. 10.2 Меню поле-объект

Некоторые свойства объектов задаются командами языка ObjectPAL и в данной книге не рассматриваются.

Если после множественного выбора объектов вы проинспектируете один из них, то внесенные в его свойства изменения будут

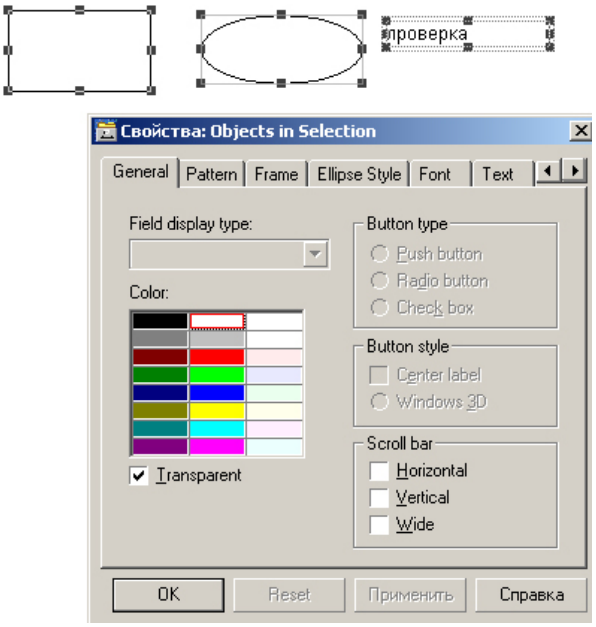


Рис. 10.3 Инспектирование одного из нескольких выделенных объектов

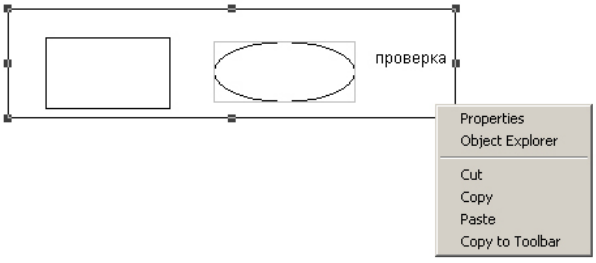


Рис. 10.4 Меню свойств контейнера

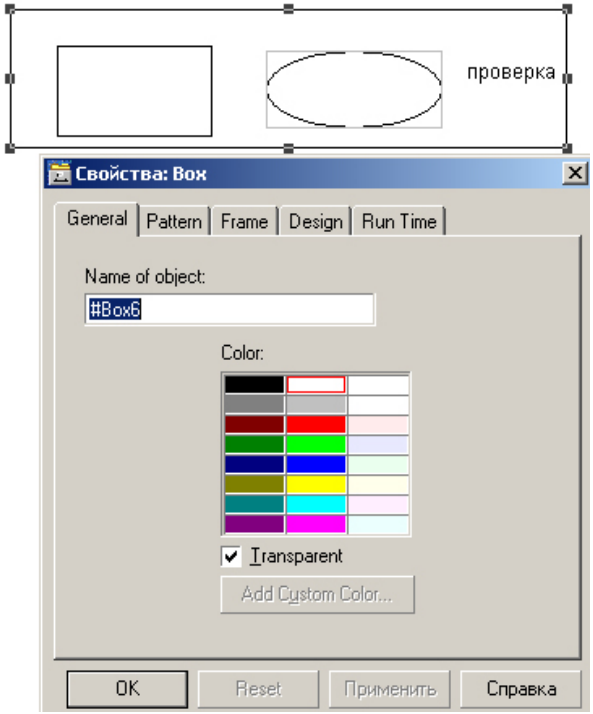


Рис. 10.5 Меню «проникающих» свойств контейнера

сделаны и в остальных выбранных объектах. На рисунке 10.3 приведен пример инспектирования одного из нескольких выделенных объектов.

Вы также можете вызвать меню «проникающих» свойств - общий список свойств всех выделенных объектов и всех объектов, вложенных в выделенные. Данное меню вызывается щелчком правой клавиши мыши при нажатой клавише Ctrl. Если для инспектирования вы используете клавиатуру, необходимо нажать комбинацию Shift+F6.

В заголовке меню «проникающих» свойств стоит фраза «Objects in ..». Оно содержит все свойства всех выбранных объектов. Некоторые свойства могут быть применимы ко всем объектам, некоторые - только к одному из них. Каждый пункт меню определяет какое-либо одно свойство всех выбранных объектов, в меню которых оно присутствует, а также соответствующее свойство всех объектов, вложенных в выбранный.

Если несколько объектов содержатся в другом объекте (например, прямоугольник, эллипс и текст заключены внутрь большого прямоугольника, который выполняет роль «контейнера»), вы можете проинспектировать как сам «контейнер», так и вложенные объекты.

В первом случае выберите «контейнер» и проинспектируйте его. Меню, вызванное вами, будет содержать только свойства самого «контейнера» (рис. 10.4). Во втором случае следует вызвать меню «проникающих» свойств (рис. 10.5). Оно содержит свойства как самого «контейнера», так и всех вложенных в него объектов.

Если вам необходимо проинспектировать всю форму или отчет, сначала убедитесь в отсутствии выделенных объектов, а затем проинспектируйте страницу формы или зону отчета. В случае формы, в зависимости от вызванного меню свойств (обычных или «проникающих») вы сможете воздействовать либо только на страницу (фон) формы, либо на все объекты, расположенные на ней (в том числе, конечно, и на саму страницу). Аналогично, в случае отчета вы изменяете свойства либо только самой зоны отчета (фона), либо зоны вместе со всеми ее объектами.

Ниже, в разделе «Плавающая палитра свойств», излагается еще один способ изменения свойств группы объектов.

Палитра свойств

Некоторые свойства Paradox изображает в виде палитр. Палитра не называет свойство, а показывает его. Поэтому в виде палитр, обычно, представляются визуальные свойства объектов (например, цвет и штриховка).

По умолчанию палитры в Paradox являются «одноразовыми» инструментами: после того, как

вы в ней выберете какое-либо свойство, она исчезает.

Палитра цветов

В меню большинства объектов(General, Pattern, Frame). присутствует пункт Color Выбрав его, вы вызываете на экран палитру Color (рис. 10.6).

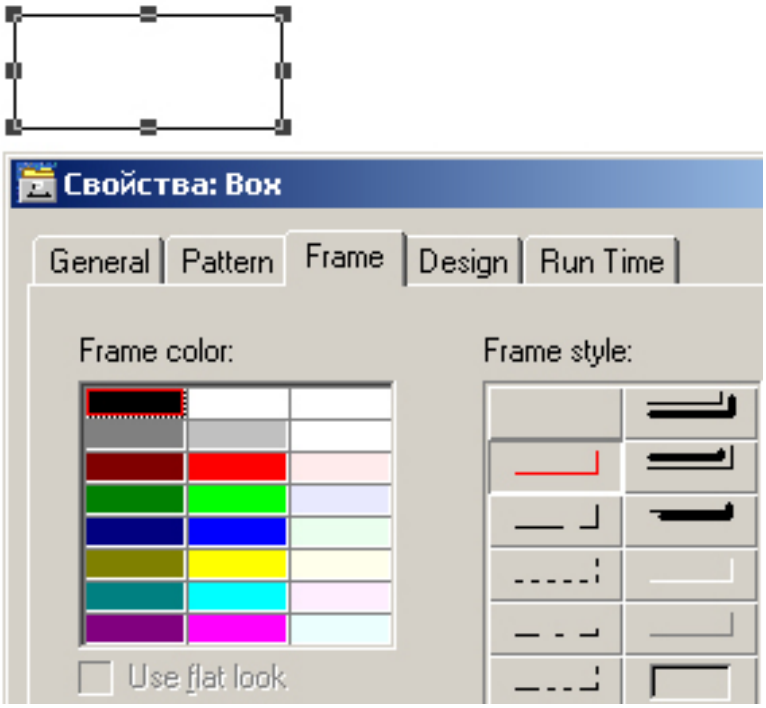


Рис. 10.6 палитра цветов

Средства и приемы разработки документов

Если вы щелкнете мышью один из ее цветов (либо выберите его при помощи клавиш управления курсором и нажмёте Enter), Paradox изменит цвет инспектируемого объекта на выбранный вами и удалит палитру с экрана. Все плавающие палитры внешне и функционально отличаются от «одноразовых» (рис. 10.7). При работе с плавающей палитрой при помощи клавиатуры инспектируемый объект изменяет свои свойства синхронно с вашим перемещением по опциям палитры. Для перемещения по функциональным областям палитры используется клавиша Tab.

Где Add Custom Color – кнопка создания собственных цветов. Крайний правый столбец на рис. 10.7 и состоит именно из новых личных цветов.

Оба типа палитры Color - «одноразовая» и плавающая -позволяют делать выбираемый цвет прозрачным, но результаты получаются различными:

- Если вы выбрали прозрачный цвет из «одноразовой» палитры, Paradox сделает инспектируемый объект прозрачным и бесцветным, т.е. будут видны все цвета лежащих под ним объектов, в том числе и фон страницы.
- Если вы воспользовались плавающей палитрой, инспектируемый объект станет прозрачным, но цветным: все лежащие под ним объекты будут видны, словно через цветное стекло (происходит смешивание цветов).

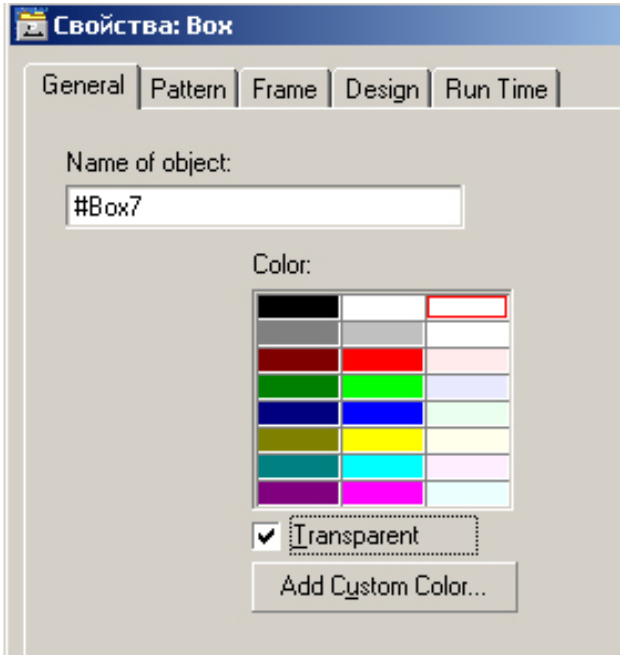


Рис. 10.7 Плавающая палитра Color Прозрачные цвета

Разработка цветов

Плавающая палитра позволяет создавать свои собственные цвета. Кнопка Custom Color становится доступной, когда вы выбираете в палитре один из пустых (неопределенных) цветов (но не белый). При нажатии ее мышью появляется диалоговое окно Custom Color (рис. 10.8).

RGB - красный, зеленый, синий
HSV - контраст, насыщенность, яркость
CMY - бирюзовый, розовый, желтый

После того как вы смешаете краски в нужных пропорциях и нажмёте OK, новый цвет появится в палитре Color.

Разработанные вами цвета Paradox сохраняет в файле PDOXWIN.INI. Поэтому однажды созданный цвет вы можете использовать во всех разрабатываемых документах.

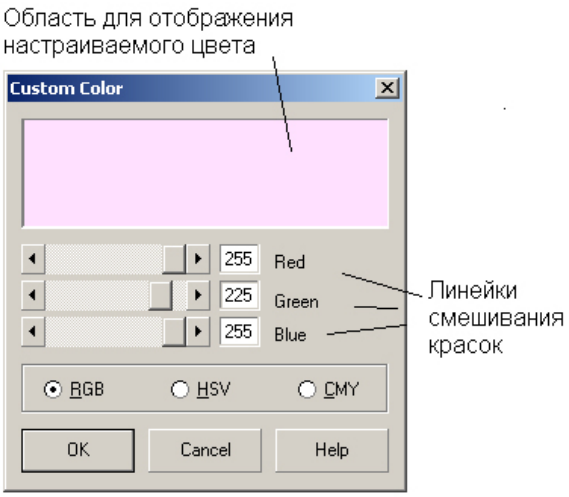


Рис 10.8 диалоговое окно Add Custom Color

Палитра рамок

Многие, объекты изображаются внутри рамки и, соответственно, в меню своих свойств имеют пункт Frame, определяющий цвет, стиль и толщину рамки.

Имейте в виду, что текстовые объекты по умолчанию изображаются без рамки. Поэтому сначала следует устанавливать стиль рамки, и только после этого изменения ее цвета и толщины будут приводить к видимым эффектам.

Пункт меню Frame / Color служит для вызова палитры Color, описанной выше. С ее помощью вы можете изменить цвет рамки инспектируемого объекта. Чтобы изменить толщину рамки, необходимо вызвать палитру Thickness с помощью пункта меню Frame / Thickness.

Пункт меню Frame / Style вызывает палитру Frame (рис. 10.9). Некоторые из стилей рамки могут быть недоступны: есть стили, использующие минимальную толщину линий рамки (см.раздел «Палитра толщины линий»).

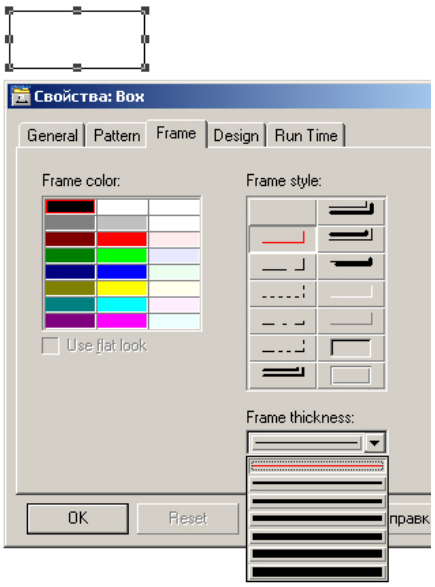


Рис. 10.9 Палитра Frame

Палитра штриховок

В меню объектов также часто присутствует пункт Pattern. С его помощью можно изменять цвет и стиль' штриховки нужного объекта.

Если вы используете пункт меню Pattern / Color, палитра Color относится к рисунку штриховки (линиям или точкам). Чтобы изменить цвет фона штриховки, воспользуйтесь пунктом Color меню самого объекта. Пункт Pattern / Style вызывает палитру Pattern (рис. 10.10).

Если окажется, что выбор нового стиля штриховки не приводит к видимому эффекту, удостоверьтесь, что цвет рисунка объекта и цвет его фона различны.

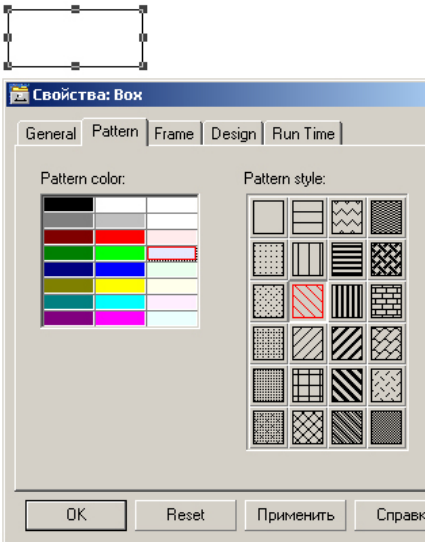


Рис.10.10 Палитра штриховок

Палитра стилей линий

Некоторые объекты Paradox в своих меню могут имеют несколько пунктов Line. С их помощью задаются цвет, стиль, тип и толщина линий. Вы также можете на концах линий, нарисованных инструментом Line, изобразить стрелки. Пункт меню Line Style вызывает палитру Line (рис. 10.9).

Учтите, что количество доступных стилей зависит от установленной толщины линий

Палитра толщины линий

Линии и рамки имеют свойство Thickness, определяющее их толщину. Если вы в диалоговом окне Page Layout укажете, что данный документ предназначен для принтера, а не для экрана, то при инспектировании объекта вместо палитры Thickness вы будете получать меню веса линии (например, Hairline (тончайшая), 1pt, 2pt и т.д.)

Палитра шрифтов

В меню объектов, которые могут содержать текст (текстовые объекты, поля, таблицы и т.п.), присутствует пункт Font. С его помощью можно по отдельности изменять все параметры изображения текстовых символов - начертание, размер, стиль и цвет. С помощью плавающей палитры шрифтов (рис. 10.11) можно варьировать все параметры шрифта одновременно.

Учтите, что выбор начертаний символов зависит от того, какие шрифты установлены в вашей системе Windows.

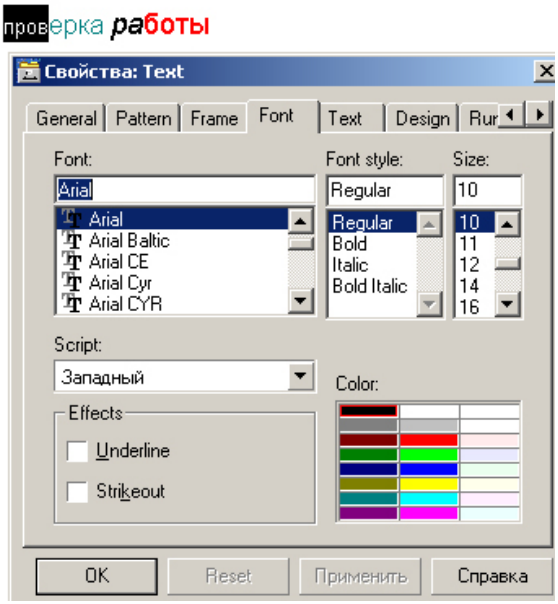


Рис. 10.11 палитра шрифтов

Присвоение объектам имен

Предположим, вы размещаете в форме поле. Поскольку оно состоит из трех частей (само поле, его метка и область редактирования), каждая из них будет иметь свой собственный номер (рис. 10.12). Обычно, собственные имена присваиваются объектом для того, чтобы проще было обращаться к ним из методов, написанных на языке ObjectPAL. Если в вашем документе много различных объектов, проще запомнить их имена, а не номера, которые присваивает Paradox

Разработка новых объектов

Чтобы разместить в документе новый объект, щелкните мышью соответствующий инструмент, затем мышью задайте положение и размер объекта. Сразу после размещения объекта автоматически включается инструмент Selection Arrow, служащий для выбора объектов. Если вам необходимо разместить несколько однотипных объектов, соответствующий инструмент следует щелкнуть при нажатой клавише Shift (он останется активным все время, пока вы не отпустите Shift).

Учтите, что объект в документе размещаются только мышью, эквивалентные приемы работы с клавиатурой отсутствуют.

Инструментарий SpeedBar

На SpeedBar окон разработки находится ряд кнопок-иконок, функции которых несколько отличаются от работы большинства кнопок в Paradox. Они называются инструментами и служат для размещения в документе объектов. Рисунок 10.13 демонстрирует инструменты и типы объектов которые они создают.

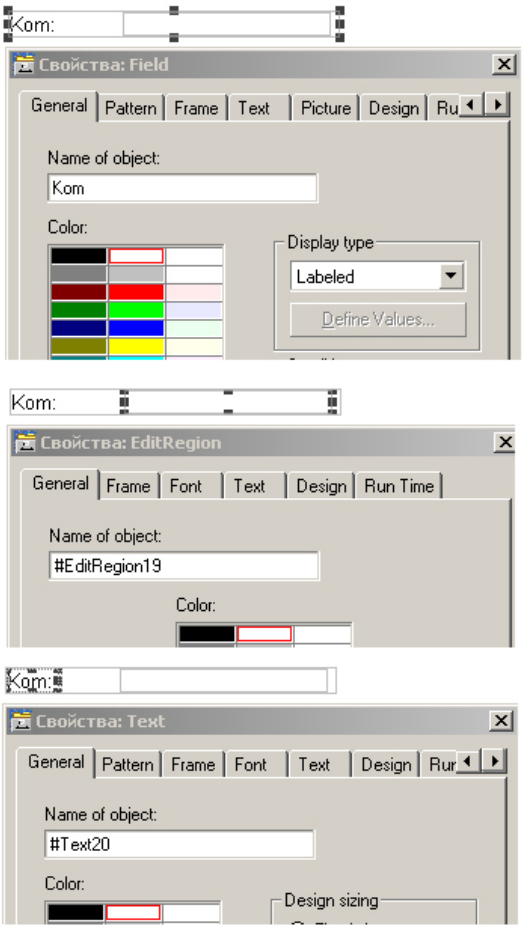


Рис. 10.12 Инспектирование различных частей поле - объекта

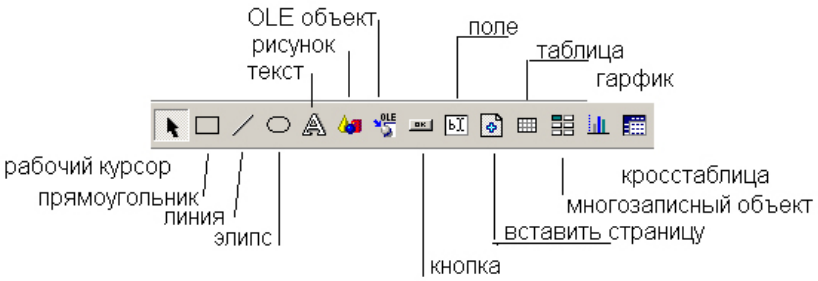


Рис. 10.13 Инструменты SpeedBar

В данном разделе рассказывается о создании и (при необходимости) об определении объектов. Свойства большинства объектов различаются в зависимости от того, что вы разрабатываете - форму или отчет, и рассматриваются отдельно в Главах 11 и 12. Кнопки используются только в формах (они описаны в Главе 11). Графики и кросстаблицы рассматриваются в Главе 13.

На рис. 10.13 изображены инструменты SpeedBar при создании форм. Из всего набора необходимо отметить объект «Кнопка». Причина - большой набор методов, описывающих событие нажатия. Т.е. при установке кнопки необходимо выбрать нужный из представленных наборов. Эти наборы разбиты на две группы. Первая определяет тип действия (перемещение, печать, закрытие или открытие других форм), а вторая группа летализирует это действие (перемешение вверх или вниз, на одну запись или группу, печать отчета или экранной формы и т.д.)

Несмотря на очень большой выбор, тексты этих методов открытые и в случае необходимости их можно корректировать. Например, при удалении записи предупреждающий текст можно заменить на русский и тогда простым операторам будет понятнее.

Размещение линий, прямоугольников и эллипсов

Для размещения в документе простейших графических объектов служат три инструмента - Box, Ellipse и Line. Щелкните инструмент, затем в нужном месте документа нажмите левую клавишу мыши и, удерживая клавишу нажатой, перемещайте мышь до придания объекту не обходимой формы и размера. Отпустив клавишу мыши, вы получите готовый объект.

Кроме прямых линий Paradox предоставляет возможность рисовать кривые (см. следующий пример).

Пример. Рисование кривой линии

1. Проинспектируйте прямую линию и выберите пункт меню Line / Type / Curved.
2. Отбуксируйте конечную точку линии. Форма изгиба линии зависит от того, в каком направлении была нарисована линия и за какую конечную точку вы ее буксируете.
3. Происперементируйте с буксировкой конечных точек, чтобы получить нужный результат.
4. Чтобы снова сделать линию прямой, проинспектируйте ее и выберите пункт меню Line Type / Straight

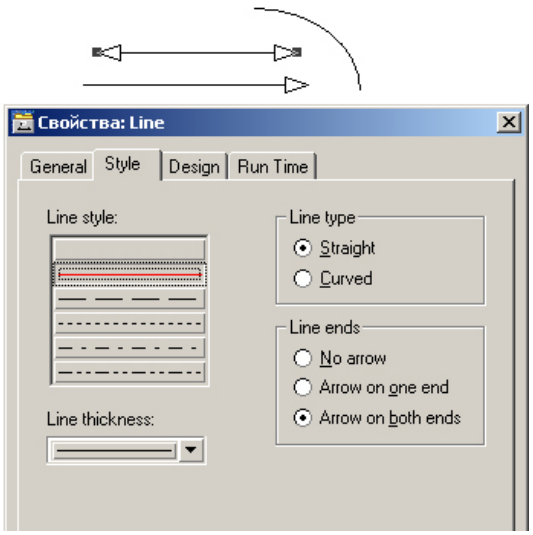


Рис. 10.14 рисование кривых и стрелок

Paradox также позволяет разместить на концах линии стрелки. Проинспектируйте линию, выберите пункт Line Ends и, затем, один из вариантов (рис. 10.14):

- No Arrow: на данной линии стрелки не изображать (эта опция включена по умолчанию).
- On One End: разместить стрелку в конечной точке линии (она будет показывать направление, в котором перемещался курсор мыши, когда рисовали линию).
- On Both Ends: разместить стрелки на обоих концах линии.

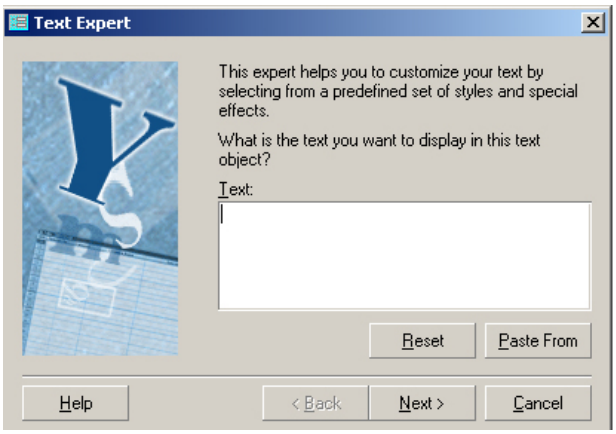
Размещение текста

С помощью инструмента Text в документе создается текстовый объект. Сам текст размещается внутри рамки текстового объекта. При этом размер символов может изменяться в зависимости от размеров рамки. Чтобы задать способ размещения текста внутри рамки, необходимо проинспектировать

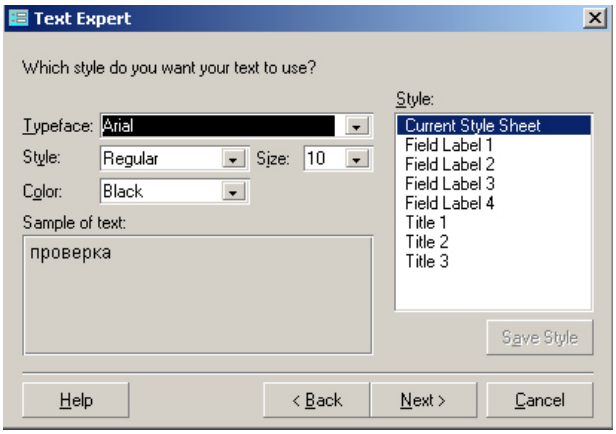
текстовый объект и выбрать пункт меню Design Sizing.

Текстовые объекты переменных размеров

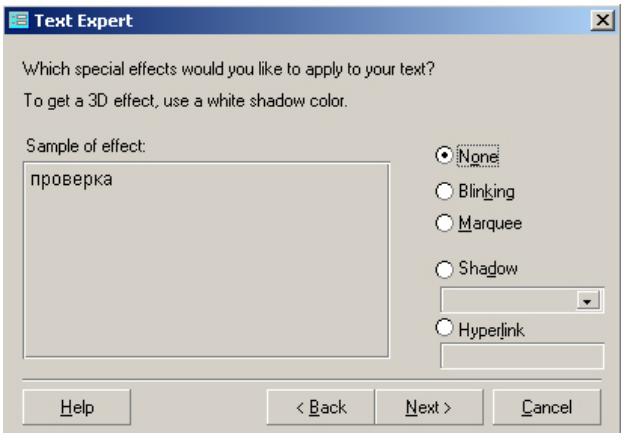
Чтобы получить текстовый объект, размер которого определяется объемом текста, щелкните инструмент Text, а затем выделите область, где будет расположен текст. После этого открывается ряд окон Text Expert (рис. 10.15)



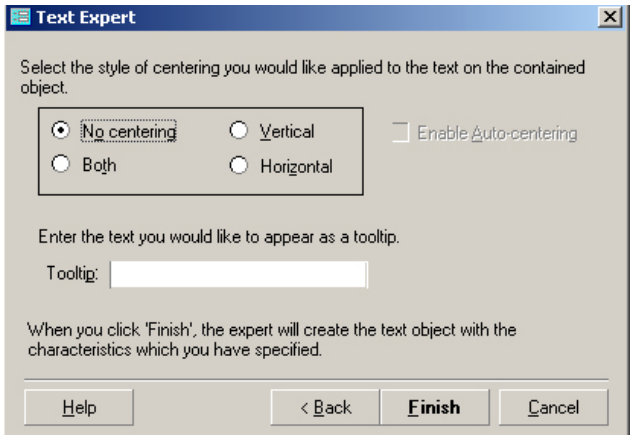
1-е окно



2-е окно



3-е окно



4-е окно

Рис.10.15 этапы ввода текста

Свойство Word Wrap такого объекта, отвечающее за перенос слов со строки на сроку, включается автоматически, когда вы нажимаете Enter. После этого опция Word Wrap в меню объекта становится недоступной.

Учтите, что при включенной опции Word Wrap такой текстовый объект можно изменять только по ширине. Если опция Word Wrap выключена, размер объекта изменять нельзя вообще (в этом случае, сначала проинспектируйте его и включите свойство Design Sizing / Fixed Size).

Размеры текстового объекта изменяются буксировкой сторон его рамки

Размещение графических изображений

Для размещения в документе графических изображений служит инструмент Graphic.

С его помощью можно импортировать графику из Windows Clipboard и файлов форматов .BMP, .PCX, TIP, .GIF, EPS, JPG

- Щелкните мышью инструмент Graphic и нарисуйте в документе рамку нужного размера. Внутри графического объекта появится надпись «Undefined Graphic». Проинспектируйте его и выберите пункт меню Define Graphic, затем:
- Выберите Paste, если вы хотите вставить графическое изображение из Windows Clipboard. (Если Clipboard пуст, данная опция недоступна).

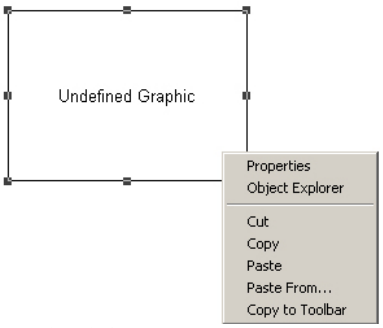


Рис. 10.16 Диалоговое окно Paste From Graphic File Масштабирование изображений

- Выберите Paste From, чтобы импортировать изображение из файла.

На экране появится диалоговое окно Paste From Graphic File (рис. 10.16).

При вставке изображения в документ Paradox изменяет его размеры до заданной вами рамки и включает его опцию Design / Size To Fit. Эту опцию следует выключить, прежде чем вы сможете изменять размеры объекта.

Внутри рамки изображение можно перемещать, буксируя его мышью. Учтите, чтобы перемещать изображение по документу, следует выбрать рамку - его «контейнер».

Свойство графического объекта Magnification позволяет пропорционально увеличивать и уменьшать его до 25%, 50%, 100%, 200% или 400% от его исходного размера (рис. 10.17).

Опция Best Fit пропорционально изменяет размеры изображения так, чтобы оно заняло все пространство внутри заданной вами рамки.

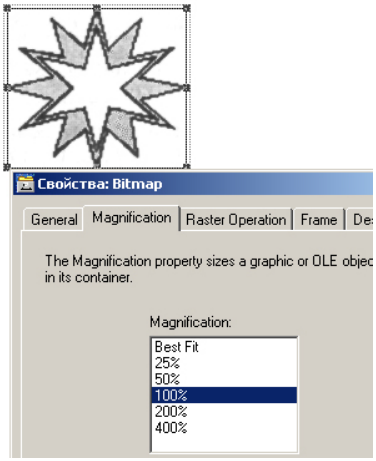


Рис. 10.17 Пропорциональные изменения графических объектов

Растровые операции

В большинстве случаев при определении в документе графического объекта вы просто хотите увидеть на экране точную копию какого-либо графического изображения. Однако, иногда бывает необходимо, чтобы графический объект и экран взаимодействовали друг с другом. Например, объект можно сделать прозрачным или проинвертировать его цвета. Эти операции называются растровыми. Для операций такого рода в меню свойств графических объектов служит пункт Raster Operation.

Растровая операция определяет, как оригинальное графическое изображение накладывается на экран инвертированием, сложением, включением или исключением цветов. По вашему выбору Paradox применит логические операции AND, OR или XOP к каждой точке изображения.

Таблица 10.1 Растровые операции

Операция	Результат
Source Copy	Оригинал копируется без изменений
Source Paint	Оригинал копируется логической операцией OR
Source And	Оригинал копируется логической операцией And
Source Invert	При копировании производится логическая операция XOR
Source Erase	Экран инвертируется, при копировании оригинала производится логическая операция AND
Not Source Copy	Оригинал инвертируется, затем копируется без изменений
Not Source Erase	При копировании производится операция OR
Merge Paint	Оригинал инвертируется, затем копируете операцией OR

Использование маски

Предположим, вы размещаете графический объект в форме с цветным фоном страницы. Если фон изображения-оригинала отличается от фона формы, вы получите совершенно ненужную в документе границу импортированной области.

Чтобы избежать этой ситуации, следует использовать маску, делающую часть рисунка-оригинала прозрачной.

Пример. Использование графической маски

Предположим, на странице формы зеленого цвета необходимо разместить графический объект, по форме отличающийся от прямоугольного. Пока фон контейнера объекта не будет зеленым, его границы видны (рис. 10.18).

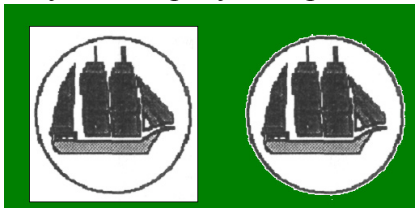


Рис. 10.18 Использование маски

1. Сделайте копию изображения оригинала. Назовите его

MASK.BMP.

2. Слева (рис.10.18) графический объект не использует маску, поэтому видна прямоугольная граница вокруг него
3. Справа графический объект использует маску С помощью графического редактора измените MASK. BMP так, чтобы те его части, которые должны быть прозрачными, стали черными, а все остальное - белым (рис. 10.19).
4. Разместите рамку графического объекта в окне Form Design, проинспектируйте ее и выберите пункт меню Define Graphic / Paste From. В диалоговом окне Paste From Graphic File укажите имя файла MASK. BMP.
5. Проинспектируйте графический объект и выберите пункт Raster Operation (Source Paint (рис. 10.20)
6. Разместите в форме (на любом свободном месте) изображениеоригинал.
7. Проинспектируйте его и выберите пункт Raster / Operation / Source And (рис. 10.21).
8. Выберите оба объекта одновременно (клавиша Shift плюс щелчок мышью).
9. Проинспектируйте один из объектов и выберите пункт Frame / Style и удалите рамки обоих объектов.
10. Выберите пункт Design / Align / Align Left, затем Design / Align / Align Top и, наконец - Design / Group.
11. Когда оригинал наложится на маску, его прямоугольная область станет прозрачной и будет иметь цвет фона страницы (рис. 10.22).

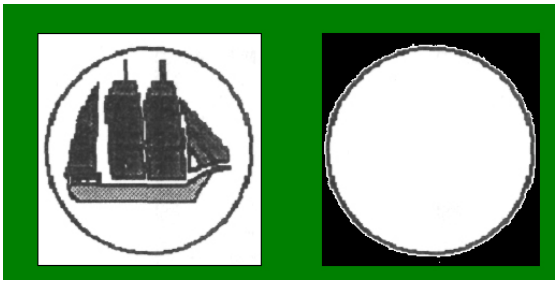


Рис. 10.19 Исходный графический объект (слева) и маска(справа)

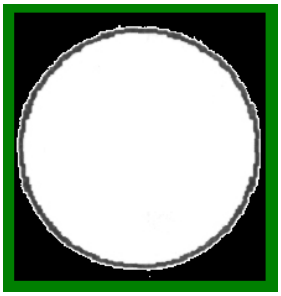


Рис. 10.20 Результат действия операции Source Paint на маску

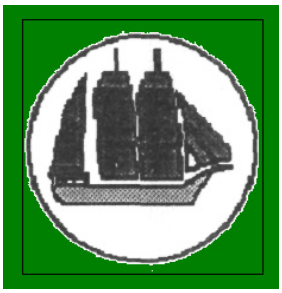


Рис. 10.21 Результат действия операции Source And на графический объект

Рис.10.22 Результат использования маски

Размещение OLE-объектов

Использование OLE-технологии позволяет Paradox иметь доступ и воспроизводить графические изображения, тексты, звук и таблицы, принадлежащие другим Windows программам, поддерживающим OLE-технологию, Они являются по отношению к Paradox серверами, а сам Paradox – клиентом. Более подробная информация объектами изложена в Главе 14.

Помните, когда вы определяете OLE-объект, Paradox подгоняет его размеры под заданный вами «контейнер» и включает опцию объекта Design / Size To Fit. Поэтому, прежде чем пытаться изменить размеры объекта, выключите ее.

Размещение полей

Когда вы начинаете разрабатывать новый документ и выбираете в качестве исходного любой чертеж, отличный от бланка, Paradox размещает на нем поля вашей таблицы. Для размещения в документе дополнительных полей служит инструмент Field.

По умолчанию, инструмент Field создает поля с метками. Поля - это объекты, состоящие из трех частей: в сам объект типа поле (поле-объект) вложены метка (текстовый объект) и область редактирования, в которой отображается значение поля (рис. 10.23).

Рис. 10.23 Компоненты поле-объекта

Для каждого поле - объекта необходимо определить собственно поле какой-либо таблицы, значение которого должно быть помещено в данный объект. Определение является одним из свойств поле - объекта. Проинспектируйте поле - объект и выберите пункт его меню Defined Field. Paradox предоставит меню, состоящее из имен полей. Выберите какое-либо из них, чтобы определить данный поле - объект (рис. 10.24).

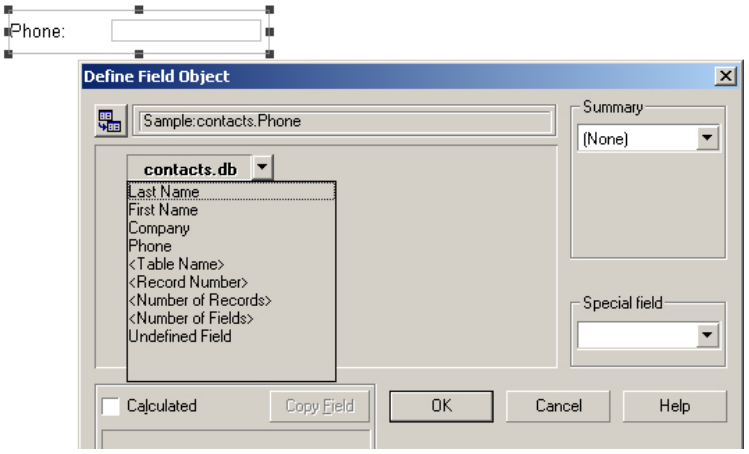


Рис. 10.24 Определение поле-объекта

Поле может быть «вычисляемым». Выбирается одно поле из таблицы, нажали на кнопку Copy Field, ввели знак необходимого арифметического действия, потом скопировали другое поле из таблиц, входящих в модель данных и т.д. Если в формуле есть ошибка, при нажатии на кнопку OK она будет указана. Поле может быть определено и как результат групповой обработки, причем результат групповой обработки может входить и в формулу (рис. 10.25).

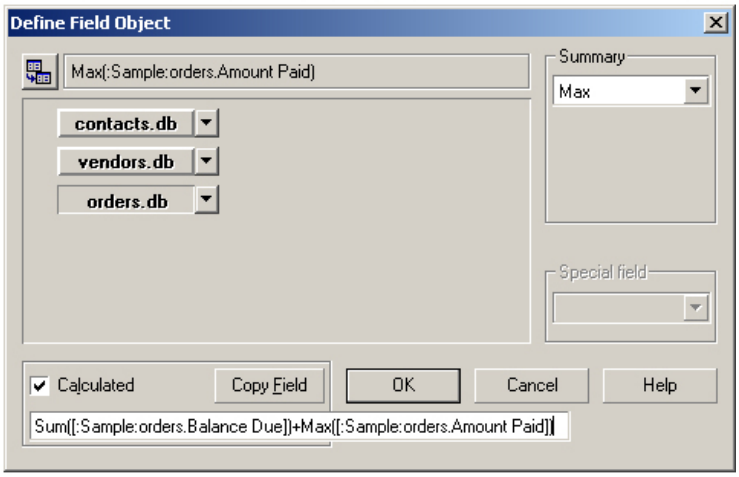


Рис. 10.25 Диалоговое окно Define Field Object

Специальные поля

Специальные поля содержат не данные из таблицы, а информацию о самой таблице или разрабатываемом документе.

Чтобы выбрать специальное поле, относящееся к таблице (ее имя, номер текущей записи, количество записей или номера полей), в диалоговом окне Defined Field Object войдите в раскрывающийся список, прикрепленный к названию таблицы. Специальные поля находятся в меню после полей таблицы и заключены в угловые скобки <>

Специальные поля, относящиеся к документу (сегодняшняя дата - Today, текущее время - Now, номер страницы - Page Number, полное количество страниц -Numbers of Page), находятся в раскрывающемся списке Special Field (рис. 10.26).

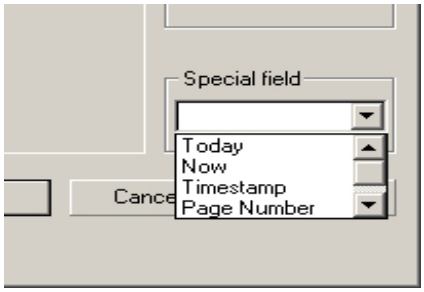


Рис. 10.26 Специальные поля

Размещение таблиц

Если в диалоговом окне Design Layout вы выбрали исходным чертеж в виде таблицы (см. Главу 9), то в ваш документ будет помещена табличная сетка, содержащая все поля вашей таблицы (рис. 10.27).

По умолчанию сетка соответствует структуре таблицы. Она является сложным объектом, в ее состав входят:

- Поле-объекты - поля исходной таблицы
- Текстовые объекты - метки полей
- Столбцы - вертикальные колонки полей (их можно удалять и вставлять)

- Строки - горизонтальные ряды полей (их можно инспектировать)
- Заголовки - ряд меток полей

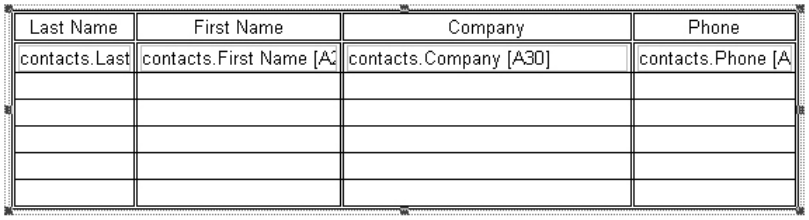


Рис. 10.27 табличная сетка в окне Form Design

Новая табличная сетка

Если в диалоговом окне Design Layout вы выбрали в качестве исходного чертежа бланк (пустой чертеж), или вам необходимо поместить в документ дополнительную таблицу, воспользуйтесь инструментом Table. Новая табличная сетка может быть связанной либо независимой - это определяется моделью данных документа (см. Главу 9).

После размещения новой сетки необходимо определить, какие поля должны в ней отображаться. Это можно сделать тремя способами:

- Проинспектировать каждое поле и выбрать пункт Defined Field.
- Проинспектировать запись и выбрать пункт Define Record (далее вы выберете таблицу, поля которой будут определять все поля сетки).
- Проинспектировать сетку и выбрать пункт Define Table: Paradox предложит выбрать таблицу, поля которой будут определять все пола сетки.

На рисунке 10.28 показаны области табличной сетки, где надо произвести щелчок правой клавишей мыши, чтобы проинспектировать перечисленные объекты.

Если вы инспектируете запись или сетку, вам предоставляется список всех таблиц, находящихся в модели данных. Когда вы выберете одну из них, она заменит собой все, что было до этого определено в табличной сетке (включая установки свойств и фрагменты на ObjectPAL).

Чтобы изменить определение табличной сетки, проинспектируйте ее или ее запись и выберите в меню многоточие (...). Появится диалоговое окно Define Table Object (рис. 10.29), в котором вы сможете:

- Выбрать поля, которые следует поместить в табличную сетку
- Изменить порядок следования полей

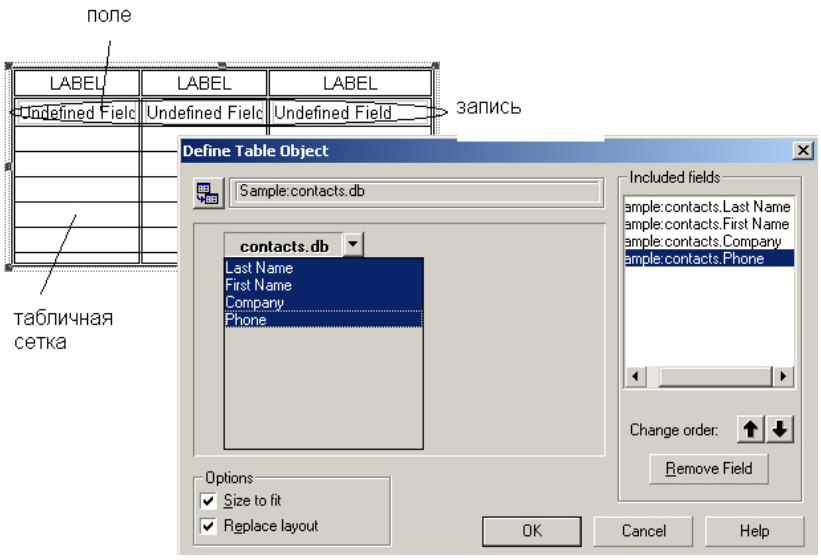


Рис. 10.28 Области табличной сетки

- Ввести в сетку дополнительные поля. Если в окне Define Table Object щелкнуть имя какой-либо таблицы, оно появится рядом с кнопкой Data Model, а в списке Included Files будет находиться фраза <No fields included>.

Если при этом вы нажмете ОК, то для вашей сетки будет определена данная таблица, но все ее поле-объекты останутся неопределенными (вам придется инспектировать и определять их по отдельности).

Чтобы задать для поле-объектов поля данной таблицы, войдите в раскрывающийся список, присоединенный к имени таблицы, и выберите в нем все необходимые поля. Их имена будут помещены в список Included Files (рис. 10.32).

Кроме выбора полей таблиц и установки порядка их следования в табличной сетке документа, в диалоговом окне Define Table Object вы также можете:

- Включить опцию Size To Fit, чтобы табличная сетка автоматически растянулась или сжалась по ширине в соответствии с суммарной длиной определенных для нее полей (в противном случае сетка останется тех же размеров, какие вы ей задали при размещении в документе).
- Включить опцию Replace Layout, чтобы все ранее определенные в табличной сетке поля были заменены на выбранные вами в окне Define Table Object. По умолчанию Paradox добавляет новые поля к уже существующим в сетке, даже если они не определены. Кроме того, при включенной опции Replace Layout между новыми полями и ранее определенными могут возникать конфликты (последние в этом случае становятся неопределенными).
- Можно добавить и потом определить поля, которые первоначально были опущены. Для чего надо проинспектировать таблицу и в свойствах выбрать Insert Column

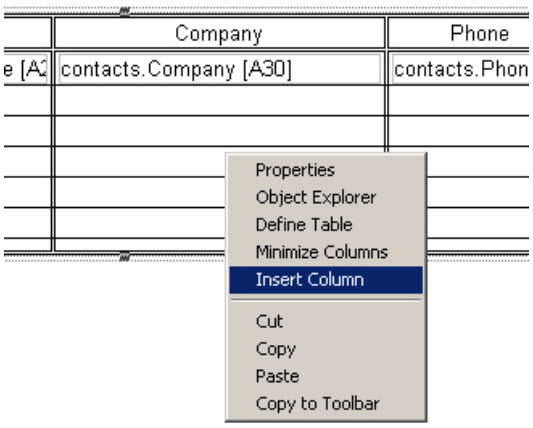


Рис. 10.29 добавление поля

Размещение многозаписных объектов

Многозаписный объект отображает одновременно несколько записей таблицы в виде повторяющихся определенное количество раз по вертикали и горизонтали областей. Поля внутри областей располагаются в любом заданном вами порядке: вы определяете способ представления одной записи и указываете, сколько таких записей следует разместить на странице документа.

Многозаписный тип чертежа документа можно задать в диалоговом окне Design Layout, когда вы разрабатываете новую форму или отчет. Очень часто, многозаписные объекты используются для печати почтовых наклеек: каждая наклейка представляет собой группу полей (имя, адрес, город, почтовый индекс), повторяющуюся в каждой записи. На рисунках 10.30 и 10.31 приведен пример многозаписного чертежа отчета и сам отчет по таблице Customer.

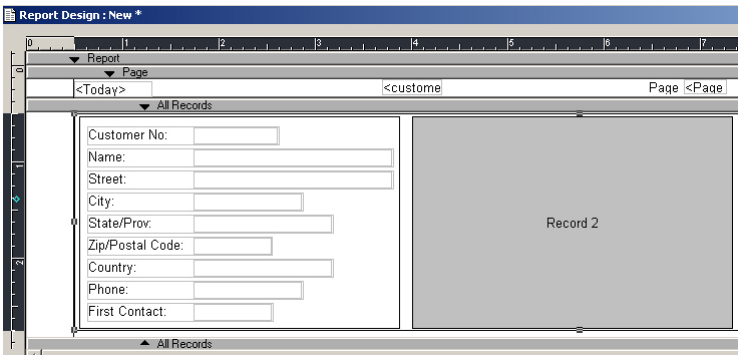


Рис. 10.30 Многозаписный чертеж отчета

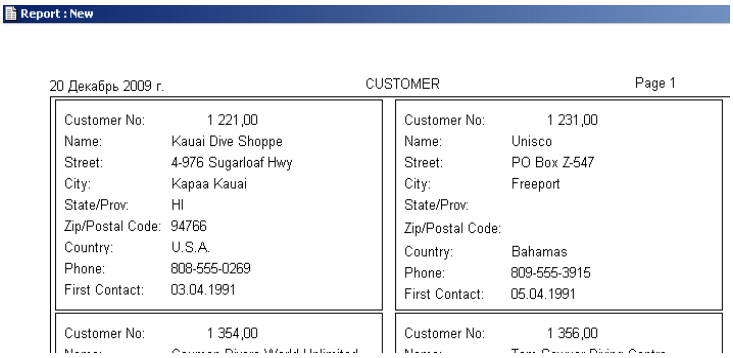


Рис. 10.31 Многозаписный отчет

Чтобы разместить в документе новый многозаписный объект, щелкните на SpeedBar инструмент Multi-Record и при помощи мыши задайте размер и форму объекта (рис. 10.32).

Если необходимо изменить размер записей объекта, выберите первую запись и мышью отбуксируйте одну из «ручек». Paradox соответствующим образом изменит размеры остальных записей.

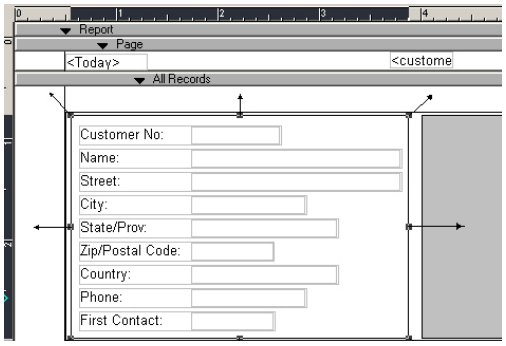


Рис. 10.32 Изменение размеров многозаписного объекта

Определение многозаписного объекта

Чтобы определить поля, которые должны быть отображены в многозаписном объекте, проинспектируйте его и выберите пункт меню Define Record. Вы увидете список всех таблиц, включенных в модель данных. Выберите нужную таблицу, и Paradox поместит ее поля в первую запись объекта. Вы также можете взять любую другую таблицу, выбрав многоточие (...) и открыв диалоговое окно Define Multi-Record Object. Это окно выглядит и работает абсолютно аналогично диалоговому окну Define Table Object (рис. 10.25).

Вы также можете, не задавая таблицы для многозаписного объекта, разместить с помощью инструмента

Field в его первой записи поля, а затем определить их, проинспектировав каждое поле по отдельности.

Определение структуры многозаписного объекта

Проинспектируйте объект и выберите пункт Record Layout. Появится диалоговое окно Record Layout (рис. 10.33), в котором вы сможете задать:

- Количество записей по длине и ширине страницы документа
- Расстояние между записями по вертикали и горизонтали (расстояние задается в единицах, определенных в диалоговом окне Grid Settings, рассмотренном в разделе «Использование сетки» ниже в данной главе)
- Порядок следования записей - сверху вниз, затем слева направо, если включить опцию Top-Down, Then Left-Right, либо слева направо, затем сверху вниз, если включить опцию Left-Right, Then Top-Down.

Учтите, что количество повторений записей по вертикали может зависеть от свойств многозаписного объекта Run Time / Show All Records и Delete When Empty (СМ. Главу 12).

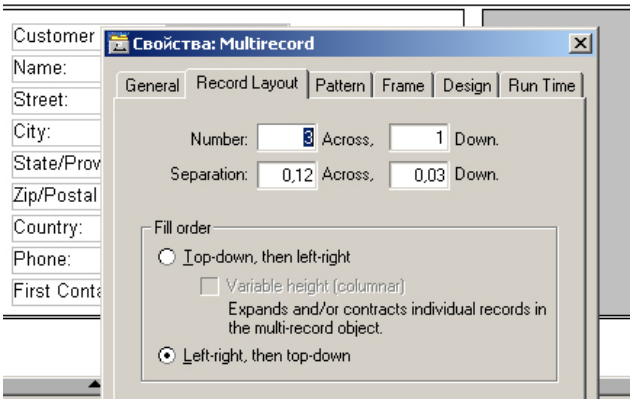


Рис. 10.33 Диалоговое окно Record Layout

Design-свойства объектов

В меню всех объектов присутствует пункт Design. Когда вы, инспектируя объект, выбираете этот пункт, вам предоставляется меню свойств, применимых к объекту только в окне разработки документа. Эти свойства помогают вам работать с объектами в окнах Form Design и Report Design.

В зависимости от типа объекта набор Design-свойств различен. Например, в меню линии нет пункта Contain Objects, поскольку линия не может содержать внутри себя другие объекты. Другие объекты (таблицы) всегда являются «контейнерами», и вы не можете выключить его опцию Contain Objects.

В таблице 10.2 приведены наборы Design-свойств каждого типа объекта.

* Объекты типа Button и Crosstab используются только в окне Form Design.

Таблица 10.2 Design-свойства

	Design-свойство			
Объект	Pin Horizontal	Pin Vertical	Contain Objects	Size To Fit
Box	+	+	+	
Line	+	+		
Ellipse	+	+	+	
Text	+	+	+	
Graphic	+	+	+	+
OLE	+	+	+	+
Button*	+	+	+	
Field	+	+		+
Table	+	+		+
Multi-Record	+	+		
Graph	+	+	+	
Crosstab*	+	+		

Вложение объектов

Если один объект находится полностью внутри другого, он может быть вложен во внешний объект. Вложенные объекты перемещаются вместе со своим «контейнером» и уничтожаются, когда вы уничтожаете «контейнер».

У всех объектов, имеющих свойство Contain Objects, данная опция по умолчанию включена. Это означает, что вам достаточно разместить один объект внутри другого, и он уже станет вложенным. Существует несколько способов разместить один объект внутри другого:

- Создать новый объект в пределах границ существующего
- Отбуксировать существующий объект внутрь другого
- Отбуксировать или изменить размеры «контейнера» так, чтобы он «захватил» полностью другой объект
- Вставить объект из Windows Clipboard внутрь существующего

При включенной опции Snap To Grid могут возникнуть сложности при вложении объектов друг в друга, поскольку оба объекта могут «пытаться» привязаться к одной и той же линии сетки окна разработки. В этом случае следует изменить их размеры относительно друг друга или выключить опцию Snap To Grid.

Чтобы объект перестал быть вложенным, вы можете:

- Проинспектировать его «контейнер» и выключить опцию Design / Contain Objects
- Отбуксировать его за пределы «контейнера» (достаточно, чтобы его часть была вынесена за пределы «контейнера»).

При определенных обстоятельствах некоторые объекты нельзя «вынуть» из их «контейнера». Например, нельзя из объекта типа поле с меткой «вынуть» ни метку, ни саму область редактирования поля, поскольку данный объект по определению включает в себя эти две части.

При удалении объектов, связанных вложением, помните, что удаление вложенного объекта никак не затрагивает «контейнер».

Фиксация положения объекта

В окне разработки любой объект можно отбуксировать в любое место документа. Чтобы зафиксировать объект на своей позиции, можно воспользоваться свойствами Design\Pin Horizontal и Design]Pin Vertical. Фиксация объекта особенно удобна, когда его необходимо переместить только в вертикальном направлении или только в горизонтальном.

При фиксации объекта следует помнить следующее:

- Объект фиксируется относительно своего «контейнере», т.е. «контейнер» по-прежнему можно перемещать по документу.
- Если после перемещения или изменения размеров какого-либо объекта внутри него оказывается зафиксированный объект, последний не становится вложенным.
- Paradox может перемещать фиксированные объекты, например, при использовании команды Design (Align: фиксация предохраняет объект только от случайных перемещений мышью.

Свойства Pin Horizontal и Pin Vertical имеются у всех без исключения объектов в окнах Form Design и Report Design. Комбинируя различным образом данные опции, вы можете зафиксировать на чертеже документа любой объект нужным вам образом.

Кроме фиксации объектов в окнах разработки документа, вы можете использовать ее при формировании (для печати или предварительного просмотра) отчёта (см. раздел «Run Time-фиксация объектов» ниже в данной главе).

Подгонка размеров объекта

Поля, таблицы, графические изображения и OLE-объекты имеют в своих меню пункт Design] Size To Fit. Еел1% вы включаете данное свойство, объект автоматически
подгоняет свои размеры под заданную вами границу - рамку объекта.

Предположим, вы создали поле-объект небольших размеров, а затем определили его полем Customer No. Если опция Size To Fit включена, метка поля и область редактирования, а вместе с ними и сам поле-объект автоматически примут соответствующие размеры, чтобы вместить необходимую информацию. Если вы переопределите объект полем Qty, его размер автоматически уменьшится.
Действие опции Size To Fit несколько различается для разных типов объектов:

- Таблицы: Если размер таблицы выходит за пределы допустимого, у нее появляется горизонтальная линейка прокрутки, а опция Size To Fit включается. Если вы вручную изменяете размеры таблицы при включенной опции Size To Fit, Paradox изменяет ширину столбцов таблицы.
- Поля: Опция Size To Fit остается включенной, даже если есть проблемы, связанные с размером поле-объекта. Вы можете изменить его вручную, но Paradox автоматически установит его заново, как только вы измените какое-либо из свойств данного поле-объекта.
- Графические и OLE-объекты: Опция Size To Fit остается включенной, даже если есть проблемы, связанные с размером объекта. Вы не можете вручную изменять размеры графических и OLE-объектов при включенной опции Size To Fit.

Run Time - свойства объектов

В меню всех объектов присутствует пункт Run Time, содержащий свойства объектов, которые проявляются при просмотре или печати документа.

Набор Run Time-свойств зависит от типа объекта и типа документа. На рис 10.34 показаны допустимые для каждого типа объекта в режиме Run Time и Design - свойства.

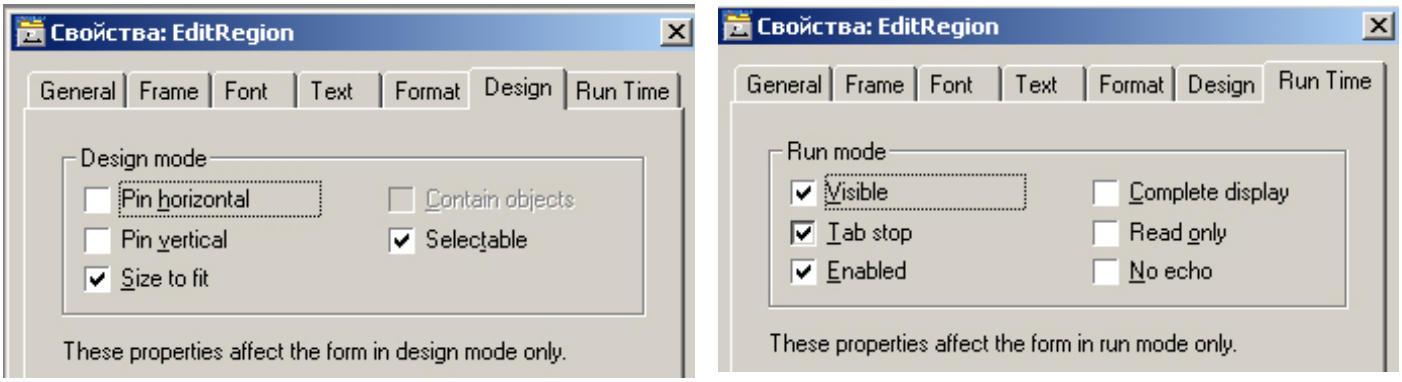


Рис 10.34 свойства в режиме Run Time и Design

Невидимые объекты

Все объекты в окне Form Design имеют в своих меню пункт Run Time / Visible. По умолчанию все объекты являются видимыми (опция Visible включена). Если выключить опцию Visible, при просмотре или печати документа Paradox скроет данный объект.

В основном, эта опция полезна для тех, кто разрабатывает приложение на ObjectPAL, в котором объекты становятся видимыми, только когда это действительно нужно.

В окне Report Design линии и прямоугольники имеют свойство Run Time / Invisible. Невидимые объекты могут служить для контроля размеров других объектов (подробности изложены в Главе 12). Если включить опцию Invisible, Paradox скроет сам объект, но оставит видимыми все содержащиеся в нем.

Run Time-фиксация объектов

Когда вы просматриваете или печатаете отчет, некоторые его объекты заполняются данными, что может вызвать изменение их размеров (способы контролирования автоматического изменения размеров объектов изложены в Главе 12). Изменившийся в размере объект может вызвать сдвиг остальных объектов.

Рис. 10.35 подгонка размера по умолчанию

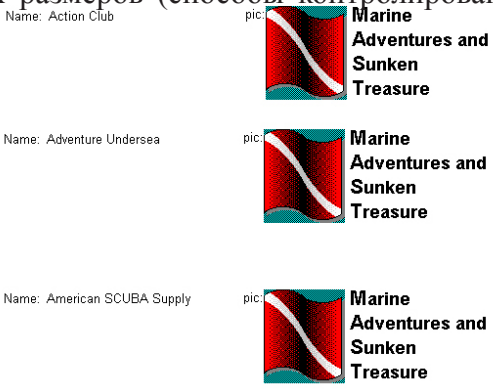


Рис. 10.36 Графический объект зафиксирован в горизонтальном направлении

Например, вы поместили в отчет поле Name таблицы Customer. Пока вы работаете в окне Report Design, данный поле-объект будет иметь постоянный размер. При запуске отчета для про-

смотра или печати поле-объект по умолчанию подгоняет свои размер под объем содержащихся в нем данных (рис. 10.35).

Если вы проинспектируете графический объект и включите опцию Run Time / Pin Horizontal, он будет предохранен от нежелательных сдвигов (рис. 10.36). При этом позаботьтесь о том, чтобы между объектами осталось достаточно места, иначе они могут наложиться друг на друга.

Информация по работе с поле-объектами, таблицами и многозаписными объектами при создании отчетов подробно изложена в Главе 12.

Присоединение к объектам методов

ObjectPAL - это язык разработки приложений в среде СУБД Paradox. Методами называются процедуры, написанные на ObjectPAL, присоединяемые к объектам форм. Вы можете разработать методы, которые будут управлять данными, взаимодействовать с пользователем базы данных, выполнять определенные операции и т.д.

Все объекты формы, в том числе сама страница (фон) формы, имеют в меню свойств пункт Object Explorer. С его помощью производится присоединение к объекту методов. (Информация по данному вопросу подробно излагается в руководстве по ObjectPAL). Некоторые методы вложены в события стандартно и их можно использовать не владея языком ObjectPAL. На рис.10.37 показан вид окна Object Explorer

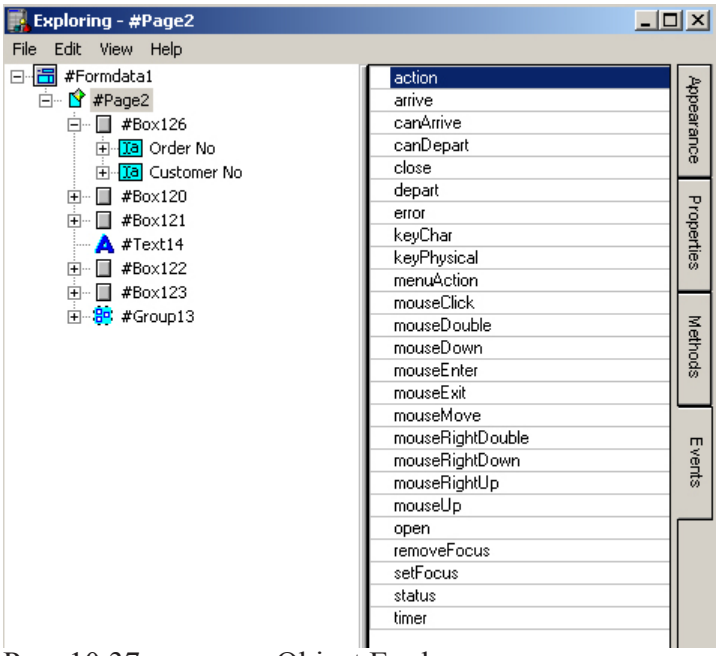


Рис. 10.37 вид окна Object Explorer

Например, если надо, чтобы при нажатии кнопки происходила печать отчета, выбираем Application code / print a report и все. Ниже приведен листинг метода, который будет содержаться в объекте (красным выделен комментарий)

```
method pushButton(var eventInfo Event)
var
        rHandle         Report ;// tmp handle to report
        rpiHandle       ReportPrintInfo         ;// tmp handle to
print info structure
endVar
        ;// Properties of the ReportPrintInfo can be viewed
using the following
        ;// line of code:
        ;//
        ;rpiHandle.view()
        ;// Set name of report.
        ;//
        rpiHandle.name = «customer.rsl»
        rHandle.open(rpiHandle.name ,winStyleHidden)     ;//display print dialog
        ;// Print the report
        ;//
        ; Print is always returning FALSE - prebug logged Dec1 /98
        ; can't put error check in until bug is fixed
        rHandle.print()
```

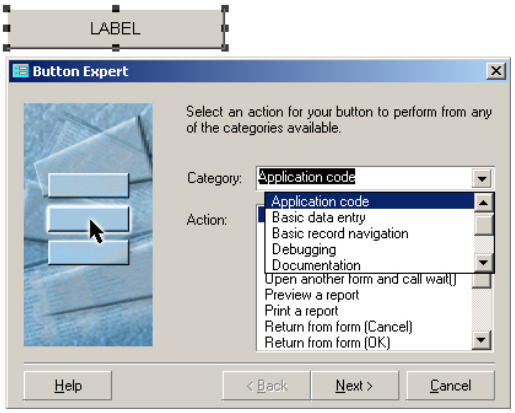


Рис.10.38 показан выбор стандартных методов для объекта «кнопка»

```
if NOT rHandle.print() then
        msgStop(«Error printing « + rpiHandle.name,
                «Please make sure the report exists and try again.»)
endIf

endMethod
```

Как видно из рис.10.38 в самом Paradox вложено довольно большое количество методов. Однако, можно писать и свои собственные. Для этого имеется встроенный язык PAL, который будет рассмотрен во второй книге.

Кнопки Speed Bar

Кроме инструментов на SpeedBar окон Form Design и fieport Design расположены несколько кнопок, служащих для быстрого вызова наиболее часто использующихся команд Paradox.

Рис.10.39 кнопка speed bar

- **Cut:** (5-я) удаляет выбранный объект (или объекты) из окна разработки и помещает его в Windows Clipboard.
- **Copy:** (6-я)копирует выбранный объект (или объекты) в Windows Clipboard без удаления оригинала из окна разработки.
- **Paste:** (7-я)вставляет содержимое Windows Clipboard в выбранный объект окна разработки.
- **View Data:** (11-я) вызывает разрабатываемый документ для просмотра с его помощью данных. Если вы работаете над формой, вы сможете отредактировать или просмотреть данные в окне Form, если над отчётом - вы получите на экране его постраничную распечатку.
- **Print:** (4-я) печатает документ.
- **Open Folder:** (8-я) вызывает окно Folder вашего рабочего каталога.
- **Buttom:** (8-я во втором ряду) установка объекта «кнопка»

И т.д.

Приемы работы в окнах разработки

Окна form Design и Report Design содержат широкий набор команд меню и средств разработки, общих для обоих окон. В данном разделе описываются приёмы их использования.

Наложение объектов

Объекты в документе могут накладываться друг на друга. Вы можете изменять порядок следования «слоев», в которых находятся объекты или группы объектов, командами Design (Bring To Front и Design) Send To Back.

Данные команды также изменяют порядок выбора объектов клавишей Tab (это относится к работе с документом только в окнах разработки).

Группы объектов

Если необходимо, чтобы несколько отдельных объектов вели себя, как один, выберите их и воспользуйтесь командой Design / Group (для отчетов).

Внутри группового объекта не может находиться объект, не являющийся членом группы. Чтобы изменить состав группового объекта, вы должны либо определить новый групповой объект либо расформировать существующий и, затем, определить новый набор объектов как группу.

В состав группы вы также можете включать уже существующие групповые объекты. При этом они становятся вложенными группами.

Создав групповой объект, вы можете инспектировать как его отдельные объекты, так и всю группу целиком, если щелкните правой клавишей мыши какую-либо точку внутри группового объекта, не принадлежащую ни одному из объектов, составляющих его. Проинспектировав групповой объект, вы, например, можете присоединить к нему методы или расформировать его.

Копирование объектов

В окнах разработки существует два способа создания копии объекта. Во-первых, вы можете это сделать с помощью Clipboard, используя пару команд Edit / Copy и Edit / Paste. Этим способом копию объекта можно разместить в любом месте документа.

Во-вторых, вы можете воспользоваться командой Edit / Duplicate, копирующей выбранный объект в один прием, минуя Clipboard. Копия размещается рядом с оригиналом.
Учтите, что копировать объекты можно только в пределах одного окна.

Использование линеек

В окнах разработки имеются горизонтальная и вертикальная линейки, которые вы можете использовать при размещении, изменении размеров и перемещении объектов. Для включения и выключения линеек выполнить Tools / Serrings / Preferences / designer (рис.10.40)

Когда вы выбираете какой-либо объект, его проекции на линейки выделяются цветом, указывая вам его размер и положение.

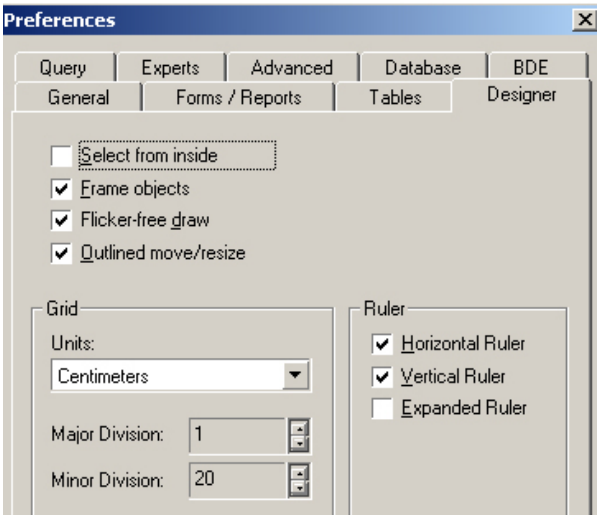


Рис.10.40 установка линеек и масштаба сетки

Установка табуляций

Чтобы установить на дополнительной линейке табулятор, включите инструмент табуляции и щелкните мышью на том месте, где он должен быть размещён.
Paradox поддерживает четыре типа табуляторов:

- Left tabs - выравнивание по левой границе текста (все символы печатаются слева от табулятора).
- Right tabs - выравнивание по правой границе текста (все символы печатаются слева от табулятора).
- Center tabs - выравнивание текста по центру, который указывается табулятором.
- Decimal tabs - числа выравниваются по десятичной точке (все символы, введенные до десятичного разделителя, располагаются слева от позиции, отмеченной табулятором, остальные - справа).

Табуляторы можно перемещать, буксируя их в пределах линейки. Удаляются они буксировкой за пределы линейки.

Установка абзаца

Чтобы параграф начинался с абзаца (красной строки), необходимо отбуксировать маркер абзаца в нужное место линейки. Он может располагаться как слева, так и справа от левой границы (отступа) текста.

Установка отступов

По умолчанию левый и правый отступы определяются границами текстового объекта. Однако, их можно изменять, буксируя соответствующие маркеры.

Выравнивание текста

Вы можете использовать четыре типа выравнивания границ текста - по левой границе, по правой, по обеим границам и по центру (рис. 10.41, 10.42). Перед вводом текста щелкните мышью иконку нужного типа выравнивания либо выделите уже введенный текст и воспользуйтесь иконкой.

В поле Tooltip можно ввести комментарий, который будет появляться при установки мыши на надпись.

рис 10.41 выравнивание введенного текста

Сетка окна разработки

Сетка представляет собой набор вертикальных и горизонтальных линий на странице документа, помогающих расположению объектов. Она включается командами View / grid. Параметры сетки задаются выбором следующих команд: Tools / Setting / Preferences / designer (рис.10.40)

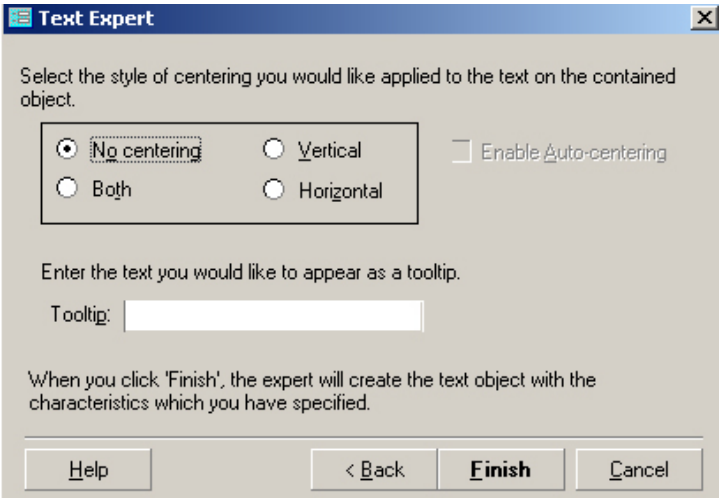


Рис 10.42 выравнивание текста при вводе

Поле системных сообщений

В нижней части Paradox Desktop располагается поле, служащее для вывода сообщений системы Parsdox. Когда вы работаете в окне разработки документа, в правой части поля видно имя выбранного объекта, а в левой – произхводимую над ним операцию.

Например, если вы перемещаете объект, поле системных сообщений показывает его имя, размер и текущие координаты (если выполнены команды View / size and position)

Команды Zoom

Если необходимо рассмотреть какую – либо часть объекта или, наоборот, увидеть его целиком, используется команда Zoom (рис. 10.43).

Как видно на рисунке, имеется три команды автоматического масштабирования:

- Fit Wight – пропорционально изменяет размер изображения, чтобы показать весь документ по ширине
- Fit Helght - пропорционально изменяет размер изображения, чтобы показать весь документ по длине
- Best Fit - пропорционально изменяет размер изображения, чтобы показать весь документ по ширине и длине

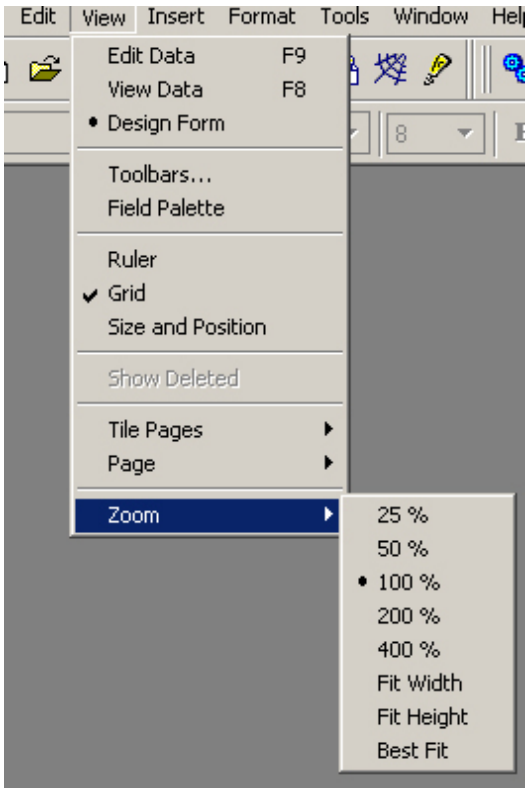


Рис . 10.43 команда Zoom

Выравнивание объектов

Paradox представляет несколько способов выравнивания выбранных объектов (рис.10.44). Для выбора нескольких объектов в форме надо пользоваться кнопкой Shift / левая кнопка мыши.

- Align Bottom: располагает объекты так, чтобы их нижние края находились на одном уровне с нижним краем самого нижнего объекта.
- Align Top: выравнивает объекты по верхнему краю верхнего объекта.
- Align Middle: располагает средние точки объектов на одной горизонтальной лини
- Align Left: располагает объекты так, чтобы их левые края находились на одной вертикали с левым краем левого объект.
- Align Right: аналогично выравнивает объекты по правому краю правого объекта.
- Align Center: располагает средние точки объектов на

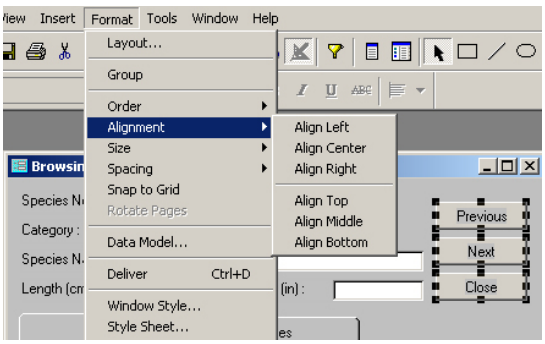


Рис . 10.44 команды выравнивания объекта

одной вертикальной линии.

Учтите, что объекты, находящиеся в разных зонах, не могут быть выравнены по вертикали. Если включена опция Snap to Grid ? то все объекты привязываются к ближайшим узлам сетки. Выравнивание никак не влияет на вложенность объектов.

Выравнивание размеров объектов

Чтобы придать некоторым объектам одинаковый размер или регулярным образом расположить несколько объектов, вы можете воспользоваться командами рис.10.45. Предположим, вам нужна группа абсолютно одинаковых кнопок. Выберите их и дайте команду Format / Size. Вы получите несколько опций для выбора объекта, который будет принят за образец для остальных:

- Minimum Width; все объекты примут размеры самого узкого из них.
- Maximum Width; все объекты примут размеры самого широкого.
- Minimum Heigth; все объекты примут размеры наименьшего по высоте среди них.
- Maximum Heigth; все объекты примут размеры наибольшего по высоте.

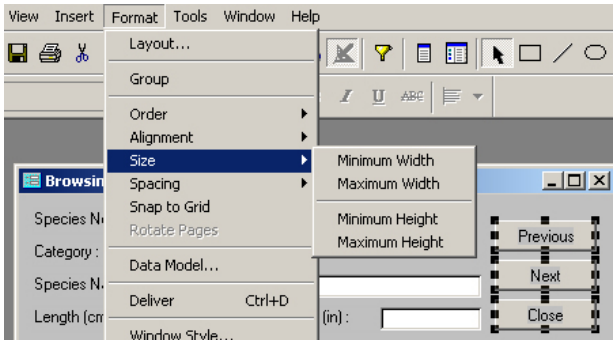


Рис. 10.45 выравнивание размеров объектов

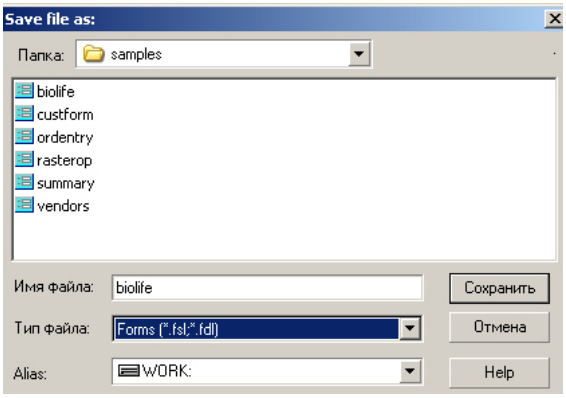


Рис. 10.46 окно сохранения документа

Если один из выбранных объектов не может изменить свои размеры, Paradox игнорирует его и изменяет размеры остальных объектов.

Сохранение документа.

Чтобы сохранить разрабатываемый документ необходимо дать команду File / save или File / save as (рис.10.46). По умолчанию документ сохраниться в рабочем каталоге. Однако все параметры (кроме расширения) можно поменять. В данном примере форма сохраниться с расширением .fsl. Для сохранения в формате .fdl (когда документ используется, но не может быть изменен) надо выполнить Format / deliver. В этом случае нельзя ничего изменить и файл будет помещен рядом с файлом, имеющим расширение .fsl. Небходимо отметить, что первыми будут выполняться файлы с расширением .fdl.

Глава 11 Разработка форм

В главе 9 было описано построение модели данных и выбор исходного чертежа для новой формы. В Главе 10 вы познакомились со средствами и приёмами, общими для окон разработки документов Form Design и Report Design.

Данная глава содержит информацию, специфическую для работы в окне Form Design. Вообще говоря, вы можете работать с формами, не пользуясь окном Form Design, поскольку существует возможность быстрого вызова форм (см. Главу 4), которые Paradox генерирует автоматически. Даже если вы разрабатываете сложную многотабличную форму, значительный объем работы выполняется в диалоговом окне Design Layout (см. Главу 10).

Назначение окна Form Design состоит в том, чтобы вы могли уточнить черте ж вашей формы, настроить ее для большего удобства работы пользователя с данными.

В окне Form Design вы можете:

- Перемещать объекты
- Добавлять и удалять такие объекты, как линии, поля, таблицы и графики
- Инспектировать и изменять свойства любого объекта формы
- Присоединять к объектам методы, написанные на ObjectPAL

Настройка формы, используемой по умолчанию

Данный раздел демонстрирует настройку формы, создаваемой Paradox автоматически, на примере таблицы Orders.

Пример. Настройка формы в окне Form Design

При быстром вызове Paradox автоматически создает однотабличную форму, в которой все поля таблицы располагаются в одну колонку вдоль левой границы экрана (рис. 11.1).

Такую форму можно получить двумя способами:

- Дайте команду File / New / Form, в окне Data Model выберите таблицу Orders, а в окне Design Layout - предлагаемый по умолчанию чертеж формы.
- Работая с таблицей Orders в окне TaBle, дайте команду Table (Quick Form, либо нажмите F7, либо щелкните мышью кнопку Quick Form на SpeedBar.

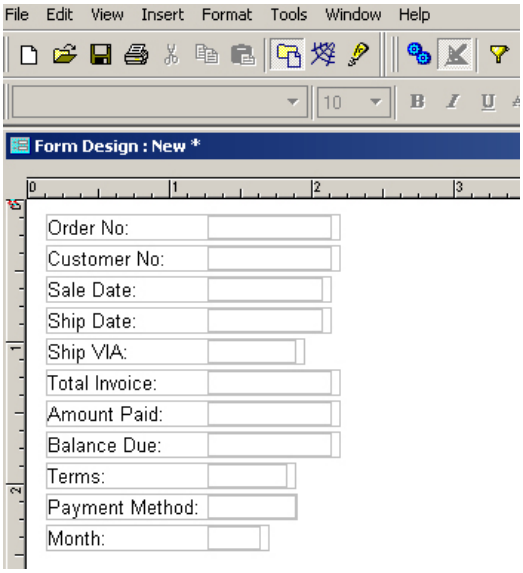


Рис. 11.1 Чертеж однотабличной формы, созданный по умолчанию

По умолчанию Paradox помещает в форму все поля таблицы. Вы можете удалить те из них, которые считаете в данной форме ненужными.

1. Удалите поле Amount Paid: выберите его и нажмите Del. Аналогичным образом удалите поле Month.
2. Буксируя поля, сгруппируйте информации (рис. 11.2).
3. Создайте текстовый объект - заголовок формы. включите инструмент Text, щелкните мышью в нужном месте экрана: Paradox поместит в эту точку текстовый курсор. Введите Order Information и щелкните мышью вне объекта, чтобы удалить текстовый курсор (рис. 11.3).
4. Проинспектируйте новый объект и установите Font / Style / Bold: текст будет выделен жирным шрифтом.
5. Методом множественного выбора выберите метки всех полей (рис. 11.4). (Нажмите клавишу Shift и, удерживая ее нажатой, щелкните мышью метки всех полей. Если опция Select From Inside выключена, вам придется каждую метку щелкнуть дважды, поскольку первым щелч-

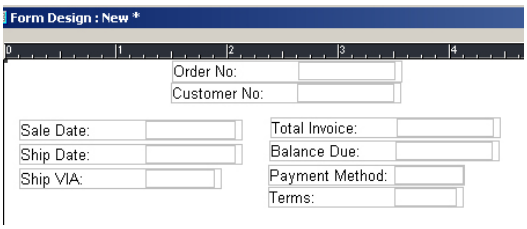


Рис. 11.2 Поля таблицы, сгруппированные по видам информации

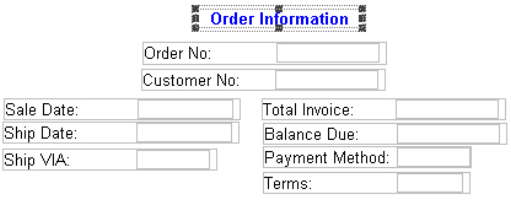


Рис. 11.3 Чертеж однотабличной формы с тестовым объектом

ком вы выбираете поле-объект, и лишь вторым - его метку).

6. Проинспектируйте выбранные метки. Установите опцию Font / Style / Italic: все метки будут изображены курсивом.
7. Проинспектируйте еще раз заголовок и включите опцию Text / Center: текст заголовка будет отцентрирован относительно границ текстового объекта.
8. Включите инструмент Box и изобразите рамку вокруг полей Sale Date, Ship Date, Ship VIA, Total Invoice, Balance Due, Payment Method, Terms (рис. 11.5).

Настроенная форма должна привлекать внимание пользователя к наиболее важной информации, содержащейся в ней, и обеспечивать простоту ввода данных.

Рассмотренный пример демонстрирует лишь малую долю возможностей, предоставляемых окном разработки Form Design. Более широкое их использование зависит от воображения и вкуса разработчика.

9. Проинспектируйте рамку и установите с помощью опции Frame / Style ее стиль.
10. Проинспектируйте страницу (фон) формы и с помощью палитры
11. Выберите текстовый объект заголовка и области редактирования всех полей. Проинспектируйте их и с помощью палитры Color задайте белый цвет.

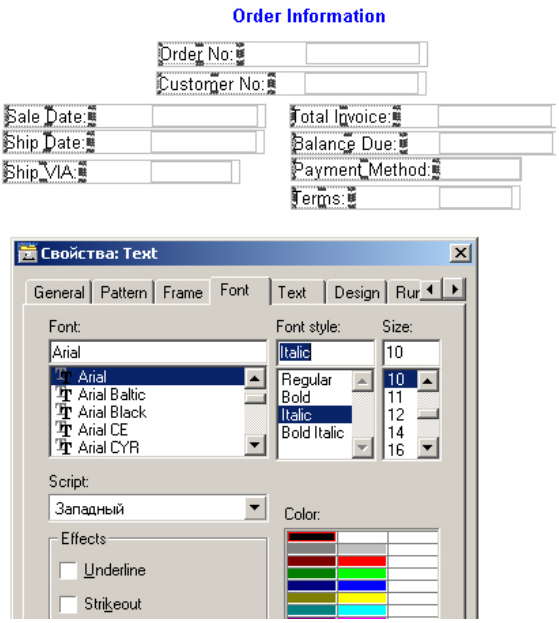


Рис. 11.4 Инспектирование полей в окне Form Design

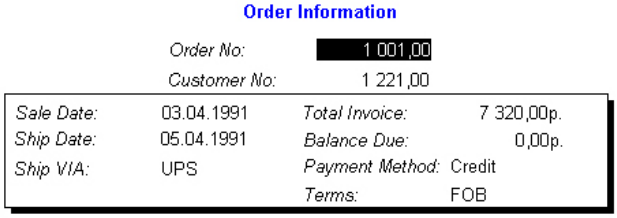


Рис. 11.5 Размещение прямоугольника на чертеже формы

Чтобы привести объекты к единым размерам и выровнять их взаимное расположение, вы можете воспользоваться командами Align и Adjust Sizing меню Design, описанными в Главе 11.

Выбор формата страницы

Задать размер страницы формы можно с помощью команды Form / Page Setup. В обоих случаях вам предоставляется диалоговое окно Page Layout (рис. 11.6).

В окне Page Layout вы можете выбрать стандартный размер страниц либо задать свой собственный, а также указать, для чего вы разрабатываете форму - для экрана компьютера или для печати на принтере (в последнем случае вы можете задать ориентацию формы на листе бумаги).

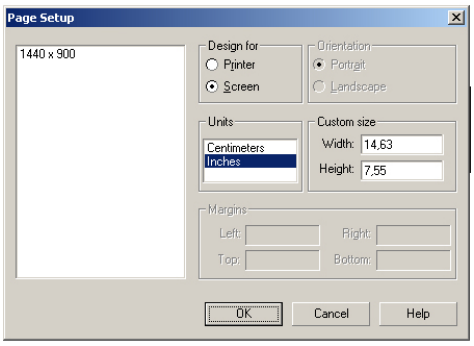


Рис. 11.6 Диалоговое окно Page Layout

Экранные формы

По умолчанию Paradox считает, что форма разрабатывается для работы с нею на экране компьютера. При этом вы можете использовать все экранные шрифты, установленные в вашей системе Windows Будучи распечатанной, форма может сильно отличаться от экранного варианта, если набор шрифтов принтера не совпадает с экранными шрифтами.

Панель Screen Size окна Page Layout для экранных форм показывает размер экрана в пикселах, который определяется экранным драйвером, а размеры формы задаются на панелях - Custom Size и Units,

Формы для распечатывания

При работе с формой, предназначенной для распечатывания, Paradox ограничивает вас шрифтами, установленными для вашего принтера, но гарантирует идентичность экранного и «твердого» вариантов формы.

Панель Paper Sizes окна Радо Layout предлагает выбор стандартных форматов страниц, но вы также можете задать свой собственный формат непосредственно на панели Custom Size.

Объекты форм

Данный раздел описывает свойства объектов формы. В таблице приведен алфавитный список всех возможных свойств, их значение и типы объектов, которые могут обладать данным свойством. Свойства графиков и кросстаблиц подробно рассматриваются в Главе 13.

Таблиц» 11.1 Свойства объектов формы

Свойство	Описание	Типы объектов
Alignment	Выравнивает текст по левой границе, по правой, по обеим границам либо по центру	Текст, поле
Button / Check Box	Превращает объект в кнопку-включатель	кнопка
Button / Push	Превращает объект в командную кнопку (данный тип кнопки используется по умолчанию)	кнопка
Button / Radio	Превращает объект в кнопку-селектор (радио-кнопку)	кнопка
Center Label	Центрирует текстовую метку командной кнопки	кнопка
Color	Изменяет цвет выбранного объекта или его части	Прямоугольник, линия, овал, текст, поле, мультизапись, график, кросстаблица,таблица
Data Type	Выбирает тип графика -табулярный, одно- или двухмерный	График
Define Crosstab	Связывает данные таблицы с объектом типа кросстаблица	кросстаблица
Define Field	Связывает поле таблицы с поле-объектом	поле
Define Graph	Связывает данные таблицы с объектом типа график	График
Define Graphic	Связывает данные таблицы с графическим объектом	Графический объект
Define OLE	Помещает OLE -значение в OLE-объект	OLE объект
Define Table	Связывает данные таблицы с табличной сеткой	таблица
Design Sizing / Fixed Size	Фиксирует размер текстового объекта вне зависимости от объема содержащегося текста	текст
Design Sizing / Fit Text	Текстовый объект автоматически изменяет свои размеры в зависимости от объема текста	текст
Design Sizing / Grow Only	Текстовый объект может автоматически только расширяться	текст

Design / Contain Objects	Все объекты, находящиеся внутри границ данного объекта, являются вложенными в него	
Design / Pin Horizontal	Фиксирует объект на чертеже в его текущей позиции по горизонтали	Все объекты
Design / Pin Vertical	Фиксирует объект на чертеже в его текущей позиции по вертикали	Все объекты
Design / Size To Fit	Позволяет объекту изменять свои размеры в зависимости от объема данных	Таблица, поле, графика,
Detach / Header	Отделяет заголовок табличной сетки от самой таблицы	таблица
Display Type / Check Box	Изображает значения поля в виде кнопки-включателя	поле
Display Type / Drop -Down Edit	Изображает значения поля в виде раскрывающегося списка	Поле
Display Type / Labeled	Изображает поле-объект с текстовой меткой	Поле
Display Type / List	Изображает значения поля в виде списка	Поле
Display Type / Radio Buttons	Изображает значения поля в виде набора радио- кнопок.	Поле
Display Type / Unlabeled	Изображает поле-объект без текстовой метки	поле
Font	Выбирает начертание, стиль, цвет и размер шрифта	Текст, поле, таблица, график, кросстаблица
Format / Date Format	Изменяет формат отображения поля типа дата	Поле
Format / Logical Format	Изменяет формат отображения dBASE-логического поля	поле
Format / Number Format	Изменяет формат отображения числового поля	Поле
Format / Time Format	Изменяет формат отображения поля, содержащего значение времени суток	Поле
Format / Timestamp Format	Изменяет формат отображения поля, содержащего одновременно дату и время суток	поле
Frame	Изменяет стиль, цвет и толщину рамки объекта	Прямоугольник, текст, график, графика, поле, мультизапись
Graph Type	Выбор типа графика	График
Grid	Изменяет цвет и стиль линий табличной сетки, позволяет отображать линии, разделяющие записи	Таблица, кросстаблица
Horizontal Scroll Bar	Отображает под объектом горизонтальную линейку прокрутки	Графика, поле, таблица, кросстаблица

Line Ends	Позволяет изображать стрелки на одном или обоих концах линии	линия
Line Spacing	Изменяет межстрочный интервал текста	текст
Line Style	Выбор стиля линии - сплошная, пунктирная, комбинированная	Линия, овал
Line Type	Выбор прямой линии или кривой	линия
Magnification	Масштабирует объект внутр»и его рамки «контейнера»	графика
Methods	Присоединяет к объекту методы на языке ObjectPAL	Все объекты
Options	Позволяет форматировать график-задавать: оси, сетку, метки, легенду и заголовок	график
Pattern	Выбирает тип штриховки объекта	Линия, овал, текст, график, таблица, поле, мультизапись
Raster Operation	Изменяет способ взаимодействия графического изображения с объектами, находящимися под ним	график
Record Layout	Изменяет способ расположения и количество повторяющихся областей многозаписного объекта	Мультизапись
Run Time / Complete Display	Позволяет отображать memo или форматированное memo поле полностью или только ту часть, которая хранится непосредственно в таблице	Поле
Run Time / No Echo	Позволяет вводить данные, но не отображает их	поле
Run Time / Read Only	Предохраняет данные от редактирования	поле
Run Time / Tab Stop	Позволяет использовать клавишу Tab для перемещения к данному объекту в окне Form	Поле, график, кнопка
Run Time / Visible	Делает объект видимым либо невидимым в окне Form	Все объекты
Search Text	Производит поиск и замену символов в тексте	Текст
Style	Придает кнопке внешний вид либо в стиле Windows, либо в стиле Borland	кнопка
Thickness	Устанавливает толщину линии или рамки	Линия, овал, текст, прямоугольник, поле, график, графика, мультизапись

Vertical Scroll Bar	Изображает справа от объекта вертикальную линейку прокрутки	текст, поле, графика, мультизапись
Word Wrap	Автоматически переносит данные на новую строку при достижении правой границы объекта	Текст, поле

Прямоугольники, эллипсы, линии

Прямоугольники и эллипсы, обычно, используются для изображения рамок вокруг других объектов. Получить объект, оформленный рамкой, можно как размещением рамки вокруг объекта, так и, наоборот, помещением объекта внутрь рамки.

Настроив соответствующим образом свойства прямоугольника или эллипса (рис. 11.7), вы можете придать ему необходимый внешний вид и функциональность.

При удалении рамки помните, что объекты, находящиеся внутри нее, будут также удалены, если включено свойство прямоугольника или эллипса Contain Objects (для восстановления случайно удаленного объекта можно воспользоваться командой Edit / Undo).

С помощью инструмента Line вы можете изобразить в форме линии самого разного вида (подробности изложены в Главе 10).

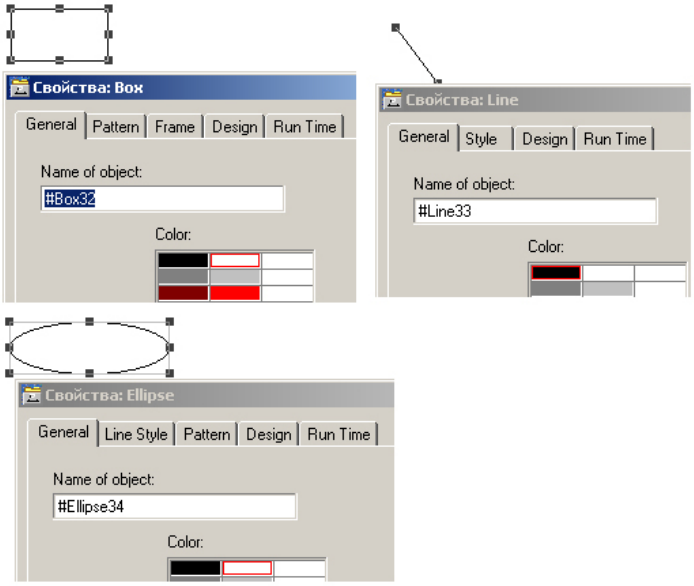


Рис. 11.7 Меню свойств прямоугольников, эллипсов, линий

Текстовые объекты

Текст используется для создания заголовков, комментариев, инструкций, меток других объектов и т.д.

Область действия задаваемых вами свойств текстового объекта (рис. 11.8) зависит от того, как вы его проинспектировали:

- Если вы выбрали объект целиком (после щелчка мышью в любом другом месте экрана вы щелкнули его правой клавишей),
- вызванное вами меню свойств относится ко всему тексту внутри объекта (при этом он окружается рамкой с «ручками»).
- Если вы инспектируете предварительно выделенную часть текста, ее свойства изменяются независимо от остального текста объекта.
- Если вы установили курсор мыши на текстовый объект и щелкнули правой клавишей, вы изменяете свойства вводимого текста, никак не затрагивая при этом уже существующий.

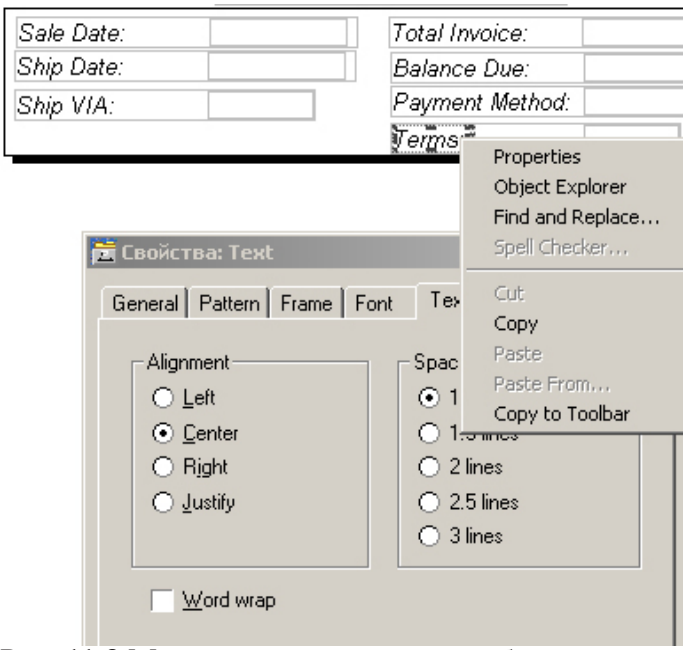


Рис. 11.8 Меню свойств текстовых объектов

Если опция Word Wrap текстового объекта выключена, он может содержать только одну строку. При этом клавиша Enter не работает.

Для выравнивания текста по границам объекта, установки межстрочного интервала, отступов, табуляций и абзацев вы можете использовать дополнительную линейку окна Form Design (см. Главу 10).

То, как вы создаете текстовый объект, определяет начальные установки его свойств Design Sizing (см. Главу 10). Проинспектировав объект, вы можете изменить их. Учтите, что свойства Design Sizing действуют только в окне разработки, но не при просмотре или распечатывании документа.

Следующие свойства появляются в окне General

- Fixed Size - размер объекта не зависит от объема находящегося в нем текста. Размер объекта вы можете изменить вручную. Для просмотра большого количества текста в небольшом объекте к нему может присоединить вертикальную линейку прокрутки (свойство Vertical Scroll Bar)
- Fit Text - размер объекта определяется объемом находящегося в нем текста. Если при этом опция Word Wrap выключена, объект может изменяться только по ширине.
- Grow Only - объект может увеличиваться при вводе в него дополнительного текста, но не может сжиматься при удалении текста (это свойство, обычно, используется в метках полей). Как и в предыдущем случае, при выключенной опции Word Wrap высота объекта остается неизменной.

Чтобы отредактировать текущий текстовый объект, поместите в него текстовый курсор, щелкнув мышью в нужном месте текста. Если вы предпочитаете , работать с клавиатурой, выберите объект клавишей Tab и нажмите Spacebar. При редактировании вы можете, как обычно, использовать клавиши Del, Backspace и другие, а также кнопки SpeedBar и команды меню Edit.

Учтите, что введенный вами текст Paradox сохраняет только при сохранении всего документа в окне Form Design.

Графические объекты

Информация, связанная с созданием, размещением и использованием свойств (рис. 11.9) графических объектов , была изложена в Главе 10.

В форме вы также можете оснастить графический объект вертикальной и горизонтальной линейками прокрутки. Команда Past From позволяет вставить рисунки из файлов, имеющих графический формат.

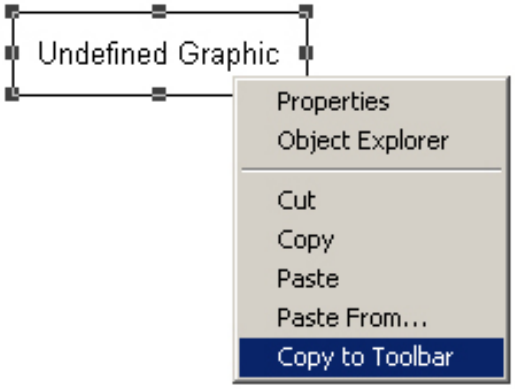


Рис. 11.9 Меню свойств графических объектов

OLE-объекты

OLE-объекты позволяют разместить в форме целые файлы из других программ и поддерживать с ними связь. Создание и размещение OLE-объектов рассматривается в Главе 14.

OLE-объекты обладают стандартными свойствами (рис. 11.10), все они были подробно описаны в Главе 10.

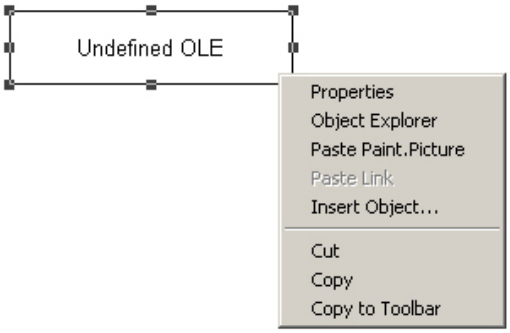


Рис. 11.10 Меню свойств OLE-объектов

Кнопки

Инструмент Button окна Form Design предназначен для размещения в форме различного типа кнопок. При нажатии кнопки мышью вызывается процедура, которая определяется присоединенными к кнопке методами, написанными на языке ObjectPAL. При создании кнопки можно использовать стандартные методы которые вставляются при помощи двух окон (рис.11.11),.

Второй метод позволяет через Object Explorer не только вставлять методы пользователя, но и обрабатывать события, которые производятся над объектом (рис.11.12)

При создании кнопки она по умолчанию содержит текстовый объект с одним словом «LABEL». Свойство кнопки Center Label автоматически располагает его посередине кнопки. Вы можете удалить текстовый объект и вместо него разместить на кнопке графический объект -иконку (чтобы кнопка при этом функционировала правильно, ее опция Contain Objects должна быть включена).

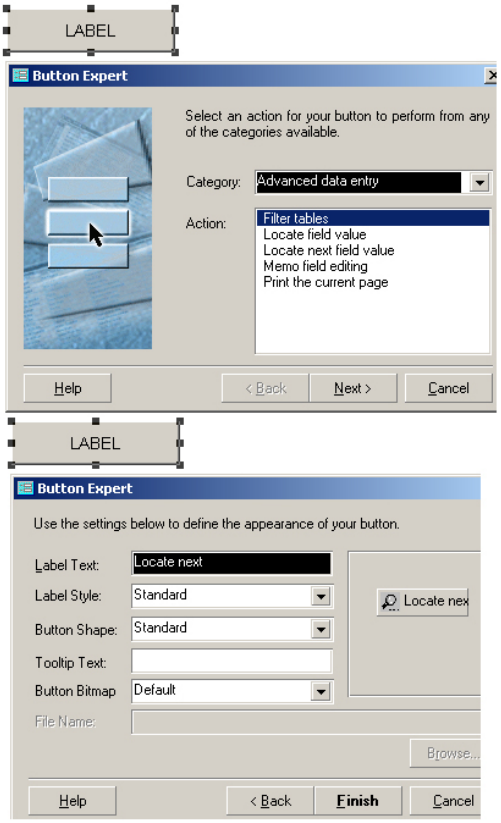


Рис. 11.11 вставка стандартных методов

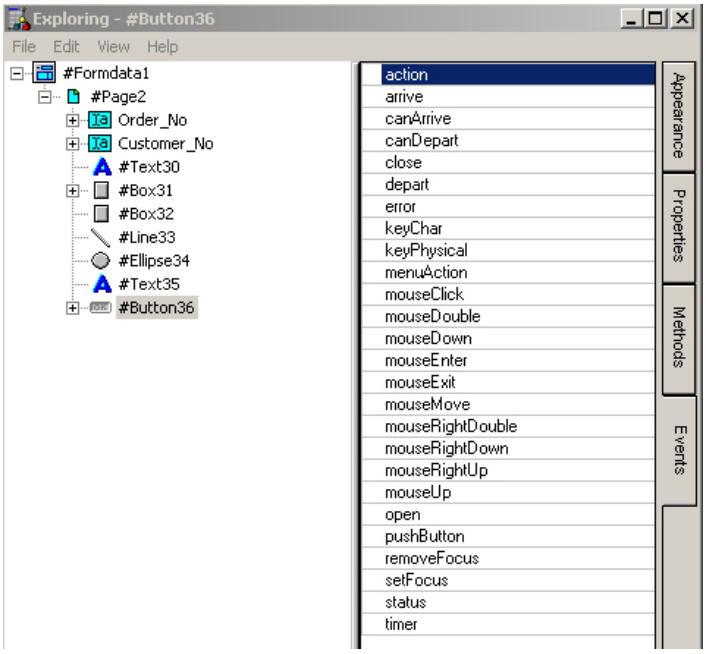


Рис. 11.12 окно Object Explorer :закрепление методов и обработка событий

Поле-объекты

В форме поле - объекты отображают данные из таблицы (таблиц), на основе которой разработана форма. Размещение и определение поле - объектов, а также большинство их стандартных свойств (рис. 11.13) было рассмотрено в Главе 10.

С помощью свойства вы можете изменить способ изображения поле -объекта. По умолчанию включена опция Labeled: поле-объект изображается с меткой. Чтобы убрать метку, включите опцию Edit.

Учтите, что значения, превышающие заданный размер поля таблицы, обрезаются при вводе. Кроме того, значения должны соответствовать правилам проверки корректности, определенным для данного поля таблицы. Как видно из рис.11.14 текущее состояние поля можно менять, причем это можно делать не только в ручном режиме, но и при помощи языка программирования PAL.

В верхней левой части находится иконка окна Data Model и ниже указаны файлы, которые подключены к этой форме. Открыв любой из перечисленных файлов кроме полей пользователя можно включить служебные поля, которые находятся в конце списка (рис.11.15). В поле Summery указывается значение (сумма, среднее, минимальное, максимальное и т.д.). в данном случае берется просто значение. В поле Special field можно указать день, номер страницы, время.

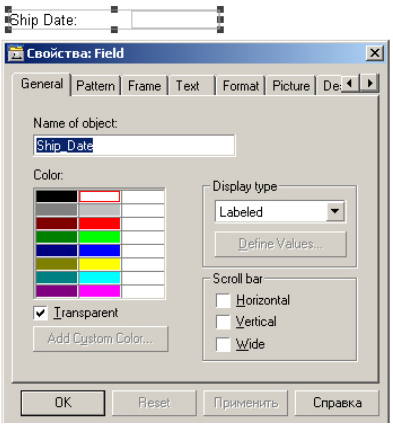


Рис. 11.13 Меню свойств полей

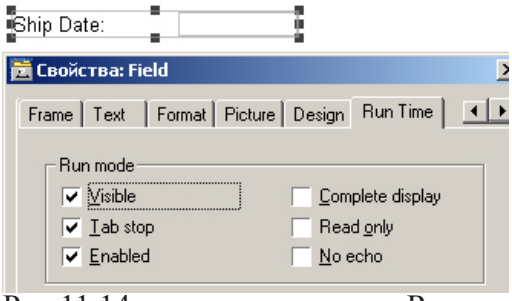


Рис.11.14 исходные значения Run Time свойств

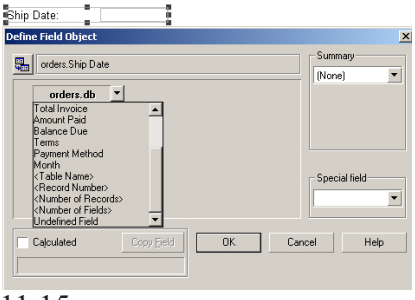


рис.11.15 окно определения поля

Статистические операции, перечисленные в таблице 11.2 производятся над некоторым набором записей таблицы. Если вы разрабатываете однотабличную форму, то этот набор - вся ваша таблица. В случае многотабличной формы набор определяется иерархической моделью данных формы.

Таблице 11.2 Статистические операторы

Оператор	Функция
Sum	Сумма значений
Count	Количество непустых значений
Mm	Минимальное значение
Max	Максимальное значение
Std	Стандартное отклонение
Var	Дисперсия
Avg	Среднее значение

Поле может быть и вычисляемым. Для этого надо поставить метку в окне Calculated. В этом случае автоматически снимается метка с поля Tab Stop (рис.11.14) и курсор на этом поле не будет останавливаться. В формуле могут использоваться любые поля из таблиц, включенных в форму. Значение Summery также учитывается.

При использовании знаков арифметических действий их приоритет стандартный, т.е. сначала выполняется умножение и деление, потом сложение и вычитание. Для выбора поля используется кнопка Copy Field.

Ввод данных в поле производится в режиме редактирования (таблица открыта). Если это поле используется в вычислениях, то это значения не имеет, но в других случаях (например, выполнение запроса) таблица, содержащее это поле должна быть закрыта.

Это условие можно обработать при помощи обработки событий или описав условие дополнительными методами в Object Explorer.

В зависимости от типа поля, можно изменить его формат (рис.11.16). Кроме стандартных форматов можно использовать свои. Например, надо использовать четыре знака после запятой (часто используется как тариф). После нажатия кнопки Create New Format откроется окно изображенное на рис. 11.17.

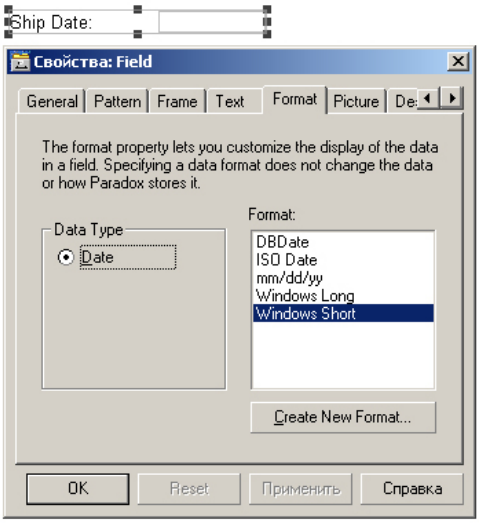


Рис. 11.16 Различные форматы полей

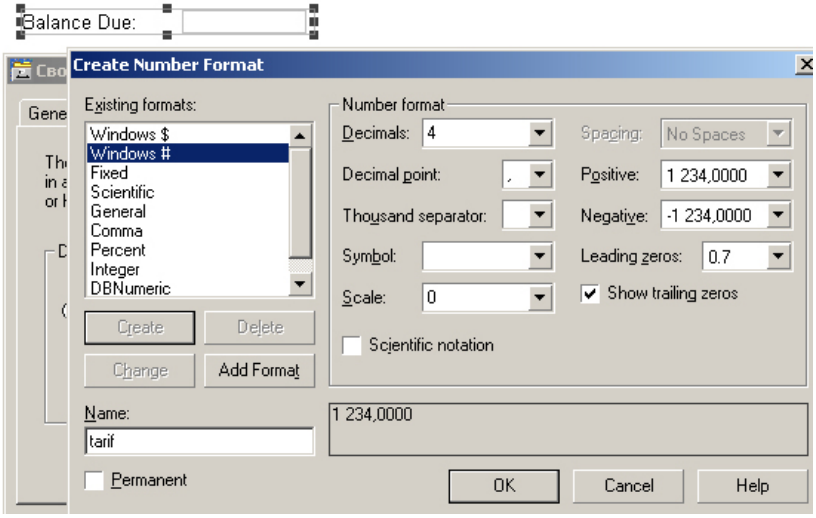


Рис. 11.17 создание личного формата

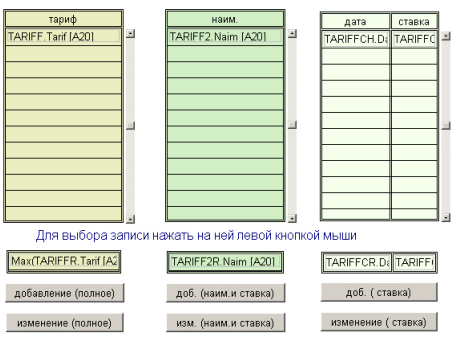


Рис.11.18 рабочая форма для выбора тарифа.

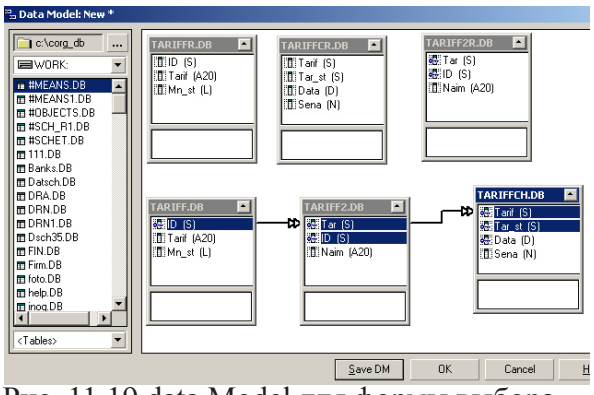


Рис. 11.19 data Model для формы выбора тарифа

Фильтр и шаблоны используются в таблицах и в форму переносят все свои свойства. Каждое поле одной таблицы может быть связанно с полем другой таблицы, входящей в форму. Связи задаются в Data Model. Например, имеем три таблицы, определяющих тариф. Рабочая форма для выбора тарифа показана на рис.11.18. Т.е. в крайней левой таблице выбирается вид тарифа. Автоматически в средней выбираются все наименование, которые связаны с этим видом. Последний этап – после выбора наименование автоматически по нему выбираются все ставки и начало их действия.

Для связывания необходимо, чтобы таблицы имели одинаковые поля (по названию и размерности). Эти поля были ключевыми и находились (как все ключевые поля) в начале. Таблицы из каталога выбираем двойным счелчком мыши. Поля связываем обычным перетаскиванием одного поля на место другого.

Разорвать связь можно вызвав правой кнопкой свойства и выбрать Unlink. Использование полей с рисунками и Мемо аналогично описанному выше. Использование графического поля в форме показано на рис.11.20. использование горизонтальных и вертикальных линеек позволяет сохранить масштаб.

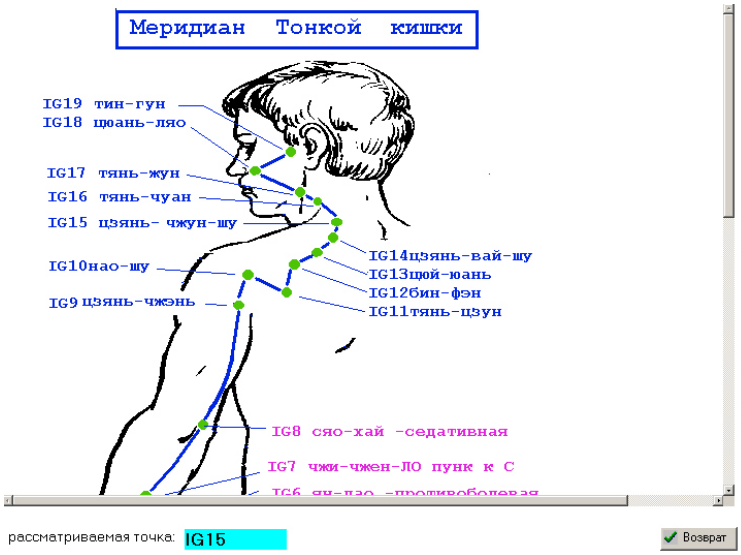


Рис.11.20 использование графического поля в форме

Рассмотрим более детально работу со статистическими операторами. Предположим, ваша модель данных аналогично изображенной на рис. 11.21. Двойные стрелки означают, что каждой записи одной таблицы соответствует несколько записей другой. В данном случае вы можете вычислять статистические характеристики полей таблицы Orders, относящихся к одной записи таблицы Customer. Иными словами, Paradox производит вычисления над полями записей, связанных с текущей главной записью. Аналогично, текущая запись таблицы Orders определяет набор записей таблицы Lineitem, статистические характеристики которого вы можете получить.

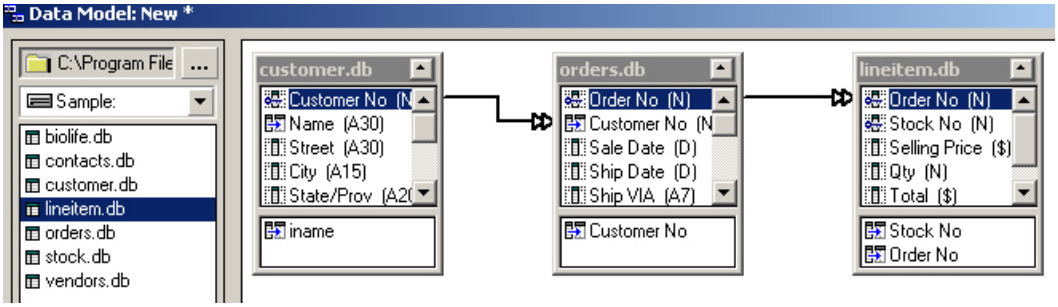


Рис. 11.21 Сложная модель данных

Обратите внимание, что в случае модели данных Customer->Orders-»Lineltem нельзя вычислять статистические характеристики набора наименований товаров, заказанных одним клиентом, поскольку Paradox способен производить такие расчеты только над одним уровнем иерархии модели данных.

Рис. 11.22 иллюстрирует использование статистических полей в форме, которая построена на основе рассмотренной выше модели данных Учтите, что в формах на основе модели данных типа 1-»M-»M статистические поля должны размещаться либо в повторяющейся области связанной таблицы (в табличной сетке или в многозаписном объекте), над

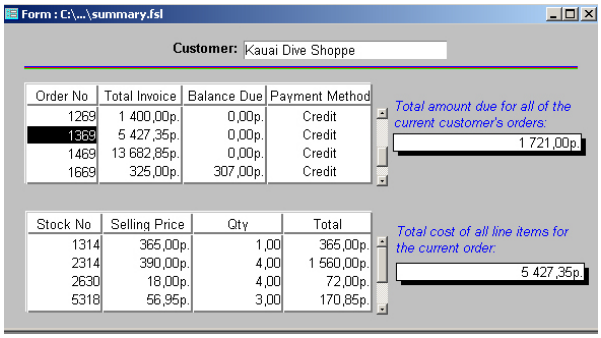


Рис. 11.22 Чертеж трехтабличной формы со статистическими полями

записями которой производятся вычисления, либо в повторяющейся области таблицы, которая находится одной ступенью выше в иерархии данных.

Включение поля графика несколько отличается от остальных полей. Проинспектировав иконку «График» сначала выделяется поле, где будет расположен сам график. Далее автоматически открывается окно, где надо указать таблицу (таблицы) или запрос, поля которого будут показаны в виде графика. В следующем окне выбираем будет ли это статистическая обработка или независимые значения. В третьем окне выбираем тип графика. В четвертом и пятом – поле по оси X и оси У. и в последнем указываем названия. В дальнейшем эти значения можно изменить, вызвав свойства. Учтите, что в зависимости от положения курсора на графике, будут вызываться свойства именно этого участка. На рис.11.23 показан один из вариантов

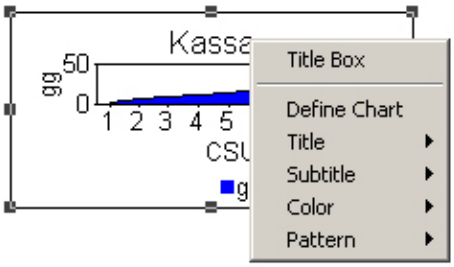


Рис. 11.23 изменение свойств графика.

Таблицы

Таблица является совокупностью других объектов. Табличные объекты формы имеют широкие возможности настройки для получения необходимого представления данных. Форма в табличном виде представлена на рис.11.24.

Размер поля можно изменять по высоте и ширине простым буксированием границ (курсор в этом случае принимает форму обоюдоострой стрелки). Колонки можно удалять и добавлять, менять их порядок. Для изменения места необходимо выделить колонку и потом отбуксировать ее в нужное место.

путевые листы

дата / N п/л	таб N	ФИО / N авто, тип авто	код марш	авто / топливо	регуляр-ность / общ.проб.	план / факт(руб)
03.11.2009 26120	160035	Юмачиков Р.Р. О126 ПАЗ 3225	1	ПАЗ 3225 пропан	0,971 155,00	1 752,00 -616,00
03.11.2009 26121	110032	Потапенко А.Н. Р594 ЛИАЗ 677	2	ЛИАЗ 677 пропан	0,971 160,00	2 240,00 -496,00
03.11.2009 26122	160206	Эберт А.А. Р093 ПАЗ 672	5	ПАЗ 672 пропан	0,971 165,00	2 200,00 -904,00
03.11.2009 26124	190227	Ребров Д.А. О642 Газель	2	Газель метан	0,971 170,00	2 000,00 0,00
03.11.2009 140533	140533	Солнышкина В.Г. Р594 ЛИАЗ 677	2	ЛИАЗ 677 пропан	0,971 0,00	2 240,00 -496,00
04.11.2009 26081	910862	Хаджаев М. Х. А859 Газель	2	Газель	150,00	1 230,00 -280,00
04.11.2009 26108	880722	Михцев М.М. А550 Газель	53	Газель	600,00	500,00 0,00
04.11.2009 26123	610077	Задорожный П.П. Т754 ЛАЗ 695	1	ЛАЗ 695 пропан	0,971 115,00	632,00 0,00
04.11.2009 26125	920879	Дерунец А.В. А159 Газель	1	Газель	277,00	1 060,00 -280,00

удалить | выбор таб.N | добавить | изменить | отчеты

Рис. 11.24 форма в табличном виде

Для наглядности часто используют графические поля с таблицами

Чтобы удалить колонку, выберите ее и нажмите клавишу Del (колонка выбирается щелчком левой клавиши мыши в одной из пустых строк таблицы).

Если необходимо вставить новую колонку, нгадо выделить всю таблицу и в свойствах выбрать Insert Column. Колонка добавится с левого края. Потом поле колонки надо определить и отбуксировать на нужное место.

Вы также можете в колонку таблицы поместить более, чем одно поле. В этом случае надо помнить, что если используются линейки прокрутки, то поле не должно касаться границ колонки. Иначе при прокрутке поле будет неподвижно.

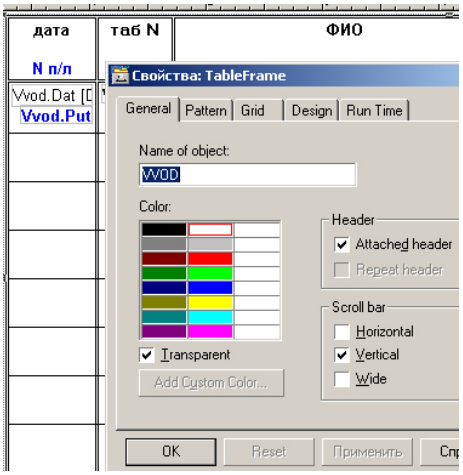


Рис. 11.25 Меню свойств таблиц

В таблицу можно помещать не только поля, но и другие объекты: линии, прямоугольники, эллипсы и даже графики и другие таблицы. Кроме настройки общей структуры таблицы, вы можете изменять индивидуальные свойства ее элементов, инспектируя строку заголовков, строку поле-объектов, отдельные заголовки и поле-объект.
Большинство свойств таблицы было рассмотрено в Главе 10.

На рисунке 11.26 показаны разные стили линий табличной сетки. Обратите внимание, что раз-

делительная линия между строками (свойство Grid\ Record Divider) изображается только в окне Form.

Опция Attached Header (рис.11.25) позволяет отделить строку заголовков от тела таблицы и расположить их на некотором расстоянии друг от друга или вообще удалить.

Если вы поместили несколько полей в одной графе, обработка и выбор свойств каждого поля надо делать индивидуально.

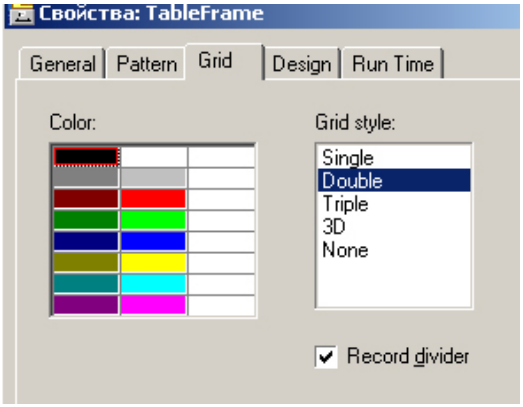


Рис. 11.26 Стили табличной сетки

Многозаписные объекты

Многозаписные объекты используются, если вы хотите отобразить одновременно несколько записей, но не в виде таблицы. Размещение и определение многозаписных объектов было рассмотрено в Главе 10, рис.10.33

В дополнение к свойствам поле-объектов, вы можете инспектировать область главной записи (ее свойства определяют свойства всех повторяющихся областей), а также всего многозаписного объекта в целом (его свойства относятся к площади страницы (фона) формы, занимаемой объектом).

Многостраничные формы

Команда insert / Page добавляет к форме чистую страницу. Paradox всегда вставляет новую страницу после существующих. Однако, вы можете изменить порядок следования страниц или воспользоваться командой Paste, чтобы поместить новую страницу между существующими.
Чтобы увидеть на экране одновременно все страницы формы, используйте команду Properties / Zoom / Best Fit.

Если вы хотите удалить страницу вместе со всеми содержащимися на ней объектами, выберите ее и нажмите Del, либо дайте команду Edit / Cut, либо щелкните мышью кнопку Cut на SpeedBar. Чтобы скопировать страницу, выберите ее и дайте команду Edit / Copy либо нажмите кнопку Copy на SpeedBar. Затем, чтобы вставить копию страницы или страницу, удаленную командой Cut, дайте команду Edit / Paste или нажмите кнопку Paste на SpeedBar. Страница вставляется перед текущей.

Изменить порядок следования страниц вы также можете с помощью команды Form / Page / Rotate, перемещающей выбранную страницу в последнюю позицию.

Команда Form / Tile Page позволяет контролировать изображение страниц разрабатываемой формы:

Данная команда, обычно, оказывается бесполезной, если вы задали большие размеры страницы в диалоговом окне Page Layout. В этом случае пользуйтесь командой масштабирования Properties / Zoom.

Для перемещения по страницам формы используются команды меню Page или вертикальная линейка прокрутки окна Form Design. В последнем случае перемещение к странице не приводит автоматически к ее выбору.

В отличие от отчетов в формах не используется разделитель страниц («page break»). Вместо него вы можете просто добавить в форму новую страницу.

При многостраничной форме надо помнить, что переносе задачи на другой размер экрана, могут появляться куски следующей или предыдущей страницы.

Настройка окна формы

Кроме свойств самой формы вы можете произвести настройку ее окна. Команда Format / Window Style служит для вызова диалогового окна Form Window Properties (рис. 11.27).

С помощью панели Window Style вы можете задать тип окна - обычное или диалоговое. Если вы выбрали предлагаемый по умолчанию обычный тип окна, некоторые опции будут включены и одновременно недоступны. Это означает, что это стандартные свойства окна, которые нельзя изменять. До-

ступными для изменения остаются:

- Заголовок окна. (Введите его в текстовое окошко Title)
- возможность использования линеек прокрутки
- Опция Size To Fit, позволяющая автоматически изменять размеры окна в соответствии с размерами страницы формы.
- Изображение стандартного меню формы. Обычно, оно выключается, если вы разработали с помощью ObjectPAL свое собственное меню.
- Вы также можете разработать форму в виде диалогового окна, использующего присоединенные к объектам формы методы, написанные на ObjectPAL. Это означает, что форма:
- Появляется на экране поверх остальных окон
- Может перемещаться пользователем, но пользователь не может изменить размеры окна формы.
- Если на панели Window Style выбрано Dialog Box, для вас доступны все опции кроме Size Го Fit и Standard Menu (при этом они включены)

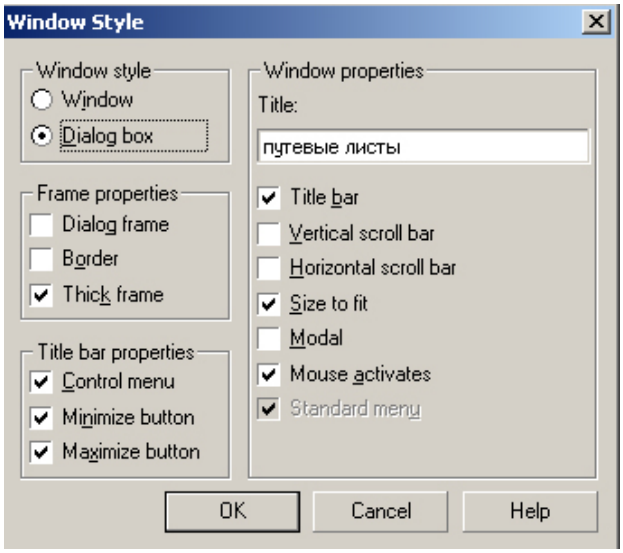


Рис.11.27 настройка окна формы

Вы можете изменять следующие свойства диалогового окна:

- Dialog Frame: рамка окна определяется установками Conrol Panel системы Windows.
- Border: рамка окна черного цвета.
- Thick Frame: широкая рамка черного цвета (данная опция недоступна, если включена опция Dialog Frame,).
- Control Menu: в левом верхнем углу окна располагается стандартное управляющее меню системы Windows.
- Minimize Button: в правом верхнем углу окна располагается кнопка минимизации окна.
- Maximize Button: в правом верхнем углу окна располагается кнопка максимизации окна.
- Title Bar: диалоговое окно оснащается строкой заголовка. Текст заголовка вводится в окошко Title.
- Vertical Scroll Bar и Horizontal Scroll Bar присоединяют к окну линейки прокрутки.
- Model: не позволяет пользователю работать где-либо кроме данного диалогового окна до его закрытия.
- Mouse Activates: выключение данной опции не разрешает покидать диалоговое окно щелчком мыши за его пределами. Например, если вы разработали свой собственный SpeedBar, то отключение опции Mouse Activates позволяет избежать перехода в окно SpeedBar каждый раз, когда вы нажимаете мышью один из его инструментов.
- Чтобы установки окна Form Window Properties начали действовать, необходимо сохранить форму, закрыть окно Form Design и открыть форму в окне Form.

Помните, что диалоговое окно без управляющего меню можно закрыть, нажав Alt+F4. Поэтому, если вы убираете Title Bar, то надо ставить кнопку закрытия формы, которая входит в стандартный набор.

Компиляция формы

Для компиляции формы служит команда Form / Deliver. Скомпилированная форма хранится в файле с расширением .FDL и может быть открыта только в окне Form. В скомпилированную форму нельзя вносить изменения ее исходный текст становится защищенным.

Обычно, компиляция форм производится после завершения разработки приложения в среде Paradox, перед вручением их пользователям базы данных.

Распечатывание формы

Несмотря на то, что формы предназначены для изображения данных на экране компьютера, вы можете распечатать форму на принтере прямо из окна Form.

Вы также можете распечатать чертеж формы из окна Form Design с помощью команды File /

Print. Вам будет предоставлено диалоговое окно в соответствии с установленным принтером.

Если надо распечатывать экранную форму во время работы, можно добавить в форму стандартную кнопку «Печать формы». Для этого при установке кнопки надо выбрать:

- Category - Advanced data entry
- Action - Print the current page

Учтите, если вы печатаете форму с данными, Paradox выводит на принтер только экранный отпечаток. Чтобы распечатать все записи таблицы, используйте отчет.

Глава 12 Разработка отчётов

В Главе 9 было рассказано, как использовать окна Data Model и Design Layout для определения информации, которую необходимо включить в отчет, и выбора исходного чертежа для нового отчета. Инструментальные средства, методы и концепции, общие для окон Report Design и Form Design, были описаны в Главе 10.

В данной главе обсуждаются функции, команды и способы представления выходных данных, которые можно использовать только в окне Report Design.

Вообще говоря, можно создавать отчеты и работать с ними даже не входя в окно Report Design. Paradox позволяет вызывать отчет со стандартным чертежом (см. Главу 4), что дает возможность быстро сформировать отчет. Кроме того, в окне Design Layout (см. Главу 9) предусмотрен широкий набор возможностей для генерации достаточно сложных многотабличных отчётов, что также может избавить от необходимости работать в окне Report Design.

Диалоговое окно Report Design используется в том случае, если вы хотите самостоятельно спроектировать отчет. Работа в окне Report Design позволяет:

- Размещать данные в зонах отчета
- Группировать данные по вашему усмотрению
- Перемещать объекты
- Устанавливать разделение страниц
- Вводить и удалять такие элементы отчёта, как рамки, поля, таблицы или графики
- Инспектировать и изменять свойства любого объекта на экран
- Просматривать на экране окончательный вид документа
- Использование стандартного отчета

В данном разделе демонстрируется использование стандартного отчета на примере модели данных, связывающей таблицы Customer и Orders. Вы познакомитесь с тем, как можно перемещать существующие объекты, вводить новые и контролировать внешний вид объектов в отчете.

В следующем примере показано, как вызвать стандартный многотабличный отчет и скорректировать его в соответствии с вашими потребностями. Этот пример служит только иллюстрацией возможностей окна Report Design, а об их реализации будет рассказано позднее.

Пример. Подготовка отчета в окне Report Design

1. Дайте команду File / New / Report
2. С помощью окна Data Model свяжите таблицы Customer и Orders.
3. Перейдите в окно Report Design. Затем выполните следующие действия:

- Удалите из таблицы Customer все поля, кроме Customer No и Name
- Из таблицы Orders удалите поля Sale Date, Ship Date, Amount Paid, Payment Method и Month

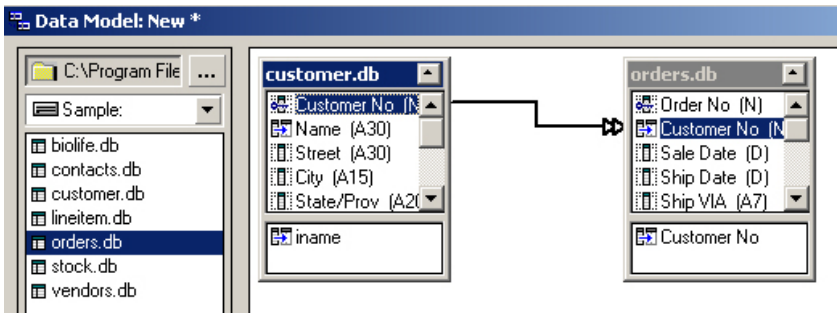


Рис.12.1 результат связывания таблиц

В стандартном отчете (рис. 12.2) поля главной таблицы выводятся в виде отдельных записей, а содержимое связанной таблицы - в табличной форме. В зоне Report размещаются заголовок и заключение отчета. По умолчанию эта зона пуста. Чтобы установить заголовок отчета, необходимо выполнить следующие действия:

- Щелкнуть мышью на границе зоны Report и отбуксировать верхнюю границу зоны вверх. При этом в зоне появятся несколько пустых строк.
- Разместить в зоне текстовый объект.

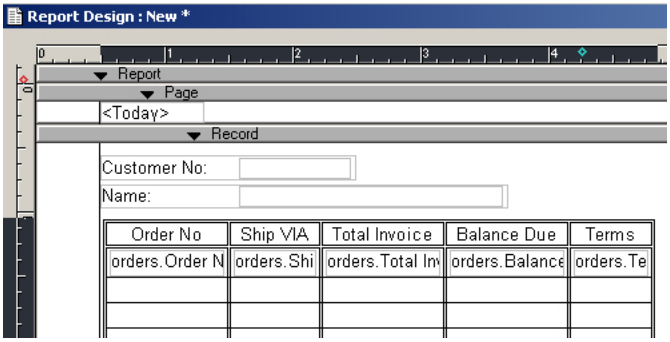


Рис. 12.2 Чертеж отчета на основе таблиц Customer и Orders

- Ввести в объект текст заголовка отчёта (например, Customer Orders (рис. 12.3).

В зоне Page отчёта размещаются верхние и нижние колонтитулы - вспомогательная информация, которая печатается на верхнем и нижнем полях каждой страницы отчета. По умолчанию Paradox помещает в эту зону три объекта: поле Today (текущая дата), специальное поле для имени главной таблицы модели данных и поле Page Number (номер страницы). Чтобы увидеть все эти поля, может понадобиться горизонтальная прокрутка окна.

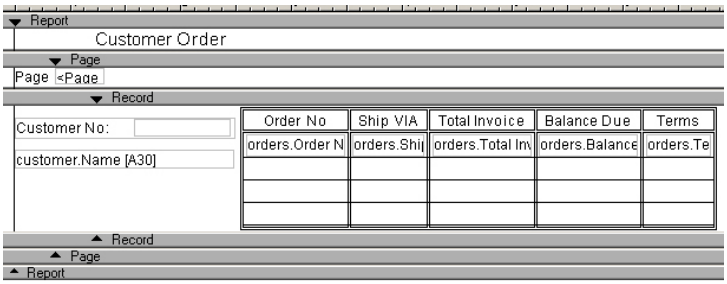


Рис. 12.3Чертеж отчета с текстовым объектом в качестве заголовка

- Удалите поле Today и поле имени таблицы поочередно каждое поле и нажмите клавишу Del,).
- Отбуксируйте поле Page к левой границе зоны.

Вы также можете отбуксировать целую таблицу. Разместите, например, таблицу Orders справа от полей таблицы Customer.

Теперь, чтобы увидеть всю страницу отчета по ширине, дайте команду View / Zoom / Fit Width.

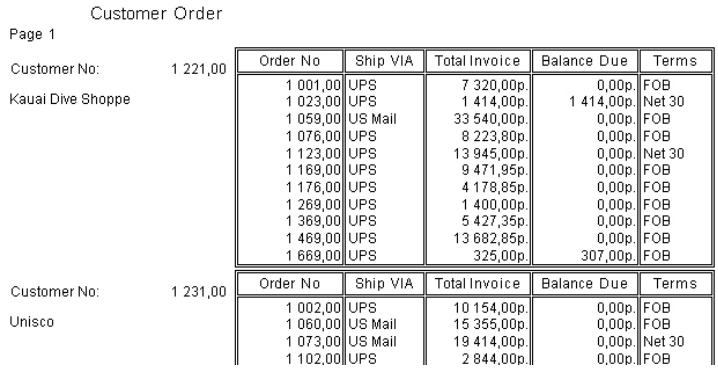


Рис. 12.4 Отчет на основе таблиц Customer и Orders

Чтобы выключить отображение метки поля Name, проинспектируйте это поле и выберите пункт меню Display Type / Unlabeled. С помощью команды File / Save сохраните подготовленный отчет. Дайте файлу имя Corders. Чтобы сформировать отчет, щелкните мышью кнопку View Data на SpeedBar (рис. 12.4). С помощью навигационных кнопок SpeedBar можно просматривать страницы готового отчета, а перемещение в пределах текущей страницы обеспечивается линейками прокрутки окна Report Design

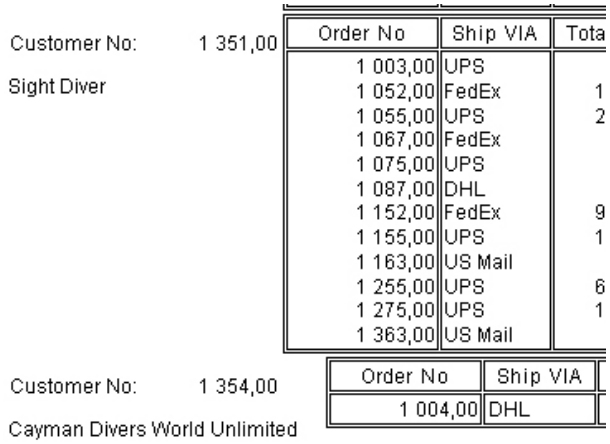


Рис. 12.5 выталкивание объекта

Отчет необходимо доработать, поэтому вернитесь в окно Report Design, щелкнув кнопку Design на SpeedBar.

Чтобы избавиться от пустых строк в отчете, измените высоту рамки таблицы Orders, а затем уменьшите размер зоны Record.

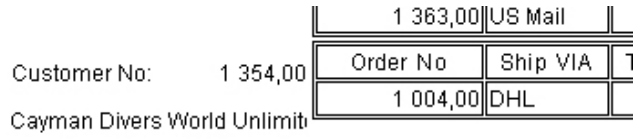


Рис.12.6 действие опции Run Time / Fit Width.

Для уменьшения размера зоны нужно щелкнуть левой клавишей мыши на какой-либо из пустых строк внутри зоны Record, а затем буксировать нижнюю границу зоны вверх до тех пор, пока она почти не коснется нижней границы рамки таблицы.

Чтобы увидеть, к чему привели сделанные изменения, снова сформируйте отчет. Обратите внимание, что в результате увеличения ширины поля Name, таблица сдвигается вправо (рис. 12.5). Вернитесь в окно Report Design, щелкнув кнопку Design на SpeedBar.

Проинспектируйте поле Name и выключите опцию Run Time / Fit Width. Теперь увеличение ширины поля Name при формировании отчета будет запрещено, и таблица не будет «выталкиваться» вправо.

Проинспектируйте поле Name, измените шрифт для этого поля на жирный. Аналогичным об-

разом установите для полей Page Number и Customer Number шрифт типа курсив

Чтобы в поле name фраза не обрезалась, выделим его и поставим метку в properties / text / word warp .

Чтобы увидеть окончательный результат, еще раз сформируйте отчет.

Дайте команду печати отчета File / Print (рис. 12.7)

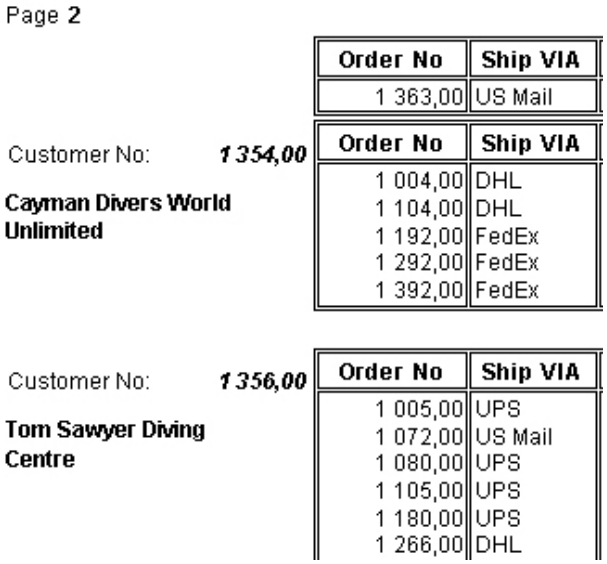


Рис. 12.7 Окончательный вид отчета на бумаге

Форматирование страницы отчета

Размеры страницы отчёта можно установить в окне Page Setup, которое вызывается file / Page Setup (рис. 12.8).

Paradox позволяет проектировать отчеты для просмотра на экране и для вывода на печать.

По умолчанию Paradox формирует отчет для вывода на печать. Если в окне Page Layout на панели Design For включена опция Printer, Paradox допускает использование только тех шрифтов, которые инсталлированы в вашем активном принтере. Это ограничивает возможности отображения информации на экране, зато гарантирует идентичность внешнего вида документа на экране и на бумаге.

Вы можете выбрать для отчета один из предлагаемых в списке Paper Sizes стандартных форматов страницы либо самостоятельно задать размеры страницы на панели Custom Size. Панель Margins позволяет изменять размеры полей страницы. Кроме того, с помощью панели Orientation вы можете задать вертикальное или горизонтальное положение страницы.

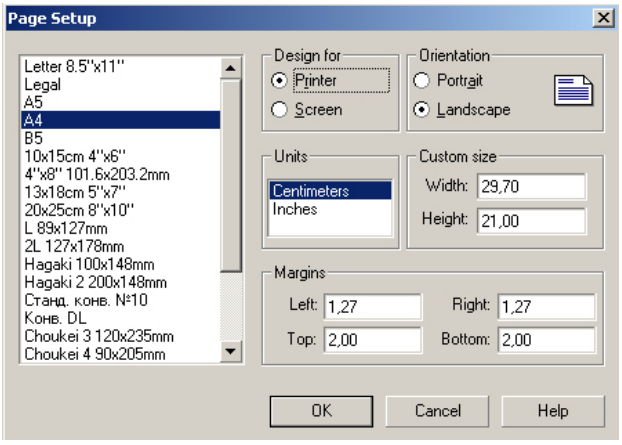


Рис. 12.8 Диалоговое окно Page Setup

Проектирование отчета для вывода на экран

Если вы хотите использовать в отчете экранные шрифты, включите опцию Screen на панели Design For. Эти шрифты не обязательно совпадают со шрифтами принтера, поэтому напечатанный отчет может отличаться от своего экранного представления. Конечно, если шрифты принтера совпадают с экранными, эта проблема не возникает.

При проектировании отчета для вывода на экран обязательно используется вертикальная ориентация отчета. Список Paper Sizes в окне Page Layout заменяется на список Screen Sizes, в котором отображаются размеры экрана (в пикселах) в текущем режиме видеоадаптера. Вы можете изменить формат страницы, введя нужные размеры в окошки панели Custom Size. Единицы измерения (дюйм или сантиметр) задаются на панели Units

Проектирование зон отчета

Для описания строения отчета в Paradox используются зоны. Зоны располагаются на экране в виде горизонтальных полос и отражают структуру сформированного отчета. В отчетах используются четыре типа зон:

1. Зона Report содержит данные для вывода в начале и в конце отчета. в этой зоне расположена информация, которая имеет общее отношение к отчету. На рис.12.7 видно, что наименование отчета не печатается уже на второй странице.
2. Зона Page определяет колонтитулы страниц отчета. Т.е. тут размещается информация, которая пе-

чатается в начале каждой страницы.
3. Зоны Group (необязательные и на рис.12.7 ее нет) используются для группирования и указания порядка следования записей таблицы, по которой формируется отчет.
4. Зона Record содержит записи таблицы, по которой формируется отчет.

С помощью пункта меню View / Band Label можно включить или выключить метки зон. Метки отображаются только в окне Report Design, поэтому выбор режима их вывода на экран никак не сказывается на внешнем виде готового отчета.

Изменение размеров зон

Зоны Page, Record и Report автоматически включаются в отчет. Эти зоны нельзя удалить; поэтому, если вам не нужна какая-либо из них, ее следует просто очистить от объектов и сжать по вертикали, чтобы в отчет не попадали пустые строки.

Изменяя размеры зоны, можно добавлять в отчет или удалять из него пустые строки. Для изменения размера зоны сначала необходимо ее выбрать, щелкнув левой клавишей мыши в любом месте внутри зоны или по ее границе. Существует три способа узнать, какая зона выбрана в данный момент:

- Если включена опция Properties / Band Labels, метка выбранной зоны выделяется цветом (метки выбранной зоны отмечены инверсным цветом по отношению к цвету меток невыбранных зон).
- Выбранная зона выделена цветом на вертикальной линейке у левого края окна Report Design.
- Имя выбранной зоны указывается в правой части поля системных сообщений (status bar) Desktop.

Изменить размеры зоны можно только мышью, эквивалентных команд клавиатуры не предусмотрено. Изменение размера выбранной зоны производится буксировкой ее границы по вертикали. Буксировать можно как верхнюю, так и нижнюю границы зоны. Если в зоне есть объект, буксировка верхней границы зоны добавляет или удаляет пустые строки над этим объектом, а буксировка нижней границы - под объектом.

Помните, что размер зоны не может быть меньше размеров находящихся в ней объектов. В таблицах 12.1-12.6 описаны различные манипуляции с границами зон.

Таблица 12.1 Изменение размеров заглавия отчета

Цель	Выбранная зона	Буксировать
Добавить строки над объектом в заглавии отчета	Report	Границу зоны Report вверх
Удалить строки над объектом в заглавии отчёта	Report	Границу зоны Report вниз
Добавить строки под объектом в заглавии отчёта	Report	Границу зоны Page вниз
Удалить строки под объектом в заглавии отчёта	Report	Границу зоны Page вверх

Таблица 12.2 Изменение размеров верхнего колонтитула

Цель	Выбрана зона	Буксировать
Добавить строки над объектом в верхнем колонтитуле	Page	Границу зоны Page вверх
Удалить строки над объектом в верхнем колонтитуле	Page	Границу зоны Page вниз
Добавить строки под объектом в верхнем колонтитуле	Page	Границу зоны Group или Record вниз*
Удалить строки под объектом в верхнем колонтитуле	Page	Границу зоны Group или Record вверх*

* Зона Group необязательна; если вы ввели эту зону, она появляется между зонами Page и Record.

Таблица 12.3 Изменение размеров зоны Record

Цель	Выбрана зона	Буксировать
Добавить строки над объектом в зоне Record	Record	Верхнюю границу зоны Record вверх
Удалить строки над объектом в зоне Record	Record	Верхнюю границу зоны Record вниз
Добавить строки под объектом в зоне Record	Record	Нижнюю границу зоны Record вниз
Удалить строки под объектом в зоне Record	Record	Нижнюю границу зоны Record вверх

Таблица 12-4. Изменение размеров нижнего колонтитула страницы

Цель	Выбрана зона	Буксировать
Добавить строки над объектом в нижнем колонтитуле	Page	Границу зоны Group или Record вверх
Удалить строки над объектом в нижнем колонтитуле	Page	Границу зоны Group или Record вниз
Добавить строки под объектом в нижнем колонтитуле	Page	Границу зоны Page вверх
Удалить строк под объектом в нижнем колонтитуле	Page	Границу зоны Page вниз

Таблица 12.5 Изменение размеров заключения отчета

Цель	Выбрана зона	Буксировать
Добавить строки над объектом в заключении отчета	Report	Границу зоны Page вверх
Удалить строки над объектом в заключении отчета	Report	Границу зоны Page вниз
Добавить строки под объектом в заключении отчета	Report	Границу зоны Report вниз
Удалить строки под объектом в заключении отчета	Report	Границу зоны Report вверх

Зоны типа Group не вводятся в чертеж отчёта по умолчанию. Если вами введена такая зона ее верхняя и нижняя границы располагаются между соответствующим границами зон Раде и Record

Таблица 12.6 Изменение размеров зоны Group

Цель	Выбрана зона	Буксировать
Добавить строки над объектом в верхнем колонтитуле зоны Group	Group	Границу зоны Group вверх
Удалить строки над объектом в верхнем колонтитуле зоны Group	Group	Границу зоны Group вниз
Добавить строки под объектом в верхнем колонтитуле зоны Group	Group	Границу зоны Record вниз
Удалить строки под объектом в верхнем колонтитуле зоны Group	Group	Границу зоны Record вверх
Добавить строки над объектом в нижнем колонтитуле зоны Group	Group	Границу зоны Record вверх
Удалить строки над объектом в нижнем колонтитуле зоны Group	Group	Границу зоны Record вниз

Добавить строки под объектом в нижнем колонтитуле зоны Group	Group	Границу зоны Group вниз
Удалить строки под объектом в нижнем колонтитуле зоны Group	Group	Границу зоны Group вверх

Зона Report

Зона Report предназначена для описания заглавия и заключения отчёта. Заглавие и заключение печатаются один раз в самом начале и в конце отчета. Типичная информация для заглавия отчета - его название и эмблема вашей фирмы В заключении отчета можно поместить обобщающую информацию, предназначенную для анализа данных. В принципе информация в заглавии и заключении может быть любой.

Пример. Размещение графического объекта в заглавии отчета

Чтобы поместить графический объект в заглавие отчета, необходимо:

1. Открыть окно Report Design с новым отчетом по таблице Customer.
2. Выбрать зону Report.
3. Освободить место в зоне, отбуксировав верхнюю границу зоны вверх.
4. Включить инструмент Graphic на SpeedBar.
5. Изобразить в нужном месте зоны Report рамку графического объекта.
6. Проинспектировать рамку графического объекта и выбрать пункт Define Graphic / Paste From, чтобы открыть диалоговое окно Paste From Graphic File (рис. 12.9).

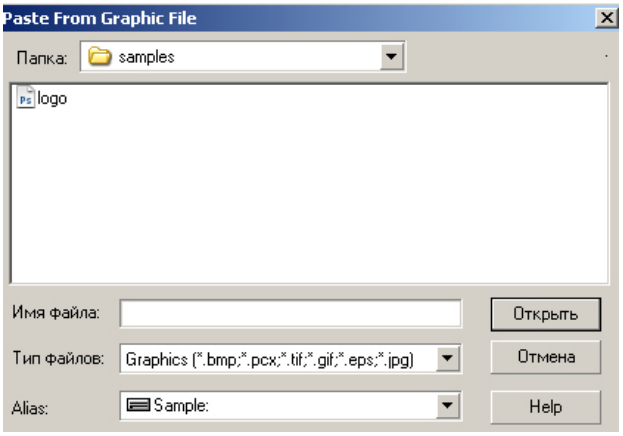


Рис 12.9 Диалоговое окно Paste From Graphic File

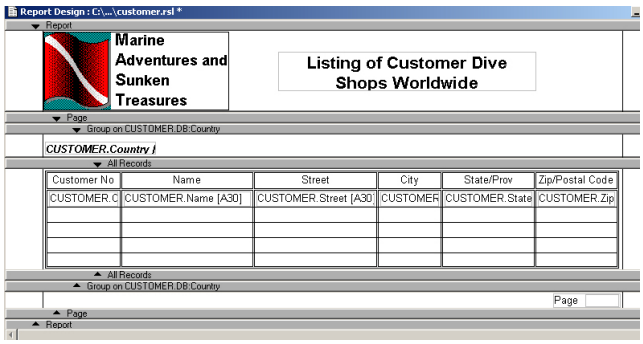


Рис. 12.10 Размещение графического объекта в заглавии отчета

Зона Page

Зона Раде определяет содержание колонтитулов каждой страницы отчета. По умолчанию в верхнем колонтитуле размещены три объекта:

- Поле Today, показывающее в верхней части каждой страницы дату формирования отчета
- Поле-объект, содержащий заголовок страницы - имя главной таблицы
- Поле Page, подзывающий номер страницы

Любой из этих объектов можно удалить или изменить. В зоне Page вы можете разместить любые объекты, которые должны выводиться в верхнем и нижнем поле каждой страницы отчета.

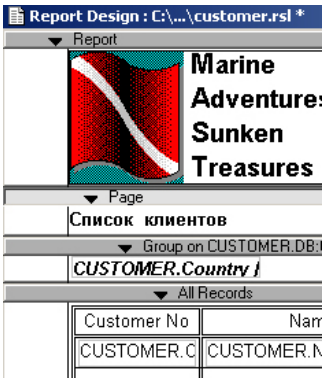


Рис. 12.11 Изменение верхнего колонтитула страницы

Пример. Изменение верхнего колонтитула страницы

В качестве исходного воспользуйтесь отчётом из предыдущего примера. Измените верхний колонтитул страниц отчёта следующим образом:

1. Удалите поле Today (выберите его и нажмите клавишу Del). Аналогичным образом удалите заголовок страницы и номер страницы. (Чтобы увидеть на экране поле Page может потребоваться горизонтальная прокрутка.)

2. Нажмите мышью инструмент Text на SpeedBar. Разместите в зоне Page текстовый объект.
3. Введите в текстовый объект строку Список клиентов.
4. Щелкните мышью вне текстового объекта (или нажмите Esc), чтобы отменить выбор данного объекта.
5. Проинспектируйте объект и выберите пункт Font / Style / Bold (см. рис. 12.11).

Учтите, что зона Page в отличие от остальных не растягивается по вертикали при выводе отчета на экран или принтер. Это означает, что Paradox отсекает часть объекта (например, таблицы), не помещающуюся в заданных границах зоны.

Зона Record

В зоне Record содержится основная часть отчета -записи таблицы, по которой формируется отчет.

- Если проектируется отчет табличного вида (это вид отчета задается по умолчанию), записи таблицы изображаются в табличной сетке, расположенной в зоне Record.
- При проектировании однозаписного отчета Paradox размещает в зоне Record поле-объекты, определенные полями таблицы Вы можете их перемещать, удалять или изменять.
- Если проектируется многозаписный отчет, Paradox размещает в зоне Record многозаписный объект с поле - объектами, соответствующими полям таблицы. Вы можете модифицировать этот многозаписный объект по вашему усмотрению.
- Если вы проектируете пустой отчет (бланк), Paradox не поместит в отчет ни одного объекта. Вы можете сделать это самостоятельно, используя инструменты SpeedBar.

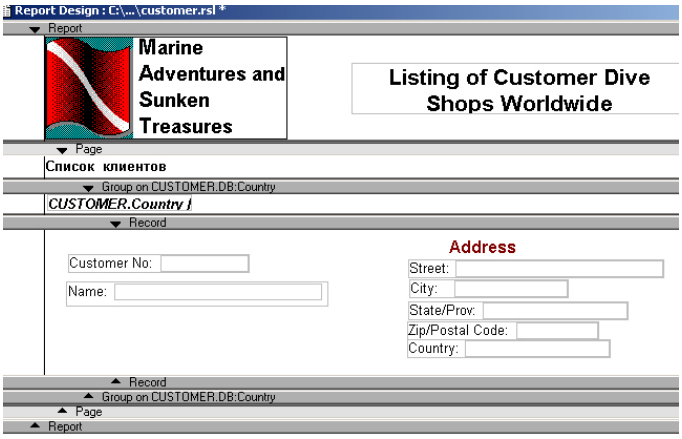


Рис. 12.12 Проектирование зоны Record

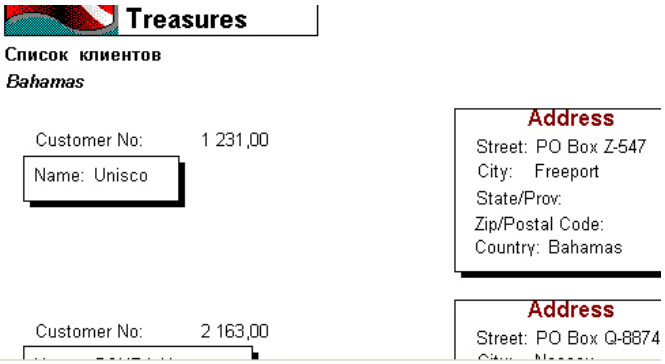


Рис. 12.13 Отчет по таблице Customer

Пример. Проектирование зоны Record

В качестве исходного воспользуйтесь отчетом из предыдущего примера. Измените зону Record следующим образом:

1. Выберите табличную сетку в зоне Record и удалите ее нажатием клавиши Del.
2. Нажмите мышью инструмент Field на SpeedBar. Разместите в зоне поле -объект. Проинспектируйте его и выберите пункт Define Field / CUSTOMER.DB:Customer No
3. Аналогичным образом разместите поле-объект Name.
4. Нажмите мышью инструмент Text на SpeedBar. Разместите
5. текстовый объект справа от поле-объекта Name
6. Введите в текстовый объект слово «Address « и нажмите Enter для создания пустой строки.
7. Вставьте в новой строке неопределенный поле- объект. При выделенном поле – объекте нажмите Ctrl / C (копировать в буфер обмена)
8. Нажмите Ctrl / V (вставить из буфера обмена)еще четыре раза, чтобы создать на отдельных строках пять полевых объектов
9. Проинспектируйте каждый из неопределенных объектов и определите их полями Street, City, State/ Prov, Zip/Postal Code и Country таблицы Customer.
10. Поле Name и поля предыдущего абзаца обведите прямоугольником
11. Проинспектируйте эти прямоугольники группу, и выберите пункт Frame / Style. С помощью пали-

тры Frame установите стиль рамки с тенью (рис. 12.12).

Если вы не хотите, чтобы при формировании отчета поле Name «выталкивало» вправо текстовый объект Address, необходимо либо проинспектировать поле Name и выключить опцию Run Time\Fit Width, либо проинспектировать поле Address и включить опцию Run Time / Pin Horizontal (см. далее в этой главе раздел «Run Time-свойства объектов», где описано взаимодействие объектов при просмотре или печати отчета).

Когда вы печатаете или просматриваете отчет, Paradox изображает все записи таблицы в том виде, как была спроектирована зона Record. На рисунке 12.13 изображен отчет, спроектированный в предыдущем примере .

Зона Group

Paradox позволяет вводить в отчет зоны Group, использующиеся для разбиения информации в отчете на отдельные группы. Признаком для объединения может быть значение (или диапазон значений) какого-либо поля записи или определенное количество записей таблицы.

Если записи группируются по значению поля, то они одновременно и сортируются по этому полю. Если, например, вы группируете данные по полю Country таблицы Customer, записи будут упорядочены по алфавиту.

Paradox всегда размещает зону Group между зонами Page и Record. Зона Group создается командой insert / Band При этом на экране появляется диалоговое окно Define Group (рис 12.14)

Записи в отчете можно сгруппировать по значению поля. Это также один из возможных способов сортировки информации в отчете. Например, вы можете подготовить отчет, в котором покупатели сгруппированы по странам или штатам, либо просмотреть счета, сгруппированные по способу оплаты или пересылки, либо посмотреть товарные запасы, сгруппированные по классам изделий
Пример. Группирование данных в отчете Customer по полю Country

Чтобы сгруппировать записи таблицы Customer по значению поля Country, выполните следующие операции Выбирите Insert / Band на SpeedBar. Paradox откроет окно Define Group, в котором в списке Table указана таблица Customer, а все поля таблицы перечислены в списке Field (рис. 12.14).

Выберите в списке Field поле Country. В текстовом окошке Band Label появится надпись Group on CUSTOMER.DB:Country. Нажмите. OK, и Paradox вставит зону Group между зонами Record и Page и разместит в ней поле Country.

Чтобы визуально обозначить границы групп, можно поместить в зоне Group какой-либо разделитель (например, горизонтальную линию).

При формировании отчета Paradox объединит записи в группы по значениям поля Country (рис. 12.11).

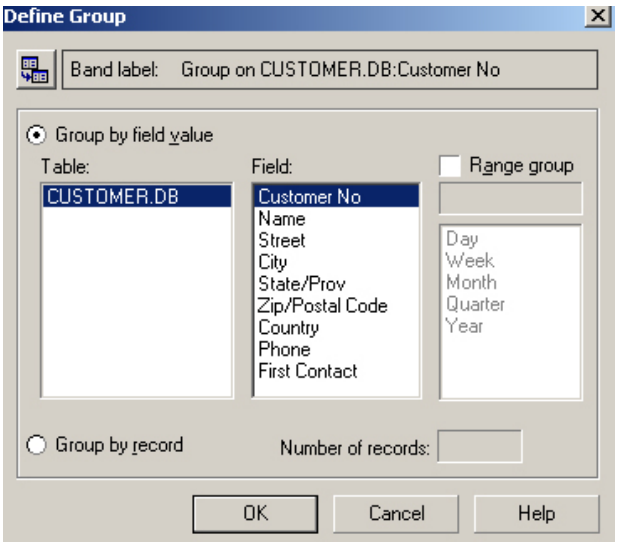


Рис. 12.14 Диалоговое окно Define Group
Группирование по значению поля

Группирование по диапазону значений поля

Вы можете объединить в группы записи таблицы, для которых значение выбранного поля находится в заданном диапазоне. Например, можно сгруппировать записи таблицы Orders по месяцам или кварталам или объединить записи Lineitem (виды заказанных вашими клиентами товаров) по значению в поле Qty (количество изделий).

После ввода зоны Group ее можно проинспектировать и изменить порядок сортировки и другие свойства групп записей

Для группирования записей по диапазону значении следует открыть окно Define Group (рис 12 14) Paradox показывает все таблицы модели данных отчёта в списке Table и все доступные поля в списке Field (Поскольку группировав по BLOB-полям невозможна они не отображается.) Выберите

нужное поле и включите опцию Range Group. Способ указания диапазона зависит от типа выбранного вами поля.

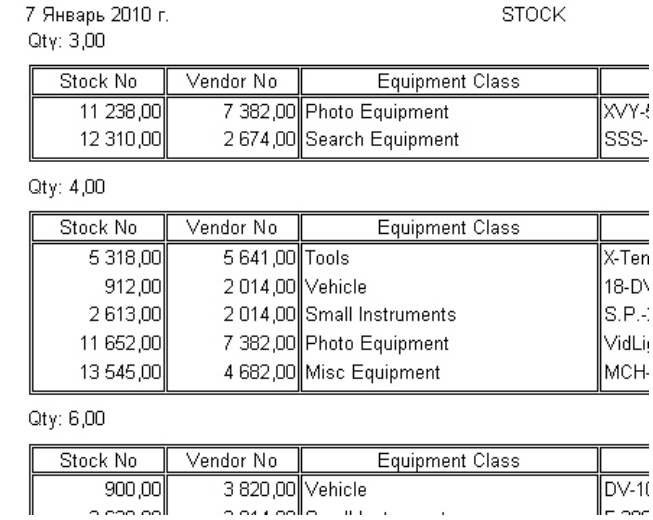
Если выбранное поле числового или денежного типа, вы должны указать в окошке Range Group шаг диапазона. При этом первая группа объединит записи со значениями поля от нуля до указанной вами шага диапазона, вторая - записи, попавшие в следующий промежуток значений, и т.д.

Paradox не выводит в отчет «пустые» (не содержащие ни одной записи) группы.

Пример. Группирование по диапазону числовых значений

Предположим, вы хотите провести количественную оценку ваших товарных запасов, данные по которым содержатся в таблице Stock.

1. Войдите в окно Report Design для проектирования нового отчета по таблице Stock.
2. Дайте команду Insert / Band. Paradox откроет окно Define Group.
3. В окне Define Group отметьте поле Oty, включите опцию Range Group и введите число 2 в окошко Range Group.
4. Paradox создаст зону Group.
5. Сформируйте отчет (рис. 12.15).


7 Январь 2010 г. STOCK

Qty: 3,00

Stock No	Vendor No	Equipment Class	
11 238,00	7 382,00	Photo Equipment	XVY-5
12 310,00	2 674,00	Search Equipment	SSS-

Qty: 4,00

Stock No	Vendor No	Equipment Class	
5 318,00	5 641,00	Tools	X-Ten
912,00	2 014,00	Vehicle	18-D\
2 613,00	2 014,00	Small Instruments	S.P.-:
11 652,00	7 382,00	Photo Equipment	VidLi
13 545,00	4 682,00	Misc Equipment	MCH-

Qty: 6,00

Stock No	Vendor No	Equipment Class	
900,00	3 820,00	Vehicle	DV-1(

Рис. 12.15 группировка по диапазону значений

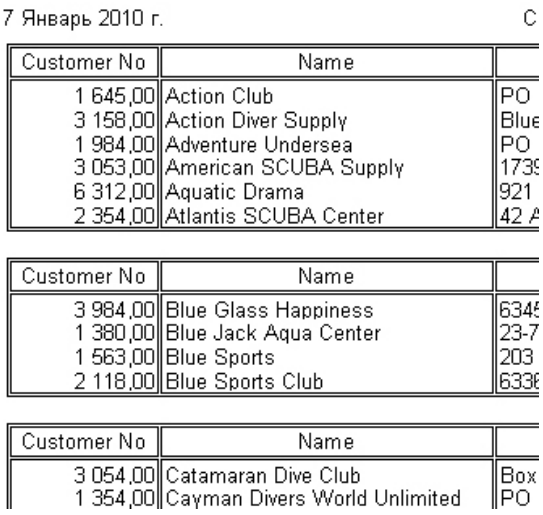
Сначала Paradox создает группу, содержащую значения поля Qty равными 0 и 1. Поскольку таких записей в таблице Stock нет, группа оказывается пустой и в сформированный отчет не выводится. Следующая группа должна иметь в поле Qty значения 2 и 3. Поскольку записей со значением поля Oty, равным 2, в таблице Stock нет, в группе оказываются только записи с Oty=3. Далее Paradox создает группу со значениями 4 и 5. Оба эти значения присутствуют в таблице Stock, и соответствующие записи включаются в отдельную группу.

Если поле, по которому группируются записи, типа дата, вы можете выбрать в качестве шага диапазона:

- Day - один день
- Week - неделя (с воскресенья по субботу)
- Month - месяц
- Quarter - квартал
- Year - год

Помните, что хронологический порядок значений учитывается при любом выбранном шаге: например, при группировании по месяцам записи для апреля 1989 г. и апреля 1990 г. окажутся в разных группах.

Если вы группируете записи по алфавитно-цифровому полю Paradox - таблицы или символьному полю dBASE-таблицы, в окошке Range Group необходимо указать количество символов, задающих группу.


7 Январь 2010 г. C

Customer No	Name	
1 645,00	Action Club	PO
3 158,00	Action Diver Supply	Blue
1 984,00	Adventure Undersea	PO
3 053,00	American SCUBA Supply	173
6 312,00	Aquatic Drama	921
2 354,00	Atlantis SCUBA Center	42 A

Customer No	Name	
3 984,00	Blue Glass Happiness	6345
1 380,00	Blue Jack Aqua Center	23-7
1 563,00	Blue Sports	203
2 118,00	Blue Sports Club	6336

Customer No	Name	
3 054,00	Catamaran Dive Club	Box
1 354,00	Cayman Divers World Unlimited	PO

Рис. 12.16 Группирование по алфавитно-цифровым диапазонам

Значение 1 заставит Paradox объединять в группы все записи, значение поля которых начинается с одной и той же буквы.

Значение 2 объединяет в группы записи с совпадающими первыми двумя буквами в выбранном поле и т д.

Пример. Группирование по алфавитно-цифровым диапазонам

Предположим, вы хотите сгруппировать клиентов по первым двум буквам названия их компании.

1. Откройте окно Report Design для проектирования нового отчета по таблице Customer.
2. Дайте команду insert / Band. Paradox откроет окно Define Group.
3. В этом окне выберите поле Name, включите опцию Range Group и введите 1 в окошко Range

Group.

4. Нажмите OK. Paradox создаст зону Group с нужными вам характеристиками (12.16).

Группирование по количеству записей

Отчет можно разбить на группы, каждая из которых состоит из заданного количества записей. Это облегчает просмотр отчета, если сортировка записей по какому-то признаку не нужна. В окне Define Group включите опцию Group By Record и введите количество записей в группе в окошко Number of Records.

Пример. Группирование по количеству записей

Предположим, вы хотите разбить таблицу Customer на группы по три записи.

1. В окне Define Group включите опцию Group By Record. В окошко Number of Records введите число 3.
2. Нажмите OK

Paradox создаст зону Group в окне Report Design При просмотре на экране или выводе отчета на печать записи будут разбиты на группы по три (рис. 12.17).

7 Январь 2010 г. C

Customer No	Name	
1 221,00	Kauai Dive Shoppe	4-97
1 231,00	Unisco	PO
1 351,00	Sight Diver	1 N

Customer No	Name	
1 354,00	Cayman Divers World Unlimited	PO
1 356,00	Tom Sawyer Diving Centre	632
1 380,00	Blue Jack Aqua Center	23-7

Customer No	Name	
1 384,00	VIP Divers Club	32 I
1 510,00	Ocean Paradise	PO
1 513,00	Fantastique Aquatica	Z32

Рис.12.17 группировка по три записи

Использование нескольких зон Group

В одном отчете можно задать несколько зон Group, при этом следует вводить новые зоны таким образом, чтобы наименьшая по количеству объединяемых записей зона располагалась внутри более крупных зон. Например, сначала следует задать зону по полю Country, а уже потом по полю City и далее, возможно, по полю Zip Code. Всегда следует начинать с логически более общей категории, а затем сужать (детализировать) отбор

Используйте две зоны, если вы хотите, например, разбить записи на группы заданной длины в пределах указанного диапазона значений поля, или в случае, когда вам надо разбить по диапазонам записи в группе, состоящей из заданного количества записей.

Если в отчете присутствует больше одной зоны Group, вы можете отбуксировать любую из этих зон в нужную вам позицию чтобы изменить последовательность группирования данных Чтобы облегчить позиционирование зоны, во время буксиров»и ее границы изображаются на экране тонкими линиями Зона Group - единственная, которую можно удалить из отчета. Для этого выберите ее и нажмите клавишу Del.

Свойства зон

Меню свойств зоны можно получить, проинспектировав любую точку зоны (в том числе и метку).

Все зоны обладают свойством RunTime / Breakable и Run Time / Shrinkable:

- Breakable означает, что зона, не помещающаяся на странице целиком, может быть продолжена на следующей
- Shrinkable разрешает Paradox удалять в зоне пустые строки, если это позволит разместить зону целиком на одной странице

Если включена координатная сетка окна Report Design (команда Show Grid), в меню свойств зоны появляется пункт Move Grid to Band, который «привязывает» сетку к верхней границе зоны.

По умолчанию свойства Shrinkable и Breakable включены. Это означает, что Paradox сначала пытается разместить зону на одной странице, сжимая ее, и лишь потерпев неудачу, продолжает печатать зону на следующей странице.

Изменение порядка вывода верхних колонтитулов

По умолчанию Paradox выводит заглавие (содержимое верхней части зоны Report) выше верхнего колонтитула страницы, задаваемого в верхней части зоны Раде. Этот порядок можно изменить, проинспектировав зону Report и выключив опцию Precede Page Header. После этого Paradox будет печатать заглавие отчета под колонтитулом страницы. (В окне Report Design это изменение никак не отразится, поскольку зоны сами по себе не перемещались.)

Вы также можете подавить вывод колонтитулов страницы (верхнего и нижнего либо одного из них) на первой странице отчета. По умолчанию свойство Print on 1st Page зоны Page включено для каждого колонтитула, но вы можете его выключить.

Свойства зоны Group

Проинспектировав зону Group, можно изменить определение группы, установить способ изображения заголовков, порядок сортировки внутри группы и условия вывода в отчет некоторых объектов.

Для изменения определения группы используется пункт меню зоны Define Group. Paradox выводит на экран список всех полей, по значениям которых можно сформировать группу. Выберите нужное поле чтобы просмотреть или изменить текущее определение группы в диалоговом окне Define Group.

- Если вы хотите, чтобы заголовки групп печатались и в начале группы, и в начале каждой страницы (если часть группы перенесена на следующую страницу), включите опцию Heading / Page and Group.
- Если вы хотите, чтобы заголовки групп печатались только в начале группы, включите опцию Heading / Group Only.
- Пункт меню Sort Order служит для определения порядка сортировки записей внутри групп:
    - Ascending - по возрастанию значений
    - Descending - по убыванию значений
- Пункт Sort Order недоступен в меню группы, объединяемой по заданному количеству записей.

Объекты, находящиеся в заголовке (верхней части зоны), обладают свойством Conditional. Опция этого свойства позволяет вам указывать условия, при которых объект выводится в отчет:

- Print at Group & Page: объект выводится в отчет в начале каждой группы и в начале каждой страницы, независимо от того, переносится ли часть группы на следующую страницу.
- Print Only at Group: объект выводится в отчет в начале каждой группы, но не в верхнем колонтитуле страницы (кроме случая, когда группа начинается с новой страницы)
- Print Only at Page: объект выводится в колонтитуле той страницы, на которую переносится часть группы. (Такой объект никогда не выводится на первой странице отчета.)

Использование sidebar

Sidebar представляет собой поле вдоль левой границы окна Report Design, которое служит для:

- Визуального отображения выбранной зоны
- Установки разделителей страницы (page break)

Установка разделителя страниц

Чтобы установить разделитель страниц, необходимо щелкнуть мышью на sidebar. При этом Paradox установит специальный символ (маркер) на sidebar и начертит горизонтальную линию, пересекающую чертеж отчета. На рисунке 12.18 изображен маркер разделения страниц в зоне Group отчета по таблице Customer.

Позицию разделителя страницы можно изменить буксировкой маркера no sidebar. При установке разделителя придерживайтесь следующих правил:

- Разделитель страниц устанавливается в любой зоне кроме Page.
- Линия разделения на страницы не может пересекать ни один объект зоны.
- Чтобы удалить разделитель страниц, нажмите мышью маркер и отбуксируйте его за пределы sidebar.

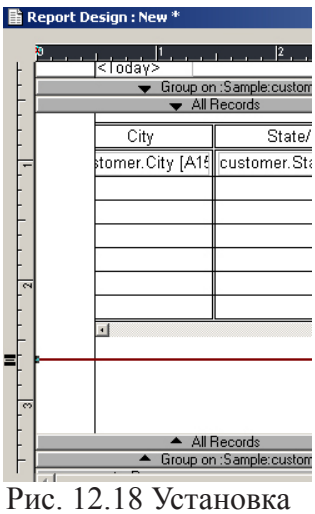


Рис. 12.18 Установка разделителя страницы

Просмотр отчета

В окне Report Design вы работаете с объектами, не содержащими данных. Поля, таблицы, графические и многозаписные объекты заполняются информацией только при формировании отчета. Отчет можно сформировать либо для вывода на печать, либо для предварительного просмотра на экране.

Предварительный просмотр отчета позволяет определить, как он будет выглядеть на бумаге и, при необходимости, отредактировать его. Ниже в этой главе (раздел «Run Time-свойства «) содержится подробная информация об управлении поведением объектов при формировании отчета. Для просмотра отчета достаточно щелкнуть мышью кнопку View Data на SpeedBar, либо нажать клавишу F8, либо дать команду Report / Preview. Во время подготовки данных Paradox выводит на экран сообщение «Preparing Report». Просмотр отчета производится в окне Report (работа в окне Report описана в Главе 4).

Просмотрев отчет, вы можете определить, что необходимо в нем изменить, и затем для внесения изменений в чертеж отчета вернуться в окно Report Design, щелкнув кнопку Design на SpeedBar, нажав клавишу F8 или дав команду Report (Design.

Объекты, использующиеся в отчетах

В Главе 10 было рассказано, как с помощью инструментов SpeedBar создавать и определять объекты в окнах разработки документов. В каждом из окон объекты размещаются и определяются одинаково, но обладают разными наборами свойств. В данном разделе рассматриваются свойства объектов, уникальные для окна Report Design.

В отчетах Paradox вы можете использовать все объекты кроме кнопок и кросстаблиц. Ниже приведены основные отличия свойств объектов в отчетах и формах:

- Объекты в отчетах не имеют свойства Methods, что не позволяв»! присоединять к объекту методы, написанные на ObjectPAL
- Run Time-свойства объектов в отчетах сложнее, чем в формах, что позволяет более точно контролировать взаимодействие объектов при заполнении их данными.
- Во время формирования отчета объекты в нем не могут использовать линейки прокрутки: эти линейки доступны в окне Report Design только при проектировании отчета и исчезают при формировании отчета, когда объекты расширяются при заполнении их данными.

В таблице 12.7 в алфавитном порядке перечислены свойства объектов, используемых в отчетах, описано значение каждого свойства и показано, к каким объектам оно применимо.

Таблица 12.7. Свойства объектов, используемых в отчетах

Свойство	Описание	Типы объектов
Alignment	Выравнивает текст по левой границе, по правой, по обеим границам либо по центру	Текст, поле
Color	Изменяет цвет выбранного объекта или его части	Прямоугольник, линия, овал,текст, поле, график, многозаписный, таблица
Define Field	Связывает поле таблицы с поле-объектом	поле
Define Graph	Связывает данные таблицы с объектом типа график	график
Define Graphic	Связывает данные таблицы с графическим объектом	рисунок
Define OLE	Помещает OLE -значение в OLE-объект	OLE поле
Define Record	Определяет табличный или многозаписный объект	многозаписный, таблица
Define Table	Связывает данные таблицы с табличной сеткой	таблица
Delete When Empty	Не изображает объект, если он не содержит данных	многозаписный, таблица

Design Sizing / Fixed Size	Фиксирует размер текстового объекта вне зависимости от объема содержащегося текста	текст
Design Sizing / Fit Text	Текстовый объект автоматически изменяет свои размеры в зависимости от объема текста	Текст
Design Sizing / Grow Only	Текстовый объект может автоматически только расширяться	текст
Design / Contain Objects	Все объекты, находящиеся внутри границ данного объекта являются вложенными в него	Все объекты внутри границ
Design / Pin Horizontal	Фиксирует объект на чертеже в его текущей позиции по горизонтали	Все объекты
Design / Pin Vertical	Фиксирует объект на чертеже в его текущей позиции по вертикали	Все объекты
Design / Size To Fit	Позволяет объекту изменять свои размеры в зависимости от объема данных	Таблица, поле, рисунок, OLE
Detach Header	Отделяет заголовок табличной сетки от самой таблицы	таблица
Display Type / Check Box	Изображает значения поля в виде кнопки-включателя	поле
Display Type / Drop-Down Edit	Изображает значения поля в виде раскрывающегося списка	поле
Display Type / Labeled	Изображает поле-объект с текстовой меткой	поле
Display Type / List	Изображает значения поля в виде списка	Поле
Display Type / Unlabeled (Edit)	Изображает поле-объект без текстовой метки	поле
Font	Выбирает начертание, стиль, цвет и размер шрифта	Текст, поле
Format / Date Format	Изменяет формат отображения поля типа дата	поле
Format / Logical Format	Изменяет формат отображения dBASE-логического поля	поле
Format / Number Format	Изменяет формат отображения числового поля	поле
Format / Time Format	Изменяет формат отображения поля, содержащего значение времени суток	Поле
Format / Timestamp Format	Изменяет формат отображения поля, содержащего одновременно дату и время суток	поле
Frame	Изменяет стиль, цвет и толщину рамки объекта	Прямоугольник, текст, график, рисунок, OLE, поле, многозаписный
Graph Type	Выбор типа графика	График
Grid	Изменяет цвет и стиль линий табличной сетки, позволяет отображать линии, разделяющие записи	таблица
Horizontal Scroll Bar	Отображает под объектом горизонтальную линейку прокрутки	Таблица, рисунок, OLE
Line	Изменяет стиль, толщину или цвет линии	Эллипс
Line Ends	Позволяет изображать стрелки на одном или обоих концах линии	линия
Line Spacing	Изменяет межстрочный интервал текста	текст
Line Style	комбинированная	Линия, эллипс
Line Type	Выбор прямой или кривой линии	линия

Magnification	Масштабирует объект внутри его рамки»контейнера»	Рисунок, OLE
Options	Позволяет форматировать график-задавать: оси, сетку, метки, легенду и заголовок	график
Pattern	Выбирает тип штриховки объекта	Эллипс, текст, график, поле, таблица, прямоугольник, многозаписный
Raster Operation	Изменяет способ взаимодействия графического изображения с объектами, находящимися под ним	рисунок
Record Layout	Изменяет способ расположения и количество повторяющихся областей многозаписного объекта	многозаписный
Repeat Header	Задает количество записей по горизонтали и вертикали в многозаписном объекте	многозаписный
Run Time / Breakable	Если объект не помещается на данной странице, его продолжение переносится на следующую	Текст, поле, таблица, многозаписный, прямоугольник, линия
Run Time / Field Squeeze	Если внутрь текстового объекта вставлен поле-объект, текст справа от поля может сдвигаться в зависимости от длины поле-объекта	текст
Run Time / Fit Height	Объект изменяет свою высоту в зависимости от объема данных	Текст, поле, прямоугольник, эллипс
Run Time / Fit Width	Объект изменяет свою ширину в зависимости от объема данных	Текст, поле, прямоугольник, эллипс
Run Time / Invisible	Объект становится невидимым в сформированном отчете	Эллипс, линия
Run Time / Line Squeeze	Если внутрь текстового объекта вставлен поле-объект, строка, его содержащая, удаляется, если в поле находится пустое значение	текст
Run Time / Orphan/ Widow	Запрещает вывод одиночных строк текста в нижней и верхней части страницы	Текст
Run Time / Pin Horizontal	Фиксирует объект по горизонтали при формировании отчета	Все объекты
Run Time / Pin Vertical	Фиксирует объект по горизонтали при формировании отчета	Все объекты
Run Time /Show All Columns	Объект может изменить размеры, чтобы вывести все колонки таблицы	таблица
Run Time / Show All Records	Объект может изменить размеры, чтобы вывести все записи таблицы	Таблица, многозаписный
Run Time / Shrinkable	Часть пустых строк объекта может быть удалена, чтобы объект поместился на странице целиком	Прямоугольник, эллипс
Search Text	Производит поиск и замену символов в тексте	Текст
Thickness	Устанавливает толщину линии или рамки	Линия, текст, график, поле, прямоугольник, многозаписный, рисунок, OLE
Vertical Scroll Bar	Изображает справа от объекта вертикальную линейку прокрутки	OLE, текст, рисунок
Word Wrap	Автоматически переносит данные на новую строку при достижении правой границы объекта	текст

Многие свойства объектов в окнах Form Design и Report Design действуют совершенно одинаково. Эти свойства и правила их применения были описаны в Главе 10.

Текстовые объекты

В основном, правила работы с текстом в отчете те же, что и в форме. В текстовых объектах можно устанавливать шрифт, выравнивание по границам объекта, межстрочные интервалы и автоматическую подстройку размеров объекта под объем текста. О форматировании текста было подробно рассказано в Главе 11.

Работая в окне Report Design, вы можете присоединить к текстовому объекту линейку вертикальной прокрутки. Линейка позволяет ввести в текстовый объект любое количество информации, не изменяя размеров объекта.

При формировании отчета Paradox может увеличить размер текстового объекта по вертикали так, чтобы отобразить все его содержимое, что иногда вызывает смещение расположенных ниже объектов. Однако, этот эффект вы можете контролировать с помощью Run Time - свойств объектов (см. раздел «Run Time-свойства» ниже в этой главе).

Редактирование текстового объекта

Работая в окне Report Design, вы можете вводить в объект текст и редактировать его. (При просмотре отчета редактирование невозможно). Чтобы отредактировать существующий текстовый объект, выберите его и еще раз щелкните его мышью: внутри объекта появится текстовый курсор.

Если вы предпочитаете работать с клавиатурой, выберите объект с помощью клавиши Tab и нажмите клавишу Spacebar (пробел). Клавиши управления курсором, Backspace и Del в текстовых объектах действуют так же, как при редактировании полей таблиц или форм. Кроме того, можно пользоваться командами меню Edit и кнопками на SpeedBar.

Paradox сохраняет данные, введенные в текстовый объект, только при сохранении проекта отчета.

Вставка поле-объекта внутрь текстового объекта

Paradox позволяет размещать в тексте поле-объекты: просто при вводе или редактировании нажмите клавишу F5 и Paradox вставит в текст неопределенный поле-объект без метки. После этого вы можете определить его по вашему усмотрению. При формировании отчета Paradox извлекает текстовое значение поля таблицы и вставляет его в указанную позицию в объекте, а последующий текст, соответственно, смещается.

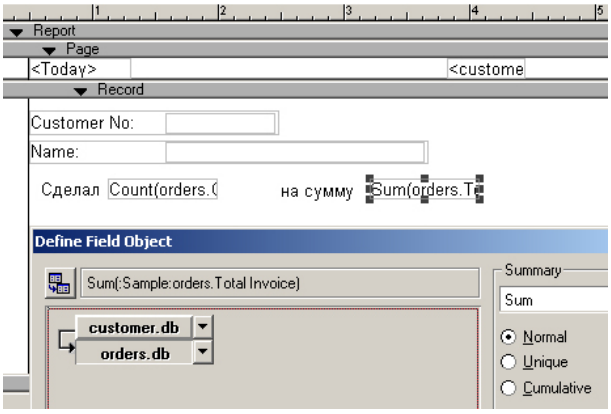


Рис. 12.19 Вставка поля в текст (макет)

Пример. Вставка поля в текст

Предположим, вы хотите получить в отчете предложение Сделал X заказов на сумму $X. В основе отчета лежит модель данных Customer -*Orders.

- Поместите поле-объект в зону Record отчета. Определите его полем CUSTOMER.DB:Customer No.
- Аналогичным образом разместите CUSTOMER. DB:Name.
- Разместите в зоне Record текстовый объект, введите в него Сделал и нажмите клавишу Spacebar для образования пробела между текстом и значением поля, а затем вставьте неопределенное поле.
- Снова поставьте пробел между значением поля

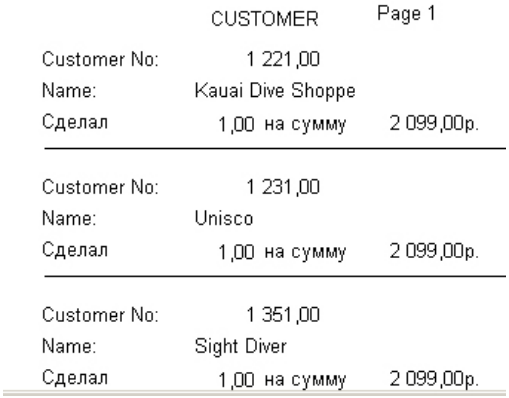


Рис. 12.20 Вставка поля в текст (отчет)

и последующим текстом, введите заказов на сумму и нажмите F5 для размещения еще одного неопределенного поля. Определите его.

- Проинспектируйте первый поле-объект и выражением CountfORDERS.Order No).
- Второй поле-объект определите выражением SumfORDERS. Total Invoice).
- При формировании отчета Paradox смещает окружающий поля текст, а размеры полей устанавливает в соответствии с объемом находящейся в них информации (рис. 12.19).

Графические и OLE-объекты

В основном, работа с графическими и OLE-объектами в отчетах и формах одинакова. В обоих окнах проектирования используются одни и те же методы масштабирования объекта, выбора рамки, применения растровых операций (для графических изображений) и определения объектов (подробности описаны в Главе 10).

Когда вы определяете в окне Report Design графический или OLE-объект, содержащая его рамка-контейнер автоматически подстраивает свои размеры под размер объекта, так как по умолчанию включено свойство Design / Size to Fit. Чтобы иметь возможность изменять объекта, это свойство следует выключить.

Если вы сделали размеры контейнера меньше размеров графического или OLE-объекта, то можете присоединить к объекту линейки горизонтальной и вертикальной прокрутки.

При просмотре или выводе отчета на печать свойство Run Time / Size To Fit определяет, будут ли границы контейнера подстраиваться под размеры содержимого или останутся фиксированными, отображая только часть графического или OLE-объекта.

Поле-объекты

В основном, работа с пол е-объектам и в отчетах и формах одинакова. Для изменения цвета, штриховки, рамки, формата, шрифта и при определении объекта используются одни и те же приемы (подробности описаны в Главе 11).

При размещении поле - объектов в многотабличных отчетах необходимо придерживаться следующих правил:

- Поле-объект, определенный полем главной таблицы, может размещаться в любой зоне отчета
- Поле-объект, определенный полем связанной таблицы, может размещаться только в повторяющейся области вывода данных связанной таблицы (в табличной сетке или в многозаписном объекте).

Способы изображения поле-объектов

При выборе способа отображения поле-объекта пункт Drop-Down Edit недоступен, так как при формировании отчета (как для просмотра, так и для печати) нельзя изменять содержимое поля. Наряду со стандартными способами изображения (поле-объект с меткой и без метки) Paradox допускает использование в отчетах списков, набора радио¬кнопок и переключателей, поскольку каждый из этих способов допускает просмотр текущего значения поля, но не предусматривает редактирования.

Статистические поля

Под статистиками полями понимаются результаты специальных вычислений, производимых над заданным множеством значений в таблице. Используя статистические поля, можно суммировать, вычислять среднее значение или количество значений, содержащихся в поле таблицы. Можно определить максимальное и минимальное значение поля, а также стандартное отклонение и дисперсию значений поля.

Статистические поля определяются в диалоговом окне Define Field Object (рис. 12.23) в поле Summary .

Функция	Результат
Sum	Сумма непустых значений
Count	Количество непустых значений
Min	Минимальное значение
Max	Максимальное значение
Std	Стандартное отклонение значений
Var	Дисперсия значений
Avg	Среднее значение
First	Первое значение
Last	Последнее значение
Prev	Предыдущее значение

Статистические вычисления производятся над определенным набором записей. Этот набор зависит как от модели данных, так и от расположения статистического поля на чертеже отчёта.

При размещении статистических полей помните о следующих правилах:

- Статистические поля, находящиеся в верхней и нижней частях одной зоны, дают один и тот же результат. Это означает, что вы можете с равным успехом располагать статистические поля, например, в начале и в конце отчета или страницы.
- Если статистическое поле находится в зоне Report, вычисления производятся над всеми записями таблицы.
- Если статистическое поле расположено в зоне Page, вычисления производятся над записями, находящимися на данной странице.
- Если статистическое поле расположено в зоне Group, вычисления производятся над записями, входящими в данную группу.
- Если статистическое поле расположено в зоне Record, то вычисления зависят от следующих ситуаций:
- Если в отчете нег зоны Group, вычисления производятся над всеми записями таблицы.
- Если зона Group присутствует в отчете, вычисления производятся над записями, принадлежащими данной группе.
- Если отчет табличного или многозаписного типа и свойство Run Time| Show All Records многозаписного объекта или табличной сетки выключено, вычисления производятся над находящимися в них записями. В этом случае таблица или многозаписный объект действуют как зона Group, группирующая по количеству записей.

Статистические вычисления рекомендуется производить над ключевыми полями таблицы, поскольку в них отсутствуют пустые значения.

Пример. Использование статистического поля в однотабличном отчете

Предположим, вы хотите узнать количество ваших клиентов в каждой стране.

1. Создайте зону Group по значениям поля Country таблицы
2. Customer.
3. Включите инструмент Field на SpeedBar. Разместите поле - объект в зоне Group ниже поля Country.
4. Принспектируйте поле-объект. На экран будет выведено диалоговое окно Define Field Object.
5. В раскрывающемся списке полей таблицы Customer выберите
6. поле Customer No.
7. В раскрывающемся списке Summary выберите функцию Count. В окошке в верхней части окна появится надпись Count(CUSTOMER:Customer No) (рис. 12.21).
8. Нажмите OK. в поле-объекте появится надпись ount(CUSTOMER.Customer No).
9. Напечатайте отчет.

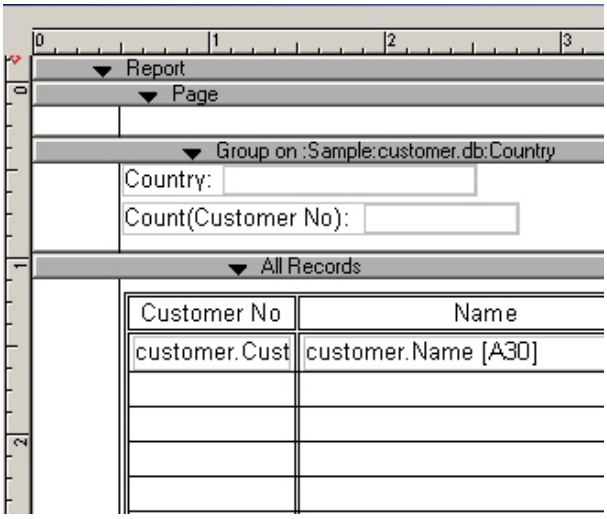


Рис. 12.21 Выбор поля и статистической функции

Для каждого неповторяющегося значения поля Country Paradox выведет в отчет название страны, количество клиентов в этой стране и таблицу с информацией о каждом клиенте (рис. 12.22).

Если статистическое поле работает с главной таблицей многотабличного отчёта, вычисления производятся над записями наиболее глубоко вложенной группы.

При работе с моделью данных типа 1-1 или 1-M Paradox объединяет две таблицы до того, как нач-

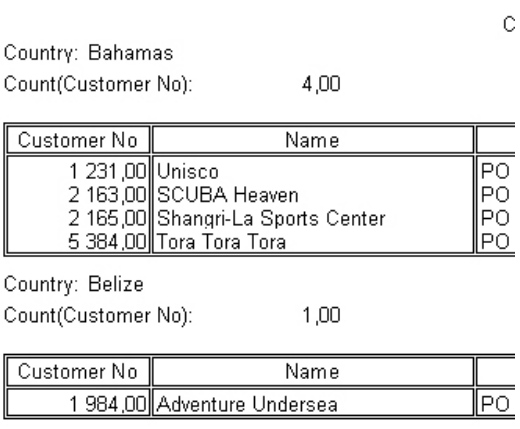


Рис. 12.22 Использование статистического поля в однотабличном отчете

нёт вычисления

Если статистическое поле размещено в зоне Record при модели данных 1-M, вычисления производятся только для текущей записи главной таблицы, которая в данном случае играет роль зоны Group, объединяющей записи связанной таблицы в группы.

При размещении статистического поля в связанной таблице многотабличного отчёта установленные выше правила для зон Record, Page и Group остаются в силе. В дополнение к ним вводятся следующие правила:

- Если статистическое поле находится в зоне Record, вычисления производятся над записями, связанными с текущей записью главной таблицы.
- Если статистическое поле находится в табличной сетке или многозаписном объекте главной таблицы, вычисления производятся по каждой записи главной таблицы.

Учтите, что, например, в модели данных Customer-* Orders-»Lineitem (Клиент-»Заказ-+Заказанные товары) нельзя вычислить сумму, на которую клиент заказал отдельных видов товаров, можно только узнать сумму покупок в каждом заказе. При статистических вычислениях Paradox может «подняться» только на один уровень в иерархии данных.

Учтите также, что статистическое поле, определенное для несвязанной таблицы в многозаписном отчете, производит суммирование по всей таблице.

С помощью диалогового окна Define Field Object вы можете указать следующие модификаторы статистических полей

- Normal: вычисления производятся для всех непустых значений, включая повторяющиеся.
- Unique: вычисления производятся для всех непустых неповторяющихся значений. При этом имейте в виду:
- Использование модификатора Unique с функциями Sum и Avg может привести к неправильным результатам, поскольку часть значений (повторяющихся) будет исключена из расчёта.
- Модификатор Unique, обычно, применяются для подсчёта уникальных значений в каком-либо множестве Например, сколько разных наименований товаров перечислено в данном счете?
- Cumulative: «накапливает» вычисленные значения. Если, например, вы определяете операцию Cumulative Sum для поля Balance Due (текущий баланс клиента), то первое значение поля будет нулевым, а затем к нему каждый раз будет прибавляться новое значение

Вычисляемые поля

Вычисляемые поля подчиняются тем же правилам, что и статистические поля. Определение вычисляемых полей в отчётах производится аналогично формам (см. Главу 11).

В вычисляемых полях вы также можете использовать статистические. Например, можно сгруппировать таблицу Orders по значениям поля Customer No, затем создать статистическое поле (Total Due), суммирующее значения поля Balance Due. Таким образом, вы будете знать баланс каждого покупателя. Пусть ваша новая политика сводится к тому, что каждый должник должен заплатить штраф размером 5 долларов. В этом случае вы можете создать вычисляемое поле, определённое формулой [Orders.Total Due]+5. Тогда при формировании отчёта Paradox подсчитает сумму по полю Balance Due для каждого клиента, а затем прибавит к ней число 5.

В статистических полях нельзя использовать вычисляемые поля (формулы типа Sum([Lineitem. Selling Price]*[Lineitem.Qty]) не допускаются).

Табличные и многозаписные объекты

В основном, правила работы с табличными и многозаписными объектами в отчетах и формах одинаковы. В этих объектах можно изменять шрифты, цвет, штриховку, рамку и параметры сетки. Кроме того, в отчетах можно использовать линейки прокрутки для просмотра данных и повторять заголовки таблиц на каждой странице.

При работе в окне Report Design к табличному объекту вы можете присоединить горизонтальную линейку прокрутки, что позволит при проектировании отчета видеть все поля таблицы.

По умолчанию Paradox удаляет линейку прокрутки при формировании отчёта и растягивает табличный объект так, чтобы отображались все его записи. При этом могут сдвинуться объекты, находящиеся ниже таблицы. Этот эффект можно контролировать с помощью Run Time-свойств объектов.

Если таблица настолько широка, что не помещается на экране, Paradox автоматически вводит горизонтальную линейку прокрутки. При формировании отчета таблица по умолчанию расширяется по горизонтали для отображения всех полей. В диалоговом окне Print File вы можете указать, что делать с данными, не помещающимися на странице по ширине. (Процессом расширения объектов можно также контролировать с помощью Run Time-свойств.)

Если таблица расположена на нескольких страницах, можно воспользоваться свойством Repeat Heading, управляющим выводом заголовка таблицы на каждой странице. По умолчанию оно включено, а при включенной опции Detach Header недоступно.

Run Time-свойства

В меню свойств каждого проектируемого объекта есть пункт Run Time, содержащий список свойств, действующих только при просмотре отчета или выводе на печать.

Набор Run Time-свойств зависит от типа инспектируемого объекта. В таблице 10.3 (Глава 10) перечислены все типы объектов и Run Time-свойства, которыми они обладают.

Фиксация положения и размеров объектов

При формировании отчета некоторые объекты (поля, таблицы, многозаписные и графические объекты) заполняются информацией. При этом они могут расширяться или сжиматься. Изменение размеров этих объектов вызывает смещение других объектов на странице.

Предположим, вы поместили в отчет поле Common Name из таблицы Bioiife и рядом с ним графический объект Graphic (рис. 12.23).

При работе в окне Report Design поле-объект размеров не изменяет, однако, при формировании отчета из-за разной длины данных в этом поле происходит смещение графического объекта Graphic (рис 12.23)

Если проинспектировать поле Graphic и включить опцию Run Time / Pin Horizontal, Paradox запретит перемещение данного объекта при изменении размеров других «объектов (рис. 12.24).

Одним из возможных последствий фиксации объекта может быть то, что часть расширяющегося объекта окажется под зафиксированным.

Это произошло потому, что объект Graphic был размещен в окне Report Design после поле-объекта Common Name. В данном случае вы можете проинспектировать поле Common Name и выбрать пункт Design / Move To Front, чтобы Paradox отображал поле Common Name поверх объекта Graphic

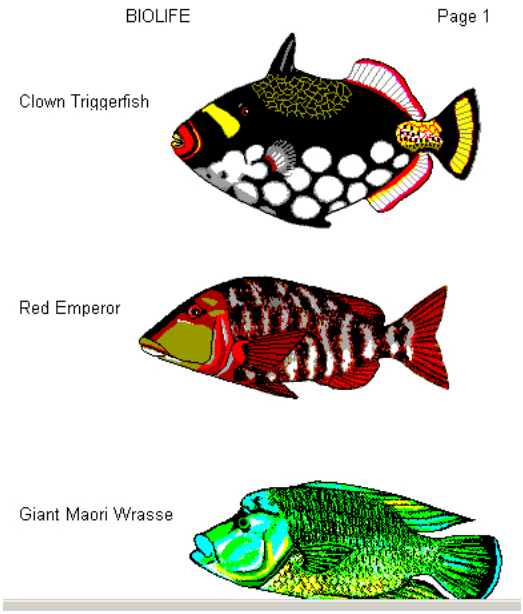


Рис. 12.23 Смещение графического объекта при формировании отчета

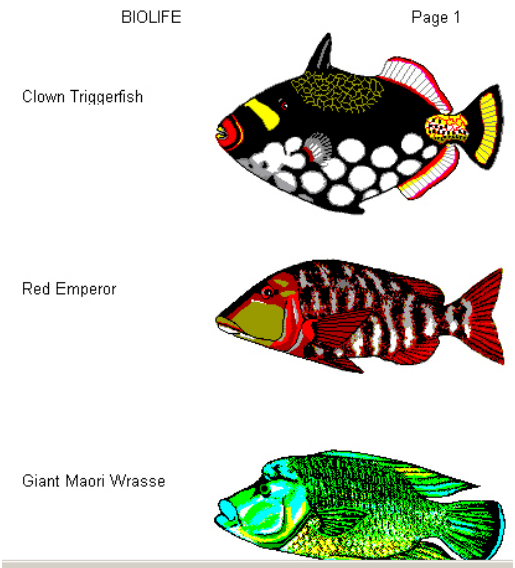


Рис. 12.24 Графический объект зафиксирован с помощью свойства Run Time - Pin Horizontal

Прямоугольники, эллипсы, текстовые и поле-объекты обладают свойствами Run Time / Fit Height и Run Time / Fit Width. Если эти свойства выключены, размеры и форма объектов при формировании отчета не изменяются, а не помещающаяся в границах объекта часть данных отсекается. Если эти сврйства включены, объект увеличивается или уменьшается в соответствии с объемом содержащихся в нем данных

Выключая свойства Run Time / Fit Height или

Run Time / Fit Width, позаботьтесь о том, чтобы объект был достаточно велик для отображения всей находящейся в нем информации.

Отображение всех столбцов и записей

свойством Run Time / Show All. Если оно включено для таблицы, Paradox увеличивает ее по вертикали, причем создает столько новых страниц, сколько необходимо для отображения всех записей. Если оно включено для многозаписного объекта, его расширение определяется опциями, установленными в окне Record Layout: если установлена опция Top-Down, Then Left-Right, Paradox создает дополнительные колонки записей, если Left-Right, Then Top-Down - дополнительные ряды записей.

Если свойство Show All Records выключено, Paradox отображает фиксированное количество записей.

Табличные объекты также обладают свойством Snow ALL Columns

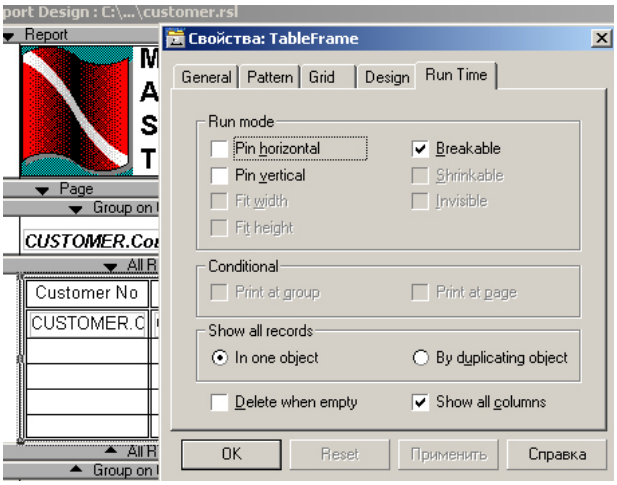


Рис. 12.25 определение свойства Snow ALL Columns

Выравнивание смещающихся объектов

Предположим, вы выровняли все объекты в окне Report Design и обнаружили, что при формировании отчета только с одним из них происходит «выталкивание». В этом случае для группирования и управления выравниванием объектов можно использовать невидимые линии или прямоугольники.

Пример. Выравнивание выталкиваемых объектов

Предположим, вы спроектировали отчет как показанный рисунке 12.26. При формировании отчета поле Vendor Name может сдвинуть прямоугольник (но не эллипс) (рис. 12.27).

Если же в окне Report Design между полем и двумя остальными объектами поместить вертикальную линию, поле при расширении будет выталкивать , в первую очередь, линию, которая затем сместит оба объекта сразу (рис. 12.28). Линию можно сделать невидимой, включив свойство Run Time / Invisible или выбрать цвет фона

Учтите, что объекты в нижней группе страницы могут быть не выровнены, поскольку вертикальная линия может оказаться слишком длинной и поэтому целиком будет перенесена на следующую страницу. Чтобы решить эту проблему, вместо линии используйте невидимый прямоугольник.

Пример. Использование расширяющегося контейнера и зафиксированной линии

Предположим, вы хотите, чтобы поля с адресом клиентом были выведены в нижней трети страницы, например, чтобы адрес был виден в прорезь на конверте.

1. Разработайте отчет на основе модели данных Customer-Orders.
2. Включите свойство Run Time / Show All Records

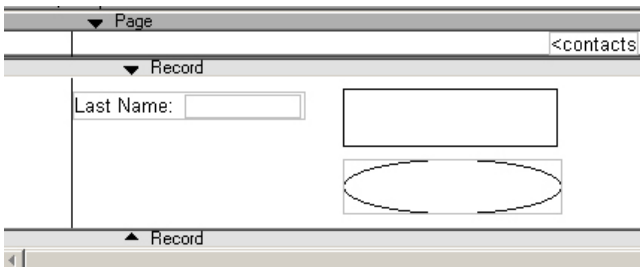


Рис. 12.26 Чертеж отчета с несколькими объектами

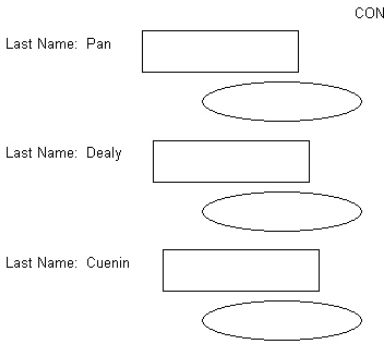


Рис. 12.27 отчет с несколькими объектами

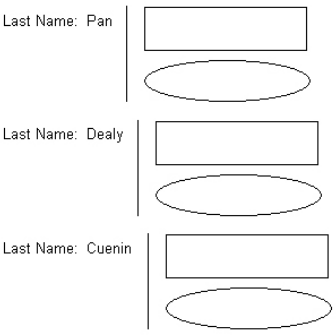


Рис. 12.28 Использование вертикальной линии для одновременного смещения

табличной сетки,
3. чтобы таблица могла быть перенесена на следующую страницу.
4. Разместите на чертеже прямоугольник, содержащий таблицу и некоторое свободное место для отделения адреса от таблицы.
5. Включите его свойства Run Time / Fit Heighf и Invisible.
6. Разместите внутри прямоугольника невидимую вертикальную линию, предотвращающую сжимание прямоугольника в случае, если в таблице окажется небольшое количество записей.

Контейнер обеспечивает необходимое количество пустых строк между таблицей и текстовым объектом с полями таблицы Customer.

Если вы используете прямоугольник-контейнер, чтобы некоторая группа объектов всегда оказывалась на одной странице, не забудьте выключить его свойство Run Time / Breakable.

Компиляция отчёта

Paradox предоставляет возможность защиты отчета от внесения в него изменений. Это полезно при распространении готового отчета среди других пользователей. Закончив проектирование отчета, вы можете его скомпилировать командой Format / Deliver в окне Report Design. При этом Paradox создает новый файл с расширением .RDL . Например, при компиляции отчета с именем CUSTORD.RSL создается файл CUSTORD.RDL.

Исходный .RSL-файл при компиляции не изменяется, и его можно использовать для дальнейшей доработки отчета в окне Report Design.

Отчет, содержащийся в .RDL-файле, можно только просматривать на экране и распечатывать на принтере. При попытке открыть .RDL-файл в окне Report Design Paradox выдаст сообщение о невозможности редактирования отчета и откроет файл в окне Report.

Вызов формы в качестве отчёта

Paradox позволяет открыть форму в качестве отчета и наоборот - отчет в качестве формы. Предположим, вами разработана удачная форма. При желании вы можете использовать форму в качестве отчета и напечатать ее. Чертеж формы используется в зоне Record отчета.

Поведение некоторых объектов в формах и в отчетах различается:

- Вычисляемые и статистические поля производят вычисления над разными множествами данных, поэтому для получения правильных результатов может понадобиться их корректировка,
- Чертеж многотабличной формы, не использующий опцию Nested (вложение объектов), неприменим в отчете: невложенные объекты становятся неопределёнными.
- При использовании многостраничной формы Paradox вводит разделители страниц в соответствующих местах зоны Record.
- В отчетах поля не могут изображаться в виде кнопок.
- В отчетах не используются кросстаблицы.
- Графики по данным главной таблицы становятся неопределенными.

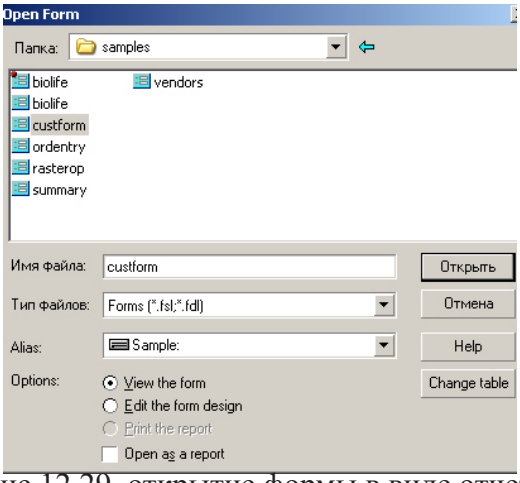


Рис.12.29 открытие формы в виде отчета

Для использования формы в качестве отчета дайте команду File / Open / Form. На экране появится окно Open Document, описанное в Главе 2. В раскрывающем списке панели Open As выберите пункт Report, затем нажмите OK, и Paradox создаст новый отчет на основе чертежа формы (форма при этом не изменяется)

Вывод отчёта на печать

Для вывода отчета на печать можно дать команды Report / Print или File / Print либо нажать кнопку Print на SpeedBar:

- Если пункт меню Report / Print или File / Print выбран из окна Report Design, вы можете напечатать чертеж отчёта (пункт Print / Design,) либо сформировать отчет по данным таблицы (пункт

Print / Report,).

- Если из окна Report выбран пункт меню Report / Print / Current Page или File / Print / Page, будет напечатана текущая страница отчета.
- При выборе пункта меню Report / Print /Reporf или File / Print /Report будет напечатан весь отчет или диапазон страниц, указанный в окне Print File Paradox открывает диалоговое окно Print File независимо от того, выбрали вы печать страницы, диапазона страниц или всего отчета.
- Опции панели Overflow Handling служат для того, чтобы указать Paradox, что делать с данными отчета, не помещающимися по ширине страницы:
- Clip To Page: Paradox отбрасывает не поместившиеся данные.
- Create Horizontal Overflow: для не поместившихся данных формируется дополнительная страница.
- Panel Vertically: для каждой страницы отчета печатается дополнительная страница независимо от того, есть ли в этом необходимость.

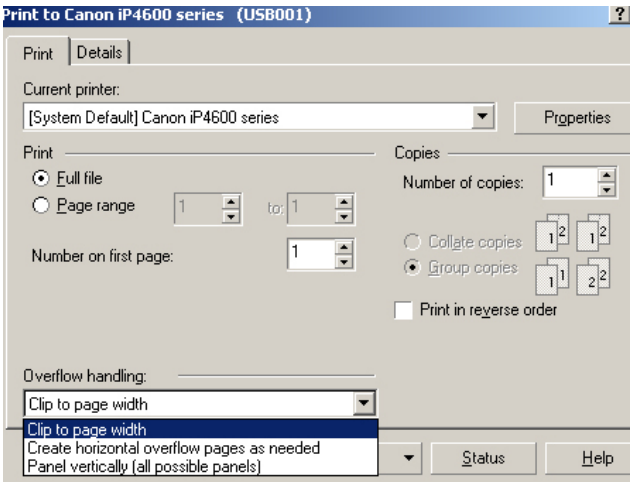


Рис. 12.30 Диалоговое окно Print File

Печать отчёта в локальной сети

Если вы являетесь сетевым пользователем базы данных, по которой формируется ваш отчет, вы рискуете получить «устаревший» документ, поскольку другие пользователи за время печати могут изменить нужные вам данные. Для контроля подобных ситуаций в окне Report Design присутствует пункт меню Format / Restart Options, с помощью которого вызывается диалоговое окно Restart Options (рис. 12.31), содержащее следующие опции:

- Restart report if data changes: вы печатаете самые последние данные, не блокируя работу других пользователей с нужными вам таблицами. Если данные в процессе печати изменились, печатание отчета начинается сначала. (Данная опция устанавливается по умолчанию для Paradox-таблиц и неприменима к dBASE-таблицам.)
- Lock tables to prevent data change, вы блокируете доступ других сетевых пользователей к нужным вам таблицам на время печати отчета.
- Lock and copy tables, run from copies: вы блокируете доступ других сетевых пользователей к нужным вам таблицам на минимальное время - пока Paradox не сделает с них временные копии, по которым и сформирует отчет. (При использовании данной опции ваш компьютер должен иметь достаточные ресурсы для создания копий таблиц.)
- Ignore data changes and continue, вы игнорируете все изменения, производимые другими пользователями базы данных, во время печати вашего отчета. (Данная опция устанавливается по умолчанию для dBASE-таблиц.)

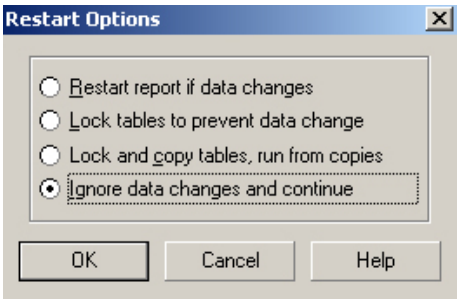


Рис. 12.31 Диалоговое окно Restart Options

Глава 13 Кросстаблицы и графики

В данной главе рассматриваются вопросы создания и применения кросстаблиц и графиков. Использование кросстаблиц и графиков позволяет сосредоточить внимание на отдельных группах табличных данных и представить их более наглядно и выразительно.

Очень часто данные табличных полей принадлежат какому-либо определенному набору значений - категорий. Кросстаблицы производят статистическую обработку данных таблицы, разбивая записи на группы по значению в одном или нескольких полях. Графики представляют эти категоризированные и статистически обработанные данные визуально. Конструируя график, не нужно предварительно создавать кросстаблицу: Paradox автоматически кросстабулирует изображаемые в виде графика данные.

Если графики можно использовать как в формах, так и в отчетах, то Кросстаблицы - только в формах. Кросстаблицы и графики создаются на основе модели данных того документа, в котором они размещаются.

Кросстаблицы

Кросстаблицы представляют собой мощный способ анализа табличных данных. Кросстаблицы статистически обрабатывают информацию из одного или нескольких полей и показывают итоговую информацию в виде таблицы формата, подобного электронным таблицам.

Кросстаблицы фактически предоставляют доступ к «скрытой» информации, содержащейся в ваших таблицах, производя следующие операции:

- Классификацию данных по одной или нескольким категориям -7
- Статистическую обработку данных внутри категорий
- Сортировку статистической информацию
- Отображение даннх в формате электронной таблицы

Одномерные кросстаблицы

На рисунке 13.1 изображена кросстаблица, созданная по полю Payment Method таблицы Orders. Кросстаблица показывает, как размещенные в вашей фирме заказы распределились по способам оплаты. В данном случае значения поля Payment Method являются категориями данных.

Paradox позволяет представлять информацию в кросстаблицах как горизонтально (рис. 13.1), так и вертикально (рис. 13.2), причем вычисления и гене-рация кросстаблицы в последнем случае производятся быстрее.

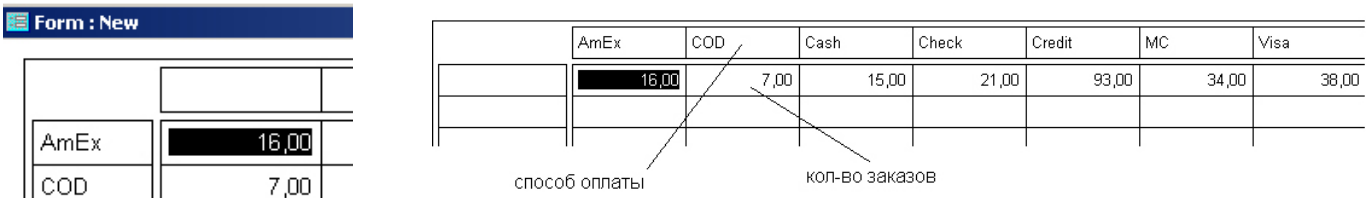


Рис. 13.1 Горизонтальная одномерная кросстаблица

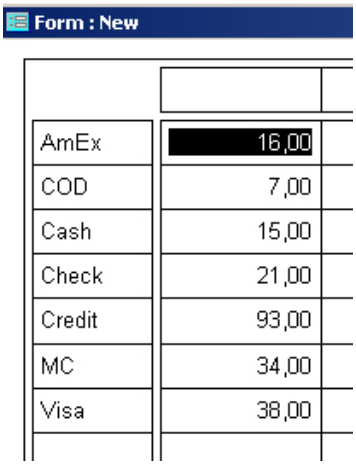
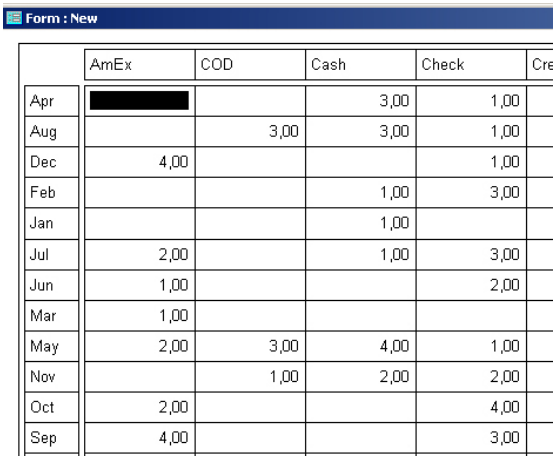


Рис. 13.2 Вертикальная одномерная кросстаблица

Двухмерные кросстаблицы

Более сложным типом кросстаблицы, группирующим информацию сразу по нескольким категориям, является двухмерная кросстаблица. Например, на рисунке 13.3 изображена кросстаблица по полям Month и Payment Method таблицы Orders, с помощью которой вы можете узнать распределение заказов по месяцам года и способам оплаты. Обратите внимание, что за-



	AmEx	COD	Cash	Check	Cre
Apr			3,00	1,00	
Aug		3,00	3,00	1,00	
Dec	4,00			1,00	
Feb			1,00	3,00	
Jan			1,00		
Jul	2,00		1,00	3,00	
Jun	1,00			2,00	
Mar	1,00				
May	2,00	3,00	4,00	1,00	
Nov		1,00	2,00	2,00	
Oct	2,00			4,00	
Sep	4,00			3,00	

рис. 13.3 двухмерная кросстаблица

головки строк и столбцов кросстаблицы отсортированы в алфавитном порядке. Такая кросстаблица может быть полезной для анализа покупательского спроса клиентов за период времени.

Многотабличные кросстаблицы

Еще одним типом кросстаблиц является многотабличная кросстаблица. Она формируется на основе информации из нескольких таблиц, которые связаны между собой однозначным отношением (связью типа один-к-одному или много-к-одному).

На рисунке 13.4 изображена кросстаблица по таблицам Stock и Vendor, показывающая количество единиц продукции того или иного класса на складе и на-именование поставщика

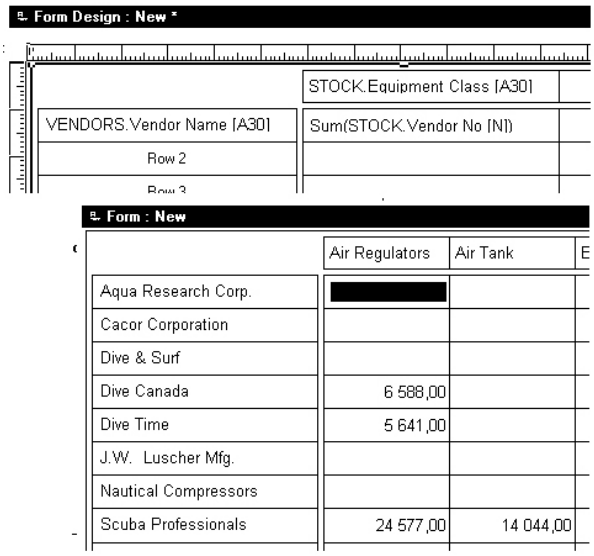


Рис. 13.4 Кросстаблица, использующая связь один-к-одному

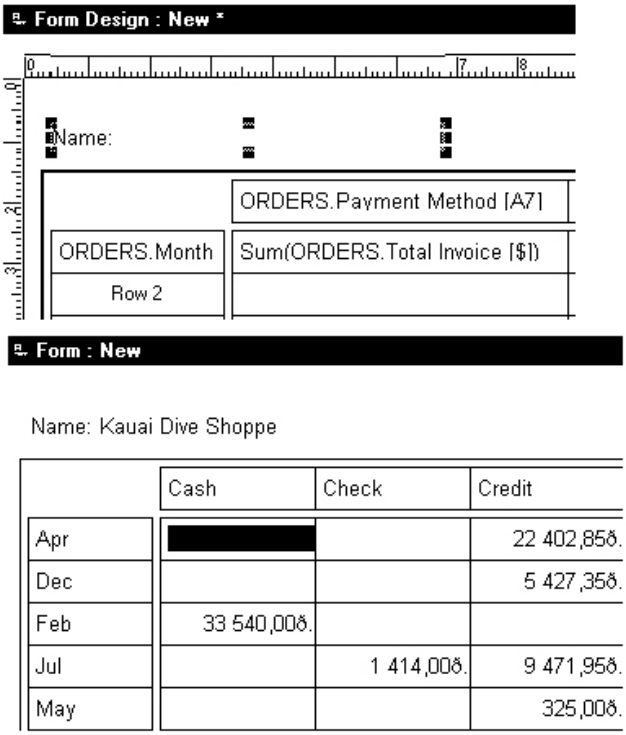


Рис. 13.5 Кросстаблица, построенная по связанной таблице

Кросстаблицы дочерних таблиц

Предположим, ваши таблицы связаны многозначным отношением (связью типа один-к-многим), и вам необходимо построить кросстаблицу по записям связанной таблицы, относящимся к текущей записи главной таблицы. Например, в своем документе вы используете таблицы Customer и Orders, связанные по полю Customer No (каждый клиент имеет несколько заказов). Поскольку эта связь явно указывается вами в модели данных документа, то Paradox «знает», что любые операции над таблицей Orders могут производиться только над теми ее записями, которые связаны с текущей записью таблицы Customer. Такая кросстаблица представлена на рисунке 13.5. В ней показано распределение сумм заказов (поле Total Invoice таблицы Orders) данного клиента по способам оплаты и по месяцам года.

Разработка кросстаблиц

Создать кросстаблицу можно одним из следующих способов:

- Открыть нужную таблицу и нажать мышью кнопку Quick Crosstab на SpeedBar или выбрать пункт меню Table / Quick Crosstab.
- С помощью инструмента Crosstab на SpeedBar окна Form Design размес-тить в форме нужной таблицы заготовку кросстаблицы.

Следует отметить, что аналогично другим инструментам на SpeedBar вы мо-жете задать исходные свойства кросстаблицьы, проинспектирова кнопку Crosstab и установив нужные свойства.

Быстрый вызов кросстаблицы

Для быстрого построения кросстаблицы произведите следующие действия:

1. Командой File / Open / Table откройте нужную таблицу.
2. Нажмите кнопку Quick Crosstab на SpeedBar или выберите пункт меню Table / Quick Crosstab.

Paradox предоставит вам диалоговое окно Define Crosstab (рис. 13.6).

3. В диалоговом окне Define Crosstab укажите поля, значения которых вы хотите использовать в качестве заголовков столбцов строк (категорий) кросстаблицы, и поля, данные которых должны быть подвергнуты статистической обработке.
4. Нажмите OK. Paradox произведет необходимые вычисления и создаст кросстаблицу в новом окне Form. Для модификации кросстаблицы в окне Form Design вы можете нажать кнопку Form Design на SpeedBar, выбрать пункт меню Form / Design

Создание кросстаблицы в окне Form Design

Для создания кросстаблицы из окна Form Design необходимо выполнить следующие действия:

1. Вызовите диалоговое окно File / New / Form.
2. Разработайте модель данных формы (см. Главу 9).
3. Нажмите кнопку OK. Появится диалоговое окно Design layout.
4. Включите опцию Blank на панели Style.
5. Нажмите кнопку OK. Появится диалоговое окно Form Design.
6. Нажмите инструмент Crosstab на SpeedBar разместите на чертеже формы заготовку кросстаблицы (рис. 13.7).

Чтобы определить структуру кросстаблицы, можно проинспектировать ее различные части: заголовки строк и столбцов, область данных, а также всю кросс-таблицу целиком (в последнем случае вызывается диалоговое окно Define Cross-tab).

Определить поля, значения которых будут использоваться в кросстаблице в качестве:

- заголовков столбцов
- заголовков строк (категорий)
- статистических данных
- Задать тип производимой статистической операции

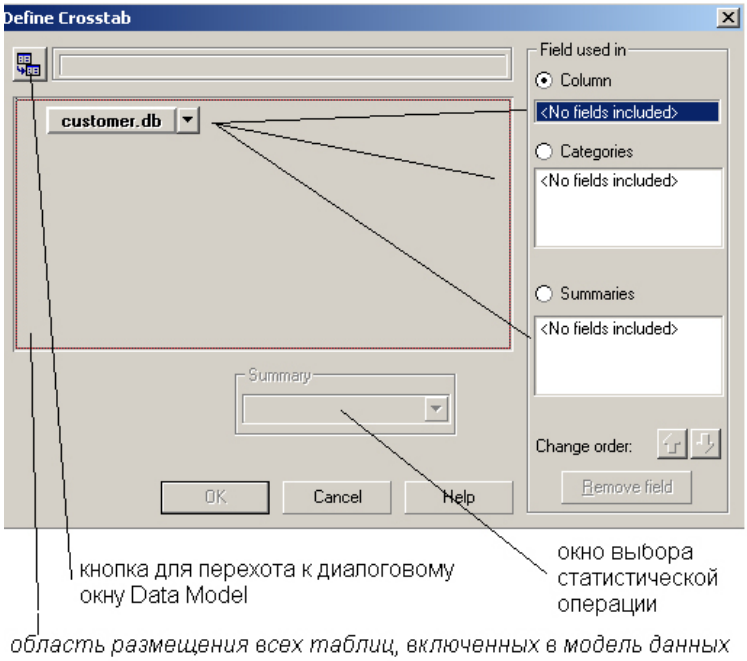


Рис. 13.6 Диалоговое окно Define Crosstab

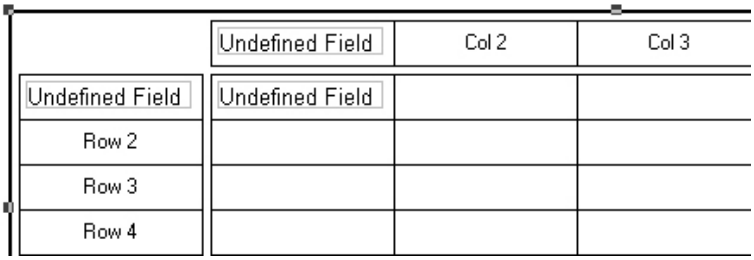


Рис.13.7 заготовка кросстаблицы

Задание заголовков столбцов

Для задания заголовков столбцов включите опцию Column на панели Field Used In и выберите из раскрывающегося списка, присоединенного к имени таблицы, нужное поле (вы можете выбрать только одно поле). Если вы создаете одномерную вертикальную кросстаблицу (как на рис. 13.1), заголовки столбцов зада-вать не нужно.

Задание заголовков строк

Для задания заголовков строк (категорий) включите опцию Categories на панели Field Used In и из раскрывающегося списка, присоединенного к имени таблицы, выберите нужное поле (поля). (Неко-

торые поля, например, уже включенные в кросстаблицу, могут быть недоступны.)

Если вы создаете одномерную горизонтальную кросстаблицу (как на рис. 13.1), задавать поля для категорий не нужно. Для двухмерной кросстаблицы необходимо задать одно поле в качестве заголовка столбцов и одно или несколько полей из доступных в качестве заголовков категорий. Поля категорий вносятся в список Categories. Порядок их следования определяет порядок сортировки статистических данных на каждом уровне категорий. Этот порядок можно изменить, отметив нужное поле и воспользовавшись кнопками-стрелками Change Order.

Например, если вы определите кросстаблицу следующим образом: поле Payment Method таблицы Orders -заголовки столбцов, поля Customer No и Month - категории, а поле Total Invoice подвергнуть статистической обработке, то получится кросстаблица, показанная на рисунке 13.9.

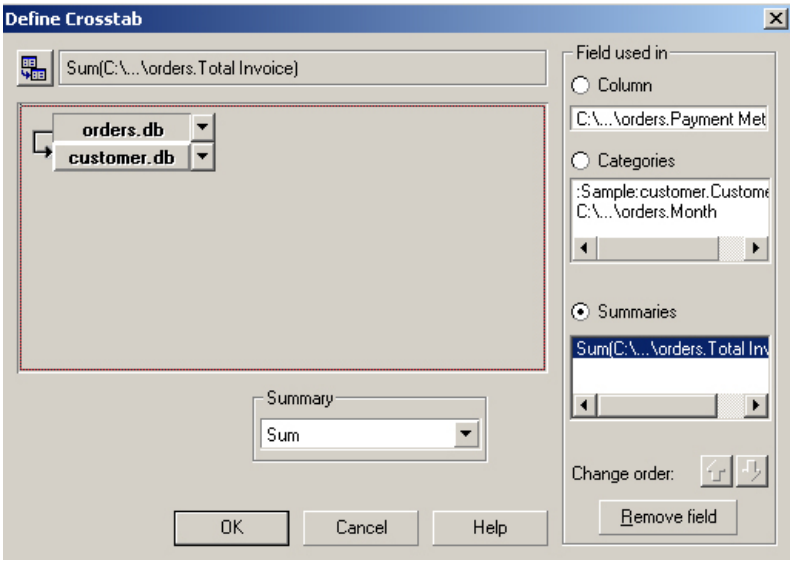


Рис. 13.8 создание заголовков

	AmEx	COD	Cash	Check	Credit	MC	Visa
Apr			106 941,00р.	3 525,00р.	197 320,75р.	78 955,50р.	37 94
Aug		5 388,80р.	51 107,70р.	18 532,00р.	30 728,25р.	9 932,05р.	40 38
	56 839,95р.			51 730,80р.	8 514,35р.	1 809,85р.	8 58
Sep	40 822,30р.			35 920,85р.	148 104,75р.	29 294,00р.	3 63
1 221,00 Dec					2 099,00р.		

Рис. 13.9 Кросстаблица по категориям Month и Customer No

Определение данных для статистической обработки

Если кросстаблица одномерная, то статистическая обработка производится по категориям того поля, которое используется в качестве заголовка столбца или строки.

Если кросстаблица двухмерная, то для определения полей, подвергаемых статистической обработке, включите опцию Summaries на панели Field Used In и выберите из раскрывающегося списка, присоединенного к имени таблицы, нужные поля. (Некоторые поля, например, уже включенные в кросстаблицу в качестве заголовков столбцов и строк, могут быть недоступны.) Вы можете выбрать несколько полей таблицы (порядок их следования определяет последовательность, в которой статистически обработанные данные появляются ячейке кросстаблицы).

Обратите внимание, что одно и то же поле может быть включено в список Summaries несколько раз. Однако, помните, что количество полей в списке Summaries, помноженное на количество столбца кросстаблицы, не должно превышать 250.

Задание статистической операций

Порядок следования полей в списках Categories и Summaries может быть изменен с помощью кнопок-стрелок Change Order, находящихся в нижней части панели Field Used In. Отметьте поле в списке (кнопки-стрелки Change Order станут доступными) и, нажимая мышью нужную кнопку-стрелку, переместите его в нужную позицию.

Аналогичные изменения можно

Таблица 13.1 Статистические операции для полей Paradox-таблиц

	Типы полей					
Оператор	A	N	$	D	S	M
SUM		+	+		+	
COUNT	+	+	+	+	+	
MIN	+	+	+	+	+	
MAX	+	+	+	+	+	
AVG		+	+		+	

производить в окне Form Design, буксируя мышью поле - объекты.

Чтобы удалить одно или несколько полей из кросстаблицы, выберите в диалоговом окне Define Crosstab нужное поле в одном из списков: Columns, Categories или Summaries - и нажмите кнопку Remove Field.

Таблица 13.2 Статистические операции для полей dBASE-таблиц

	Типы полей				
Оператор	C	F	N	D	L
SUM		+	+		
COUNT	+	+	+	+	+
MIN	+	+	+	+	+
MAX	+	+	+	+	+
AVG		+	+		

Формирование кросстаблицы из Define Crosstab

После того как вы заполните необходимые поля диалогового окна Define Crosstab (по крайней мере одно поле для заголовка столбцов или строк) и нажмете OK, Paradox сформирует кросстаблицу. В зависимости от способа создания кросстаблицы (быстрый вызов или в окне Form Design), Paradox откроет ее либо в новом окне Form, либо в том окне Form, в котором вы разместили заготовку кросстаблицы.

Инспектирование кросстаблицы

Проинспектировать кросстаблицу как целый объект можно, щелкнув правой клавишей мыши в ее левом верхнем углу. При этом вы можете сделать следующее:

- Создать кросстаблицу по умолчанию
- Вызвать диалоговое окно Define Crosstab
- Задать формат кросстаблицы

Создание кросстаблицы по умолчанию

Чтобы создать кросстаблицу по умолчанию, необходимо выполнить следующие действия:

1. Проинспектируйте левый верхний угол заготовки кросстаблицы.
2. Выберите пункт Define Field из меню свойств кросстаблицы.
3. В предоставленном меню выберите таблицу (при многотабличной модели данных связанные таблицы будут недоступны).

Paradox автоматически ставит в соответствие поля таблицы заголовкам столбцов, категориям и статистическим данным по следующим правилам:

- Неповторяющиеся значения первого поля не BLOB-типа становятся заголовками столбцов.
- Неповторяющиеся значения второго поля не BLOB-типа становятся заголовками строк (категориями).
- Значения третьего поля не BLOB-типа подвергаются статистической обработке.
- Если в таблице всего два поля, Paradox создаёт одномерную кросстаблицу.

Диалоговое окно Define Crosstab

Возможно, вам удобнее определять структуру кросстаблицы в окне Define Crosstab, чем инспектировать каждый составляющий ее объект в окне Form Design. (Как описывалось выше, в диалоговом окне Define Crosstab можно из списка полей таблицы выбрать сразу несколько полей, тогда как из меню свойств за один раз можно выбрать только одно поле).

Для того чтобы открыть диалоговое окно Define Crosstab из окна Form Design, необходимо выполнить следующие действия:

1. Проинспектируйте левый верхний угол кросстаблицы.
2. Выберите из меню свойств пункт Define Crosstab.
3. В предоставленном списке выберите пункт (...). Paradox откроет диалоговое окно Define Crosstab.

Задание формата кросстаблицы

Помимо пункта Define Crosstab в меню кросстаблицы присутствуют пункты, служащие для ее форматирования. С их помощью вы можете нужным вам образом настроить внешний вид кросстаблицы.

Инспектирование составных частей кросстаблицы

После того как вы разместили на чертеже формы заготовку кросстаблицы, ее поле-объекты не определены и содержат фразу «Undefined Field». Чтобы связать их с конкретными полями таблиц, их нужно поочередно проинспектировать, выбрать из меню свойств пункт Define Field и указать имя нужного поля таб-лицы, чтобы открыть диалоговое окно Define Field Object (см. Главы 10,11).

Вы можете проинспектировать строку заголовков столбцов, столбец заголовков строк, область статистических данных, чтобы определить их полями таблиц или чтобы отформатировать данную область. Обратите внимание, что при выборе этих областей они не окружаются рамкой с «ручками», поскольку перемещать их нельзя (однако, вы можете изменять их размеры, буксируя соответствующие границы).

Формирование кросстаблицы

Определив структуру кросстаблицы, вы можете сформировать ее одним из следующих способов:

- Щелкните мышью кнопку View Data на SpeedBar
- Нажмите клавишу F8
- Выберите пункт меню Form / View Data

При формировании кросстаблицы Paradox выполняет запрос. Если при этом происходит ошибка (например, таблица Answer содержит слишком большое количество полей или на жестком диске вашего компьютера оказывается недостаточно свободного места), Paradox создаёт кросстаблицу с пустыми полями.

Графики

Графики Paradox обладают широкими возможностями визуального представления информации. Они позволяют быстро и достаточно просто анализировать табличные данные и замечать те особенности и взаимные зависимости данных, которые при обычном просмотре таблицы не видны. Вы можете просматривать различные виды графиков и одновременно работать с данными. При вызове графика, то Paradox сначала создаёт на основе табличных данных кросстаблицу, а затем формирует ее графическое представление. Поэтому, прежде чем приступать к рассмотрению возможностей графиков, внимательно изучите, как работают кросстаблицы.

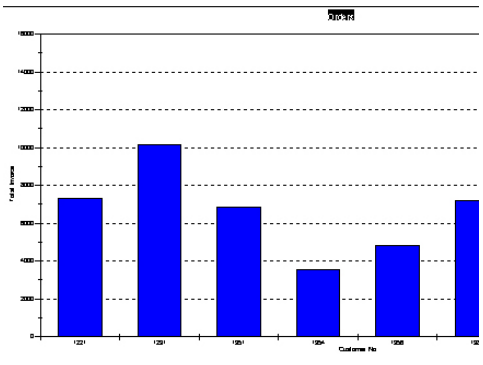


Рис. 13.10 Табулярный график

Табулярный график

Paradox по умолчанию создаёт табулярный график, производящий статистическую обработку данных одного поля таблицы по категориям, которые задаются значениями другого поля. Табулярный график - единственный тип графика, который можно использовать в отчётах. На рисунке 13.10 представлен Табулярный график, сформированный по данным таблицы Orders.

По оси X отложены значения поля Customer No в порядке возрастания но-меров, по оси Y - значения поля Total Invoice с фиксированным шагом, начиная с нуля. Высота столбцов на графике соответствует значениям поля Total Invoice для каждого клиента, представленного полем Customer No.

Одномерный статистической график

Статистическим графиком называется график, для оси Y которого задана статистическая операция (см. раздел «Задание статистической операции» выше в данной главе). Простейшим видом статистического графика является одномерный график.

На рисунке 13.11 показан одномерный статистический график, сформированный по таблице Orders. По оси X отложены значения поля Ship Via (способ транспортировки), по оси Y - суммарные значения поля Total Invoice для каждой категории Ship Via. График показывает общие суммарные расходы по каждому способу транспортной доставки грузов.

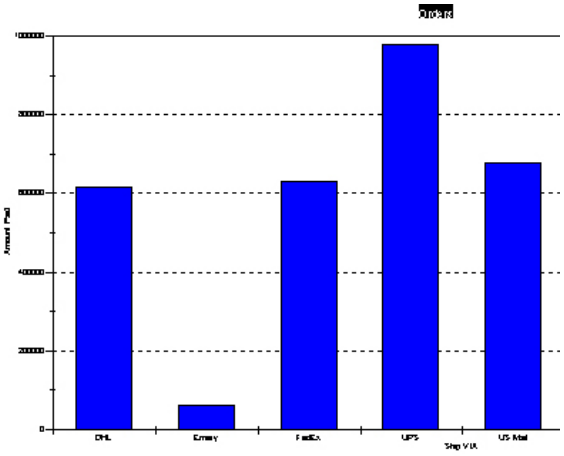


Рис. 13.11 Одномерный статистический график

Двухмерный статистический график

Двухмерный статистический график отличается от одномерного тем, что по оси X откладываются значения сразу двух полей таблицы. На рисунке 13.12 изображен двухмерный график, построенный по данным таблицы Orders. Этот график подобен графику на рисунке 13.11 за исключением того, что данные поля Total Invoice разделены дополнительно на две группы по значениям поля Terms (сроки транспортной доставки)

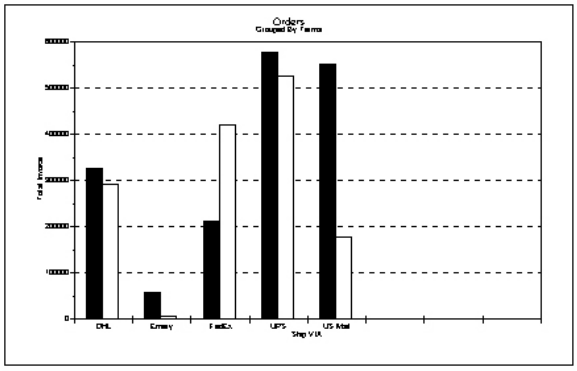


Рис. 13.12 Двухмерный статистический график

Многотабличные графики

Paradox предоставляет возможность построения графиков на основе информации из нескольких таблиц, связанных между собой однозначным отношением (см. Главу 9). Такие графики называются многотабличными. На рисунке 13.13 представлен двухмерный статистический график, построенный по данным двух связанных таблиц -Stock и Vendors. Этот график показывает, сколько единиц продукции того или иного класса имеется на складе и у какого поставщика она была куплена

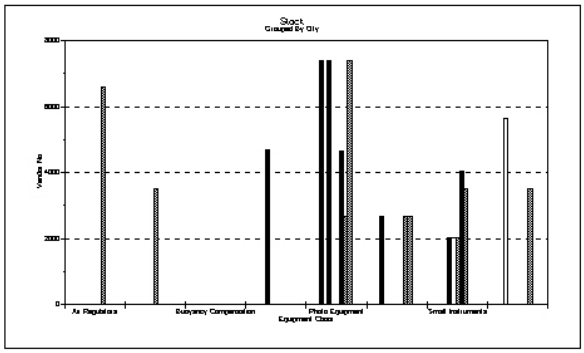


Рис. 13.13 Двухмерный статистический многотабличный график

Графики на основе данных связанных таблиц

Как и в случае кросстаблиц_ формируемых по данным связанных таблиц, графики подчиняются тем же условиям. Paradox может построить график только на основе данных из тех записей связанной таблицы, которые относятся к теку-щей записи главной таблицы

На рисунке 13.14 изображен статистический график, построенный по таблице Orders, связанной многозначным отношением с таблицей Customer. График показывает суммы, на которые сделаны заказы (поле Total Invoice - ось Y) данным клиентом (поле Order No - ось X). Переходя от записи к записи в таблице Customer, вы можете просмотреть аналогичные графики для всех клиентов.

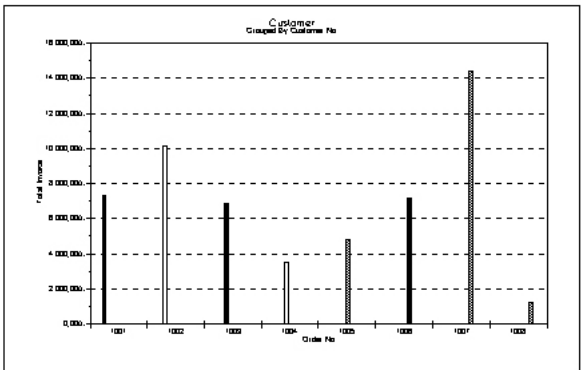


Рис. 13.14 Двухмерный статистический график, построенный по связанной таблице

Разработка графика

В отличие от кросстаблиц, Paradox предоставляет возможность создания графиков и в формах, и в отчетах (в отчет вы можете поместить только табулярные графики). Так же как и в случае кросстаблиц, Paradox предоставляет два способа построения графиков:

- Откройте нужную таблицу и нажмите мышью кнопку Quick Graph на Speed-Bar, либо нажмите комбинацию клавиш Ctrl+F7 на клавиатуре, либо выберите пункт меню Table / Quick Graph.
- Откройте новую форму или отчет по нужной таблице и с помощью инструмента Graph на SpeedBar окна Form Design или Report Design разместите на чертеже документа заготовку графика. Как и любой другой инструмент окна разработки, вы можете проинспектировать кнопку Graph на SpeedBar и задать исходные параметры всех разрабатываемых вами графиков.

Быстрый вызов графика

Для быстрого построения графика необходимо проделать следующие действия:

1. Командой File / Open / Table откройте нужную таблицу.
2. Нажмите мышью кнопку Quick Graph на SpeedBar, На экране появится диалоговое окно Define Graph (рис. 13.15).
3. В данном диалоговом окне укажите поля, значение которых вы хотите откладывать по осям X и Y либо подвергнуть статистической обработке (см. раздел «Работа в диалоговом окне Define Graph « далее в этой главе).
4. Нажмите OK. Paradox произведет необходимые вычисления и построит график в новом окне Form или Report. Для модификации графика вы можете перейти в окно Form Design или Report Design: нажав кнопку Form Design или Report Design на SpeedBar, либо выбрав пункт меню Form / Design или Report / Design, либо нажав клавишу F8.

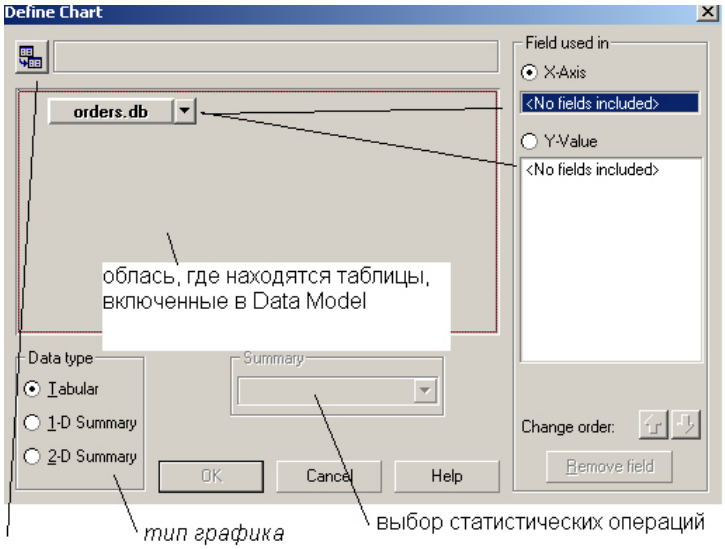


Рис. 13.15 Диалоговое окно Define Graph

Разработка графика в окнах Form Design и Report Design

Чтобы разработать график в окне Form Design или Report Design, необходимо выполнить следующие действия:

1. Командой File / New / Form или File| / New / Report вызовите диалоговое окно Data Model.
2. Включите в модель данных нужную таблицу В случае многотабличной модели данных вы должны задать взаимные связи между таблицами (см. Главу 9).
3. Нажмите OK. На экране появится диалоговое окно Design layout.
4. Включите опцию Blank на панели Style.
5. Нажмите OK. Появится окно Form Design или Report Design.
6. Включите инструмент Graph на SpeedBar и разместите на чертеже документа заготовку графика

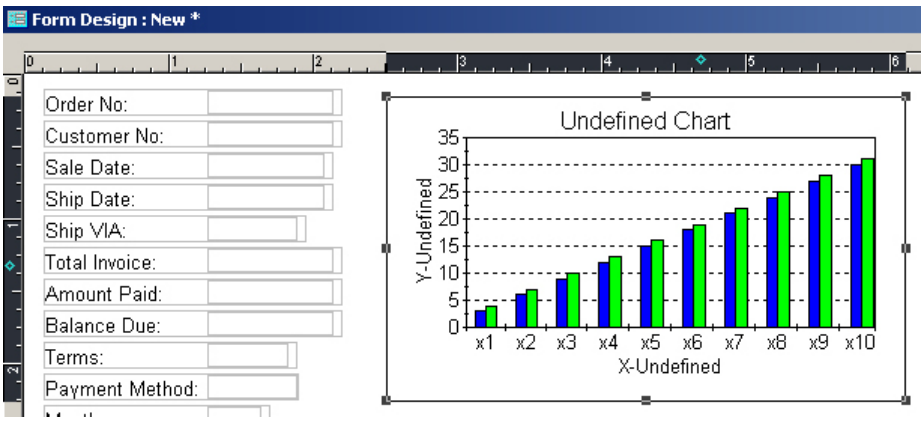


Рис. 13.16 модель графика в Form Design

(рис. 13.15).

Вы можете проинспектировать любой неопределенный объект заготовки (оси, заголовок, серии и фон), а также весь объект целиком, чтобы вызвать диалоговое окно Define Graph, котором можно определить сразу все объекты графика.

Учтите, что график можно разместить в зоне Record отчета только в том случае, если она относится к записям связанной таблицы при модели данных типа 1-»M, либо в зоне Record уже размещена табличная сетка или многозаписный объект.

Работа в диалоговом окне Define Graph

Окно для выбора структуры графика может иметь различное содержание а зависимости от места положение курсора.

Для определения структуры графика (рис. 13.17) вы должны выполнить следующие действия в диалоговом окне Define Graph:

- Выберите тип графика - табулярный, одномерный или двухмерный.
- Определите поля, значения которых будут откладываться по осям X и Y.
- Если вы выбрали двухмерный график, укажите поля таблицы, данные кото-рых должны быть подвергнуты статистической обработке.
- Задайте тип статистической операции.

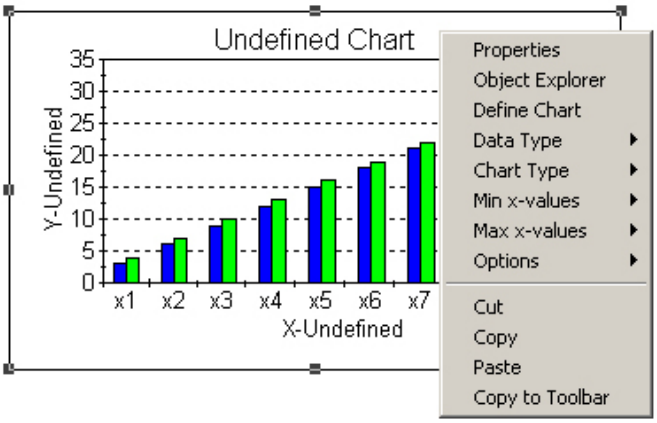


Рис. 13.17 выбор структуры графика

Выбор типа графика

Тип графика указывается на панели Date Type диалогового окна Define Graph:

1. Tabular - табулярный (устанавливается по умолчанию)
2. 1 -D Summary (одномерный статистический график)
3. 2 -D Summary (двухмерный статистический график)

Вид панели Field Used In изменяется в зависимости от типа выбранных вами данных.

Табулярный график

Если вы выбрали табулярный тип графика, то в диалоговом окне Define Graph (рис. 13.15) необходимо задать:

- Поле для оси X
- Любое количество из оставшихся полей для оси Y (данные этих полей изображаются в виде серий)

Одномерный статистический график

Если вы выбрали одномерный статистический график, то в диалоговом окне Define Graph (вид окна аналогичен изображенному на рис. 13.15) необходимо задать:

- Поле для оси X
- Любое количество из оставшихся доступных полей для оси Y, значения которых должны быть подвергнуты статистической обработке
- Статистическую операцию

Двухмерный статистический график

Если вы выбрали двухмерный статистический график, то в диалоговом окне Define Graph (рис. 13.15, 2-D Summary) необходимо задать:

- Поле для оси X
- Одно из оставшихся доступных полей для оси У, значения которого должны быть подвергнуты статистической обработке
- Еще одно поле, по значениям которого должны группироваться обрабатываемые данные Статисти-

ческую операцию.

Задание оси X

Когда вы впервые входите в диалоговое окно Define Graph, опция X-Axis уже активна. Вам остается из раскрывающегося списка, присоединенного к имени таблицы, выбрать нужное поле, данные из которого будут откладываться по оси X.

Задание оси Y

Чтобы задать значения, откладываемые по оси Y, включите опцию Y-Axis и в раскрывающемся списке, присоединенном к имени таблицы, выберите нужные поля (некоторые поля, например, ранее включенные в график, могут быть недоступны для выбора).

Если график табулярного типа, то поле, откладываемое по оси Y, может быть только числовым.

Если вы строите одномерный статистический график, то можете выбрать для оси Y любое количество доступных полей. По умолчанию Paradox производит суммирование значений числовых полей и подсчет неповторяющихся значений алфавитно-цифровых и полей типа дата. Если вы хотите изменить операцию статистической обработки, выберите одно из полей на панели Y-Value, а затем в раскрывающемся списке Summary укажите операцию, которую необходимо при-менить к данному полю. (Список доступных статистических операций для Paradox и dBASE-таблиц приведен в таблицах 13.1 и 13.2 соответственно ранее в этой главе.)

Если вы строите двухмерный статистический график, то можете выбрать для оси Y только одно из доступных полей таблицы и задать статистическую операцию для обработки его значений. Кроме того, вы можете указать еще одно поле таблицы, по значениям которого следует сгруппировать обрабатываемые данные: включите опцию Grouped By и выберите из раскрывающегося списка по-лей таблицы нужное вам (некоторые поля, например, ранее включенные в график, могут быть недоступны для выбора).

Порядок следования полей в списке Y-Value табулярного и одномерного статистического графика определяет серии графика. Вы можете изменить его, воспользовавшись кнопками-стрелками Change Order, расположенными в нижней части панели Field Used In.

Если вы хотите удалить какое-либо поле, выбранное для той или иной оси или группы графика, выберите его в соответствующем списке и нажмите кнопку Remove Field.

Инспектирование осей X и Y

В заготовке графика оси имеют метки «X-Undefined» и «У-Undefined» соответственно. Проинспектировав их, вы можете изменить поля, значения которых откладываются по осям (выбор полей зависит от типа графика), типы статистической обработки значений (для оси Y) и формат элементов оси: заголовка, масштабной разметки и сам масштаб оси.

Инспектирование серий

Серии заготовки графика изображаются в виде групп прямоугольников разного цвета. Проинспектируйте каждую серию и определите для них изображаемые поля таблицы (пункт Define Y-Value) и свой формат.

Для табулярного и одномерного статистического графика вы можете определить дополнительные серии с помощью пункта Define New Y-Value меню свойств оси Y. Для двухмерного статистического графика можно определить только одну серию.

Инспектирование заголовка графика

В заголовке заготовки графика стоит фраза «Undefined Graph». Проинспек-тировав заголовок, с помощью меню его свойств вы можете:

- Построить график по умолчанию
- Вызвать диалоговое окно Define Graph
- Для двухмерного статистического графика указать дополнительное группирующее поле
- Задать заголовок графика

- Задать подзаголовок
- Отформатировать область заголовка

Чтобы задать для двухмерного статистического графика дополнительное груп-пирующее поле, выберите пункт Define Group из меню свойств заголовка графика. (Данные, подвергаемые статистической обработке, также группируются по категориям оси X.)

Инспектирование фона

Проинспектируйте любую точку фона графика, и с помощью меню его свойств задайте параметры его изображения.

Виды графиков

В этом разделе описываются основные виды графиков и их использование с табличными данными различных типов.

Количественные графики

На рисунке 13.18 представлен трехмерный ступенчатый график, который сравнивает поля Qty (количество единиц продукции) таблиц Stock (склад) На рисунке 13.19 представлен двухмерный линейный график на основе тех же данных. Здесь видно явное несоответствие типа графика и используемой им информации. Точки связаны между собой прямыми, которые выглядят как некая не-прерывная функция кода продукции.

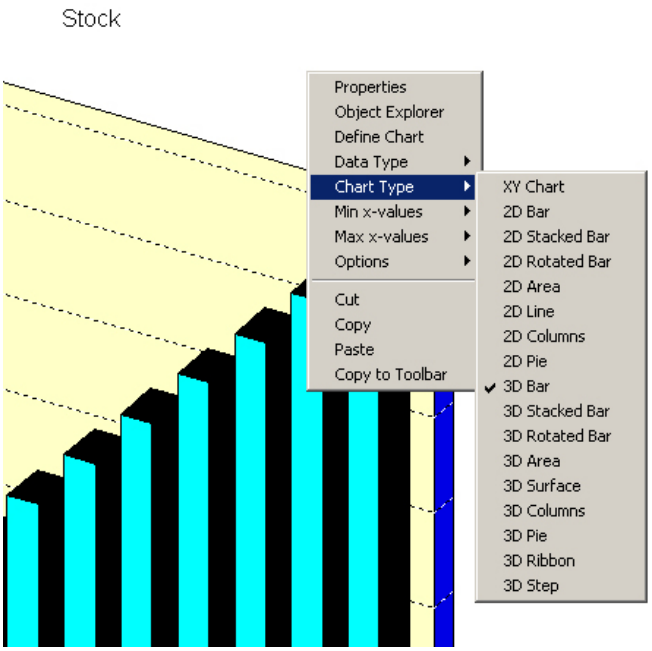


Рис.13.18 модель графика 3D bar

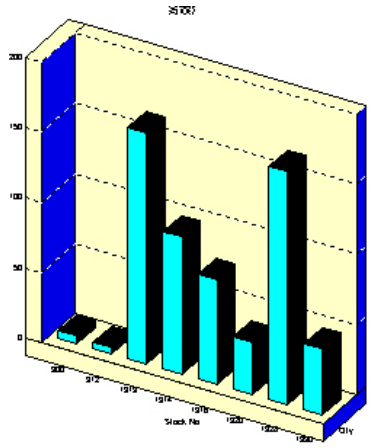


Рис. 13.18а График типа 3D Bar

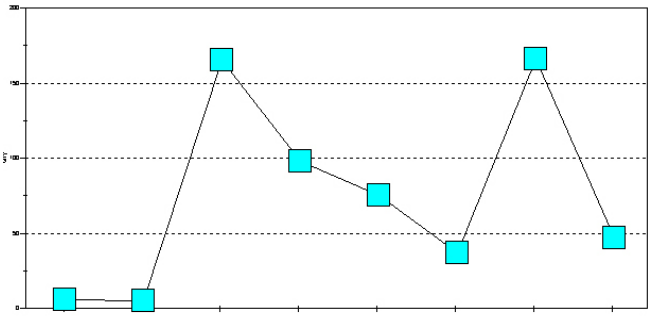


Рис. 13.19 График типа 2D

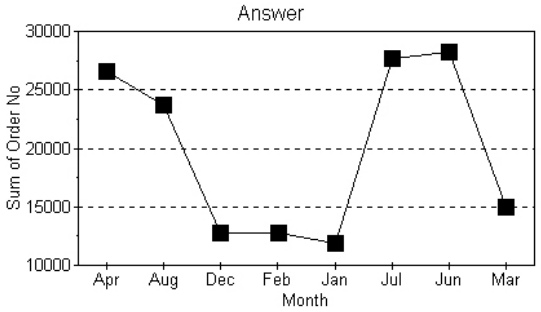


Рис. 13.20 График типа 2D Line

Линейно-временные графики

Рисунок 13.20 иллюстрирует корректное использование двухмерного линейного графика. На графике отражено распределение поступивших заказов на оборудование по месяцам. График построен на основе результатов запроса к таблице Orders. По оси X отложены значения поля Month No (номера месяцев), по оси Y -количество заказов.

На рисунке 13.21 представлен другой вид двухмерного столбцового графика, а на рисунке 13.22 - трехмерный столбцовый график для таблицы Orders. Графики показывают распределение заказов по месяцам, сгруппированные по спо-собам доставки оборудования (поле Ship Via).

На рисунке 13.22 показаны те же данные в виде областного графика.

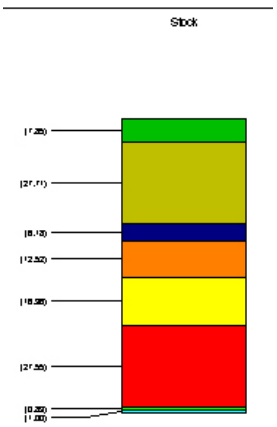


Рис. 13.21 График типа Stacked Bar

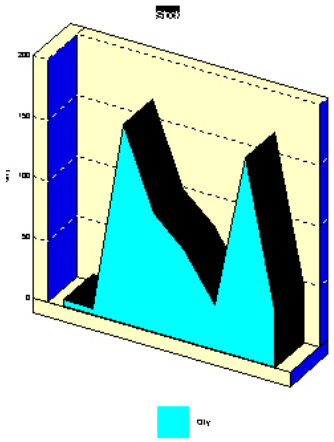


Рис 13 22 График типа 3D Area

Размещение графиков в отчетах

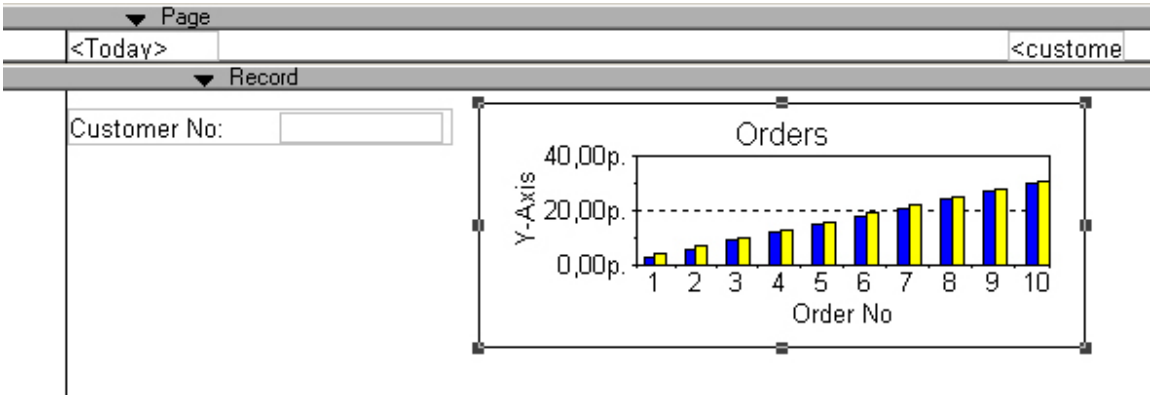


Рис. 13.23 вставка графика в отчет

Пример размещения графика в отчете иллюстрируется на рисунке 13.23. Отчет основан на модели данных типа 1-»M: таблица Customer - главная, Orders - связанная. Записи сгруппированы по полю Customer No . График таблицы Orders расположен в зоне Group и представляет заказы, сделанные каждым клиентом.

Глава 14 Обмен данными

Чрезвычайно важной и полезной чертой графической оболочки Windows является возможность динамического обмена данными между ее программами в реальном масштабе времени. Для обмена данными в Windows используются два механизма: Динамический Обмен Данными (DDE) и Связывание и Встраивание Объектов (OLE). Если DDE позволяет обмениваться данными, то OLE позволяет хранить данные из других Windows-программ и иметь непосредственный доступ к функциональным возможностям этих программ

Программа, данные которой являются источником при обмене, называется сервером, а программа, которая получает данные при обмене, называется клиентом. Paradox использует оба механизма обмена данными и способен быть DDE-клиентом, DDE-сервером и OLE-клиентом.

Использование механизма DDE позволяет поддерживать динамические связи между полями Paradox-таблиц и данными в других Windows-программах.

Использование механизма OLE позволяет встраивать в Paradox целые файлы из OLE-сервера. При этом вы имеете доступ к OLE-серверу непосредственно из Paradox и можете с его помощью производить необходимые изменения встроенных данных.

Механизм DDE

Используя DDE, вы, например, можете поместить в запрос к Paradox-таблицам данные из другой программы (сервера): запрос будет автоматически выполняться всякий раз при обновлении этих данных в программе-сервере. И, наоборот, данные из таблицы Paradox можно поместить в другую программу (клиент): все изменения этих данных внутри Paradox будут в то же время происходить и в программе-клиенте.

Использование Paradox в качестве DDE-сервера

Когда вы помещаете данные из Paradox в другие Windows-программы, то используете Paradox в качестве DDE-сервера (это возможно сделать только в окне Table).

Предположим, что в электронной таблице (например, Quattro Pro for Windows) выполняется некоторая вычислительная процедура. Значение, над которым вы хотите выполнить вычисление, находится в поле Paradox-таблицы. Скопируйте значение поля Paradox-таблицы в Clipboard Windows, а затем вставьте его в нужную ячейку электронной таблицы командой Paste Link. При этом вы не просто копируете значение, а задаете механизм динамического обмена данными (DDE), который сообщает электронной таблице, где искать значение для вычислений в ячейке.

Когда вы перемещаетесь по записям Paradox-таблицы, значение в связанной ячейке электронной таблицы изменяется в соответствии со значением, находящимся в указанном вами поле записи, которая в данный момент является текущей.

Пример. Использование Paradox в качестве DDE-сервера

Вы можете использовать в качестве DDE-клиента Excell. В качестве примера выбран Office 2007.

1. открыть Excell
2. выбрать: вставка – сводная таблица – использовать внешний источник данных – выбрать подключение – выбрать таблицу (в качестве примера, Order)

С электронными таблицами Quattro Pro, которые входят в состав WordPerfect Office механизм работы аналогичен.

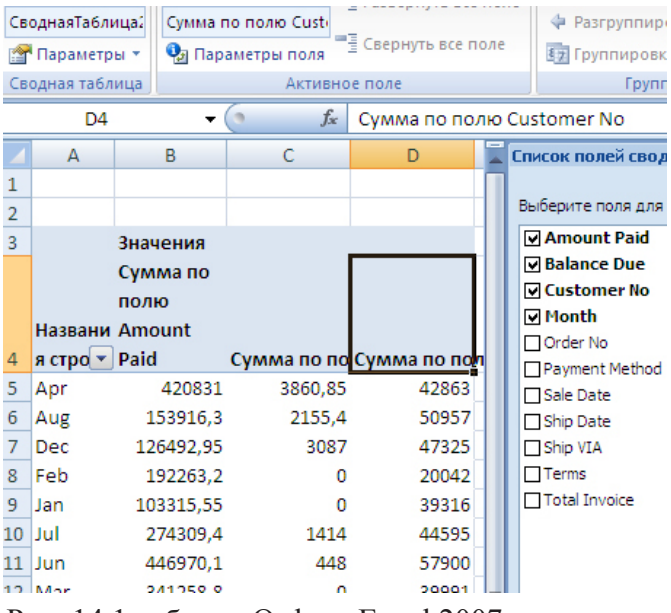


Рис. 14.1 таблица Order в Excel 2007

Отключение DDE-связей

Когда вы устанавливаете DDE-связь для Quattro Pro , выполняя команду Paste Link в DDE-клиенте, Paradox автоматически включает опцию Table / Notify On. Чтобы

отключить DDE-связь, необходимо выбрать пункт меню Table / Notify On. Выключив эту опцию, вы разрываете DDE-связь с программой-клиентом. Например, вы можете временно отключить DDE-связь, чтобы внести изменения, которые не должны отражаться в программе-клиенте, пока вы снова не включите опцию Table / Notify On.
Имейте в виду, если вы связали DDE-связью всю таблицу Paradox, данные внутри DDE-клиента будут обновляться при сохранении записи таблицы.

Использование DDE в запросах

Обычно, Paradox используется в качестве DDE-клиента тогда, когда необходимо брать данные из другой программы и на их основе производить запросы к таблицам.

Вы можете использовать таблицу Paradox в качестве DDE-сервера, а запрос - в качестве DDE-клиента. При этом каждое изменение поля таблицы, связанного DDE-связью, будет вызывать выполнение запроса и, соответственно, обновление таблицы Answer.

Пример. Использование DDE в запросах

Предположим, что вы хотите выполнить отдельный запрос для каждого клиента в таблице Customer.

1. Откройте окно Query и введите в него таблицы Orders и lineitem.
   Составьте запрос, как показано на рисунке 14.2.

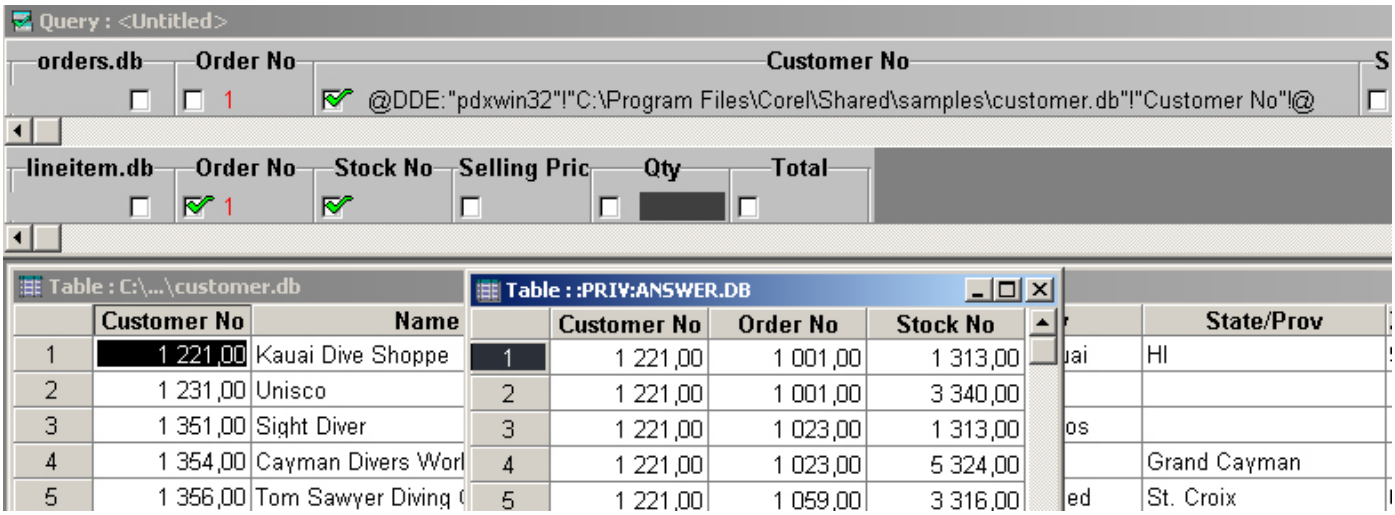


Рис. 14.2 запрос с использованием DDL связи

   Откройте таблицу Customer в окне Table.
2. В таблице Customer выберите поле Customer No с номером
3. 1221 и скопируйте его в Clipboard, нажав кнопку Copy на
4. SpeedBar
5. В окне Query установите текстовый курсор в поле Customer No
6. таблицы Orders. Дайте команду Edit / Paste special, в текущем поле
7. появится информация о DDE-связи с таблицей Customer (рис. 14.2)
8. Нажмите кнопку Run Query на Speedbar. Paradox создаст таблицу Answer со всеми значениями поля Customer No равными 1221.
9. Активируйте окно Query, выбрав заголовок окна. Включите опцию Query / Wait for DDE
10. Активизируйте окно с таблицей Customer. Выбирите поле Customer No равное 1221. Нажмите клавишу «стрелка вниз» для перехода к следующей записи со значением поля равным 1231. При этом выполнится запрос для нового значения поля Customer No

Использование Paradox в качестве DDE-клиента

Чтобы использовать Paradox в качестве DDE-клиента, вы должны поместить информацию о связи с данными из другой Windows-программы в какое-либо алфавитно-цифровое поле Paradox-таблицы. Для этого скопируйте значение из DDE-сервера в Windows Clipboard, затем выберите нужное алфавитно-цифровое поле и дайте команду Edit / Paste special.

Например, если вы связали ячейку D2 на странице A электронной таблицы Notebook 1 в Quattro Pro for Windows с алфавитно-цифровым полем таблицы Paradox, то в этом поле появится следующая строка:

©DDE-QPW! | C:\QPW\NOTEBK1 .WB1 !AD$2!@.

Значение ячейки электронной таблицы не появляется в поле таблицы Paradox. Чтобы получить доступ к нему и DDE-серверу, необходимо выбрать это поле и нажать комбинацию Shift+F2.

Механизм OLE

сли вы хотите хранить данные из других Windows-программ в таблицах и иметь доступ к их функциям непосредственно из Paradox, используйте механизм OLE. Если DDE позволяет Paradox иметь доступ к источнику данных, но хранит только указатель на источник, то OLE позволяет не только хранить, но и отображает данные из других Windows-программ. Например, вы можете поместить документ, состоящий сотен страниц, в единственное OLE-поле и просматривать его непосредственно из Paradox.

Paradox может быть только OLE-клиентом. Нельзя поместить данные из Paradox в другие программы, используя механизм OLE.

Как только вы помещаете данные из других Windows-программ в OLE-поля таблиц, форм или отчетов, вам предоставляется возможность вызова программы-сервера, с помощью которой вы можете, например, отредактировать данные OLE-поля.

Размещение OLE-данных в полях

Размещение OLE-данных в поле таблицы Paradox производится в окне Table или Form:

1. Откройте таблицу, которая имеет OLE поле
2. Стать курсором на OLE поле и включить режим редактирования
3. Выбрать режим Edit / insert / Object

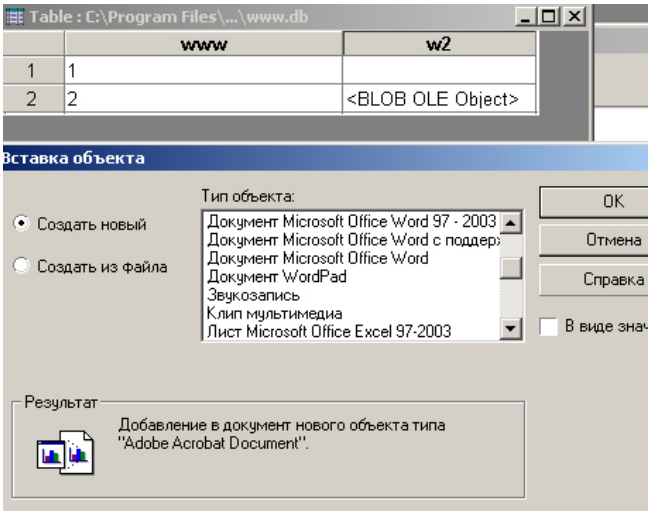


Рис.14.3 вставка объекта

Часть 2

WordPerfect Office PAL

(Paradox Application Language)

Примеры используются из реальнгых задач. Каждый из примеров работает в операционных системах от Windows 95 до Windows 7 включительно.

Введение

В этой книге описан язык приложений PAL (Paradox Application Language).На примерах из практически используемых программ показан процесс самого програм-мирования. Учитывая, что PAL не очень отличается от других языков программиро-вания для Windows, алгоритмы приведенных программ можно использовать и с дру-гими языками. Книга является дополнением и продолжением к первой части, где описан сам Paradox, который поставляется в составе редакции Professional и Education пакета WordPerfect Office 12. СУБД Paradox представляет собой удобную реляционную базу данных, которая предлагает множество способов хранения и из-влечения данных.

Отличия существующих версий практически не сказываются на самих про-граммах. Все приведенные в книге примеры работают на версиях Paradox 7 и 9, в среде win9.x, XP, 2000, windows 7. Если что-то срабатывает не правильно, то для решения проблем надо проверить настройки Windows и Borland Database Engine (для Paradox 7 - IDAPI Configuration Utility) о чем детально описано в первой части.

Основные термины, используемые в книге

Свойства - выводит окно диалога со списком объектов и их свойств. Можно выбрать название объекта, чтобы вывести свойства для данного объекта. Затем можно вставить свойство в редактируемый метод.

Константы - выводит список констант. ObjectPAL обеспечивает много констант для задания: цвета, формы указателя мыши, атрибутов пунктов меню и стиля окон. Можно выбрать требуемую константу и нажать кнопку "Вставить константу" для вставки ее в редактируемый метод.

Исходные тексты - создает и выводит на экран отчет с текстами всех исходных программ из текущей формы или отчета..

Упаковка - создает откомпилированную форму или отчет, убирая из них исходные тексты программ на языке ObjectPAL. После этого невозможно редактирова-ние пользователями оформления или исходных программ данной формы или отчета.

Отладка - Команды меню Отладка используются для отладки программы на языке ObjectPAL.

Анализ - Выводит и изменяет (не обязательно) значение переменной. Данный пункт доступен только в случае, если выполнение приостановлено в контрольной точке.

Контрольная точка - Устанавливает контрольные точки в методе для преры-вания выполнения на указанных строках. Можно устанавливать столько контрольных точек, сколько позволяет системная память. Нельзя установить контрольные точки в написанных пользователем процедурах, но по ним можно пройти с помощью команды.

Список точек - Создает список номеров строк с контрольными точками и предоставляет возможность удалять контрольные точки.

Трассировка выполнения - Открывает окно, заполняемое строками программы по мере их выполнения. Установка данного флажка сохраняется вместе с формой, поэтому нет необходимости устанавливать его каждый раз, когда нужна трассировка выполнения.

Параметры - трассировки - Выводит окно диалога со списком встроенных методов для выделенного объекта. Позволяет пометить встроенные методы, о которых по мере выполнения программы будет выводиться информация. Необходимо иметь хотя бы один объект, для которого определена программа на языке ObjectPAL.

Активен оператор DEBUG - Останавливает выполнение программы при обна-ружении конструкции DEBUG. Установка оператора DEBUG в методе действует аналогично контрольной точке, заданной для данной строки. Преимуществом является то, что эти установки сохраняются при сохранении программы, и их не нужно переустанавливать, как контрольные точки. Установка данного флажка сохраняется вместе с формой.

Разрешить выход - по Ctrl+Break Позволяет использовать комбинацию клавиш Ctrl+Break для приостановки выполнения программы и открытия окна отлад-чика, содержащего активный метод или процедуру, как при обнаружении контрольной точки. Если данный флажок не установлен, нажатие комбинации клавиш Ctrl+Break прерывает выполнение программы.

Исходный текст - Выводит в окно редактора программу другого метода.

Источник - Выводит на экран метод, содержащий текущую контрольную точку, и устанавливает курсор на строку, содержащую эту контрольную точку. Это полезно при поиске в нескольких различных методах и в нескольких окнах, когда необ-ходимо быстро вернуться в контрольную точку. Команда доступна только при ост-новке выполнения программы в контрольной точке.
Обойти - Обеспечивает пошаговое перемещение вдоль метода с обходом вы-зова процедур и функций. Команда доступна только при остановке выполнения программы в контрольной точке.
Зайти - Обеспечивает пошаговое перемещение вдоль метода, включая вызовы процедур и функций. Команда доступна только при остановке выполнения программы в контрольной точке.
Выйти из метода - Прерывает выполнение программы и закрывает все окна отладчика. Команда доступна только при остановке выполнения программы в кон-трольной точке.
Запуск - Осуществляет выход из отладчика, сохраняет изменения и осуществляет переход из режима Конструктора в режим Просмотра.
Главное окно - выводит окно диалога для установки свойств Главного окна.
Размер окна устанавливает стандартные размеры окон редактора и отладчика.
Внешний - редактор выводит окно диалога, в котором можно задать аль-тернативный редактор для написания и редактирования методов.
Выводить сообщения - определяет, выводить ли сообщения, когда компилятор обнаруживает неописанные переменные.

Объекты и методы

В Paradox все является объектом - от кнопки и поля, созданных с помощью инструментов на панели управления, и до таблиц и текстовых файлов, хранящихся на диске, до меню и всплывающих меню, создаваемых в программе. Paradox выде-ляет два вида объектов:

- объекты интерфейса - объекты, размещаемые в форме, типа кнопки
- объекты-данные - файлы, типы данных и программные структуры

Все объекты Paradox имеют свойства: цвет, шрифт, ширина строки и т.п. Методы - программа, определяющая отклик объекта на события

Объекты и методы можно модифицировать. Любой объект, который можно создавать и модифицировать в Paradox интерактивно, можно создавать и изменять и с помощью ObjectPAL

Создание приложений Paradox состоит, главным образом, в размещении объ-ектов в формах и в написании методов ObjectPAL, определяющих отклик этих объ-ектов на события. Такие приложения иногда называются приложениями "Эй ты, сделай это". Часть "Эй ты" (называемая событием) происходит, когда пользователь делает что-либо с объектом - например, указывает на кнопку в форме и нажимает кнопку мыши. Часть "сделай это" определяется методами - программой, которая за-пускается при обработке события объектом. Некоторые объекты могут выводить на экран другие объекты (такие как ссылочные списки) или переключаться на другую стадию приложения (например, на другую форму, запрос или отчет).

Объекты

Объекты языка ObjectPAL группируются по типам. Окно справочной системы по типам ObjectPAL представляет эти типы и дает пользователю возможность полу-чить справку по методам каждого типа.

Для каждого метода приводится синтаксис, описание и пример программы, которую можно скопировать и вставить в программу пользователя с помощью временного буфера.

Методы

Метод - это программа, определяющая поведение объекта. Методы определяют от-клик объекта на события. Методы ObjectPAL разделяются на три категории:

- Встроенные методы, связанные с каждым объектом Paradox
- Методы стандартной библиотеки ObjectPAL
- Специальные методы, создаваемые пользователями

События

Примерами событий являются:

- нажатие кнопки мыши
- освобождение кнопки мыши
- перемещение указателя мыши над объектом
- нажатие клавиши
- перемещение курсора в поле
- перемещение курсора за пределы поля
- выбор пункта из меню

События могут происходить и по другим причинам. Например, события таймера происходят через определенные интервалы времени. События могут также быть вызваны и из методов пользователей.

С помощью языка ObjectPAL можно создавать методы, которые определяют отклик объектов на события. Все события имеют стандартные методы для событий ObjectPAL. Нет необходимости писать методы для всех событий, которые может воспринимать объект, а события никогда не проходят незамеченными.

Встроенные методы

Каждый объект Paradox имеет встроенные методы (например, open, close и mouseUp) для каждого события, на которое он может реагировать. Эти встроенные методы задают стандартное поведение объекта в отклик на данное событие. Поль-зователь может добавить собственную программу ко встроенному методу с помо-щью редактора ObjectPAL. Для того чтобы отредактировать встроенный метод объ-екта нужно вывести меню объекта и выбрать пункт "Методы" из меню свойств. Затем следует выделить один или несколько методов в окне диалога "Методы" и нажать кнопку ОК, для того чтобы открыть одно или несколько окон редактора ObjectPAL. Можно прямо набирать текст метода в окне редактора ObjectPAL или использовать временный буфер для копирования, перемещения и вставки методов и частей методов от других объектов.

Окно диалога "Методы"

Окно диалога "Методы" используется для выбора из списка встроенных или специальных методов, или для создания нового специального метода.

Встроенные методы Выберите встроенный метод для редактирования и нажми-те кнопку ОК. Откроется окно редактора.

Нестандартные методы Выберите специальный метод для редактирования.

Новый метод Создайте новый специальный метод, введя его название в поле ввода и нажав кнопку ОК. Откроется окно редактора, в котором можно набрать текст нового метода.

Окно диалога "Методы" включает также следующие элементы:

Элемент	Для описания
Var	Переменных
Const	Констант
Type	Типов
Procs	Процедур
Users	Процедур, используемых методами объектов

Каждый из этих элементов имеет свое собственное окно редактора, открываемое при выборе данного элемента пользователем. При описании переменной, константы или процедуры в одном из этих окон, они становятся доступными для всех методов, определенных для данного объекта.

Для того чтобы получить справку по конкретному методу или процедуре, нажмите кнопку "Поиск" и введите имя метода или процедуры в окно диалога "Поиск". Можно использовать также "Список методов по алфавиту".

Библиотеки

Библиотека является коллекцией специальных методов и процедур. Библио-теки полезны для сохранения и поддерживания часто используемых программ, и для совместного использования специальных методов и переменных несколькими формами. При выборе команды File / new / Library Paradox открывает окно библиотеки.

При нажатии правой кнопки мыши в окне библиотеки, появляется окно диалога "Методы". Во многих случаях работа с библиотекой подобна работе с формой. Напри-мер, библиотека имеет встроенные методы. Можно добавлять программу в библиотеку, как и в форму, с помощью окна диалога "Методы" и редактора ObjectPAL. Как и с формой, можно открывать окна редактора для описания специальных методов, процедур, переменных, констант, типов данных и внешних программ.

Однако, существуют и некоторые важные отличия:

- В момент исполнения библиотека не выводится в окно.
- Библиотека не может содержать объекты интерфейса, она может содержать только программу.
- В библиотеке конструкции, которые используют системный указатель Self не ссылаются на библиотеку, - вместо этого они ссылаются на объект, который вызвал данный метод.

Правила видимости переменных отличаются для библиотек.

Добавление программы в библиотеку

Для того чтобы добавить программу в библиотеку, нажмите правую кнопку мыши в окне конструктора библиотеки для открытия окна диалога "Методы". Опре-делите затем программу, как и для любого другого объекта. Можно добавлять про-грамму к новой или к уже существующей библиотеке.

После завершения набора программы можно

- сохранить исходный и исполняемый код с помощью команды File / Save или File / Save as.

С помощью окна диалога "Методы" и окна редактора ObjectPAL можно добавить программу в библиотеку следующими способами:

- Определить программу для встроенных методов.
- Добавить специальные методы.
- Добавить специальные процедуры.
- Описать переменные, константы, типы данных и внешние программы.

Добавление специальных методов

Специальные методы можно добавить в уединенную программу с помощью окна диалога "Методы".

Д ля того чтобы вывести на экран окно диалога "Методы", нажмите правую кнопку мыши в окне уединенной программы или нажмите комбинацию клавиш Ctrl+Spacebar. Затем введите название нового специального метода, как и для лю-бого другого объекта, и нажмите кнопку OK, для того чтобы открыть другое окно ре-дактора.

Добавление специальных процедур

В окне диалога "Методы" выберите элемент Procs, чтобы открыть окно редактора, в котором можно описывать специальные процедуры для уединенной программы.

Описание переменных, констант, типов данных и внешних прграмм

С помощью окна диалога "Методы" можно описать переменные, константы, типы данных и внешние программы, выбрав элементы Var, Const, Type или Uses, соответственно, для того чтобы открыть подходящее окно редактора. Элементы опи-санные в этих окнах являются глобальными для программы, но к ним невозможен доступ из других форм или объектов. Однако, другие формы и объекты могут вызывать уединенную программу, которая имеет доступ к этим элементам.

На рис.1.1 показано окно выбора типа данных. Слева указаны типы, а справа те методы, в кото

рых эти типы используются.

Одним из условий написания метода (программы) является то, что сначала должны быть описаны все переменные, константы и т.д. Для этого можно использо-вать окно рис.1.1, стать на тип и нажать insert Type.

Перечень переменных и констант находится между фразами VAR… ENDVAR. Пробелы и пустые строки на содержание метода не влияют. Если в тексте встреча-ется точка с запятой (;), то справа от нее текст считается как комментарий

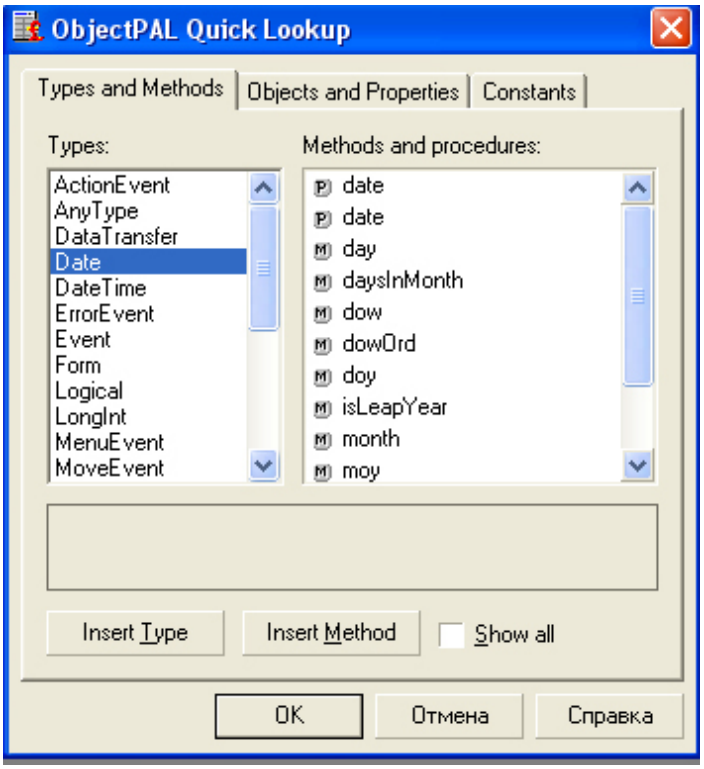


рис.1.1. Окно перечня "тип и методы"

Справочник по типам объектов

Методы и процедуры ObjectPAL охватывают шесть категорий. Каждая катего-рия в свою очередь подразделяется на несколько типов объектов. Основные эле-менты языка являются общими для всех типов.

- Объекты модели данных
- Объекты интерфейса
- Системные типы объектов
- Средства отображения
- Основные типы данных
- События
- Основные элементы языка

Уровень функциональности PAL устанавливается после загрузки Paradox вы-бором следующих режимов: Properties / ObjectPAL.

Установленный в данный момент уровень функциональности ObjectPAL определяет количество методов доступных для каждого типа. Уровень функционально-сти ObjectPAL можно изменить в окне диалога Настройки главного окна.

Примеры из справочной системы можно копировать и вставлять в программы пользователя через временный буфер.

Выберите команду Редактирование / Копировать из меню справочной системы, а затем выделите нужный блок программы и нажмите кнопку Копировать.

Поместите курсор в программу в то место, куда нужно вставить пример. Выберите команду меню Редактирование / Вставить.

Объекты модели данных :

- DataBase
- Query
- Table
- Tcursor

DataBase

Переменная типа Database является средством для работы с базой данных (т.е. с каталогом). При запуске приложения Paradox открывается стандартная база данных (рабочий каталог). В стандартной базе данных сохраняется полный адрес текущего рабочего каталога. При работе с таблицами, находящимися только в этом каталоге, нет необходимости открывать какую-либо другую базу данных. При работе с таблицами, хранящимися в других местах, следует объявить переменную типа Database и использовать конструкцию open, чтобы создать средство для работы с дру-гой базой данных. (Это

гораздо удобнее при программировании, чем задавать каж-дый раз полный путь при обращении к таблице).

Используя метод open и псевдоним (сокращенное имя), можно, как показано в следующем примере, указать, какую базу данных следует открыть:

```
var
  custInfo Database
endVar
                                        ; addAlias определен для типа Session
addAlias("CustomerInfo", "Standard", "D:\\pdoxwin\\tables\\cstdata")
custInfo.open("CustomerInfo")  ; открывается база данных CustomerInfo
                                        ; необходимо, чтобы CustomerInfo являлось правильным псевдонимом
```

В предыдущем примере Paradox получает информацию о существовании двух баз: стандартной и CustomerInfo. Переменная custInfo является указателем базы данных CustomerInfo, то есть custInfo можно использовать в конструкциях для ссылки на базу данных CustomerInfo. Предположим, например, что существуют два файла с именем ORDERS.DB (один в рабочем каталоге, а другой в CustomerInfo), и требуется узнать, являются ли эти файлы таблицами. В следующем примере ORDERS.DB проверяется сначала в рабочем каталоге, а затем, используя custInfo в качестве указателя базы данных, в каталоге CustomerInfo.

```
var
  custInfo Database
endVar
addAlias("CustomerInfo", "Standard", "D:\\pdoxwin\\tables\\cstdata")
custInfo.open("CustomerInfo")

if isTable("orders.db") then   ; проверяется ORDERS.DB в рабочем каталоге
  msgInfo("Рабочий каталог", "ORDERS.DB - таблица")
endIf

if custInfo.isTable("orders.db") then   ; custInfo используется как маркер
                                        ; базы данных CustomerInfo
  msgInfo("CustomerInfo", "ORDERS.DB - таблица")
endIf
```

Если используется open, но не задается база данных, в Paradox подразумевается, что требуется указатель стандартной базы данных. В следующем примере создается указатель стандартной базы данных, который можно передавать специ-альному методу, требующему в качестве аргумента указатель базы данных.

```
var defaultDb Database endVar
defaultDb.open()  ; открывается стандартная база данных
```

Использование указателя стандартной базы данных может сделать программу более легкой для понимания, особенно в тех случаях, когда одновременно приходится обращаться к нескольким каталогам.

Query

В языке ObjectPAL переменная типа Query соответствует запросу по образцу (QBE - Query By Example). Средствами ObjectPAL можно создавать и исполнять запросы внутри методов аналогично тому, как это делалается в интерактивном режиме. Образ запроса может храниться в файле или строке, либо размещаться непо-средственно в тексте программы.

Большинство методов для работы с запросами определены для типа Database, так как в не

которых приложениях пользователь должен указать базу данных, содержащую таблицы, по которым делается запрос.

Table

Переменная типа Table представляет собой описание таблицы. Она отличается от переменной типа TCursor, которая является указателем данных, и от объекта-таблицы и табличного окна - объектов, выводящих данные на экран.

С помощью объектов типа Table можно добавлять, копировать, создавать и индексировать таблицы, выполнять вычисления в столбцах, получать информацию о структуре таблицы и т.п., но нельзя редактировать записи. Для этих целей служат объекты типа TCursor или объекты-таблицы (объекты типа UIObject).

TCursor

Объект типа TCursor (табличный указатель) является указателем на данные в таблице, который позволяет пользователю работать с данными, не выводя таблицу на экран. Он не является аналогом или копией таблицы - редактирование записей в объекте TCursor изменяет связанную с ним таблицу, а все виды блокировки таблицы действуют и для табличного указателя.

Для получения информации об объектах, связанных с табличным указателем, следует обращаться к типам Table и TableView.

Системные типы объектов

- DDE
- FileSystem
- Library
- Session
- System
- TextStream

DDE

Динамический обмен данными (DDE) является протоколом Windows, обеспе-чивающим совместное использование данных с другими приложениями, поддержи-вающими протокол DDE. Используя методы DDE, пользователь получает доступ к данным, созданным и записанным в других приложениях. Можно также использовать DDE методы для передачи команд и данных в другие приложения.

Примечание: Paradox и ObjectPAL поддерживают еще один протокол разделения данных между приложениями, OLE.

FileSystem

Переменные типа FileSystem (файловая система) обеспечивают информацию о дисковых файлах, каталогах и дисках и доступ к ним. Переменная типа FileSystem является указателем, который может использоваться в конструкциях ObjectPAL для работы с каталогом или файлом. Во многих случаях первым шагом при работе с переменными FileSystem является проверка наличия какой-либо информации с помощью метода findFirst. Полезно помнить о такой возможности для инициализации переменных типа FileSystem

Library

Библиотека (Library) является объектом Paradox, в котором хранятся специ-альные методы, специальные процедуры, переменные, константы и типы данных, определяемые пользователем. Библиотеки полезны для записи и поддержки часто используемых программ, а также для хранения специальных методов и переменных, используемых в нескольких формах.

Во многом работа с библиотекой аналогична работе с формой. Так, например, для создания формы выбираются пункты меню File / New / Form, а для создания библиотеки File / New / Library.

Аналогично форме библиотека содержит встроенные методы. Так же, как и для формы, используя окно диалога Методы и редактор Ob-jectPAL, можно добавлять программы в библиотеку. Аналогично форме пользова-тель может открывать окна редактирования для описания специальных методов, процедур, переменных, констант, типов данных и внешних программ.

Session

Объект типа Session (сеанс) предоставляет средства для использования методов доступа к базе данных. Открытие приложения Paradox по умолчанию открывает один сеанс, но используя средства ObjectPAL можно открыть и другие сеансы; открытие других сеансов не является необходимым условием использования процедур типа Session. Возможное число дополнительно открываемых сеансов зависит от системной среды. Открытию каждого сеанса соответствует увеличение на единицу зарегистрированного количества пользователей.

Только один сеанс может проходить в интерактивном режиме с использованием средств Paradox. Все остальные сеансы должны управляться инструкциями ObjectPAL.

Блокировки, накладываемые инструкциями ObjectPAL, взаимодействуют на равных условиях с блокировками, накладываемыми при интерактивном взаимодей-ствии в том же сеансе

System

Для типа System определены процедуры вывода сообщений, определения параметров системы пользователя, работы с окном поиска файлов, справочной системой и т.д..

TextStream

Объект типа TextStream является последовательностью символов, прочитан-ной (или записанной) в текстовом файле. TextStream содержит только символы ANSI; информация о форматировании, такая как шрифт, выравнивание и отступы не учитывается. Однако, неотображаемые символы, такие как возврат каретки и новая строка (CR/LF), учитываются.

Paradox поддерживает указатель позиции в файле, который действует анало-гично курсору в текстовом процессоре. Этот указатель показывает, как далеко (на сколько символов) он находится от начала файла. Подсчет начинается с единицы (а не с нуля, как в некоторых других языках программирования).

Объекты интерфейса

- Menu
- PopupMenu
- UIObject

Menu

Объект типа Menu представляет собой список элементов в строке меню при-ложения. При выборе пользователем пункта меню возвращается текст данного пунк-та. Меню, создаваемые с помощью ObjectPAL, полностью заменяют встроенные ме-ню Paradox (однако, пользователь может вернуть последние с помощью removeMe-nu).

По умолчанию, меню не сохраняется при переходе от формы к форме; каждая форма имеет связанную с ней систему меню. Если пользователь создает меню для формы, данное меню появляется только тогда, когда форма является активной. Если при этом открывается вторая форма, она использует встроенные меню, а не меню, созданное для первой формы. Если для каждой формы создается специальное меню, пользователь может создать в приложении систему меню, зависимую от контекста.

Если требуется выводить одно и то же специальное меню для двух или более форм, следует задать для свойства StandardMenu каждой формы значение Off. В результате Paradox будет сохранять текущее меню, если пользователь переходит с одной формы на другую. Свойство StandardMenu может быть использовано для создания единой системы меню для всего приложения.

Примечание: Обычно в приложениях используются одновременно объекты типа Menu и PopUpMenu (всплывающее меню).

PopupMenu

Всплывающее меню (PopUpMenu) представляет из себя список элементов, расположенных в вертикальном порядке, появляющийся в ответ на событие (обычно нажатие кнопки мыши). При выборе пользователем пункта всплывающего меню, текст данного пункта передается в метод.
Всплывающее меню отличается от обычного меню (тип Menu), то есть списка элементов, выводящегося горизонтально в строке меню приложения. Но тип PopUpMenu наследует также ряд методов, определенных для типа Menu.

Примечание: Выбор пункта всплывающего меню не инициирует встроенный метод menuAction, за исключением случаев, когда всплывающее меню связано с обычным меню.
Используя методы типа PopUpMenu, пользователь может:

- Создать всплывающее меню
- Вывести всплывающее меню на экран и возвратить выбранный элемент
- Просматривать пункты всплывающего меню
- Обеспечить доступ к меню с клавиатуры

UIObject

Объекты типа UIObject (UI - сокращение для User Interface) создают пользова-тельский интерфейс для приложения: все, что пользователь может поместить в форму, является объектом интерфейса. Только объекты интерфейса имеют встро-енные методы. К объектам интерфейса относятся изображения, рамки, кнопки, перекрестные таблицы, эллипсы, объекты-поля, формы, диаграммы, линии, объекты - наборы записей, объекты OLE, страницы, объекты-записи, объекты-таблицы и поля ввода.

Примечание: Свойства формы аналогичны свойствам объекта интерфейса: она имеет встроенные методы, для которых можно определять дополнительный код, и может реагировать на события. Существует также отдельный тип Form для мето-дов и процедур, которые работают только с формами.

Многие методы объектов интерфейса дублируют методы табличного указате-ля TCursor. Методы объектов интерфейса, которые работают с таблицами, взаимо-действуют с соответствующей таблицей через видимый объект. Результаты дейст-вий с объектом интерфейса, затрагивающие таблицу, становятся сразу видны в объекте, с которым связана таблица. Методы табличного указателя, напротив, работают с таблицей как бы за сценой; действия, влияющие на таблицу не обязательно видны в каком-либо объекте, даже в тех случаях, когда табличный указатель установлен на той же таблице, с которой связан видимый объект.

Некоторые операции с таблицами выполняются значительно быстрее с табличным указателем, чем с объектом интерфейса. Например, если необходимо вы-полнить таблично-ориентированную операцию, вызывающую большое число обновлений экрана (что требует много времени), можно использовать следующий метод: табличный указатель описывается и присоединяется к объекту, уже связанному с таблицей (например, к объекту-таблице), операция выполняется с табличным указателем, после чего экранный объект вновь синхронизируется с табличным указателем. При присоединении табличного указателя к объекту, связанному с таблицей, указатель встает на текущую запись для этого объекта. После выполнения операций с табличным указателем, например, поиска записей, табличный указатель может оказаться установленным на записи, отличной от текущей записи объекта. Для того, чтобы установить указатель объекта на ту же запись, на которой расположен табличный указатель, применяется метод resync или moveToRecord; а чтобы заставить табличный указатель встать на ту же запись, что и объект, используется метод attach. См. пример для метода insertRecord в данном разделе.

Средства отображения.

- Application
- Form
- Report
- TableView

Application

Переменная типа Application является указателем для работы с главным ок-ном в текущем приложении Paradox. Переменные типа Application могут быть использованы в программе пользователя для управления размером, положением и видом главного окна приложения.

Хотя одновременно могут исполняться несколько приложений, объекты типа Application не могут устанавливать связь или взаимодействовать. Переменные типа Application относятся только к текущему главному окну Paradox. Можно, однако, использовать переменные типа Session для открытия множественных каналов доступа к данным (см. тип Session).

Поскольку существует только одно текущее приложение, для создания указателя приложения нужно только объявить переменную типа Application. Пока эта переменная доступна, она служит в качестве указателя приложения: с ее помощью пользователь получает доступ к методам типа Application. Например, в следующем примере переменная thisApp объявляется и используется в коде метода.

```
; downSize::pushButton
method pushButton(var eventInfo Event)
var
  thisApp      Application
  x, y, w, h   LongInt
endVar
thisApp.getPosition(x,y,w,h)          ; текущее положение окна
thisApp.getPosition(x,y,w*0.9,h*0.9)   ; сжатие окна на 10%
endMethod
```

Все методы типа Application определены для типа Form.

Form

Переменная типа Form обеспечивает средство для работы с формами Paradox. Методы, определенные для типа Form, предоставляют пользователю следующие возможности:

- Загружать форму в окно конструктора и сохранять результаты разработки
- Открывать и закрывать форму
- Подключаться к открытой форме
- Добавлять и удалять таблицы из модели данных формы
- Создавать списки имен объектов, свойств и исходных программ для методов
- Определять и изменять положение окна формы, а также уменьшать и увеличивать его размеры
- Передавать форме события, такие как mouseUp или keyPhysical
- Получать и определять методы для формы

Report

Переменная типа Report является указателем отчета. Переменные данного типа используются в программах для работы с отчетами на экране, для просмотра и печати отчетов. Методы, определенные для типа Report позволяют управлять раз-мерами, позицией и появлением окна.

Для загрузки файла отчета в окно конструктора отчетов следует использовать метод load; для открытия отчета в окне отчетов используется метод open, и для открытия и печати отчета используется print. Нельзя определять методы для объектов, содержащихся в отчете.

Тип Report наследует также методы, определенные для типа Form.

TableView

Объект типа TableView выводит данные таблицы в свое собственное окно. Этот объект отличается от объекта-таблицы, который является содержащимся в форме объектом интерфейса (UIObject), и от объекта типа TCursor, являющегося указателем данных в таблице.

Когда описывается переменная типа TableView, а затем с использованием этой переменной открывается таблица, создается указатель на окно таблицы, кото-рый можно использовать в программе

для управления окном таблицы.

Методы типа TableView являются подмножеством методов типа Form. Можно использовать их для управления размером, положением и появлением окна таблицы. Хотя для окна таблицы можно запустить и завершить режим редактирования, нельзя напрямую использовать язык ObjectPAL для редактирования данных в окне таблицы. Можно использовать ObjectPAL для управления свойствами объекта типа TableView в трех основных областях:

- Окно таблицы в целом - например, основной цвет, стиль сетки, число записей.
- Данные на уровне полей таблицы - например, шрифт, цвет, формат вывода и значение текущей записи.
- Заголовок окна таблицы - например, шрифт, цвет и выравнивание.

Тип TableView наследует ряд методов, определенных для типа Form.

События

- ActionEvent
- ErrorEvent
- Event
- KeyEvent
- MenuEvent
- MouseEvent
- MoveEvent
- StatusEvent
- TimerEvent
- ValueEvent

ActionEvent

Событие типа ActionEvent возникает, в основном, при редактировании или перемещении по таблице. Тип ActionEvent имеет ряд методов, определенных для типа Event.

Единственным встроенным методом, инициируемым событием ActionEvent, является метод action.

Обычно при обработке событий типа ActionEvent используются константы действия языка ObjectPAL. Так например, для запрета редактирования таблицы пользователями можно использовать программу, подобную нижеследующей:

```
; this TableFrame::action
method action(var eventInfo ActionEvent)
; при попытке перейти в режим редактирования выводится окно диалога
if eventInfo.id() = DataBeginEdit then  ; DataBeginEdit это константа;
  messageStop("Стоп", "Эту таблицу редактировать нельзя!"
  eventInfo.setErrorCode(1)
endif
endMethod
```

Список констант действия приводится в диалоговом окне Константы. Для вывода списка на экран следует открыть окно редактора ObjectPAL и выбрать пункты меню Язык ObjectPAL|Константы. Затем из колонки Типы констант следует выбрать пункт, начинающийся с Action, например, ActionDataCommands. Список констант появится в колонке Константы.

Дополнительная информация и примеры в Главе 6 книги "Язык Object PAL. Руководство программиста".

Тип ActionEvent наследует несколько методов, определенных для типа Event.

actionClass
id
setId

ErrorEvent

Тип ErrorEvent содержит методы, которые могут использоваться для получе-ния и для передачи информации об ошибках, возникающих при исполнении программ ObjectPAL. Тип ErrorEvent наследует несколько методов, определенных для типа Event.

Единственным встроенным методом, инициируемым событием ErrorEvent, является error.

Event

Тип Event является базовым для событий. Другие типы событий (например, ActionEvent) являются производными от данного типа. Многие из методов, обсужающихся в данном разделе, используются другими типами событий.

Event инициирует следующие встроенные методы: open, close, setFocus, removeFocus, newValue и pushButton.

KeyEvent

Объект типа KeyEvent принимает и выдает информацию о событиях клавиатуры. События клавиатуры запускают следующие встроенные методы: keyChar и keyPhysical.

Тип KeyEvent наследует методы, определенные для типа Event.

MenuEvent

Переменная типа MenuEvent содержит данные, относящиеся к выбору пунктов меню в строке меню приложения. При выборе пункта меню пользователем инициируется метод menuAction. Модифицируя встроенный метод menuAction, пользователь определяет реакцию объекта.

Тип MenuEvent наследует ряд методов, определенных для типа Event.

MouseEvent

Объект типа MouseEvent отвечает на вопросы о действиях с мышью, включая следующие:

- Где находится указатель мыши?
- Была ли нажата кнопка мыши?
- Какая кнопка мыши нажималась или удерживалась при операции?

Тип MouseEvent наследует ряд методов, определенных для типа Event. События мыши инициируют следующие встроенные методы: mouseClick, mouse-Down, mouseUp, mouseDouble, mouseRightUp, mouseRightDown, mouseRightDouble, mouseMove, mouseEnter и mouseExit.

MoveEvent

Методы, определенные для типа MoveEvent позволяют считывать и задавать информацию о событиях, происходящих при переходе в форме с одного объекта на другой. Тип MoveEvent наследует ряд методов, определенных для типа Event. См. раздел Event.

MoveEvent инициирует следующие встроенные методы: arrive, canArrive, canDepart и depart.

StatusEvent

Методы, определенные для типа StatusEvent управляют сообщениями, выводящимися в строку состояния главного окна Paradox. Тип StatusEvent наследует так-же ряд методов, определенных для типа Event.

Используя методы типа StatusEvent, можно, определяя программы для встроенных методов, выяснять причины выводящихся сообщений и их место на экране. Пользователь может блокировать вывод сообщений или выводить их в другую часть экрана, например, в другие окна строки состояния или в какой-либо объект (в объект-поле или текстовый файл). Можно также задавать текст выводящегося сообщения.

Для направления сообщения в разные окна строки состояния используются константы типа StatusReasons: ModeWindow1, ModeWindow2, ModeWindow3 и Sta-tusWindow.

TimerEvent

Методы типа TimerEvent (событие таймера) дают информацию, которую и-пользует встроенный метод timer каждого объекта интерфейса. Тип TimerEvent наследует ряд методов, определенных для типа Event.

Для того, чтобы указать, когда посылать события таймера объекту, используется метод setTimer типа UIObject. Затем для управления откликом объекта на остановку таймера следует использовать встроенный метод объекта timer. Для отключения таймера объекта используется метод killTimer типа UIObject. В следующем примере показано, как использовать эти методы с событиями таймера.

В данном примере предполагается, что форма содержит объектнабор записей, связанный с таблицей Customer. Контейнер записи в объекте-наборе записей называется custRecord.

Предположим, имеется программа по вводу данных пользователем, и нужно дать пользователю 60 секунд для редактирования записи. Через 60 секунд требуется предупредить пользователя. Для этого в методе action объекта custRecord проверяется каждое действие. Если это действие типа DataArriveRecord, в методе с помощью вызова killTimer останавливаются все старые таймеры и с помощью метода setTimer запускается новый таймер для объекта-записи. Когда таймер останавлива-ется, появляется сообщение, предупреждающее пользователя. Для упрощения из-менения временного интервала, в окне Const объекта custRecord определяется кон-станта:

```
; custRecord::Const
const
  alertTime = 60000 ; предупреждение через 60 секунд
endConst
```

Следующая программа определяется для метода action объекта custRecord.

```
; custRecord::action
method action(var eventInfo ActionEvent)
if eventInfo.id() = DataArriveRecord then ; если открывается новая запись
  self.killTimer()                        ; если все старые таймеры еще не
                                          ; остановлены, останавливаем их
  self.setTimer(alertTime)                ; запускаем таймер для этой записи
endIf
endMethod
```

Следующая программа определяется для метода timer объекта custRecord.

```
; custRecord::timer
method timer(var eventInfo TimerEvent)
beep()
msgInfo("Предупреждение", "Вы обрабатываете запись уже в течение" +"одной минуты.")
self.killTimer()
endMethod
```

ValueEvent

Методы ValueEvent управляют изменениями значений полей. Встроенный метод changeValue является единственным методом, который запускается событием типа ValueEvent. Встроенный метод newValue не вызывается вместе с событием типа ValueEvent; вместо этого newValue работает с событиями типа Event.

Тип ValueEvent наследует ряд методов, определенных для типа Event.

Не следует путать методы changeValue и newValue. Встроенный метод changeValue вызывается, когда предполагается изменение значения поля. changeValue дает пользователю возможность проверить значение и решить, нужно ли заносить значение этого поля. Встроенный метод newValue сообщает о том, что поле уже получило новое значение; newValue вызывается обычно постфактум (поля, определенные как кнопки и списки, ведут себя подругому). Заметим также, что встроенный метод newValue не то же самое, что метод newValue, описанный в данном разделе.

Основные типы данных

К ним относятся следующие:

- AnyType
- Array
- Binary
- Currency
- Date
- DateTime
- DynArray
- Graphic
- Logical
- LongInt
- Memo
- Number
- OLE
- Point
- Record
- SmallInt
- String
- Time

AnyType

Значение типа AnyType может принадлежать к одному из типов, перечисленных в следующей таблице.

Тип	Описание
AnyType	Любой из основных типов данных
Array	Индексированный набор данных
Binary	Данные во внутреннем машинном представлении
Currency	Данные для работы с денежными значениями
Date	Данные типа календарных дат
DateTime	Объединенные данные о дате и времени
DynArray	Динамический массив
Graphic	Графическое изображение
Logical	True или False
LongInt	Данные для представления относительно больших целочисленных значений
Memo	Тексты большого объема
Number	Числовые значения с плавающей точкой
OLE	Связь с другим приложением
Point	Информация о позиции на экране
Record	Структура, определяемая пользователем
SmallInt	Данные для представления относительно небольших целочисленных значений
String	Символьные строки
Time	Временные данные

Значение типа AnyType никогда не может иметь сложный тип, как TCursor или TextStream. Переменная типа AnyType наследует характеристики присваиваемого ей значения. То есть она приобретает свойства строки при присвоении ей значения типа String, числовой переменной при присвоении значения типа Number и т.д.

Язык ObjectPAL включает объекты-данные типа AnyType таким образом, что переменные основных типов могут использоваться в нем без предварительного описания. Предпочтительнее, однако, всегда объявлять переменные, когда это возможно.

Array

Массив содержит значения (называемые элементами) в ячейках подобно то-му, как почтовые ячейки содержат почтовые сообщения. В языке ObjectPAL сущест-вуют только одномерные массивы, подобные единственному ряду ячеек, в котором каждая ячейка содержит по одному элементу. Тип Array наследует также методы, определенные для типа AnyType.

Для использования в методах массивы должны быть описаны через указание имени, длины (числа элементов) и типа данных для элементов.

Примечание: В языке ObjectPAL элементы массива нумеруются с 1, а не с 0, как в ряде других языков.

Примечание: В языке ObjectPAL существуют также динамические массивы. См. ме-тоды и процедуры для типа DynArray.

Binary

Двоичный объект (называемый также большим двоичным объектом: BLOB - binary large object) содержит данные, которые могут быть прочитаны и проинтерпре-тированы лишь компьютером. Примером двоичного объекта может служить звуковой файл: человек не может прочитать и проинтерпретировать этот файл в том виде, как он записан, это может проделать только компьютер.

При объявлении переменной типа Binary создается дескриптор двоичного объекта. Это переменная, на которую можно ссылаться в программе пользователя и с помощью которой можно перемещать данные из файла на диске в таблицу и об-ратно или передавать данные из файла или таблицы в метод или процедуру.

Тип Binary наследует также методы, определенные для типа AnyType.

Currency

Значения типа Currency (Денежные) могут находиться в диапазоне от 3.4 * ± 10-4930 до 1.1 * ± 104930 с точностью до шести десятичных знаков. Число знаков, выводящихся на экран, определяется установками на панели управления системы Windows. На таблицы эти установки не влияют - в таблицы всегда записываются данные с шестью десятичными знаками. Тип Currency наследует также методы, оп-ределенные для типов AnyType и Number.

Date

В языке ObjectPAL значения дат могут представляться в одном из следующих форматов: месяц/день/год, день-Месяц-год или день.месяц.год. Переменные типа даты должны быть описаны в явном виде. Например, блок

```
var
  d Date
endVar
d = date("12/21/1997")
```

присваивает d значение даты 21 декабря 1997 г. Нельзя опускать кавычки во-круг значения даты, иначе ObjectPAL выполнит операцию деления.

Тип Date наследует методы, определенные для типов AnyType и DateTime.

Формат значений типа Date задается методом formatSetDateDefault (типа System) или конструкциями форматирования ObjectPAL.

Средства ObjectPAL могут быть использованы для проведения вычислений над любыми допустимыми значениями дат, однако, даты, записанные в таблицах Paradox, должны принадлежать диапазону от 1 января 100 г. до 31 декабря 9999 г.

Даты 20го века можно задавать, указывая две цифры для года, например,

```
myDay = date("11/09/59")
```

Даты для столетий со 2го по 10е должны включать три разряда для года (например, 12/17/243), а для столетий от 11го по 19е все четыре разряда для года (на-пример, 12/17/1043). Полностью опускать год недопустимо.

Тип Date наследует ряд методов, определенных для типа DateTime.

DateTime

В переменной типа DateTime данные хранятся в форме часы-минуты-секунды миллисекунды год-месяц-день. Значения типа DateTime используются только в вычислениях ObjectPAL; значения этого типа нельзя записывать в таблицы Paradox. Значения DateTime должны быть описаны в явном виде. Так, например, следующая конструкция присваивает переменной dt типа DateTime время 11 часов 10 минут и 40 секунд и дату 21 декабря 1997 г.

```
var dt DateTime endVar
dt = DateTime("11:10:40 am 12/21/97")
```

Кавычки вокруг значения являются обязательными.

В качестве разделителей могут быть использованы следующие символы: пустой символ, символ табуляции, пробел, запятая(,), дефис (-), косая черта (/), точка (.), двоеточие (:) и точка с запятой (;). Формат значений типа DateTime задается методом formatSetDateTimeDefault (типа System) или конструкциями форматирования ObjectPAL.

Значения типа DateTime должны задаваться полностью; ни одно из полей не может быть опущено, хотя для любого поля нулевые значения являются допустимыми.

См. также методы и процедуры, определенные для типов Date и Time.

Тип DateTime наследует методы, определенные для AnyType. Аналогично, оба типа Date и Time наследуют методы, определенные для DateTime.

DynArray

Тип DynArray определяет динамический массив, то есть массив с изменяемой структурой. Динамический массив является компактной запоминающей структурой для любой комбинации типов данных. DynArray позволяет организовать быстрый поиск значений даже в динамических массивах, содержащих большое число элементов.

Эти массивы называются динамическими, так как от пользователя не требуется задание их размеров; длина динамического массива автоматически изменяется при добавлении или удалении элементов. Размер динамического массива ограничен только системной памятью.

Примечание: Язык ObjectPAL поддерживает также массивы фиксированной и переменной длины.

В отличие от массива фиксированной длины индексы динамического массива не являются целыми числами. Индексами могут служить любые допустимые выражения ObjectPAL, значения которых преобразуются к типу String. Каждый индекс в динамическом массиве связывается со значением. Тип DynArray наследует также методы, определенные для типа AnyType.

Graphic

Переменная типа Graphic является указателем для манипулирования графическим объектом. Это означает, что переменные данного типа могут использоваться в программах на языке ObjectPAL для работы с графическими объектами. Графические объекты содержат и позволяют выводить на экран изображения в растровом формате (BMP). Однако, СУБД Paradox может импортировать следующие графические форматы: bitmap (расширения файлов .BMP), encapsulated Postscript (EPS), graphic interchange format (GIF), Paintbrush (PCX) и tagged information file format (TIF).

Применяя методы типа Graphic readFromClipboard, writeToClipboard, readFromFile и writeToFile, можно использовать графические переменные для перемещения изображений между формами (и отчетами), таблицами, временным буфером и дисковыми файлами.

Дополнительная информация и примеры находятся в книге "Язык ObjectPAL. Руководство программиста".

Тип Graphic наследует также методы, определенные для типа AnyType.

Logical

Логическая переменная может иметь два значения: True(Истина) или False(Ложь). Вместо True можно использовать константы ObjectPAL Yes или On, а вместо False - No или Off.

Логическая переменная занимает 1 байт памяти. Логические операторы имеют следующий порядок приоритета: NOT, AND и OR.

Логические переменные часто используются для ответа на вопросы о состоянии других объектов или результатах операций, например,

- Является ли данная таблица пустой?
- Выводится ли данная форма в виде пиктограмма?

Успешно ли создан текстовый файл в результате данной операции?

LongInt

Значения типа LongInt являются длинными целыми числами, т.е. содержат большое число разрядов. Переменная типа LongInt занимает в памяти 4 байта.

Язык ObjectPAL преобразует к типу LongInt значения в диапазоне от -2,147,483,648 до 2,147,483,647. Попытка присвоить переменной типа LongInt значение, не входящее в данный диапазон, приводит к ошибке, например,

```
var
  x, y, z LongInt
endVar

x = 2147483647  ; верхний предел для переменной типа LongInt
y = 1
z = x + y       ; приводит к ошибке
```

Для работы со значениями, близкими к границам диапазона, следует записывать результат в переменную типа Number.

Примечание: Методы стандартной библиотеки, определенные для типа Number, могут использовать также и переменные типа LongInt. Синтаксис остается неизменным, а возвращаемые значения принадлежат к типу Number. Тип LongInt наследует также методы, определенные для AnyType.

Memo

MEMO-поля содержат текст и форматированные данные - до 512MB в таблицах Paradox. Используя методы типа Memo readFromFile и writeToFile, можно перемещать MEMO-поля между формами (и отчетами), таблицами и дисковыми файла-ми.

Можно также использовать оператор (=) для присвоения значения MEMO-поля переменным типа Memo или String.

Примечание: Для переменных типа Memo не определены операторы арифметических действий и сравнения.

Если значение MEMO-поля присваивается переменной типа String, результирующая строка будет содержать только текст без форматирования. Если значение MEMO-поля присваивается переменной типа Memo, сохраняются как текст, так и форматирование.

Тип Memo наследует также методы, определенные для типа AnyType.

Number

Переменные типа Number представляют собой числовые значения с плаваю-щей точкой, состоящие из значащей части (дробной части, например, 3.224), умноженной на степень 10. Число может содержать до 18 значащих цифр и находиться в диапазоне от ± 3.4 * 10 -4930 до ± 1.1 * 10 4930. Попытка присвоения переменной типа Number значения вне данного диапазона приводит к ошибке.

Примечание: Тип Number наследует методы, определенные для типа AnyType. Процедуры и методы стандартной библиотеки, определенные для типа Number, работают также и с переменными типа LongInt и SmallInt. Синтаксис инструкций остается тем же, а возвращаемые значения принадлежат к типу Number. Так, например, работает следующая программа, хотя sin и не содержится в списке методов для типа LongInt:

```
var
  abc LongInt
  xyz Number
```

```
endVar
abc = 43
xyz = abc.sin()
```

Примечание: Язык ObjectPAL допускает и альтернативную запись:
имяМетода (переменнаяОбъекта , аргумент [,аргумент])

Здесь имяМетода представляет имя метода, переменная переменнаяОбъекта представляет объект и аргумент представляет аргумент или список аргументов. Так, например, в следующей конструкции используется стандартный синтаксис ObjectPAL для расчета синуса числа:
theNum.sin()

В альтернативной записи та же операция имеет вид:
sin(theNum)

Для ясности и согласованности программ предпочтительнее использовать стандартный синтаксис, но там, где удобнее, допустимо использование альтерна-тивного синтаксиса.

Примечание: Экранный формат чисел, возвращаемых методами типа Number, может изменяться в зависимости от установленного пользователем числового формата Windows, но внутреннее представление чисел ObjectPAL остается неизменным.

OLE

OLE (Object Linking and Embedding - связь и включение объектов) - это протокол Windows для работы с составными документами. Это механизм работы с составными документами, обеспечивающий доступ к функциям другого приложения без необходимости выходить из Paradox и открывать это приложение для внесения изменений в документ этого приложения.

Например, предположим, что имеются таблицы, содержащие графические изображения и требуется создать приложение Paradox, обеспечивающее возмож-ность их редактирования. Одним из способов решения проблемы может быть создание графических объектов с помощью графической программы, являющейся сервером OLE (определение ниже). Затем функции графической программы делаются доступными для других пользователей с помощью методов ObjectPAL типа OLE (разумеется, в предположении, что графическая программа входит и в систему пользователя).

Тип OLE наследует также методы, определенные для типа AnyType. Более подробное описание содержится в разделе AnyType .

Примечание: ObjectPAL и Paradox поддерживают еще один протокол разделения данных - протокол динамического обмена данных DDE (Dynamic Data Exchange).

При описании методов OLE используются следующие термины:

- Сервер OLE - приложение, в котором доступ к содержащимся в нем документам обеспечивается средствами OLE. Paradox не является сервером OLE.
- Клиент OLE - приложение, в котором средства OLE могут быть использованы для доступа к документам, создаваемым сервером OLE. Paradox является клиентом OLE.
- Объект OLE - документ, созданный с помощью сервера OLE. Он содержит данные, которые могут потребоваться в приложении Paradox.
- Переменная OLE - переменная ObjectPAL, объявленная как принадлежащая к типу OLE. Переменная OLE обеспечивает средство для работы с объектами OLE. Другими словами, в программах на языке ObjectPAL переменные OLE могут исполь-зоваться для операций над объектами OLE.

Point

Переменная типа Point (точка) содержит информацию о позиции точки на экране. В языке ObjectPAL экран понимается как двумерная сетка с началом в верхнем левом углу контейнера объекта интерфейса. Положительные значения координаты x отсчитываются направо, а положительные значения координаты y вниз от точки начала координат. Переменная типа Point содержит два значения, x и y, измеряемые в единицах твип (твип равен 1/1440 дюйма, т.е. 1/20 принтерной точки).

етоды, определенные для типа Point, считывают и задают информацию о координатах и относительном положении точек на экране. Так, например, с помощью точек задаются свойства size и position объекта

интерфейса.

Примечание: В языке ObjectPAL значения точек отсчитывются относительно контейнера рассматриваемого объекта интерфейса. Если, например, рамка содержит кнопку, координаты кнопки отсчитывются относительно рамки. Если кнопка помещается на пустую страницу, координаты кнопки отсчитывются относительно страницы. Такая относительная система координат используется в методах, принимающих в качестве аргументов или возвращающих значения типа Point.

Тип Point наследует также методы, определенные для типа AnyType. Более подробная информация содержится в разделе AnyType.

Record

Тип Record (запись) в языке ObjectPAL предназначен для определения сразу целого набора данных, и является аналогом записей (record) в языке Pascal или структур (struct) в языке C. Объекты типа Record, определяемые в программах ObjectPAL, отличаются от записей в таблицах.

Конструкции, описывающие переменные типа Record, имеют вид:

```
TYPE
recordName = RECORD
               имяПоля типПоля
             [ имяПоля типПоля ]*
           ENDRECORD
ENDTYPE
```

Один или несколько элементов имяПоля определяют поля (столбцы) записи, а типПоля один из возможных типов данных. Записи (тип Record) следует объявлять в окне Type объекта интерфейса.

После объявления объектов типа Record для их сравнения можно использовать операторы сравнения = и <>. Можно также использовать оператор присваива-ния (=) для копирования содержимого одной записи в другую.

Тип Record наследует также методы, определенные для типа AnyType.

SmallInt

Значения типа SmallInt являются короткими целыми числами, то есть могут быть представлены малым (коротким) набором цифр. Переменная типа SmallInt занимает в памяти 2 байта.

Язык ObjectPAL преобразует к типу SmallInt значения в диапазоне от -32,768 до 32,767. Попытка присвоить переменной типа SmallInt значение, не входящее в данный диапазон, приводит к ошибке, например,

```
var
  x, y, z SmallInt
endVar
x = 32767  ;  верхний предел для переменной типа SmallInt
y = 1
z = x + y  ;  приводит к ошибке
```

Для работы со значениями, близкими к границам диапазона, следует записывать результат в переменную того типа, для которого такие значения являются до-пустимыми. Например,

```
var
  x, y SmallInt
  z    LongInt   ; результат может быть записан в переменную z, объявленную как тип LongInt
endVar
x = 32767        ;  верхний предел для переменной типа SmallInt
y = 1
z = x + y
z.view()         ; выводится 32768 - допустимое значение для переменной z типа LongInt
```

Примечание: Значение типа SmallInt -32,768 не может быть записано в таблицу Paradox, т.к. в

системе Paradox -32,768 представляет собой пустое значение (Blank). Однако, данное значение можно использовать в вычислениях и записывать в таблицы dBASE.

Примечание: Тип SmallInt наследует также методы, определенные для AnyType. За дополнительной информацией следует обратиться к разделу AnyType. Методы стандартной библиотеки, определенные для типа Number, могут использовать также и переменные типа LongInt и SmallInt. Синтаксис остается неизменным, а возвращаемые значения принадлежат к типу Number. Так, например, следующая программа будет выполняться несмотря на то, что sin не содержится в списке методов, определенных для типа SmallInt:

```
var
  abc LongInt
  xyz Number
endVar
abc = 43
xyz = abc.sin()
```

Примечание: Язык ObjectPAL допускает и альтернативную запись:
имяМетода (переменнаяОбъекта, аргумент [,аргумент])

Здесь имяМетода представляет имя метода, переменная переменнаяОбъекта представляет объект и аргумент представляет аргумент или список аргументов. Так, например, в следующей конструкции используется стандартный синтаксис ObjectPAL для расчета синуса числа:

```
theNum.sin()
```

В альтернативной записи та же операция имеет вид:

```
sin(theNum)
```

Для ясности и согласованности программ предпочтительнее использовать стандартный синтаксис, но там, где удобнее, допустимо использование альтернативного синтаксиса.

String

Переменная типа String (строка) может содержать до 32,000 символов (для текста большего объема следует использовать объекты типа Memo). Строка в ка-вычках может содержать до 255 символов. Для задания пустых строк следует ис-пользовать двойные кавычки (""). Строка занимает в памяти по одному байту на символ.

Тип String наследует также методы, определенные для типа AnyType.

Примечание: ObjectPAL поддерживает альтернативный синтаксис:
имяМетода (переменнаяОбъекта, аргумент [,аргумент])

имяМетода представляет имя метода, переменнаяОбъекта является пере-менной, представляющей объект, а аргумент заменяет один или несколько аргументов. Например, следующая конструкция использует стандартный синтаксис ObjectPAL для возвращения представления строки символами нижнего регистра:

```
theString.lower()
```

В следующем операторе используется альтернативный синтаксис:

```
lower(theString)
```

Для ясности и согласованности рекомендуется использовать стандартный вариант синтаксиса, но в подходящем случае можно применять и альтернативный вариант.

Time

Переменная типа Time сохраняет значение времени в виде часы-минуты-секунды-миллисекунды. В качестве разделителей могут использоваться следующие символы: пустой символ, табуляция, пробел, запятая , перенос (-), косая черта (/), точка (.), двоеточие (:), и точка с запятой (;).

Значения типа Time должны быть описаны в явном виде. Например, следующая конструкция

присваивает переменной ti типа Time значение 11 часов 10 минут 40 секунд утра:

```
var ti Time endVar
ti = Time("11:10:40 am")
```

Кавычки вокруг этого значения обязательны. Правильность заданного значения зависит от текущего формата времени Paradox. Например, если в данный момент установлен двенадцатичасовой формат (такой как hh:mm:ss), методы типа Time рассматривают hh:mm:ss как верный формат времени. В языке ObjectPAL существуют процедуры formatSetTimeDefault и formatSetDateTimeDefault типа System для установки формата времени Paradox.

Тип Time наследует ряд методов, определенных для типов AnyType и DateTime.

Основные элементы языка

=	iif	switch
const	loop	try
disableDefault	method	type
doDefault	passEvent	uses
enableDefault	proc	var
for	quitLoop	while
forEach	return	
if	scan	

В данном разделе представлены основные структурные элементы языка ObjectPAL. Большинство из этих элементов не связано с конкретными типами объектов, они применимы для всех типов. Эти элементы можно использовать для присвоения значений, вызова функций из библиотек динамической компоновки (DLL), создания управляющих структур типа циклов if...then...else...endIf, while...endWhile и структур switch...case...endSwitch. С их помощью можно также определять методы, процедуры, константы, переменные и типы данных.

=

Синтаксис: элемент = выражение

Описание: Конструкция = присваивает элементу значение выражения. Любое предыдущее значение элемента теряется. При присвоении значения объекту элемент может содержать полное имя объекта.

В конструкции, содержащей более одного оператора =, первый из них выполняет присвоение, все остальные сравнивают две величины или два выражения.

При использовании конструкции = с объектами интерфейса значение одного объекта интерфейса присваивается другому. Предположим, например, что форма содержит поля: полеОдин и полеДва. Следующая команда копирует значение полеДва в полеОдин.

```
полеОдин = полеДва   ; полеОдин получает значение поляДва
```

Можно использовать конструкцию = с переменными типа UIObject. Язык ObjectPAL использует метод attach аналогично тому, как языки C и Pascal используют указатели. Например,

```
var ui UIObject endVar
ui.attach(полеОдин) ; заставляет ui "указывать" на полеОдин
ui.view()           ; выводит на экран в окно диалога значение ui (аналогичное полеОдин)
ui = полеДва        ; ui получает значение поляДва (значение поляОдин также изменяется)
ui.view()           ; выводит на экран в окно диалога значение ui (аналогичное полюДва)
```

const

Ключевое слово: Описывает константы.

Синтаксис: const

```
    имяКонстанты = типДанных(значение)|значение
        endConst
```

Описание const описывает одну или несколько постоянных величин. тип-Данных указывает тип данных константы. Если типДанных опускается, то тип данных определяется по значению как LongInt, Number, SmallInt или String.

Пример const

```
  a = -1000                          ; тип SmallInt определяется значением правой части
  x = 123.45                         ; определяется тип Number
  newYear = Date ("01/01/99)         ; присваивается значение типа Date
  companyName = String("Borland") ; присваивается строка
endConst
ui.color = Red                        ; устанавливает красный цвет ui, а следовательно, и поляОдин.
```

Представленная ниже команда передает объекту ui всю информацию ObjectPAL о полеОдин:
ui.attach(полеОдин)

И напротив, следующий оператор присваивает ui (а также полюОдин) только значение поляДва:
ui = полеДва

Пример:

```
var
  x AnyType
  ar Array[5] AnyType
  w Logical
  y, z SmallInt
  fred, sam UIObject
endVar

x = 5.14              ; x получает значение 5.14 (тип Number)
ar[1] = "Привет"      ; первый элемент массива ar получает значение "Привет" (строкового типа)
y = 5                 ; y получает значение 5
z = 12                ; z получает значение 12
x = "вот"                  ; x получает новое значение: строка "вот"
myTable.моеПоле = y + z ; моеПоле получает значение y+z
amountField = tempAmountField
bigBox.bigCircle.smallBox.smallCircle.color = Blue
                      ; свойство color (цвет) объекта smallCircle получает значение Blue
                      ; первый знак = присваивает значение, все остальные выполняют сравнение
w = (y = z)                ; w получает значение True, если y = z,
                           ; и значение False в другом случае
fred.attach(полеОдин)      ; делает fred "указателем" на полеОдин
sam = fred                 ; присваивает sam значение fred
```

disableDefault

Ключевое слово Блокирует выполнение стандартного кода встроенно-го метода
Синтаксис disableDefault
Описание disableDefault препятствует выполнению встроенной программы события. Обычно, встроенный код неявным образом выполняется в конце метода перед конструкцией endMethod. Использование в методе ключевого слова disableDefault отключает неявный вызов встроенной программы.

Пример В этом примере при нажатии клавиши пользователем полю присваивается значение "привет". Вызов disableDefault отключает встроенную программу, поэтому символ в поле не высвечивается. Конструкция message выводит набранный символ на экран в строку состояния.

```
method keyChar(var eventInfo KeyEvent)
  self.value = "привет"       ; "привет" выводится в поле
  disableDefault              ; отключается встроенная программа
  message(eventInfo.char()) ; символ выводится в окно состояния
endMethod
```

doDefault

Ключевое Слово : Выполняет стандартный код встроенного метода.
Синтаксис doDefault
Описание doDefault выполняет встроенную программу события немедленно, а не в конце метода. При этом неявный вызов встроенного кода отключается. Если метод содержит более одной конструкции doDefault, выполняется только первая из них.

Пример В следующем методе при нажатии кнопки делается двухсекундная задержка, после чего выдается системный сигнал и кнопка отпускается. Встроенный код неявно вызывается перед конструкцией endMethod:

```
method pushButton(var eventInfo Event)
  sleep(2000)
  beep()
endMethod
```

Во втором примере вызов doDefault отпускает кнопку до задержки и выдачи сигнала, а также отключает встроенный код в конце метода:

```
method pushButton(var eventInfo Event)
  doDefault
  sleep(2000)
  beep()
endMethod
```

enableDefault

Ключевое Слово: Разрешает выполнение стандартного кода встроенного метода
Синтаксис enableDefault
Описание enableDefault разрешает нормальное выполнение встроенной программы в конце метода перед конструкцией endMethod. Сравните enableDefault с ключевым словом doDefault, которое немедленно запускает встроенный код.

Пример

```
method menuAction(var eventInfo MenuEvent)
var theChoice String endVar
disableDefault
theChoice = eventInfo.menuChoice()
switch
  case theChoice = "Open" : doOpen()
  case theChoice = "Quit" : doQuit()
  otherwise               : enableDefault
endSwitch
endMethod
```

for

Ключевое слово Выполняет последовательность конструкций указанное число раз.
Синтаксис for счетчик [from начЗначение] [to конЗначение] [step шаг]
Конструкции
endFor
Описание for выполняет последовательность Конструкций столько раз, сколько определено счет-

чиком, значение которого содержится в переменной счетчик и управляется с помощью необязательных ключевых слов from, to и step. Любая комбинация этих величин может быть использована для определения количества повторений конструкций цикла. Не обязательно описывать переменную счетчик явным образом, однако, при этом цикл будет выполняться быстрее.

начЗначение, конЗначение и шаг являются значениями или выражениями, представляющими начальное и конечное значения, а также значение шага счетчика цикла Значения этих счетчиков могут иметь любой тип данных, входящих в тип AnyType, за исключением типов данных Point, Memo, Graphic, String, OLE и Binary.

Можно использовать for без ключевых слов from, to и step:

- Если пропущено значение начЗначение, счетчик начинается с текущего значения переменной счетчик.
- Если пропущено значение конЗначение, цикл выполняется бесконечное количество раз.
- Если пропущено значение шага, счетчик увеличивается на единицу за каждый цикл
- начЗначение, конЗначение и шаг запоминаются во временном буфере, а не вычисляются каждый раз в процессе выполнения цикла.

При использовании в теле цикла ключевого слова quitLoop, цикл for...endFor заканчивается. Если в теле цикла используется ключевое слово loop, то конструкции следующие за словом loop пропускаются, значение счетчика увеличивается и итерации продолжаются с начала цикла for.

Если значение step положительное, и есть предложение to, итерации продолжаются, пока значение счетчик меньше или равно конЗначению. При отрицательном значении step, итерации выполняются, пока значение переменной цикла больше или равно начЗначению. В любом случае, как только значение переменной цикла счетчик достигает предела, определяемого конструкцией step, цикл for завершается, а переменная счетчик сохраняет свое значение, как это показано в примере.

Если переменной цикла счетчик ранее не было присвоено значение, то конст-рукция from создает эту переменную и присваивает ей начЗначение.

Пример Ниже представлен простой цикл for. Обратите внимание на значение переменной цикла i после завершения цикла for.

```
var i SmallInt endVar
for i from 1 to 3
  i.view("Внутри цикла for")          ; i = 1, i = 2, i = 3
endFor
i.view("После завершения цикла") ; i = 4
```

forEach

Ключевое слово Повторяет указанную последовательность конструк-ций для всех элементов динамического массива.

Синтаксис
```
forEach имяПеременной in имяДинМассива
  Конструкции
endForEach
```

Описание forEach перебирает все элементы динамического массива. В общем случае, нельзя использовать конструкцию for для перебора элементов динамического массива, так как индексы динамического массива не обязательно являются целыми числами.

Поскольку индексы динамического массива не целые числа, элементы динамического массива не являются последовательно упорядоченными. Конструкция forEach обрабатывает элементы динамического массива в произвольном порядке. Не нужно полагаться на определенный порядок индексов.

Отсутствие имени динамического массива имяДинМассива в конструкции forEach приводит к ошибке при компиляции метода.

Если в теле цикла forEach используется конструкция quitLoop, цикл forEach...endForEach завершается. При использовании в теле цикла конструкции loop, операторы следующие за loop пропускаются, а итерации продолжаются с нача-ла цикла forEach.

Пример. В следующем примере конструкция forEach используется для вывода на экран элементов динамического массива, созданного оператором sysInfo:

```
var
  SystemArray DynArray[] AnyType
  Element AnyType
endVar
sysInfo(SystemArray)
forEach Element IN SystemArray
  message(Element, " : ", SystemArray[Element])
  sleep(1500)
endForEach
```

if

Ключевое слово Выполняет одну из двух последовательностей конст-рукций, в зависимости от значения логического выражения.

Синтаксис
```
if Условие then
    Конструкции1
  [ else
    Конструкции2 ]
endIf
```

Описание Когда ObjectPAL встречает конструкцию if, то определяется значение Условия. Если оно имеет значение True, то выполняется набор Конструкций1 . В противном случае, Конструкции1 пропускаются и, если имеется ключевое слово else, выполняется последовательность Конструкций2. В обоих случаях выполнение продолжается после ключевого слова endIf.

Конструкция If может охватывать несколько строк, особенно при большом количестве элементов в Конструкции1 или в Конструкции2. Хорошим стилем является выделение отступом предложений then и else, для наглядного представления по-следовательности выполнения операторов:

```
if Condition then
  Конструкции1
else
  Конструкции2
endIf
```

Рассмотрим пример конструкции if:

```
if Stock < 100 then
  AddStock()              ; выполняется специальный метод AddStock()
  Stock = Stock + 10      ; прибавляет 10 к значению Stock
endIf
```

Конструкция if может быть вложенной, то есть, Конструкции1 или Конструк-ции2также могут содержать структуры if. Вложенная конструкция if должна полностью содержаться внутри управляющей структуры if. Иными словами, каждому вложенному оператору if должен соответствовать вложенный оператор endIf. Каждая структура if...endIf должна заключать в себе программу или программу и другую пол-ную структуру if...endIf:

```
if Условие then
  if Условие then
    Конструкции
  endIf
endIf
```

Пример В следующем примере представлена программа с вложенной конструкцией if.

```
if skipLevel = "Начальный уровень" then
  if skillBox.color = "Red" or skillBox.color = "Yellow" then
    skillBox.color = "Green"
  endIf
endIf
```

iif

Ключевое слово Возвращает первое или второе значение, в зависимости от значения логического выражения.
Синтаксис iif(Условие, ЗначениеИстина, ЗначениеЛожь)

Описание Оператор iif (immediate if - непосредственно выполняемый if) позволяет выполнять ветвление в пределах одной команды. iif можно использовать там же, где и любое другое выражение. Он особенно полезен в вычисляемых полях форм и отчетов, где запрещено использование конструкций if... endIf. Часто используется в формах для полей, которые расчитываются. Например, для учета электроэнергии используются счетчики с трансформаторами и без, соответственно расход будет равен или разности начального и конечного показания, или разности, умноженной на коэффициент трансформации. Ниже приведен пример синтаксиса, а реальное представление на рис.1.2

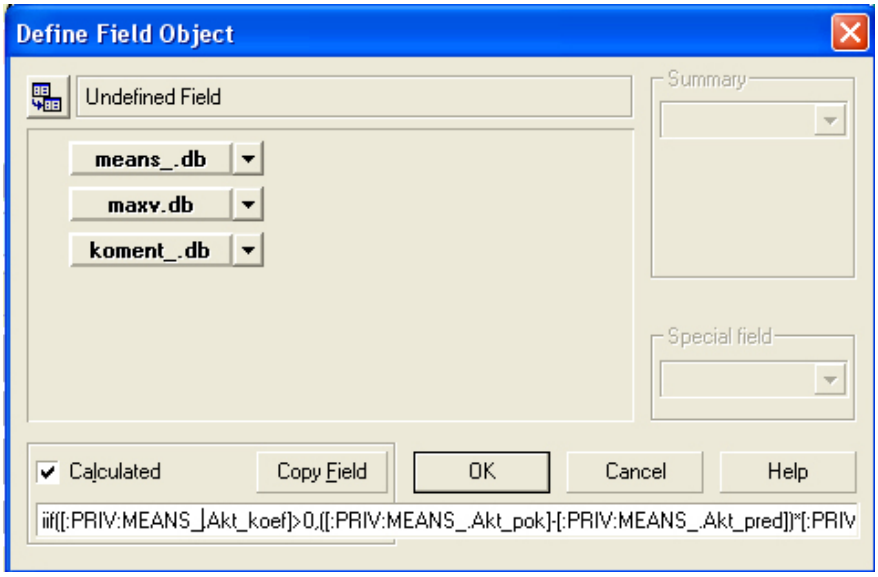


Рис.1.2 Использование iff в вычисляемом поле

Пример a = iif(x > 1, b, c) ; если x > 1, a = b, иначе a = c

loop

Ключевое слово Передает управление в начало ближайшего охваты-ваюшего цикла for, forEach, scan или while.
Синтаксис loop
Описание Результатом выполнения оператора loop внутри структур for, forEach, scan или while, является пропуск конструкций, заключенных между loop и ключевыми словами endFor, endForEach, endScan или endWhile, и переход к началу структуры. Вызов loop в других случаях приводит к ошибке.

```
Пример      var x SmallInt endVar
            for x from 1
                if x <> 5 then
                  loop   ; переход на начало конструкции, получение следующего значения x
                  message("Никогда не появляется") ; этот оператор никогда не выполняется
                   else
                quitLoop                              ; прерывает цикл
              endIf
            endFor
            message(x) ; показывает 5
```

method

Ключевое Слово Определяет метод языка ObjectPAL
Синтаксис method Имя ([описаниеПараметров [,описаниеПараметров]*]) [типВозврата]
[type раздел]
[const раздел]
[var раздел]

Конструкции
endMethod

Описание Ключевое слово method отмечает начало метода. Как минимум , конструкция должна включать:

Имя метода,Имя
Скобки, даже если метод не имеет аргументов

Конструкцию Конструкции, содержащую тело метода
Определение оканчивается обязательным ключевым словом endMethod.

Дополнительно можно описать константы, типы данных, переменные и процедуры перед ключевым словом method. Можно также определять переменные и константы после этого слова.

Также необязательным является наличие одного или более описаний параметров, обозначенных в прототипе как описание Параметров. Каждое такое описание имеет вид

[var|const] параметр тип

Необязательный элемент типВозврата определяет тип данных возвращаемой методом величины. типВозврата является необязательным, так как метод может и не возвращать значение. Однако, если метод возвращает значение, тип этого значения должен быть определен.

Методы и процедуры во многом схожи. Принципиальная разница состоит в следующем:

Методы являются видимыми и экспортируемыми (доступными для других объектов), в то время как процедуры являются частными (локальными) в пределах иерархии вложенности.

В ObjectPAL , как говорилось выше, каждый объект имеет свои методы. Поэтому при задании методов название его и последняя строка в большинстве случаев формируются автоматически.
Метод может содержать определение процедуры, тогда как процедура не может содержат определение метода.
Примечание: Область действия метода зависит от того, где он определяется.

Пример

```
method pushButton (var eventInfo Event)
  var
    txt String
    myNum Number
  endVar
  myNum = 123.321
  txt = String(myNum)
  msgInfo("myNum = ", txt)
endMethod
```

passEvent

Ключевое слово Передает событие в контейнер объекта.
Синтаксис passEvent
Описание passEvent передает атрибуты события контейнеру объекта. Использование passEvent в методе не влияет на неявный вызов встроенной программы.
Пример Следующая программа определяется для объекта-поле. Она выполняется, когда указатель мыши попадает в этот объект. Если при нажатии кнопки мыши удерживается нажатой клавиша Shift, программа вызывает disableDefault для отключения встроенной программы и passEvent для передачи события контейнеру объекта-поле. Такой прием полезен, если нужно заставить несколько объектов оди-наковым образом реагировать на данное событие.

```
method mouseDown(var eventInfo Mousevent)
 if eventInfo.isShiftKeyDown() then
   disabledefault
```

```
    passEvent ; передает обработку события контейнеру
  endIf
endMethod
```

proc

Ключевое слово Определяет процедуру языка ObjectPAL.

Синтаксис proc ИмяПроцедуры ([описаниеПараметров [,описаниеПараметров]*]) [типВозврата]
[const раздел]
[type раздел]
[var раздел]
Конструкции
endProc

Описание Описание процедуры начинается с ключевого слова proc. Далее пользователь задает:

- Имя процедуры, ИмяПроцедуры .
- Скобки, даже если процедура не имеет аргументов.
- Одно или несколько описаний параметров, представленных в прототипе элементами описаниеПараметров , где каждое описание имеет вид
  [var|const] параметр тип
- типВозврата для определения типа данных возвращаемого значения (если процедура возвращает значение).
- Разделы описаний переменных, констант и типов.
- Конструкцию Конструкции, содержащую тело процедуры.

Определение оканчивается обязательным ключевым словом endProc.

В теле процедуры для передачи значения в вызывающий метод или процедуру можно использовать ключевое слово return.

Если процедура используется в выражении, она должна возвращать значение, например:

```
x = NumValidRecs("Orders") ; NumValidRecs - это процедура
```

Методы и процедуры во многом схожи. Принципиальная разница состоит в следующем:

- Методы являются видимыми и экспортируемыми (доступными для других объектов), в то время как процедуры являются частными (локальными) в пределах иерархии вложенности.
- Метод может содержать определение процедуры, тогда как процедура не может содержат определение метода.

Примечание: Область действия процедуры зависит от того, где она определяется.

Пример

```
proc inc (x SmallInt) SmallInt
return x = x + 1
endProc
method pushButton(var eventInfo Event)
  var x SmallInt endVar
  x = 5
  x = inc(x)   ; вызов процедуры
  message(x) ; выводит на экран 6
endMethod
```

quitLoop

Ключевое слово Заканчивает выполнение циклов for, forEach, scan или while, в которых оно появляется.

Синтаксис quitLoop

Описание quitLoop осуществляет немедленный выход из ближайшего охватывающего цикла for, forEach, scan или while. Выполнение метода продолжается за ближайшим ключевым словом endFor, endForEach, endScan или endWhile.

Вне структуры любого из перечисленных циклов quitLoop вызывает ошибку.

Пример В данном примере quitLoop используется в цикле, в котором определяется, содержит ли массив элементы с неприсвоенными значениями.

```
var
  myArray Array[12]
  notAssigned Logical
endVar
notAssigned = False
for i from 1 to myArray.length()
  if not isAssigned(myArray[i]) then
    notAssigned = True
    quitLoop
  endIf
endFor
```

return

Ключевое слово Возвращает управление из метода или процедуры, с возможностью передачи назад значения.

Синтаксис return [Выражение]

Описание return используется для передачи управления от текущих процедуры или метода в процедуру или метод, которые их вызывали. При использовании return нужно учитывать следующее:

- Прежде чем использовать return, процедура должна быть описана как возвращающая значение.
- Если return используется в теле процедуры, процедура заканчивается.
- Если return используется в методе (но вне тела процедуры), то метод завершается.

Можно передать значение Выражения при выходе из процедуры или метода. Если процедура вызывается в выражении, тогда процедура должна возвращать значение, которое используется при вызове процедуры.

```
y = myProc(x) + 3 ; myProc это процедура
```

Когда процедура вызывается в изолированной конструкции, любое возвращаемое значение игнорируется. Например:

```
myProc()
```

Если после return не следует Выражение, на строке за return не может быть ничего кроме комментария.

Нельзя возвращать такие типы данных как DDE, Database, Query, Session, Table или TCursor.

Нет необходимости использовать return для передачи управления назад в метод или процедуру верхнего уровня, поскольку это происходит автоматически по завершении процедуры или метода нижнего уровня. Однако, если метод или процедура описаны, как возвращающие значение, необходимо использовать return для возвращения этого значения; оно не может быть передано автоматически.

Пример Следующая простая программа прибавляет единицу к значению переменной и возвращает новое значение вызывающему методу:

```
proc addOne (x SmallInt) SmallInt
  return x + 1
endProc
```

Во встроенном методе конструкция return выполняет встроенную программу, если она явным образом не отключена. Например, следующая программа вызывает return, когда пользователь вводит символ "?" в объект-поле. Вызов disableDefault, не позволяет встроенной программе вывести знак "?" в объект-поле.

```
method keyChar(var eventInfo KeyEvent)
  if eventInfo.char() = "?" then
    disableDefault
    return
  endIf
endmethod
```

scan

Ключевое слово Сканирует таблицу с помощью табличного указателя (TCursor) и выполняет инструкции ObjectPAL.

Синтаксис scan tcПеременная [for логическоеВыражение] :
Конструкции
endScan

Двоеточие используется, чтобы отделить scan от следующего оператора for.

Описание scan анализирует значение табличного указателя tcПеременная (объект типа TCursor) и выполняет Конструкции (набор инструкций ObjectPAL) для каждой записи. scan всегда начинается с первой записи таблицы и последовательно проходит по каждой записи. Когда конструкции в цикле scan изменяют индексированное поле, запись перемещается на положение в таблице, определяемое заданной сортировкой, поэтому одна запись может быть проанализирована в цикле несколько раз.

Если применяется предложение for, Конструкции выполняются только для записей, удовлетворяющих логВыражению. Все остальные записи пропускаются. Если таблица является пустой, или нет записей, удовлетворяющих условию for, цикл scan не дает никакого результата.

Примечание: for является ключевым словом для конструкции scan, поэтому, чтобы отличать его от цикла for, за ним должно стоять двоеточие.

scan является очень мощным средством, так как можно сначала создать прототип последовательности конструкций для отдельной записи в таблице, а затем поместить эту последовательность внутрь цикла scan, чтобы заставить ее работать для всей таблицы.

Внутри тела цикла scan можно использовать ключевые слова loop, return и quitLoop. Результатом выполнения оператора loop является пропуск конструкций, заключенных между loop и ключевым словом endScan, переход к следующей записи и возврат к началу цикла scan. quitLoop полностью заканчивает выполнение цикл, покидая запись, обрабатываемую в данный момент.

Поскольку цикл scan повторяет всю последовательность инструкций для каждой записи, в него не следует включать действия, которые нужно выполнить для таблицы один раз. Эти конструкции лучше поставить вне цикла scan. scan автоматически переходит от одной записи к другой в таблице, поэтому не требуется вызов оператора nextRecord.

Пример 1 В данном примере цикл scan используется для обновления таб-лицы Employee. Обрабатывается поле Dept каждой записи, и если найдено значение "Персонал", оно заменяется на "Сотрудники".

```
var
  empTC TCursor
endVar

emtTC.open("employee.db") ; Эти конструкции нужно выполнить
empTC.edit()              ; только один раз

scan empTC for empTC.Dept = "Персонал" :  ; двоеточие является обязательным
  empTC.Dept = "Сотрудники"
endScan

empTC.endEdit()
empTC.close()
```

Пример 2 В этом примере в поле “kod” ставятся порядковые номера записей

```
method run(var eventInfo Event)
var
t tcursor
s SmallInt
endvar
;------------
s=1
t.open("trtok")
t.edit()
scan t:
  t."kod"=s
  s=s+1
endscan
t.endedit()
t.close()
endMethod
```

switch

Ключевое слово Выполняет одну или несколько альтернативных последовательностей конструкций, в зависимости от выполнения условий из списка.

Синтаксис switch
CaseСписок
[otherWise: Конструкции]
endSwitch

CaseСписок содержит произвольное количество конструкций вида:
case Условие : Конструкции

Описание При определении, какую последовательность Конструкций вы-полнять, оператор switch использует значение конструкции Условие из списка CaseСписок, если выполняется хотя бы одно условие. Оператор switch эквивалентен набору операторов if, а каждый элемент списка CaseСписок одиночной конструкции if.

Условия определяются в порядке расположения в списке:

- Если одно из них имеет значение True, выполняется соответствующая последовательность Конструкций, а все остальные пропускаются.
- Если ни одно из них не получает значение True, и имеется необязательное предложение otherWise, выполняется Конструкции в otherWise.
- Если ни одно из них не получает значение True, и нет предложения otherWise, то конструкция switch не выполняет никаких действий.

Таким образом, выполняется не более одной последовательности Конструкций. Выполнение метода продолжается с оператора, следующего за endSwitch.

Пример В следующем примере создается массив из ста случайных чисел, которые затем сортируются в порядке возрастания с помощью алгоритма попарного сравнения:

```
method pushButton(var eventInfo Event)
var
  sz, i , itmp, j,k SmallInt
  a Array[100] SmallInt
  tmp Number
endVar

  sz = 100
  a.fill(0)
```

```
for i from 1 to sz step 1
  tmp = Rand()
  switch
      case tmp << .1 : a[i] = 1
      case tmp << .2 : a[i] = 2
      case tmp << .3 : a[i] = 3
      case tmp << .4 : a[i] = 4
      case tmp << .5 : a[i] = 5
      case tmp << .6 : a[i] = 6
      case tmp << .7 : a[i] = 7
      case tmp << .8 : a[i] = 8

      case tmp << .9 : a[i] = 9
      otherwise:     a[i] = 10
  endSwitch
endFor

for i from 1 to sz-1 step 1
  for j from 1 to sz-i step 1
    if a[j] <<>> a[j+1] then
      a.exchange(j, j+1)
    endIf
  endFor
endFor

endMethod
```

try

Ключевое слово Отмечает блок проверяемых конструкций и определяет отклик при возникновении ошибок.

Синтаксис

```
try
    [Конструкции] ; блок транзакции
  onFail
    [Конструкции] ; блок обработки ошибок
   [reTry]        ; необязательный блок
endTry
```

Описание Блок try...onFail предоставляет возможность встроить в приложение механизм обработки ошибок.

Блок транзакции - это набор конструкций, для каждой из которых необходимо успешное выполнение. Если транзакция выполняется, программа переходит на оператор endTry. Если при выполнении транзакции встречается ошибка, то выполняется блок обработки ошибок. Для повторного выполнения блока транзакции можно использовать конструкцию reTry.

Причиной того, что попытка приводит к ошибке, является программный вызов системной процедуры fail внутри блока транзакции или внутри процедур, вызывае-мых в этом блоке. В результате этого системные функции не возвращают вызывающим программам состояние ошибки или нулевое значение.

При этом локальные переменные этих процедур удаляются из стека, а любые специальные объекты (такие как большие текстовые блоки) удаляются из памяти. Если используются ссылочные объекты (например, таблицы), они закрываются, а ожидаемое обновление отменяется. Результат таков, как будто транзакция никогда не выполнялась. Сохраняются только изменения переменных вне блока и данные, успешно добавленные в таблицы и обработанные до появления ошибки.

Если в блоке обработки выясняется, что код ошибки не входит в набор ожи-даемых кодов, или

является слишком сложным для обработки на данном уровне, можно снова вызвать системную процедуру fail для передачи этого кода. Если отсутствует блок try...onFail более высокого уровня, то выполнение всего приложения прерывается, отменяются существующие действия, закрываются ресурсы, и приложение завершается.

Пример В данном примере осуществляется попытка задать свойство Color некоторым объектам интерфейса, и, если это свойство не может быть присвоено, для обработки ситуации используется блок try...onFail.

```
method pushButton(var eventInfo Event)
var s String endVar
box1.box2.color = Blue      ; это выполняется
s = "box5"                  ; объект box5 не существует

try
  box1.(s).color = Red      ; попытка задать цвет для box5
onFail                      ; обработка ошибки
  msgStop("Ошибка!","Не могу найти " + s)
  s = "box2"                ; объект box2 существует
reTry                       ; повторная попытка

s = "box6"                  ; объект box6 не существует
try
  box1.(s).color = Green
onFail
  fail(peObjectNotFound,"Объект " + s + " не существует.")
endTry
endMethod
```

type

Ключевое слово Описывает типы данных.

Синтаксис
```
type
[имяНовогоТипа = существующийТип]*
endType
```

Описание С помощью type можно определять новые типы данных. Один раз определив тип, можно использовать его для описания переменных в методах.

Например, в приложении, для отслеживания количества деталей на складе можно описать тип числоДеталей, а затем описать переменную типа числоДеталей:

```
type
  числоДеталей = SmallInt  ; описывает новый тип
endType

var                        ; используем новый тип для описания переменной
  pQty числоДеталей ; pQty имеет тип SmallInt
endVar                     ; так как числоДеталей это SmallInt
```

В дальнейшем, если количество деталей приблизится к числу 32767 (максимальное значение для типа SmallInt), нужно только изменить определение типа, например,

```
type
  числоДеталей = LongInt ; изменяется описание
endType
```

```
var                          ; используем новый тип для описания переменной
  pQty числоДеталей          ; pQty имеет тип LongInt
endVar                       ; так как числоДеталей это LongInt
```

Пример Полезным типом является запись (record). Записи, определенные в окне объекта Type не имеют связи с таблицами. Они подобны записям в языке Pascal или структурам в языке C, так как позволяют объединить вместе под одним именем несколько связанных элементов данных. Например, следующая программа описывает запись Employee, которую можно использовать для описания переменных в методах и процедурах:

```
type
  Служащий = record
        Фамилия      String
        Имя          String
        Должность    String
        Оклад        Currency
        ДатаНайма    Date
        endRecord
endType
```

uses

Ключевое слово Описывает программы внешней библиотеки, которые можно использовать в методе или процедуре.

Синтаксис uses [библиотека|ObjectPAL]
имяПрограммы (списокПараметров)
endUses

Описание Блок uses, описанный в окне объекта Uses, делает программы из внешних библиотек доступными для методов языка ObjectPAL. Эти программы должны быть одного из следующих видов:
Программы, написанные на языке ObjectPAL и содержащиеся в библиотеках ObjectPAL

Программы, написанные на языке ObjectPAL и определенные для формы. Синтаксис вызова методов, определенных для формы, такой же, как и при вызове из библиотеки.

Программы, написанные на языках ассемблер, C, C++ или Pascal и содержащиеся в библиотеке ObjectPAL или библиотеке динамической компоновки Windows (DLL - Dynamic-link library). Библиотека динамической компоновки является библиотекой, содержащей исполняемый код или данные, которые можно присоединять к приложению на стадии выполнения. Используя эти библиотеки, можно добавлять свойства и функции без модификации откомпилированного приложения ObjectPAL.

Блок uses для библиотеки ObjectPAL несколько отличается от блока uses для библиотеки динамической компоновки, поэтому они обсуждаются в различных разделах.

Для того, чтобы использовать методы из библиотеки ObjectPAL или методы, определенные для формы, блок uses записывается в окне объекта Uses в следую-щем виде:

```
uses ObjectPAL
  [имяМетода ( [var | const] списокАргументов) [типВозврата]]*
endUses
```

Ключевое слово ObjectPAL определяет, что вызываются методы из библиотек ObjectPAL или форм, а не методы из библиотек динамической компоновки.

Внимание: Перед вызовом метода из библиотеки эта библиотека должна быть открыта. Для вызова метода из формы форма также должна быть открыта и запущена на выполнение.

имяМетода - это имя метода, который нужно вызвать. списокАргументов - разделяемый запятыми список пар аргументов и типов данных, перед которыми могут быть вставлены соответствующие ключевые слова var или const.

Необязательный аргумент типВозврата определяет тип данных значения, возвращаемого методом,

если это предусмотрено.

В одном окне Uses можно определить более одного библиотечного метода, а также методы более чем из одной библиотеки.

Внимание: Аргументы и типы данных должны быть описаны в окне Uses в строгом соответствии с их описанием в библиотеке.

В следующем примере описаны два библиотечных метода, buildMenu и calcInterest. buildMenu имеет один аргумент, строковую константу названиеСтраницы. calcInterest - два аргумента: числовую переменную индСтавка (передаваемую по ссылке) и переменную целого типа числоПериодов (передаваемую по значению).

```
uses ObjectPAL
  buildMenu(const названиеСтраницы String)
  calcInterest(var индСтавка Number, числоПериодов SmallInt)Number
endUses
```

Следующая программа, определенная в окне Uses кнопки, описывает метод calcInterest, который затем может вызываться для этой кнопки. Заметим, что пользователь должен описать только те методы библиотеки, которые он будет использо-вать.

```
uses ObjectPAL
  calcInterest(var индСтавка Number, числоПериодов SmallInt)Number
endUses
```

Другая программа, определенная во встроенном методе кнопки pushButton, открывает библиотеку и вызывает этот метод:

```
method pushButton(var eventInfo Event)
  var
    mathLib  Library
    intRate  Number
    nPeriods SmallInt
    Interest Number
  endVar

  if mathLib.open("mathlib.lsl") then
    intRate = mortgage.intRate.value
    nPeriods = mortgage.nYears.value * 12
    interest = mathLib.calcInterest(intRate, nPeriods)
    interest.view("Interest")
  endIf
endMethod
```

В этом примере точечная нотация определяет путь поиска метода calcInterest. Следующая конструкция предлагает осуществлять поиск в библиотеке, представленной переменной mathLib типа Library.

```
interest = mathLib.calcInterest(intRate, nPeriods)
```

Концепция вызова специальных методов, определенных для формы, анало-гична рассмотренной: для ссылки на форму, методы которой вызываются, следует использовать точечную нотацию. В следующем примере предполагается, что ранее была описана переменная формы codeForm, открыта

форма, а метод getObjHelp описан в соответствующем окне Uses.

```
interest = codeForm.getObjHelp(self.name)
```

Для использования процедур из библиотеки динамической компоновки следует поместить блок uses в одно из следующих мест:

- В окно Uses объекта интерфейса
- В текст встроенного метода
- В текст специального метода
- В текст специальной процедуры

Ответ на вопрос, куда помещать этот блок, зависит от предполагаемой облас-ти определения (доступности) процедуры.

Вне зависимости от того, куда помещен этот блок, его основная структура (продемонстрированная в следующем псевдокоде) одинакова:

```
uses имяБиблиотеки
   имяПрограммы (списокПараметров) типВозврата
endUses
```

Аргумент имяБиблиотеки указывает имя файла библиотеки динамической компоновки, причем имяБиблиотеки должно быть правильным именем файла DOS из не более чем восьми символов. Paradox предполагает расширение файлов .DLL или .EXE. Windows осуществляет поиск в следующем порядке:

1. Текущий каталог.
2. Каталог Windows (каталог, содержащий файл WIN.COM). Чтобы получить эту информацию, можно использовать процедуру windowsDir типа FileSystem (обыч-но это C:\WINDOWS).
3. Системный каталог Windows (каталог, содержащий такие системные файлы, как KERNEL.EXE). Для получения этой информации, можно использовать процедуру windowsSystemDir типа FileSystem (обычно это C:\WINDOWS\SYSTEM).
4. Каталоги, определенные в переменной среды PATH. За дополнительной информацией следует обращаться к документации по системе DOS.
5. Список каталогов, определенных в сети.

Примечание для профессиональных программистов в среде Windows: при вызове программ из уже загруженной библиотеки динамической компоновки (DLL) (например, DLL-библиотеки, загружаемой автоматически системой Windows), вместо имени файла можно использовать имяБиблиотеки для ссылки на имя модуля библиотеки динамической компоновки. Обратитесь к документации по языку программирования за дополнительной информацией об именах модулей DLL.

Блок uses может содержать один или более одного аргумента имяПрограммы, и каждое имя может иметь свой собственный списокПараметров. В нем указываются одно или более имен переменных и типов данных. Если программа возвращает значение, то типВозврата определяет тип данных возвращаемого значения. Средствами ObjectPAL приведенное описание переменных проверяется на точное соответствие с описанием в самой программе, и это единственная проводящаяся проверка.

В блоке uses типы данных описываются следующими ключевыми словами:

Тип данных	ObjectPAL	C	Pascal
16 битовое целое	CWORD	int	Integer
32 битовое целое	CLONG	long	Longint
64 битовое с пл.точкой	CDOUBLE	double	Double
80 битовое с пл.точкой	CLONGDOUBLE	long double	Extended
указатель	CPTR	char far *	String
двоичные, графические	CHANDLE	Handle (Windows)	THandle (Windows)

Эти ключевые слова могут использоваться только в пределах блока uses. Не следует использовать их в других местах.

Для вызова программы из метода следует предварительно описать переменные, которые будут использоваться как аргументы. Например,

```
; эта часть набирается в окне объекта Uses
uses myStuff ; считываются программы из библиотеки MYSTUFF.DLL
  doSomething(thisNum CLONG, thatNum CLONG) CDOUBLE ; описывается программа
endUses

; теперь модифицируется метод объекта mouseUp
method mouseUp(var eventInfo MouseEvent)
var
  thisNum, thatNum LongInt ; определяются переменные, передаваемые программе
  myResult Number
endVar

thisNum = 3,155.111
thatNum = 5,535.345
myResult = doSomething(ThisNum, ThatNum) ; вызов программы, получение результата
```

В предыдущем примере аргументы в блоке uses...endUses были описаны как CLONG и CDOUBLE, а в методе переменные были описаны как LongInt и Number. Это результат того, что типы данных языка ObjectPAL являются более сложными (и более емкими) чем соответствующие типы данных в языках C или Pascal:

CWORD соответствует SmallInt
CLONG соответствует LongInt
CDOUBLE и CLONGDOUBLE соответствуют Number
CPTR соответствует String
CHANDLE соответствует Binary и Graphic

Примечание: Не следует модифицировать содержание передаваемого аргумента типа CPTR.

Пример В этом примере используются программы из библиотеки динамической компоновки MINMAX.DLL, написанной на языке Turbo Pascal for Windows. Программа на языке Pascal, определяющая библиотеку динамической компоновки, имеет следующий вид:

```
{ это программа на языке Pascal, которая определяет DLL }
library MinMax;

function Min(X, Y: Integer): Integer; export;
begin
  if X < Y then Min := X else Min := Y;
end;

function Max(X, Y: Integer): Integer; export;
begin
  if X > Y then Max := X else Max := Y;
end;

exports
  Min index 1,
  Max index 2;

begin
end.
```

Далее приводится программа на языке ObjectPAL, в которой используется программа из библиотеки динамической компоновки (DLL). Вначале приводится код для окна Uses, а затем программа, которая модифицирует метод кнопки pushButton:

```
; эта часть набирается в окне Uses для кнопки
uses MInMax ; загружаются программы из MINMAX.DLL
  Min (x CWORD, y CWORD) CWORD ; описываются программы
  Max (x CWORD, y CWORD) CWORD
endUses
```

Следующая программа модифицирует стандартный метод кнопки pushButton:

```
method pushButton(var eventInfo Event)
  var
    x, y, z SmallInt
  endVar
  x = 2
  y = 6
  z = Min(x, y)          ; вызов Min из DLL
  msgInfo("Min", z)
  z = Max(x, y)          ; вызов Max из DLL
  msgInfo("Max", z)
endMethod
```

var

Ключевое слово Описывает переменные.

Синтаксис
```
var
[имяПеременной [ , имяПеременной ] * имяТипа ] *
endVar
```

Описание Блок var...endVar описывает переменные, связывая имяПеременной с типом данных имяТипа. При описании более одной переменной одинакового типа на одной строке для отделения имен используется запятая.

Примечание: Область действия переменной зависит от того, где она описана.

Пример

```
var
  myChars, xx String
  myNum Number
  orders, sales, parts TCursor
  proteus AnyType
  myBox UIObject
  a, b Array[5] SmallInt
  myOtherNum Number
endVar
```

while

Ключевое слово Повторяет последовательность конструкций до тех пор, пока указанное условие имеет значение True.

Синтаксис
```
while Условие
  [Конструкции]
endWhile
```

Описание Выполнение цикла while начинается с оценки логического выра-жения Условие. Если

оно имеет значение False, Конструкции не выполняются. Если оно равно True, последовательно выполняются Конструкции между Условием и словом endWhile. Затем управление передается к началу цикла, и значение Условия определяется снова. Эти шаги повторяются до тех пор, пока значение Условия не станет равным False. В этом случае цикл завершается, а управление переходит к следующему оператору за endWhile.

Можно использовать ключевое слово loop внутри тела цикла while, чтобы немедленно вернуть управление к началу цикла, пропуская конструкции между словами loop и endWhile. Можно также использовать ключевое слово quitLoop для окончательного выхода за пределы цикла. Допускается любой уровень вложенности циклов while друг в друга.

Циклы while и for похожи, но обычно используются по разным причинам. for используется для выполнения последовательности операторов известное число раз. while же используется для повторения последовательности конструкций произвольное количество раз.

Пример

```
; этот пример создает массив имен
var
  myNames TCursor
  namesArray Array[] String
  n SmallInt
endVar

myNames.open("names.db")
namesArray.grow(1)
namesArray[1] = myNames."Фамилия"
n = 1

while myNames.nextRec()
n = n + 1
```

Уединенные программы

Уединенная прграммы состоит из кода всвоем собственном файле, не определенного для формы. Она является объектом и выводится на экран в главное окно в виде пиктограммы. Для программы можно

- определять один или более методов.
- описывать переменные, константы, типы данных и процедуры.
- вызывать специальные библиотеки динамической компоновки (DLL).

Программа не имеет типа, не выводится на экран в окно и не содержит никаких объектов интерфейса. Программа имеет встроенные методы run, error и status. (Status доступен только на профессиональном уровне функциональности ObjectPAL.) Эти методы можно выполнить интерактивно в Paradox или вызвать их метода или процедуры ObjectPAL. Как и другие объекты, программа имеет окна для описания переменных, констант, процедур, типов данных и внешних программ. Для нее можно также описать специальные методы. Программы следует использовать при необхо-димости выполнить код без открытия и вывода на экран окна формы.

Из программ имеется полный доступ к стандартной библиотеке ObjectPAL, поэтому из программ можно управлять другими объектами. Например, можно вызывать другие уединенные программы, открывать и работать с таблицами, формами, отчетами и выполнять запросы. Можно вызывать методы, определенные для других объектов, а также получать и устанавливать их свойства.

Для того чтобы изменить уровень функциональности ObjectPAL, в которов Вы работаете, выберите команду Properties / Desktop. В окне диалога "Настройки главного окна" можно изменить уровень функциональности ObjectPAL с начального на профессиональный.

Создание уединенной программы

Исторически от версии DOS уединенная программа называется SCRIPTS (сценарии). Это название образовалось из-за того, что в версии DOS программы можно было не писать. Там из меню выбирались пункты Scripts / BeginRecord и от этого момента все что делается, запоминается Paradox. Для завершения написания сценария работы выбирался пункт Scripts / EndRecord и дается название скрипту.

Для создания уединенной программы можно воспользоваться любым из сле-дующих двух способов:

- выберите команду File / New / Script.
- В окне Project Viewer выбрать Scripts / New

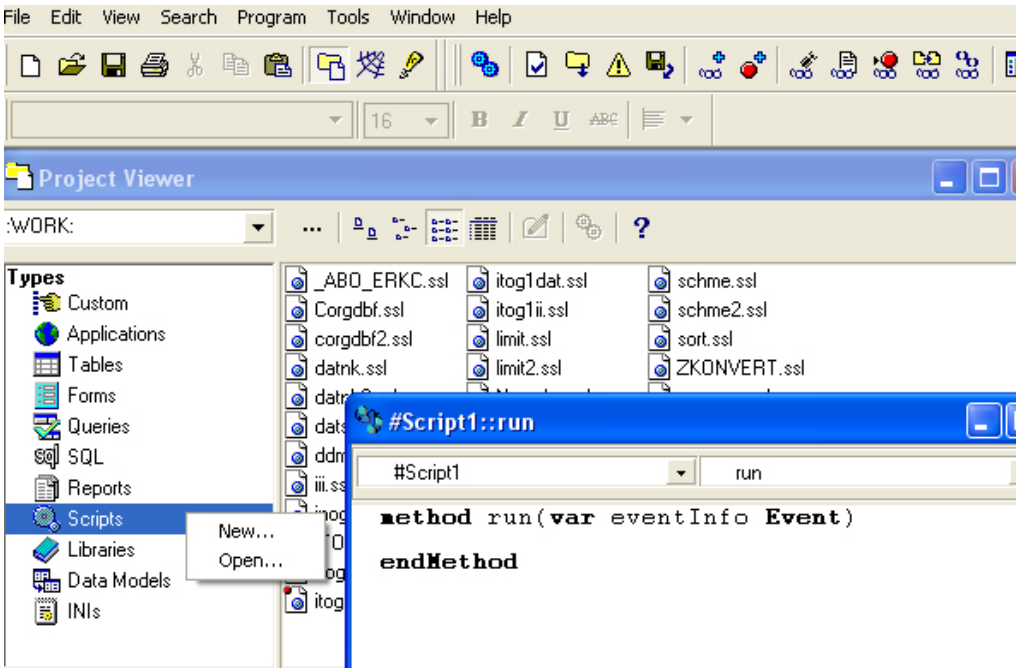


Рис.1.3 Создание уединенной программы.

В результате откроется окон редактора ObjectPAL для метода run уединенной программы. Теперь можно набрать текст программы. Это стандартное окно редакто-ра ObjectPAL, поэтому можно редактировать, проверять синтаксис и отлаживать метод run, как и для любого другого объекта.

Из уединенной программы, как и из любого другого объекта, можно открывать и закрывать From a script, as from any other object, you can open and close other forms, create objects, get and set properties and values, display messages, and trigger methods.

С помощью команды Язык ObjectPAL | Методы можно открыть окно диалога "Методы". В этом окне можно описать переменные, константы, типы данных, процедуры, специальные методы и библиотеки динамической компоновки (DLL), необхо-димые для использования. Следует помнить, что все описанное доступно только для метода уединенной программы run.

После завершения редактирования закройте окно. На экран при этом будет выведен запрос на ввод имени для программы. Введите имя и нажмите кнопку ОК для сохранения программы на диске. Как и форму, программу можно записать и с помощью команды file / save или команды file / save as, а также упаковать командой Program / deliver (Язык ObjectPAL/| Упаковка.).
Записанные обычным образом программы можно изменять, упакованные программы - нельзя.

Добавление кода в программу

Для того чтобы добавить код в уединенную программу, следует выбрать команду file / open / script или file / new / script. Наберите текст добавляемого кода в открывшемся окне редактора. По завершении набора выберите команду file / save, для того чтобы записать текст и исполняемый код уединенной программы на диск. Для сохранения только исполняемого кода выберите команду Program / deliver (Язык ObjectPAL / Упаковка.).

С помощью окна диалога "Методы" и окна редактора ObjectPAL можно добавлять код в уединенную программу следующими способами:

- Определить программу для встроенных методов.
- Добавить специальные методы.
- Добавить специальные процедуры.
- Описать переменные, константы, типы данных и внешние программы.

Определение кода для встроенных методов

Каждая уединенная программа имеет следующие встроенные методы: run, error и status. (Status доступен только на профессиональном уровне функциональности ObjectPAL.) Можно определить код для этих встроенных методов, как и для любых других объектов. Это можно сделать в окне редактора.

Запустить эти методы можно интерактивно в Paradox, а также вызвав их из метода или процедуры на языке ObjectPAL.

Символ ; (точка с запятой) указывает, что далее следует коментарий.

Редактирование уединенной программы

Уединенную программу можно редактировать в окне редактора ObjectPAL. Для редактирования программы :

1. Выберите команду file / open / script и выделите программу для редактирования.
2. Установите флажок Design ("Конструктор") и нажмите кнопку OK. ObjectPAL откроет программу и выведет на экран в окно редактора встроенный метод run.
3. Воспользуйтесь редактором ObjectPAL для редактирования программы, как и для любого другого метода.

Редактор ObjectPAL

Редактор ObjectPAL поддерживает функции текстового редактора Windows вместе со специальными функциями для редактирования методов ObjectPAL. Редактор работает также с отладчиком, чтобы обеспечить интегрированную среду для создания, тестирования и исправления методов.

Редактор ObjectPAL работает одинаковым образом при работе с объектом, формой, библиотекой или с уединенной программой.

Запуск редактора

Для того чтобы запустить редактор в форме или в объекте формы:

1. Нажмите правую кнопку мыши для вывода меню свойств объекта.
2. Выберите пункт "Методы" или "Событие" в списке свойств. В окне диа-лога "Exploring" появится список методов, которые можно редактировать. На рис 1.4 показана последовательность действий: выбор поля "февр." - Object Explorer – Events (событие). В данном случае для поля "февр." можно написать

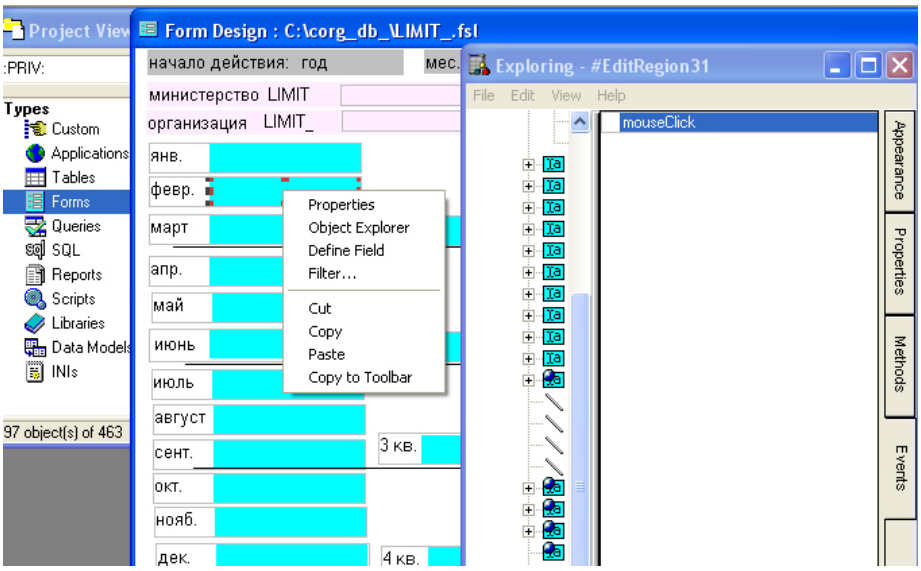


Рис. 1.4 Выбор метода

один метод для события "нажатие мыши"

3. После выбора метода появится окно редактора с некоторым стандартным текстом.

в библиотеке:
1. Откройте окно библиотеки и нажмите правую кнопку мыши.
2. После открытия окна диалога "Методы" выберите метод для редактирования.

в уединенной программе:
1. Уединенная программа автоматически открывается в окне редактора.
2. На первой строке располагается название метода, а на последней окончание метода. Курсор располагается на третьей строке, где можно начинать набор текста метода.

Редактор всегда находится в режиме вставки, и в нем доступны обычные средства редактирования.

Можно открывать сразу несколько окон редактирования.

Примечание: Переменные, константы и процедуры, описанные в окне метода, доступны только данному методу. Для того чтобы сделать переменные, константы или процедуры доступными для всех методов объекта, выберите необходимые из пунктов Uses, Type, Const , Var и Procs в окне диалога "Методы". Для каждого из них будет открыто свое отдельное окно в Paradox.

Работа в редакторе ObjectPAL

В окне редактора разрешаются обычные методы редактирования с двумя исключениями:
Редактор всегда находится врежиме вставки.
- Редактор не переносит автоматически строки текста на новую строку. Строка расширяется вправо по мере набора, до тех пор пока не будет нажата клавиша Enter для начала новой строки.
- Использование клавиатуры:

Ctrl+левая стрелка	перемещает курсор на одно слово влево.
Ctrl+правая стрелка	перемещает курсор на одно слово вправо.
Home	перемещает курсор к началу строки.
End	перемещает курсор к концу строки.
Ctrl+Home	перемещает курсор к началу текста.
Ctrl+End	перемещает курсор к концу текста.
Page up	перемещение на одно полное окно назад.
Page down	перемещение на одно полное окно вперед.
Backspace	удаляет символ слева от курсора.
Delete	удаляет символ справа от курсора.
Insert	не действует, так как редактор всегда находится в режиме вставки. При наборе символы проталкиваются вправо.
Ctrl+Insert	копирует выделенный текст во временный буфер.
Shift+Insert	вставляет текст из временного буфера в метод.
Tab	вставляет символ табуляции и смещает текст вправо.

Выделение текста:
- Для выделения слова нужно дважды нажать на нем клавишу мыши.
- Для выделения всей строки нужно нажать клавишу мыши слева от этой строки. (При этом вид курсора должен измениться из вертикальной черты в стрелку.)

Для выделения блока текста,
- нажмите кнопку мыши и удерживая ее перемещайте курсор
- нажмите клавишу Shift и используйте клавиши управления курсором
- нажмите кнопку мыши для указания начальной позиции, а затем зажмите клавишу Shift для расширения выделения

Панель управления редактора ObjectPAL

print
Кнопка "Печатать"
Нажмите кнопку "Печатать" для распечатки редактируемой программы.

Если Вы находитесь
- в уединенной программе - Paradox печатает всю программу
- в библиотеке без открытого в окне редактора метода - все программы данной библиотеки
- в библиотеке с методом, открытым в окне редактора - программу открытого метода.
- в форме с открытым окном редактирования метода - программу открытого метода.

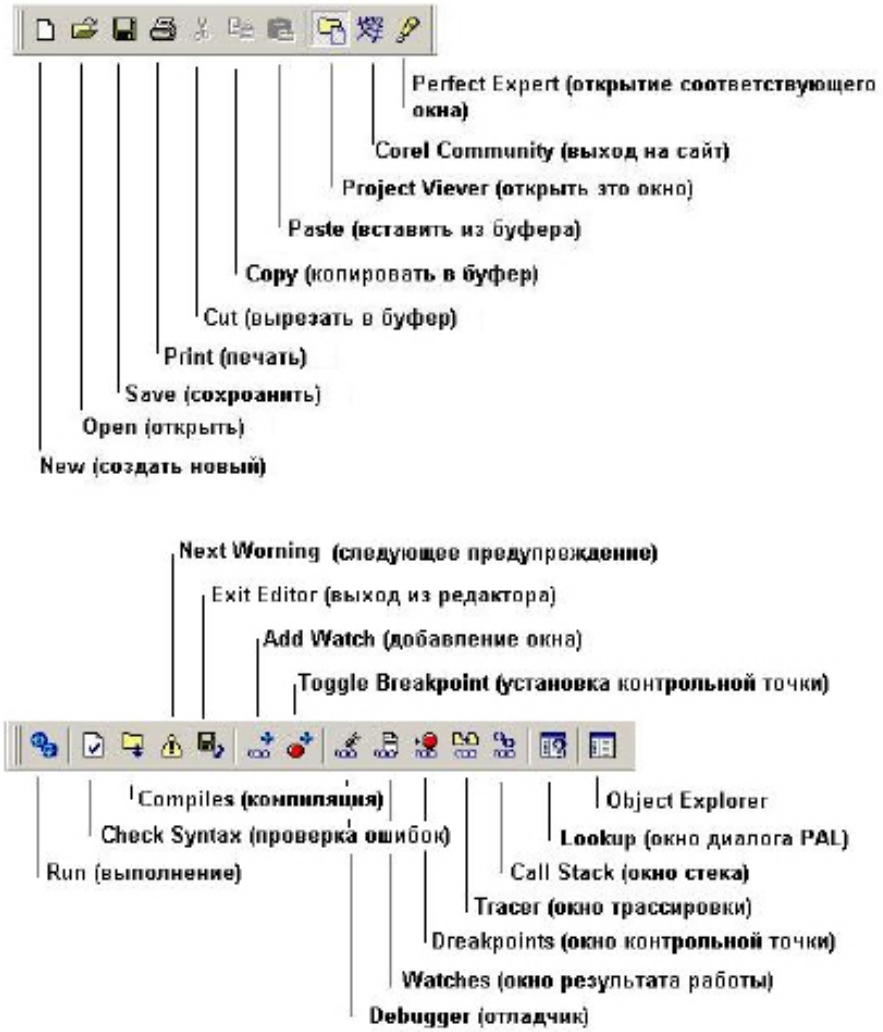


Рис 1.5 Панель управления редактора ObjectPAL

Cut (вырезание с помещением в буфер)
Кнопка "Переместить в буфер"
Нажатие кнопки "Переместить в буфер" на панели управления подобно выбору команды Редактирование / Вырезать из меню. Кнопка "Переместить в буфер" удаляет выделенный текст или объект и пемещает его в буфер Windows.
Затем можно использовать кнопку "Вставить из буфера" или команду Редактирование / Вставить для вставки содержимого временного буфера в другой файл, или в другое место данного файла.
Содержимое временного буфера не уничтожается после вставки, и его можно вставлять требуемое количество раз.
Для удаления выделенного фрагмента без изменения содержимого временного буфера нажмите клавишу Del или выберите команду Редактирование / Удалить.
Комбинация клавиш Shift+Del

Copy (копирование в буфер)
Кнопка "Копировать в буфер"
Нажатие кнопки "Копировать в буфер" на панели управления аналогично выбору в меню команды Редактирование / Копировать.
Нажмите кнопку "Копировать в буфер" для копирования выделенного текста или объектов во времен-

ный буфер, но не для удаления чего-либо из документа или запроса.
Для того чтобы вставить содержимое временного буфера в документ пользо-вателя используется следующее

- Команда Редактирование / Вставить
- Комбинация клавиш Shift+Ins
- Кнопка "Вставить из буфера"

Содержимое временного буфера не уничтожается после вставки, и его можно вставлять требуемое количество раз.
Комбинация клавиш Ctrl+Ins

Past (вставка из буфера)
Кнопка "Вставить из буфера"
Нажатие кнопки "Вставить из буфера" на панели управления аналогично выбору в меню команды Редактирование / Вставить.
Эффект от нажатия кнопки "Вставить из буфера" зависит от того, какое окно является активным, и от того, находится ли пользователь в режиме конструктора или просмотра данных.
Содержимое временного буфера не уничтожается после вставки, и его можно вставлять требуемое количество раз.
Комбинация клавиш Shift+Ins

Design (режим редактора)
Действие кнопки противоположено действию предыдущей.
Check Syntax (проверка синтаксиса)
Кнопка "Проверка синтаксиса"
Нажмите кнопку "Проверка синтаксиса", для того чтобы откомпилировать все методы в форме, уединенной программе или библиотеке. (а не только в текущем ок-не редактора). При обнаружении синтаксической ошибки откроется окно соответствующего метода, причем курсор будет располагаться или в текущей строке или в следующей строке метода, где ошибка, а в строку состояния будет выведено сооб-щение об ошибке.
Примечание: При изменении программы сразу в нескольких окнах редактора следует сохранить изменения перед проверкой синтаксиса. В противном случае, при проверке синтаксиса могут появиться непредвиденные ошибки, из-за того, что проверяется не самая последняя версия программы.

Compile (упаковка)
Делает компиляцию и проверку синтаксиса
Go to the next warning
Переход к следующему предупреждению. Следующее сообщение- переключает режим вывода предупреждений от компилятора. При установке данного флажка сообщения в строке состояния предупреждают о неописанных переменных и о других условиях, которые могут послужить причиной ошибки на стадии исполнения программы
Save source and exit the editor
Кнопка "Сохранить текст и выйти из редактора"
Нажатие кнопки "Сохранить текст и выйти из редактора" приводит к сохранению программы в памяти и закрытию окна редактора. Пользователь продолжает оставаться в окне конструктора.

Next warning
Переход к следующему предупреждению. Следующее сообщение - переключает режим вывода предупреждений от компилятора. При установке данного флажка сообщения в строке состояния предупреждают о неописанных переменных и о других условиях, которые могут послужить причиной ошибки на стадии исполнения программы

Exit the editor

Кнопка "Сохранить текст и выйти из редактора"
Нажатие кнопки "Сохранить текст и выйти из редактора" приводит к сохране-нию программы в памяти и закрытию окна редактора. Пользователь продолжает ос-таваться в окне конструктора.

Add watch
Открытие нового окна просмотра переменных

Toggle breakpoint
Установка (снятие) контрольной точки

В режиме отладки можно установить контрольную точку. При выполнении ме-тодов программа дойдет до этой контрольной точки и далее можно будет по шагам проверять ее работоспособность.

Debugger
Открывает окно отладчика.
Если установлена контрольная точка, то это окно открывается автоматически
stop execution - остановка выполнения и переход в отладчик

call stack window, tracer window, the breakpoint window, the watches window
Кнопки открытия (закрытия) соответствующих окон

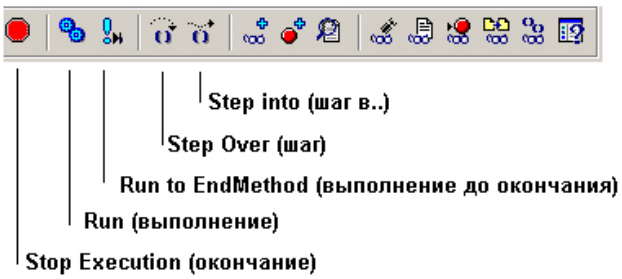


Рис 1.6 окно отладчика

Точечная нотация

В ObjectPAL используется метод точечной нотации. Это когда связь метода и объекта осуществляется через точку (в тексте программы выделено красным):

```
method pushButton(var eventInfo Event)
var
  tc1  TCursor
  samp DataBase
  tbl  Table
endVar

addAlias("SampleTables", "Standard", "c:\\pdoxwin\\sample")

samp.open("SampleTables")

tbl.attach("Customer.db", samp)
tbl.setIndex("NameAndState")

if not tc1.open(tbl) then
   errorShow()
endIf

endMethod
```

Программы входа в директории.

Как отмечалось в первой части книги, в Paradox существуют рабочий каталог (:work:) и личный (:priv:). В рабочем хранятся все основные базы, а в личном промежуточные результаты. Такой подход дает возможность работать в многопользовательском режиме (сетевой режим). Если ваша задача не расчитана на работу в сети, то вполне достаточно просто определить рабочий каталог. Когда работаете сами, то определить рабочий каталог можно в окне Project Viewers (там содержится более десятка ва-

риантов и все новые автоматически запоминаются).

Для сетевого варианта надо помнить, что все пользователи Paradox должны в настройках BDE (Borland Database Engine)(или IDAPI Configuration Utility для версии 7) иметь одинаковый сетевой путь для файла pdoxusrs.net

Глава 2. Примеры использования PAL

Установки Paradox

Для начала надо определить среду, в которой будем работать. Для этого надо вызвать окна настроек. Два будут описаны ниже, а настройки BDE в приложении, т.к. они относятся не только к Paradox, а полностью к Windows.

На рис.2.2. показано окно Developer Prefereces. Слева устанавливает "полный" перечень заполнения справочных окон, а правая перекрашивает коментарий (то, что в тексте программы после точки с запятой)

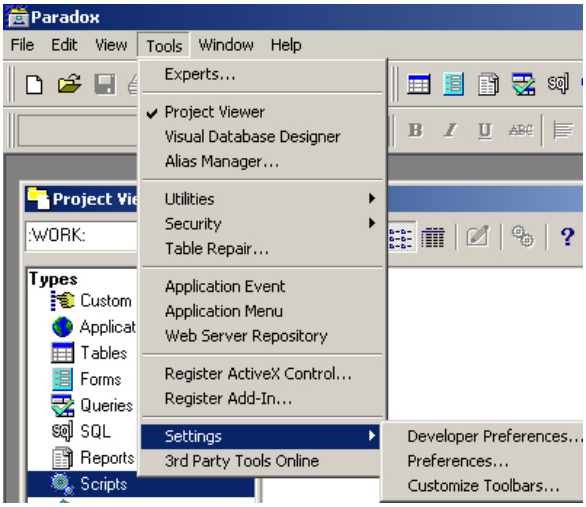


Рис.2.1 Вызов окон настроек внутри Paradox

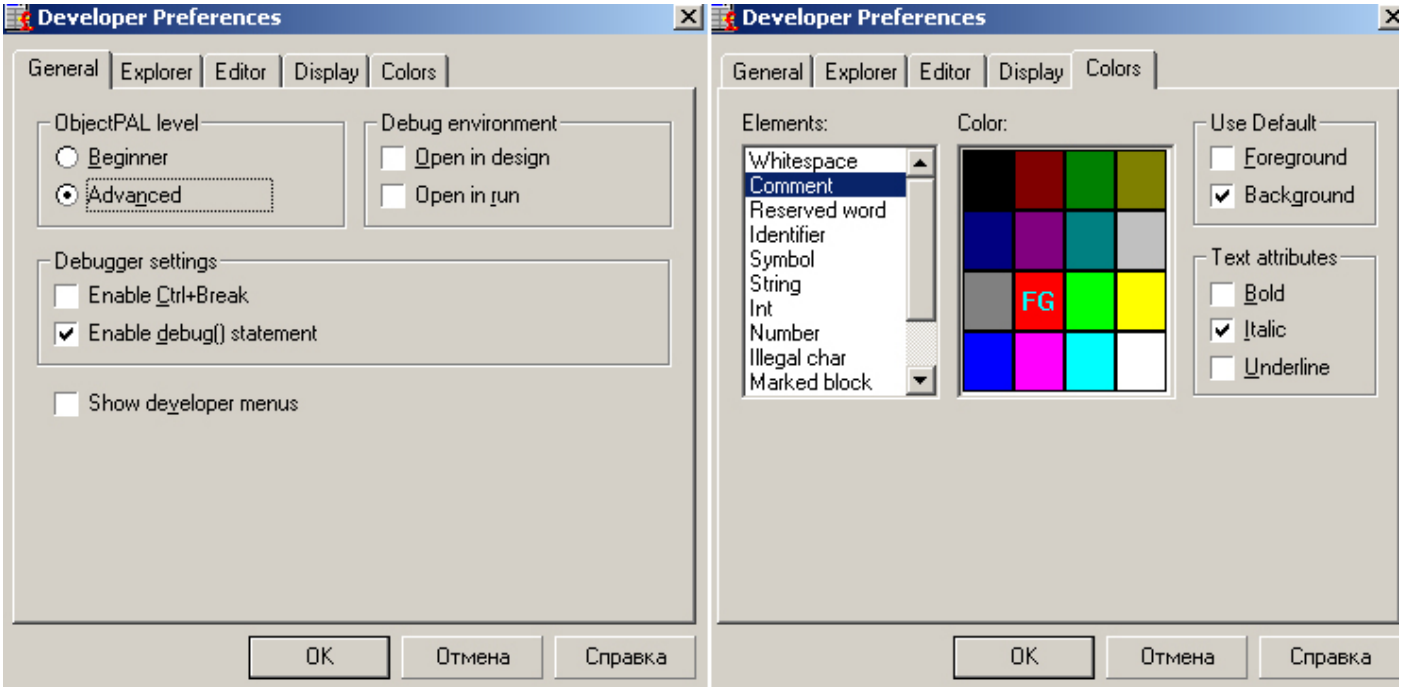


Рис.2.2. Окно Developer Prefereces

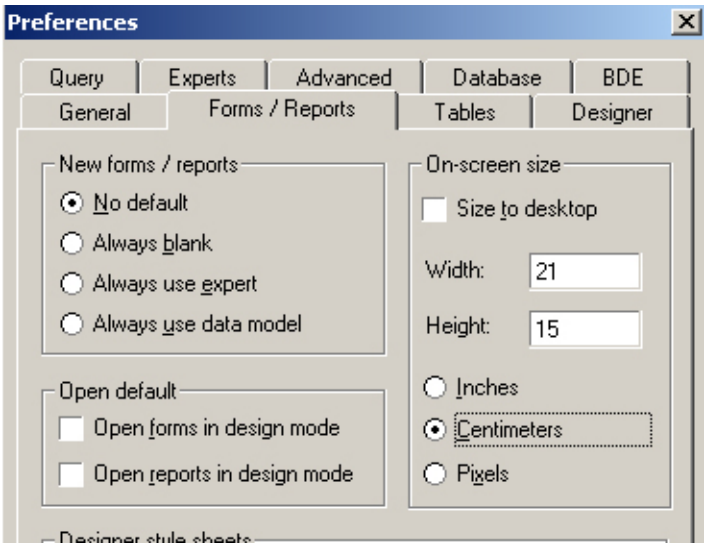


Рис.2.3. Начальная установка размеров экранной формы

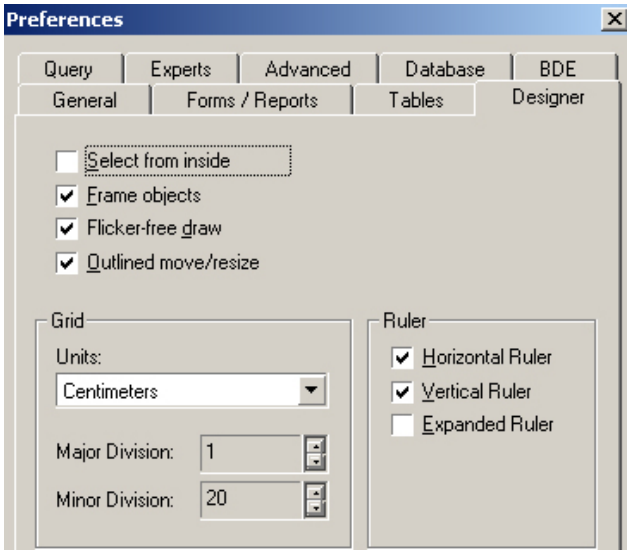


Рис.2.4.Установка размерности для линеек

На рис.2.5. сначала показан адрес "личного" каталога. Этот адрес можно менять программно.

Далее идет очень важный параметр - пустое поле равносильно нулю. Дело в том, что любая цифра загромождает форму (отчет), а вычисляемые поля всегда имеют цифру. Для устранения этого неудобства и устранения возможных ошибок и устанавливается этот параметр.

Последним указывается сетевое имя. Адрес общего сетевого ресурса устанавливается в BDE

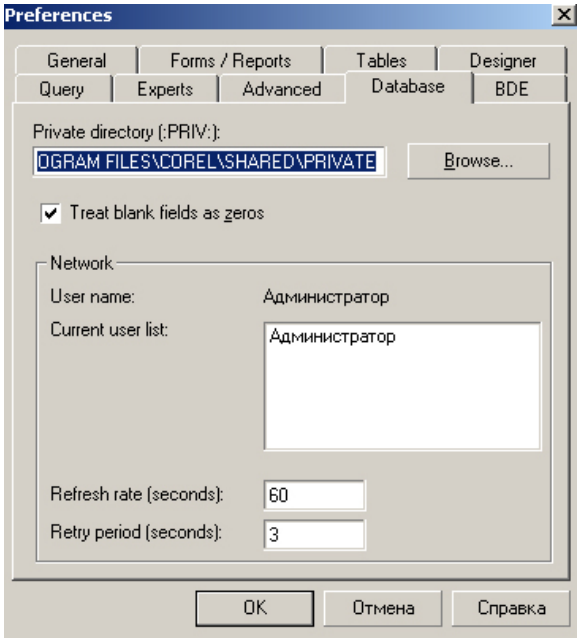


Рис.2.5 Установки для таблиц

Программа начала работы

Следующая программа устанавливает рабочий директорий g:\\2fiz и открывает начальную форму для задачи, которая находится в этом директории. Для удобства обычно выносят иконку задачи на рабочий стол. Эта программа и необходима для запуска программы. Если в программе встречается точка с запятой, то далее в этой строке все воспринимается как комментарий.

При написании программы почти все операторы можно брать из тех примеров, которые приведены в Paradox. Вызов справки показан на рис. 2.6.

Перечень базовых команд помещен в приложении, однако в зависимости от версии Paradox этот перечень может быть дополнен.

При выборе типа также можно не набирать ни одной буквы. Для этого после ввода названия переменной (в нашем случае f99) надо, находясь на свободном месте нажать правую кнопку мыши выбрать ObjectPAL Quick Lookup (рис.2.7)

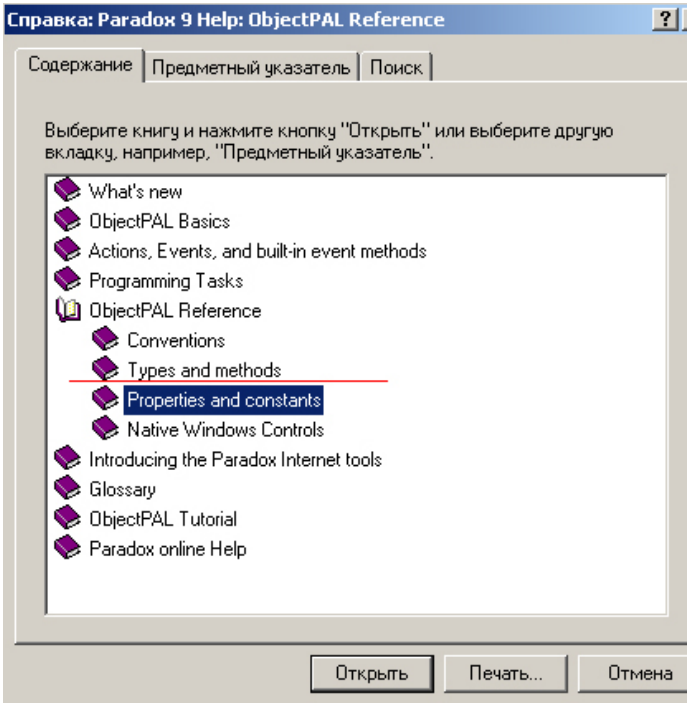


Рис.2.6 вызов справки по синтаксису команд и методов

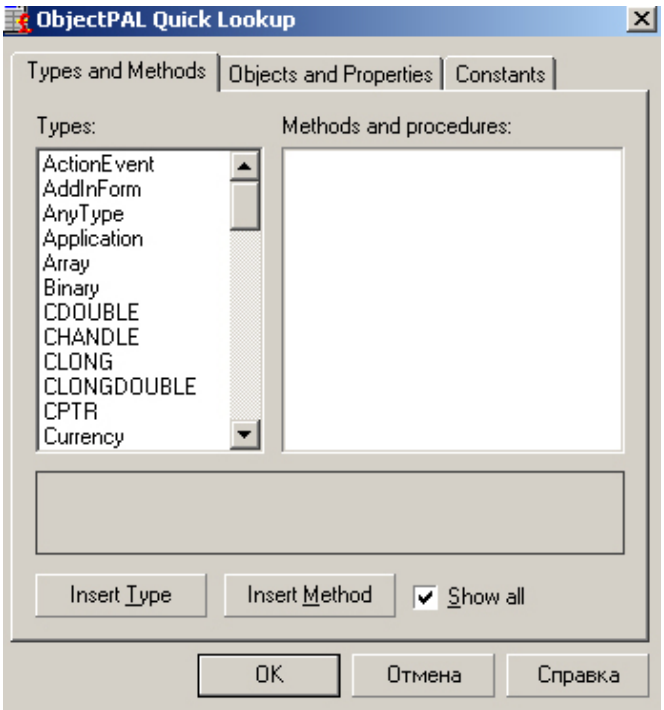


Рис.2.7 выбор типа переменной

В методе, который приведен ниже, при указании пути необходимо отметить использование знака обратной косой черты:

- Включите в строковую константу код ASCII требуемого символа, перед которым поставьте знак \.

Ниже приведены несколько строк с использованием обратной косой черты (\).

```
"Quote mark coming\"        ; текст заканчивается кавычками
"Backslash   away\\"        ; текст заканчивается обратной
                            ; косой чертой
"Give me a Ctrl-R:\018"     ; текст заканчивается символом
                            ; Ctrl-R
"\012"                      ; символ перевода страницы
```

Ниже приведен текст самой программы.

```
method run(var eventInfo Event)
var
f99 form
endvar
;==============
close()
setworkingdir("c:\\avtos\\")
errorshow()
sleep(1500)

setprivdir("c:\\avtos_\\")
sleep(1500)

f99.open(":priv:_nach")
f99.wait()

endmethod
```

Рассмотрим более подробно действие каждого оператора.

close() – закрывает все рабочие окна и подготавливает Paradox к изменению рабочего каталога

setWorkingDir("c:\\avtos") - установить рабочий каталог C:\avtos

errorshow() - при возникновении ошибки - она выводится на экран

sleep(1500) - задержка нужна для стабилизации процесса

method run(var eventInfo Event)

setprivdir("c:\\avtos_\\") - ecnfyjdrf kbxyjuj rfnfkjuf

f99.open(":priv:_nach") - открытие формы _nach из личного каталога

f99.wait() - ожидание принудительного закрытия формы

В дальнейшем иконку файла этого скрипта можно поместить на рабочий стол и ваше приложение Paradox всегда будет само устанавливать для себя необходимые каталоги.

В заключении остановимся на самой первоначальной форме. Ее вид может быть произвольный. Вы можете организовать начальную форму в виде окна Word, создать рекламу, а можно просто вывести пустую форму с кнопками, которые называются в соответствии с их функциональными значениями.

Дополнительно надо обратить внимание на то, что в работе все рабочие формы делаются в диалоговом окне. Это сделано для того, чтобы скрыть все рабочие кнопки и придать самой форме приличный вид.

В процессе отладки всегда какая-то форма открыта и если вы забыли перевести форму в диалоговый режим, при выполнении скрипта она откроется, но откроется под последней формой, представленной в диалоговом режиме и вы ее не увидите. В этом случае стандартным режимом комбинацией клавиш Alt+F4 окно закрывается и делаются соответствующие исправления.

Вариант первоначальной формы показан на рис.2.8.

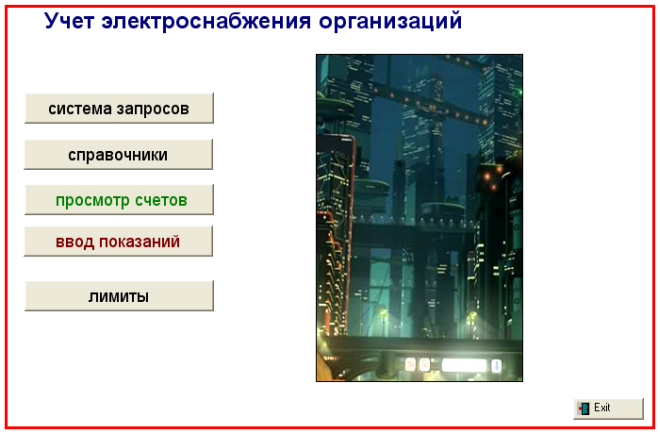


Рис.2.8 вариант первоначальной формы

Если в процессе создания программы размеры форм не менять, то они будут постоянные и вновь открытая будет скрывать предыдущую.

Т.е. если оставить эту начальную форму открытой на все время работы в задаче, то это позволит при помощи стандартной кнопки "Exit" выйти не только из задачи, но и из Paradox.

Все остальные кнопки - пользовательские. Для любой из этих кнопок вставка метода производится следующим образом:

1. ставится сама кнопка
2. закрывается окно выбора стандартных методов
3. открывается окно Explorer
4. выбирается метод для события "нажатие кнопки"
5. вносится текст самого метода.

Для вормирования стандартного метода (в данном случае Exit) выбираем:

- (левая часть рис.2.9) категорию
- действие
- нажимает Next и получаем следующее окно (правая часть рисунка)

В верхнем окне указано наименование кнопки. В данном случае показан стандартный текст, но написать туда можно любую фразу.

Потом три строки определяюттип, стиль, комментарий и последняя выбирает иконку. В данном случае тоже используется стандартная, но можно вставить свою или вообще не ставить.

Изменить все параметры кнопки можно в любое время. Например, иконка и текст являются независимыми объектами, а сама кнопка для них - контейнер.

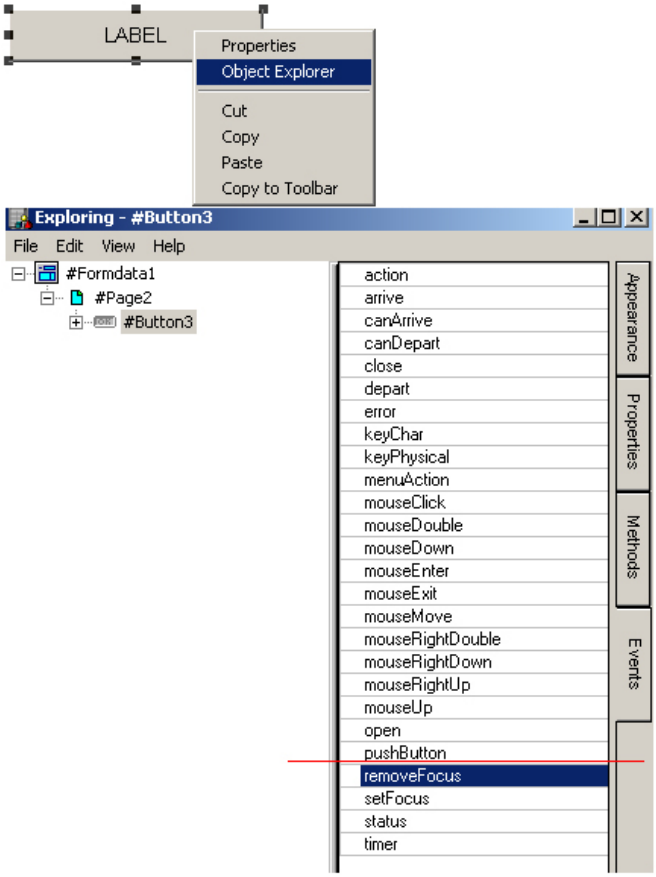


Рис.2.8. окно Object Explorer

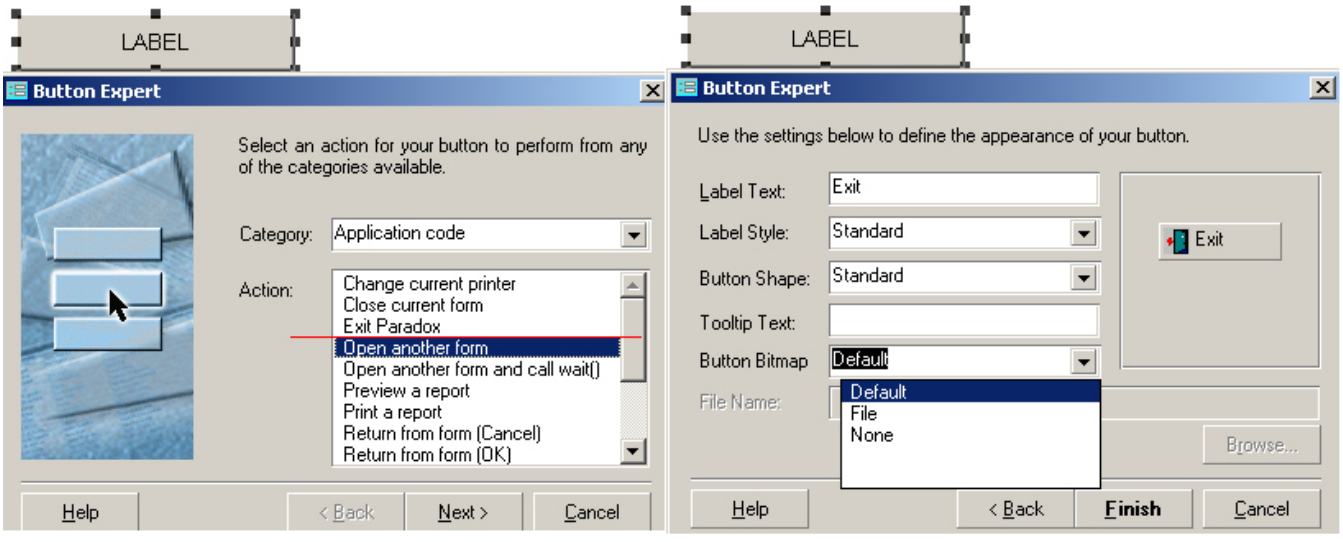


Рис. 2.9. Формирование стандартного метода кнопки

Стандартные методы можно редактировать. Например, кнопка, которая часто используется - "Удалить запись". Алгоритм удаления построен так, что перед удалением задается вопрос - "уверены ли вы". Стандартно этот вопрос имеет выд на рис. 2.10.

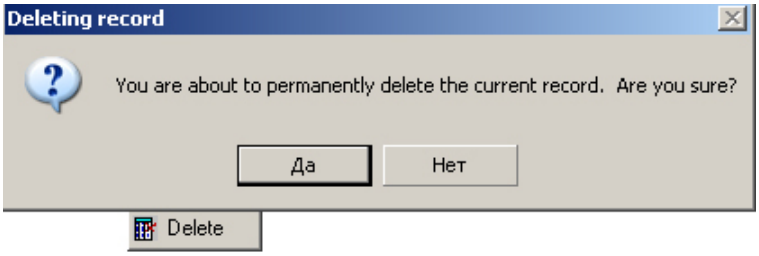


Рис.2.10 Процесс удаления записи

Листинг метода :

```
method pushButton(var eventInfo Event)
var
    lEditState          Logical          ;// tmp handle to the edit state
endVar
    ;// Get edit state and store it for later use
    lEditState = isEdit()
    ;// Toggle edit mode if necessary.  If deletion should be prevented while not
    ;// in edit mode, uncomment the three lines preceding edit() and comment or
    ;// delete the edit()
    if lEditState = FALSE then
;       msgStop("Error deleting record",
;           "You cannot delete a record while not in edit mode.")
;       return
        if NOT edit() then
            errorShow()          ;// custom error handling can go here
            return
        endIf
    endIf
    ;// Confirm the deletion
    if msgQuestion("Deleting record",
        "You are about to permanently delete the current record.  Are you sure?") = "Yes" then
        if NOT active.deleteRecord() then
            switch
                ;// Check for insufficient rights problems
                case    errorHasErrorCode(peNoTableRights) OR
                        errorHasErrorCode(peNotEnoughRights) OR
                        errorHasErrorCode(peNotSuffSQLRights) :
                            msgStop("Error deleting record",
                    "You have insufficient rights to delete records from this table.")
                ;// Edit mode, should not happen due to above code, but
                ;// a good check
                case errorHasErrorCode(peNotInEditMode) :
                msgStop("Error deleting record", "You must be in edit to delete a record.")
                ;// Further custom error handling can go here

                otherWise :
                    errorShow()
            endSwitch
        endIf
    endIf
    ;// Restore the edit state, if necessary
    if lEditState = FALSE then
        endEdit()
    endIf
endMethod
```

Синим выделена фраза, которая выводится в сообщении. Заменим эту фразу:

```
if msgQuestion("Deleting record",
        "Запись под курсором будет удалена. Нужно ли это???????") = "Yes" then
        if NOT active.deleteRecord() then
                switch
```

В результате получим:

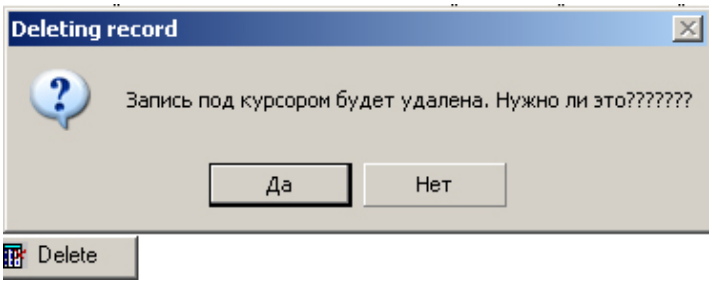


Рис.2.11. Измененное сообщение

Использование кнопок для изменения данных

В некоторых случаях процесс ввода с клавиатуры можно заменить программой, которая будет автоматически изменять данные и производить контроль достоверности. Такой принцип позволяет избежать возникновение различного рода ошибок таких как ошибочное нажатие клавиши, ввод неверного формата и т.п.

В качестве простого примера возмем задачу, когда надо задать год и месяц для какого - либо расчета. В качестве элементов управления будем опять использовать кнопки. Но рис.2.12 приведена часть формы для этой задчи:

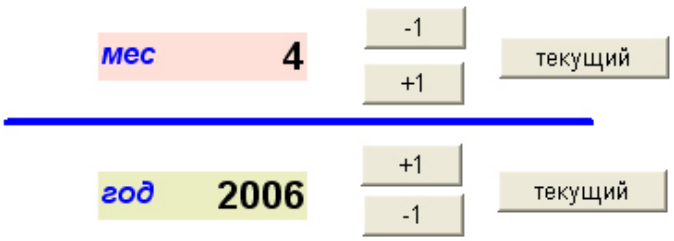


Рис.2.12 выбор года и месяца

```
method pushButton(var eventInfo Event)
;кнопка установки текущего месяца
var
t tcursor
mm,gg SmallInt
d date
endvar
;-----------
d=today()                ;переменной d присваивается сегодняшнее число
t.open(":priv:mmgg")     ;таблица открывается
t.edit()                 ; включается режим редактирования
  t."mm"=d.month()       ; полю mm присваивается значение текущего месяца
t.endedit()              ; режим редактирования закрывается
t.close()                ;таблица закрывается

endMethod

method pushButton(var eventInfo Event)
; кнопка увеличения месяца на 1
var
t tcursor
mm,gg SmallInt
```

```
endvar
;-----------
t.open(":priv:mmgg")
t.edit()
mm=t."mm"
if mm>11 then  t."mm"=1 ;делается анализ до 12 включительно
        else  t."mm"=t."mm"+1
endif
t.endedit()
t.close()

endMethod
```

Аналиц на 12-й месяц делается оператором **if,** где производится сравнение, т.е. если месяц равен 12, то следующий будет 1, а в промежутке от 1 до 11 будет добавляться 1

Методы других кнопок почти не отличается от приведенных методов. Все отличие только в том, что переменная mm не увеличивается, а уменьшается на единицу.

На практике такие методы никогда не набираются в отдельности. Делается все много проще. Когда мы оформили полностью первую кнопку, она выделяется в форме и помещается в буфер обмена. Следующим этапом производится считывание из буфера обмена и в результате получаем две одинаковые кнопки. Потом изменяем название кнопки. Для этого ставим курсор на запись, выделяем ее первым нажатием на левую кнопку мыши, вторым нажатием преобразуем курсор (устанавливается режим редактирования записи) и исправляем запись.

Для изменения метода выделяем кнопку нажатием на левую клавишу мыши, правой открываем ее свойства, выбираем Object Explorer / Events / PushButton, и двойным нажатием на левую клавишу мыши по необходимому методу (в нашем случае pushbutton и он отмечен звездочкой) открываем сам метод. Дальше просто изменяем текст метода. Сохранение изменений можно не делать, т.к. при закрытии формы Paradox сам предложит записать изменения или проигнорировать, т.е. аналогично работе с Word

Методы в форме со справочниками

Для устранения возможных ошибок при вводе данных используются справочники. На рис.2.13 показана форма для ввода путевых листов автопредприятия. Модель данных для этой формы показана на рис.2.14. Как видно из модели - в форму входят пять, не связанных менжду собой, баз данных.

На рис.2.13 видно, что при вводе данных водителя и характеристики транспорта можно допустить ошибки. Чтобы избежать этого, используются справочники.

Рассмотрим поле "таб№". В Run Time свойствах поля (рис.2.15)запрещаем остановку курсора в этом поле (метка Tab Stop) и дополнительно запрещаем редактирование этого поля (метка Read Only).

Потом открываем окно Object Explorer / события. Интерес представляет событие "нажатие мыши". Т.е. если по полю кликнуть мышкой, то должно произойти событие. Метод этого события приведен ниже:

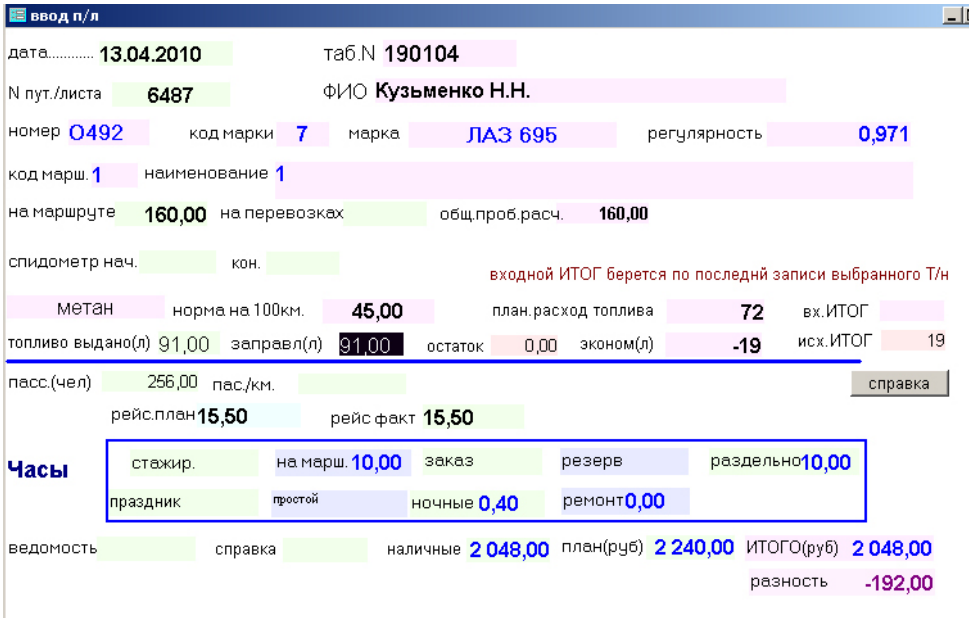


Рис.2.13. Форма для ввода путевых листов

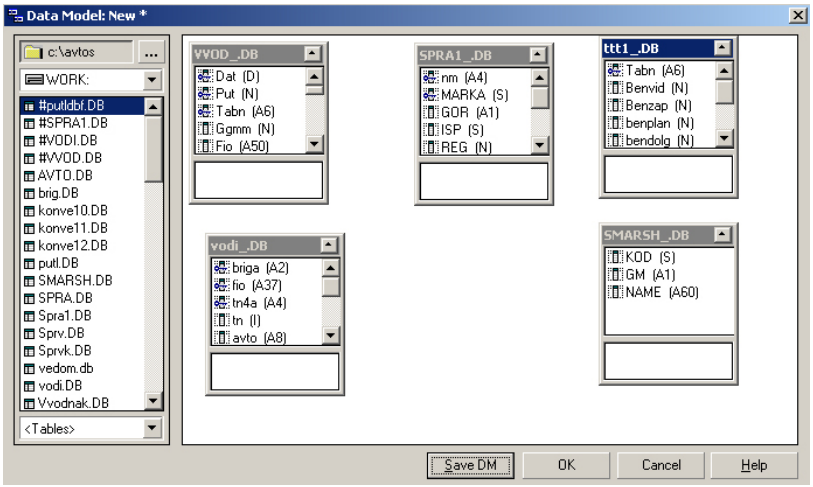


Рис.2.14 модель данных

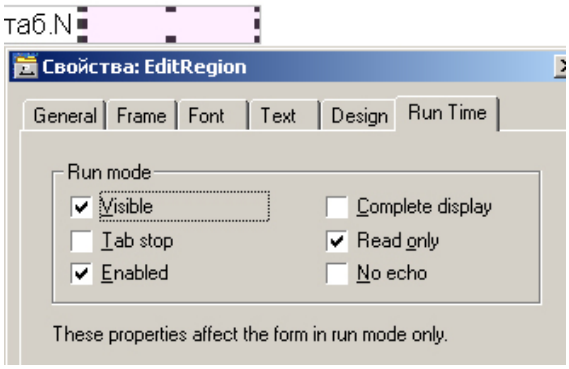


Рис.2.15 Run Time свойства поля

```
method mouseClick(var eventInfo MouseEvent)
var
t,t1,t2 tcursor   ; определяем переменные для табличного курсора
f form            ; переменная для формы
endvar
  endedit()       ; выходим из режима редактирования
f.open(":priv:vodi")   ;открываем форму для справочника водителей
f.wait()                ;ожидаем принудительного закрытия

t.open(":priv:vvod_")
t.edit()                      ;включили режим редактирования
t1.open(":priv:vodi_")
t."tabn"=t1."tna"             ;заменили поля одной таблицы на толя другой
t."fio"=t1."fio"
t1.close()                    ;закрыли таблицу переменной t1
t.endedit()                   ;закрыли редактирование
t.close()                     ;закрыли таблицу переменной t
;=================================
ttt=2
 executeqbefile("top355t3",":priv:ttt3");SELECT MAX(D.Dat) AS maxd        from vvod
                                        ;FROM ":PRIV:Vvod.DB" D, ":PRIV:VODi_.DB" D1
                                        ;WHERE (D1.Tna = D.Tabn)
 errorshow()
        t1.open(":priv:ttt3")
        if t1.isempty() then ttt=ttt-1 ; если в текущем месяце записей не было то
                        executeqbefile("top355t4",":priv:ttt3")  ;from vvodnak
        endif
                t1.close()
t.open(":priv:ttt1_")
t.edit()
t.empty()
t.insertrecord()
t1.open(":priv:ttt3")
        if t1.isempty() then t."benitn"=blank()      ;    если остатка нет, то поле очищается
                else
                if ttt=2 then executeqbefile("top355x",":priv:ttt2")      ;last from  vvod
                                errorshow()
                endif
```

```
                if ttt=1 then executeqbefile("top355y",":priv:ttt2")      ;last from  vvodnak
                                errorshow()
                endif
t2.open(":priv:ttt2")
t."benitn"=t2."benitk" ;остаток предыдущий равен началу текущей записи
t2.close()
          endif
t.endedit()
t.close()
endMethod
```

Метод разбит на две части. В первой формируется зама запись, полученная из справочника водителей. заканчивается эта часть двойной чертой. Потом идет вторая часть, которая выбирает остатки топлива по этому табельному номеру.

Сначала выбирается дата последней записи путевого листа этого табельного номера в текущем месяце. Если записей не было, то аналогичная выборка делается в архиве.

Если и в архиве отсутствуют записи - поле обнуляется (очищается). Если дата определена, то по ней выбирается вся запись и из этой записи остаток переписывается в текущую как начальное значение долга (остатка) по топливу.

Работа с памятью

Array -

Массив содержит значения (называемые элементами) в ячейках подобно тому, как почтовые ячейки содержат почтовые сообщения. В языке ObjectPAL существуют только одномерные массивы, подобные единственному ряду ячеек, в котором каждая ячейка содержит по одному элементу. Тип Array наследует также методы, определенные для типа AnyType.

Для использования в методах массивы должны быть описаны через указание имени, длины (числа элементов) и типа данных для элементов.

Примечание: В языке ObjectPAL элементы массива нумеруются с 1, а не с 0, как в ряде других языков.

Примечание: В языке ObjectPAL существуют также динамические массивы.DynArray.

addLast	Добавляет элемент в конец массива переменной длины.
append	Добавляет содержимое массива в конец другого массива.
contains	Осуществляет поиск символьного шаблона среди элементов массива.
countOf	Подсчитывает, сколько элементов с данным значением содержится в массиве.
empty	Очищает массив.
exchange	Обменивается содержимое двух ячеек массива.
fill	Заполняет массив одним значением.
grow	Увеличивает длину массива переменной длины
indexOf	Возвращает позицию элемента в массиве.
insert	Вставляет одну или несколько пустых ячеек в массив.
insertAfter	Вставляет элемент данных в массив переменной длины после указанного элемента.
insertBefore	Вставляет элемент данных в массив перед указанным элементом
insertFirst	Вставляет элемент в начало массива.
isResizeable	Сообщает, можно ли изменить длину массива.
remove	Удаляет один или несколько элементов из массива.
removeAllItems	Удаляет из массива все элементы с заданным значением.
removeItem	Удаляет из массива элемент с заданным значением.
replaceItem	Заменяет значение элемента массива на новое.

setSize	Задается длина массива.
size	Возвращает число элементов в массиве.
view	Выводит содержимое массива в окно диалога.

DynArray -

Тип DynArray определяет динамический массив, то есть массив с изменяемой структурой. Динамический массив является компактной запоминающей структурой для любой комбинации типов данных. DynArray позволяет организовать быстрый поиск значений даже в динамических массивах, содержащих большое число элементов.

Эти массивы называются динамическими, так как от пользователя не требуется задание их размеров; длина динамического массива автоматически изменяется при добавлении или удалении элементов. Размер динамического массива ограничен только системной памятью.

В отличие от массива фиксированной длины индексы динамического массива не являются целыми числами. Индексами могут служить любые допустимые выражения ObjectPAL, значения которых преобразуются к типу String. Каждый индекс в динамическом массиве связывается со значением. Тип DynArray наследует также методы, определенные для типа AnyType.

contains	Осуществляет поиск заданного значения среди индексов динамического массива.
getKeys	Загружает индексы существующего динамического массива в массив переменной длины.
removeItem	Удаляет указанный элемент из динамического массива.
size	Возвращает число элементов в динамическом массиве.
view	Выводит содержимое динамического массива в окно диалога.

Более детально с работой массивов можно ознакомиться в соответствующих разделах помощи. Далее пародолжим разбирать реальную ситуацию в форма выбора водителей.

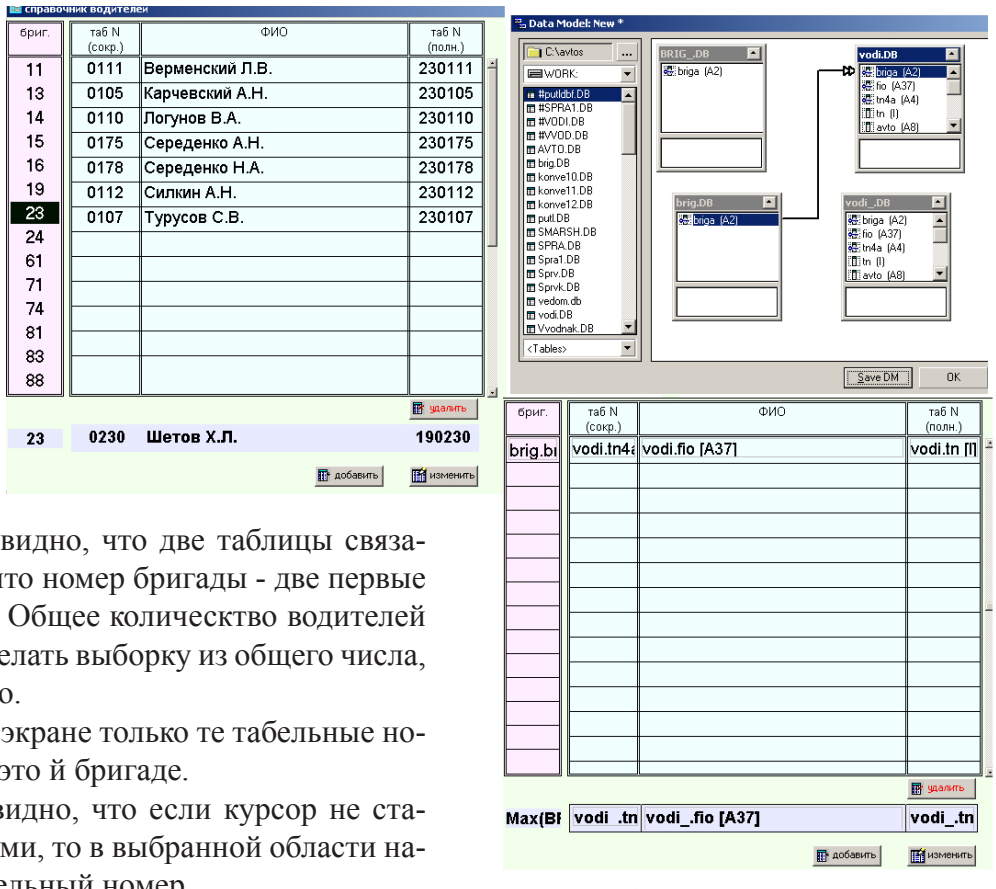


Рис.2.16. Форма для выбора водителей

Из модели данных видно, что две таблицы связаны. Это обусловлено тем, что номер бригады - две первые цифры табельного номера. Общее колическтво водителей довольно большое и если делать выборку из общего числа, то это будет затруднительно.

В данном случае на экране только те табельные номера, которые относятся к это й бригаде.

На рисунке также видно, что если курсор не ставился в таблицу с водителями, то в выбранной области находится совсем другой табельный номер.

Попробуйте исправить эту недоработку самостоя

тельно.

Работает форма следующим образом:

- при установке курсора (или клика мышкой) в поле (одинаково для базы бригад и водителей) содержимое этого поля выбирается в рабочую базу, данные из которой потом переписываются в путевой лист. В этой же форме можно добавлять, изменять или удалять записи.

На рис.2.17 показано, какие два события обрабатываются для вуделенного поля.

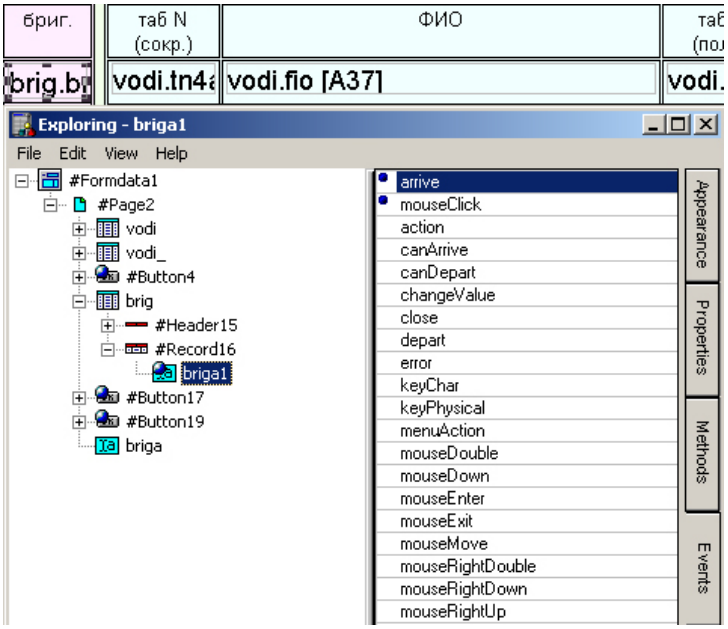


Рис.2.17 Окно Exploring для поля базы бригад

Для выбора бригад:

```
method arrive(var eventInfo MoveEvent)
var
t tcursor              ; переменная для табличного курсора
a array[]AnyType   ;переменная для массива, тип произвольный, скобки квадратные [  ]
endvar

t.open(":priv:brig_")    ; открыли рабочую базу бригад
t.edit()                 ; перевели в режим редактирования
t.empty()                ; очистили ее
brig.copytoarray(a)      ;запись под курсором копируется в область памяти
t.insertrecord()         ; в рабочей базе вводится режим дозаписи
t.copyfromarray(a)       ; из области памяти копируется в рабочую базу
t.endedit()              ; окончание радактирования
t.close()                ; закрытие рабочей базы

endMethod
```

Для выбора водителей:

```
method arrive(var eventInfo MoveEvent)
var
t,t1,t2 tcursor
a array[]AnyType
endvar

t.open(":priv:vodi_")  ; открывается база vodi_, которая находится в личном каталоге
t.edit()
t.empty()
t1.open(":priv:vodi1")
t1.edit()
t1.empty()
vodi.copytoarray(a)
t.insertrecord()
t.copyfromarray(a)
t1.insertrecord()
t1.copyfromarray(a)
```

```
t.endedit()
t.close()
t1.endedit()
t1.close()
endMethod
```

Как видно метод выбора водителей отличается от метода выбора бригад тем, что формируется две рабочие базы, которые необходимы для метода кнопки *"изменить":*

```
method pushButton(var eventInfo Event)
; кнопка -изменить- в форме выбора водителей
var
t,t1 tcursor    ; две переменных для табличного курсора
f form          ; переменная для формы
k SmallInt      ; переменная для короткого целого
endVar

f.open(":priv:vodi1")          ; открыли форму из личного каталога
f.action(databeginedit)        ; включили режим редактирования
f.wait()                       ; ждем принудительного закрытия

k=0
t1.open(":priv:soob")       ;открыли таблицу "soob"
t1.edit()
t1.empty()                     ; очистили ее
t.open(":priv:vodi_")          ; открыли рабочую базу водителей
t.edit()
        if t."briga".isblank() then k=1    ; если поле "бригада" пустое, то в базу soob добавляется запись
                        t1.insertrecord()
                        t1."soob"="не введен номер бригады"
        endif
        if t."tn4a".isblank() then k=1      ; tn4a -поле табельного номера
                        t1.insertrecord()
                        t1."soob"="не введен табельный номер"
        endif
        if t."fio".isblank() then k=1         ; поле фамилии
                        t1.insertrecord()
                        t1."soob"="не введена фамилия"
        endif
        if k>0 then    t.endedit()   ; если есть ошибка то
                        t.close()
                        t1.endedit()
                        t1.close()
                                f.open("priv:soob") ;открытие формы с сообщениями ошибок
                                f.wait()
                else      ; иначе (т.е.ошибок нет)
                t."tna"=t."briga"+t."tn4a"    ;вычисляется полный табельный номер
                t."tn"=LongInt(t."tna")        ;поле из текстового переводится в формат длинного целого
                t."brig"=LongInt(t."briga")  ;;поле из текстового переводится в формат длинного целого
                t."tn4"=LongInt(t."tn4a")    ;;поле из текстового переводится в формат длинного целого
                                        t.endedit()
                                        t.close()
```

```
                                    t1.endedit()
                                    t1.close()
        executeqbefile("vodi1")     ; запрос может выполняться только с таблицами, которые не в режиме
;редактирования
        errorshow()
         add(":priv:vodi_","vodi") ;добавление отредактированной записи в основную базу
         errorshow()
         t1.open(":priv:vodi_")
         t.open(":priv:brig_")      ;рабочая база бригад
         t.edit()
         t.empty()                   ; ее очистка
         t.insertrecord()
         t."briga"=t1."briga"       ;замена поля из базы водителей
         t.endedit()
         t.close()
         t1.close()
         add(":priv:brig_","brig")  ; добавление номера бригады в основную запись
         errorshow()
endif
endMethod
```

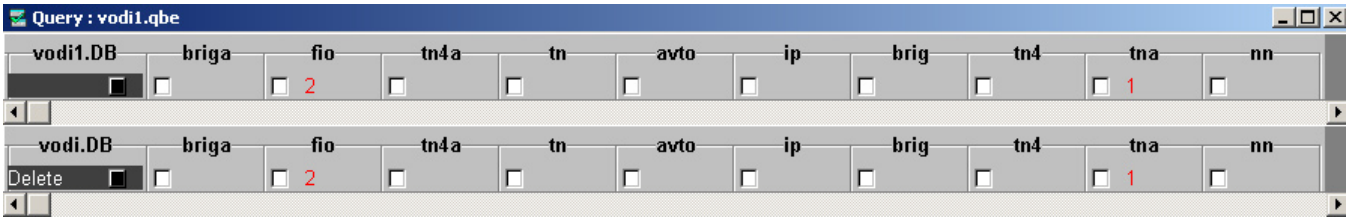


Рис.2.18 запрос для удаления записи из базы водителей

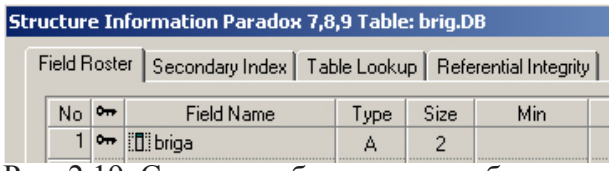


Рис. 2.19. Структура базы номеров бригад

При анализе возникает вопрос: почему перед добавлением записи в базу водителей выполняется запрос, который удаляет предыдущую запись, а пере добавлением номера бригады такого удаления нет. Дело в том, что база номеров бригад состоит из одного роиндексированного поля (рис.2.19)., а индекс подразумевает, что запись по этому поля уникальна.

Вычисления

Немного выше уже производили сложение двух алфавитных полей:

```
t."tna"=t."briga"+t."tn4a"    ;вычисляется полный табельный номер
```

Аналогично производятся вычисления и для цифровых полей. В форме на рис.2.13 поле "исх. ИТОГ" является вычисляемым На рис.2.20 показан синтаксис вычисляемого поля. В формуле используются круглые скобки, знаки арифметических действий. Определение полей через точечную нотацию:

- наименование и место положения базы
- точка
- наименование поля

Для написания формулыполя лучше вводить в автоматическом режиме:

- выбирается поле из таблицы, которая входит в модель данных
- нажимается клавиша Copy Field (в результате поле копируется в формулу)
- вводятся скобки и арифметические знаки

При выходе из формулы производится проверка на ошибки. Поле нельзя покинуть, если есть

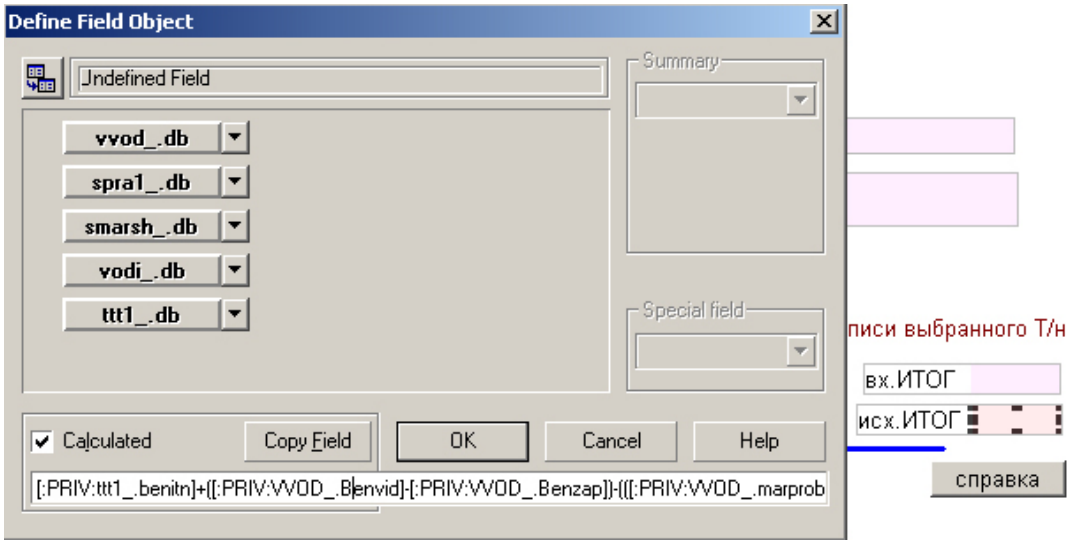


Рис.2.20 Вычисляемоек поле в форме

ошибки.

Примечание: анализ на наличие ошибое делается в каждом поле. Наиболее частыми ошибками являются :

- ввод несуществуюшей даты (30.02.2000)
- в качестве десятичного разделителя используется запятая вмето точки

В этом случае курсор с поля не уходит.

Вычисляемое поле в форме никак не связано с методом. Т.е. оно служит только для контроля ввода. Когда форму закрыли, то все расчеты надо повторить, но уже в самом методе. Ниже приведен метод открытия и последующей обработки формы ввода путевух листов (рис.2.13)

```
method pushButton(var eventInfo Event)
;кн.ДОБАВИТЬ в форме просмотра п_л
var
t,t1,t2,t3,t4,t5 tcursor
f form
tab AnyType
tt,dd,mm,gg number
endVar

t.open(":priv:vvod_")   ; открыли рабочую базу для ввода путевого листа
t.edit()
if t.isempty() then t.insertrecord()  ; если пустая - вставили текущую дату
                    t."dat"=today()
endif
t.endedit()
t.close()                    ; закрыли рабочую базу для ввода путевого листа

f.open(":priv:vvod")          ;открыли форму для ввода путевых листов
f.action(databeginedit)       ;разрешили ввод
f.wait()                      ; ждем принудительного закрытия

t4.open(":priv:v")            ; t1,t2,t3,t4 - рабочие базы из других форм, которые используются в
t3.open(":priv:smarsh_")      ; этой форме. Например, :priv:vodi_ из формы выбора водителей,
t2.open(":priv:spra1_")       ;которая рассматривалась выше
t1.open(":priv:vodi_")
```

```
t.open(":priv:vvod_")  ; окткыли рабочую базу, куда производился ввод
t.edit()
d=t."dat"         ;присвоили переменной значение поля даты рабочей базы
dd=day(d)         ; определили день

mm=month(d)  ;месяц
gg=year(d)        ;год
t."ggmm"=gg*100+mm       ;===== вычислини значение ггггмм -поля цифровые
t."tabn"=t1."tna"
t."fio"=t1."fio"
t."nomavto"=t2."nm"
t."marka"=t2."marka"
t."tipavto"=t2."name"
t."regul"=t2."reg"
t."ben100"=t2."t100"
t."top"=t2."nip"
t."markod"=t3."kod"
t."mar"=t3."name"
;t."mar_per"=t4."kkk"
t4.close()        ;закрываем рабочие области других форм
t3.close()        ;закрываем рабочие области других форм
t2.close()        ;закрываем рабочие области других форм
t1.close()        ;закрываем рабочие области других форм
t."obprtob"=t."marprob"+t."perprob"
t."benplan"=t."obprtob"*t."ben100"/100   ;бензин на план=общий пробег*норма на 100км/100
 if t."plan">0 then t."prplan"=(t."ved"+t."sprav"+t."nal")-t."plan"     ;обработка плана в рублях
                   else t."prplan"=blank()     ;поле очищается, чтобы не загромождать форму
endif
if t."pejplan">0 then t."regPl"=t."pejfakt"/t."pejplan"
                      else t."regPl"=blank()
endif
t."n"=1

tab=t."tabn"
tt=tab                        ;табельный номер переводится в цифровой формат
t."brig"=int(tt/10000)  ; для определения  номера бригады берется целая часть от полного табельного
;номера деленного на 10000

t.endedit()
t.close()

add(":priv:vvod_",":priv:vvod")  ;дописали введенную запись в общую базу
errorshow()

executeqbefile("top77a",":priv:top77") ;подготовка данных для перерасчета остатков по топливу
          ;SELECT DISTINCT all
          ;FROM ":PRIV:Vvod.DB" D, ":PRIV:VVOD_.DB" D1
          ;WHERE  (D.Dat >= D1.Dat)   AND (D1.Tabn = D.Tabn)   AND (D1.Ggmm = D.Ggmm)
errorshow()
;---------------------------
t.open(":priv:top77")
n1=nrecords(t)
n2=n1
```

```
n3=t."benitn"
t.edit()
scan t:                    ; сканирование базы, где проиводится перерасчет остатков по топливу
        n2=n2-1
        message(n1,n2) ;сообщения о номере обрабатываемой записи будут в режиме отладка

        t."beneko"=t."benplan"-t."benzap"
        t."bendolg"=t."benvid"-t."benzap"
        t."benitn"=n3
        t."benitk"=t."benitn"+t."bendolg"-t."beneko"
        n3=t."benitk"

endscan                    ;==================
t.endedit()
t.close()
;=======================================
executeqbefile("konve10i") ;in vvod chengeto from top77
errorshow()

endMethod
```

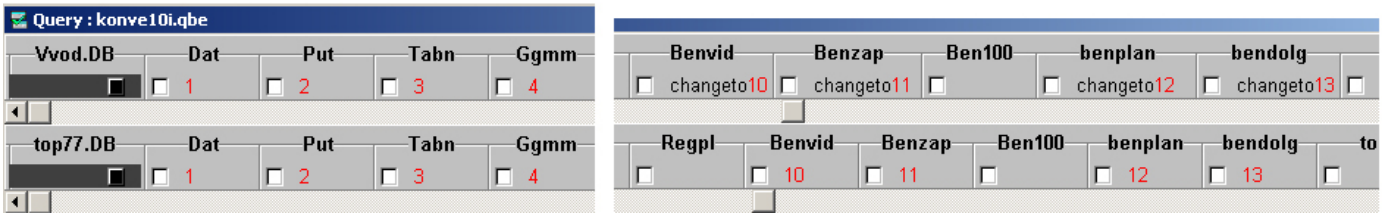


Рис.2.21 рабочая область запроса konve10i

Арифметические операции с датами

Сначала вспомним настройки BDE (их можно изменить из панели управления Windows или Пуск / все программы / Paradox9 / Utilities / Borland Database Engine). На рис.2.22 показан один из вариантов конфигурации.

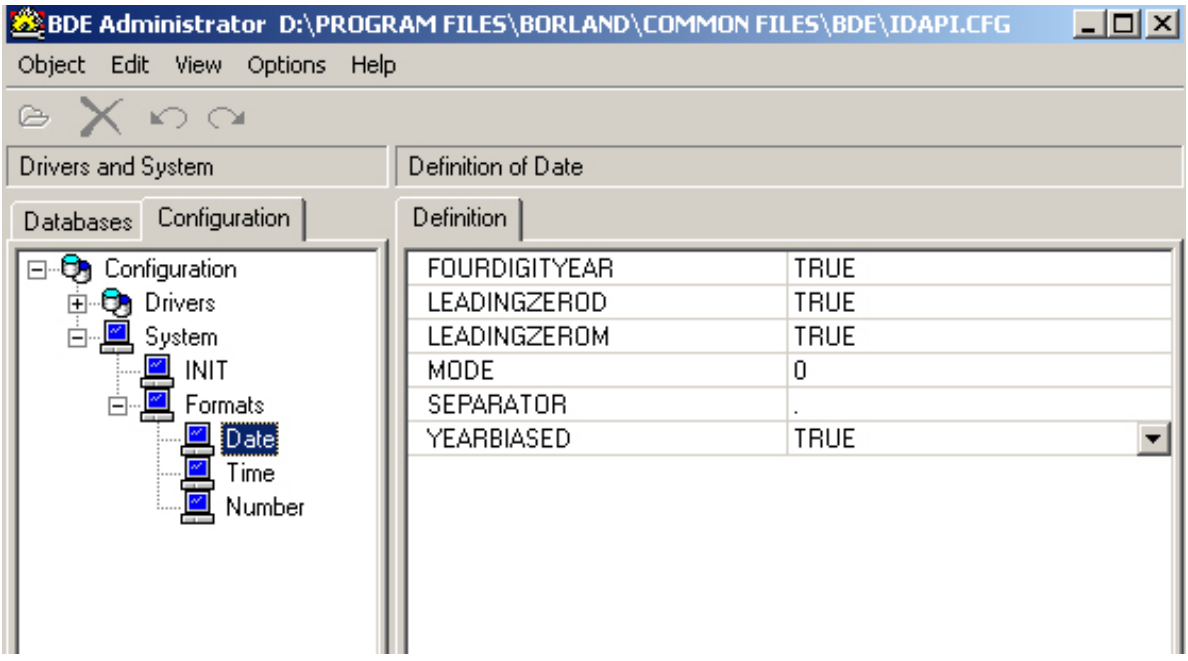


Рис.2.22 конфигурация Borland Database Engine

где:

FOURDIGITYEA	устанавливает количество знаков в заначении года TRUE - четыре знака 1991 FALSE - два знака 91
LEADINGZEROD	ноль в значении дня TRUE - 12.01.1991 FALSE - 12.1.1991
LEADINGZEROM	ноль в значении месяца TRUE - 01.30.1991 FALSE -1.30.1991
MODE	порядок написания день-месяц-год 0 - МДГ 1 - ДМГ 2 - ГМД
SEPARATOR	разделитель м ежду значениями. По умолчанию предлага-ется косая черта, но у нас принята точка
YEARBIASED	добавляет новер столетия. ессли дата 07.21.91 то TRUE - 7.21.1991 FALSE - 7.21.0091

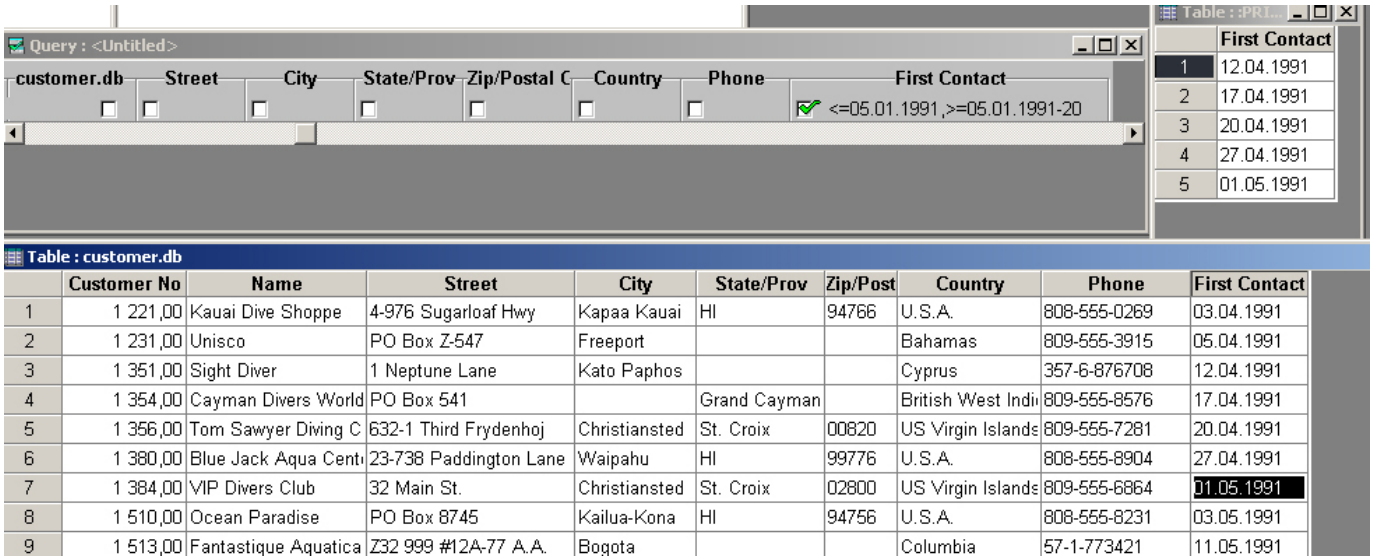


Рис.2.23. запрос на выбор вычисляемых дат

По данным рис.2.22 видно, что порядок следования установлен: месяц-день-год. Для наглядности процесса используем таблицу, которая уже рассматривалась в первой части и входит в состав поставки Paradox (samples / customer). В последнем поле, где формат "дата", видно, что, что установлен порядок день-месяц-год, т.е. отличный от установленного в BDE.

Из данной таблицы необходимо выбрать даты операций, которые производились за 20 дней до 1 мая. На рис.2.23 показан запрос и результат его выполнения. Формат дат в запросе соответствует настройкам BDE.

Рассмотрим метод из задачи "контроль исполнительской документации". Задача в том, чтобы отобрать отчетность, которую надо представить в течение трех дней.

```
method pushButton(var eventInfo Event)
var
    t tcursor
    f form
    dat1 Date
    d1,m1,g1 SmallInt
    d2,m2,g2 SmallInt
```

```
endvar
;----------------
t.open(":priv:datnk")
t.edit()
t.empty()
t.insertrecord()
t."datn"=today()
t.endedit()
 t.close()

t.open(":priv:day_")
t.edit()
t.empty()
t.insertrecord()
t."day"=3                    ;предварительное значение для формы
t.endedit()
t.close()

t.open(":priv:baza_")
t.edit()
t.empty()
t.endedit()
t.close()

t.open(":priv:baza_1")
t.edit()
t.empty()
t.endedit()
t.close()                     ; окончание подготовки рабочих баз

f.open(":priv:dats1")         ;открытие формы рис.2.24 для внесения начальной даты и периода
f.action(DataBeginEdit)
f.wait()

executeQBEfile("ask1")        ;изменение даты окончания просмотра
errorshow()

t.open(":priv:datnk")
dat1=t."datn"
dat2=t."datk"
  d1=day(dat1)
  m1=month(dat1)
  g1=year(dat1)
d2=day(dat2)
m2=month(dat2)
g2=year(dat2)
 t.close()

t.open(":priv:ask1")
t.edit()
t.empty()
t.insertrecord()
t."d1"=d1
```

```
t."m1"=m1
t."g1"=g1
t."d2"=d2
t."m2"=m2
t."g2"=g2
t.endedit()
t.close()

if m2>m1 then ; выбор данных
                executeQBEfile("ask2",":priv:r");результат помещается в базу r личного каталога
                errorshow()
                executeQBEfile("ins4");r insert in :priv:baza_
                errorshow()
                executeQBEfile("ask22",":priv:r");
                errorshow()
                executeQBEfile("ins4");r insert in :priv:baza_
                errorshow()

else
    executeQBEfile("ask23",":priv:r");
    errorshow()
    executeQBEfile("ins4");r insert in :priv:baza_
    errorshow()
endif

executeQBEfile("ins5"); insert in :priv:baza_1
errorshow()

f.open(":priv:baza_")   ; просмотр результата
f.wait()

endmethod
```

рис.2.24 форма для задания условий выборки

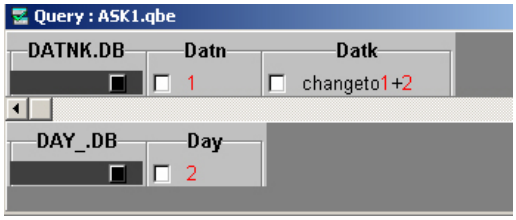


Рис.2.25 изменение даты окончания

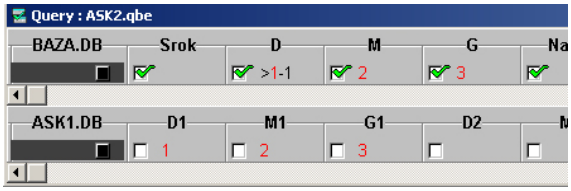


Рис.2.26 выбор данных

В показанных примерах для определения периода использовалось количесство дней на которое производится увеличение или уменьшение конечной даты.

Аналогично можно определить разность между датами. Обычно период, указанный в днях удовлетворяет всем запросам. Если надо работать с календарными периодами, то из даты надо выделить необходимое значение, а потом опять собрать их в дату. Синтаксис этой операции показан ниже:

date
Процедура Преобразует значение к типу Date.
Тип Date
Синтаксис date ([const значение AnyType]) Date
Описание Метод преобразует значение к типу Date. Недопустимое значение даты приводит к ошибке. Если аргумент значение не задается, процедура date возвращает текущую дату как значение типа Date.
Пример В данном примере строковая переменная преобразуется к типу Date, значение даты используется в вычислениях, и результат выводится в окно диалога:

```
; thisButton::pushButton
method pushButton(var eventInfo Event)
var
  s String
  d Date
endVar

s = "11/11/99"   ;  s это строка
d = date(s) + 7   ;  строка преобразуется к типу даты

d.view()          ;  значение d (11/18/99) выводится в окно диалога

endMethod
```

Выполнение внешних программ

Иногда необходимо выполнить программу, написанную на другом языке программирования. Для этой цели используется команда **Execute**.Синтаксис для этой команды приведен ниже:
execute
Метод Посылает команды через связь DDE
Тип DDE
Синтаксис execute (const команда String) Logical
Описание Строка команда посылается в приложение по связи DDE. Характер строки команды должен изменяться в соответствии с требованиями данного приложения. Так, например, командная строка для текстового процессора может не восприниматься электронной таблицей, а электронные таблицы различных изготовителей для выполнения одинаковых действий могут использовать различные команды.
Пример В следующей программе функция ObjectVision @SETTITLE используется для задания текста, выводящегося в заголовок окна ObjectVision.

```
method pushButton(var eventInfo Event)
var
  d1 DDE
endVar

;  устанавливается связь с формой ObjectVision с именем Address
d1.open("vision", "C:\\vision\\forms\\Address.ovd")

;  исполняется команда ObjectVision @SETTITLE
d1.execute("[@SETTITLE(\"Управление принято!\")]")

endMethod
```

execute
Процедура Исполняет команду DOS.
Тип System
Синтаксис execute (const имяПрограммы String [, const задержка Logical [, const режимЭкрана SmallInt]]) Logical
Описание Исполняется команда DOS, задаваемая строкой имяПрограммы. Необязательный аргумент режимЭкрана задает режим экрана, используемый при исполнении команды.

Если команда не содержится в каталоге пользователя, в аргументе имяПрограммы следует задать полный путь, используя при этом двойные обратные косые.

Пример Следующая программа запускает приложение Clock системы Windows со стандартным стилем окна и переходит в режим ожидания завершения программы.

```
; showClock::pushButton
method pushButton(var eventInfo Event)
execute("clock.exe", Yes, WinStyleDefault)  ; исполняется Windows Clock
endMethod
```

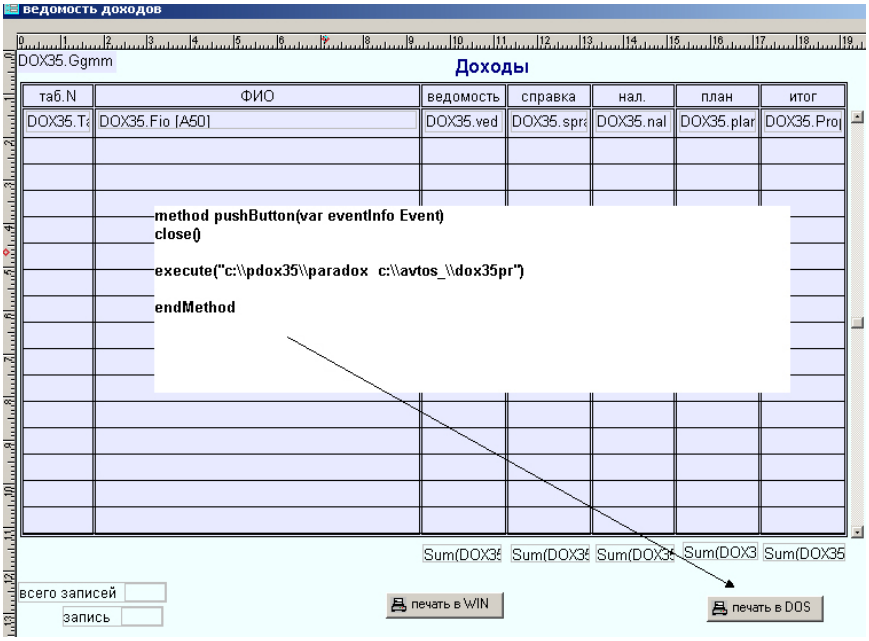


Рис.2.25 пример использования внешней программы

На рис. 2.25 показана возможность печати ведомости на матричном принтере. Сначала указывается полный путь к средству, а потом полный путь к самой программе. Выполняемая программа - скрипт на paradox-3.5 (DOS версия) приведен ниже:

```
ClearAll {Tools} {More}
{Directory}  {c:\\avtos_\\}  {OK}
{Report} {Output} {dox35} {1}
{Printer} {Scripts} {End-Record}
exit
```

Использование управляющих клавиш при вводе

В некоторых слычаях использование мыши затрудняет работу. Например, ввод квитанций. В этом случае желательно пользоваться клавиатурой. в PAL каждой клавише соответствует своя константа:

Константа	Тип	Описание
VK_ADD	SmallInt	Клавиша сложения
VK_BACK	SmallInt	Клавиша Backspace
VK_CANCEL	SmallInt	Используется для обработки нажатия комбинации клавиш Ctrl-break
VK_CAPITAL	SmallInt	Клавиша перехода на символы верхнего регистра
VK_CLEAR	SmallInt	Клавиша очистки
VK_CONTROL	SmallInt	Клавиша Ctrl
VK_DECIMAL	SmallInt	Клавиша ввода десятичого разделителя
VK_DELETE	SmallInt	Клавиша Delete

VK_DIVIDE	SmallInt	Клавиша деления
VK_DOWN	SmallInt	Клавиша
VK_END	SmallInt	Клавиша End
VK_ESCAPE	SmallInt	Клавиша Escape
VK_EXECUTE	SmallInt	Клавиша запуска на выполнение
VK_F1	SmallInt	Клавиша F1
VK_F10	SmallInt	Клавиша F10
VK_F11	SmallInt	Клавиша F11
VK_F12	SmallInt	Клавиша F12
VK_F13	SmallInt	Клавиша F13
VK_F14	SmallInt	Клавиша F14
VK_F15	SmallInt	Клавиша F15
VK_F16	SmallInt	Клавиша F16
VK_F2	SmallInt	Клавиша F2
VK_F3	SmallInt	Клавиша F3
VK_F4	SmallInt	Клавиша F4
VK_F5	SmallInt	Клавиша F5
VK_F6	SmallInt	Клавиша F6
VK_F7	SmallInt	Клавиша F7
VK_F8	SmallInt	Клавиша F8
VK_F9	SmallInt	Клавиша F9
VK_HELP	SmallInt	Клавиша Help
VK_HOME	SmallInt	Клавиша Home
VK_INSERT	SmallInt	Клавиша Insert
VK_LBUTTON	SmallInt	Левая кнопка мыши
VK_LEFT	SmallInt	Клавиша
VK_MBUTTON	SmallInt	Средняя кнопка трехкнопоченой мыши
VK_MENU	SmallInt	Клавиша меню
VK_MULTIPLY	SmallInt	Клавиша умножить
VK_NEXT	SmallInt	Клавиша Page Down
VK_NUMLOCK	SmallInt	Клавиша Num Lock
VK_NUMPAD0	SmallInt	Клавиша 0 цифровой клавиатуры
VK_NUMPAD1	SmallInt	Клавиша 1 цифровой клавиатуры
VK_NUMPAD2	SmallInt	Клавиша 2 цифровой клавиатуры
VK_NUMPAD3	SmallInt	Клавиша 3 цифровой клавиатуры
VK_NUMPAD4	SmallInt	Клавиша 4 цифровой клавиатуры
VK_NUMPAD5	SmallInt	Клавиша 5 цифровой клавиатуры
VK_NUMPAD6	SmallInt	Клавиша 6 цифровой клавиатуры
VK_NUMPAD7	SmallInt	Клавиша 7 цифровой клавиатуры
VK_NUMPAD8	SmallInt	Клавиша 8 цифровой клавиатуры
VK_NUMPAD9	SmallInt	Клавиша 9 цифровой клавиатуры
VK_PAUSE	SmallInt	Клавиша Pause
VK_PRINT	SmallInt	Специальная клавиша OEM
VK_PRIOR	SmallInt	Клавиша Page Up
VK_RBUTTON	SmallInt	Правая кнопка мыши
VK_RETURN	SmallInt	Клавиша Return (Etnter)
VK_RIGHT	SmallInt	Клавиша

VK_SELECT	SmallInt	Клавиша выбора
VK_SEPARATOR	SmallInt	Клавиша ввода символа разделителя
VK_SHIFT	SmallInt	Клавиша Shift
VK_SNAPSHOT	SmallInt	Клавиша печати экрана для системы Windows 3.*
VK_SPACE	SmallInt	Пробел
VK_SUBTRACT	SmallInt	Клавиша вычитания
VK_TAB	SmallInt	Клавиша Tab
VK_UP	SmallInt	Клавиша

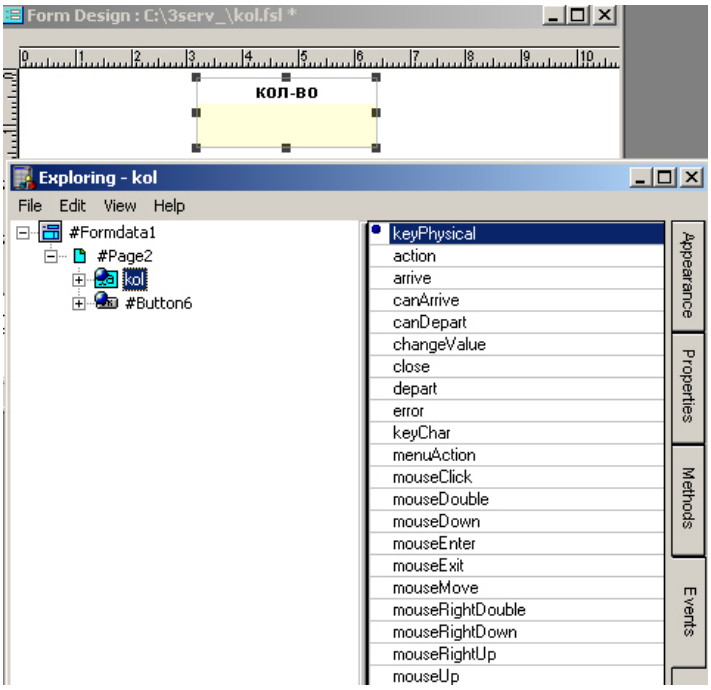


Рис.2.26 метод KeyPhysical

На рис.2.26 показан метод, который закрывает форму при нажатии клавиши Enter.

Более детально с методом KeyPysical можно ознакомится в окне помощи Paradox

```
method keyPhysical(var eventInfo KeyEvent)
var
k string
endvar

k=""
k=eventinfo.vchar()
if k="VK_RETURN" then close()
endif
endMethod
```

Работа с текстовыми файлами.

Часто обмен между организациями производится в виде текстовых файлов. Рассмотрим программу, которая переносит эту информацию в Paradox. Например, используем файл передачи данных по автортанспорту:

```
…
ВерсПрог:РЭО
@@@
ИдДок:0722222222**072222222200300254302
ДатаДок:30.10.2003
ИдНомер:XTA22222222222222
ЗаводНомер:1238521
ВидТран:510
ГодИзгТран:1991
СрокПолИсп:  12
ТипТран:Седан
НазнТран:В
МаркТран:ВАЗ 2105
НомерДвиг:2222-2222222
МощДвиг:85.1,63.5
НаимДокРег:Справка-счет выдан 03.09.2003 N ДКП-999999 ГУП СПАС
```

ДокРег:590,07ЛЛ999997
ДатаДокРег:03.09.2003
РегЗнак:Т999ЛЛ07
ДатаРег:03.09.2003
ФИОФЛ:ИВАНОВ,ИВАН,ИВАНОВИЧ
Пол:1
ДатаРожд:12.12.1977
АдрМЖ:643,,07,Прохладненский р-н,ПРОХЛАДНЫЙ,,ПЕТРОВКА,22
Гражд:643
УдЛичн:21,88 22 232323
ДатаУдЛичн:17.07.2002,
ДоляСобст:1
@@@
ИдДок:0719999999**071111111200300254303
ДатаДок:30.10.2003
ИдНомер:XTA123450L1234561
ЗаводНомер:1234561
ВидТран:510
…
===

Вначале идет заголовок, потом информация по каждому транспортному средству. Разделение между блоками - @@@. Признак конца файла – три знака равно.

Каждая строка имеет свой идентификатор, который заканчивается двоеточием. Потом идет информационная часть.

На первом этапе надо информацию для каждой строки одного блока поместить в свое поле. Это можно сделать следующим скриптом:

```
method run(var eventInfo Event)
importASCIIFix ( "g.txt", "orders.db", "inpstr.db, False ")

endmethod
```

где g.txt - текстовый файл, который надо перевести в таблицу Paradox
oders.db - таблица Paradox, куда будет помещен текстовый файл (рис.2.28)
inpstr.db – таблица-шаблон (рис.2.27)

	Field Name	Type	Size	Key	Koment
1	Im	A	10		имя
2	Type	A	4		тип
3	Start	S			нач.позиция
4	Length	S			кон.позиция

Рис.2.27 структура таблицы-шаблона

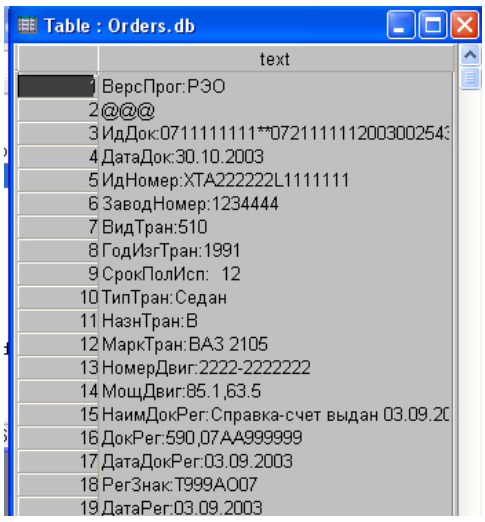


Рис.2.28 результат работы метода

Ниже приведен метод, который полностью решает поставленную задачу:

```
method run(var eventInfo Event)
;перевод текстового файла  в базу.
var
t,t1,t2,t10 tcursor
ar array[]AnyType
a string
ai,k,k1 LongInt
p,p1,p2,p3,p4,p5,p6,p7,p8,p9 AnyType
p10,p11,p12,p13,p14,p15,p16,p17 AnyType
p18,p19,p20,p21,p22,p23,p24 AnyType
dd Date
f,f1,f10,f11 form
endvar
;====================
importASCIIFix ( "g.txt", "out.db", "inpstr.db", False )
errorshow()

 t.open("trfl2")
  t.edit()
    t.empty()
  t.endedit()
t.close()
;=======================
k=1
t1.open("out")
  t.open("trfl")
    t.edit()
  t.empty()
;========  в trfl.db (2 поля) ===============
scan t1:
a=t1."text"              ; строковой переменной присваивается значение записи
a.breakApart(ar,":") ; признак разделения по полям - двоеточие
  t.insertAfterRecord()
  t.copyfromarray(ar)
k=k+1
message("scan2  "," прошло записей  ",k)
endscan
; ------
  t.endedit()
  t.close()
t1.close()
errorshow()
;===определение переменных====================
scan t:
;перед формированием новой записи все переменные должны быть обнулены. Это
;необходимо для случая, когда нет какой – либо записи. Если переменные не обнулить, то
;при отсутствии записи будет повторяться предыдущее значение.
p1=0
p2=0
p3=0
p4=0
p5=0
p6=0
```

```
p7=0
p8=0
p9=0
p10=0
p11=0
p12=0
p13=0
p14=0
p15=0
p16=0
p17=0
p18=0
p19=0
p20=0
p21=0
p22=0
p23=0
p24=0
p25=0
k=1
t.open("trfl")
k=k-1
;1======
if t."kod"="ИдДок" then ;t2."id"
                    p1=t."name"
              if p1.isBlank() then p1=0
                          else message("нет")
              endif
  else message("1=0")
endif
;2==========
if t."kod"="ДатаДок" then ;t2."datdok"
                    p2=t."name"
               if p2.isBlank() then p2=0
                          else message("ku")
              endif
               else message("2=0")
endif
;3==========
if t."kod"="ИдНомер" then ;t2."idnom"
                    p3=t."name"
               if p3.isBlank() then p3=0
                          else message("ku")
              endif
           else message("3=0")
endif
;4==========
if t."kod"="ЗаводНомер" then ;t2."zavnom"
                     p4=t."name"
               if p4.isBlank() then p4=0
                          else message("ku")
              endif
           else message("4=0")
```

```
endif
;5==========
if t."kod"="ВидТран" then ;t2."vidtr"
                    p5=t."name"
                if p5.isBlank() then p5=0
                            else message("ku")
                endif
            else message("5=0")
endif
;6==========
if t."kod"="ГодИзгТран" then ;t2."godiz"
                    p6=t."name"
                if p6.isBlank() then p6=0
                            else message("ku")
                endif
            else message("6=0")
endif
;7==========
if t."kod"="СрокПолИсп" then ;t2."sropois"
                     p7=t."name"
                if p7.isBlank() then p7=0
                            else message("ku")
                endif
            else message("7=0")
endif
;8==========
if t."kod"="ТипТран" then ;t2."niptr"
                    p8=t."name"
                 if p8.isBlank() then p8=0
                            else message("ku")
                 endif
            else message("8=0")
endif
;9==========
if t."kod"="НазнТран" then ;t2."nazntr"
                     p9=t."name"
                if p9.isBlank() then p9=0
                            else message("ku")
                endif
            else message("7=0")
endif
;10==========
if t."kod"="МаркТран" then ;t2."Marka"
                    p10=t."name"
                 if p10.isBlank() then p10=0
                            else message("ku")
                 endif
            else message("8=0")
endif
;11==========
if t."kod"="НомерДвиг" then ;t2."nomdv"
                    p11=t."name"
                 if p11.isBlank() then p11=0
```

```
                    else message("ku")
            endif
        else message("9=0")
endif
;12==========
if t."kod"="МощДвиг" then ;t2."mosno"
                    p12=t."name"
            if p12.isBlank() then p12=0
                    else message("ku")
            endif
        else message("10=0")
endif
;13==========
if t."kod"="НаимДокРег" then ;t2."naimdok"
                    p13=t."name"
            if p13.isBlank() then p13=0
                    else message("ku")
            endif
        else message("11=0")
endif
;14==========
if t."kod"="ДокРег" then ;t2."dokreg"
                    p14=t."name"
            if p14.isBlank() then p14=0
                    else message("ku")
            endif
        else message("12=0")
endif
;15==========
if t."kod"="ДатаДокРег" then ;t2."datdokr"
                    p15=t."name"
            if p15.isBlank() then p15=0
                    else message("ku")
            endif
        else message("13=0")
endif
;16==========
if t."kod"="РегЗнак" then ;t2."regzn"
                    p16=t."name"
            if p16.isBlank() then p16=0
                    else message("ku")
            endif
        else message("14=0")
endif
;17==========
if t."kod"="ДатаРег" then ;t2."datreg"
                    p17=t."name"
            if p17.isBlank() then p17=0
                    else message("ku")
            endif
        else message("17=0")
endif
;18==========
```

```
if t."kod"="ФИОФЛ" then ;t2."fio"
                    p18=t."name"
              if p18.isBlank() then p18=0
                          else message("ku")
              endif
            else message("18=0")
endif
;19==========
if t."kod"="Пол" then ;t2."pol"
                    p19=t."name"
              if p19.isBlank() then p19=0
                          else message("ku")
              endif
            else message("19=0")
endif
;20==========
if t."kod"="ДатаРожд" then ;t2."datrog"
                     p20=t."name"
              if p20.isBlank() then p20=0
                          else message("ku")
              endif
            else message("20=0")
endif
;21==========
if t."kod"="АдрМЖ" then ;t2."adr"
                    p21=t."name"
              if p21.isBlank() then p21=0
                          else message("ku")
              endif
            else message("21=0")
endif
;22==========
if t."kod"="Гражд" then ;t2."grag"
                    p22=t."name"
              if p22.isBlank() then p22=0
                          else message("ku")
              endif
            else message("22=0")
endif
;23==========
if t."kod"="УдЛичн" then ;t2."udli"
                    p23=t."name"
               if p23.isBlank() then p23=0
                          else message("ku")
              endif
            else message("23=0")
endif
;24==========
if t."kod"="ДатаУдЛичн" then ;t2."datudli"
                    p24=t."name"
               if p24.isBlank() then p24=0
                          else message("ku")
              endif
```

```
            else message("24=0")
endif
;25==========
if t."kod"="ДоляСобст" then ;t2."dol"
                        p25=t."name"
              if p25.isBlank() then p25=0
                           else message("ku")
              endif
            else message("25=0")
endif
;==========формирование записи=====
if t."kod"="@@@" then
                  t2.open("trfl2_")
                  t2.edit()
                  t2.insertrecord()
t2."id"=p1
        ;p1.view()
t2."datdok"=p2
t2."idnom"=p3
t2."zavnom"=p4
t2."vidtr"=p5
t2."godiz"=p6
t2."sropois"=p7
t2."niptr"=p8
t2."nazntr"=p9
t2."marka"=p10
t2."nomdv"=p11
t2."mosno"=p12
t2."naimdok"=p13
t2."dokreg"=p14
t2."datdokr"=p15
t2."regzn"=p16
t2."datreg"=p17
t2."fio"=p18
t2."pol"=p19
t2."datrog"=p20
t2."adr"=p21
t2."grag"=p22
t2."udli"=p23
t2."datudli"=p24
t2."dol"=p25

                  t2.endedit()
                  t2.close()
                  add("trfl2_","trfl2")
                  errorshow()
              else message("ku-ku")
endif
endscan
  t.endedit()
  t.close()
errorshow()
endmethod
```

Table : Trfl2.db

	Id	Datdok	Idnom	Zavnor
1	0711000111**07222222200300253650	30.10.2003	XXX11100XX0111111	123
2	0711000112**07222222200300253651	30.10.2003	0	123
3	0711000123**07222221200300253652	30.10.2003	0	545
4	0711000124**07222221200300253653	30.10.2003	XXX22222NN2222222	878122

Рис.2.29 итоговая таблица

Перевод базы Paradox в файл обмена

Рассмотрим обратный процесс перевода, т.е. имея базу данных надо получить файл обмена. Этот процесс будет состоять из следующих этапов:

1. Перевод исходной базы в аналогичную по структуре, но где все поля текстовые
2. Объединение всех полей в одно и подсоединение идентификатора строки
3. Перевод таблицы Paradox в текстовый файл

Restructure Paradox 5.0 for Windows Table: Trfl2.db

Field Roster | Secondary Index | Table Lookup | Passwords | Referential Ir

No	Field Name	Type	Size	Min	M
1	Id	A	50		
2	Datdok	D			
3	Idnom	A	20		
4	Zavnom	I			
5	Vidtr	S			
6	Godiz	S			
7	Sropois	S			
8	Niptr	A	10		
9	Nazntr	A	3		
10	Marka	A	15		
11	Nomdv	A	20		
12	Mosno	A	15		
13	Naimdok	A	20		

Рис.2.30 исходная таблица

Restructure Paradox 5.0 for Windows Table: Trfl2t.db

Field Roster | Secondary Index | Table Lookup | Passwords | Referentia

No	Field Name	Type	Size	Min
1	Id	A	50	
2	Datdok	A	10	
3	Idnom	A	20	
4	Zavnom	A	12	
5	Vidtr	A	5	
6	Godiz	A	5	
7	Sropois	A	5	
8	Niptr	A	10	
9	Nazntr	A	3	
10	Marka	A	15	
11	Nomdv	A	20	
12	Mosno	A	15	
13	Naimdok	A	20	

Рис.2.31. исходная таблица с текстовыми полями

Надо отметить, что в приведенном ниже скрипте идентификатор будет записываться в каждое поле. Поэтому все поля надо увеличить на максимальную длину поля идентификатора строки, т.е. на 15 знаков.

Скрипт, для перевода таблицы trfl2.db в текстовый файл приведен ниже:

```
method run(var eventInfo Event)
; перевод базы Paradox в текстовый файл
var
t,t1 tcursor
a array[]AnyType
endvar
;---------перезапись из trfl2.db в trfl2t.db---------
t.open("trfl2t")
t.edit()
t.empty()
  t1.open("trfl2")
```

```
      scan t1:
      t1.copytoarray(a)
      t.insertrecord()
      t.copyfromarray(a)
     endscan
t1.close()
t.endedit()
t.close()
;==================
executeQBEfile("trchen") ;данный запрос добавляет в каждое поле идентификатор записи (рис.2.32)
errorshow()
;---------------------------------
t.open("out")
t.edit()
t.empty()
  t1.open("trfl2t")

      scan t1:
;1
t.insertrecord()
t."text"=t1."id"
; если длина поля не увеличена на 15, то надо записать
; t."text"=" ИдДок:"+t1."id" и т.д.
;2
t.insertafterrecord()
t."text"=t1."datdok"
;3
t.insertafterrecord()
t."text"=t1."idnom"
;4
t.insertafterrecord()
t."text"=t1."zavnom"
;5
t.insertafterrecord()
t."text"=t1."vidtr"
;6
t.insertafterrecord()
t."text"=t1."godiz"
;7
t.insertafterrecord()
t."text"=t1."sropois"
;8
t.insertafterrecord()
t."text"=t1."niptr"
;9
t.insertafterrecord()
t."text"=t1."nazntr"
;10
t.insertafterrecord()
t."text"=t1."Marka"
;11
t.insertafterrecord()
t."text"=t1."nomdv"
```

```
;12
t.insertafterrecord()
t."text"=t1."mosno"
;13
t.insertafterrecord()
t."text"=t1."naimdok"
;14
t.insertafterrecord()
t."text"=t1."dokreg"
;15
t.insertafterrecord()
t."text"=t1."datdokr"
;16
t.insertafterrecord()
t."text"=t1."regzn"
;17
t.insertafterrecord()
t."text"=t1."datreg"
;18
t.insertafterrecord()
t."text"=t1."fio"
;19
t.insertafterrecord()
t."text"=t1."pol"
;20
t.insertafterrecord()
t."text"=t1."datrog"
;21
t.insertafterrecord()
t."text"=t1."adr"
;22
t.insertafterrecord()
t."text"=t1."grag"
;23
t.insertafterrecord()
t."text"=t1."udli"
;24
t.insertafterrecord()
t."text"=t1."datudli"
;25
t.insertafterrecord()
t."text"=t1."dol"
;
t.insertafterrecord()
t."text"="@@@"
;
;----эта пустая запись добавляется для устранения ошибки, которая может
;появляться в некоторых версиях Paradox. Ошибка заключается в том, что
; последняя запись вставляется при сканировании не в конец блока, а в
;конец базы
t.insertafterrecord()
t."text"=""
    endscan
```

```
t1.close()
t.endedit()
t.close()
;-----запрос, который удаляет искусственную запись
executeQBEfile("trdel")        ;DELETE
                               ;FROM "OUT.DB"
                               ;WHERE
                               ;(text IS NULL)
errorshow()
;----------перевод базы out.db в текстовый файл outtex.txt
; структура и содержание базы inpstr.db описано выше
exportASCIIFix ( "out.db", "outtex.txt", "inpstr.db", True )

endmethod
```

Исходная база данных

Table : Trfl2t.db

	Id	Datdok	Idnom	Zavnom	Vidtr	G
1	0711000111**072222222200300253650	30.10.2003	XXX11100XX0111111	123456	510	1
2	0711000112**072222222200300253651	30.10.2003	0	123457	510	1
3	0711000123**072222221200300253652	30.10.2003	0	54545	510	1
4	0711000124**072222221200300253653	30.10.2003	XXX22222NN2222222	87812255	510	1
5	0711053360**072101001200300253654	30.10.2003	XTA210600X4178630	4178630	510	1

Добавление идентификатора

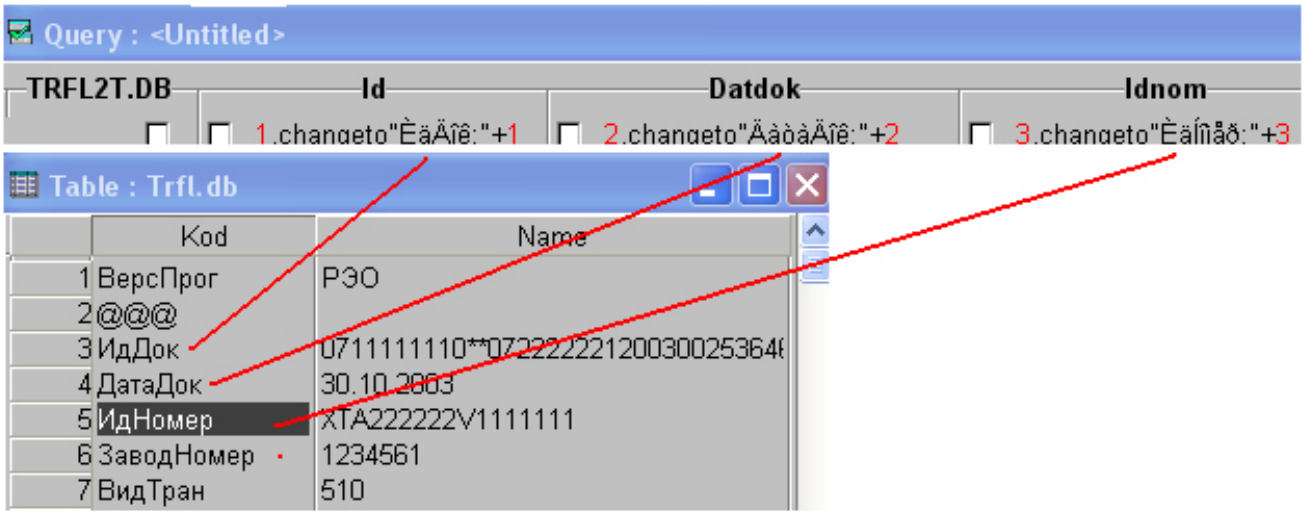


Рис.2.32.добавление идентификаторов в поля

В результате получим текстовый файл, приведенный ниже:

ИдДок:0700011110**07111111200300253646
ДатаДок:30.10.03
ИдНомер:XXX222222X1111111
ЗаводНомер:123456
ВидТран:510
ГодИзгТран:1996
СрокПолИсп: 7
ТипТран:Седан
НазнТран:В
МаркТран:ВАЗ 21074

МаркТран:2106-4426310
МощДвиг:101.2,75.5
НаимДокРег:
ДокРег:590,07НН999999

На приведенных примерах описан перевод однотипной структуры текстового файла. Если надо учитывать совместно заглавную и информационную части, то переводятся две базы независимо до предпоследнего этапа. Например, создаются ба-зы out.db и out1.db.

Потом эти базы складываются. Складываются он при помощи QBE запроса и в текстовый переводится уже суммарная база.

Перевод алфавитно-цифрового поля в базу данных

Рассмотрим пример, когда надо перенести данные из одной задачи в другую. Исходные данные имеют вид:

Table : c:\1AB_G\1AB_FIZ\N.DB

	NUM	TYPE	DT	NOTES
1	00101	1	01.09.2001	63(0.56)
2	00102	1	04.10.2001	40(0.28)28(0.56)
3	00103	1	04.09.2001	37(0.56)
4	00104	1	11.10.2001	100(0.56)
5	00105	1	20.09.2001	57(0.56)
6	00106	1	07.09.2001	83(0.56)
7	00106	1	02.10.2001	64(0.56)
8	00108	1	07.09.2001	140(0.56)
9	00108	1	27.09.2001	120(0.56)
10	00108	11	02.10.2001	контр.не может попасть нуж
11	00109	1	25.09.2001	120(0.28)66(0.56)
12	00110	1	18.09.2001	40(0.28)
13	00110	1	19.09.2001	40(0.28)
14	00111	1	14.09.2001	100(0.28)

где:
- Num – номер лицевого счета
- Type – тип оплат
- Dt – дата оплаты
- Notes – количество киловат электроэнергии и в скобках тариф

Рис.2.33 исходжная таблица

Задача состоит в том, чтобы алфавитное поле NOTES разбить на цифровые поля для возможности делать с ними арифметические операции.
Рассмотрим скрипт, который реализует эту задачу:

```
method run(var eventInfo Event)
var
   t,t1,t2,t10 tcursor
   ar array[]AnyType
   a string
   ai,k,k1 LongInt
   dd Date
   f,f1,f10,f11 form
endvar
;============================
message("1n - добавление эталона")
;учитывая непредсказуемость запроса (выборка может делаться по лицевому счету и дате)
```

```
; надо обеспечить условие, чтобы в любой ситуации поле NOTES имело как простые, так и
; льготные киловаты. Для этого выполняется QBE запрос, который показан на рис.2.33.

executeQBEfile("ndop"); in N.db indert from NDOP.db
   errorshow()

message("scan")
t.open("n")
k=t.nrecords() ; определение количества записей
k1=k
  t1.open("n1")
  t1.edit()
  t1.empty()
scan t:           ;переводбазы N.db в аналогичную базу с алфавитными полями
 t.copytoarray(ar)
t1.insertAfterRecord()
t1.copyfromarray(ar)
k=k-1
message("scan1  ",k1," осталось  ",k)
endscan
;-------------------

executeQBEfile("3"); удаление пустых в n1
    errorshow()

message("1n - сложение колонок для получения одного поля")
executeQBEfile("2","n2")
    errorshow()
;=========================
k=k1
t1.open("n2")
  t.open("n2nn")
  t.edit()
  t.empty()
;=========================

scan t1:
a=t1."text"
a.breakApart(ar," ()") ; разделитель полей – пробел, открывающаяся скобка,
                       ; закрывающаяся скобка
  t.insertAfterRecord()
  t.copyfromarray(ar)
k=k-1
message("scan2  ",k1," осталось  ",k)
endscan

  t.endedit()
  t.close()
t1.close()
errorshow()

executeQBEfile("k7del2");удаление, где К1 и К2 пустые, т.е. ошибочные записи в исходной таблице
endmethod
```

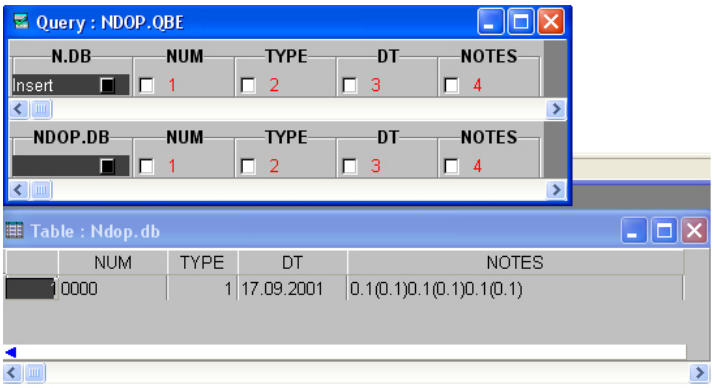


Рис.2.33 запрос и таблица эталон

Table : N2.db

	Text
1	0000 17.09.01 0.1(0.1)0.1(0.1)0.1(0.1)
2	00101 01.09.01 63(0.56)
3	00102 04.10.01 40(0.28)28(0.56)
4	00103 04.09.01 37(0.56)
5	00104 11.10.01 100(0.56)
6	00105 20.09.01 57(0.56)
7	00106 02.10.01 64(0.56)
8	00106 07.09.01 83(0.56)
9	00108 07.09.01 140(0.56)
10	00108 27.09.01 120(0.56)

Рис.2.34 исходная таблица с эталоннгой записью

В итоге получим таблицу (рис.35). Обратите внимание на поля K1, S1 и K2, S2. В одном поле десятичным разделителем является точка, в другом – запятая. Дело в том, что Windows и Paradox на компьютере, где формировался этот пример, настроены на десятичный разделитель "запятая". В тексте (см.выше) разделитель – точка. Поэтому поля "S1, S2" должны быть алфавитными. Если они будут цифровые, то в них никакой информации не будет. Перевести их в цифровые можно добавив еще одно сканирование, т.е. базу n2nn считать промежуточной, а потом из ее переписать в рабочую базу.

На основе приведенных примеров можно, например, выяснить распределение слов в каком -либо тексте. Для этого надо отсканировать текст, создать таблицу с количеством полей, равному максимальному количеству слов и используя разделитель "пробел", перевести содержимое текста в бузу данных. И когда в каждом поле будет по одному слову, можно делать любой анализ.

Table : N2nn.db

	Num	Tipe	Dat	K1	S1	K2	S2
1	00104	1	18.10.2002	80,00	0.3800		
2	00105	1	17.10.2002	114,00	0.7500		
3	00106	1	07.10.2002	80,00	0.6700		
4	00108	1	09.10.2002	80,00	0.6700		
5	00109	1	10.10.2002	40,00	0.3800	93,00	0.7500
6	00111	1	09.10.2002	188,00	0.3400		
7	00112	1	09.10.2002	26,00	0.3400		
8	00114	1	11.10.2002	89,00	0.7500		
9	00117	1	09.10.2002	100,00	0.6700		
10	00119	1	07.10.2002	50,00	0.6700		
11	00121	1	30.10.2002	63,00	0.7500		

Рис.2.35 итоговая таблица

Перевод числа в текст

Описанная ниже программа полностью автономна. В ней используется одна таблица, откуда берется числовое значение, а потом в неё же помещается текстовое значение этой цифры. Вызывать эту программу можно двумя способами:

- через библиотеку
- как независимый метод

Для работы с библиотекой необходимо пользоваться следующими инструментами:

1. Создание библиотеки

Для создания новой библиотеки ObjectPAL следует выбрать команду Файл / Создать / Библиотека. Нажмите правую кнопку мыши в окне конструктора библиотеки, чтобы открыть окно диалога "Методы". Затем набирайте текст программы, как и для любого другого объекта.

После завершения набора программы можно

- сохранить исходный и исполняемый код с помощью команды Файл / Сохранить.
- сохранить только исполняемый код с помощью команды Язык ObjectPAL / Упаковка.

2.Добавление программы в библиотеку

Для того чтобы добавить программу в библиотеку, нажмите правую кнопку мыши в окне конструктора библиотеки для открытия окна диалога "Методы". Определите затем программу, как и для любого другого объекта. Можно добавлять программу к новой или к уже существующей библиотеке.

После завершения набора программы можно
- сохранить исходный и исполняемый код с помощью команды Файл | Сохранить.
- сохранить только исполняемый код с помощью команды Язык ObjectPAL | Упаковка.

С помощью окна диалога "Методы" и окна редактора ObjectPAL можно добавить пограмму в библиотеку следующими способами:
- Определить программу для встроенных методов.
- Добавить специальные методы.
- Добавить специальные процедуры.
- Описать переменные, константы, типы данных и внешние программы.

3.Вызов библиотечных методов

Для вызова метода из библиотеки необходимо сначала описать этот метод в окне Uses вызывающего объекта. Например, предположим нужно, чтобы метод кнопки pushButton вызвал специальный метод из библиотеки. Опишите этот метод в окне кнопки Uses (или в окне Uses объекта, который содержит кнопку), чтобы Paradox знал, где искать этот метод и какие он требует аргументы.

Примечание: Можно написать программу, вызывающую библиотечный метод, и эта программа будет компилироваться в режиме конструктора, даже если эта библиотека не существует. Однако, библиотека должна существовать для исполнения данной формы.

4.uses

Ключевое слово Описывает программы внешней библиотеки, которые можно использовать в методе или процедуре.

Синтаксис uses [библиотека|ObjectPAL]
имяПрограммы (списокПараметров)
endUses

Описание Блок uses, описанный в окне объекта Uses, делает программы из внешних библиотек доступными для методов языка ObjectPAL. Эти программы должны быть одного из следующих видов:
- Программы, написанные на языке ObjectPAL и содержащиеся в библиотеках ObjectPAL
- Программы, написанные на языке ObjectPAL и определенные для формы. Синтаксис вызова методов, определенных для формы, такой же, как и при вызове из библиотеки.
- Программы, написанные на языках ассемблер, C, C++ или Pascal и содержащиеся в библиотеке ObjectPAL или библиотеке динамической компоновки Windows (DLL - Dynamic-link library). Библиотека динамической компоновки является библиотекой, содержащей исполняемый код или данные, которые можно присоединять к приложению на стадии выполнения. Используя эти библиотеки, можно добавлять свойства и функции без модификации откомпилированного приложения ObjectPAL.

Блок uses для библиотеки ObjectPAL несколько отличается от блока uses для библиотеки динамической компоновки, поэтому они обсуждаются в различных разделах.

Для того, чтобы использовать методы из библиотеки ObjectPAL или методы, определенные для формы, блок uses записывается в окне объекта Uses в следующем виде:

uses ObjectPAL
[имяМетода ([var | const] списокАргументов) [типВозврата]]*

```
endUses
```

Ключевое слово ObjectPAL определяет, что вызываются методы из библиотек ObjectPAL или форм, а не методы из библиотек динамической компоновки.

Внимание: Перед вызовом метода из библиотеки эта библиотека должна быть открыта. Для вызова метода из формы форма также должна быть открыта и запущена на выполнение.

имяМетода - это имя метода, который нужно вызвать. списокАргументов - разделяемый запятыми список пар аргументов и типов данных, перед которыми могут быть вставлены соответствующие ключевые слова var или const.

Необязательный аргумент типВозврата определяет тип данных значения, возвращаемого методом, если это предусмотрено.

В одном окне Uses можно определить более одного библиотечного метода, а также методы более чем из одной библиотеки.

Внимание: Аргументы и типы данных должны быть описаны в окне Uses в строгом соответствии с их описанием в библиотеке.

В следующем примере описаны два библиотечных метода, buildMenu и calcInterest. buildMenu имеет один аргумент, строковую константу названиеСтраницы. calcInterest - два аргумента: числовую переменную индСтавка (передаваемую по ссылке) и переменную целого типа числоПериодов (передаваемую по значению).

```
uses ObjectPAL
  buildMenu(const названиеСтраницы String)
  calcInterest(var индСтавка Number, числоПериодов SmallInt)Number
endUses
```

Следующая программа, определенная в окне Uses кнопки, описывает метод calcInterest, который затем может вызываться для этой кнопки. Заметим, что пользователь должен описать только те методы библиотеки, которые он будет использовать.

```
uses ObjectPAL
  calcInterest(var индСтавка Number, числоПериодов SmallInt)Number
endUses
```

Другая программа, определенная во встроенном методе кнопки pushButton, открывает библиотеку и вызывает этот метод:

```
method pushButton(var eventInfo Event)
  var
    mathLib  Library
    intRate  Number
    nPeriods SmallInt
    Interest Number
  endVar

  if mathLib.open("mathlib.lsl") then
    intRate = mortgage.intRate.value
    nPeriods = mortgage.nYears.value * 12
    interest = mathLib.calcInterest(intRate, nPeriods)
    interest.view("Interest")
  endIf
endMethod
```

В этом примере точечная нотация определяет путь поиска метода calcInterest. Следующая кон-

струкция предлагает осуществлять поиск в библиотеке, представленной переменной mathLib типа Library.

```
interest = mathLib.calcInterest(intRate, nPeriods)
```

Концепция вызова специальных методов, определенных для формы, аналогична рассмотренной: для ссылки на форму, методы которой вызываются, следует использовать точечную нотацию. В следующем примере предполагается, что ранее была описана переменная формы codeForm, открыта форма, а метод getObjHelp описан в соответствующем окне Uses.

```
interest = codeForm.getObjHelp(self.name)
```

Для использования процедур из библиотеки динамической компоновки следует поместить блок uses в одно из следующих мест:

В окно Uses объекта интерфейса
В текст встроенного метода
В текст специального метода
В текст специальной процедуры

Ответ на вопрос, куда помещать этот блок, зависит от предполагаемой области определения (доступности) процедуры.

Вне зависимости от того, куда помещен этот блок, его основная структура (продемонстрированная в следующем псевдокоде) одинакова:

```
uses имяБиблиотеки
  имяПрограммы (списокПараметров) типВозврата
endUses
```

Аргумент имяБиблиотеки указывает имя файла библиотеки динамической компоновки, причем имяБиблиотеки должно быть правильным именем файла DOS из не более чем восьми символов. Paradox предполагает расширение файлов .DLL или .EXE. Windows осуществляет поиск в следующем порядке:

1. Текущий каталог.
2. Каталог Windows (каталог, содержащий файл WIN.COM). Чтобы получить эту информацию, можно использовать процедуру windowsDir типа FileSystem (обычно это C:\WINDOWS).
3. Системный каталог Windows (каталог, содержащий такие системные файлы, как KERNEL.EXE). Для получения этой информации, можно использовать процедуру windowsSystemDir типа FileSystem (обычно это C:\WINDOWS\SYSTEM).
4. Каталоги, определенные в переменной среды PATH. За дополнительной информацией следует обращаться к документации по системе DOS.
5. Список каталогов, определенных в сети.

Примечание для профессиональных программистов в среде Windows: при вызове программ из уже загруженной библиотеки динамической компоновки (DLL) (например, DLL-библиотеки, загружаемой автоматически системой Windows), вместо имени файла можно использовать имяБиблиотеки для ссылки на имя модуля библиотеки динамической компоновки. Обратитесь к документации по языку программирования за дополнительной информацией об именах модулей DLL.

Блок uses может содержать один или более одного аргумента имяПрограммы, и каждое имя может иметь свой собственный списокПараметров. В нем указываются одно или более имен переменных и типов данных. Если программа возвращает значение, то типВозврата определяет тип данных возвращаемого значения. Средствами ObjectPAL приведенное описание переменных проверяется на точное соответствие с описанием в самой программе, и это единственная проводящаяся проверка.

В блоке uses типы данных описываются следующими ключевыми словами:

Тип данных	ObjectPAL	C++	Pascal

16 битовое целое	CWORD	int	Integer
32 битовое целое	CLONG	long	Longint
64 битовое с пл.точкой	CDOUBLE	double	Double
80 битовое с пл.точкой	CLONGDOUBLE	long double	Extended
указатель	CPTR	char far *	String
двоичные, графические	CHANDLE	Handle (Windows)	THandle (Windows)

Эти ключевые слова могут использоваться только в пределах блока uses. Не следует использовать их в других местах.

Для вызова программы из метода следует предварительно описать переменные, которые будут использоваться как аргументы. Например,

```
; эта часть набирается в окне объекта Uses
uses myStuff ; считываются программы из библиотеки MYSTUFF.DLL
  doSomething(thisNum CLONG, thatNum CLONG) CDOUBLE ; описывается программа
endUses

; теперь модифицируется метод объекта mouseUp
method mouseUp(var eventInfo MouseEvent)
var
  thisNum, thatNum LongInt ; определяются переменные, передаваемые программе
  myResult Number
endVar

thisNum = 3,155.111
thatNum = 5,535.345
myResult = doSomething(ThisNum, ThatNum) ; вызов программы, получение результата
```

В предыдущем примере аргументы в блоке uses...endUses были описаны как CLONG и CDOUBLE, а в методе переменные были описаны как LongInt и Number. Это результат того, что типы данных языка ObjectPAL являются более сложными (и более емкими) чем соответствующие типы данных в языках C или Pascal:

CWORD соответствует SmallInt
CLONG соответствует LongInt
CDOUBLE и CLONGDOUBLE соответствуют Number
CPTR соответствует String
CHANDLE соответствует Binary и Graphic

Примечание:Не следует модифицировать содержание передаваемого аргумента типа CPTR.

Пример В этом примере используются программы из библиотеки динамической компоновки MINMAX.DLL, написанной на языке Turbo Pascal for Windows. Программа на языке Pascal, определяющая библиотеку динамической компоновки, имеет следующий вид:

```
{ это программа на языке Pascal, которая определяет DLL }
library MinMax;

function Min(X, Y: Integer): Integer; export;
begin
  if X < Y then Min := X else Min := Y;
end;

function Max(X, Y: Integer): Integer; export;
begin
  if X > Y then Max := X else Max := Y;
```

```
end;
exports
  Min index 1,
  Max index 2;
begin
end.
```

Далее приводится программа на языке ObjectPAL, в которой используется программа из библиотеки динамической компоновки (DLL). Вначале приводится код для окна Uses, а затем программа, которая модифицирует метод кнопки pushButton:

```
; эта часть набирается в окне Uses для кнопки
uses MInMax ; загружаются программы из MINMAX.DLL
  Min (x CWORD, y CWORD) CWORD ; описываются программы
  Max (x CWORD, y CWORD) CWORD
endUses
```

Следующая программа модифицирует стандартный метод кнопки pushButton:

```
method pushButton(var eventInfo Event)
  var
    x, y, z SmallInt
  endVar
  x = 2
  y = 6
  z = Min(x, y)          ; вызов Min из DLL
  msgInfo("Min", z)
  z = Max(x, y)          ; вызов Max из DLL
  msgInfo("Max", z)
endMethod
```

Как видно из описания работы с библиотекой, это довольно сложный процесс. Намного проще использовать оператор PLAY. Перед использованием метод желательно упаковать. В упакованном виде редактировать его нельзя, а выполняется он быстрее. Упаковывать можно формы, отчеты, библиотеки, скрипты (методы). В данном случае для упаковки надо при открытом листинге использовать: Program / deliver. В результате получим файл с таким же наименованием, но расширением SDL.

Примечание: если расширение не указано, то всегда будет выполняться упакованный формат:

play("num.ssl") - ыполняется не упакованный метод

play("num") -если есть упакованный, то будет выполняться он, иначе единственный который есть с этим названием

На рис.2.36 показана таблица и ее структура для перевода цифрового значения в текстовое.

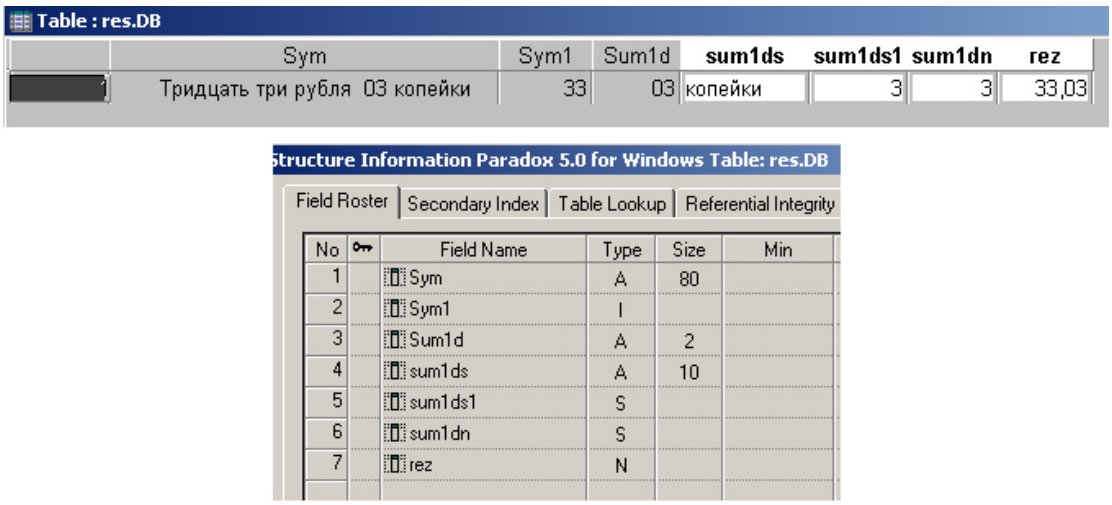


Рис.2.36. Таблица и ее структура для перевода числа в текст

```
method run(var eventInfo Event)
;перевод числа в текст
var
t1,tt,t3 tcursor
c,c1,c2,c3,c4 string
c5,c6,c7,c8,c9 String
c88 string
n1,n2,n3,nn,nn1,nn4 Number
s1,s2,s11,ss SmallInt
d Date
ddm,ddd,ddg SmallInt
endvar

c9=" "
c1=" "
c2=" "
c3=" "
c4=" "
c6=" "
s1=0
s11=0
ss=0

t1.open(":priv:res")
n2=t1."rez"
        ;n2.view()
t1.close()
if n2>32000 then nn=n2/1000
                                        nn4=int(nn)
                                        nn4=nn4*1000
                                        nn=n2-nn4
                                        nn.round(3)
                                        nn1=int(nn)
                                        n3=(nn+0.001)-nn1
                                        nn2=n3*100
                                        s2=int(nn2)
                                        ;s2.view()
                                        n1=nn4+nn1
                else
n2.round(3)
;n2.view()
n1=int(n2)
;n1.view()
n3=(n2+0.001)-n1
;n3.view()
n2=n3*100
s2=int(n2)
;s2.view()
endif
;
n2=n1-900000
if n2>=0 then c6="Девятьсот"
ss=1
```

```
n1=n1-900000
 else
  n2=n1-800000
  if n2>=0 then c6="Восемьсот"
ss=1
n1=n1-800000
          else
           n2=n1-700000
           if n2>=0 then c6="Семьсот"
ss=1
n1=n1-700000
            else
                    n2=n1-600000
                    if n2>=0 then c6="Шестьсот"
ss=1
n1=n1-600000
                     else
                      n2=n1-500000
                      if n2>=0 then c6="Пятьсот"
ss=1
n1=n1-500000
        else
         n2=n1-400000
         if n2>=0 then c6="Четыреста"
ss=1
n1=n1-400000
          else
           n2=n1-300000
           if n2>=0 then c6="Триста"
ss=1
n1=n1-300000
            else
             n2=n1-200000
             if n2>=0 then c6="Двести"
ss=1
n1=n1-200000
              else
               n2=n1-100000
               if n2>=0 then c6="Сто"
ss=1
n1=n1-100000
          else c6=" "
                s1=1
message("нет сотен")
endif
endif
endif
endif
endif
endif
endif
endif
endif
```

```
;\\   десятки тысяч-----------------------------
n2=n1-90000
if n2>=0 then
  if ss=0 then c5="Девяносто"
        else c5="девяносто"
  endif
ss=1
n1=n1-90000
 else
  n2=n1-80000
  if n2>=0 then
   if ss=0 then c5="Восемьдесят"
         else c5="восемьдесят"
  endif
ss=1
n1=n1-80000
  else
   n2=n1-70000
   if n2>=0 then
     if ss=0 then c5="Семьдесят"
           else c5="семьдесят"
  endif
ss=1
n1=n1-70000
    else
     n2=n1-60000
     if n2>=0 then
        if ss=0 then c5="Шестьдесят"
              else c5="шестьдесят"
  endif
ss=1
n1=n1-60000
      else
       n2=n1-50000
       if n2>=0 then
         if ss=0 then c5="Пятьдесят"
               else c5="пятьдесят"
  endif
ss=1
n1=n1-50000
       else
        n2=n1-40000
        if n2>=0 then
          if ss=0 then c5="Сорок"
                else c5="сорок"
  endif
ss=1
n1=n1-40000
          else
           n2=n1-30000
           if n2>=0 then
                    if ss=0 then c5="Тридцать"
                          else c5="тридцать"
```

```
  endif
ss=1
n1=n1-30000
          else
           n2=n1-20000
           if n2>=0 then
              if ss=0 then c5="Двадцать"
                    else c5="двадцать"
  endif
ss=1
n1=n1-20000
            else
             n2=n1-19000
             if n2>=0 then
           if ss=0 then c5="Девятнадцать"
                 else c5="девятнадцать"
  endif
ss=1
n1=n1-19000
                                          else
             n2=n1-18000
             if n2>=0 then
                if ss=0 then c5="Восемнадцать"
                      else c5="восемнадцать"
  endif
ss=1
n1=n1-18000
            else
             n2=n1-17000
             if n2>=0 then
                if ss=0 then c5="Семнадцать"
                      else c5="семнадцать"
  endif
ss=1
n1=n1-17000
            else
             n2=n1-16000
             if n2>=0 then
               if ss=0 then c5="Шестнадцать"
                     else c5="шестнадцать"
         endif
ss=1
n1=n1-16000
            else
             n2=n1-15000
             if n2>=0 then
                if ss=0 then c5="Пятнадцать"
                      else c5="пятнадцать"
  endif
ss=1
n1=n1-15000

            else
```

```
;\\   десятки тысяч-----------------------------
n2=n1-90000
if n2>=0 then
  if ss=0 then c5="Девяносто"
        else c5="девяносто"
  endif
ss=1
n1=n1-90000
 else
  n2=n1-80000
  if n2>=0 then
   if ss=0 then c5="Восемьдесят"
         else c5="восемьдесят"
  endif
ss=1
n1=n1-80000
  else
   n2=n1-70000
   if n2>=0 then
     if ss=0 then c5="Семьдесят"
           else c5="семьдесят"
  endif
ss=1
n1=n1-70000
    else
     n2=n1-60000
     if n2>=0 then
        if ss=0 then c5="Шестьдесят"
              else c5="шестьдесят"
  endif
ss=1
n1=n1-60000
      else
       n2=n1-50000
       if n2>=0 then
         if ss=0 then c5="Пятьдесят"
               else c5="пятьдесят"
  endif
ss=1
n1=n1-50000
       else
        n2=n1-40000
        if n2>=0 then
          if ss=0 then c5="Сорок"
                else c5="сорок"
  endif
ss=1
n1=n1-40000
          else
           n2=n1-30000
           if n2>=0 then
                  if ss=0 then c5="Тридцать"
                        else c5="тридцать"
```

```
  endif
ss=1
n1=n1-30000
          else
           n2=n1-20000
           if n2>=0 then
              if ss=0 then c5="Двадцать"
                    else c5="двадцать"
  endif
ss=1
n1=n1-20000
            else
             n2=n1-19000
             if n2>=0 then
           if ss=0 then c5="Девятнадцать"
                 else c5="девятнадцать"
  endif
ss=1
n1=n1-19000
                                          else
             n2=n1-18000
             if n2>=0 then
                if ss=0 then c5="Восемнадцать"
                      else c5="восемнадцать"
  endif
ss=1
n1=n1-18000
            else
             n2=n1-17000
             if n2>=0 then
                if ss=0 then c5="Семнадцать"
                      else c5="семнадцать"
  endif
ss=1
n1=n1-17000
            else
             n2=n1-16000
             if n2>=0 then
               if ss=0 then c5="Шестнадцать"
                     else c5="шестнадцать"
         endif
ss=1
n1=n1-16000
            else
             n2=n1-15000
             if n2>=0 then
                if ss=0 then c5="Пятнадцать"
                      else c5="пятнадцать"
  endif
ss=1
n1=n1-15000

            else
```

```
                else c4="четяре тысячи"
  endif
ss=1
n1=n1-4000
          else
           n2=n1-3000
                                        if n2>=0 then
             if ss=0 then c4="Три тысячи"
                   else c4="три тысячи"
  endif
ss=1
n1=n1-3000
            else
             n2=n1-2000
             if n2>=0 then
                if ss=0 then c4="Две тысячи"
                      else c4="две тысячи"
  endif
ss=1
n1=n1-2000
              else
               n2=n1-1000
               if n2>=0 then
                  if ss=0 then c4="Одна тысяча"
                        else c4="одна тысяча"
  endif
ss=1
n1=n1-1000
              else
                  c4=" "
                  s1=s1+100
if s1<=110 then c4="тысяч"
else
message("нет сотен")
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
;\\  сотни-----------------------------------
n2=n1-900
if n2>=0 then
   if ss=0 then c3="Девятьсот"
         else c3="девятьсот"
   endif
ss=1
n1=n1-900
 else
```

```
  n2=n1-800
  if n2>=0 then
      if ss=0 then c3="Восемьсот"
             else c3="восемьсот"
   endif
ss=1
n1=n1-800
   else
    n2=n1-700
    if n2>=0 then
       if ss=0 then c3="Семьсот"
              else c3="семьсот"
   endif
ss=1
n1=n1-700
     else
      n2=n1-600
      if n2>=0 then
         if ss=0 then c3="Шестьсот"
                else c3="шестьсот"
   endif
ss=1
n1=n1-600
       else
        n2=n1-500
        if n2>=0 then
           if ss=0 then c3="Пятьсот"
                  else c3="пятьсот"
   endif
ss=1
n1=n1-500
        else
         n2=n1-400
         if n2>=0 then
            if ss=0 then c3="Четыреста"
                   else c3="четыреста"
   endif
ss=1
n1=n1-400
           else
            n2=n1-300
            if n2>=0 then
               if ss=0 then c3="Триста"
                      else c3="триста"
   endif
ss=1
n1=n1-300
             else
              n2=n1-200
              if n2>=0 then
                 if ss=0 then c3="Двести"
                        else c3="двести"
   endif
```

```
ss=1
n1=n1-200
              else
               n2=n1-100
               if n2>=0 then
                  if ss=0 then c3="Сто"
                        else c3="сто"
  endif
ss=1
n1=n1-100
              else c3=""
                  s11=1
 message("нет сотен")
endif
endif
endif
endif
endif
endif
endif
endif
endif
;\\    десятки ------------------------------------------------
n2=n1-90
if n2>=0 then
   if ss=0 then c2="Девяносто"
          else c2="девяносто"
  endif
ss=1
n1=n1-90
 else
  n2=n1-80
  if n2>=0 then
     if ss=0 then c2="Восемьдесят"
            else c2="восемьдесят"
  endif
ss=1
n1=n1-80
  else
   n2=n1-70
   if n2>=0 then
      if ss=0 then c2="Семьдесят"
             else c2="семьдесят"
  endif
ss=1
n1=n1-70
    else
     n2=n1-60
     if n2>=0 then
        if ss=0 then c2="Шестьдесят"
               else c2="шестьдесят"
  endif
ss=1
```

```
n1=n1-60
      else
       n2=n1-50
       if n2>=0 then
          if ss=0 then c2="Пятьдесят"
                else c2="пятьдесят"
  endif
ss=1
n1=n1-50
       else
        n2=n1-40
        if n2>=0 then
           if ss=0 then c2="Сорок"
                 else c2="сорок"
  endif
ss=1
n1=n1-40
         else
          n2=n1-30
          if n2>=0 then
             if ss=0 then c2="Тридцать"
                   else c2="тридцать"
  endif
ss=1
n1=n1-30
           else
            n2=n1-20
            if n2>=0 then
               if ss=0 then c2="Двадцать"
                     else c2="двадцать"
  endif
ss=1
n1=n1-20
             else
              n2=n1-19
              if n2>=0 then
                 if ss=0 then c2="Девятнадцать"
                       else c2="девятнадцать"
  endif
ss=1
n1=n1-19
             else
              n2=n1-18
              if n2>=0 then
                 if ss=0 then c2="Восемнадцать"
                       else c2="восемнадцать"
  endif
ss=1
n1=n1-18
             else
              n2=n1-17
              if n2>=0 then
                 if ss=0 then c2="Семнадцать"
```

```
                    else c2="семнадцать"
  endif
ss=1
n1=n1-17
            else
             n2=n1-16
             if n2>=0 then
                if ss=0 then c2="Шестнадцать"
                      else c2="шестнадцать"
  endif
ss=1
n1=n1-16
            else
             n2=n1-15
             if n2>=0 then
                if ss=0 then c2="Пятнадцать"
                      else c2="пятнадцать"
  endif
ss=1
n1=n1-15
            else
             n2=n1-14
             if n2>=0 then
                if ss=0 then c2="Четырнадцать"
                      else c2="четырнадцать"
  endif
ss=1
n1=n1-14
            else
             n2=n1-13
             if n2>=0 then
                if ss=0 then c2="Тринадцать"
                      else c2="тринадцать"
  endif
ss=1
n1=n1-13
            else
             n2=n1-12
             if n2>=0 then
                if ss=0 then c2="Двенадцать"
                      else c2="двенадцать"
  endif
ss=1
n1=n1-12
            else
             n2=n1-11
           if n2>=0 then
             if ss=0 then c2="Одиннадцать"
                   else c2="одиннадцать"
  endif
ss=1
n1=n1-11
            else
```

```
            n2=n1-10
           if n2>=0 then
             if ss=0 then c2="Десять"
                   else c2="десять"
  endif
ss=1
n1=n1-10
            else
             c2=""
             s11=s11+10
             message("нет десятков рублей")
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif
;\\   единицы ------------------------------------------
n2=n1-9
if n2>=0 then
  if ss=0 then c1="Девять рублей"
        else c1="девять рублей"
  endif
ss=1
n1=n1-9
 else
  n2=n1-8
  if n2>=0 then
    if ss=0 then c1="Восемь рублей"
          else c1="восемь рублей"
  endif
ss=1
n1=n1-8
  else
   n2=n1-7
   if n2>=0 then
      if ss=0 then c1="Семь рублей"
            else c1="семь рублей"
  endif
ss=1
n1=n1-7
```

```
    else
     n2=n1-6
     if n2>=0 then
        if ss=0 then c1="Шесть рублей"
              else c1="шесть рублей"
  endif
ss=1
n1=n1-6
      else
       n2=n1-5
       if n2>=0 then
         if ss=0 then c1="Пять рублей"
               else c1="пять рублей"
  endif
ss=1
n1=n1-5
        else
         n2=n1-4
         if n2>=0 then
           if ss=0 then c1="Четыре рубля"
                 else c1="четыре рубля"
  endif
ss=1
n1=n1-4
          else
           n2=n1-3
           if n2>=0 then
              if ss=0 then c1="Три рубля"
                    else c1="три рубля"
  endif
ss=1
n1=n1-3
            else
             n2=n1-2
             if n2>=0 then
               if ss=0 then c1="Два рубля"
                     else c1="два рубля"
  endif
ss=1
n1=n1-2
              else
               n2=n1-1
               if n2>=0 then
                 if ss=0 then c1="один рубль"
                       else c1="один рубль"
  endif
ss=1
n1=n1-1
              else
                s11=s11+100
                c1=""
  if s11<=110 then c1="рублей"
 else
```

```
 ;message("нет сотен")
c1="рублей"
endif
endif
endif
endif
endif
endif
endif
endif
endif
endif

t1.open(":priv:res")
t1.edit()

if t1."rez"<0.01 then c2=" ноль "
endif
;-----------------
t1."sym"=c6+c9+c5+c9+c4+c9+c3+c9+c2+c9+c1+"  "+t1."sum1d"
t1.endedit()
t1.close()

executeQBEfile("dra2a") ;in dra2a sym=sym+sum1d="копеек."
errorshow()

endmethod
```

Заключение

Материал, изложенный в книге полностью провер и все практические примеры взяты из реальных задач. В основном все написано на Parsdox 7 (Borland), но и в версии 9 все работает без ошибок, т.е. все примеры работают в обеих версиях.

ПРИЛОЖЕНИЕ 1
Перечень команд PAL

abs	Number	возвращает абсолютное значение числа
accessRight	DDE	сообщает о правах доступа к файлу (называемых также атрибутами файла)
acos	Number	возвращает арккосинус числа
action	Form	исполняет команду действие
action	TableView	выполняет команду-действие
action	UiObject	выполняет указанное действие
actionClass	ActionEvent	возвращает номер класса события ActioEvent
add	TCursor	добавляет записи из одной таблицы в другую
addAlias	Session	устанавливает псевдоним базы данных для текущего сеанса
addArray	Menu	добавляет элементы массива в конец списка меню
addArray	PopUpMenu	добавляет элементы массива в конец всплывающего меню
addBar	PopUpMenu	добавляет во всплывающее меню вертикальную разделительную черту
addBreak	Menu	начинет новую строку меню
addBreak	PopUpMenu	начинает новую колонку всплывающего меню
addLast	Array	добавляет элемент в конец массива переменной длиы
addMatch	String	осуществляет поиск указанной строки в тексте
addPassword	Session	вводит паролдь доступа к защищенной таблице
addPopUp	Menu	связывает всплывающее меню с элементом строки меню
addPopUp	PopUpMenu	добавляет всплывающее меню в строку меню
addSeparator	PopUpMenu	добавляет во всплывающее меню горизонтальную разделительную черту
addStaticText	Menu	добавляет в меню текст который не является пунктом меню
addStaticText	PopUpMenu	добавляет во всплывающее меню текстовую строку не являющююся пунктом меню
addtext	Menu	добавляет в меню текст как пункт меню
addtext	PopUpMenu	добавляет во всплывающее меню текстовую строку как пункт меню
advancedWildcardsinLocate	Session	устанавливает возможность использования в данном сеансе расширенных операторов задания шаблона
ansiCode	String	возвращает ANSI-код символа
append	Array	добавляет содержимое массива в конец другого массива
asin	Number	возвращает арксинус числа
atFirst	TCursor	сообщает установлен ли табличный указатель на первую запись таблицы
atFirst	UiObject	проверяет установлен ли указатель на первой записи таблицы

atLast	TCursor	сообщает установлен ли табличный указатель на последнюю запись таблицы
atLast	UiObject	проверяет установлен ли указатель на последней записи таблицы
attach	Form	связывает переменную типа Form с открытой формой
attach	Table	связывает переменную типа Table с таблицей на диске
attach	TCursor	связывает табличный указатель с объектом интерфейса
attach	UiObject	связывает переменную типа UiObject с указанным объектом интерфейса
attachToKeyViol	TCursor	устанавливает табличный указатель на существующую запись с таким же ключем как и запись для которой делалась попытка занесения
beep	System	подает стандартный звуковой сигнал Windows
beginTransaction	DataBase	запускает транзакцию
bitAND	Longlnt	выполняет побитовую операцию AND над двумя значениями
bitOR	Longlnt	выполняет побитовую операцию OR над двумя значениями
bitOR	SmallInt	выполняет побитовую операцию OR над двумя значениями
bitXOR	LongInt	выполняет побитовую операцию XOR над двумя значениями
bitXOR	SmallInt	выполняет побитовую операцию XOR над двумя значениями
blank	AnyType	возвращает пустое значение
blankAsZero	Session	устанавливает режим интерпритации пустых значений как нулевых
bot	TCursor	отслеживает попытку перехода на запись до первой записи таблицы
breakApart	String	разделяет строку на подстроки
bringToTop	Form	размещает окно поверх набора экранных окон и делает его активным
broadCastAction	UiObject	сообщает о действии объекту
cancelEdit	TCursor	заканчивает режим редактирования без сохранения изменений в текущей записи
cancelEdit	UiObject	отменяет изменения записи без выхода из режима редактирования
canReadFromClipboard	OLE	проверяет можно ли перенести в переменную OLE объект OLE из временного буфера
cAverage	Table	возвращает среднее значение поля (столбца) таблицы
cAverage	TCursor	возвращает среднее значение поля (столбца) таблицы
cCount	Table	возвращает число непустых значений поля (столбца) тааблицы

cCount	TCursor	возвращает число значений поля (столбца) таблицы
ceil	Number	округляет числовое выражение до ближайшего сверху целого числа
char	KeyEvent	возвращает символ соответствующий нажатой клавише
charAnsiCode	KeyEvent	возвращет знчение код ANSI. соответствующе нажтию клавиши
chr	String	возвращает односимвольную строку в коде ANSI
chrOEM	String	возвращает односимвольную строку в коде OEM
chrToKeyName	String	возвращает строку содержащую виртуальный ключевой код односимвольной строки
close	DataBase	закрывает базу данных
close	DDE	закрывает связь DDE
close	Form	закрывает окно
close	Library	закрывает библиотеку
close	Report	закрывает окно
close	Session	закрывает сеанс
close	System	закрывает текущую форму
close	TableView	закрывает окно таблицы
close	TCursor	закрывает табличный указатель
close	TextStream	закрывает текстовый файл
cMax	Table	возвращает максимальное значение поля (столбца) таблицы
cMax	Table	возвращает максимальное значение поля (столбца) таблицы
cMin	Table	возвращает минимальное значение поля (столбца) таблицы
cMin	Table	возвращает минимальное значение поля (столбца) таблицы
cNpv		возвращает итоговое сальдо для набора значений в поле (столбце) таблицы расчитанное на основании учетной или % ставки
cNpv	TCursor	возвращает итоговое сальдо для набора значений в поле (столбце) таблицы расчитанное на основании учетной или % ставки
commit	TextStream	записывае содержимое текстового буфера на диск
commitTransaction	DataBase	заносит все проишедшие в процессе транзакции изменения
compact		фактически изымает из таблицы удаленные записи
compact	TCursor	удаляет из таблицы удаленные записи
constantNameToValue	System	возвращает числовое значение константы
constantValueName	System	сообщает имя константы
contains	Array	осуществляет поиск символьного шаблона среди элементов массива

contains	DynArray	осуществляет поиск заданного значения среди индексов динамического массива
contains	Menu	сообщает содержится ли данный пункт в меню
convertPointWithRespectTo	UiObject	изменяет систему отчета для координат точки
copy	FileSystem	копирует файл
copy	Table	копирует таблицу
copy	TCursor	копирует таблицу
copyFromArray	TCursor	копирует данные из массива в поля текущей записи
copyFromArray	UiObject	копирует данные из массива в запись таблицы
copyRecord	TCursor	копирует запись на которую установлен один табличный указатель на запись, связанную с другими табличными указателями
copyToArray	TCursor	копирует поля текущей записи в массив
copyToArray	UiObject	копирует данные из записи в массив
cos	Number	возвращает косинус угла
cosh	Number	возвращает гиперболический косинус числа или выражения
count	Menu	возвращает число элементов меню
countOf	Array	подсчитывает сколько элементов с данным значением содержится в массиве
cpuClockTime	System	возвращает число миллисекунд после начальной загрузки
create	Form	создает пустую форму в окне конструктора
create	TextStream	создает текстовый файл
create	UiObject	создает объект
cSamStd	Table	возвращает несмещенное среднеквадратичное отклонение для поля (столбца) таблицы
cSamStd	TCursor	возвращает несмещенное среднеквадратичное отклонение для поля (столбца) таблицы
cSamVar	Table	возвращает несмещенную дисперсию для поля (столбца) таблицы
cSamVar	Table	возвращает несмещенную дисперсию для поля (столбца) таблицы
cStd	Table	возвращает смещенное среднеквадратичное отклонение для поля (столбца) таблицы
cStd	Table	возвращает смещенное среднеквадратичное отклонение для поля (столбца) таблицы
cSum	Table	возвращает сумму значение числового поля (столбца) таблицы
cSum	Table	возвращает сумму значение числового поля (столбца) таблицы
currency	Currency	преобразует значение к типу Currency
currentPage	Report	возвращает номер текущей страницы отчета
currRecord	Table	считывает текущую запись в буфер записи
currRecord	UiObject	считывает текущую запись в буфер записи
cVar	Table	возвращает смещенную дисперсию поля (столбца) таблицы

cVar	Table	возвращает смещенную дисперсию поля (столбца) таблицы
dalete	DataBase	удаляет таблицу из базы данных
dalete	FileSystem	удаляет файл
dalete	UiObject	удаляет объект из формы
daleteDir	FileSystem	удаляет каталог
daleteRecord	Table	удаляет запись на которой установлен табличный указатель
daleteRecord	UiObject	удаляет текущую запись из таблицы
dasableBreakMessage	Form	запрещает прерывание программы командой CTRL/Break
data	MenuEvent	возвращает информацию о собитии типа MenuEvent
dataModelAddTable	System	включает таблицу в модель данных формы
dataModelHasTable	System	сообщает входит ли таблица в модель данных формы
dataModelRemiveTable	System	удаляет таблицу из модели данных формы
dataType	AnyType	возвращает строку представляющую тип данных переменной
date	Date	преобразует значение к типу Date
dateTime	DateTime	преобразует значение к типу DateTime
dateVal	Date	возвращает значение с типом даты
day	DateTime	выделяет день месяца в переменной типа DateTime
daysinMonth	DateTime	возвращает число дней в месяце
debug	System	останавливает выполнение метода и активизирует отладчик
delayScreenUpdates	Form	включает и выключает задержку обновления экрана
design	Form	переводит исполняемую форму в окно конструктора
design	Report	переводит текущий отчет из режима просмотра в режим конструктора
didFlyArray	Table	отслеживает перескок текущей записи в другую позицию в результате изменения значения ключа
distance	Point	возвращает расстояние между точками
dlgAdd	System	вызывает окно диалога добавления записей
dlgCopy	System	вызывает окно диалога Копирование
dlgCreate	System	вызывает окно диалога Создание
dlgDelete	System	вызывает окно диалога Удаление
dlgEmpty	System	вызывает окно диалога Очистка таблицы
dlgExport	System	вызывает окно диалога Экспорт таблицы
dlgInportASCIIFix	System	вызывает окно диалога Импорт текстового файла с фиксированной длиной строки
dlgInportASCIIVar	System	вызывает окно диалога Импорт текстового файла с разделителями

dlgInportSpreadsheet	System	вызывает окно диалога Импорт из элеткронной таблицы
dlgNetDrivers	System	вызывает окно диалога Текущие драйверы
dlgNetLocks	System	создается и выврдится таблица с информацией о блокировках
dlgNetRefresh	System	вызывается окно диалога Интервал автоматического обновления
dlgNetRetry	System	вызывается окно диалога Период автоматического повторения
dlgNetSetLocks	System	вызывает окно диалога Блокировки таблицы
dlgNetSystem	System	вызывает окно диалога Системная информация
dlgNetUserName	System	вызывает окно диалога Сетевое имя пользователя
dlgNetWho	System	вызывает окно диалога текущие пользователи
dlgRaname	System	вызывает окно диалога Переименование
dlgRestructure	System	вызывает окно диалога Изменение структуры
dlgSort	System	вызывает окно диалога Сортировка таблицы
dlgSubtract	System	вызывает окно диалога Вычитание записей
dlgTaleInfo	System	вызывает окно диалога Информация о структуре
dmAddTable	Form	добавляет таблицу к модели данных формы
dmGet	Form	извлекает значение поля из таблицы входящей в модель данных
dmHasTable	Form	сообзает содержится ли таблица в модели данных формы
dmLinkToindex	Form	связывает две таблицы в модели данных формы базируясь на списке полей и названии индексов
dmPut	Form	заносит данные в таблицу входящую в модель данных формы
dmRemoveTable	Form	удаляет таблицу из модели данных формы
dmUnlink	Form	устраняет связ между двумя таблицами в модели данных
dow	DateTime	возвращает день недели для переменной типа DateTime
dowOrd	DateTime	возвращает порядковый номер дня недели
doy	DateTime	возвращает порядковый номер дня в году
drives	FileSystem	возвращает буквы определенных в системе дисков доступных для Windows
dropIndex	Table	удаляет индексный файл связанный с таблицей
dropIndex	Table	удаляет индексный файл связанный с таблицей
edit	OLE	запускает сервер OLE и разрешает пользователю редактирование объекта или выполнение других действий
edit	Table	переводит табличный указатель в режим редактирования
edit	UiObject	переводит таблицу в режим редактирования
empty	Array	очищает массив
empty	DynArray	удаляет все элементы динамического массива

empty	Menu	удаляет все элементы меню
empty	Table	удаляет все записи из таблицы
empty	UiObject	удаляет все записи из таблицы
end	Table	перемещает табличный указатель на последнюю запись в таблице
end	TextStream	устанавливает указатель файла на конец текстового файла
end	UiObject	осуществляет переход на последнюю запись в таблице
endEdit	TCursor	заканчивает редактирование и вносит изменения проделанные для текущей записи
endEdit	UiObject	осуществляет выход из режима редактирования и принимает изменения текущей записи
enumAliasLoginInfo	Session	помещает данные об указанном псевдониме в таблицу
enumAliasNames	Session	создает таблицу Paradox со списком имен псевдонимов баз данных доступным в текущем сеансе
enumDataBaseTables	Session	создает таблицу Paradox со списком таблиц в базе данных
enumDesktopWindowsName	System	создает таблицу или массив со списком открытых окон Paradox
enumDriverCapabilities	Session	создает три таблицы Paradox со списком характеристик текущего драйвера
enumDriverInfo	Session	создает список доступных драйверов
enumDriverNames	Session	создает таблицу Paradox со списком имен драйверов доступных в текущем сеансе
enumDriverTopics	Session	создает таблицу Paradox со списком доступных в текущем сеансе описаний для каждого типа драйверов
enumEngineInfo	Session	создает таблицу Psrsdox со списком свойств набора методов доступа открытого интерфейса к базам данных (ODAPI)
enumFieldNames	Table	заполняет массив именами полей таблицы
enumFieldNames	TCursor	заполняет массив именами полей таблицы
enumFieldNames	UiObject	заполняет массив именами полей таблицы
enumFieldNamesInIndex	Table	заполняет массив именами ключевых полей таблицы
enumFieldNamesInIndex	TCursor	заполняет массив именами полей индекса таблицы
enumFieldStruct	Table	создает таблицу Paradox описывающую структуру полей таблицы
enumFileList	FileSystem	записывает информацию о файлах в таблицу или массив FileSystem
enumFolder	Session	создает таблицу Paradox или массив со списком файлов в папке
enumFonts	Session	создает таблицу со списком шрифтов в системе пользователя
enumFormNames	System	создается массив со списком открытых форм

enumIndexStruct	Table	создает таблицу Paradox описывающую структуру вторичных индексов таблицы
enumIndexStruct	Table	создает таблицу Paradox описывающую структуру вторичных индексов таблицы на котором установлен табличный указатель
enumLocks	TCursor	создает таблицу Paradox описывающую блокировки установленные в данный момент для объекта TCursor
enumLocks	UiObject	создает таблицу Paradox описывающую блокировки установленные в данный момент для объекта интерфейса; возвращает количество блокировок
enumObjectNames	UiObject	заполняет массив именами объектов в форме
enumOpenDatabases	Session	создает таблицу Paradox со списком открытых баз данных
enumRefIntStruct	Table	создает таблицу Paradox содержащую информацию о целостности данных таблицы
enumRefIntStruct	Table	создает таблицу Paradox с информацией о целостности данных для объекта TCursor
enumReportNames	System	создает массив со списком открытых отчетов
enumRTLClassNames	System	создает таблицу со списком типов объектов ObjestPal
enumRTLConstants	System	создает таблицу со списком констант ObjectPal
enumRTLMethods	System	создает таблицу со списком методов ObjectPal
enumSecStruct	Table	создает таблицу Paradox с информацией о защите таблицы
enumSource	Form	создает таблицу со списком методов для каждого объекта в форме
enumSource	Library	записывает библиотечные программы в таблицу Paradox
enumSource	UiObject	заполняет таблицу исходными текстами программ методов формы
enumSourceToFile	Form	создает файл со списком методов для каждого объекта в форме
enumSourceToFile	Library	записывает библиотечные программы в текстовый файл
enumSourceToFile	UiObject	записывает в текстовый файл исходный текст программы для формы или объекта
enumTableProperties	TCursor	записывает свойства объекта TCursor в таблицу Paradox
enumUlClasses	UiObject	создает таблицу со списком классов объектов интерфейса
enumUlObjectNames	Form	создает таблицу с именами объектов интерфейса
enumUlObjectNames	Report	создает таблицу со списком содержащихся в отчете объектов интерфейса
enumUlObjectNames	UiObject	считывает имена всех объектов в форме и записывает их в таблицу
enumUlObjectPropeties	Form	создает таблицу со списком свойств каждого из объектов интерфейса содержащихся в форме

enumUlObjectPropeties	Report	создает таблицу со списком свойств для всех содержащихся в отчете объектов интерфейса
enumUlObjectPropeties	UiObject	считывает свойства каждого объекта в форме и записывает эти данные в таблицу Paradox
enumUsers	Session	создает таблицу Paradox со списком пользователей имеющих открытый канал к методам доступа ODAPI
enumVerbs	OLE	создает динамичесий массив содержащий список директорий потдерживаемых сервером OLE
enumWindowsNames	System	создает таблицу или массив со списком открытых окон
eof	TCursor	отслеживает попытку выхода за последнюю запись таблицы
eof	TextStream	отслеживает попытку выхода за пределы текстового файла
errorClear	System	очищает стек ошибок
errorCode	Event	проверяет состояние флага ошибки
errorCode	System	возвращает число описывающее последнюю ошибку на стадии выполнения или состояние ошибки
ErrorLog	System	вводит информацию в стек ошибок
errorMessage	System	возвращает текст последнего сообщения об ошибке
errorPop	System	удаляет верхний слой информации из стека ошибок
errorShow	System	вызывает на экран окно диалога ошибок
errorTrapOnWarnings	System	определяет будут ли предупреждения рассматриваться как критические ошибки
exchange	Array	обменивается содержимое двух ячеек массива
execMethod	Library	вызывает специальный метод не требующий аргументов
execMethod	UiObject	вызывает специальный метод не требующий аргументов
execute	DDE	посылает команды через связ DDE
execute	System	исполняет команду DDE
executeQBE	DataBase	выполняет запрос по образцу
executeQBE	Query	выполняет запрос по образцу
executeQBEFile	DataBase	создается и исполняется файл запроса по образцу
executeQBEString	DataBase	выполняется запрос по образцу записанный в виде строки
executeSQL	SQL	выполняет инструкции SQL
executeSQLFile	SQL	выполняет инструкции SQL содержащиеся в файле
executeSQLString	SQL	выполняет инструкции SQL содержащиеся в текстовой строке
existDrive	FileSystem	сообщает подключен ли данный диск к системе
exit		закрывает сеанс Paradox и возвращается в Windows

exp	Number	возвращает экспоненту числа
exportASCIIFix	System	экспортирует данные из таблицы в текстовый файл с записями фиксированной длины
exportASCIIVar	System	экспортирует данные из таблицы в текстовый файл с разделителями
exportSpreadsheet	System	экспортирует данные из таблицы базы данных в электронную таблицу
fail	System	вызывает появление ошибки в методе
familyRights	Table	проверяет права пользователя создавать и модефицировать объекты в семействе таблицы
familyRights	TCursor	проверяет права пользователя создавать и модефицировать объекты связанные с таблицей
fieldName	Table	возвращаетет имя поля таблицы по заданному номеру поля
fieldNo	Table	возвращает позицию поля в таблице
fieldNo	TCursor	возвращает позицию поля в таблице
fieldRights	TCursor	проверяет имеет ли пользователь права на чтение или модификацию поля в таблице
fieldSize	TCursor	возвращает размер поля
fieldType	Table	возвращает тип поля в таблице
fieldType	TCursor	возвращает тип данных поля
fieldUnits2	TCursor	возвращает количество десятичных разрядов определенных для числового поля в таблице dBASE
fieldValue	TCursor	считывает значение указанного поля
fileDrowser	System	вызывает окно поиска фалов и возвращает имена одного или нескольких файлов выбираемых пользователем
fill	Array	заполняет массив одним значением
fill	String	возвращает строку содержащую набор символов
findFirst	FileSystem	осуществляет поиск файлов с заданным именем в файловой системе
findNext	FileSystem	осуществляет поиск нескольких файлов с заданным именем в файловой системе
forceRefresh	TCursor	устанавливает TCursor на текущие данные таблицы
forceRefresh	UiObject	синхронизирует объект интерфейса с данными в связанной с ним таблице а также инициирует пересчет высичляемых полей
format	String	возвращает отформатированную строку для вывода на принтер
formatDelete	System	удаляет формат
formatExist	System	сообщает существует ли данный формат
formatSetCurrencyDefault	System	задает стандартный экранный формат для значений типа Currency
formatSetDateDefault	System	задает стандартный экранный формат для значений типа Date

formatSetDateTimeDefault	System	задает стандартный экранный формат для значений типа DateTime
formatSetLogicalDefault	System	задает стандартный экранный формат для значений типа Logical
formatSetLongintDefault	System	задает стандартный экранный формат для значений типа Longint
formatSetNumberDefault	System	задает стандартный экранный формат для значений типа Number
formatSetSmallIntDefault	System	задает стандартный экранный формат для значений типа SmallInt
formatSetTimeDefault	System	задает стандартный экранный формат для значений типа Time
formReturn	Form	возвращает управление приостановленному методу
freeDiskSpace	FileSystem	возвращает объем свободного пространства на диске
fullName	FileSystem	возвращает полное имя файла
fv	Number	возвращает итоговую сумму серии равных платежей
getAliasPath	Session	возвращает путь соответствующий указанному псевдониму
getAliasPropertygetAliasPath	Session	возвращает значение указанного свойства для заданного псевдонима
getBoundingBox	UiObject	возвращает координаты рамки органичивающей объект
getDestination	MoveEvent	сообщает какой объект является адресатом события типа MoveEvent
getDir	FileSystem	возвращает имя каталога на который указывает переменная типа FileSystem
getDrive	FileSystem	возвращает букву диска на который указывает переменная типа FileSystem
getFileAccessRights	FileSystem	возвращает строку описывающую права доступа к файлу (называемыми также атрибутами файла)
getKey	DynArray	загружает индексы существующего динамического массива в массив переменной длины
getLanguageDriver	TCursor	возвращает имя текущего языкового драйвера для таблицы
getLanguageDriverDesc	TCursor	возвращает описание текущего языкового драйвера для таблицы
getMenuChoiceAttribute	Menu	возвращает экранные атрибуты для пункта меню
getMousePosition	MouseEvent	возвращает позицию мыши
getMouseScreanPosition	System	возвращает позицию мыши как значение Point
getNetUserName	Session	возвращает сетевое имя пользователя используемое в данном сеансе
getObjectHit	Logical	создает указатель для объекта интерфейса принимающего событие
getPosition	Form	сообщает позицию окна на экране

getPosition	UiObject	определяет позицию объекта
getProperty	UiObject	возвращает значение указанного свойства
getPropertyAsString	UiObject	возвращает значение указанного свойства как строку
getRGB	UiObject	определяет красную зеленую и синюю компоненты цвета
getServerName	OLE	сообщает соответствующее объекту OLE имя сервера
getTarget	Event	создает указатель адресата события
getTitle	Form	возвращает текст заголовка окна
getValidFileExtentions	FileSystem	возвращает допустимые расширения имени для заданного объекта
grow	Array	увеличивает длину массива переменной длины
hasMenuChoiceAttribute	Menu	сообщает содержит ли данный пункт меню указанный экранный атрибут
hasMouse	UiObject	сообщает установлен ли указатель мыши на объекте
helpOnHelp	System	выводит информацию об использовании справочной системы
helpQuit	System	передает справочной системе сообщение о закрытии
helpSetIndex	System	задает индекс справочной системе
helpShowContext	System	выводит справку соответствующую выбранному контексту
helpShowIndex	System	выводит индекс указанного файла справочной системы
helpShowTopic	System	выводит текст справки для указанного раздела
helpShowTopicInKeywordTable	System	выводит сооб.спавочной системы для раздела задаваемого ключевым словом из альтернативной табл. ключевых слов
hide	Form	делает окно невидимым
HideSpeedBar	Form	делает невидимым панель управления
home	TCursor	осуществляет переход к первой записи таблицы
home	TextStream	установливает указатель на начало текстового файла
home	UiObject	осуществляет переход на первую запись в таблице
hour	DateTime	выделяет число часов из переменной типа DateTime
id	ActionEvent	возвращает идентификационный номер события ActionEvent
id	MenuEvent	возвращает идентификатор события типа MenuEvent
ignoreCaseInLocate	Session	устанавливает задаваемого ключевым словом из альтернативной табл. ключевых слов
ignoreCaseInStringCompares	String	задает или отменяет учет регистра символов при сравнении строк
importASCIIFix	System	импортирует в таблицу данные из текстового файла с фиксированной длинной строк

importASCIIVar	System	импортирует данные из текстового файла с разделителями в таблицу
importSpreadsheet	System	импортирует данные из электронной таблицы
indexOf	Array	возвращает позицию элемента в массиве
initRecord	TCursor	очищает буфер записи
insert	Array	возвращает одну или несколько пустых ячеек в массив
insertAfter	Array	востанавливает элемент данных в массиве переменной длины после указанного элемента
insertAfteRecord	TCursor	вставляет запись в таблицу после текущей записи
insertAfteRecord	UiObject	вставляет запись в таблицу после текущей записи
insertBeforeRecord	TCursor	вставляет запись в таблицу до текущей записи
insertBeforeRecord	UiObject	вставляет запись в таблицу до текущей записи
insertFirst	Array	вставляет элемент в начало массива
insertRecord	TCursor	добавляет запись в таблицу
insertRecord	UiObject	добавляет запись в таблицу
int	SmallInt	преобразует значение к целочисленному типу
isAbove	Point	сообщает находится ли точка выше другой точки
isAdvancedWildcardsInLocate	Session	сообщает используется ли при операциях поиска в текущем сеансе расширенные операторы шаблонов
isAitKeyDown	KeyEvent	сообщает была ли нажата клавиша ALT во время события клавиатуры
isAssigned	AnyType	сообщает присвоено ли переменной значение
isAssigned	DataBase	сообщает присвоено ли значение переменной типа Database
isAssigned	Session	сообщает присвоено ли значение переменной типа Session
isAssigned	Table	проверяет было ли присвоено значение переменной типа Table
isBelow	Point	сообщает находится ли точка ниже другой точки
isBlank	AnyType	проверяет имеет ли значение пустое значение
isBlankZero	Session	сообщает будут ли рассматриваться при вычислениях пустые значения как нулевые
isContainerValid	UiObject	проверяет допустимость контейнера объекта
isControlKeyDown	KeyEvent	проверяет была ли нажата клавиша CTRL во время события клавиатуры
isControlKeyDown	MouseEvent	сообщает удерживает ли пользователь в нажатом состоянии клавишу CTRL во время события клавиатуры
isDir	FileSystem	сообщает является ли заданная строка именем каталога
isEdit	TCursor	проверяет находится ли табличный указатель в режиме редактирования

isEdit	UiObject	проверяет находится ли объект в режиме редактирования
isEmpty	Table	проверяет наличие записей в таблице
isEmpty	TCursor	проверяет наличие записей в таблице
isEmpty	UiObject	проверяет наличие записей в таблице
isEncrypted	Table	проверяет является ли таблица зашифрованной
isEncrypted	TCursor	проверяет является ли таблица зашифрованной
isExecuteQBEFileLocal	DataBase	выясняет как будет выполняться запрос QBE: локально или на сервере
isExecuteQBELocal	DataBase	выясняет как будет выполняться запрос QBE: локально или на сервере
isExecuteQBEStringLocal	DataBase	выясняет как будет выполняться запрос QBE: локально или на сервере
isFile	FileSystem	сообщает является ли заданная строка именем файла в текущей системе файлов
isFirstTime	Event	сообщает оьрабатывается ли событие формой первый раз до передачи объекту
isFixed	FileSystem	сообщает является ли данный диск фиксированным
isFixedType	AnyTape	проверяет был ли тип данных переменной описан в явном виде
isFromUI	KeyEvent	сообщает генерируется ли событие клавиатуры при взаимодействии пользователя с Paradox
isFromUI	MenuEvent	сообщает создается ли событие MenuEvent при интерактивном взаимодействии пользователя с Paradox
isFromUI	MouseEvent	сообщает создается ли событие мыши при взаимодействии пользователя с Paradox
isIgnoreCaseInLocate	Session	сообщает игнорируется ли в текущем сеансе регистр символов при операциях поиска
isIgnoreCaseStringCompares	String	проверяет учитывается ли регистр символов при сравнении строк
isInside	MouseEvent	сообщает находится ли курсор мыши внутри границ объекта-адресата
isLastMouseClickedValid	UiObject	проверяет правильность объекта на котором была последний раз нажата кнопка мыши
isLastMouseRightClickedValid	UiObject	проверяет правильность объекта на котором была последний раз нажата правая кнопка мыши
isLeapYear	DateTime	сообщает содержится ли в году 366 дней
isLeft	Point	сообщает находится ли точка слева от другой точки
isLeftDown	MouseEvent	сообщалось удерживалась ли нажатой левая (или основная) кнопка мыши во время события типа VouseEvent
isMaximized	Form	сообщает выводится ли окно на экран в максимальном размере
isMiddleDown	MouseEvent	сообщает удерживлась ли в нажатом положении средняя кнопка мыши во время события типа MouseEvent

isMinimized	Form	сообщает выводится ли окно в виде пиктограммы
isOnSQLServer	TCursor	выясняет связан ли TCursor с таблицей на SQL-сервере
isOpenOnUniqueIndex	TCursor	выясняет связан ли TCursor с уникальным индексом
isPreFilter	Event	сообщает будет ли событие обрабатываться самой формой
isRecordDeleted	TCursor	проверяет была ли удалена текущая запись (только Для таблиц dBASE)
isRecordDeleted	UiObject	проверяет была ли удалена текущая запись (только Для таблиц dBASE)
isRemote	FileSystem	сообщает является ли данный диск удаленным (сетевым) диском
isRemovable	FileSystem	сообщает является ли данный диск удаленным
isResizeable	Array	сообщает можно ли изменить длину массива
isRight	Point	сообщает находится ли точка справа от другой точки
isRightDown	MouseEvent	сообщает кдерживалась ли нажатой правая кнопка мыши во время события типа MouseEvent
isShared	Table	проверяет является ли таблица в настоящий момент доступной для совместного использования
isShared	TCursor	проверяет является ли таблица в настоящий момент доступной для совместного использования
isShiftKeyDown	KeyEvent	сообщает была ли нажата клавиша SHIFT во время события клавиатуры
isShiftKeyDown	MouseEvent	сообщает была ли нажата клавиша SHIFT во время события типа MouseEvent
isShowDeletedOn	TCursor	проверяет показываются ли удаленные записи в таблице dBASE
isSpace	String	проверяет содержит ли строка только пробелы (или является пустой строкой)
isSpeedBarShowing	Form	сообщает является ли видимоф панель управления
isTable	Database	сообщает содержится ли указанная таблица в базе данных
isTable	Table	проверяет существование таблицы в базе данных
isTargetSelf	Event	сообщает является ли объектадресатом события
isValid	TCursor	проверяет является ли содержимое поля допустимым и полным
isVisible	Form	сообщает выведена ли какая-либо часть окна на экран
keyChar	Form	передает событие методу формы keyChar
keyChar	UiObject	передает событие методу объекта keyChar
keyNameToChr	String	возвращает односимвольную строку представленную строкой виртуального ключевого кода

keyNameToVKCode	String	возвращает числовой код Windows (VK_Code) строки виртуального ключевого кода
keyPhysical	Form	посылает событие методу формы keyPhysical
keyPhysical	UiObject	посылает событие встроенномы методу объекта keyPhysical
killTimer	UiObject	останавливает таймер объекта
ln	Number	возвращает натуральный логарифм числового выражения
load	Form	открывает форму в окне конструктора
load	Report	открывает отчет в окне конструктора
locate	TCursor	осуществляет поиск поля с указанным значением
locate	UiObject	осуществляет поиск указанного значения
locateNext	TCursor	осуществляет поиск поля с указанным значением
locateNext	UiObject	осуществляет поиск указанного значения поля вперед от текущей записи
locateNextPattern	TCursor	осуществляет поиск следующей записи которая включает поле содержащее указанный шаблон символов
locateNextPattern	UiObject	осуществляет поиск следующей записи которая включает поле содержащее указанный шаблон символов
locatePattern	TCursor	осуществляет поиск записи содержащей поле удовлетворяющее указанному шаблону символов
locatePattern	UiObject	осуществляет поиск записи содержащей поле имеющее указанный шаблон символов
locatePrior	TCursor	ищет поле с указанным значением
locatePrior	UiObject	осуществляет поиск указанного значения поля в обратном напрвлении от текущей записи
locatePriorPattern	TCursor	осуществляет поиск предшествующей записи которая включает поле содержащее указанный шаблон или символы
locatePriorPattern	UiObject	осуществляет поиск предшествующей записи содержащей поле которое содержит указанный шаблон символов
lock	Session	накладывает блокировку на одну или несколько таблиц
lock	Table	задает определенную блокировку указанной таблицы
lock	TCursor	задает определенную блокировку указанной таблицы
lockRecord	TCursor	устанавливает условие блокировки изменений текущей записи
lockRecord	UiObject	устанавливает условие блокировки изменений текущей записи
lockStatus	TCursor	возвращает число блокировок наложенных на таблицу

lockStatus	UiObject	возвращает число блокировок наложенных на таблицу
log	Number	возвращает десятичный логарифм числового выражения
logical	Logical	преобразует выражение к типу Logical
longInt	LongInt	приводит значение к типу Longint
loor	Number	округляет числовое выражение до ближайшего снизу целого числа
lower	String	преобразует символы строки в символы нижнего регистра
lTrim	String	удаляет из начала строки пробелы
makeDir	FileSystem	Создает новый каталог
match	String	Сравнивает строку с шаблоном
max	Number	Возвращает максимальное из двух чисел
maximize	Form	Устанавливает максимальный размер окна
memo	Memo	(процедура) - преобразует значение к типу Memo
menuAction	Form	Передает событие методу формы menuAction
menuAction	UiObject	Передает событие встроеному методу объекта menuAction
menuChoice	MenuEvent	Возвращает строку содержащую выбранный пункт меню.
message	System	Выводит сообщение в строку состояния
methodDelete	Form	Удаляет из формы метод определенного для формы
methodDelete	UiObject	Удаляет специальный метод
methodGet	Form	Считывает из формы метод определенного для формы
methodGet	UiObject	Возвращает текст указанного метода
methodSet	Form	Задает описание метода определенного для формы
methodSet	UiObject	Задает текст указанного медода
milliSec	DateTime	Выделяет число миллисекунд из значения типа Date Time
min	Number	Возвращает минимальное из двух чисел
minute	DateTime	Выделяет числоминут из значения типа Date Time
mod	Number	Возвращает остаток при делении двух целых чисел
month	DateTime	Выделяет порядковый номер месяца из значения типа DateTime
mouseClick	UiObject	Генерирует нажатие клавиши мыши и передает это событие объекту
mouseDouble	Form	передает событие методу формы mouseDouble
mouseDouble	UiObject	передает событие встроеному методу объекта mouseDouble
mouseDown	Form	передает событие методу формы mouseDown
mouseDown	UiObject	передает событие встроенному методу объекта mouseDown

mouseEnter	Form	передает событие методу формы mouseEnter
mouseEnter	UiObject	передает событие встроенному методу объекта mouseEnter
mouseExit	Form	передает событие методу формы mouseExit
mouseExit	UiObject	передает событие встроенному методу объекта mouseExit
mouseMove	Form	передает событие методу формы mouseMove
mouseMove	UiObject	передает событие встроенному методу объекта mouseMove
mouseRightDouble	Form	передает событие методу формы mouseRightDouble
mouseRightDouble	UiObject	передает событие встроенному методу объекта mouseRightDouble
mouseRightDown	Form	передает событие методу формы mouseRightDown
mouseRightDown	UiObject	передает событие встроенному методу объекта moseRightDown
mouseRightUp	Form	передает событие методу формы mouseRightUp
mouseRightUp	UiObject	передает событие встроенному методу объекта mouseRightUp
mouseUp	Form	передает событие методу формы mouseUp
mouseUp	UiObject	передает событие встроенному методу объекта mouseUp
moveTo	Form	осуществляет переход на форму
moveTo	UiObject	перемещает фокус ввода на указанный объект
moveToPage	Form	выводит указанную страницу формы
moveToPage	Report	выводит на экран заданную страницу отчета
moveToRecNo	TCursor	перемещает указатель на указанную запись таблицы
moveToRecNo	UiObject	осуществляет переход на указанную запись в таблице dBASE
moveToRecord	TableView	осуществляет переход на заданную запись в таблице
moveToRecord	TCursor	перемещает указатель на указанную запись таблицы
moveToRecord	UiObject	осуществляет переход на указанную запись в таблице
moy	DateTime	выделяет месяц из значения типа DateTime в виде строки
msgAbortRetryIgnore	System	выводит на экран окно диалога содержащее событие и три кнопки - Отмена(Abort) Повтор(Retry) Игнорировать (Ignore)
msgInfo	System	выводит на экран окно диалога содержащее информационную пиктограмму сообщение и кнопку OK

msgQuestion	System	выводит на экран окно диалога содержащее информационную пиктограмму со знаком вопроса и кнопки Да и Нет
msgRetryCancel	System	выводит на экран окно диалога содержащее сообщение и две кнопки Повтор (Retry) и Отмена(Cancel)
msgStop	System	выводит на экран окно диалога содержащее сообщение пиктограмму со знаком останова и кнопку ОК
msgYesNoCancel	System	выводит на экран окно диалога содержащее сообщение и три кнопки Да(Yes) Нет(N0)
name	FileSystem	возвращает имя файла
nextRecord	TCursor	перемещает табличный указатель к следующей записи таблицы
nextRecord	TCursor	перемещает табличный указатель к следующей записи таблицы
nextRecord	UiObject	осуществляет переход на следующую запись таблицы
nFields	Table	возвращает количество полей в таблице
nFields	TCursor	возвращает количество полей в таблице
nKeyFields	Table	возвращает количество полей в первичном или текущем индексе таблицы
nKeyFields	TCursor	возвращает количество полей в текущем индексе таюлицы
nKeyFields	UiObject	возвращает количество ключевых полей в таблице
nRecords	TCursor	возвращает количество записей в таблице
nRecords	UiObject	возвращает количество записей в таблице
numVal	Number	преобразует значение к типу Number
oemCode	String	возвращает код OEM символа
open	Database	открывает базу данных
open	DDE	открывает связь DDE с другими приложениями
open	Form	открывает окно
open	Library	устаанавливает связь типа Library с библиотекой и делает доступными тексты библиотек
open	Report	открывает отчет
open	Session	открывает сеанс
open	TableView	открывает окно таблицы
open	TCursor	открывает таблицу
open	TextStream	открывает текстовый файл в указанном режиме
openAsDialog	Form	открывает окно формы как окно диалога
pixelsToTwips	System	преобразует экранные координаты из пикселов в твипы
pixelsToTwips	UiObject	преобразует экранные координаты из пикселов в твипы
play	System	выполняет уединенную программу
pmt	Number	возвращает сумму рпериодического платежа требующуюся для выплаты
point	Point	преобразует выражение к типу Point

position	TextStream	возвращает позицию указателя в текстовом файле
postAction	Form	ставит действие в очередь действий с отложенным исполнением
postAction	UiObject	заносит действие в очередь действий для последующего выполнения
pow	Number	возводит число в степень
pow10		возводит 10 в заданную степень
priorRecord	TCursor	перемещает табличный уазатель на предыдущую запись в таблице
pritect	Table	осуществляет шифровку таблицы и назначает ей основной пароль
privDir	FileSystem	возвращает имя личного каталога пользователя
pushButton	UiObject	генерирует событие pushButton и посылает его объекту
pv	Number	возвращает текущую стоимость серии равных выплат
qLocate	TCursor	осуществляет поиск указанного значения поля в индексированной таблице
rand	Number	генерирует случайное число в диапазоне от 0 до 1
readChars	TextStream	считывает указанное количество символов из текстового файла
readEnvironmentString	System	считывает элемент операционной среды DOS
readFromClipboard	Graphic	считывает изображение из временного быфера
readFromClipboard	OLE	вставляет объект OLE из буфера в переменную OLE
readFromFile	Binary	данные считываются из файла и записываются в переменную типа Binary
readFromFile	Graphic	считывае изображение из файла
readFromFile	Memo	считывает MEMO поле из файла
readLine	TextStream	считывает строку текстового файла
readProfileString	System	считывает элемент файла настройки
reason	ErrorEvent	сообщает причину возникновения ошибки
reason	Event	сообщает причину события
reason	MenuEvent	сообщает тип выбранного меню
reason	MoveEvent	сообщает причину перемещения
reason	StatusEvent	сообщает причину события типа StatusEvent
recNo	TCursor	возвращает порядковый номер текущей записи
recordStatus	TCursor	проверяет статус записи
recordStatus	UiObject	проверяет статус записи
reIndex	Table	перестраивает указанные индексные файлы
reIndex	TCursor	перестраивает указанные индексные файлы
reIndexAll	Table	обновляет все индексные файлы связанные с таблицей
reIndexAll	TCursor	обновляет все индексные файлы связанные с таблицей

remove	Array	удаляетодин или несколько элементов из массива
remove	Menu	удаляет элемент из меню
removeAlias	Session	удаляет псевдоним из сеанса
removeAllItems	Array	удаляет из массива все элементы с заданными значениями
removeAllPasswords	Session	удаляет все введенные в сеансе пароли
removeItem	DynArray	удаляет указанный элемент из динамического массива
removeItems	Array	удаляет из массива элемент с заданным значением
removeMenu	Menu	удаляет специальное меню и возвращает стандартное
removePassword	Session	удаляет введенный в сеансе пароль
rename	FileSystem	переименовывает файл
rename	Table	переименовывает таблицу
replaceItem	Array	заменяет значение элемента массива на новое
resync	UiObject	вновь синхронизирует объект с табличным указателем
retryPeriod	Session	возвращает длительность периода повторения в секундах операций с заблокированной записью или таблицей
rgb	UiObject	определяет цвет
rollBackTransaction	Database	производит откат (отказ от транзакции). Отменяет все операции проделанные в режиме поддержки сервером транзакций.
rTrim	String	удаляет пробелы с конца строки
run	Form	запускает на исполнение форму которая в данный момент открыта в окне конструктора
run	Report	исполняет отчет итз окна конструктора
save	Form	записывает форму на диск
saveCFG	Session	записывает в файл информацию о псевдонимах используемым в текущем окне
search	String	возвращает позицию одной строки внутри другой
second	DateTime	выделяет число секунд из значения типа DateTime
seqNo	TCursor	возвращает порядковый номер текущей записи
setAliasPassword	Session	вводит запоминайщийся в памяти пароль для указанного псевдонима
setAliasProperty	Session	устанавливает значение указанного свойства для заданного псевдонима
setAltKeyDown	KeyEvent	имитирует нажатие и удерживание клавиши ALT во время события клавиатуры
setChar	KeyEvent	ставит в соответствие событию клавиатуры символ ANSI
setControlKeyDown	KeyEvent	имитирует нажатие и удерживание клавиши CTRL во время события клавиатуры

setControlKeyDown	MouseEvent	имитирует нажатие и удерживание клавиши CTRL во время события типа MouseEvent
setData	MenuEvent	задает информацию о событии типа MenuEvent
setDir	FileSystem	задает путь к каталогу для переменной типа FileSystem
setDrive	FileSystem	делает указанный диск рабочим
setErrorCode	Event	устанавливает для события код ошибки
setFieldValue	TCursor	присваивает значение указанному полю
setFilter	Table	указывает диапазон включаемых записей
setFilter	TCursor	указывает диапазон записей доступных для табличного указателя
setFilter	UiObject	устанавливает диапазон записей табличного объекта которые могут быть включены в рассмотрение
setFlyAwayControl	TCursor	управляет привязкой табличного указателя к записи позиция которой меняется в результате использования метода unlockRecord
setId	ActionEvent	задает идентификационный номер события ActionEvent
setId	MenuEvent	задает идентификатор события меню
setIndex	Table	задает индекс для таблицы
setInside	MouseEvent	помещает указатель мыши внутрь текущего объекта
setItem	DDE	задает элемент для диалога DDE
setLeftDown	MouseEvent	имитирует нажатие левой кнопки мыши
setMenuChoiceAttribute	Menu	устанавливает экранные атрибуты для элемента меню
setMenuChoiceAttributeByld	Menu	устанавливает экранные атрибуты для элемента меню
setMiddleDown	MouseEvent	имитирует нажатие средней кнопки мыши
setMousePosition	MouseEvent	задает для события позицию мыши
setMouseShape	System	задает форму курсора мыши
setPosition	Form	размещает окно на экране
setPosition	TextStream	задает позицию указателя в текстовом файле
setPosition	UiOblect	задает позицию объекта
setProperty	UiOblect	присваивает свойству указанное значение
setReason	ErrorEvent	задает причину возникновения события ErrorEvent
setReason	Event	устанавливает для события константу причины
setReason	MenuEvent	задает константу причины при создании события меню
setReason	MoveEvent	определяет причину возникновения события MoveEvent
setReason	StatusEvent	задает причину для возникновения события типа StatusEvent
setRetryPeriod	Session	задает длительность периода повторения в секундах действий с заблокированной записью или таблицей
setRightDown	MouseEvent	имитирует нажатие правой кнопки мыши

setShiftKeyDown	MouseEvent	имитирует нажадие и удержание клавиши SHIFT
setSize	Array	задается длина массива
setStatusValue	StatusEvent	задает текст выводящийся в строку состояния
setTimer	UiObject	запускает таймер объекта
setTitle	Form	задается текст заголовка окна
setVChar	KeyEvent	ставит в соответствие событию клавиатуры виртуальный символ Windows
setVCharCode	KeyEvent	ставит в соответствие событию клавиатуры виртуальный символ Windows
setX	MouseEvent	задается горизонтальная координата позиции курсора мыши (относительно верхнего левого угла текущего объекта)
setX	Point	задает X-координату точки
setXY	Point	задаюся X и У кординаты точки
setY	MouseEvent	задается вертикальная координата курсора мыши
setY	Point	задает У-координату точки
sgrl	Number	возвращает квадратный корень числа
show	Form	возвращает окну первоначальный размер после минимизации; делает скрытую формыу видимой
show	Menu	выводит меню на экран
show	PopUpMenu	выводит всплывающее меню и возвращает выбранный пункт
showDeleted	Table	управляет показом удаленных записей в таблице dBASE
showDeleted	TCursor	управляет показом удаленных записей в таблице dBASE
showSpeedBar	Form	делает видимой панель управления
sin	Number	возвращает синус угла
sinh	Number	возвращает гиперболический синус угла
size	Array	возвращает число элементов в массиве
size	Binary	возвращает количество байтов содержащихся в переменной типа Binary
size	DynArray	возвращает число элементов в динамическом массиве
size	FileSystem	возвращает размер файла
size	String	возвращает длину строки
size	TextStream	возвращает количество символов в текстовом файле
skip	TCursor	осуществляет перемещение вперед или назад на указанное количество записей в таблице
skip	UiObject	осуществляет перемещение вперед или назад на указанное количество записей в таблице
sleep	System	задает паузу определенной длительности
smallInt	SmallInt	преобразует значение к типу SmallInt
sort	Table	сортирует таблицу

sortTo	TCursor	осуществляет сортировку таблицы
sound	System	задает звуковой сигнал заданной частоты и длительности
space	String	создает строку из указанного числа пробелов
splitFullFileName	FileSystem	разделяет полное имя файла на компоненты
startUpDir	FileSystem	возвращает строку содержащую путь к стартовому каталогу пользователя
statusValue	StatusEvent	возвращает текст выводящийся в строку состояния
string	String	преобразует значение к типу String
strVal	String	преобразует значение в строку
substr	String	возвращает фрагмент строки
subtract	Table	удаляет из одной таблицы записи содержащиеся в другой
subtract	TCursor	удаляет из одной таблицы записи содержащиеся в другой
switchIndex	TCursor	указывает другой индекс для просмотра записей таблицы
switchIndex	UiObject	указывает другой индекс для просмотра записей таблицы
sysinfo	System	создает динамический массив содержащий информацию о системе на которой исполняется Paradox
sysinfor	System	создает динамический массив содержащий информацию о системе на которой исполняется Paradox
tableName	TCursor	возвращает имя таблицы связанной с табличным указателем
tableRight	Table	указывает право пользователя выполнять определенные действия с таблицей
tableRight	TCursor	позволяет определить возможность выполнения над таблицей определенных операций
tan	Number	возвращает тангенс угла
tanh	Number	возвращает гиперболический тангенс числа
time		преобразует значение к типу TIME или возвращает текущее время
time	FileSystem	возвращает время и дату последней модификации файла
toANSI	String	преобразует строку символов OEM в символы ANSI
toDay	Date	возвращает текущую дату
toOEM	String	преобразует строку символов ANSI в символы OEM
totalDiskSpace	FileSystem	возвращает емкость диска
tracerClear	System	очищает окно трассировщика ObjectPAL
tracerHigh	System	удаляет с экрана окно трассировщика
tracerOff	System	закрывает окно трассировщика
tracerOn	System	включает окно трассировки

tracerSave	System	записывает в файл содержимое окна трассировки
tracerShow	System	делает окно трассировки видимым
tracerToTop	System	располагает окно трассировки поверх остальных
tracerWight	System	выводит сообщение в окно трассировки
transctionActive	Database	диагностирует наличие активной транзакции для указанной базы данных
twipsToPixels	System	преобразует экранные координаты из твипсов в пикселы
twipsToPixels	UiObject	преобразует экранные координаты из твипсов в пикселы
Type	Table	возвращает имя таблицы
Type	TCursor	возвращает тип таблицы
umber	Number	преобразует значение к типу Number
unAssign	AnyType	устанавливает для переменной неприсвоенное состояние
unAttach	Table	прерывает связь между переменной типа Table и таблицей на диске
unDeleteRecord	TCursor	восстанавливает текущую удаленную запись таблицы dBASE
unDeleteRecord	UiObject	восстанавливает текущую удаленную запись таблицы dBASE
unLock	Session	снимает блокировку с одной или нескольких таблиц
unLock	Table	снимает блокировку с указанной таблицы
unLock	TCursor	снимает указанное условие блокировки с табличного указателя
unLockRecord	TCursor	снимает блокировку с текущей записи
unLockRecord	UiObject	снимает условие блокировки изменений с текущей записи
unProtect	Table	расшифровывает таблицу и удаляет ее основной пароль
upDateRecord	TCursor	при нарушении уникальности ключа обновляет существующую запись данными из новой записи
upper	String	преобразует символы строки в символы верхнего регистра
usesindexes	Table	указывает индексные файлы для использования и обновления с таблицей dBASE
vChar	KeyEvent	возвращает виртуальный символ Windows
vCharCode	KeyEvent	возвращает челочисленное значение виртуального символа Windows
version	System	возвращает номер используемой версии Paradox
view	Array	возвращает содержимое массива в окно диалога
view	DynArray	выводит содержимое динамического массива в окно диалога
view	Record	выводит в окно диалога значение записи
view	UiObject	выводит значение объекта в окно диалога

vkCodeToKeyName	String	преобразует константу виртуального ключевого кода в строку
wait	Form	приостанавливает выполнение метода
wait	TableView	приостанавливает выполнение метода
wasLastClicked	UiObject	проверяет является ли объект последним объектом на котором была нажата кнопка мыши
wasLastRightClicked	UiObject	проверяет является ли объект последним объектом на котором была нажата правая кнопка мыши
windowClientHandle	Form	возвращает дескриптор окна
windowHandle	Form	возвращает дескриптор окна
windowsDir	FileSystem	возвращает путь к каталогу Windows
windowsSystemDir	FileSystem	возвращает путь к системному каталогу Windows
winGetNessageID	System	возвращает идентификатор сообщения Windows
winPostMessage	System	направляет сообщение состеме Windows
writeEnvironmentString	System	устанавливает значение переменной среды в DOS
writeLine	TextStream	записывает строку в текстовый файл
writeProfileString	System	записывает в файл информацию о конфигурации системы
writeQBE	DataBase	записывает в файл образ запроса или строку запросов
writeQBE	Query	записывает образ запроса в указанный файл
writeSQL	SQL	записывает инструкции SQL или строку SQL в файл
writeString	TextStream	записывает строку символов в текстовый файл
writeToClipboard	Graphic	записывает графический файл во временный буфер
writeToClipboard	OLE	копирует переменную OLE во временный буфер
writeToFile	Binary	записывает данные из переменной типа Binary в файл
writeToFile	Graphic	записывает графическое изображение в файл диска
writeToFile	Memo	записывает MEMO-поле в файл
x	MouseEvent	возвращает горизонтальную координату позиции курсора мыши
x	Point	возвращает X-координату точки
y	MouseEvent	возвращает вертикальную координату позиции курсора мыши
y	Point	возвращает У-координату точки
year	DateTime	возвращает число года из переменной типа DateTime

Содержание