**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра «Биомедицинская инженерия»

УТВЕРЖДАЮ
Проректор по учебной работе
(ЮЗГУ) О.Г. Локтионова
«22» октября 2017 г.

# «МЕДИЦИНСКИЕ БАЗЫ ДАННЫХ И ЭКСПЕРТНЫЕ СИСТЕМЫ»

Методические рекомендации по выполнению лабораторных работ для студентов направления подготовки 12.03.04 “Биотехнические системы и технологии” (бакалавр)

Курск 2017

УДК 621.(076.1)

Составители: А.А.Кузьмин

Рецензент:
Доктор технических наук, профессор *А.Ф.Рыбочкин*

Медицинские базы данных и экспертные системы.: методические рекомендации по выполнению лабораторных работ / Юго-Зап. гос. ун-т; сост.: А.А.Кузьмин. - Курск, 2017. 145 с.: ил.88, табл. 24.

Содержат сведения о составе лабораторных работ. Указывается порядок выполнения лабораторных работ, структура отчета, задания.

Предназначены для студентов направления подготовки 12.03.04 дневной и заочной форм обучения

Текст печатается в авторской редакции

Подписано в печать . Формат 60х84 1/16.
Усл.печ.л. Уч.-изд.л. Тираж 100 экз. Заказ. Бесплатно.
Юго-Западный государственный университет.
305040, г.Курск, ул. 50 лет Октября, 94.

## ЛАБОРАТОРНАЯ РАБОТА №1
# ОРГАНИЗАЦИЯ ДАННЫХ В ВИДЕ ТАБЛИЦ

Цель работы: Изучение принципов создания и модификации таблиц БД в среде СУБД.

Краткие теоретические сведения.

Рассмотрим основные принципы работы с таблицами базы данных (в дальнейшем БД) – это создание, занесение и изменение данных.

В Access существует два таких понятия, как *база данных* и свободная *таблица*. База данных (контейнер или файл .DBC) хранит сведения о таблице, ее индексах, отношениях с другими таблицами, триггерах и другую информацию. Свободные таблицы — это просто DBF-файлы, не включенные ни в одну базу данных. Если у Вас есть справочная таблица, используемая несколькими базами данных, храните ее как свободную. Каждая таблица может принадлежать только одной базе данных.

Для более простого понимания, работу с таблицами будем рассматривать на следующем примере: пусть нам необходимо создать базу данных регистратуры, в которой будет иметься три таблицы. Первая – данные о врачах, вторая – данные о пациентах, третья – учет посещения больными врачей.

Данные о врачах – **DoctorTab**

**ID_doc** – уникальный идентификатор врача
**NameDc** – имя доктора
**PatrDc** – отчество доктора
**SurnDc** – фамилия доктора
**SpecDc** – специализация врача
**Exper** – стаж работы (полных лет)
**Status** – совместитель/штатный (Да/Нет)

Данные о пациентах – **PatientTab**

**ID_pat** – уникальный идентификатор пациента
**NamePt** – имя пациента
**PatrPt** – отчество пациента
**SurnPt** – фамилия пациента
**AddressPt** – адрес пациента
**TelephonePt** – телефон пациента
**GroupB** – группа крови
**GroupRisk** – группа риска

Данные о посещении врачей пациентами – **VisitTab**

**ID_doc** – уникальный идентификатор врача

**ID_pat** – уникальные идентификатор пациента

DateVisit **– дата посещения**

**Complaint** – жалоба больного

Создадим базу данных регистратуры – **RegistryBD** для этого проделаем следующие шаги:

1. В меню **File** выберите команду **New**. Перед Вами появится диалоговое окно **New** (рис. 1), выберите в нем **Database** и нажмите кнопку **New file**.

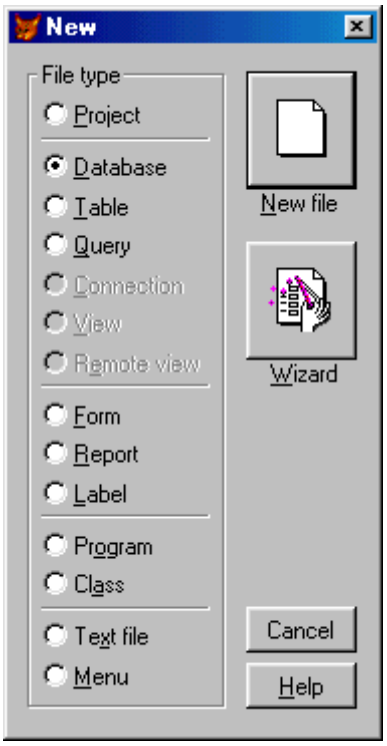


**Рис. 1.** Диалоговое окно **NEW**

После чего, появится диалоговое окно Create (рис. 2). Убедитесь, что Вы находитесь в правильном каталоге. Введите имя БД – RegistryBD и нажмите клавишу *Enter.*

На экране появиться контейнер БД – окно конструктора базы данных. Он пока пустой, но постепенно мы наполним его таблицами, связями между ними и правилами функционирования БД. Перейдем к правилам создания таблиц.

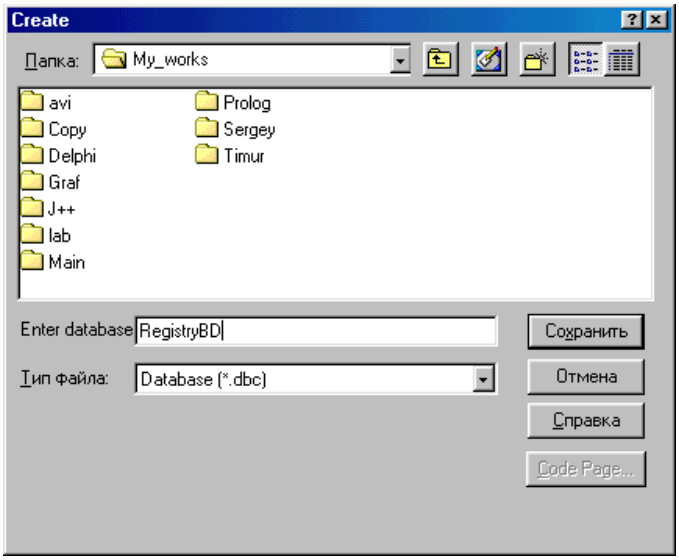


**Рис. 2.** Диалоговое окно **Create**

2. В меню **Database** выберем команду **New Table**, для включения в базу данных новой таблицы. Появится диалоговое окно **New Table**, в котором можно выбрать способ создания таблицы: либо самостоятельно, либо с помощью Мастера создания таблиц (**Table Wizard**). Выберем способ самостоятельного создания – **New Table**. В диалоговом окне **Create** запишем для нашей новой таблицы имя – **DoctorTab**. После этого на экране откроется окно Конструктора таблиц (**Table Designer**). Это окно имеет три вкладки, переключение между которыми выполняется простым щелчком мыши на соответствующей вкладке:

- **Fields** – предназначена для описания атрибутов полей таблицы и правил работы с данными этих полей.
- **Indexes** – предназначена для создания или изменения индексов таблицы.
- **Table** – предназначена для описания правил работы с данными, хранящимися в таблице.

Конструктор таблиц, открытый на вкладке **Fields**, показан на рис. 3.

На вкладке **Fields** для каждого поля создаваемой таблицы мы должны указать:

1. В столбце Name – имя поля. Оно может включать до 128 знаков, представляемых символами алфавита (как латинского, так и любого другого поддерживаемого системой), знака подчеркивания и цифр. Первым символом в имени поля не может быть цифра. Привилегия иметь такие длинные имена полей имеет только таблица, включенная в состав БД.

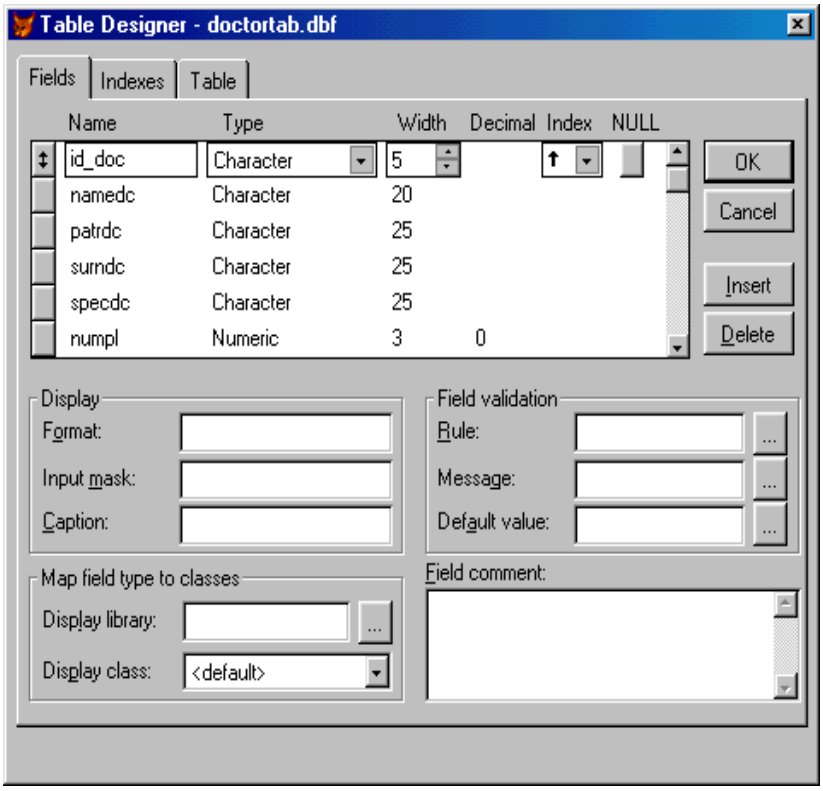


**Рис. 3.** Конструктор таблиц

2. В столбце Type – тип поля. При указании этого атрибута могут использоваться типы данных, приведенных в табл. 1. Для некоторых типов в столбцах Width и Decimal необходимо указать длину поля и число десятичных разрядов. В этом случае в табл. 1 в соответствующих колонках поставлены знаки n и d.

3. В столбце **Index** мы можем указать необходимость создания индекса обычного типа с возрастающей (Ascending) или убывающей (Descending) последовательностью значений. В дальнейшем можно изменить тип индекса и указать другие условия на вкладке **Indexes** Конструктора таблицы.

4. В столбце **NULL** можно отметить допустимое использование значений **NULL** в данном поле. Значения **NULL** позволяют выполнять специфические задачи: определить отсутствие значений в поле; иметь значения, отличные от пустой строки или цифры 0; эти значения располагаются впереди других значений; используются при вычислениях и в большом количестве функций.

Таблица 1 Типы данных полей БД

| Тип данных | Обозначение | Длина | Число десятичных разрядов | Объем | Описание |
|---|---|---|---|---|---|
| Character | C | N | | 1 байт на символ | Символьное выражение может содержать любые символы (до 254 для одного поля). |
| Currency | Y | | | 8 байт | Денежное выражение для числовой величины. Выводит число с четырьмя десятичными разрядами и установленным обозначением используемой денежной единицы. |
| Date | D | | | 8 байт | Выражение для даты может содержать день, месяц и год. |
| DateTime | T | | | 8 байт | Выражение дата и время может содержать время, день, месяц и год. |
| Logical | L | | | 1 байт | Булево выражение .T. или .F. |
| Numeric | N | n | d | 1-20 байт | Числовое выражение может содержать целые или дробные числа со знаком. |
| Integer | I | n | | 4 байта | Целое число. |
| Double | B | | d | 8 байт | Число с плавающей точкой двойной точности. |
| Float | F | n | | 1-20 байт | То же, что и числовое выражение. Оставлено для совместимости. |
| General | G | | | 4 байта | Поле для ссылки на объект OLE. |
| Memo | M | | | 4 байта | Поле примечаний для ссылки на блок данных. |
| Character (binary) | C | n | | 1 байт на символ | Символьное выражение, не подвергаемое трансляции в другую кодовую страницу. |
| Memo (binary) | M | | | 4 байта | Поле примечаний для ссылки на блок, не подвергаемых трансляции в другую кодовую страницу. |

5. Физический порядок расположения полей в таблице можно изменить, используя столбец слева от названия полей. Поля, используемые для индексации, лучше располагать наверху списка.

6. В блоке **Display** указываются атрибуты, связанные с режимом вывода данных поля.

6.1. **Format** (Формат). В текстовом поле **Format** задается формат поля, используемый для отображения значений этого поля на экране.

Например, формат номера телефона может содержать символы и дефисы. Тогда, заменив символы цифрой 9и введя дефисы, будем иметь формат: 99-99-99.

6.2. **Input mask** (Маска ввода). В текстовом поле задается шаблон ввода данных в требуемое поле.

Так, введя один раз дефисы в маску ввода, не придется каждый раз набирать дефисы при вводе телефонного номера по шаблону 99-99-99. Ниже перечислены некоторые специальные символы, используемые для задания шаблонов.

A – допускает ввод только букв;

L – допускает ввод только логических значений T/L;

X – допускает ввод любых символов;

9 – допускает ввод только цифр;

! – преобразует буквы алфавита в прописные и т.д.

6.3. **Caption** (Подпись). Подпись используется вместо имени поля при отображении значений на экране или в отчете.

Заголовок поля **Caption** можно очень эффективно использовать как "дублер" имени поля везде, где требуется указать его назначение пользователю (например, на русском языке).

7. В блоке **Field validation** определяются правила проверки данных при вводе и редактировании.

7.1. **Rule** (Условие на значение). В этом текстовом поле задаются выражения для проверки вводимых значений. При этом возможно использование генератора выражений (кнопка справа).

Например, если мы хотим ограничить наименьшее значение, которое можно ввести поле NumPl, числом 1, то должны указать в Rule выражение: NumPl >= 1. Теперь при попытке ввести в поле NumPl число, меньшее 1, будет выводиться сообщение об ошибке. Правила проверки полей срабатывают всегда, когда пользователь ввел новое значение и переводит курсор на следующий элемент управления, а также в командах, которые добавляют или изменяют значение поля.

7.2. **Message** (Сообщение об ошибке). Вывод на экран сообщения об ошибке при нарушении правила ввода, задаваемого **Rule**. Текст появляющегося сообщения об ошибке можно определить в **Message** (в кавычках). Для правила проверки поля NumPl можно, например, указать: "Номер участка не может быть меньше 1".

7.3. **Default value** (Значение по умолчанию). Значение по умолчанию автоматически появляется в поле перед вводом данных. Пользователь либо оставляет значение по умолчанию, либо изменяет его. В качестве значения по умолчанию также можно использовать функцию, например, DateTime () – добавление текущей даты. Важная сфера применения значений по умолчанию – автоматическая поддержка уникальных кодов. Кнопка справа от каждого атрибута в этом блоке вызывает Построитель выражений, с помощью которого легко разобраться с указанием сложных условий.

8. В блоке **Map field type to classes** можно задать для поля класс элемента управления, с помощью которого будут отображаться данные при работе с формой. Еще более важным обстоятельством является возможность выбора этого класса из библиотеки разработчика. Такая возможность существенно облегчает процесс разработки пользовательского интерфейса.

9. В блоке **Field** comment для полей таблицы записывается комментарий, который может пригодиться при разработке или модернизации приложения.

Кнопки **Insert** и **Delete** предназначены для добавления или удаления полей. После описания всех полей нажмите кнопку ОК.

На вкладке Fields введите все поля приведенные в примере для таблицы **DoctorTab** по следующей схеме:

После чего нажимаем на кнопку ОК – процесс создания таблицы закончен. Самостоятельно попробуйте создать таблицы **PatientTab** и **VisitTab** по следующим схемам (табл.3, табл.4).

В итоге должна получиться база данных, представленная на рис. 4.

Таблица 2 Структура таблицы **DoctorTab**

| Name | Type | Width | Decimal | Index | NULL |
|---|---|---|---|---|---|
| *Id_doc* | Character | 5 | | ↑ | |
| namedc | Character | 20 | | | |
| patrdc | Character | 25 | | | |
| surndc | Character | 25 | | | |
| specdc | Character | 25 | | | |
| exper | Numeric | 3 | 0 | | |
| *status* | Logical | 1 | | | |

Таблица 3 Структура таблицы **PatientTab**

| Name | Type | Width | Decimal | Index | NULL |
|---|---|---|---|---|---|
| *Id_pat* | Character | *5* | | ↑ | |
| namept | Character | 20 | | | |
| patrpt | Character | 25 | | | |
| surnpt | Character | 25 | | | |
| addresspt | Character | 50 | | | |
| telephonept | Character | 8 | | | |

Таблица 4 Структура таблицы **VisitTab**

| Name | Type | Width | Decimal | Index | NULL |
|---|---|---|---|---|---|
| *Id_doc* | Character | 5 | | | |
| *Id_pat* | Character | 5 | | | |
| datevisit | Date | 8 | | | |
| complaint | Memo | 4 | | | |

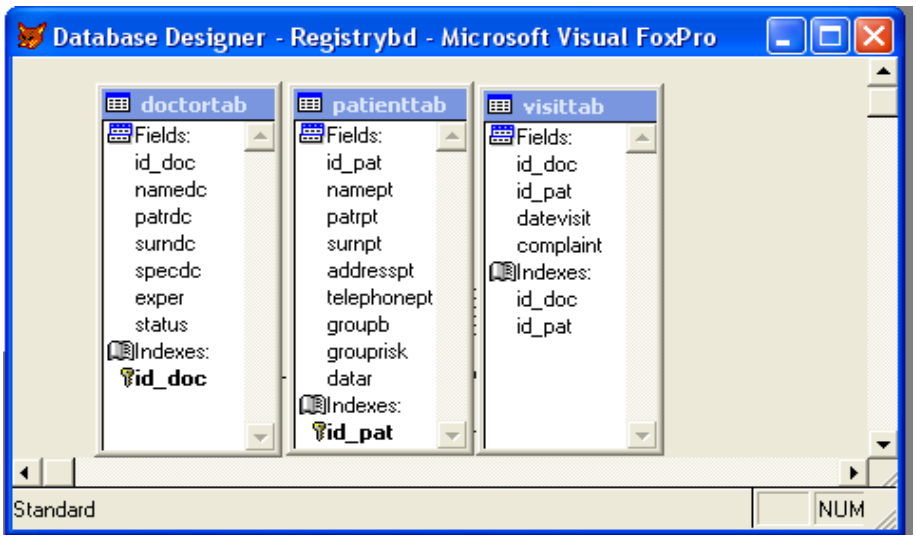


**Рис. 4.** Структура базы данных регистратуры

Порядок работы с вкладкой Indexes будет рассмотрен при изучении темы «Индексация баз данных».

Рассмотрим правила использования вкладки **Table** диалогового окна **Table Designer** (Рис. 5).

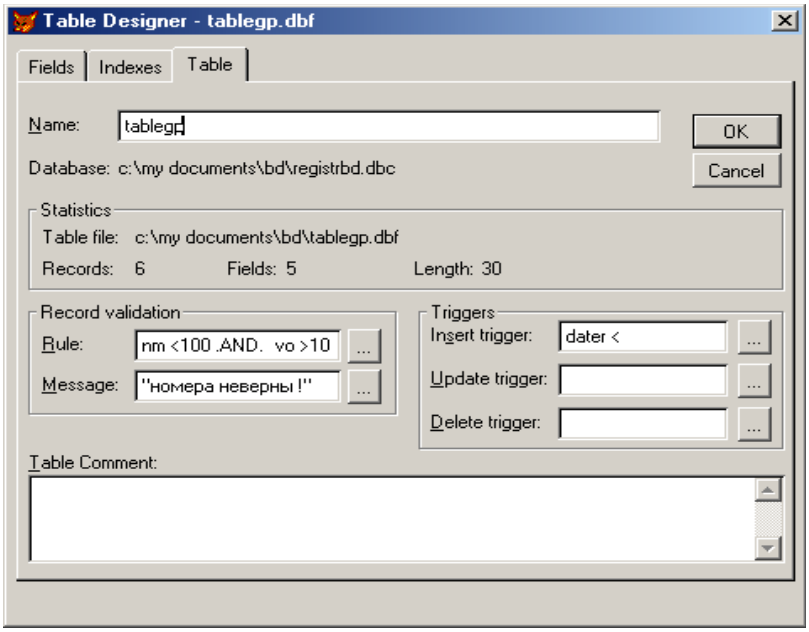


**Рис. 5.** Вкладка Table окна Table Designer

Вкладка **Table** предназначена для задания критериев проверки всей *записи* таблицы, так как проверка значений вводимых *полей* (вкладка **Fields**) может оказаться недостаточной. Для этого используются следующие группы полей:

- **Record validation** (Проверка записи) – определяет условия, при которых Visual FoxPro принимает целую запись, как неверную. При составлении выражений используют имена различных полей табли-

цы (генератор выражений - кнопка справа). Если выражение истинно, выдается сообщение об ошибке.

- **Triggers** (Триггер) – определяет выражение, которое проверяется при появлении *события* (вставки, удаления, изменения, записи). Триггер задается именем соответствующей процедуры в текстовом поле триггера, либо выражением, которое также может включать имена разных полей таблицы.

**Выводы**

В общем случае для определения структуры таблиц некоторой многотабличной БД необходимо:

*1. Создать БД*

1.1. Выбрать команду **File → New**.

1.2. Установить переключатель в положение **Database**.

1.3. Щелкнуть на кнопке **New file** для отображения диалогового окна **Create**.

1.4. В раскрывающемся списке **Save in** окна **Create** выбрать папку для сохранения БД.

1.5. Набрать имя БД и щелкнуть кнопкой **Save**.

*2. Создать таблицы БД*

2.1. Выбрать команду Database → New table.

2.2. В окне New table выбрать кнопку New table.

2.3. В окне Create набрать имя новой таблицы.

2.4. Ввести имена и определить свойства полей и записей таблицы по правилам, описанным в разделе 2.

2.5. Щелкнуть на кнопке ОК окна Table Designer.

2.6. При ответе на вопрос «Хотите вводить данные в таблицу?» щелкните на кнопке No.

2.7. Конструктор БД Database Designer отобразит окно созданной таблицы. Действия 2.1.-2.6 повторить для всех создаваемых таблиц.

Порядок создания таблиц БД с помощью диспетчера проекта

Рассмотрим еще один способ создания базы данных – с помощью проекта. С помощью проекта начинается разработка любого приложения. *Проект* представляет собой совокупность всех файлов, данных, описаний экранов и меню, объектов Visual FoxPro и их классов, информация о которых хранится в файле с расширением PJX.

В Visual FoxPro существует специальное встроенное окно **Project Manager**, отображающее графическое представление рабочего пространства проекта. В дальнейшем будем называть это окно *окном проекта* или *диспетчером проекта*.

Для примера создадим базу данных регистратуры, состоящую из трех таблиц:

Данные о врачах – **DoctorTab**

**ID_doc** – уникальный идентификатор врача
**NameDc** – имя доктора
**PatrDc** – отчество доктора
**SurnDc** – фамилия доктора
**SpecDc** – специализация врача
**Exper** – стаж работы (полных лет)
**Status** – совместитель/штатный (Да/Нет)

Данные о пациентах – **PatientTab**

**ID_pat** – уникальный идентификатор пациента
**NamePt** – имя пациента
**PatrPt** – отчество пациента
**SurnPt** – фамилия пациента
**AddressPt** – адрес пациента
**TelephonePt** – телефон пациента
**GroupB** – группа крови
**GroupRisk** – группа риска

Данные о посещении врачей пациентами – **VisitTab**

**ID_doc** – уникальный идентификатор врача
**ID_pat** – уникальные идентификатор пациента
DateVisit **– дата посещения**
**Complaint** – жалоба больного

Для того чтобы создать новый проект, следует в меню **File** выбрать команду **New**. Перед Вами появится диалоговое окно **New** (рис. 1), выберите в нем **Project** и нажмите кнопку **New file**. Перед Вами появится диалоговое окно **Create** (рис. 6). Убедитесь, что Вы находитесь в правильном каталоге. Введите имя проекта и нажмите клавишу *Enter*.

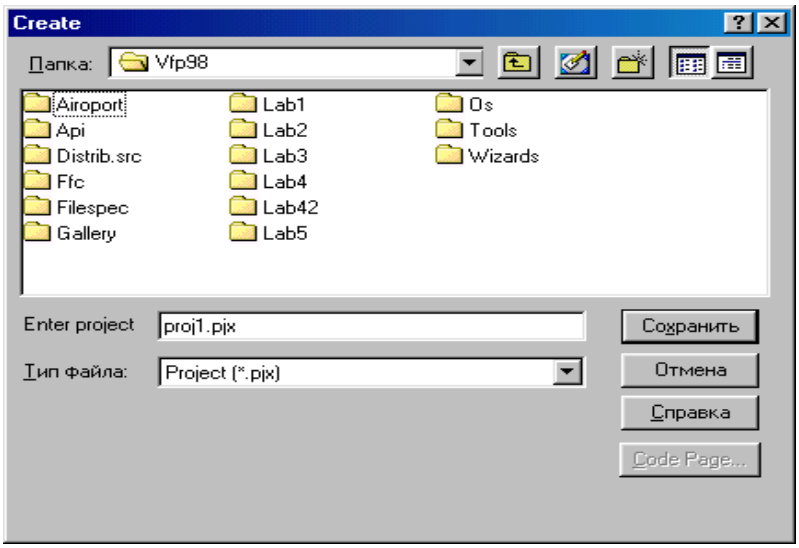


**Рис. 6.** Диалоговое окно **Create**

Перед Вами появится окно *диспетчера проекта* (рис. 7).

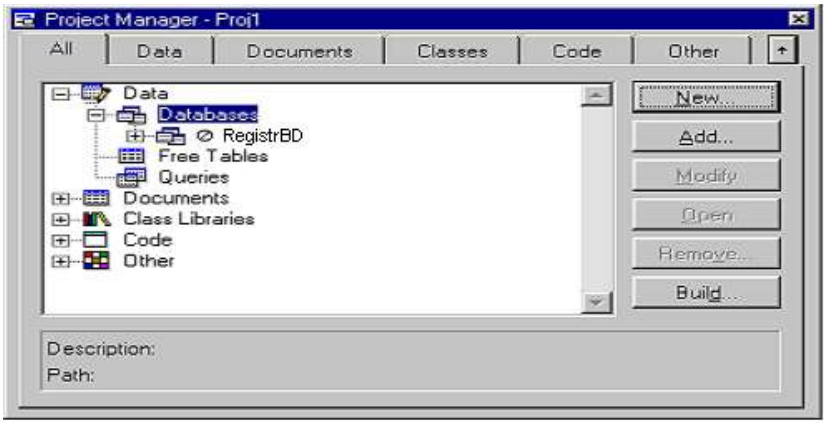


**Рис. 7.** Диспетчер проектов

Окно Диспетчера проектов (рис. 7) — это центр, вокруг которого строится приложение. Именно Диспетчер проектов позволяет Вам разом окинуть все составляющие части приложения. Шесть вкладок окна Диспетчера проектов показывают:

- **All** — все компоненты;
- **Data** — только базы данных, свободные таблицы и запросы;
- **Documents** — только формы, отчеты и этикетки;
- **Classes** - только библиотеки классов и классы;
- **Code** — только программы, библиотеки API и приложения;
- **Other** — меню, текстовые файлы и другие файлы.

Вкладки соответствуют основным группам составляющих проекта. Щелкнув вкладку **All**, Вы увидите все компоненты. Щелчок любой из остальных позволит ограничить обзор определенной группой компонентов.

В верхнем правом углу окна Диспетчера проектов есть кнопка, позволяющая минимизировать или максимизировать окно. Чтобы освободить место на экране, минимизируйте его. Когда оно представлено только панелью с вкладками, Вы можете вызвать одну из групп щелчком соответствующей вкладки.

Вы можете разместить эту панель рядом со стандартной панелью инструментов в верхней части основного окна Visual FoxPro, если окно Диспетчера проектов минимизировано. Просто переместите ее к стандартной панели инструментов и оставьте там. Более того, можно «оторвать» одну из вкладок окна Диспетчера проектов и перемещать ее по экрану, как заблагорассудится. Щелчок кнопки "X" вернет ее на место, и она «приклеится» к основной панели. Если щелкнуть изображение кнопки (той, что прикалывают бумагу), «вырванная» вкладка перейдет в режим «всегда наверху». Чтобы панель инструментов вновь превратилась в окно с вкладками, щелкните в любом месте за пределами вкладок и переместите панель куда-нибудь в середину экрана.

Кнопки с правой стороны окна Диспетчера проектов служат для манипулирования компонентами проекта:

- **New** — для создания нового компонента.
- **Add** — для добавления ранее созданного компонента в проект.
- **Modify** — для редактирования компонента. На редактирование компонент можно вызвать двойным щелчком.
- **Preview** — для предварительного просмотра формата отчетов и этикеток. Вы можете вызвать отчет или этикетки на просмотр без обращения к фактически используемым данным. В зависимости от типов компонентов проекта, с которыми Вы работаете в данный момент, вместо кнопки *Preview* появляются *Run*, *Open* или *Browse.*
- **Remove** — удаляет компонент из проекта.
- **Build** — собирает приложение. Вы можете получить APP или EXE. В основном мы будем использовать формат APP.

Окно Диспетчера проектов можно растягивать по мере увеличения числа компонентов в проекте.

Информация о проекте доступна в любое время по нажатию клавиш Ctrl+J и представлена окном (рис. 8) с тремя вкладками: *Project, Files, Servers.*

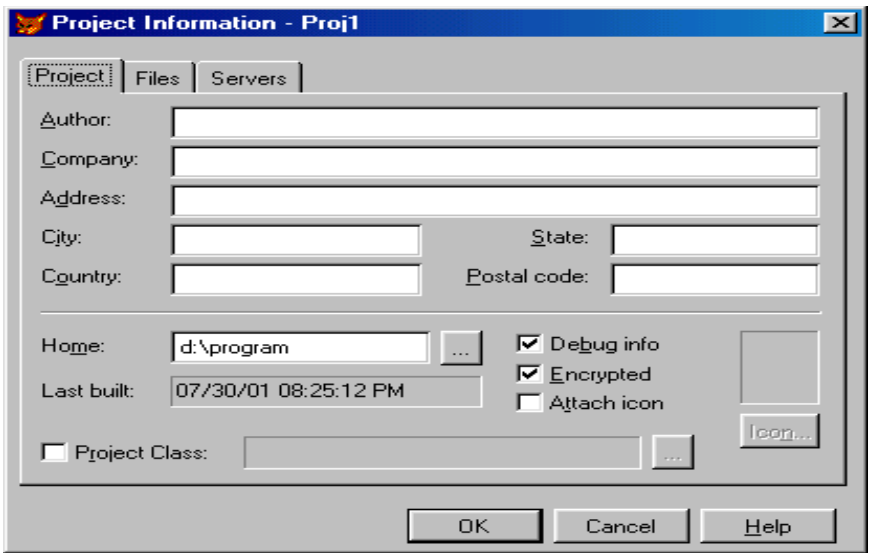


**Рис. 8.** Окно информации о проекте

Щелкнув вкладку **Project**, можно ввести следующую информацию:

- Имя, адрес и другие сведения о разработчике.
- Каталог, в котором размешен проект.
- Нужно ли включать информацию для отладки в собранное приложение. Это немного увеличивает размер файла приложения, но если у Вас нет веских причин делать иначе, эту информацию лучше включить.
- Нужно ли зашифровывать файл приложения. Если APP- или EXE-файл зашифрован, попытка разобрать Ваше приложение на составные части приведет к значительным затратам труда. Лично я никогда этого не делаю — есть более удачные способы обеспечить лояльность заказчика.
- Какой значок использовать для обозначения минимизированного приложения. Тут можно блеснуть талантом художника. Интерес к подобного рода инструментам оказался столь велик, что Visual FoxPro поставляется вместе с редактором изображений.

Вкладка **Files** открывает список всех компонентов проекта. Это исчерпывающий перечень элементов в виде дерева, расположенного в центре основного окна Диспетчера проектов.

Отдельные элементы структуры проекта могут быть показаны полностью или как укрупненные группы. Если слева от значка стоит плюс, то, щелкнув значок, Вы раскроете группу на уровень ниже. Если минус — щелчок сворачивает группу. Кроме того, меню, вызываемое правой клавишей мыши, включает команду **Expand All**, позволяющую раскрыть все группы до самого нижнего уровня.

Для добавления базы данных в проект щелкните вкладку **Data**. Как видите, в эту категорию попадают базы данных, свободные таблицы и запросы (что такое запросы будет рассмотрено позже). Для первого проекта используйте базу данных. Выберите категорию **Databases** (базы данных), щелкните кнопку **New** с правой стороны диспетчера проекта, а затем **New database** и появится диалоговое окно **Create**, предлагающее Вам имя data1.dbc щелкните кнопку **Create**, и перед Вами появится окно Конструктора баз данных.

Создать таблицы в проекте можно:

1. В меню **Database** выбрать команду **New Table**, для включения в базу данных новой таблицы. Этот способ был рассмотрен выше при создании базы данных регистратуры.

2. В окне диспетчера проектов на вкладке **Data** раскрыть список **Databases**, далее созданную базу данных и выбрать пункт **Tables**. С правой стороны диспетчера проекта щелкнуть кнопку **New**, после чего появится диалоговое окно **New Table**, работа с которым была рассмотрена выше.

Следует отметить, что в проект также можно добавлять уже раннее созданные базы данных и таблицы – с помощью команды **Add**.

Вопросы организации связей между структурами таблиц и их отображения рассмотрены далее.

**Выводы**

Для создания проекта и добавления в него таблиц необходимо:

*1. Создать проект*

1.1. Выбрать команду **File → New**.

1.2. Установить переключатель в положение **Project**.

1.3. Щелкнуть на кнопке **New file** для отображения диалогового окна **Create**.

1.4. В раскрывающемся списке **Save in** окна **Create** выбрать папку для сохранения проекта.

1.5. Набрать имя проекта и щелкнуть кнопкой **Save**.

*2. Создать БД*

2.1. Выбрать вкладку **Data**.

2.2. В списке элементов выбрать **Databases**.

2.3. Щелкнуть на кнопке **New** с правой стороны диспетчера проекта для отображения диалогового окна **Create**.

2.4. В раскрывающемся списке **Save in** окна **Create** выбрать папку для сохранения БД.

2.5. Набрать имя БД и щелкнуть кнопкой **Save**.

*3. Создать таблицы БД*

3.1. Выбрать в меню команду **Database → New table** или в окне диспетчера проекта выбрать **Databases → имя БД → Tables** и щелкнуть на кнопке **New** с правой стороны диспетчера проекта.

3.2. В окне **New table** выбрать кнопку **New table**.

3.3. В окне **Create** набрать имя новой таблицы.

3.4. Ввести имена и определить свойства полей и записей таблицы по правилам, описанным в разделе 2.

3.5. Щелкнуть на кнопке ОК окна **Table Designer**.

3.6. При ответе на вопрос «Хотите вводить данные в таблицу?» щелкните на кнопке No.

Конструктор БД **Database Designer** отобразит окно созданной таблицы. Действия 3.1.-3.6 повторить для всех создаваемых таблиц.Порядок заполнения и модификации таблиц

Основным средством редактирования данных в FoxPro являются полноэкранные средства **Browse** и **Edit** (**Change**). **Browse** позволяет редактировать данные в наиболее привычном для пользователя виде – табличном, а **Edit** – в виде колонки полей. Для просмотра данных из таблицы, открытой в какой-либо рабочей области, открывается отдельное окно. Для просмотра или редактирования данных достаточно отыскать нужную таблицу в Диспетчере проектов и выбрать Browse. Это же действие доступно в окне Data Session или в меню View. Вид открывающегося при этом окна Browse показан на **рис. 9**


**Рис. 9.** Окно **Browse**

Этот способ визуализации данных очень удобен, т.к. позволяет просматривать сразу несколько записей, но, как правило, все поля таблицы не умещаются одновременно в окне, даже раскрытом на весь экран. Просмотр данных в виде **Edit** позволяет работать сразу со всеми полями в одной записи, как это видно на рис. 10.

Переключаться между этими двумя режимами просмотра можно с помощью соответствующих команд в меню View. При этом работа с данными не прерывается, а только меняется вид представления информации.

Обратите внимание, что если в таблице описаны заголовки полей, они используются в качестве идентификаторов.

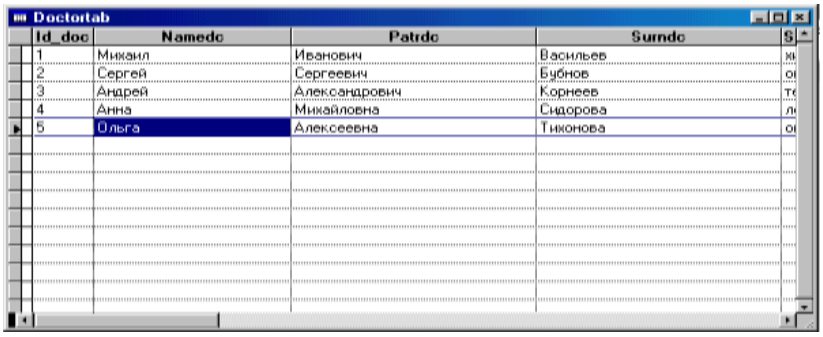


**Рис. 10.** Окно **Edit**

Для ввода и редактирования данных могут использоваться приемы, обычно применяемые при работе в программах для Windows. Используйте возможности, предлагаемые в меню **Edit**. Нажатие на клавишу Tab или Enter приводит к перемещению курсора в следующее поле, а для возврата в предыдущее удобное использовать сочетание клавиш Shift + Tab. Учтите, что при достижении курсором последнего символа в поле во время ввода данных Visual FoxPro подает звуковой сигнал и переводит курсор в следующее поле.

Окна **Browse** или **Edit** являются мощными средствами просмотра и редактирования данных. Дополнительные возможности заложены в меню Table и View. Назначение команд этих меню приведены в таблице 11.

Таблица 11. Средства работы в конструкторе БД

| Команда меню | Назначение |
|---|---|
| Меню Table | |
| Properties | Позволяет установить характеристики для таблицы, открытой в данной рабочей области. При ее выполнении появляется диалоговое окно Work Area Properties. |
| Font | Вызывает появление стандартного диалогового окна Windows, которое позволяет выбрать шрифт и подобрать его характеристики для комфортной работы с данными. |
| Go to Record | Позволяет быстро перейти к нужной записи, выбрав один из следующих пунктов: Top – на первую запись, Bottom – на последнюю запись, Next – на следующую запись, Previous – на предыдущую запись, Record# – на запись с указанным |

| | |
|---|---|
| | номером. Не забудьте, что данные в окне просмотра располагаются в порядке их ввода, если только вы не используете какие-либо индексы. Locate позволяет найти требуемую запись, только по ее содержимому, указав соответствующее выражение для поиска. |
| Append New Record | Позволяет добавить в таблицу одну новую запись. |
| Toggle Deletion Mark | Дает возможность пометить для удаления текущую запись таблицы или убрать эту отметку, если текущая запись была помечена для удаления. |
| Append Records | Позволяет перейти в режим добавления записей, при котором новая будет автоматически добавляться после ввода данных. Добавленная запись будет сохранена, если вы ввели в нее хотя бы один символ. |
| Delete Records | Выводит диалоговое окно, позволяющее указать записи, которые необходимо пометить для удаления. |
| Recall Records | Выводит диалоговое окно, позволяющее указать записи, в которых необходимо убрать пометки для удаления. |
| Remove Deleted Record | Запускает команду PACK для физического удаления помеченных для этого записей. |
| Replace Field | Позволяет указать записи, данные в которых нужно заменить на указанное значение. |
| Size Field | Дает возможность изменить ширину колонки в окне для вывода данных из текущего поля. При этом вы изменяете ширину колонки, а не размер поля в таблице. При достижении нужной ширины нажмите клавишу Enter. Если ширина колонки меньше чем размер поля в таблице, данные будут прокручиваться при перемещении курсора внутри колонки. |
| Move Field | Позволяет переместить текущую колонку в окне. |
| Resize Partitions | Позволяет изменить размеры частей окна просмотра или разбить окно на две части, если этого не было сделано ранее. Разбивка окна позволяет оставить на экране одно или несколько полей при горизонтальном прокручивании данных (см. рис. 6 ). |
| Link Partitions | Позволяет синхронизировать перемещение по записям таблицы. При отмене этой команды в каждой части окна просмотра записи будут перемещаться независимо. |
| Change Partitions | Обеспечивает переход из одной части окна в другую без использования мыши. |

| | |
|---|---|
| Rebuild Indexes | Позволяет привести индексы в соответствие с данными в таблице. |
| Меню View | |
| Browse | Переводит окно в режим таблицы. |
| Edit | Переводит окно в режим колонки. |
| Append Mode | Включает автоматическое добавление новых записей. |
| Database Designer | Вызывает Конструктор БД. |
| Table Designer | Вызывает Конструктор таблиц для изменения структуры открытой таблицы. |
| Grid Lines | Включает или выключает отображение разделительных линий между полями и записями. |
| Toolbars | Вызывает диалоговое окно для выбора панели инструментов. |

Для работы с данными, расположенными в полях примечаний, достаточно два раза щелкнуть мышкой в нужной ячейке окна или переместить туда курсор и нажать клавиши Ctrl+PgDn.

Значительно более широкие возможности полноэкранного редактирования данных могут быть реализованы, если мы воспользуемся командами Browse и Edit (Change) одновременно, (перемещая окно Edit вправо с помощью мыши – Рис. 11.).

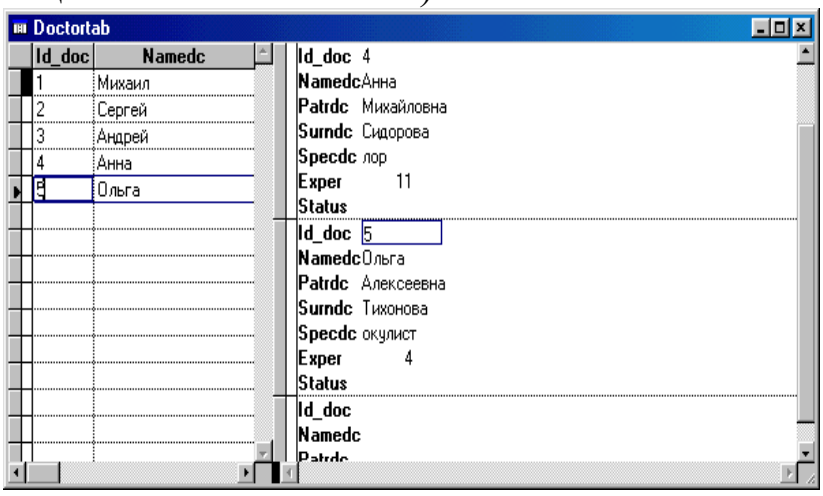


**Рис. 11.** Работа с данными в окне, разбитом на две части (**Browse** и **Edit**)

**Выводы**

Для *отображения* данных модифицируемой таблицы необходимо:

1. Открыть БД в конструкторе **Database Designer**.
2. Дважды щелкнуть на имени заполняемой таблицы БД.

Аналогичный результат (вывод содержимого таблицы в окне Browse) можно получить, выбрав опцию Browse в контекстном меню, появляющемся при нажатии правой кнопкой мыши на выбранной таблице.

При необходимости вертикального расположения столбцов таблицы выполнить команды **View → Edit**.

Для *заполнения* таблиц БД данными выполнить действия:

1. Отобразить данные заполняемой таблицы БД в окне **Browse** (Edit).
2. Выполнить команду **Table → Append New Record** и нажать**<Ctrl+Y>.**
3. Ввести данные, перемещаясь между полями нажатием **<Tab>.**
4. По окончании ввода нажать **<Ctrl+F4>** для закрытия таблицы.

Для *удаления* записей в таблице БД следует:

1. Отобразить данные заполняемой таблицы БД в окне **Browse**.
2. Пометить нужные записи на удаление (на отмену удаления) щелкнув кнопкой мыши в столбце слева от первого поля записи (либо выбрать команду **Table → Delete** для удаления записей либо команду **Records Table → Recall Records** для восстановления записей). В крайнем левом столбце при этом либо появляется, либо исчезает черный маркер.
3. Для удаления (восстановления) записей в окне **Delete** (**Recall**) необходимо:
    3.1. В поле **Scope** установить значение **Next** (Следующая), **All** (Все), **Record** (Текущая) или **Rest** (Остальные, включая текущую). Можно воспользоваться полем справа от **Scope** для точного указания количества записей, следующих за текущей.
    3.2. В текстовых полях **For** и **While** при необходимости ввести критерии удаления (восстановления) записей, используя генератор выражений **Expression Builder**.
    3.3. Щелкнуть кнопкой **Delete** (**Recall**) для перехода к просмотру таблицы.
    3.4. Выполнить команду **Table → Remove Deleted Records** и ответить на вопрос об окончательном удалении помеченных записей.

**Порядок выполнения работы**

1. Выполнить создание многотабличной БД командой **File → New.**
2. Создать таблицы с помощью конструктора таблиц **Table Designer**, используя вкладки **Fields**, **Table** и пропуская поле **Index**.
   2.1. При задании полей установить параметры отображения и проверки значений полей.
   2.2. Установить параметры проверки записей и задать выражения для триггеров.
3. Выполнить тренировочную вставку, удаление и/';[р-0 перемещение полей выбранной таблицы.
4. Заполнить созданные таблицы БД записями (по 5-10 записей).
5. Выполнить просмотр записей таблиц БД в окнах **Browse** и **Edit** .
6. Выполнить тренировочные пометки записей таблицы БД на удаление для разных областей таблицы (поле **Scope** окна **Delete**) и снять пометку удаления записей.

**Контрольные задания**

1. Создать таблицу БД командой File → New → Database. Создать проект, в который добавить уже созданную БД.
2. Создать таблицу БД, содержащую некоторое числовое поле. Ограничить его значение и выводить сообщение об ошибке при несоответствии этого поля заданному интервалу.
3. Создать таблицу БД, содержащую некоторое символьное поле. Разрешить вводить в это поле только цифровые величины.
4. Создать таблицу БД. Каждому полю таблицы присвоить значение, задаваемое по умолчанию при вводе записей в таблицу.
5. Создать таблицу БД, заполнить ее 5-10 записями. Отредактировать записи с помощью окна Browse.
6. Создать таблицу БД, заполнить ее 5-10 записями. Отредактировать записи с помощью окна Edit.
7. Создать таблицу БД, заполнить ее 5-10 записями. Просмотрите записи с помощью окна Browse. Добавьте еще три записи.
8. Создать таблицу БД, заполнить ее 5-10 записями. Просмотрите записи с помощью окна Edit. Добавьте еще три записи.
9. Создать таблицу БД, заполнить ее 5-10 записями. Пометить несколько записей для удаления. Удалить помеченные записи из таблицы.
10. Создать таблицу БД, заполнить ее 5-10 записями. Пометить несколько записей для удаления. Убрать пометки для удаления с записей.

11.Создать таблицу БД, заполнить ее 5-10 записями. Изменить шрифт отображения записей в окнах Browse и Edit.

12.Создать таблицу БД, заполнить ее 5-10 записями. В режиме Browse поменять местами первые две колонки.

13.Создать таблицу БД, заполнить ее 5-10 записями. Удалить один из столбцов таблицы.

14.Создать таблицу БД, заполнить ее 5-10 записями. Переименовать один из столбцов таблицы.

15.Создать таблицу БД, заполнить ее 5-10 записями. Добавить в таблицу еще один столбец.

**Контрольные вопросы**

1. Определить понятия таблицы, записи, поля записи.
2. Какие типы данных используются для задания полей записей?
3. Пояснить типы полей Character, Numeric, Integer, Logical, Memo, Date.
4. Каков порядок создания новой БД?
5. Как создается новая таблица для БД?
6. Определить порядок работы с вкладкой Fields конструктора Table Designer.
7. Каков принцип генерации выражений для проверки вводимых значений полей?
8. Как задаются сообщения об ошибке ввода и значения полей по умолчанию?
9. Пояснить порядок работы с вкладкой Table конструктора Table Designer.
10. Каковы режимы отображения содержимого таблиц БД?
11. Как добавить новую запись в таблицу?

## ЛАБОРАТОРНАЯ РАБОТА №2

# ИНДЕКСИРОВАНИЕ БАЗ ДАННЫХ И ОРГАНИЗАЦИЯ СВЯЗЕЙ В СУБД

Цель работы: Изучение принципов создания и использования индексов и организации структуры связей между таблицами в БД/

Краткие теоретические сведения:

1. Понятие индекса

В реляционной модели данных для связи между таблицами необходимо выделять ключевые поля. Эти поля имеют важное значение для организации быстрого поиска данных. В СУБД информация о расположении данных в ключевых полях хранится в индексах. Чаще всего индексы располагаются в отдельных файлах и тогда говорят о создании индексных файлов. Индексы не влияют на физическое расположение данных в таблицах, а только хранят информацию о порядке расположения этих данных.

В первых версиях СУБД о в одном индексном файле мог храниться только один индекс. Этот файл имеет по умолчанию расширение .IDX. Если требовалось искать данные по разным полям таблицы, приходилось открывать несколько индексных файлов и переключаться между ними. Если при редактировании данных какой-либо из индексных файлов не был открыт, информация в нем переставал соответствовать данным в таблице, и на нас обрушивался град ошибок. В дальнейшем для решения этих проблем в СУБД были включены составные индексные файлы, которые по умолчанию имеют расширение «.CDX» и могут в одном файле содержать несколько индексов. Каждый индекс в таком файле называется тегом (teg) индекса и содержит информацию о каком-либо одном ключе. Это пзволяет открыть сразу несколько тегов, открыв один составной индексный файл. Если составной индексный файл имеет то же самое имя, что и таблица, он называется структурным составным индексом и автоматически открывается при открытии таблицы. Когда составной индексный файл имеет имя, не совпадающее с именем таблицы, он называется независимым составным индексом. В большинстве случаев более предпочтительно использовать структурный составной индекс, т.к. он обновляется автоматически при обновлении данных в таблице и тем самым требует минимальных затрат на обслуживание. Однако, если вы имеет очень большое число индексов, на их обновление могут потребоваться весьма

значительные ресурсы. Выполнение программы может ощутимо замедлиться. В этом случае весьма полезно разделить все ваши индексы на две группы. В первую включить индексы, которые требуются в оперативной работе и участвуют в поиске данных и связях между таблицами, и разместить их в структурном составном индексном файле. Во вторую группу включить индексы, которые используются при составлении отчетов и могут потребоваться всего лишь несколько раз в году, и разместить их в независимом составном индексном файле. Его будет несложно привести в порядок непосредственно перед подготовкой отчета и не тратить на это время во время работы с данными.

## 2. Создание простых и составных индексов

С помощью вкладки Indexes Конструктора таблиц в СУБД можно создать только теги *структурного составного индекса*. Для создания других видов индексов необходимо использовать соответствующие команды. Если к этому моменту вы уже закрыли Конструктор таблиц с таблицей DoctorTab, то найдем ее в Диспетчере проектов. Щелкнем правой кнопкой мыши на заголовке таблицы DoctorTab и в появившемся контекстном меню выберем команду **Modify**. В Конструкторе таблиц перейдем на вкладку **Indexes**, показанную на рис. 1.

Для правильного выбора возможных опций при создании индекса приведем краткий комментарий:

1. В столбце **Name** при выборе имени тега не забудьте, что оно должно начинаться с буквы или знака подчеркивания и не может включать более 10 букв, цифр или знаков подчеркивания. Число создаваемых тегов для таблицы ограничивается только мощностью вашего компьютера, но каждый тег должен иметь уникальное имя.

2. В столбце **Type** мы можем выбрать один из четырех типов индекса. Если таблица включена в базу данных, вы можете создать первичный тег (**primary**). Этот тег в таблице может быть только один, и он не допускает наличия повторяющихся ключевых выражений. Установим первичный тег по полю ID_doc. Теперь мы не сможем для двух разных записей ввести одинаковый код продукта. Если же в этом поле до создания первичного тега уже были повторяющиеся значения, будет обнаружена ошибка, и тег не будет создан, пока мы их не устраним. Оставшиеся три типа индексов можно использовать и для таблиц, не включенных в БД.

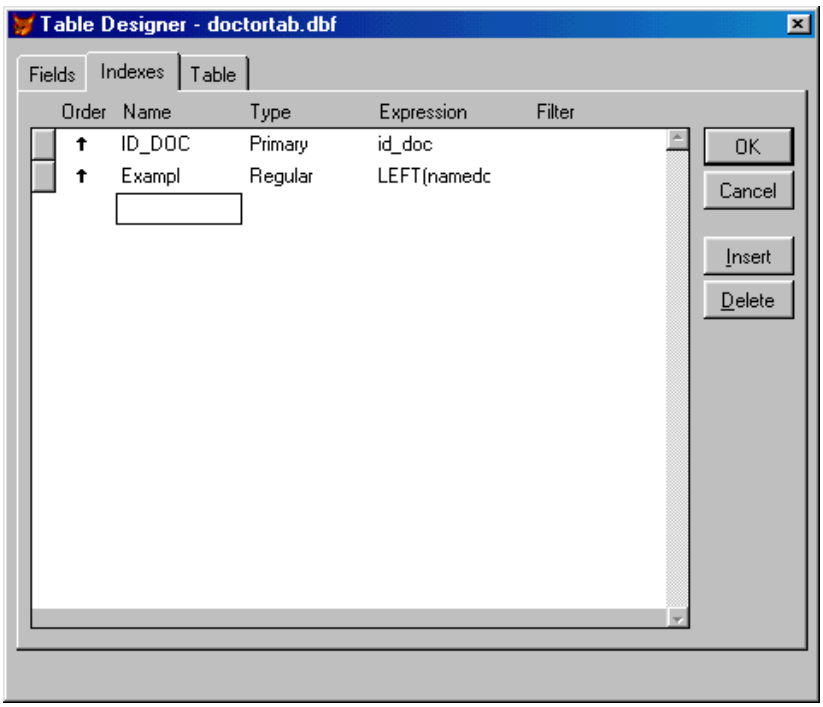


**Рис.1.** Вкладка Indexes Конструктора таблиц

Тег типа кандидат (**candidate**), так же как и первичный, не допускает наличия повторяющихся значений, но тегов этого типа в индексе может быть несколько. Его легко при необходимости преобразовать в первичный.

Уникальный тег (**unique**) обеспечивает вывод только первой записи из возможного множества одинаковых значений индексного выражения. Например, если вы создали уникальный индекс по полю, в котором записаны названия городов, и установили при просмотре данных этот тег уникальным, то вы не увидите городов с одинаковыми названиями. Уникальный тег очень удобно использовать для составления различных списков, например, городов, в которых находятся ваши клиенты. Не путайте этот тип тега с тегом типа кандидат! Тег типа кандидат не допускает дублированных значений, а уникальный тег только «отфильтровывает» такие записи. Придется смириться с некоторой путаницей, которую вносят эти не совсем удачные названия.

Регулярный тег (**regular**) не накладывает никаких ограничений ни на вводимые, ни на выводимые данные. Этот тип тега используется наиболее часто для поиска и управления порядком вывода данных.

3. В столбце **Expression** в качестве выражения для индексного ключа обычно используется имя поля, по которому создается тег. Это наиболее простой случай, и здесь не сложно в соответствующем текстовом поле набрать имя нужного поля. В некоторых случаях нам необходимо ис-

пользовать ключевое выражение, которое будет включать несколько полей. Например, если мы хотим вывести список работников в алфавитном порядке, то при создании тега только по фамилии Андрей Иванов может оказаться позади Сергея Иванова. Чтобы этого не произошло, необходимо использовать составной тег, сформированный по фамилии и имени: SurnDc + NameDc.

Чтобы использовать даты в выражениях индексации примените функцию DTOC(<дата>, 1), преобразующую дату в формат, пригодный для индексирования: YYYYMMDD. Например, DTOC({01/31/95},1) возвращает "19953101", что уже вполне пригодно для создания индекса:

ID_pat + DTOC(DateVisit,1)

Аналогично можно создавать индексные выражения, преобразуя численные величины в текстовые. Приведенная ниже команда создает индекс по идентификаторам врачей и стажу их работы:

ID_doc + STR(Exper,10)

При необходимости для составления выражения можно использовать Построитель выражений, который вызывается при нажатии кнопки справа от текстового поля.

Можно использовать фильтр для того, чтобы в тег попали только некоторые записи, удовлетворяющие записанному критерию. Например, можно создать тег для вывода списка больных только из определенного участка. Для этого необходимо в текстовое поле записать соответствующее логическое выражение или воспользоваться «Построителем выражений», нажав кнопку справа.

Если создаем тег любого типа по символьному полю, то по умолчанию данные располагаются не в алфавитном порядке, а в порядке кодов символов в ASCII-таблице. Если данные записаны на английском языке и в них нет различий в использовании строчных и прописных букв, то последовательность расположения символов будет правильной. В том случае, если используются данные на языке, отличном от английского, возникнут расхождения с требуемой алфавитной последовательностью. За изменение этого порядка отвечает установка SET COLLATE, которая определяет требуемую последовательность символов. Наиболее часто используемые значения для этого параметра:

- GENERAL – используется для большинства западноевропейских языков.

- GERMAN – соответствует стандарту телефонного справочника в Германии.

- MACHINE – эта установка принимается по умолчанию в преды-

дущих версиях FoxPro.

- RUSSIAN — используется для русского языка.

Если для текста, записанного на русском языке, применяется установка MACHINE, то в индексе сначала будут расположены все слова, начинающиеся с прописной буквы, а только затем со строчной. Например: А Б В – Л – а б в – л. При этом, если присутствуют слова с буквой "ё", то в последовательности символов она, в соответствии с ее кодом, встанет впереди буквы "а". При использовании установки RUSSIAN все символы в индексе будут расположены по алфавиту: А а Б б В в.

4. В столбце **Filter** имеется возможность наложить ограничения на записи, которые будут доступны при активизации индекса. Просматривать и редактировать можно будет лишь те записи, которые будут удовлетворять указанному выражению.

5. В столбце **Order** можно установить возрастающий или убывающий порядок расположения данных в индексе.

В дальнейшем с помощью Конструктора таблиц вы легко можете изменить созданные индексы, удалить ненужные и добавить новые.

## 3. Активизация и деактивизация индексов

Создание одного или нескольких индексов является недостаточным для того, чтобы Visual FoxPro их использовала. Для использования в программе нужного индекса необходимо выполнить следующие действия.

1. Открыть БД, содержащую индекс, который нужно активизировать.

2. В окне конструктора БД **Database Designer** щелкнуть дважды в окне нужной таблицы для появления окна Browse.

3. Выбрать команду **Table →Properties**. Появится диалоговое окно **Work Area Properties.** В правом нижнем углу расположен раскрывающийся список **Index order**. По умолчанию в нем выбрано <**no order**>.

4. Из списка **Index order** выбрать индекс для активизации.

5. Щелкнуть кнопкой **ОК** для закрытия окна **Work Area Properties.**

Теперь индекс активизирован. В режиме таблицы **(Browse)** можно видеть записи, отсортированные по ключевым полям индекса.

Для редактирования (удаления) индекса необходимо выполнить следующие действия.

1. В окне конструктора БД **Database Designer** щелкнуть правой кнопкой мыши в окне таблицы, индекс которой редактируется.
2. В контекстном меню таблицы выбрать пункт **Modify**.
3. В окне конструктора таблиц **Table Designer** выбрать вкладку **Indexes.**
4. Изменить атрибуты нужного индекса (при необходимости удаления индекса щелкнуть на кнопке **Delete**).
5. Щелкнуть кнопкой **ОК**.

После этого осуществляется возврат в окно **Database Designer.** Индекс отредактирован (или удален).

## 3. Организация связей

Одним из основных понятий в теории баз данных является понятие отношения.

Таблица состоит из строк и столбцов и имеет уникальное имя в базе данных. База данных содержит множество таблиц, связь между которыми устанавливается с помощью совпадающих полей. В каждой из таблиц содержится информация о каких-либо объектах одного типа (группы).

В качестве примера обратимся к ранее рассмотренной базе данных о врачах и пациентах. Таблица DoctorTab содержит информацию о врачах, таблица PatientTab – информация о пациентах. Связь между ними можно осуществить с помощью таблицы VisistTab (информация о посещении пациентами врачей). Например, вы можете получить информацию о том, кто и когда посещал какого-либо врача. Каждая запись в таблицах идентифицирует один объект группы (врач, пациент и прием). Отношение между объектами определяет отношение между таблицами. Предполагая, что один врач может принять несколько пациентов, а один пациент может посетить несколько врачей. Таким образом, между пациентом и его посещениями врача существует отношение один-ко-многим, такая же связь существует между врачом и его приемами. Связь таблиц осуществляется на основании данных в совпадающих полях ID_doc и ID_pat.

Visual FoxPro поддерживает четыре типа отношений между таблицами: один-к-одному, один-ко-многим, много-к-одному, много-ко-многим. Рассмотрим подробнее каждый из типов отношений.

**Отношение один-к-одному**

Отношение один-к-одному означает, что каждой записи в одной таблице соответствует только одна запись в другой таблице.

В качестве примера рассмотрим отношение между группами PERSON (Личность) и EMPLOYEE (Служащий). Структура этих таблиц представлена в табл. 1 и 2 соответственно.

Обе таблицы содержат информацию о сотрудниках поликлиники. В таблице PERSON содержатся данные о личности сотрудника, а в таблице EMPLOYEE – профессиональные сведения. Между таблицами PERSON и EMPLOYEE существует отношение один-к-одному, поскольку для одного человека может существовать только одна запись, содержащая профессиональные сведения.

Связь между этими таблицами поддерживается при помощи совпадающих полей: ID_Person (таблица PERSON) и ID_Employee (таблица EMPLOYEE). Отметьте, что эти поля имеют разные наименования. Связь между таблицами устанавливается на основании значений совпадающих полей, но не их наименований.

Таблица 1 Структура таблицы **PERSON**

| N | Наименование | Тип | Описание |
|---|---|---|---|
| 1 | ID_Person | Integer | Код личности |
| 2 | LastName | Character(16) | Фамилия |
| 3 | FirstName | Character(16) | Имя |
| 4 | SecondName | Character(16) | Отчество |
| 5 | Phone | Character(13) | Телефон |
| 6 | ZIP | Character(10) | Почтовый индекс |
| 7 | Country | Character(20) | Страна |
| 8 | Region | Character(20) | Область |
| 9 | City | Character(20) | Город |
| 10 | Address | Character(60) | Остальная часть адреса |
| 11 | DateBirth | Date | Дата рождения |
| 12 | Notes | Memo | Примечание |

Таблица 2 Структура таблицы **EMPLOYEE**

| N | Наименование | Тип | Описание |
|---|---|---|---|
| 1 | ID_Employee | Integer | Код служащего |
| 2 | JobTitle | Character(16) | Должность |
| 3 | JobLevel | Integer | Квалификация |
| 4 | Salary | Numeric(12,2) | Зарплата |

| | | | |
|---|---|---|---|
| 5 | StartDate | Date | Дата поступления на работу |
| 6 | LeaveDate | Date | Дата увольнения |
| 7 | Notes | Memo | Примечание |

**Отношение один-ко-многим**

В качестве иллюстрации данного типа отношения обратимся к рассмотренным ранее таблицам DoctorTab и VisitTab. Связь между таблицами осуществляется на основании значений совпадающих полей ID_doc. В данном случае совпадающие поля в обеих таблицах имеют одинаковые наименования.

В качестве других примеров могут быть рассмотрены отношения между предприятием и работающими на нем сотрудниками. Аналогичный тип отношений существует между компьютером и входящими в него компонентами. Как правило, при иерархической организации данных тип отношения один-ко-многим является наиболее общим.

**Отношение много-к-одному**

Отношение много-к-одному аналогично рассмотренному ранее типу один-ко-многим. Тип отношения между объектами зависит от вашей точки зрения. Например, если вы будете рассматривать отношение между приемами и врачами, то получите отношение много-к-одному.

**Отношение много-ко-многим**

Отношение много-ко-многим возникает между двумя таблицами в тех случаях, когда:

- одна запись из первой таблицы может быть связана более чем с одной записью из второй таблицы

- одна запись из второй таблицы может быть связана боле чем с одной записью из первой таблицы

В качестве примера обратимся к такому примеру, пусть у нас есть таблицы: MedTab (информация о лекарствах) и RepTab (информация о рецептах).

Таблица 3 Структура таблицы **MedTab**

| N | Наименование | Тип | Описани |
|---|---|---|---|
| 1 | ID_Pr | Integer | Код производителя |
| 2 | ID_Md | Character(5) | Код товара |
| 3 | Price | Numeric(12.2) | Цена |

Между таблицами MedTab и RepTabсуществует соотношение много-ко-многим, так как на каждый препарат может быть несколько рецептов. Аналогично каждый препарат в рецепте может производиться более чем

одним предприятием. Связь между таблицами устанавливается на основании значений в совпадающих полях ID_Md.

После анализа структуры БД и установления типа отношений между таблицами необходимо установить эти связи в СУБД. Свяжем таблицы DoctorTab, PatientTab с таблицей VisitTab для того, чтобы знать пациентов, посещающих того или иного врача. В таблицах DoctorTab и PatientTab для идентификации врачей и пациентов используются коды, которые имеют уникальные значения и не должны повторяться. Такие же коды используются в таблице VisitTab.

Убедитесь в том, что по полям ID_doc и ID_Pat созданы первичные индексы для таблиц DoctorTab и PatientTab соответственно. Если это не так, создайте эти индексы в Конструкторе таблиц. Если вы посмотрите на таблицы DoctorTab и PatientTab в Конструкторе БД, то увидите, что внизу списка полей прибавилась часть под названием Indexes, и в этой части списка появилось название нового индекса с изображением ключика слева, что свидетельствует о том, что данный индекс является первичным. В таблице VisitTab создадим два регулярных (обычных) индекса по полям ID_doc и ID_pat. Для создания связей между таблицами в Конструкторе БД нажмем кнопку мыши на первичном индексе ID_doc таблицы DoctorTab и, не отпуская ее, переместим указатель мыши на индекс ID_doc таблицы VisitTab. В окне Конструктора БД мы увидим созданную связь визуально. Точно так же надо связать таблицу PatientTab с таблицей VisitTab. Установленные после этого связи хорошо видны на рис. 2. Аналогично можно установить связи в базе данных аэропорта, результат работы представлен на рис. 3.

Для изменения типа отношений между парой таблиц необходимо:

1. Установить курсор мыши на линию, соединяющую таблицы, и жважды нажать курсор мыши;

2. В открывшемся окне **Edit Relationship** установить требуемый тип отношений между родительской и дочерней таблицей;

3. Нажать **ОК** – для сохранеия отношения, либо **Cansel** - для отказа от него.

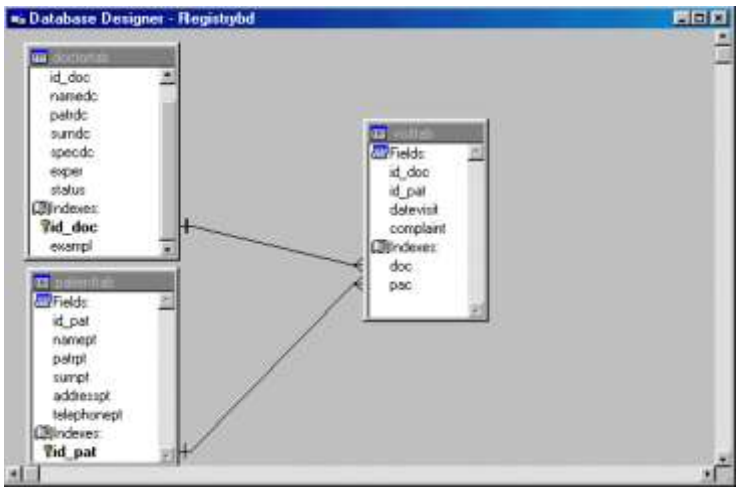


**Рис. 2.** Связанные таблицы в Конструкторе БД

Для удаления установленной связи щелкнете на линии мышкой и нажмите клавишу Del, либо установить курсор на удаляемую линию и нажать правую кнопку мыши; в появившемся контекстном меню выбрать команду **Remove Relationship**.

Изображения таблиц в окне Конструктора БД мы можем легко перемещать, изменять их размеры для наиболее удобной работы или вообще свернуть до размера заголовка. Для выполнения последней операции и ее отмены нам помогут команды **Collapse** и **Expand** в контекстном меню, появляющемся после щелчка правой кнопки мыши.

3. Поддержание целостности данных

Возможность поддержки целостности данных необходима, если наша система включает несколько связанных друг с другом таблиц. Представим простейшую ситуацию, когда в системе начисления заработной платы требуется удалить из справочника работников запись, имеющую отношение к уволенному сотруднику. Тогда во всех остальных таблицах, в которых была ссылка на эту удаленную запись, данные как бы «зависнут в воздухе». При составлении любого отчета они останутся невостребованными, и баланс не сойдется. Очевидно, что удалить запись можно только тогда, когда в системе не будет на нее ссылки из других таблиц. Это осуществляется с помощью специального инструмента - построителем целостности данных (**Referential Integrity Builder**).

Откроем Конструктор БД и в меню Database выберем команду **Referential Integrity**. Появиться окно Построителя целостности данных, приведенное на рис.3.

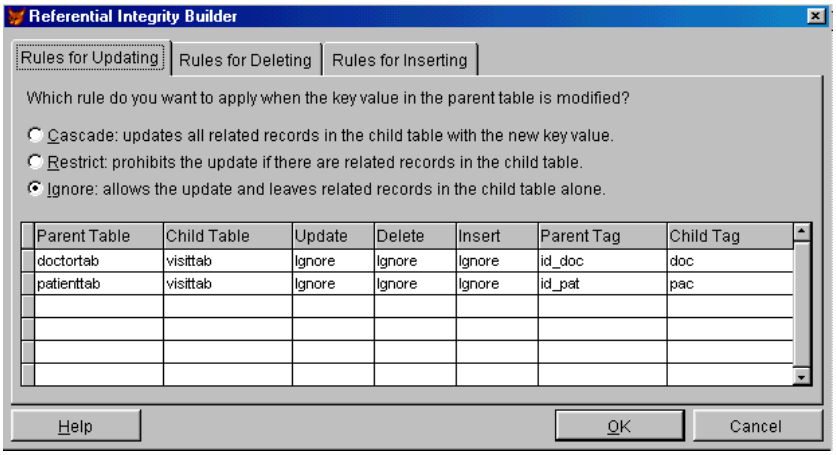


**Рис. 3.** Построитель целостности в БД

С помощью имеющихся в нем средств мы можем установить один из стандартных наборов правил для поддержания целостности нашей системы данных. Для каждого отношения в БД можно выбрать:

2. Для **изменения** записей (**Rules for Updating**):

- Cascade – вначале требуется изменить все соответствующие записи в дочерней таблице, после чего будет записано новое ключевое значение.

- Restrict – запрет на изменение, если есть зависимые записи в дочерней таблице.

- Ignore – разрешить изменение и допустить появление несвязанных дочерних записей.

3. Для **удаления** записей (**Rules for Deleting**):

- Cascade – удаление всех дочерних записей, определяемых по связанному полю, перед удалением записи из родительской таблицы.

- Restrict – запрет на удаление, если есть зависимые записи в дочерней таблице.

- Ignore – разрешить удаление, дочерние записи остаются несвязанными.

4. Для **добавления** записей (**Rules for Inserting**):

- Restrict – запрет на добавление записи, если в ключевом поле родительской таблицы отсутствует подходящая запись.

- Ignore – разрешить добавление.

Сделанный выбор используется для создания кода триггеров, которые размещаются в хранимой процедуре БД. Созданный код можно посмотреть, если на вкладке **Date** раскрыть содержание заголовка **Stored Procedures**.

Триггер – это специальный механизм, существующий в Visual FoxPro, который позволяет реализовать ограничение целостности, имеющие отношение к записям, таблицам или связям между ними. Триггер выпол-

няется в зависимости от выполняемых действий с таблицей. Основная сфера использования триггеров это:

- учет изменений, выполняемых с данными;
- поддержание целостности данных;
- выполнение функционально необходимых действий, связанных с величиной заносимой в таблицу.

**3. Порядок выполнения работы**

1. Создать многотабличную БД и предусмотреть для каждой из таблиц наличие структурного составного индекса.
2. При создании индексов предусмотреть:

- наличие как простых, так и составных индексов;
- наличие индексов для упорядочения записей в хронологическом, числовом и алфавитном порядке;
- наличие фильтров для отбора упорядоченных записей по полям различных типов (numeric, character, date).

3. Выполнить активизацию каждого из созданных индексов и вывести индексированные записи в окно Browse.
4. Установить отношения между таблицами БД, связав соответствующие индексы таблиц.
5. Выполнить тренировочные изменения типа отношений и удаление связей между таблицами.
6. Определить условия целостности БД.

**4. Задания**

1. Создать таблицу БД, содержащую поля символьного, числового вида, а также типа Date. Создайте составной индекс из символьного и числового поля.
2. Создать таблицу БД, содержащую поля символьного, числового вида, а также типа Date. Создайте составной индекс из символьного поля и поля типа Date.
3. Создать таблицу БД, содержащую поля символьного, числового вида, а также типа Date. Создайте составной индекс из числового и поля типа Date.
4. Создать БД из нескольких таблиц. Установить отношения один-к-одному между таблицами.
5. Создать БД из нескольких таблиц. Установить отношения один-ко-многим между таблицами.
6. Создать БД из нескольких таблиц. Установить отношения между таблицами. Удалить установленное отношение.
7. Создать БД из нескольких таблиц. Установить отношения между таблицами. Определите условия ссылочной целостности.

## 5. Контрольные вопросы

1. Пояснить назначение индекса.
2. Чем отличается простой индекс от составного?
3. Как перейти в режим задания индексов для таблиц БД?
4. Пояснить правила именования тегов индекса.
5. Дать характеристику возможных типов индекса.
6. Как задается индексный ключ?
7. Каков порядок задания ограничений на записи в индексе в поле Filter?
8. Каков порядок активизации тега в структурном составном индексе?
9. Как изменить индекс для таблицы?
10. Определить понятие отношение в БД.
11. Какие типы отношений существуют между таблицами в БД?
12. Пояснить понятия родительской и дочерней таблицы.
13. Как создаются связи между таблицами в конструкторе БД?
14. Каков порядок модификации и удаления связей между таблицами?
15. Как определяются правила поддержания целостности БД?

# ЛАБОРАТОРНАЯ РАБОТА №3
# ВВОД И РЕДАКТИРОВАНИЕ ДАННЫХ С ПОМОЩЬЮ МАСТЕРА ФОРМ СУБД

**Цель работы**: Изучение функциональных возможностей мастера форм и автоформ (Form Wizard, AutoForm Wizard) и получение практических навыков создания и использования форм для одной и двух связанных таблиц.

**Краткие теоретические сведения.**

В СУБД для просмотра, ввода и редактирования данных, хранящихся в таблицах, используются формы, являющиеся наглядным средством представления информации.

При создании форм в СУБД разработчик может использовать следующие средства:

**AutoForm Wizard** — мастер автоформы;

**Form Wizard —** мастер форм;

**Form Builder** — построитель формы;

**Builder** — построитель объектов формы;

**Form Designer** — конструктор форм.

Если вы хотите создать простую форму для одной таблицы, воспользуйтесь мастером автоформы. Он это сделает за считанные секунды. Чтобы создать более сложную форму для одной или двух связанных таблиц и обеспечить возможность задания отображаемых в форме полей, их стиля и указания типа кнопок управления, воспользуйтесь мастером создания форм.

Для самостоятельной разработки формы с заданными свойствами или изменения формы, созданной с помощью мастера, вам необходимо использовать конструктор форм. Можно использовать построитель формы для облегчения размещения в конструкторе форм полей и надписей, оформленных в соответствии с выбранным стилем. Помимо этого, в конструкторе форм для большинства объектов (полей, списков, переключателей, таблиц и т. п.) существуют построители, позволяющие размещать в форме заданные объекты и настраивать их свойства.

Мастер автоформы.

Чтобы создать форму, использующую одну таблицу, можно воспользоваться мастером автоформы. Для примера создадим форму для таблицы **DoctorTab**. Установите в конструкторе проекта курсор на таблицу **DoctorTab** и нажмите кнопку **AutoForm Wizard** (Мастер автоформы) на стандартной панели инструментов. Пройдет несколько секунд, и на экране появляется форма, готовая к работе (рис.1).

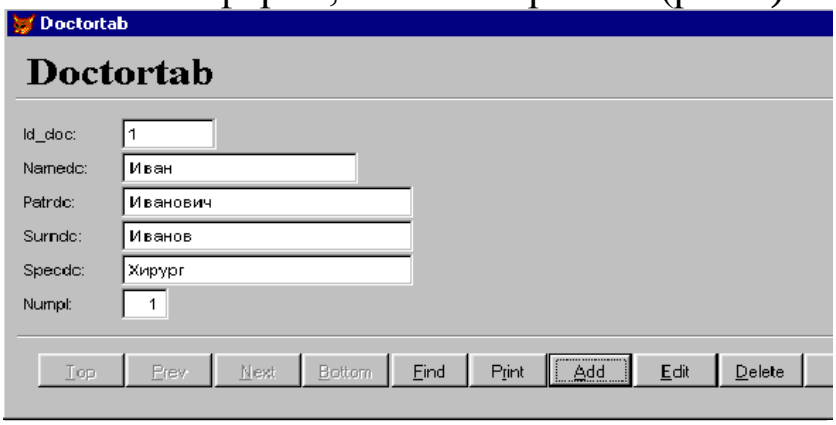


**Рис. 1.** Форма, созданная с помощью мастера автоформы

Рассмотрим создание формы с помощью мастера форм **Form Wizard** (Мастер Форм). В отличие от мастера автоформы он позволяет создавать формы, как для одиночных, так и связанных таблиц, а также настраивать поля, стиль их отображения, тип кнопок управления, размещаемых в форме. В настоящее главе мы ограничимся рассмотрением использования мастера для создания однотабличных форм.

Рассмотрим создание формы с помощью мастера из окна проекта:

Откройте базу данных проекта.

Перейдите на вкладку **Documents** (Альтернативный способ запуска мастера форм - выбор команды **File → New**).

Выберите группу **Forms** (Формы) и нажмите кнопку **New** (Новый) окна конструктора проекта. Откроется диалоговое окно **New Form** (Новая форма).

Нажмите кнопку **Form Wizard** (Мастер формы).

После запуска мастера форм открывается диалоговое окно **Wizard Selection** (Выбор мастера) (рис.2). Значение **Form Wizard** используется для создания однотабличной формы, а **One-to-Many Form Wizard** — для нескольких связанных. По умолчанию установлено первое значение. Нажмите кнопку **ОК,** чтобы запустить мастер создания однотабличной формы.

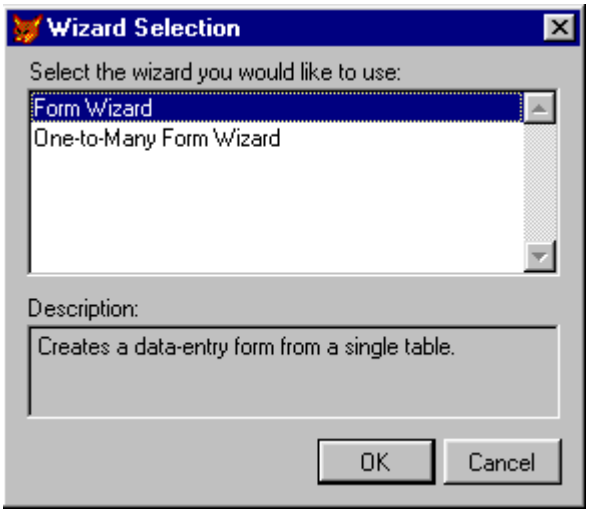


**Рис. 2.** Диалоговое окно для задания типа создаваемой формы: однотабличной или многотабличной БД

Появляется первое диалоговое окно мастера (рис. 3), в котором вам необходимо указать таблицу, для которой вы создаете форму, и выбрать поля этой таблицы, размещаемые в форме. В области **Databases and tables** (Базы данных и таблицы) расположены два списка. Верхний список содержит список открытых баз данных, нижний — список таблиц выбранной базы. Выберите из верхнего списка необходимую базу данных, а из нижнего — таблицу, для которой создаете форму (если открытые БД отсутствуют, следует нажать кнопку справа от кнопки раскрытия БД и в появившемся окне диалога «Open» выбрать нужную таблицу).

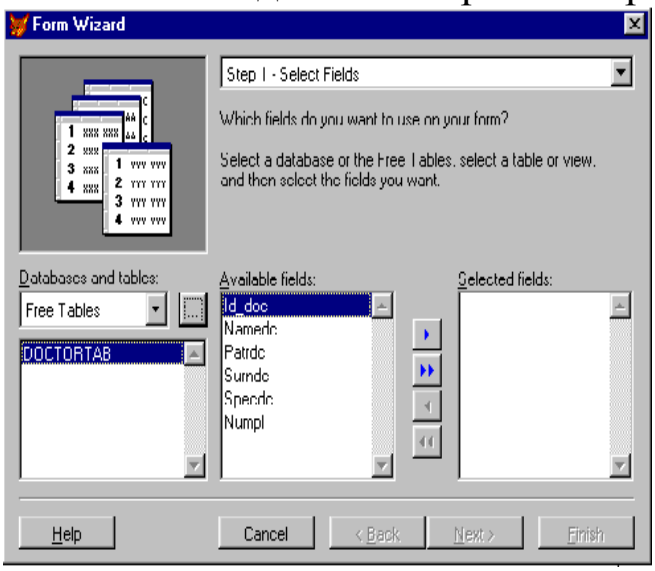


**Рис. 3.** Первый шаг в создании формы с помощью мастера

После выбора таблицы список **Available fildes** (Имеющиеся поля) будет содержать перечень всех полей таблицы. Вам необходимо из данного списка перенести в **Selected fildes** (Выбранные поля) поля, кото-

рые вы хотите разместить в создаваемой форме. Для переноса полей используйте кнопки, расположенные между списками. После создания списка полей, отображаемых в форме, нажмите кнопку **Next** для перехода к следующему шагу.

В появившемся диалоговом окне мастера следует установить стиль отображения объектов формы и типы кнопок управления (рис. 4).

Для объектов формы мастер предлагает на выбор шесть различных вариантов их оформления, которые выбираются из списка **Sty1e** (Стиль). При выборе каждого из стилей вы можете, воспользовавшись областью просмотра в верхней части диалогового окна, просмотреть, как будут выглядеть объекты формы.

Переключатель **Button type** (Тип кнопки) содержит опции, позволяющие задать тип отображения размещаемых в форме кнопок управления (табл.1).

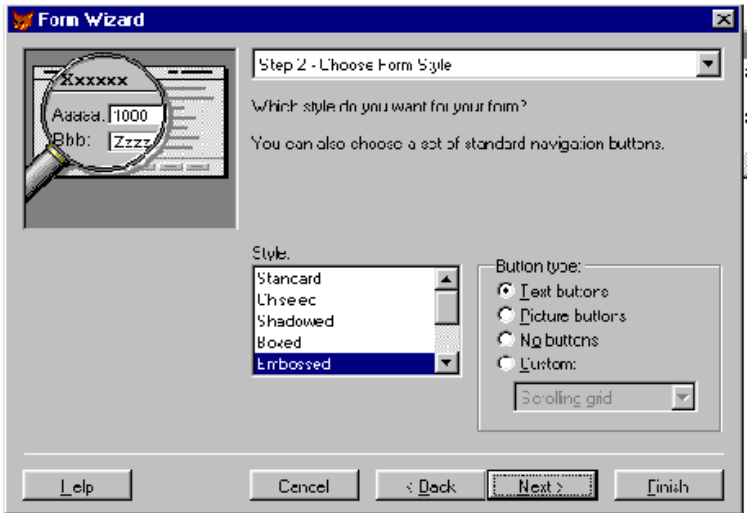


**Рис. 4.** Окно для выбора стиля отображения полей и управляющих кнопок

Таблица 1 Назначение кнопок управления

| **Опция** | Тип отображения |
|---|---|
| **Text buttons** (Текстовые кнопки) | Кнопки управления содержат текстовые надписи |
| **Picture buttons** (Графические кнопки) | Кнопки управления содержат графические изображения |
| **No buttons** (Нет кнопок) | Кнопки управления в форме отсутствуют |
| **Custom** (Другие) | В форме размещается пять кнопок управления: **Find** (Поиск), **Print** (Печать), **Add** (Добавить), **Delete** (Удалить), **Exit** (Выход). Перемещение по записям осуществляется с помощью располагаемой в форме линейки прокрутки |

Установите необходимые опции и нажмите кнопку **Next** (Далее).

На следующем (третьем) шаге задается критерий сортировки данных, отображаемых в форме (рис. 5).

С помощью кнопки **Add** (Добавить) перенесите из списка **Available fields or index** (Имеющиеся поля и индексы) в список **Selected fields** (Выбранные поля) поля, по которым будет осуществляться упорядочение. Если вы ошибочно перенесли не то поле, то его можно удалить из списка **Selected fields** с помощью кнопки **Remove** (Удалить).

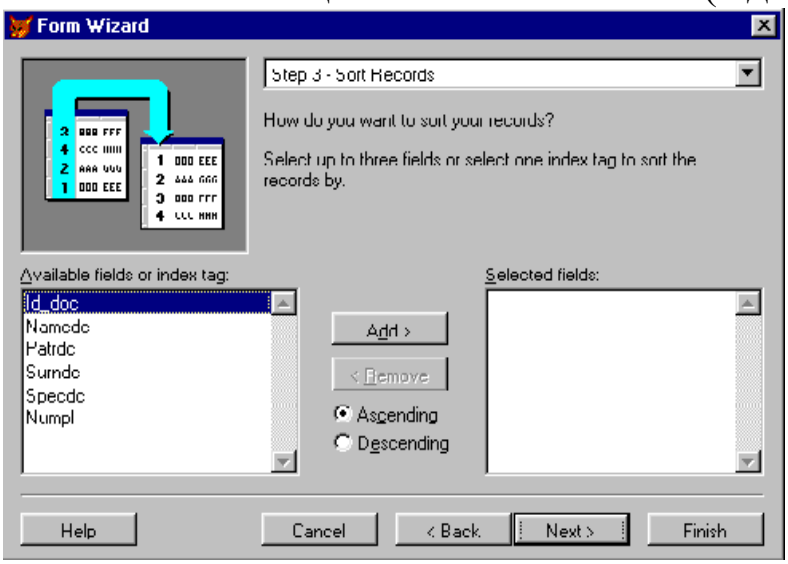


**Рис. 5.** Установка критерия сортировки данных

Опции **Ascending** (По возрастанию) и **Descending** (По убыванию) определяют, по возрастанию или по убыванию будут упорядочиваться данные в указанном поле.

Чтобы перейти к следующему шагу, нажмите кнопку **Next.**

На заключительном шаге создания формы с помощью мастера (рис. 6) вы можете задать заголовок формы в поле Type a title for your form (Тип заголовка формы), а также указать предполагаемые действия с созданной формой, используя следующие опции:

Таблица 2. Назначение опций действий с формой

| *Опция* | Действие |
|---|---|
| **Save form for later use** (Сохранить форму) | Созданная форма сохраняется на диске |
| **Save and run form** (Сохранить и запустить форму на выполнение) | Созданная форма сохраняется и запускается на выполнение |
| **Save form and modify it in the Form Designer** (Сохранить и открыть для модификации в конструкторе форм) | Созданная форма сохраняется и открывается в конструкторе форм для модификации |

В последнем диалоговом окне мастера расположены флажки:

**Use field mappings –** использовать связь полей с типами объектов.

**Add pages for fields that do not fit –** добавить вкладки для не поместившихся полей.

Воспользовавшись кнопкой **Preview** (Просмотр), вы сможете просмотреть, как выглядит создаваемая форма.

После того, как все параметры выделены, нажмите кнопку **Finish** (Готово). Откроется диалоговое окно **Save as**, где укажите имя файла и папку, в которой она должна быть размещена. Для запуска формы выберите в меню **Form** (Форма) созданную вами форму и нажмите кнопку **Run** (Запустить).

Для запуска формы можно также воспользоваться кнопкой **Run** на стандартной панели инструментов Visual FoxPro, либо, выполнив команду **Program → Do** и выбрав в окне диалога **Do** тип **“Form”** для выполнения, нажать **Do.**

На рис. 7 представлена форма, созданная с помощью мастера форм.

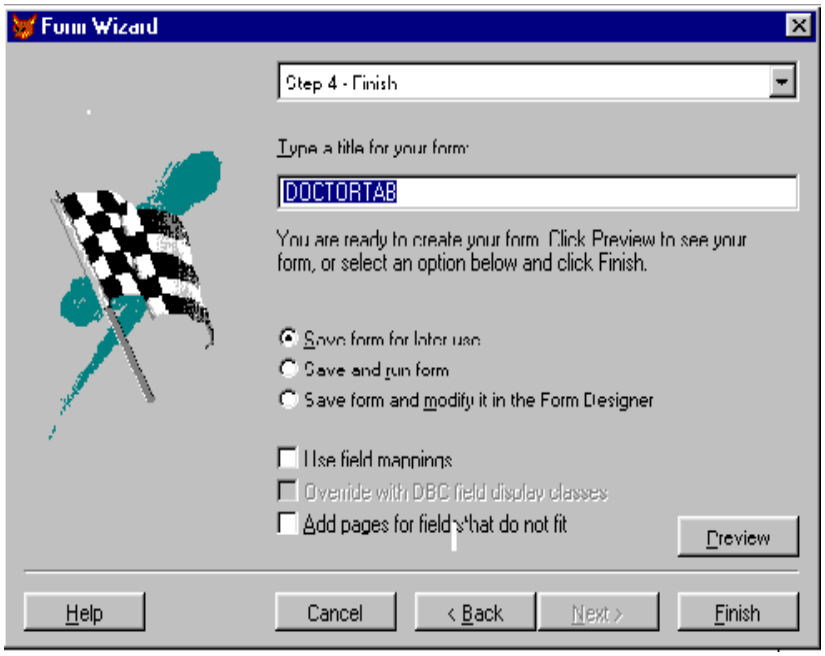


**Рис. 6.** Задание заголовка формы и выбор одного из возможных вариантов продолжения работы с ней

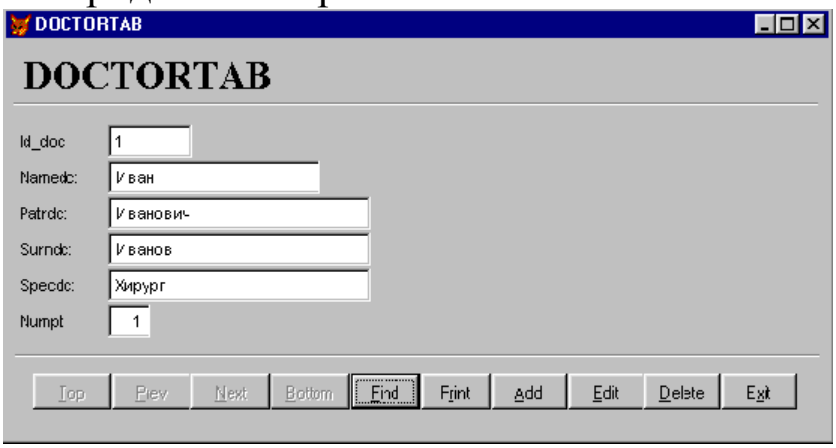


**Рис. 7.** Форма, созданная с помощью мастера форм

Рассмотрим порядок создания формы для пары связанных таблиц **Samolets** и **Model** из базы данных аэропорта:

Откройте базу данных проекта.

Выберите группу **Forms** (Формы) и нажмите кнопку **New** (Новый) окна конструктора проекта. Откроется диалоговое окно **New Form** (Новая форма).

Нажмите кнопку **Form Wizard** (Мастер формы).

После запуска мастера форм открывается диалоговое окно **Wizard Selection** (Выбор мастера) (рис.2). Выберите значение **One-to-Many Form Wizard –** для создании формы для двух связанных таблиц. Нажмите кнопку **ОК,** чтобы запустить мастер создания.

Выберите родительскую таблицу **Model,** а затем поля, которые будут редактироваться в форме (рис. 8).

Выберите дочернюю таблицу **Samolets,** а затем поля, которые будут редактироваться в форме (рис. 9).

Определите связь между таблицами выбором соответствующих полей (рис. 10).

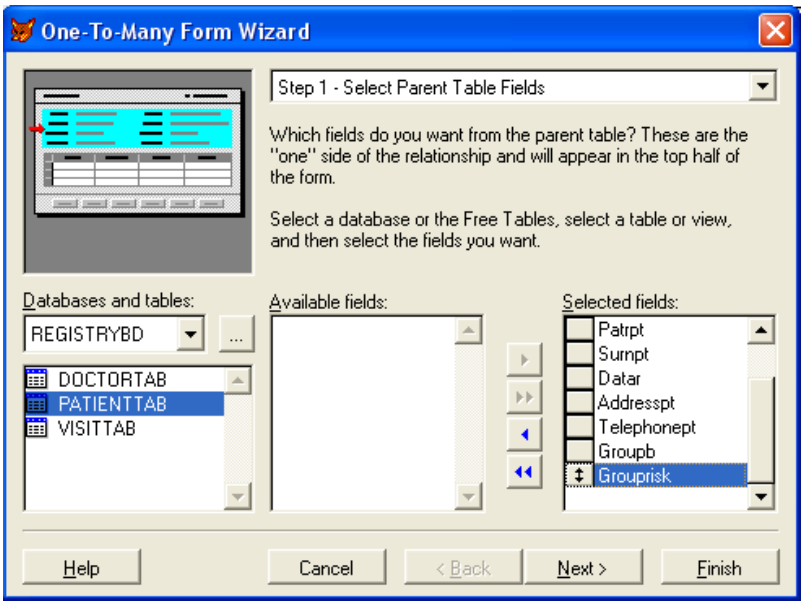


**Рис. 8.** Выбор родительской таблицы

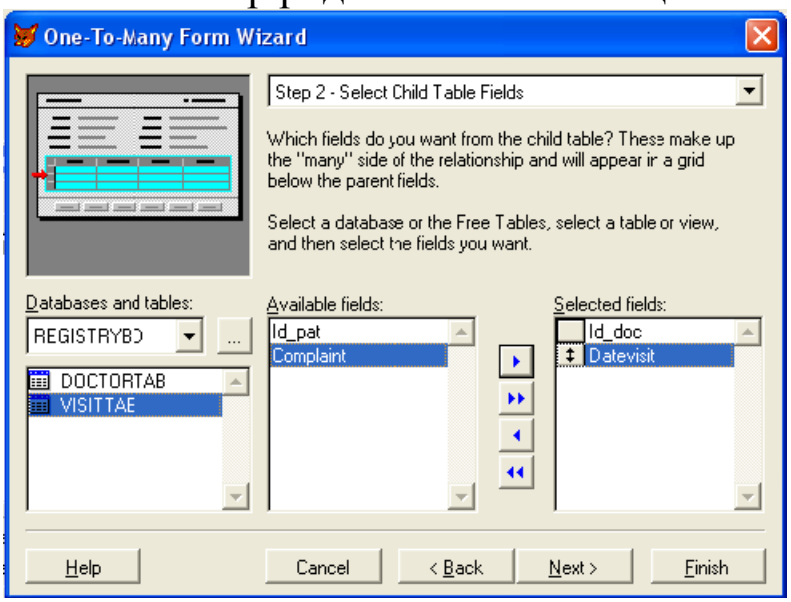


**Рис. 9.** Выбор дочерней таблицы

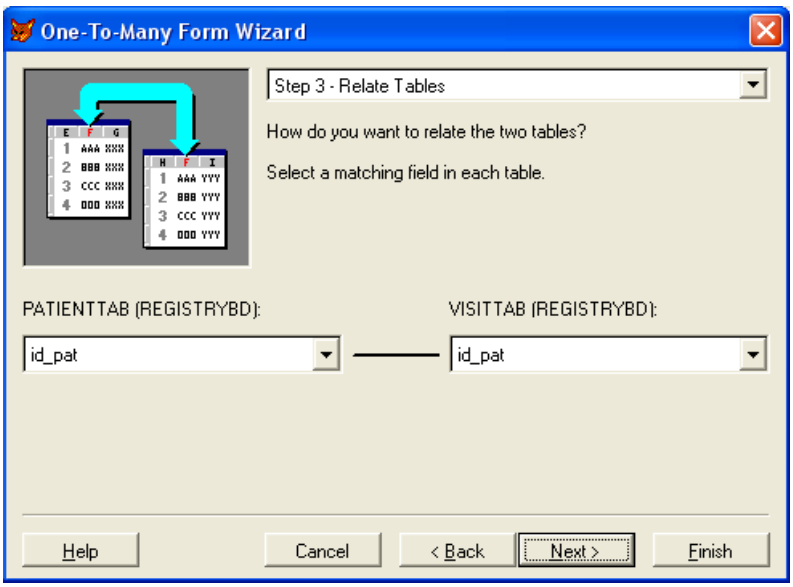


**Рис. 10.** Связывание таблиц

Выберите стиль формы (рис. 4).

Выберите поля, по которому отсортированы данные (рис. 11).

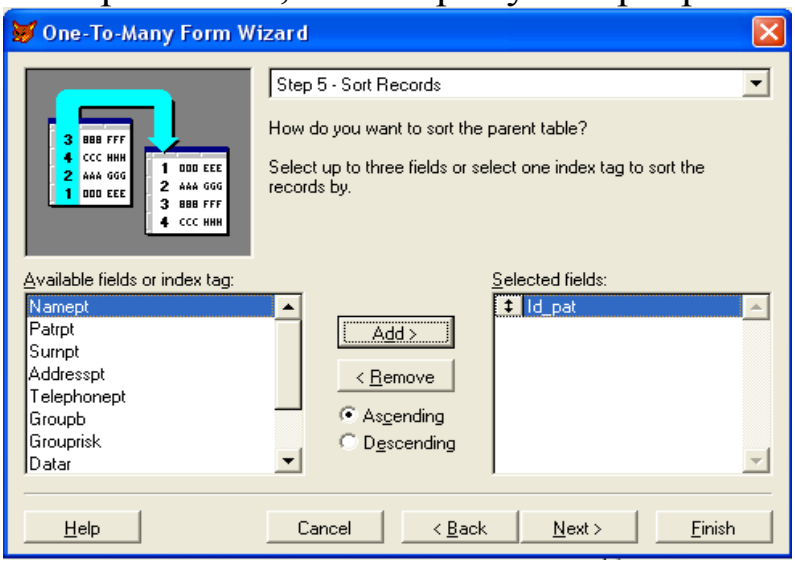


**Рис. 11.** Выбор полей для сортировки данных

Выберите один из вариантов завершения работы: просмотр формы, ее сохранение, модификация или выполнение. Для сохранения созданной формы в файле с расширением SCX открывается стандартное окно **Open**.

Полученная форма изображена на рис. 12.

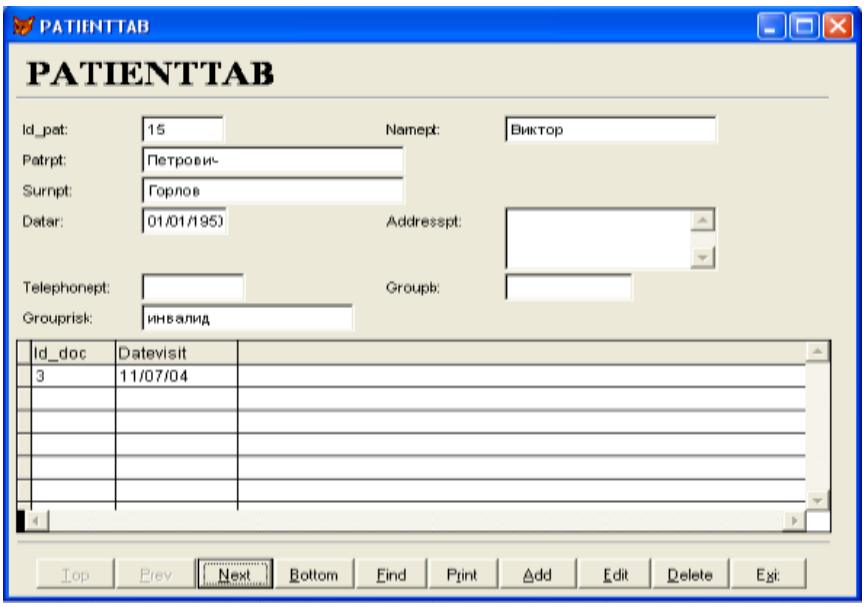


**Рис. 12.** Созданная мастером форма для связанных таблиц

**Порядок выполнения работы**

Создать и сохранить формы с помощью мастера **AutoForm Wizard.**

Запустить созданные формы на выполнение и осуществить просмотр данных и ввод новых записей.

Создать формы ввода для одной и двух связанных таблиц с помощью мастера форм **Form Wizard**. Сохранить формы.

Запустить созданные формы на исполнение. Выполнить коррекцию и добавление данных в таблицы из форм.

Повторить создание формы для двух связанных таблиц, изменив назначения родительской и дочерней таблиц. Сравнить результирующую форму с созданной ранее.

**Контрольные задания**

1. Для таблицы *Chekipag* создайте форму, в которую занесите все поля кроме поля *numsl*.
2. Для таблицы *Chekipag* создайте форму, в которой кнопки управления содержат текстовые надписи.
3. Для таблицы *Chekipag* создайте форму, в которой кнопки управления содержат графические изображения.
4. Для таблицы *Chekipag* создайте форму, в которой перемещение по записям осуществляется с помощью линейки прокрутки.
5. Для таблицы *Chekipag* создайте форму, в которой записи будут отсортированы по возрастанию заработной платы сотрудников.
6. Для таблицы *VisitTab* создайте форму, в которой записи будут отсортированы по дате посещения.
7. Для таблиц *PatientTab* и *VisitTab* создайте форму, в которой

таблица *PatientTab* – родительская, а *VisitTab* – дочерняя. Отсортируйте записи по фамилии пациентов.

8. Для таблиц *PatientTab* и *VisitTab* создайте форму, в которой таблица *PatientTab* – родительская, а *VisitTab* – дочерняя. Отсортируйте записи по идентификатору пациентов.

9. Для таблиц *PatientTab* и *VisitTab* создайте форму, в которой таблица *VisitTab* – родительская, а *PatientTab* – дочерняя. Отсортируйте записи по дате посещения.

10. Для таблиц *DoctorTab* и *VisitTab* создайте форму, в которой таблица *DoctorTab* – родительская, а *VisitTab* – дочерняя. Отсортируйте записи по фамилии, имени, отчеству врачей.

11. Для таблиц *DoctorTab* и *VisitTab* создайте форму, в которой таблица *DoctorTab* – родительская, а *VisitTab* – дочерняя. Отсортируйте записи по стажу работы врачей.

12. Для таблиц *DoctorTab* и *VisitTab* создайте форму, в которой таблица *VisitTab* – родительская, а *DoctorTab* – дочерняя. Отсортируйте записи по идентификатору врачей.

13. Для таблиц *Chekipag* и *Polek* создайте форму, в которой таблица *Chekipag* – родительская, а *Polek* – дочерняя. Отсортируйте записи по фамилии служащих.

14. Для таблиц *Chekipag* и *Polek* создайте форму, в которой таблица *Chekipag* – родительская, а *Polek* – дочерняя. Отсортируйте записи по специальности служащих.

15. Для таблиц *Chekipag* и *Polek* создайте форму, в которой таблица *Polek* – родительская, а *Chekipag* – дочерняя. Отсортируйте записи по дате полета.

**Контрольные вопросы**

1. Пояснить назначение форм и различия между мастером **Auto-Form Wizard** и **Form Wizard.**

2. Каков порядок создания формы с помощью мастера **AutoForm Wizard**?

3. Как выбрать формы мастера для одной и двух связанных таблиц?

4. Перечислить шаги создания формы для одной таблицы с помощью **Form Wizard**.

5. Как с помощью мастера **Form Wizard** изменить состав полей для размещения в форме ввода?

6. Как установить критерий сортировки данных в таблице с помо-

щью мастера форм?

7. Перечислить порядок создания формы для двух связанных таблиц (**One-to-Many**) и пояснить назначение каждого шага.

8. Как изменить назначение родительской и дочерней таблиц в мастере форм?

9. Как меняется окно вывода данных в форме при изменении родительской (дочерней) таблицы?

10. Как выбирается стиль формы в мастере **Form Wizard**?

11. Как можно запустить форму на выполнение?

# ЛАБОРАТОРНАЯ РАБОТА №4
# ПРОЕКТИРОВАНИЕ ФОРМ ВВОДА И РЕДАКТИРОВАНИЯ ДАННЫХ В КОНСТРУКТОРЕ ФОРМ СУБД

**Цель работы**: Изучение принципов создания форм ввода и редактирования данных в конструкторе форм и получение практических навыков проектирования форм для реальных баз данных.

**Краткие теоретические сведения.**

Любая форма в базе данных состоит из объектов, каждый из которых имеет характерные свойства. Для любого объекта указываются действия, выполняемые программой при наступлении определенных событий. Процесс создания формы в конструкторе форм состоит в размещении в форме объектов и определении свойств, а также связанных с ними событий и выполняемых действий.

Для открытия окна конструктора форм при создании новой формы можно воспользоваться одним из следующих способов:

1. Выполните команду **New** (Новый) из меню **File** (Файл). В открывшемся диалоговом окне **New** выберите опцию **Form** (Форма) и нажмите кнопку **New File** (Новый файл).

2. Нажмите кнопку **New** в стандартной панели инструментов Visual FoxPro. В открывшемся диалоговом окне **New** выберите опцию **Form** и нажмите кнопку **New File**.

3. Для размещения создаваемой формы в проекте выберите вкладку **Documents** (Документы), перейдите в группу **Forms** (Формы) и нажмите кнопку **New**.

Также в окне конструктора форм можно редактировать формы, созданные с помощью мастера. Для этого в группе **Forms** (Формы) выберите нужную форму и нажмите кнопку **Modify** (Модифицировать).

Окно конструктора форм содержит следующие панели инструментов: **Co1or Pa1ette** (Цветовая палитра), **Layout** (Расположение), **Form Designer** (Конструктор форм) и **Form Contro1s** (Элементы управления формы), используемые при работе в конструкторе. В окне конструктора размещена форма, с которой вы можете работать.

Процесс создания формы состоит из следующих действий:

- Настройка параметров формы
- Определение среды окружения, т.е. выбор используемых в форме таблиц и установка связей между ними
- Размещение в форме объектов: текста, полей различных типов,

линий, рисунков, кнопок управления

- Настройка свойств размещенных в форме объектов
- Сохранение формы.

Форма, как и все располагаемые в ней объекты, имеет свойства, используя которые можно задать ее размер, координаты верхнего левого угла, стиль рамки обрамления, заголовок, цвет и т. д.

Настройка параметров формы осуществляется в окне свойств **Properties** (Свойства), для открытия которого установите курсор на свободную от объектов поверхность формы и выберите команду **Properties** (Свойства) из меню **View** (Вид).

Новая форма по умолчанию располагается в верхнем левом углу главного окна Visual FoxPro. Для изменения ее положения можно использовать свойства **Left** (Левый) и **Top** (Верхний), указывающие расстояние в пикселях от левого и верхнего края, соответственно, а также мышь. Если вы используете мышь для изменения положения формы, то установите курсор на заголовок формы, нажмите кнопку мыши и, удерживая ее, переместите форму в окне конструктора в место ее предполагаемого расположения.

Для задания текста заголовка, располагающегося в верхней части формы, предназначено свойство **Caption** (Надпись) окна свойств. Чтобы отредактировать заголовок, откройте окно **Properties,** выделите свойство Caption и в поле ввода, ставшее активным, введите заголовок формы.

Если вы хотите, чтобы форма вообще не содержала заголовок, установите для свойства **TitleBar** (Строка заголовка) значение Off**.**

При создании формы, предназначенной для редактирования или просмотра данных таблиц, в конструкторе форм необходимо определить среду окружения, то есть задать таблицы, используемые в форме, и установить связи между ними.

При определении среды окружения выполните следующие действия:

- Добавьте все таблицы, используемые в форме
- Установите индексы для таблиц
- Установите между таблицами отношения, необходимые для создания формы

Вся эта информация, относящаяся к среде окружения, хранится в файле описания формы.

Для создания среды окружения формы предназначено диалоговое окно **Data Environment** (Среда окружения), открыть которое можно одним из следующих способов:

- Выбрать команду **Data Environment** (Среда окружения) из меню **View** (Вид)
- Нажать кнопку **Data Environment** на панели инструментов **Form Designer** (Конструктор форм)
- Выбрать команду контекстного меню формы **Data Environment**

Для работы в окне **Data Environment** можно использовать команды из меню **Data Environment** или контекстное меню (рис. 1), позволяющие добавить в окружение таблицы, просмотреть их в режиме **Browse,** открыть окно свойств окружения для задания различных параметров.

Для добавления новой таблицы в среду окружения можно выполнить одно да следующих действий:

- Выбрать команду контекстного меню **Add**
- Выбрать команду **Add** из меню **Data Environment**

При этом открывается диалоговое окно **Add Table or View** (Добавить таблицу или представление данных), содержащее список таблиц открытой базы данных. Опция **Views** (Представления данных) области **Se1ect** (Выбор) позволяет разместить в среде окружения созданные в базе данных представления данных.

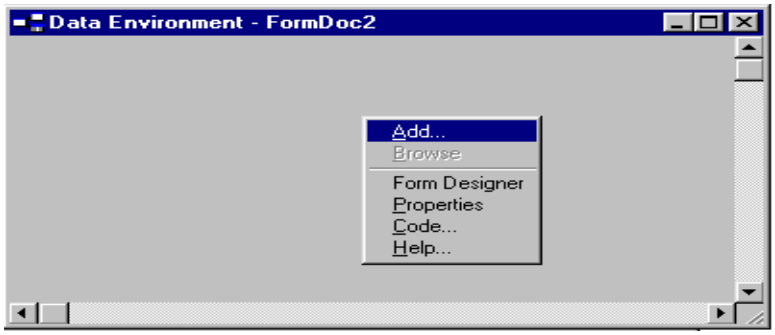


**Рис. 1.** Диалоговое окно **Data Environment** с контекстным меню

После размещения таблиц в среду окружения формы необходимо упорядочить данные, находящиеся в таблицах. Для этого выполните следующие действия:

1. Выделите таблицу, в которой хотите упорядочить данные.
2. Откройте окно свойств таблицы. Для этого установите на нее курсор, нажмите правую кнопку мыши и выберите из контекстного меню команду **Properties.**
3. Выделите свойство **Order** (Упорядочение).
4. В поле коррекции свойства нажмите кнопку раскрытия списка. Из списка индексов таблицы выберите тот, по которому хотите упоря-

дочить данные в форме.

При размещении в форме связанных таблиц они переносятся в окружение формы вместе со связями, установленными в базе данных. Вам необходимо убедиться, что связи установлены должным образом.

На рис. 2 представлено диалоговое окно **Data Environment** с размещенными в нем таблицами DoctorTab, PatientTab и VisitTab.

**Последовательность действий при настройке параметров формы**

Теперь рассмотрим последовательность действий для определения окружения и задания основных свойств формы:

1. Откройте проект RegistryBD (наш пример).
2. Выберите вкладку **Documents** (Документы), перейдите в окне проекта в группу **Forms** и нажмите кнопку **New**.

**Рис. 2.** Диалоговое окно **Data Environment**

3. В открывшемся диалоговом окне **New Form** (Новая форма) выберите опцию **New Form.** Откроется окно конструктора форм для создания новой формы.
4. Откройте окно окружения формы **Data Environment** (Среда окружения) выбрав команду **Data Environment** из меню **View**.
5. Для размещения таблицы в среде окружения выберите команду **Add** меню **Data Environment**.
6. В открывшемся диалоговом окне **Add Table or View** (Добавить таблицу или представление данных) выберите из списка таблиц открытой базы данных таблицу, для которой создаете форму, и нажмите кнопку **ОК.**
7. Откройте окно свойств таблицы, размещенной в окне окружения. Для этого установите на нее курсор, нажмите правую кнопку мыши и выберите из контекстного меню команду **Properties** (Свойства).
8. Выделите свойство **Order** (Упорядочение). Для упорядочения данных в форме в поле коррекции свойства нажмите кнопку раскрытия списка и из списка индексов таблицы выберите индекс, по которому хотите упорядочить данные.
9. Закройте окно определения среды окружения.
10. Для задания свойств формы выберите из меню **View** команду

**Properties**.Открывается окно **Properties**.

11. В окне **Properties** скорректируйте свойство **Caption** (Надпись), введя в текстовом поле заголовок формы.

12. Задайте цвет фона формы. Для этого используйте свойство формы **BackCo1or** (Цвет фона). Щелкните на нем. Затем нажмите кнопку, расположенную справа от поля редактирования свойства, и в открывшемся диалоговом окне **Цвет** выберите цвет, который вы хотите использовать для фона.

13. Свойство **AutoCenter** (Автоцентр) должно иметь значение True, чтобы форма располагалась в центре экрана.

14. Для изменения размера окна установите курсор в нижний правый угол формы. Когда он примет вид двунаправленной стрелки, нажмите кнопку мыши и, удерживая ее, измените размер формы. Установив необходимый размер формы, отпустите кнопку мыши.

15. Измените свойства **FontName** (Наименование шрифта), **FontSize** (Размер шрифта), выбрав подходящий шрифт из установленных на вашем компьютере, и его размер.

16. Сохраните форму на диске.

После того как определены параметры формы, размещены в окружении используемые таблицы, можно приступать к размещению объектов в форме.

Для размещения в форме полей таблицы и надписей к ним в конструкторе можно использовать построитель формы.

Чтобы запустить построитель форм, выберите команду **Builder** (Построитель) контекстного меню формы или нажмите кнопку **Form Builder** (Построитель формы) на панели инструментов Form Designer (Конструктор форм). Откроется диалоговое окно **Form Builder**, содержащее две вкладки:

Таблица 1

| **Вкладка** | **Назначение** |
|---|---|
| Field Selection (Выбор поля) | Выбор полей, которые будут размещены в форме (рис.3) |
| Style (Стиль) | Задание стиля отображения объектов формы (рис.4) |

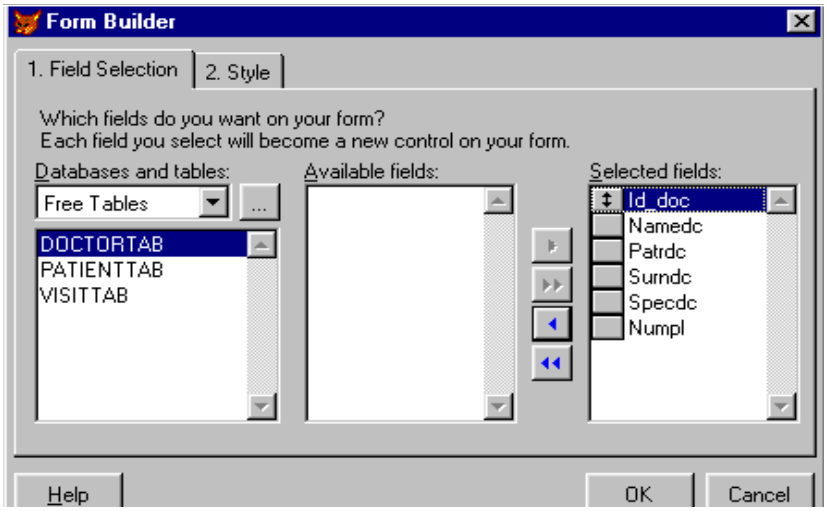


**Рис. 3.** Вкладка для выбора размещаемых в форме полей


**Рис. 4.** Вкладка для определения стиля размещаемых объектов

Используя вкладку **Field Selection** диалогового окна **Form Builder** построителя формы, сформируйте список полей таблицы, размещаемых в форме. Для этого из верхнего списка области **Databases and tables** (Базы данных и таблицы) выберите необходимую базу данных, а из нижнего — таблицу, поля которой размещаете в форме. Затем из списка **Available fields** (Имеющиеся поля) перенесите в **Selected fields** (Выбранные поля) поля, которые вы хотите разместить в создаваемой форме. Кнопки, расположенные между списками, предназначены для переноса полей.

Сформировав список полей, перейдите на вкладку **Sty1e.** Используя расположенный здесь список **Style,** задайте стиль оформления объектов, размещаемых в форме. Завершив установку параметров на обеих вкладках, нажмите кнопку **OK.** В форме будут размещены поля и надписи к ним (рис.5).

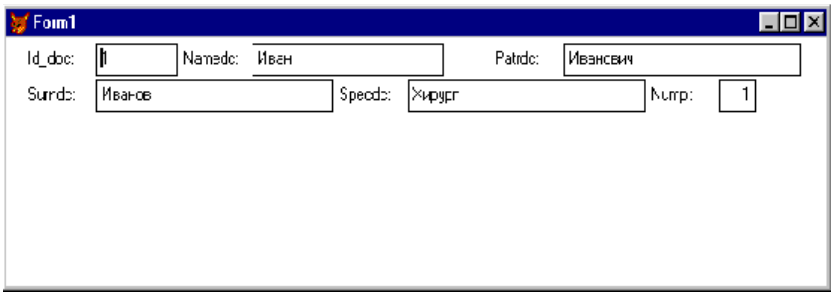


**Рис. 5.** Форма с объектами, размещенными с помощью построителя

Размещение в форме текста (заголовков, надписей к полям, поясняющей информации) осуществляется с помощью кнопки **Label** (Метка), находящейся на панели инструментов Form Controls (Элементы управления формы).

Для размещения текста выполните следующие действия:

1. Нажмите кнопку **Label** на панели инструментов Form Controls. Если данная панель отсутствует на экране, для ее отображения выберите в меню **View** команду **Form Controls Toolbar**.
2. Установите указатель мыши на место предполагаемого расположения текстового объекта и нажмите кнопку мыши. В форме появляется объект, в названии которого содержится слово **Label**.
3. Для открытия окна свойств созданного объекта выделите его и выберите в меню **View** (Вид) команду **Properties** (Свойства). Откроется окно **Properties**.
4. Фон текстового объекта определяется свойством **BackStyle** (Стиль фона). Если вы хотите, чтобы фон текста не отличался от фона формы, установите для свойства **BackStyle** значение Transparent (Прозрачный).
5. Текст задается свойством **Caption**. Выделите данное свойство, после чего в поле ввода свойства введите нужную текстовую информацию и нажмите клавишу <Enter>.
6. Задайте с помощью свойств **FontName** (Наименование шрифта) и **FontSize** (Размер шрифта) вид и размер шрифта.
7. Используя свойство **ForeCo1or**, задайте цвет текстовой информации.
8. Скорректируйте размер объекта, чтобы в нем помещалась вся надпись. Для этого используйте маркеры выделения или установите значение True в поле свойства **AutoSize** (Авторазмер).

Для отображения информации в форме используются поля различных типов. Наиболее простой тип поля — это поле ввода. Для размещения поля ввода в форме выполните следующие действия:

1. Нажмите кнопку **Text Box** (Поле ввода) на панели инструментов Form Controls.

2. Щелкните в том месте формы, в котором вы предполагаете расположить поле ввода.

3. Откройте окно свойств созданного объекта. Для этого выделите его и выполните команду **Properties** из меню **View.**

4. Чтобы связать созданное поле с полем таблицы, выберите на вкладке **Data** свойство **ControlSource** (Источник данных). В поле ввода свойства воспользуйтесь кнопкой раскрытия списка и из списка всех полей открытой таблицы выберите поле, которое хотите добавить в форму (рис. 6).

5. Используя свойство **Alignment** (Выравнивание), задайте вариант выравнивания в поле: по центру, по левому или правому краю поля.

6. Для задания стиля и цвета рамки поля используйте свойства **BorderStyle**(Стиль рамки) и **BorderColor** (Цвет рамки), соответственно.

7. С помощью свойства **DisabledBackColor** задайте цвет фона неактивного поля.

8. Применяя свойство **Comment**, вы можете задать краткое описание назначения размещенного объекта. Это описание будет полезно при разработке приложения и его сопровождении.

9. Используя свойства **FontName** (Наименование шрифта) и **FontSize** (Размер шрифта), задайте используемый при отображении информации шрифт и его размер.

10. С помощью свойства **ForeCo1o**r задайте цвет, которым будет отображаться информация в поле ввода.

11. Для отображения полей ввода в заданном формате используйте свойство **Format** (Формат).

12. Свойство **InputMask** (Маска ввода) позволяет задать шаблон.

13. Если вы создаете поле, информация которого должна быть доступна только для чтения, необходимо установить значение свойства **ReedOnly** (Только чтение) равным True (Истина).

14. Используя свойство **SpecialEffect** (Специальный эффект), заданы стиль отображения поля из двух предложенных вариантов: обычный или с эффектом объемности.

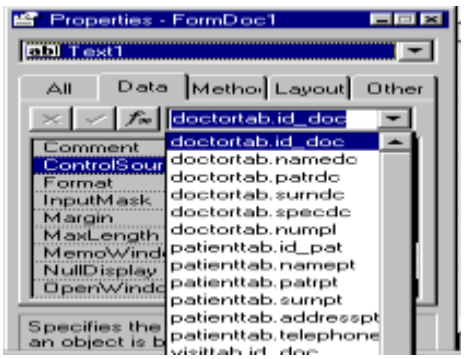


**Рис. 6.** Связывание поля формы с полем таблицы

15. Для поля можно задать длинные поясняющие надписи, выводимые в строку состояния при установке курсора мыши на поле. Для этого используется свойство **StatusBarText** (Текст строки состояния).

16. Visual FoxPro позволяет с помощью свойства **Too1TipText** (Текст подсказки) создавать текст краткого пояснения, появляющийся ниже курсора, когда он установлен на поле. Чтобы отображался текст заданного пояснения, установите для свойства **ShowsTips** (Показывать подсказки) формы значение True.

17. При добавлении новой записи в поле по умолчанию можно вводить наиболее часто встречающееся значения. Для этого используется свойство Value.

Поля **Edit Box** очень удобны для редактирования символьных полей большого размера и Memo-полей. Чтобы разместить поле редактирования в форме, выполняются следующие действия:

1. Нажмите кнопку **Edit Box** (Поле редактирования) на панели инструментов Form Controls (Элементы управления формы)
2. Щелкните в том месте формы, где вы предполагаете разместить поле редактирования.
3. Откройте окно свойств Properties (Свойства) для размещенного объекта
4. При использовании поля данного типа для просмотра и редактирования полей большого размера в его правой части можно расположить вертикальную полосу прокрутки, предназначенную для просмотра информации, не поместившейся в окне. Для этого необходимо в свойстве **ScrollBars** (Полоса прокрутки) задать значение Virtical (Вертикальная). При установке значения None (Нет) полоса прокрутки у поля будет отсутствовать.

Остальные свойства поля редактирования аналогичны свойствам поля ввода, которые описаны выше.

Как правило, в полях формы отображается информация об одной за-

писи. Для просмотра всех записей таблицы необходимо добавить в форму средства перемещения по записям. В СУБД имеются удобные инструменты, предназначенные для этих целей, — кнопки.

Для размещения кнопок управления в форме можно использовать две кнопки панели инструментов Form Controls:

- одиночные кнопки
- группа кнопок

Для размещения в форме **одной кнопки** выполните следующие действия:

1. Нажмите кнопку **Command Button** на панели инструментов **Form Controls** и щелкните мышью в месте предполагаемого размещения создаваемой Кнопки.
2. Откройте окно свойств созданного объекта.
3. Кнопка может содержать текст или графическое изображение. При создании кнопки, содержащей текст, скорректируйте свойство Caption, разместив в поле ввода значения текст, который будет отображаться на кнопке. Например, при создании кнопки для закрытия формы введите ***Выход***.
4. При создании кнопки, содержащей графическое изображение, для задания изображения, размещаемого на кнопке, воспользуйтесь свойством **Picture**. Нажмите кнопку, расположенную справа от поля ввода значения свойства. В результате откроется диалоговое окно **Open**, используя которое вы можете выбрать файл на диске, содержащий изображение, и просмотреть его в области **Picture** (Изображение). Чтобы это можно было сделать, необходимо установить флажок **Picture** (Просмотр). После выбора файла нажмите кнопку ОК для перенесения изображения на кнопку.
5. Кнопка размещена в форме. Теперь необходимо, используя автоматически вызываемый при нажатии на кнопку метод объекта **Click**. (Нажатие), определить действия, выполняемые при нажатии на эту кнопку. Отобразите в окне свойств объекта список всех методов. Для этого в окне Properties выберите вкладку **Methods** (Методы).
6. Установите курсор на метод **Click** и щелкните кнопкой мыши. Откроется окно процедур.
7. Введите команды, которые должны выполняться при нажатии на данную кнопку. Например, если вы создаете кнопку для выхода из формы, это могут быть следующие команды, использующие функцию MESSAGEBOX():

* Запрос для выхода из формы

```
IF MESSAGEBOX  ("Выходить из формы?",4+32+256, "Выход")=6
_screen.ActiveForm.Release ()
ELSE
_screen.ActiveForm.Refresh ()
ENDIF
```

8. Закройте окно процедур. Кнопка создана.

Некоторые объекты СУБД, такие, как **CommandGroup** (Группа кнопок) и **OptionGroup** (Переключатель), являются *составными объектами,* т. к. они содержат несколько объектов, имеющих свои собственные свойства.

Для работы СУБД предоставляет в распоряжение разработчика контекстное меню, содержащее команду **Edit** (Правка), переводящую объект в режим редактирования и позволяющую управлять входящими в его состав простыми объектами: перемещать их внутри рамки, изменять размеры, цвет, настраивать другие свойства. В режиме редактирования вокруг составного объекта появляется заштрихованная рамка (рис. 7).

**ДЛЯ ВЫХОДА ИЗ РЕЖИМА РЕДАКТИРОВАНИЯ НЕОБХОДИМО ЩЕЛКНУТЬ ВНЕ ОБЛАСТИ СОСТАВНОГО ОБЪЕКТА.**

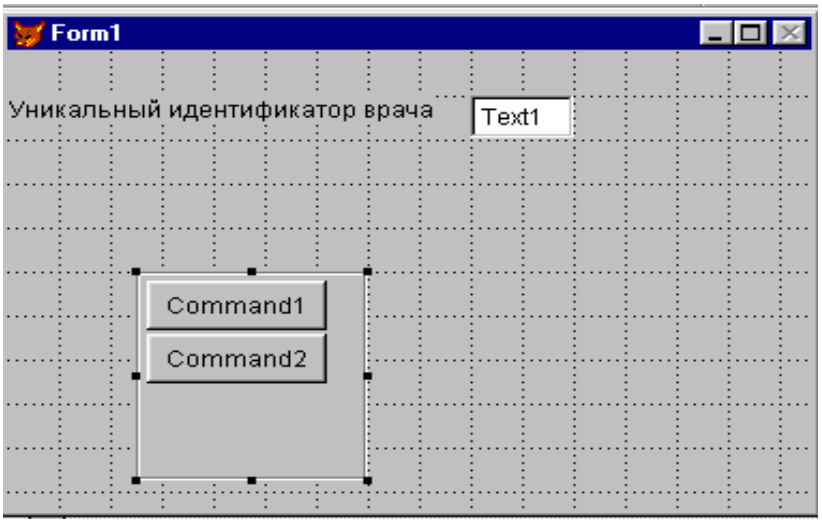


**Рис. 7.** Объект CommandGroup в режиме редактирования

Для размещения в форме группы из нескольких кнопок можно использовать инструмент **Command Group** панели **Form Controls**. Создаваемый при этом объект является составным и обладает свойством **ButtonCount** (Количество кнопок), определяющим количество входящих в его состав кнопок.

Воспользуемся данным инструментом для создания в форме кнопок перемещения по записям таблицы и кнопки выхода из формы.

1. Для создания набора кнопок нажмите кнопку **Command Group** (Группа кнопок) на панели инструментов Form Controls и щелкните в месте их предполагаемого размещения в форме.
2. Откройте окно свойств для размещенного составного объекта.
3. Свойство ButtonCount объекта определяет количество кнопок, размещаемых в объекте (по умолчанию 2). Скорректируйте его, задав необходимое количество кнопок, например 5.
4. Увеличьте с помощью мыши размеры рамки, окружающей созданный объект, чтобы в ней можно было расположить горизонтально все пять кнопок.
5. Переведите объект в режим редактирования. Для этого установите на него курсор, нажмите правую кнопку мыши и выберите из контекстного меню команду Edit (рис.7).
6. Выделяя поочередно кнопки, переместите их, расположив горизонтально в одну линию.
7. Выйдите из режима редактирования, щелкнув вне области объекта CommandGroup.
8. Скорректируйте размер рамки, окружающей составной объект.
9. Откройте окно свойств объекта типа CommandGroup. Нажмите кнопку раскрытия списка в верхней части данного окна. Отметьте, что этот список содержит все объекты, размещенные в форме, а также все элементы, входящие в составной объект, под именами Command1, Command2, Command3, Command4, Command5. Выбирая поочередно элементы в этом списке, можно изменить свойства каждой кнопки.
10. Используя свойство Caption каждого элемента составного объекта, задайте названия кнопок: Первая, Следующая, Предыдущая, Последняя и Выход.
11. Для задания цвета фона, на котором располагаются кнопки, используйте свойство BackColor объекта CommandGroup. Если вы хотите, чтобы он совпадал с цветом фона формы, установите для свойства BackStyle (Стиль фона) значение Transparent (Прозрачный).
12. Теперь необходимо определить команды, которые будут выполняться при нажатии на данные кнопки. Для этого, открывая поочередно окно процедур метода Click (Нажатие) каждого элемента, входящего в составной объект, введите следующие команды:

Для кнопки **Первая:**

* Переходим на первую запись и обновляем информацию в форме

```
IF    !BOF()
   GO TOP
 ENDIF
_screen.ActiveForm.Refresh ()
```

Для кнопки **Следующая:**

```
* Переходим на следующую запись и обновляем информацию в форме
IF !EOF()
   SKIP
 ENDIF
_screen.ActiveForm.Refresh ( )
```

Для кнопки **Предыдущая:**

```
 * Переходим на предыдущую запись и обновляем информацию в форме
 IF !BOF()
    SKIP-1
  ENDIF
 _screen.ActiveForm.Refresh ( )
```

Для кнопки **Последняя:**

```
* Переходим на последнюю запись и обновляем информацию в форме
 IF !EOF()
    GO BOTTOM
  ENDIF
 _screen.ActiveForm.Refresh ( )
```

Для кнопки **Выход:**

```
 * Запрашиваем и выходим, если ДА
 IF MESSAGEBOX  ("Выходить из формы?",4+32+256, "Выход")=6
 _screen.ActiveForm.Release ()
 ELSE
 _screen.ActiveForm.Refresh ()
ENDIF
```

13. После ввода команд закройте окна процедур.
14. Набор кнопок для перемещения по записям таблицы и выхода из формы создан. Запустите форму на выполнение по команде Run Form (Запустить форму) из меню Form (Форма). Для перемещения по записям и закрытия формы используйте кнопки, находящиеся в

нижней части окна (рис. 8).

К числу наиболее употребительных относятся команды удаления и добавления записей. Для добавления записи следует ввести:

```
APPEND BLANK
<_screen.ActiveForm>.REFRESH,
```

а для удаления записи команды имеют вид:

```
DELETE
SKIP
IF EOF()
GO TOP
ENDIF
_screen.ActiveForm.Refresh ( )
```

*Примечание: в случае удаления записи с помощью этой процедуры (а также с помощью кнопки Delete, созданной мастером форм), запись из таблицы будет не удалена, а помечена для удаления.Чтобы удалить записи, помеченные для удаления, можно воспользоваться командой* ***PACK.*** *Например, если написать следующую процедуру в методе* ***Release*** *формы:*

```
OPEN DATABASE <имя базы данных> EXCLUSIVE
use <имя очищаемой таблицы>  EXCLUSIVE
PACK dbf
use
CLOSE DATABASES,
```

*то при закрытии формы из очищаемой таблицы удалятся все записи,помеченные для удаления.*

*Можно также использовать команду* ***PACK DATABASES****, которая удаляет все помеченные для удаления записи из всех таблиц текущей базы данных.*

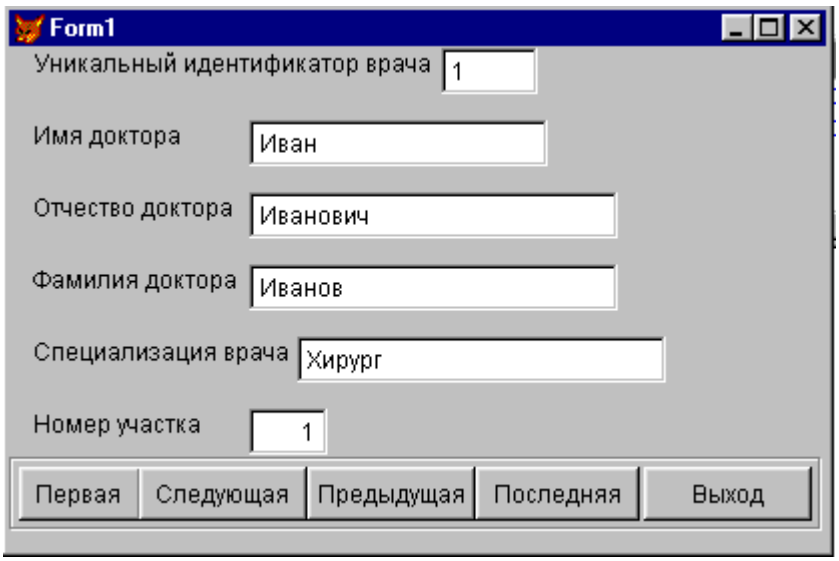


**Рис. 8.** Форма с размещенной группой кнопок

Объекты типа **Option Group** (переключатели) позволяют выбрать **одно из нескольких** значений поля или переменной. Переключатели широко используются не только в СУБД, но и в других приложениях Windows. Объекты типа **Option Group** представляют из себя составные объекты, содержащие внутри себя элементы, наделенные собственными свойствами. Ниже перечислены некоторые из свойств объектов данного типа.

Таблица 2

| **Свойство** | **Описание** |
|---|---|
| ButtonCount | Задает количество опций |
| Style | Определяет вид переключателя |
| Left,Top | Расстояние между кнопками |
| BorderStyle | Стиль обрамления |

Рассмотрим создание переключателя для просмотра и редактирования поля **Gr_kr** (группа крови), которое добавлено в таблицу PatientTab. Данное поле может принимать одно из значений: **первая, вторая, третья, четвертая**.

1. Создать форму для таблицы PatientTab, расположив в ней заголовок формы, текстовые объекты и все поля, за исключением **Gr_kr**.
2. Выбрать инструмент **Option Group** на панели инструментов FormControl.
3. Установить указатель мыши на место предполагаемого расположения поля **Gr_kr.** Удерживая кнопку мыши нажатой, установить рам-

ку требуемого размера.

4. Открыть окно **Properties** для вновь созданного объекта.

5. Скорректировать свойство **ButtonCount**, задав количество опций, равным 4.

6. Из списка объектов в верхней части окна **Properties** выбрать первую опцию переключателя Option1 . При этом объект выделяется прямоугольниками.

7. Для объекта Option1 скорректировать свойства **Caption, ForeColor, BackColor, FontName**, определяющие заголовок, цвет шрифта, фон, вид шрифта и т.д.

8. Аналогично корректируются свойства для остальных трех объектов.

9. С помощью инструмента **Label** создать надпись «Группа крови» над объектом **Option Group**.

Список опций объекта **Option Group** можно расположить горизонтально, если использовать режим **Edit (**см. рис.7).

Для размещения в форме полей, которые могут иметь **только одно из двух** допустимых значений, используются объекты типа **CheckBox**, называемые ***флажками***. Объекты данного типа могут использоваться в форме по одному или группами.

Рассмотрим следующий пример. Таблица DoctorTab может содержать поле, указывающее, является ли данная больница постоянным местом работы врача. При установке флажка значение в поле будет соответствовать 1, а при сбросе флажка — 0.

Рассмотрим подробно процедуру создания флажка для редактирования поля, указывающего, является ли данная больница постоянным местом работы врача. Это поле имеет тип Logica1 и может принимать значения 0 или 1.

1. Откройте в окне конструктора проекта форму, позволяющую просматривать список клиентов.

2. Нажмите кнопку **Check Box** (Флажок) на панели инструментов **Form Controls** (Элементы управления формы).

3. Щелкните в месте предполагаемого размещения флажка. Объект разместиться в форме.

4. Выделите созданный объект и выберите в окне конструктора проекта *у* меню **View** (Вид) команду **Properties** (Свойства). Откроется окно объекта **Properties** типа **CheckBox**.

5. Для связывания флажка с полем таблицы необходимо это поле в таблице создать (добавьте в таблицы новое поле post, которое будет

иметь тип Logical) скорректируйте свойство **ConrolSource**, задав в качестве источника данных поле Post таблицы DoctorTab.

6. Введите в поле свойства **Caption** (Надпись) текст ***Постоянное место работы.*** Данный текст будет размещен справа от флажка в окне конструктора форм, а также в форме.

7. Флажок создан. Запустите форму на выполнение. Для установки признака постоянного места работы установите флажок так, как показано на рис. 9.

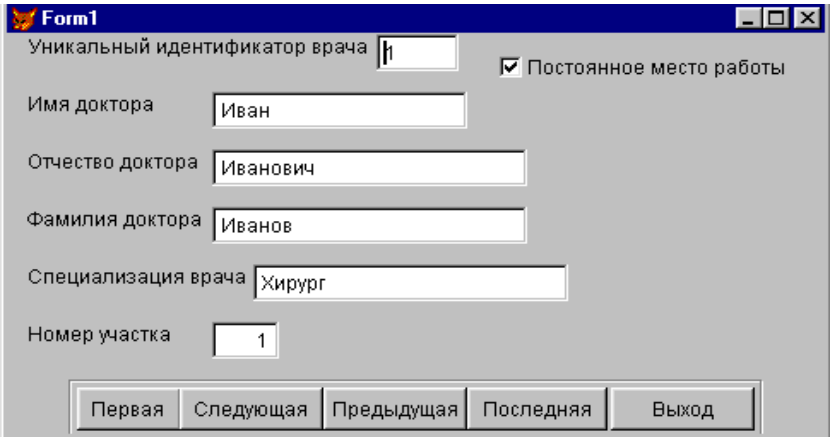


**Рис. 9.** Использование в форме объекта типа CheckBox

СУБД позволяет размещать в форме линии, прямоугольники, прямоугольники со скругленными углами, круги, эллипсы, используемые для объединения в группу похожих объектов и улучшения внешнего вида формы.

Для добавления в форму вертикальной или горизонтальной **линии** выполните следующие действия:

Нажмите кнопку **Line** (Линия) на панели инструментов **Form Controls** (Элементы управления формы).

Установите указатель мыши в то место, где должна начинаться линия, и, не отпуская кнопку мыши, переместите его до получения линии нужной длины.

Используя свойство **BorderWidth**, введите число, которое будет определять толщину линии.

Если линия вертикальная, для задания ее длины используйте свойство **Height** (Высота). Для задания длины горизонтальной линии предназначено свойство **Width** (Ширина).

Для задания цвета линии используйте свойство **BorderColor**.

С помощью свойства **BorderStyle** укажите стиль линии, применяя следующие значения:

Таблица 3

| Значение | Стиль линии |
|---|---|
| 0-Transparent | Линия отсутствует (имеет цвет фона) |
| 1-Solid (Default) | Тонкая линия |
| 2-Dash | Штриховая линия |
| 3-Dot | Пунктирная линия |
| 4-Dash-Dot | Штрих-пунктир |
| 5- Dash-Dot-Dot | Штрих-двойной пунктир |
| 6-Inside Solid | Непрерывная линия |

Для добавления в форму **контура** и настройки его свойств выполните следующие действия:

Нажмите кнопку **Shape** (Контур) на панели инструментов **Form Controls** (Элементы управления формы).

Установите указатель мыши в то место, где должен начинаться объект и переместите указатель мыши до получения квадрата или прямоугольника нужного размера.

Свойство **Curvature** (Изгиб), которое может принимать целочисленные значения в диапазоне от 1 до 99, применяется для придания созданному объекту формы, отличной от прямоугольника или квадрата.

Используя свойство **BackStyle**, задайте, будет ли созданный объект прозрачным.

С помощью свойства **FillStyle** задайте узор заполнения:

Таблица 4

| Значение | Узор заполнения |
|---|---|
| 0-Solid | Сплошное заполнение |
| 1-Transparent (Default) | Нет заполнения |
| 2-Horizontal Line | Горизонтальная штриховка |
| 3-Vertical Line | Вертикальная штриховка |
| 4-Upward Diagonal | Штриховка по диагонали слева направо |
| 5-Downward Diagonal | Штриховка по диагонали справа налево |
| 6-Cross | Горизонтально-вертикальная штриховка |
| 7-Diagonal Cross | Штриховка по диагонали в обоих направлениях |

Используя свойство **FillCo1or**, задайте цвет узора заполнения объекта.

Свойство **BorderStyle** предназначено для задания стиля рамки объекта. Оно может принимать те же значения, что и для линии.

Чтобы придать объемность контуру, используйте свойство **SpecialEffect**.

Списки в СУБД используются для отображения в форме элементов, которые могут быть заданы с помощью массива, меню, списка файлов, значений поля таблицы и т. д.

СУБД позволяет использовать разные объекты для отображения в форме одного и того же поля. Если вводимых в поле значений много, удобно применять *списки,* т. е. объекты типа **ListBox** (Список).

Для указания источника данных списка используется свойство **RowSourceType** (Тип источника данных), содержащее следующие значения:

Таблица 5

| **Значение** | **Источник данных** |
|---|---|
| 0 (None) | Значения элементов списка определяются программно с помощью методов AddItem (Добавить объект) или AddListItem (Добавить объект списка) |
| 1 (Value) | Список задается в виде строки, элементы в которой разделяются запятыми |
| 2 (Alias) | В качестве источника данных используется таблица. Количество выводимых полей таблицы определяется значением свойства ColumnCount |
| 3 (SQL Statement) | Список содержит данные, полученные в результате выполнения SQL оператора |
| 4 (Query) | Список содержит данные, полученные в результате выполнения указанного запроса. Запрос задается именем файла с расширением .OPR |
| 5 (Array) | Источником данных является заданный массив |
| 6 (Fields) | Значения элементов списка определяются полями таблицы |
| 7 (Fields) | Список содержит перечень файлов текущей папки. В свойстве RowSource вы можете задать шаблон выбора файлов |
| 8 (Structure) | В качестве источника данных используется структура таблицы |
| 9 (Popup) | Список содержит пункты всплывающего меню |

Объекты типа **ListBox** имеют дополнительные свойства, которые отсутствовали у ранее рассмотренных объектов:

Таблица 6

| Свойство | Назначение |
|---|---|
| ColumnCount | Определяет число колонок в списке |
| FirstElement | Задает первый элемент массива, который будет отображаться в списке |
| NumberOfElements | Определяет количество элементов массива, отображаемых в списке |
| RowSourse | Указывает источник данных списка |

Разместим в форме, предназначенной для редактирования списка клиентов из таблицы DoctorTab, список, который будем использовать для ввода специализации врача. В качестве источника данных для списка будем использовать таблицу.

1. Удалите из формы поле ввода специализации врача, поскольку в данном примере для ввода специализации врача будет использоваться список.
2. Нажмите кнопку **List Box** (Список) на панели инструментов **Form Controls**.
3. Щелкните на месте удаленного поля ввода.
4. Откройте окно Properties (Свойства) размещенного в форме списка.
5. Чтобы связать созданное поле с полем таблицы DoctorTab, выберите свойство **ControlSource** (Источник данных). В поле ввода значения свойства воспользуйтесь кнопкой раскрытия списка и из списка всех полей открытой таблицы DoctorTab выберите поле SpecDc.
6. Скорректируйте свойство **RowSourceType**, которое указывает тип источника данных. Выберите из списка значение **Fields** (Поля), поскольку список специализаций располагается в поле SpecDc таблицы DoctorTab.
7. Скорректируйте свойство **RowSource** (Источник данных списка), задав в поле ввода значения свойства doctortab. specdc.
8. Запустите форму на выполнение. Теперь при редактировании списка специализации врачей в поле SpecDc таблицы DoctorTab будет заноситься значение, выбираемое из списка (рис.10).

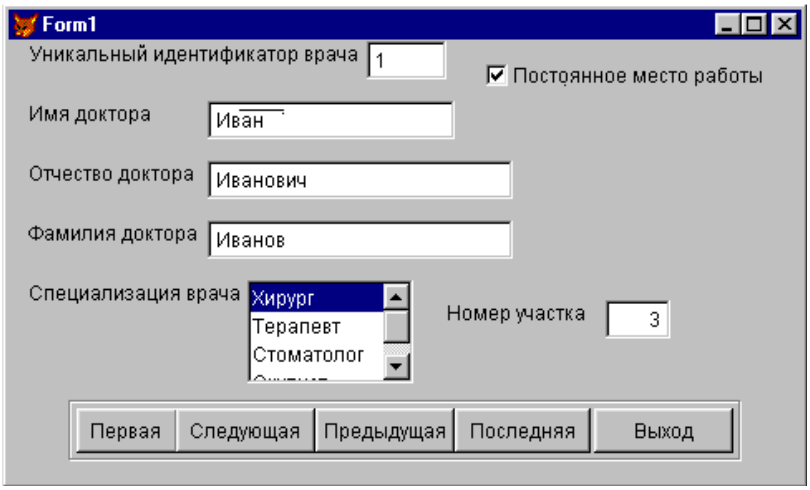


**Рис. 10.** Вывод специализации врача из списка

В СУБД существуют два вида списков. Один из них мы рассмотрели выше. Второй вид списка – объект типа **ComboBox** или раскрывающийся список, который удобно использовать в том случае, если вводимых значений много, а места в форме для расположения обычного списка не хватает.

Опишем процедуру создания раскрывающегося списка для ввода и редактирования специализации врача таблицы DoctorTab. В качестве источника данных будем использовать строку с наименованиями специальностей.

1. Откройте в окне конструктора форму для ввода информации о врачах.
2. Удалите список для ввода специализации врачей, созданный ранее.
3. Нажмите кнопку **Combo Box** (Раскрывающийся список) на панели инструментов **Form Controls** (Элементы управления формы).
4. Щелкните на месте удаленного объекта. Объект типа **Combo-Box** разместится в форме.
5. Откройте окно Properties для размещенного в форме раскрывающегося списка.
6. Свяжите раскрывающийся список с полем SpecDc таблицы DoctorTab, используя свойство **ControlSource**.
7. Из списка возможных значений свойства **RowSourceType** (Источник данных списка) выберите Value (Значение).
8. Введите в поле ввода значения свойства **RowSource** перечень допустимых элементов списка через запятую: **Хирург, Терапевт, Стоматолог, Окулист** (рис.11).

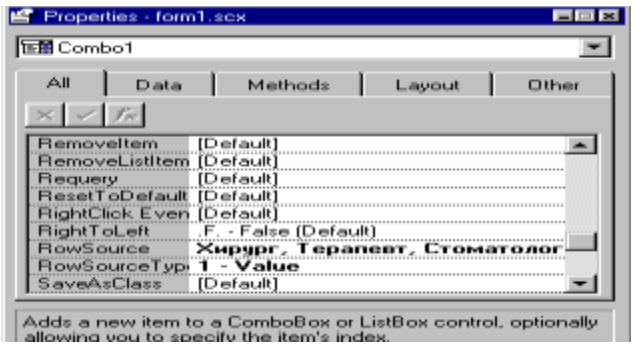


**Рис. 11.** Определение свойств раскрывающегося списка

9. Запустите форму на выполнение. Теперь при редактировании списка врачей для ввода в таблицу DoctorTab специализации нажмите кнопку раскрытия созданного списка и выберите из него нужное значение. Это значение будет введено в поле SpecDc таблицы (рис. 12).

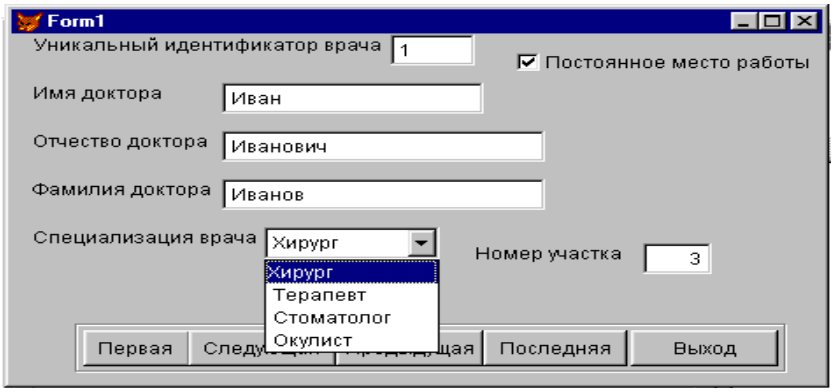


**Рис. 12.** Выбор специализации врача из раскрывающегося списка

К сложным или составным объектам относятся, прежде всего, форма, а также объекты Grid, PageFrame и другие.

Рассмотрим объект **Grid**. Этот объект предназначен для работы с несколькими записями таблицы одновременно. Внешне он похож на окно **Browse**, но в отличие от него может быть включен в форму.

Объект **Grid** позволяет работать с данными из нескольких таблиц, связанных отношением один-ко-многим. В составе объекта **Grid** могут быть объекты TextBox, CheckBox, Spinner, EditBox и др. Столбцы и заголовки столбцов объекта **Grid** управляются независимо и самостоятельно реагируют на свой набор событий.

Для более полного использования системных возможностей применяют **Grid Builder** (Построитель сетки). Запустить построитель можно одним из способов:

1. Выбрать объект **Grid**, а затем в контекстном меню – пункт **Builder.**

2. Выбрать среди элементов Form Controls объект **Builder Lock**, а затем – **Grid.**

В любом случае откроется окно **Grid Builder (**рис. 12). Это окно имеет четыре вкладки:

Таблица 7

| **Вкладка** | **Назначение** |
|---|---|
| Grid Items | Выбор базы данных, таблиц<br>и их полей |
| Style | Выбор стиля отображения таблицы |
| Layout | Присвоение в поле редактирования Caption русских названий полям,<br>а в поле Control type – определение типа объекта,<br>используемого в каждом столбце таблицы |
| Relationship | Определение отношения между<br>связанными таблицами |

Создадим объект **Grid** для таблицы PatientTab из базы данных аэропорта. Для этого на вкладке **Grid Items** выберем таблицу PatientTab и все ее поля для отображения в объекте (рис. 12). На вкладке **Style** выберем стиль для нашей таблицы – Embossed (рис. 13).

На вкладке **Layout** присвоим столбцам таблицы русские названия и при необходимости изменим ширину столбцов (рис. 14).

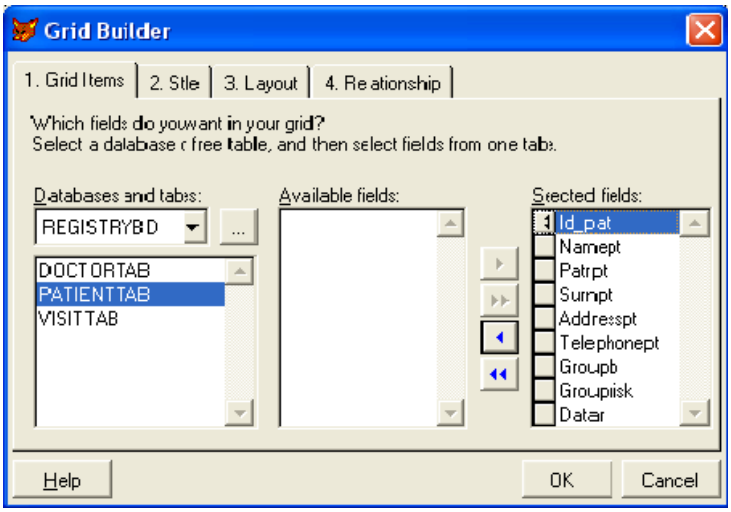


**Рис. 12.** Построитель объекта **Grid**

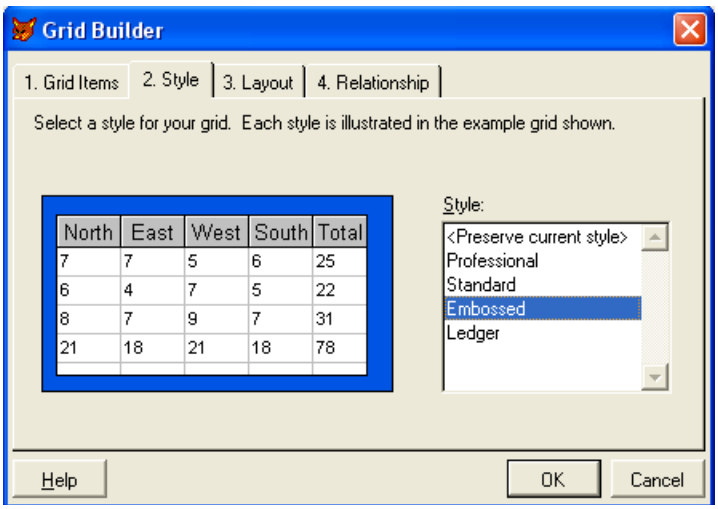


**Рис. 13.** Выбор стиля отображения таблицы

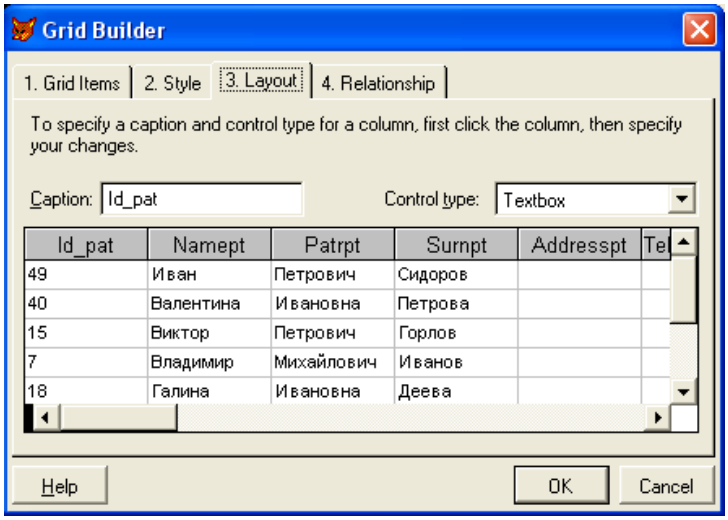


**Рис. 14.** Редактирование столбцов

Результат работы построителя представлен на рис. 15.

Каждый объект **Grid** состоит как минимум из трех объектов:

- **Column** – столбец
- **Header** – заголовок
- **Text** – информация, определяемая в столбце

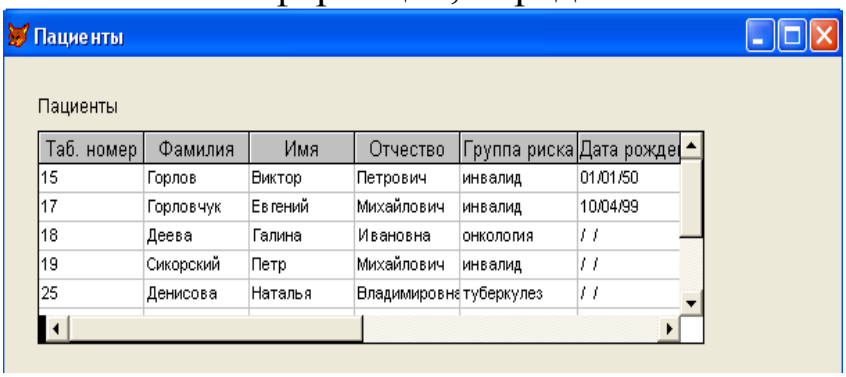


**Рис. 15.** Форма с объектом **Grid**

Как сам объект **Grid**, так и входящие в него объекты, имеют свой собственный набор свойств. Выделить объект, входящий в состав **Grid** можно:

1. Выбрать объект **Grid**, а затем в контекстном меню – **Edit**.
2. Выбрать в окне **Properties** из выпадающего списка нужный элемент.

Ниже приведены свойства объекта **Grid**:

Таблица 8

| **Свойство** | **Значение** |
|---|---|
| ColumnCount | Количество столбцов |
| GridLineColor | Цвет разделительных линий |
| GridLines | Наличие вертикальных и горизонт. разделительных линий |
| RecordMark | Наличие указателя текущей записи |
| RecordSource | Имя источника данных |
| RecordSourceType | Тип источника данных (Table, Alias,Prompt, Query, SQL Statement) |
| ChildOrder | Индекс для связывания таблиц |
| GridLineWidth | Толщина разделительных линий |
| Alignment | Выравнивание объектов Column, Header, Text |
| ControlSource | Источник информации для каждого столбца |
| ColumnOrder | Порядок столбцов |
| Caption | Заголовок объекта Header |

Имеется множество вариантов для определения используемого цвета и шрифта.

Рассмотрим изменение типа объекта, входящего в состав **Grid**. В таблице Пациенты изменим в колонке Группа риска объект **TextBox** на **ComboBox**. Для этого:

1. Выберите режим редактирования таблицы – **Edit**.
2. В окне Form Controls выберите элемент **ComboBox**, щелкните левой кнопкой на колонке, в которую Вы хотите вставить элемент (Column5)
3. Измените у объекта **Combo1** свойство **RowSourceType** на **Value**, а в свойстве **RowSource** впишите через запятую перечень допустимых элементов списка: **инвалид, СПИД, туберкулез, онкология, астма.**

4. У элемента **Column5** в свойстве **CurrentControl** выберите значение **Combo1**.

5. При необходимости измените высоту строк – свойство **RowHeight** объекта **Grid**.

Результат изменений представлен на рис. 16.

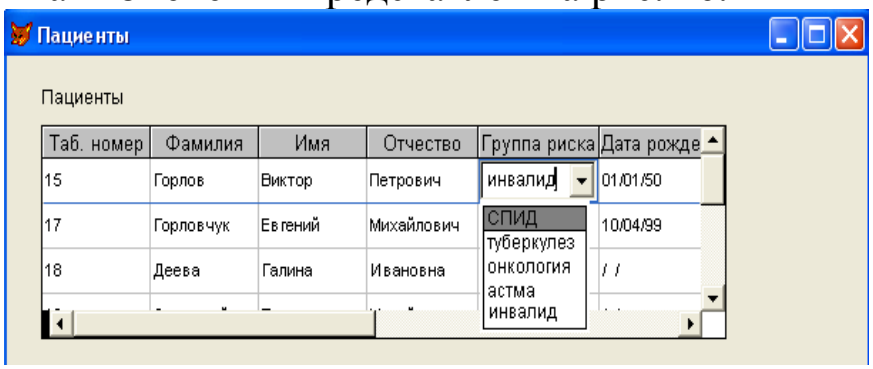


**Рис. 16.** Форма с измененным объектом **Grid**

**Порядок выполнения работы**

*Задание 1.*

1. Настроить параметры и определить среду окружения создаваемой формы в конструкторе форм.

2. Разместить в форме заголовки и надписи к полям (кнопка **Label**).

3. Разместить в форме поля ввода (кнопка **Text Box**) и редактирования символьных полей (кнопка **Edit Box**).

4. Сохранить форму на диске и запустить ее на выполнение по команде **Run Form** из меню **Form.**

*Задание 2*

1. Для формы, разработанной в соответствии с заданием 1, создать группу кнопок для перемещения по записям, добавления (удаления) записи и выхода из формы.

2. Ввести в форму флажок и переключатель для редактирования полей с двумя и несколькими возможными значениями соответственно.

3. Добавить в форму линии и контуры для улучшения ее внешнего вида.

4. Сохранить форму на диске и запустить ее на выполнение для проверки полученных результатов.

*Задание 3*

1. Для формы, разработанной в соответствии с заданиями 1,2, предусмотреть наличие списка (**List Box**) для ввода одного из нескольких возможных значений.

2. Выбрать в качестве источника данных (**RowSourceType**) списка значение **Fields/Structure** для некоторой таблицы.

3. Предусмотреть для создаваемой формы наличие хотя бы одного раскрывающегося списка (**Combo Box**).

4. Выбрать в качестве источника данных (**RowSourceType**) раскрывающегося списка значение **Value/Alias/Files**.

5. Сохранить форму на диске и запустить ее на выполнение. Сравнить результаты при варьировании источников данных для списков в п.2,4.

**Контрольные задания**

1. Создать форму для таблицы *DoctorTab*. При этом поле *Status* должно быть представлено объектом *CheckBox*. Создать кнопки *Следующая запись* и *Предыдущая запись*.

2. Создать форму для таблицы *DoctorTab*. При этом поле *SpecDc* должно быть представлено объектом *ComboBox*. Создать кнопки *Следующая запись* и *Предыдущая запись*.

3. Создать форму для таблицы *DoctorTab*. Создать кнопки *Удалить запись* и *Добавить запись*.

4. Создать форму для таблицы *PatientTab*. При этом поле *GroupB* должно быть представлено объектом *ComboBox*. Создать кнопки *Следующая запись* и *Предыдущая запись*.

5. Создать форму для таблицы *PatientTab*. Создать кнопки *Удалить запись* и *Добавить запись*. Предусмотреть удаление записей, помеченных на удаление, при закрытии формы.

6. Создать форму для таблицы *VisisTab*. Создать кнопки *Следующая запись, Предыдущая запись, Первая запись, Последняя запись*.

7. Создать форму для таблицы *Chekipag*. При этом поле *Spec* должно быть представлено объектом *ComboBox*. Создать кнопки *Следующая запись* и *Предыдущая запись*.

8. Создать форму для таблицы *Models*. Создать кнопки *Следующая запись* и *Предыдущая запись*.

9. Создать форму для таблицы *Models*. Создать кнопки *Удалить запись* и *Добавить запись*. Предусмотреть удаление записей, помеченных на удаление, при закрытии формы.

10. Создать форму для таблицы *Reis*. Создать кнопки *Удалить запись* и *Добавить запись*. Предусмотреть удаление записей, помеченных на удаление, при закрытии формы.

11. Создать форму для таблицы *DoctorTab* с использованием объекта *Grid* для таблицы. При этом поле *Status* должно быть представлено объектом *CheckBox*.

12. Создать форму для таблицы *DoctorTab* с использованием объекта *Grid* для таблицы. При этом поле *SpecDc* должно быть представлено объектом *ComboBox*.

13. Создать форму для таблицы *PatientTab* с использованием объекта *Grid* для таблицы. При этом поле *GroupB* должно быть представлено объектом *ComboBox*.

14. Создать форму для таблицы *Reis* с использованием объекта *Grid* для таблицы. При этом поле *Status* должно быть представлено объектом *CheckBox*. Добавить кнопку *Выход*.

15. Создать форму для таблицы *Models* с использованием объекта *Grid* для таблицы. Добавить кнопку *Выход*.

**Контрольные вопросы**

1. Перечислить этапы создания формы с помощью конструктора форм.
2. В чем состоит и как выполняется настройка параметров формы?
3. Как создать среду окружения формы?
4. Как разметить текстовую информацию в форме?
5. Какие действия выполняются при размещении в форме полей ввода и редактирования?
6. Пояснить порядок создания одиночных кнопок.
7. Как создается составной объект **Command Group** в конструкторе форм?
8. Как определяются команды, выполняемые при нажатии кнопок в форме?
9. Пояснить назначение флажков и переключателей.
10. Каков порядок размещения в форме переключателя (**Option Group**) и определения свойств составляющих его объектов?
11. Как разместить в форме флажок (**Check Box**) и задать его свойства?
12. Перечислить этапы создания и размещения в форме линий и контуров.
13. В каких случаях при вводе целесообразно применять списки?
14. Как указать источник данных списка?
15. Как связать список с полем таблицы?
16. Как разместить в форме раскрывающийся список?

## ЛАБОРАТОРНАЯ РАБОТА №5
# РАЗРАБОТКА ОТЧЕТОВ В СУБД

**Цель работы:** Приобретение навыков в создании отчетов с помощью СУБД.

**Краткие теоретические сведения**:

СУБД FoxPro имеет мощные средства построения отчетов для вывода данных в желательном для пользователя виде на принтер, экран или в текстовый файл.

Перед началом построения отчета необходимо продумать его расположение на странице и порядок вывода данных. Сделайте эскиз отчета на листе бумаги, учтите общую ширину всех полей, которые вы хотите распечатать, подумайте о длине заголовков. Тем самым вы получите представление о требуемом формате отчета, только не забудьте о реальных возможностях вашего принтера.

Порядок расположения данных в отчете и его элементы (для примера его длина взята в объеме трех страниц) приведены на рис. 1. Данные, помещаемые в полосу Details, распечатываются многократно с автоматическим переводом указателя записи в таблице или курсоре, которые являются источником данных. Группировка данных позволяет выводить их в систематизированном виде, например, список врачей по специальности и т. п.

**Для создания отчетов используются:**

- Мастер отчетов (Report Wizard), позволяющий быстро создать отчет, выбрав параметры сортировки и группировки данных, стиль отображения и расположение данных**.**
- Стандартный отчет (Quick Report), позволяющий создавать стандартный отчет, в котором поля отчета располагаются автоматически по внутреннему алгоритму Visual FoxPro.

- Конструктор отчета (Report Designer), в котором пользователь самостоятельно разрабатывает собственный отчет.

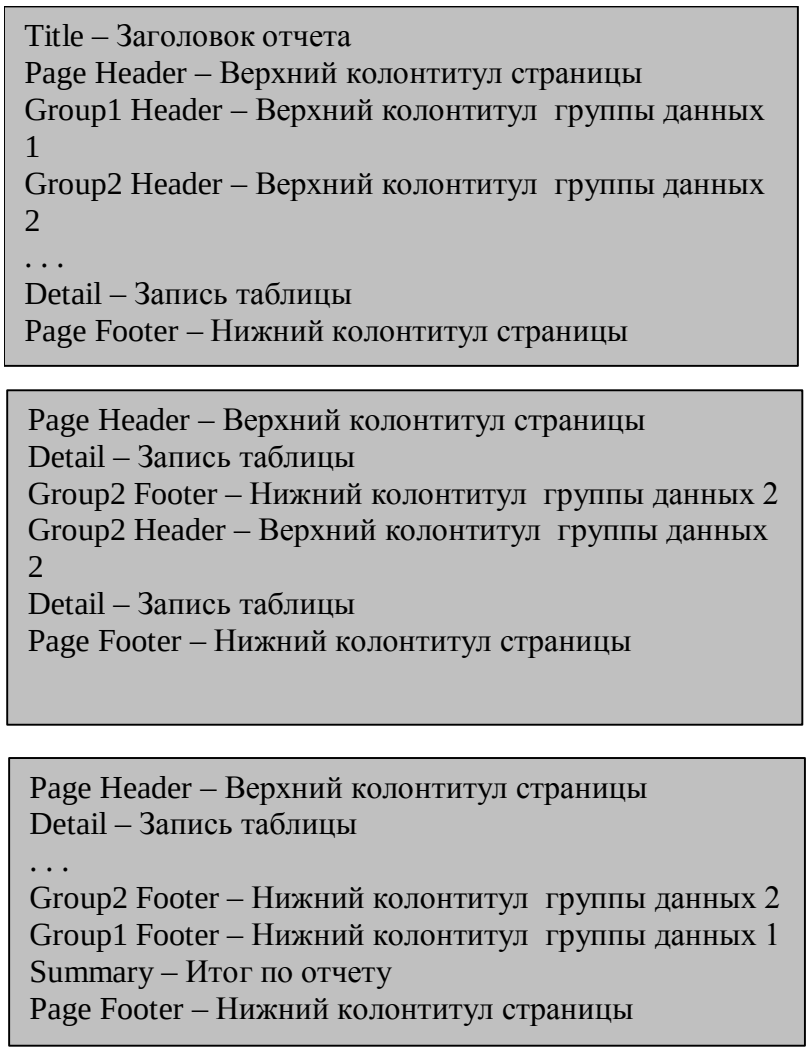


Рис. 1. Порядок вывода различных данных в отчете

Создание отчета с помощью мастера не требует специальных знаний и сводится к выбору таблиц, входящих в отчет, определения списка полей отчета и порядка их размещения.

а) Запуск. Запуск мастера по созданию отчета выполняется командой File → New выбором опции Report и нажатием кнопки Wizard.

б) Тип отчета. На экране появится окно диалога, позволяющее определить требуемый вид отчета:

- One-to-Many Report Wizard – создает отчет для таблиц с отношением один-ко-многим.

- Report Wizard – создает простой однотабличный отчет.

Рассмотрим использование мастера для создания однотабличного отчета. Выберем тип Report Wizard и нажмем кнопку OK.

в) Выбор полей. На первом шаге для выбора базы данных и таблицы используются два связанных списка в области «Databases and tables». Верхний список предназначен для выбора базы данных, нижний – для выбора таблицы из БД. Для открытия БД можно нажать кнопку справа от списка баз данных (на экране откроется окно диалога Open). Список Available fields содержит перечень всех полей выбранной таблицы, из которого с помощью кнопок переноса формируется список Selected fields полей создаваемого отчета. Нажимается Next для перехода к следующему шагу.

г) Группировка данных. При необходимости определяются поля, по которым будет осуществляться группировка данных в отчете. С помощью мастера можно создать не более трех группировок в отчете.

Для задания группировки используются три раскрывающихся списка, в каждом из которых выбирается поле для группировки данных. На этом же шаге определяются интервалы группировки и итоговые значения отчета.

Нажатие кнопки Grouping options открывает окно диалога Grouping Intervals, с помощью которого выбирается один из интервалов:

- Entire Field – интервал группировки задается исходя из полного значения поля таблицы.

- Ist Letter – группировка осуществляется по первой букве значения поля таблицы.

- 2,3,4,5 Initial Letters – группировка осуществляется по 2,3,4,5 первым буквам значения поля таблицы, соответственно.

Нажатие кнопки Summary Options открывает окно Summary Options, содержащее таблицу со строками из полей и столбцами возможных итоговых значений (Sum - итоговая сумма, Avg - итоговое среднее, Count - итоговое количество строк, Min - итоговое минимальное значение поля, Max - итоговое максимальной значение поля). В окне диалога Summary Options расположена область с опциями:

- Detail and Summary – в отчете отображается область данных, промежуточные итоговые значения по группировкам, а также конечные итоговые значения по отчету.

- Summary only - отображается область данных и конечные итоговые значения по отчету

- No totals – в отчете отображается только область данных.

Нажать Next для перехода к следующему шагу.

д) Стиль отображения. На третьем шаге раскрывающийся список Style мастера позволяет выбрать и просмотреть (в правом верхнем углу диалога) пять вариантов отображения объектов в отчете. Нажать Next.

е) Порядок размещения. Устанавливается ориентация страницы отчета и порядок размещения объектов в отчете. Нажать Next.

ж) Критерии сортировки. При необходимости из списка Available fields or index tag полей и индексов отчета выбираются поля для сортировки данных отчета и помещаются в список Selected fields.(Перемещение полей выполняется нажатием кнопок Add/Remove). Нажать Next.

з) Заголовок и вариант дальнейшей работы. В поле ввода Type a title for your report задается собственный заголовок отчета и выбирается один из трех возможных вариантов дальнейшей работы с отчетом:

- Save report for later use – сохраняет созданный отчет.

- Save report and modify it in the Report Designer - сохраняет созданный отчет и открывает его в конструкторе отчетов для модификации.

- Save and print report – сохраняет отчет и печатает его.

С помощью кнопки Preview можно просмотреть созданный отчет и для завершения работы нажать Finish.

и) Просмотр и печать отчета. Visual FoxPro располагает большим набором средств для просмотра на экране созданного отчета. Можно использовать любое из перечисленных средств:

- Команда File → Print Preview основного меню.

- Команда View → Preview основного меню.

- Команда Preview всплывающего меню.

- Кнопка Print Preview на стандартной панели инструментов Visual FoxPro.

Для печати созданного отчета можно пользоваться одним из способов:

- Команда File → Print основного меню.

- Команда Report →Run Report основного меню.

- Команда Print всплывающего меню.

- Кнопка Print или Run на стандартной панели инструментов Visual FoxPro

На экране появится окно Print, позволяющее задать параметры печати (номера печатаемых страниц отчета, критерии выбора записей для печати, ориентация страницы, размер бумаги и др.)

Мастер позволяет создавать отчеты и для двух связанных таблиц (выбором опции One-to-Many Report Wizard при задании типа отчета). При этом задается поле для связи между таблицами и последовательно выбираются для размещения в отчете поля из каждой таблицы.

Стандартный отчет автоматически размещает выбранные поля в пустое окно конструктора отчета. Для завершения создания быстрого стандартного отчета необходимо лишь требуемым образом изменить расположение полей, вставить дополнительный текст и другие объекты **отчета.**

Простейший отчет по данным из одной таблицы удобно строить с помощью команды **Quick Report** меню **Report**. Она автоматически помещает выбранные поля в окно Конструктора отчетов. По умолчанию команда **Quick Report** помещает все выбранные поля в полосу **Details**, а идентификаторы полей в полосу **Page Header**. В полосу **Page Footer** слева помещается поле с функцией **DATEQ** для вывода текущей даты, а справа - поле с системной переменной **_PAGENO** (номер страницы) и меткой **Page** перед ним. Если вам нужна простая распечатка данных, то это неплохая заготовка, в которой останется только поменять идентификаторы полей на их заголовки и, прибавив к отчету полосу **Title**, офор-

мить его название. Кстати, для подготовки отчетов можно гораздо шире, чем это было в случае с созданием формы, использовать Мастер создания отчета, так как объекты отчета не привязаны к классам.

Быстрый отчет можно создать следующим образом.

1. Открыть диалоговое окно **New**, выбрав команду **File →New .**
2. Установить переключатель в положение Report.
3. Щелкнуть на кнопке New File. Visual FoxPro откроет окно конструктора отчетов Report Designer .
4. Выбрать в меню команду Report → Quick Report. Visual FoxPro откроет диалоговое окно Open.
5. В списке Tables in Database дважды щелкнуть на названии таблицы, которая будет использоваться в отчете. Visual FoxPro откроет диалоговое окно Quick Report.
6. С помощью группы параметров Field layout выбрать тип макета отчета. По умолчанию выбран ленточный отчет, возможен вариант отчета по столбцам.
7. Открыть диалоговое окно Field Picker, щелкнув на кнопке Fields.
8. В списке All fields дважды щелкнуть на именах полей, которые будут помещены в отчет. Visual FoxPro занесет имена этих полей в список Selected fields.
9. Щелкнуть на кнопке ОК и вернуться в окно Quick Report.
10. Щелкнуть на кнопке ОК и отчет будет отображен в окне конструктора отчета Report Designer.
11. Для запуска отчета выбрать команду Report → Run Report (для просмотра - Print Preview).

На рис. 2 приведен отчет, построенный с помощью команды Quick Report для таблицы DoctorTab. Перед снятием этого изображения с экрана компьютера мы вызвали команду Data Environment меню View

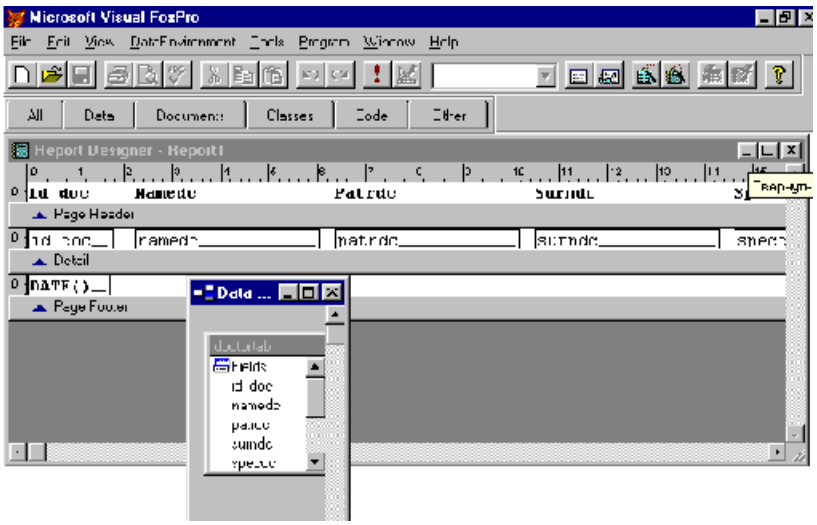


Рис. 2. Отчет, построенный с помощью команды Quick Report

Конструктор отчетов (**Report Designer**) по принципу работы похож на Конструктор форм без учета объектно-ориентированных возможностей. Основные действия по построению отчета выполняются с помощью меню **Report**, которое автоматически появляется в главном меню Visual FoxPro, и панелей инструментов Report Designer и Report Controls. В табл. 1 описаны команды меню **File**, **View** и **Format**, которые отличаются от команд, используемых при проектировании класса или формы.

Таблица 1. Средства работы в Конструкторе отчетов

| Команда меню | ***Назначение*** |
|---|---|
| **Меню File** | |
| **Page Setup** | Выводит диалоговое окно для установки параметров страницы и установок принтера. |
| Print Preview | Включает режим просмотра отчета. |
| Print | Выводит отчет на печать. |
| **Меню View** | |
| Design | Включает режим разработки отчета. |
| Preview | Включает режим просмотра отчета. |
| Data Environment | Открывает окно среды окружения. |

| | |
|---|---|
| Report Control Toolbar | Выводит или убирает с экрана панель инструментов Report Control. |
| Layout Toolbar | Выводит или убирает с экрана панель инструментов Layout. |
| Color Palette Toolbar | Выводит или убирает с экрана панель инструментов Color Palette. |
| Report Preview Toolbar | Выводит или убирает с экрана панель инструментов Report Preview. |
| Grid Line | Выводит или убирает координатную сетку. |
| Show Position | Включает или отключает отображение координат текущего объекта. |
| Меню Format | |
| Group | Объединяет несколько объектов для манипуляции с ними как с одним объектом. |
| Ungroup | Разъединяет ранее объединенные объекты. |
| Font | Выводит диалоговое окно для выбора шрифта и его начертания. |
| Text Aligment<br>Left<br>Center<br>Right | Определяет выравнивание текста:<br>Выравнивает текст по левой границе.<br>Выравнивает текст по центру.<br>Выравнивает текст по правой границе. |
| Single Space | Определяет один межстрочный интервал для меток. |
| 1 ½ Space<br>Double Space | Определяет полуторный межстрочный интервал для меток. |
| Double Space | Определяет двойной межстрочный интервал для меток |
| Reading Order | Определяет направление чтения текста для национальных версий Visual FoxPro, в которых направление чтения отличается от общепринятого. |

| | |
|---|---|
| Fill | Заполняет внутренний объем выбранных рамок различной<br><br>штриховкой. |
| Pen | Определяет толщину сплошной линии или вид линии для рамки (пунктирная, штрих-пунктирная и т. п.). |
| Mode<br>Opaque<br>Transparent | Определяет характеристики фона:<br>Определяет непрозрачный фон<br><br>для метки.<br>определяет прозрачный фон для метки. |
| **Меню Report** | |
| *Title/Summar*<br><br>*y* | Добавляет в отчет полосы заголовка и итогов. |
| Data Grouping | Выводит диалоговое окно для задания условий группировки данных. |
| Variables | Определяет переменные для использования при подсчете данных. |

| | |
|---|---|
| Default Font | Выводит диалоговое окно для выбора шрифта и установки его характеристик, которые будут использованы по умолчанию для всех текстовых объектов в отчете. |
| Private Data Session | Устанавливает или отменяет для отчета отдельную сессию данных. В случае выбора отдельной сессии перемещение указателя записи при печати отчета не будет сказываться на форме, в которой используются те же данные. |
| Quick Report | Запускает утилиту быстрого построения отчета. |
| Run Report | Запускает отчет на выполнение. |

Изменение параметров объектов выполняется с помощью команд меню **Format**, которые описаны в табл. 1.

Обратите внимание на то, что делают следующие команды:

- **To Grid**: изменяет размеры объекта в соответствии с установленной координатной сеткой, если включено **Snap to Grid.**
- **Horisontal Spacing**: позволяет увеличить (**Increase**), уменьшить (**Decrease**) или выровнять (**Make Equal**) промежутки между объектами по ширине страницы отчета.
- **Vertical Spacing**: позволяет увеличить (**Increase**), уменьшить (**Decrease**) или выровнять (**Make Equal**) промежутки между объектами по длине страницы отчета.
- **Bring to Front**: позволяет выдвинуть объект на передний план, если он оказывается закрыт другими объектами.
- **Send to Back**: позволяет отодвинуть объект на задний план, если он закрывает другой объект.

Для ускорения процесса работы с объектами при создании отчета можно пользоваться панелями инструментов, которые вызываются из меню **View**. О двух из них мы уже говорили при описании Конструктора форм. Панель инструментов **Report Controls** по своим функциям похожа на панель инструментов **Form Controls** в Конструкторе форм и предназначена для выбора объектов отчета:

- Метка для размещения поясняющего текста.

- Поле для вывода данных из полей таблиц, выражений, массивов или переменных.
- Вертикальные или горизонтальные линии для отделения отдельных смысловых частей отчета.
- Прямоугольная рамка для выделения информации.
- Рамка со скругленными углами для выделения информации.
- OLE-объект, в качестве которого может использоваться изображение или график, хранящиеся в файле (BMP, ICO), или поле типа General.

Процесс создания отчета в общем случае включает в себя или часть приведенных ниже процедур:

- Определение среды окружения.
- Размещение текста.
- Размещение полей
- Размещение линий, прямоугольников, рисунков.
- Перемещение объектов и областей с любыми объектами, текстами
- Сохранение отчета.

При определении среды окружения открываются требуемые таблицы, выбираются индексы таблиц, устанавливаются отношения между таблицами и вся эта информация сохраняется в файле описания отчета.

Для настройки среды окружения следует открыть окно одним из следующих способов:

- Выполнить команду **View Data → Environment** основного меню.
- Выбрать команду **Data Environment** всплывающего меню.

В появившемся окне диалога **Data Environment** размещаются все таблицы (командой **Add** всплывающего меню либо **Data Environment → Add** основного меню), используемые в отчете. В этом окне можно изменить существующие связи между таблицами и установить новые. После размещения в окне диалога всех таблиц и закрытия окна Visual FoxPro сохранит среду окружения создаваемого отчета.

Наличие среды окружения позволяет также легко использовать при построении отчета специально подготовленные с помощью команды **SELECT-SQL** данные. Достаточно создать представление требуемой структуры и затем просто поместить его в среду окружения **Data Environment**.

Для ввода или исправления текста в отчете выберите кнопку **Label**. Установить курсор в то место окна конструктора отчета, где необходимо разместить текст и зафиксировать его. После этого внести необходимые добавления или изменения.

Если текст состоит из нескольких строк, использовать клавишу **Enter**. Ля изменения цвета и шрифта текста с помощью мыши следует выбрать этот текст и выполнить команду **Format → Font.**

Если дважды нажать мышью на выбранный текст, откроется окно диалога **“Text”,** в котором можно определить условие печати для текста и задать его относительное расположение в полосе с помощью опций:

- **Float** – позиция текста изменяется при изменении размеров окружающих его полей.
- **Fix relative to top of band –** текст сохраняет постоянную позицию относительно верхней границы полосы.
- **Fix relative to bottom of band -** текст сохраняет постоянную позицию относительно нижней границы полосы.

5.3. Добавление заголовка и итогов

Для придания отчету законченного вида можно добавлять в него текст, рисунки, заголовок отчета, а также итоговые данные. Полоса **“Title”**заголовка показывается один раз в начале отчета, а полоса итогов **“Summary”** – один раз в конце отчета. Для добавления этих полос в отчет необходимо выполнить следующее:

1. Выбрать команду Report →Title/Summary.
2. В открывшемся окне диалога “Title/Summary” установить необходимые флажки Title band и Summary band.
3. Нажать ОК. в отчете появятся указанные полосы.

Флажок **New page** используется, если после полосы вы хотите печатать отчет с новой страницы.

Порядок размещения итоговых полей рассмотрен далее в разделе «Размещение полей» (кнопка **Calculate Field** окна диалога **Report Expression**).

При проектировании отчета применятся технологию перетаскивания. Для быстрого размещения полей в отчете можно перетаскивать их как из окна **Data Environment**, так и из Диспетчера проектов. Вы также можете перетаскивать таблицы из Диспетчера проектов в окно **Data Environment.**

Для размещения поля в отчете необходимо выбрать инструмент **Field** на панели инструментов **“Report Controls**”. Установить курсор в

нужное место окна конструктора и установить требуемый размер поля. На экране появится окно диалога “**Report Expression**” (Рис.3).

С помощью этого окна можно:

- установить положение поля в отчете;
- указать условия печати;
- определить выражение, результат вычисления которого будет выводиться в данное поле.

а) поле **Expression.**

Задать поле, которое необходимо поместить в отчет, можно прямо, указав его в поле **Expression** диалогового окна, или вызвав Построитель выражений, нажав кнопку справа от этого поля. Пользуясь Построителем, вы можете построить выражение, выбирая поля из открытых таблиц, размещенные в памяти переменные и стандартные функции Visual FoxPro в диалоговом режиме, а также проверить правильность составленного выражения.

б) поле **Format**. В поле **Format** диалогового окна **Report Expression** можно задать шаблон, который будет определять формат выводимого значения. Для определения формата можно использовать следующие символы:

- A – допускает вывод только символов алфавита.
- L – допускает вывод только логических данных.
- N – допускает вывод только букв и цифр.
- X – допускает вывод любых символов.
- 9 – допускает вывод только цифр для символьных данных и цифр и знаков для числовых данных.
- # - допускает вывод цифр, пробелов и знаков для числовых данных.
- $ - выводит знак доллара в фиксированной позиции перед числом.
- $$ - выводит знак доллара непосредственно перед числом.
- * - выводит знак «звездочка» перед числом для затруднения приписки дополнительных цифр в отпечатанном документе.
- . (точка) – определяет положение десятичной точки.
- , (запятая) – может использоваться для отделения разрядов в числе.

Кнопка справа от поля **Format** поможет вам выбрать подходящий формат в диалоговом режиме. В табл. 2 приведен список опций, которые можно задать в диалоговом режиме для полей каждого типа.

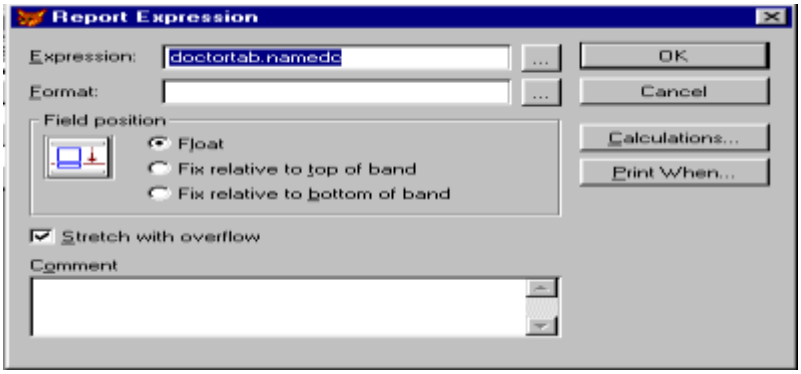


**Рис. 3.** Окно диалога Report Expression

В блоке **Field position** устанавливаются расположение и характеристики поля в полосе, а также условия печати данных, когда сами эти данные могут иметь совершенно различный объем. Включение переключателя **Float** позволяет смещать точку начала печати данных в зависимости от места на странице отчета, в котором была закончена печать данных, расположенных выше. Обычно данные печатаются в пределах того контура, который отводится каждому полю. Если при проектировании отчета вы не уверены точно в объеме данных или для сокращения пустого пространства на странице нежелательно отводить для какого-то поля максимальный объем, то вы можете установить флажок **Stretch with overflow**. Это позволит продолжить печать данных на последующих строках в пределах установленной ширины поля. Вот в этом случае и важно для объектов отчета, расположенных ниже в той же полосе (например, линия для разделения записей), установить переключатель **Float**.

Если флажок **Stretch with overflow** не выбран, то место для печати данных из текущего поля останется фиксированного размера. Все данные из поля, которые не войдут в предназначенное для них пространство, будут отсечены.

Переключатель **Fix relative to top of band** выбирается по умолчанию, что обеспечивает печать данных с начала поля.

Переключатель **Fix relative to bottom of band** позволяет привязать данные к нижней границе поля.

Таблица 2 Форматы для полей в отчете

| **Опция** | **Пояснение** |
|---|---|
| Для редактирования символьных данных | |
| **To Upper Case** | Все символы алфавита преобразуются в прописные буквы |
| **Ignore Input Mask** | Неформатные символы в шаблоне не отображаются при печати |
| Left Justify | Данные печатаются в поле выровненными влево. |
| Right Justify | Данные печатаются в поле выровненными вправо. |
| Center Justify | Данные в поле центрируются. |
| Blank if zero | Пропуск поля, если его значение равно 0. |
| (Negative) | Отрицательные числа будут заключаться в скобки. |
| CR if positive | После положительного числа печатаются две буквы CR-кредит. |
| DB if Negative | После отрицательного числа печатаются две буквы DB-дебет. |
| Leading Zeros | В поле печатаются все ведущие нули. |
| Currency | Отображает данные с добавлением наименования денежной единицы. |
| Scientific | Число печатается в экспоненциальной форме. |
| Для редактирования данных типа дата | |
| Edit "SET" Date | Данные редактируются как дата с учетом текущего формата даты, установленного командой SET DATE. |
| British Date | Данные редактируются по европейскому стандарту даты. |

При нажатии на кнопку **Calculations** на экране отобразится диалоговое окно, с помощью которого можно задать выполнение определенных вычислений, перечень которых приведен ниже.

В верхней части диалогового окна **Calculations** находится раскрывающийся список **Reset**, с помощью которого можно устанавливать момент сброса значения поля в начальное состояние. Имеющиеся в этом списке значения позволяют подсчитывать данные в целом по отчету, по странице, по группе данных или по колонке. Возможные виды вычислений, которые можно выбрать с помощью кнопок выбора в этом диалоговом окне, приведены в табл. 3.

Виды вычислений для полей отчета Таблица 3

| **Опция** | **Пояснение** |
|---|---|
| **Nothing** | Никакие вычисления не будут выполняться над этим полем (по умолчанию). |
| Count | Подсчитывает, сколько раз данное поле печатается в группе, на странице или в отчете, в зависимости от выбора Reset. |
| Sum | Вычисляет сумму значений поля нарастающим итогом. |
| Average | Вычисляет среднеарифметическое (среднее) значение поля в группе, на странице или в отчете. |
| Lowest | Выводит наименьшее значение этого поля для группы, страницы или отчета. |
| Highest | Выводит наибольшее значение поля. |
| Std. Deviation | Возвращает среднее квадратичное отклонение для значений переменной в группе, на странице или в отчете. |
| Variance | Это статистическая характеристика, измеряющая степень отклонения конкретного значения поля от среднего в группе, столбце, странице или отчете. |

Вторая кнопка, которая имеется в диалоговом окне **Report Expression**, - это **Print When**. В диалоговом окне с таким же названием, которое появляется при нажатии на эту кнопку, можно установить условия печати данных.

В блоке **Print Repeated Values** можно выбрать одну из двух кнопок выбора: **Yes** или **No**. Кнопка **Yes** выбрана по умолчанию, в этом случае в отчете печатаются все значения текущего поля. Выбор кнопки **No** подавляет печать повторяющихся значений поля, кроме первого.

Флажки в блоке **Also Print** позволяют устанавливать условия печати при переносе данных на следующий лист или колонку.

Флажок **Remove Line If Blank** позволяет *исключить из отчета пустые строки*, если в расположенных на них полях отсутствуют данные.

С помощью текстового поля **Print Only When Expression is True** вы можете указать условие для печати данных из текущего поля.

С каждым созданным объектом можно связать примечания, которые никак не влияют на отчет, но могут служить напоминанием об объекте

или, например, содержать фрагмент кода, который должен будет потом переписан в прикладную программу.

Управлять процессом печати отчета можно и с помощью специальных событий, которые формируются в начале и в конце печати каждой полосы отчета. Реакцию на эти события можно установить в диалоговом окне, которое возникает при двойном щелчке на полосе, разделяющей разделы отчета в окне Конструктора отчетов (рис. 4).

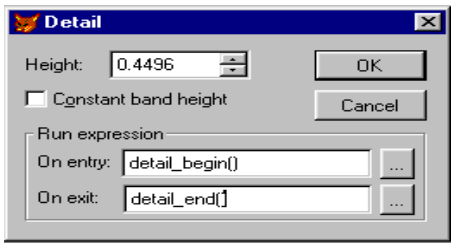


**Рис. 4.** Определение условий выполнения пользовательской функции

Как видно из рис. 4, в начале печати данного раздела отчета (например, Details) будет выполнена пользовательская функция Detail_begin(), а после завершения - Detail_end().

Группировка данных выполняется в том случае, если требуется организовать вложенность данных, например, напечатать список сотрудников по подразделениям. В этом случае сначала печатается список всех сотрудников одного подразделения затем второго, третьего и т. д. Для этого данные о сотрудниках надо сгруппировать по признаку принадлежности каждого сотрудника к тому или иному подразделению. Visual FoxPro будет просматривать все данные и сначала отбирать требуемые записи для первого подразделения, затем для второго и т. д. Нам необходимо заранее позаботиться об активизации соответствующего индекса (в нашем случае по коду подразделения) или подготовить данные в требуемой последовательности с помощью команды SELECT-SQL.

Выполнить группировку данных в отчете для их сортировки по какому-либо признаку позволяет команда **Data Grouping** меню **Report**. Допускается до 20 уровней группировки.

Для правильной группировки данные в таблице должны быть либо отсортированы, либо проиндексированы по признаку группировки.

С каждой группой можно выполнить следующие операции:

- Вычисления над записями внутри заданной группы.
- Печать текста в верхних и нижних колонтитулах для отличия отдельных групп.
- Переход на новую страницу перед началом печати каждой группы.

- Установку номера страницы в начальное состояние при печати групп с новой страницы.

Для создания группы данных необходимо выполнить следующие действия:

- Выбрать команду **Report → Data Grouping**. Появится диалоговое окно, приведенное на рис. 5.

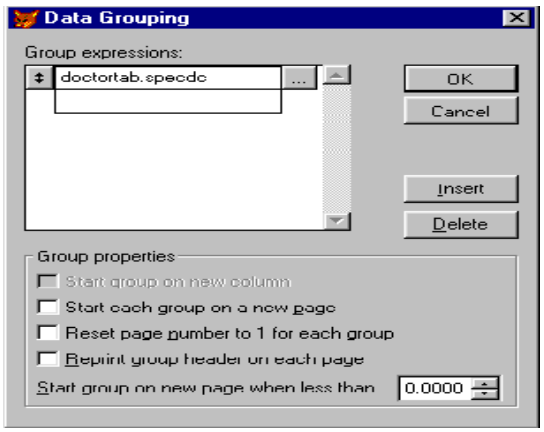


**Рис. 5.** Определение условий группировки данных

- Ввести групповое выражение, которое будет определять признак смены группы, в список **Group expressions**. С помощью расположенной справа кнопки можно вызвать Построитель выражений и сформировать групповое выражение.
- Включить нужные опции в блоке **Group properties** .

Для того чтобы добавить другие группы, повторите этот процесс. Группы перечисляются в списке **Group expressions** в порядке их создания. В окне Конструктора отчетов имена полос групп содержат номера этих групп и усеченные групповые выражения. Верхние и нижние колонтитулы группы с меньшими номерами находятся ближе к полосе **Details**. Группа с меньшим номером основывается на выражении, значение которого в отчете изменяется реже, чем для группы с большим номером. Это значит, что группа с большим номером является подгруппой группы с меньшим номером.

Группировка данных может использоваться и с целью печати определенных данных на отдельных листах, например, при распечатке счетов или накладных, которые, как правило, распечатываются на основе данных, содержащихся в одной записи. В этом случае в качестве признака группировки данных необходимо использовать номер записи, а в окне **Data Grouping** установить флажок **Start each group on a new page**.

С помощью команды **Report →Variables** меню можно определить переменные в отчете, которые будут использоваться в процессе его по-

строения. Переменные удобно использовать для хранения промежуточных результатов вычислений в качестве поля в отчете или как часть выражения. При выборе команды **Variables** на экране отображается диалоговое окно **Report Variables**, с помощью которого можно создать новую переменную, изменить существующую или убрать ее (рис. 6).

Значение переменной при вычислениях будет зависеть от задания диапазона ее действия, т. е. где ее значение будет возвращаться в первоначальное, поэтому обратите внимание на список **Reset at**.

После создания, переменные доступны для составления любых выражений в отчете через Построитель выражений.

Порядок, в котором переменные отображаются в списке **Variables**, может влиять на выходные данные. Переменные вычисляются в порядке их появления в списке. Если одна переменная используется для определения значения другой переменной, то первая переменная должна находиться в списке раньше второй.

С помощью переменных в отчете мы можем решать и довольно нетривиальные задачи. Например, для печати итоговых значений через каждые десять записей мы можем в диалоговом окне **Report Variables** создать переменную. Назовем ее nRec. Выберем в блоке **Calculate** тип выполняемых вычислений — **Count** (подсчет числа значений). В диалоговом окне **Data Grouping** укажем в качестве признака группировки: INT(nRec/10). В окне Конструктора отчетов появятся дополнительные полосы **Group Header** и **Group Footer**, в них мы можем разместить поля, для которых необходимо печатать итоговые значения.

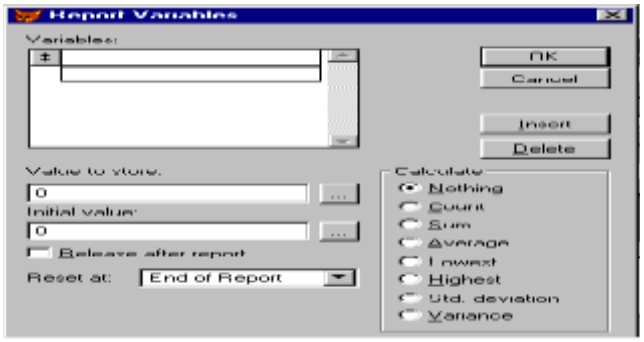


**Рис. 6.** Задание переменной в отчете

Компоновка страницы устанавливается командой **File → Page Setup** меню. Основные элементы диалогового окна **Page Setup** приведены на рис. 7. Обратите внимание, что именно здесь можно установить условия для создания многоколонного отчета.

В области **Columns** устанавливаются значения, определяющие количество колонок на странице и их размеры:

- **Number** - число колонок на странице.
- **Width** – ширина колонок в сантиметрах или дюймах**.**
- **Spacing** – расстояние между колонками**.**
- Переключатель Print area содержит опции**:**
- **Printable page** – поля страницы определены текущим драйвером.
- **Whole page** – режим печати с минимальными полями.

Переключатель **Print order** определяет порядок вывода записей в многоколонных отчетах.

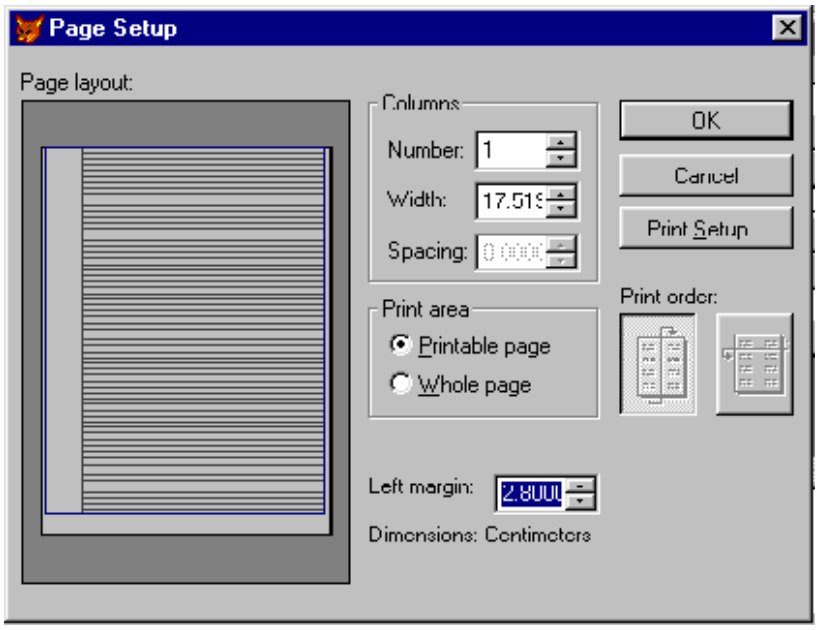


**Рис. 7.** Установка параметров страницы отчета

Счетчик **Left margin** определяет ширину левого поля отчета. Установки, выбранные в диалоговом окне **Page Setup**, сохраняются вместе с файлом отчета. При распечатке отчета на различных принтерах, которые могут оказаться у вашего заказчика, вы можете столкнуться с необходимостью некоторой корректировки полей или размера страницы. Трудно сказать, кто здесь виноват. Для нас в этом случае важно оставить возможность выполнить такую корректировку в готовом приложении. Один из наиболее безболезненных вариантов, это не включать файлы отчетов (FRX и FRT) в исполняемый файл приложения, а поставлять их отдельно, как файлы таблиц. При установке приложения не забудьте посмотреть, как выглядят ваши отчеты, и при необходимости внесите в них коррективы.

В диалоговом окне **Page Setup** вы можете выполнить и настройку принтера, нажав кнопку **Print Setup**, например, выбрать требуемый лоток подачи бумаги.

Когда отчет готов, вы можете проверить результат, выбрав команду **Preview** в меню **View**. Для управления предварительным просмотром ис-

пользуйте специальную панель инструментов **Print Preview**, с помощью которой можно перемещаться по страницам, регулировать масштаб изображения, вывести отчет на принтер или вернуться в режим проектирования.

Наиболее удобно и эффективно создавать отчет по заранее подготовленным данным. Для этого, например, можно использовать представления или запросы. Очевидно, что этот подход позволит вам также в будущем избежать излишних переделок пользовательской программы в связи с переходом на технологию клиент-сервер. Ведь в этом случае не имеет значения, где находятся те данные, которые вы используете для составления отчета. Необходимо будет только внести соответствующие изменения в представление. С другой стороны, в любой пользовательской программе требуется вывод большого числа разнообразных данных. Это грозит непомерным увеличением числа представлений в вашей БД. Решить эту проблему можно, используя параметрические представления. В качестве примера проектирования отчета для пользовательской программы и попробуем создать такой отчет.

В этом случае нам надо будет выполнить следующие шаги:

- Создать параметрическое представление и поместить его в качестве источника данных в окно среды окружения.
- Создать форму для задания параметров.
- Создать отчет, в котором перед его выполнением пользователь сможет ввести необходимые параметры.

В качестве источника данных используем параметрическое представление Forrepv.

Для ускорения создания отчета используем Мастер отчетов. В Диспетчере проектов выберем вкладку **Documents**, перейдем на заголовок **Reports** и дадим команду **New**. В появившемся диалоговом окне выберем кнопку **Report Wizard** и в следующем окне из трех вариантов Мастера отчетов выберем самый простой — **Report Wizard**. На первом шаге Мастера выберем нужное представление. На последнем шаге напишем для отчета заголовок, сохраним его под именем RepPos.FRX и загрузим для дальнейшей модернизации в Конструктор отчетов. Мы увидим отчет в виде, представленном на рис. 8.

Далее произве конечного варианта. Для этого воспользуемся панелью **Report** дем настройку отчета, для доведения его до **Controls** и **Reports Designer**. Выравнивание объекта производиться с помощью панели Layout. Для улучшения вида отчета в верхнем колонтитуле страницы **(Pager Header**) вводим русские названия полей. Для этого нажмите

кнопку **Label** на панели инструментов **Report Controls** и щелкните курсором на тексте. В окончательном виде отчет представлен на рис. 9.

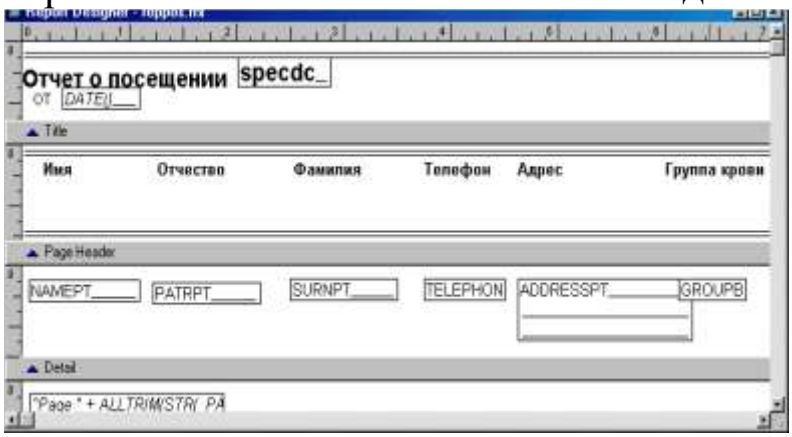


**Рис. 8.** Отчет RepPos, подготовленный мастером отчетов

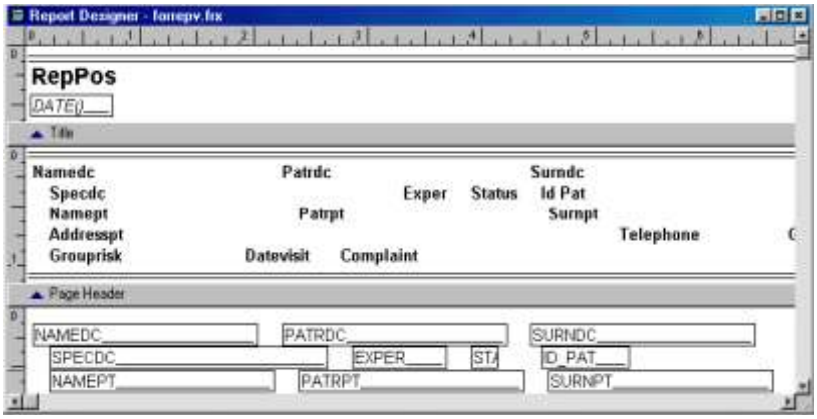


**Рис. 9.** Отчет в RepPos окончательном виде

Полученный отчет можно вывести на экран, в текстовый файл или на принтер с помощью команды:

REPORT FORM *FileName1* | ? [ENVIRONMENT] [*Scope*] [FOR *lExpression1*] [WHILE *lExpression2*] [HEADING *cHeadingText*] [NOEJECT] [NOCONSOLE] [NOOPTIMIZE] [PLAIN] [PREVIEW [[IN] WINDOW *WindowName* | IN SCREEN] [NOWAIT]] [TO PRINTER [PROMPT] | TO FILE *FileName2* [ASCII]] [SUMMARY]

*FileName1* – имя файла с шаблоном отчета.

? – выводит диалоговое окно Open для выбора файла отчета.

FOR *lExpression1*,WHILE *lExpression2 – Expression1, Expression2* определяют условия вывода отчета.

HEADING *cHeadingText* – текст дополнительного заголовка страницы отчета.

NOEJECT – отменяет прогон страницы перед началом печати.

NOCONSOLE – подавляет вывод отчета на экран.

NOOPTIMIZE – подавляет оптимизацию отчета.

PLAIN – заголовок страницы появляется только в начале отчета.

PREVIEW – показывает отчет в режиме предварительного просмотра.

TO PRINTER [PROMPT] | TO FILE *FileName2* – посылает отчет на печать или записывает в файл.

ASCII – создает на основе файла отчета текстовый файл ASCII. В этом файле отсутствуют любые включенные в отчет графические элементы.

SUMMARY – подавляет печать блока данных и выводит только суммы и частичные суммы.

Команда вывода отчета на экран в специально созданное окно будет иметь вид:

DEFINE WINDOW <имя окна> FROM 0,0 TO 150,150 TITLE 'Отчет';

CLOSE FLOAT GROW ZOOM && создаем окно

ACTIVATE WINDOW <имя окна> && активизируем окно

REPORT FORM <имя файла отчета>;

PREVIEW IN window <имя окна>;

nowait

Команда вывода отчета в файл имеет вид:

REPORT FORM <имя файла отчета>;
TO FILE <имя файла>.TXT ASCII

**Порядок выполнения работы**

*Задание 1*

1. Разработать структуру однотабличного отчета, предусмотрев наличие заголовка .

2. Создать стандартный отчет с помощью команды **Quick Report**, заменив заголовки полей отчета на русские наименования.

3. Установите нужный шрифт и цвет заголовка отчета и просмотрите его внешний вид. Сохраните отчет.

4. Разработать структуру двухтабличного отчета с заголовком, итогом по отчету, группировками (1 или 2) данных.

5. Создать двухтабличный отчет с помощью мастера **Report Wizard**, предусмотрев наличие итоговых значений по группировкам и отчету в целом. Просмотреть внешний вид и сохранить отчет.

*Задание 2*

1. Разработать структуру многотабличного отчета с заголовком, группировками данных, наличием рисунков, линий, прямоугольников.

2. Создать многотабличный отчет с помощью конструктора отчетов **Report Disagner** с итогами по группировкам и отчету в целом.

3. Завешить создание отчета компоновкой страницы и просмотром внешнего вида. Сохранить отчет.

**9. Контрольные задания**

1. Создать отчет по таблице *DoctorTab*. Сгруппировать врачей по специализации. Каждую группу специальностей выводить на новой странице. Создать отчет по таблице *DoctorTab*. Разбить врачей на две группы – совместители и штатные сотрудники.

2. Создать отчет по таблице *PatientTab*. Сгруппировать пациентов по группе крови. Уникальный идентификатор не выводить.

3. Создать отчет по таблице *PatientTab*. Сгруппировать пациентов по фамилиям. Уникальный идентификатор не выводить.

4. Создать отчет по таблице *VisitTab*. Отсортировать записи по дате посещения.

5. Создать отчет по таблице *Chekipag*. Сгруппировать сотрудников по специальности. Для каждой группы специальностей вывести сумму выплат, а также суммарную заработную плату всех сотрудников.

6. Создать отчет по таблице *Chekipag*. Вывести Ф. И. О. Сотрудников, специальность и дату поступления на работу. Отсортировать записи по дате поступления на работу.

7. Создать отчет по таблице *Reis*. Сгруппировать записи по аэропорту вылета.

8. Создать отчет по таблице *Fly*. Отсортировать записи по дате вылета. Подсчитать через сколько дней состоится рейс.

9. Создать отчет по таблице *Models*. Отсортировать записи по дальности полета. Подсчитать для каждой модели максимальное количество человек, летящих на этой модели.

10. Создать отчет по таблицам *DoctorTab* и *VisitTab*. Для каждого врача вывести Ф. И. О., специализацию и сведения о посещениях (Ф. И. О. пациента, дата посещения, жалоба). Отсортировать записи по Ф. И. О. врачей.

11. Создать отчет по таблицам *DoctorTab* и *VisitTab*. Для каждого пациента вывести Ф. И. О. и сведения о посещении врачей (Ф. И. О. врача, специализация, дата посещения, жалоба). Отсортировать записи по Ф. И. О. пациентов.

12. Создать отчет по таблицам *Chekipag* и *Polek*. Для каждого члена экипажа вывести все сведения о нем, а также номера рейсов на которых он летит, дату полета и через сколько дней вылетает рейс.

13. Создать отчет по таблицам *Models* и *Samolets*. Вывести сведения о модели и имеющихся самолетах данной модели. Отсортировать записи по дальности полета.

14. Создать отчет по таблицам *Models* и *Samolets*. Вывести сведения о модели и имеющихся самолетах данной модели. Отсортировать записи по количеству мест.

15. Создать отчет по таблицам *Reis* и *Fly*. Вывести сведения о рейсе и о совершаемых каждым рейсом полетах. Отсортировать записи по времени вылета.

**Контрольные вопросы**

1. Как вызвать мастер **Report Wizard** построения отчета?
2. Перечислить шаги создания отчета с помощью мастера.
3. Сколько группировок данных можно создать в отчете с помощью мастера?
4. Как задаются итоговые значения по группировкам данных в отчете при использовании мастера **Report Wizard?**

5. Пояснить порядок создания быстрого стандартного отчета с помощью команды **Quick Report.**

6. Перечислить возможные типы колонок окна конструктора отчетов **Report Disagner.**

7. Как ввести полосы заголовка и итогов в отчет?

8. Как установить среду окружения и в отчете?

9. Как установить положение поля в отчете?

10. Как составить выражение, результат которого выводится в поле?

11. Каков порядок ввода текста и задания его цвета и шрифта в отчете?

12. Каково назначение поля **Format** окна **Report Expression**?

13. Как разметить страницу отчета?

14. Как удалить в отчете пустые строки и подавить печать повторяющихся значений?

## ЛАБОРАТОРНАЯ РАБОТА №6.

# ВЫБОРКА ДАННЫХ В КОНСТРУКТОРЕ ЗАПРОСОВ

**Цель работы**: Изучение функциональных возможностей конструктора запросов и приобретение практических навыков создания многотабличных запросов с помощью средств СУБД.

**Краткие теоретические сведения.**

Одним из соновных назначений законченного приложения является быстрый поиск информации в базе данных и получение ответов на разнообразные вопросы. Вопросы, формулируемые по отношению к БД называются *запросами*. Для формирования запросов используются мастер и конструктор запросов, а также команда **SELECT** языка СУБД.

Для создания простейших запросов можно использовать мастер запросов **Query Wizard**, однако его возможности ограничены и далее будет рассмотрено интерактивное средство для выбора данных из одной или нескольких таблиц – конструктор запросов **Query Designer**. Результатом выборки всегда является таблица, которую можно сохранить в массиве, во вновь создаваемой таблице, отобразить на экране в виде окна **Browse** или вывести в виде отчета. При создании запроса с помощью конструктора вместо того, чтобы печатать предложения на специальном языке, достаточно просто заполнить форму запроса, которая располагается в окне конструктора запросов.

Для запуска конструктора запросов необходимо выполнить следующее.

1. Выполнить команду **File → New** и установить переключатель в положение **Query** (Запрос).
2. Запустить конструктор запросов **Query Designer**, щелкнув на кнопке **New File .**
3. Появится диалоговое окно **Add Table or View** для выбора таблиц, в котором содержатся имена таблиц для запроса (Рис.1).
4. Кнопкой **Other** можно открыть любую БД и выбрать таблицу из любого каталога. Выбрать из списка **Table in database** и перенести их кнопкой **Add** в окно конструктора запросов.
5. По завершению выбора таблиц нажать **ОК** и на экране появится окно **Query Designer**, а в основном меню пункт **Query** (Рис.2).

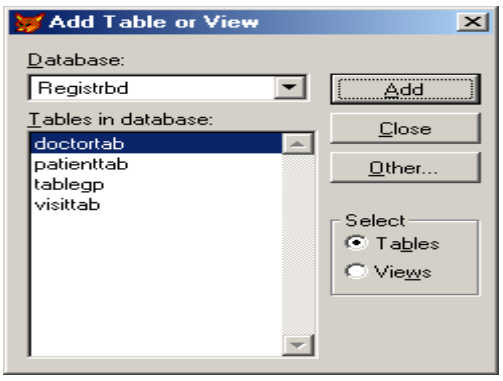


**Рис. 1.** Окно выбора таблиц

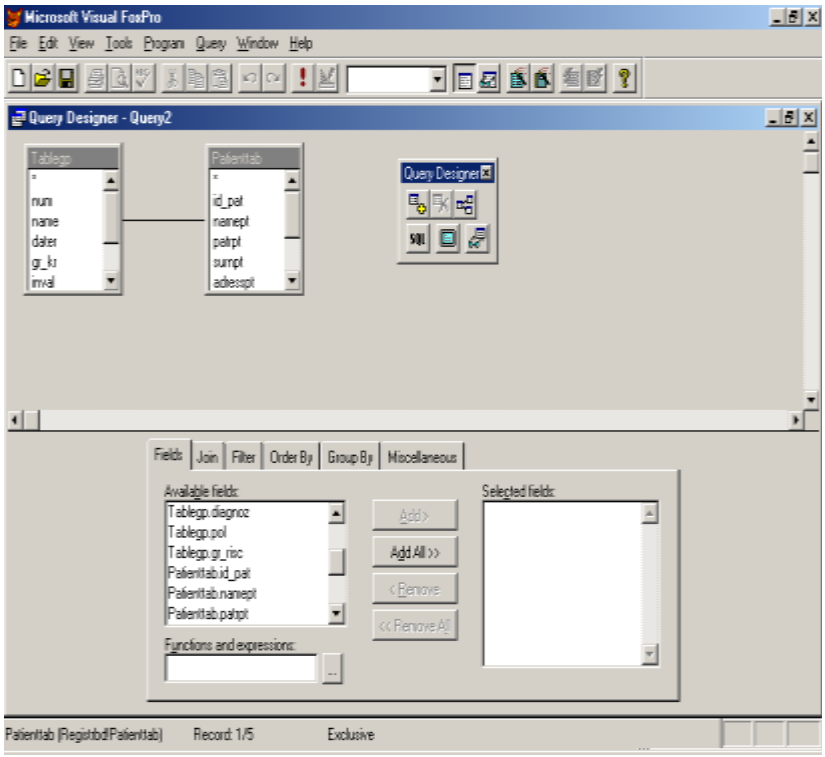


**Рис. 2.** Окно конструктора запросов

В последующих разделах рассмотрен порядок выбора полей результата запроса, указания критерия для выборки, группировки и упорядочения данных, а также варианты возможного вывода результатов выборки. Для этой цели в окне конструктора запросов предусмотрены специальные вкладки:

- **Fields** – поля исходных таблиц, выбираемые в исходную таблицу.
- **Join** – условия объединения таблиц.

- **Filter** – фильтры для выбора полей.
- **Order By** – критерий упорядочения.
- **Group By** – условие группировки.
- **Miscellaneous** – признак выборки повторяющихся значений, процент выбора данных.

Для формирования запроса можно пользоваться опциями меню **Query** и панелью инструментов **Query Designer**, которые выполняют функции по выбору перечисленных вкладок, работы с таблицами запроса, открытия окон диалога.

Заметим, что при создании многотабличного запроса в окно конструктора запросов таблицы добавляются с постоянными существующими связями между ними. В тех случаях, когда между таблицами не существует постоянной связи, с помощью диалогового окна **Join Condition** пользователь задает условие объединения таблиц путем указания общего поля для связи.

В результирующей таблице поля формируются на основе полей исходных таблиц и вычисляемых полей.

В Visual FoxPro существует несколько вариантов отбора полей результирующей таблицы:

- Открыть вкладку **Fields**, выделить в списке **Aviable fields** нужные поля и перенести их в список **Selected fields** с помощью кнопки **Add**. Кнопка **Add All** позволяет перенести сразу все поля таблицы.
- Нажать кнопку мыши на выделенных для переноса полях и , не отпуская ее перенести выбранные поля в список **Selected fields (**механизм **grand and grop –** *перенести и оставить***).** Для переноса всех полей таблицы выбрать строку **«*».**
- Из списка **Aviable fields** выбрать поле и дважды нажать кнопку мыши для перенесения его в список **Selected fields.**

***Порядок полей*** в списке **Selected fields** определяет порядок их появления в результирующей таблице. Для изменения расположения поля следует установить курсор слева от поля и переместить его в требуемое место.

Для ***просмотра*** полученного варианта запроса можно пользоваться одним из способов:

- Нажать кнопук **Run** на стандартной панели инструментов.
- Выполнить команду **Run Query** всплывающего меню.
- Выполнить команду **Query → Run Query** основного меню.

Результаты выборки находятся в режиме **BROWSE** (Рис.3). С помощью мыши можно изменить ширину и расположение столбцов, режим просмотра изменяется выбором опций **Browse/Edit** в меню **View.**

При выполнении запроса можно вычислять значения по одному или нескольким полям одной таблицы, а также объединять несколько полей исходной таблицы в одно поле.

Чтобы включить в запрос функцию поля или выражение, на вкладке **Fields** в поле **Function and expression** необходимо ввести выражение для вычисления или вызвать для этого построитель выражений **Expression Builder**. Нажать кнопку **Add** в окне конструктора для переноса выражения в список выходных полей запроса.

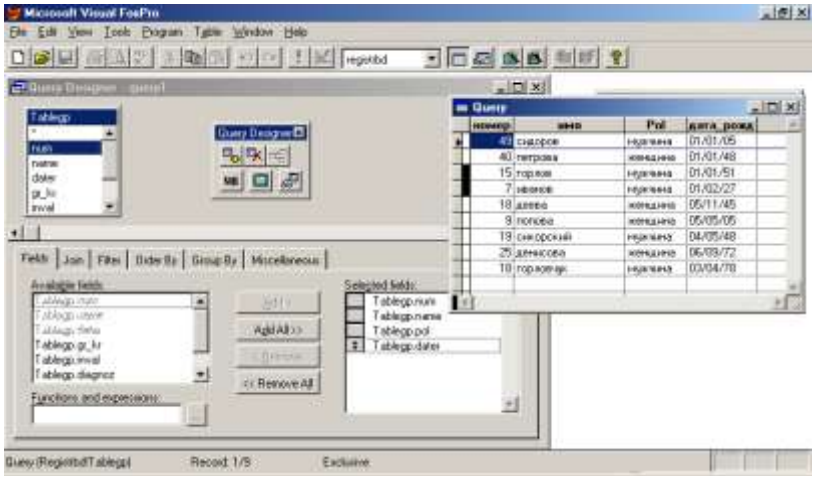

**Рис. 3.** Результат выборки по запросу

Для задания критериев выбора записей для вывода следует выбрать вкладку **Filter** окна конструктора. Во вкладке содержится вся информациядля ввода критерия отбора записей. Для задания условия выбра записей необходимо:

- выбрать поле **Field Name** во вкладке **Filter** и из раскрывающегося списка полей таблиц выбрать нужное поле;
- установить флажок опции **Not** для отрицания выражения отбора записей;
- щелкнуть кнопкой справа от раскрывающегося списка **Criteria** и выбрать нужный оператор для сопоставления значения поля со значением поля **Example**;
- в поле **Example** указать значение, используемое в качестве условия запроса;

- при необходимости различения заглавных и прописных букв нужно щелкнуть на кнопке **Case** (с учетом регистра);

раскрыть список **Logical** и выбрать нужный пункт (<none>, AND, OR) в зависимости от структуры логического выражения фильтра (<none> - для одного условия; AND, OR - для нескольких связанных условий ) и завершить построение фильтра из нескольких условий, используя кнопку **Insert**. На рис. 4. представлен результат выборки данных с заданными критериями отбора данных во вкладке **Filter** окна конструктора.

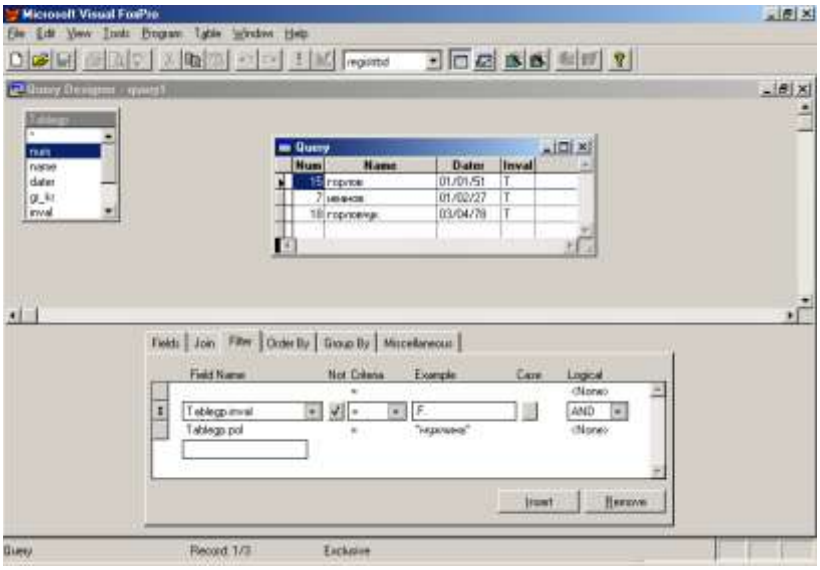


**Рис. 4.** Пример выборки данных по заданному критерию отбора

Заметим, что при составлении выражений для фильтров удобно пользоваться операторами **Between** для задания диапазона значений и **In** для задания списка возможных значений.

Вкладка **Order By** (Рис.5) управляет порядком расопложения записей в результирующей таблице. Для этого необходимо выделить курсором поля, определяющие порядок сортировки данных, и перенести их последовательно в список **Ordering criteria**.

Для каждого выбранного поля можно установить с помощью переключателя **Order** option критерий упорядочения по возрастанию **(Ascending**) или по убыванию **(Descending**).

Порядок сортировки записей результирующей таблицы определяется порядком следования полей в списке **Ordering criteria.** Для изменения порядка следования следует установить курсор на маркер перемещения слева от поля и переместить его в требуемое место.

При упорядочении по нескольким полям для каждого их полей критерий упорядочения устанавливается отдельно.

**Рис. 5.** Пример выборки упорядоченных данных

Группировка позволяет получить вычисляемую информацию о подгруппах таблицы. Чтобы сгруппировать записи в запросе по каким либо полям таблицы, следует выбрать вкладку **Group By** и перенести в список **Grouped fields** поля, по которым осуществляется группировка даннаых.

Строки таблицы можно группировать по любой комбинации ее полей. Каждое из уникальных значенийполя, по которому выполняется группировка, порождает отдельную группу данных.

При выполнении запроса можно вычислять итоговые значения по одному или нескольким полям исходной таблицы. Вычисление итоговых значений над группой записей выполняется с использованием следующих функций:

- **COUNT()** – количество строк в итоговой таблице;
- **MAX** – наибольшее значение в столбце;
- **MIN –** наименьшее значение в столбце;
- **AVG()** – среднее значение столбца численных данных;
- **SUM()** – сумма численных данных столбца.

Для включения в запрос функции или выражения необходимо:

- открыть вкладку **Fields** и нажать кнопку построителя выражения справа от поля **Function and expression**;

- в поле ввода **Expression** открывшегося окна диалога **Expression Builder** сформировать нужное выражение;

- после выхода из окна диалога **Expression Builder** нажать кнопку Add дял размещения выражения в списке **Selected fields.**

В рассмотренных выше примерах результат выборки выводился в стандартную результирующую таблицу в режиме **Browse.** Для изменения направления вывода необходимо нажать кнопку **Query Distination** на панели инструменотов **Query Designer** и в окне диалога **Query Distination** (Рис.6) выбрать одну из следующих кнопок:

- Cursor – временное хранение результатов в виде, доступном только для чтения;
- Browse – сохранение и вывод результатов в стандартной таблице в режиме Browse;
- Table – указание таблицы для хранения результатов;
- Report – отображение результатов выборки в виде отчета;
- Lable - отображение результатов выборки в виде этикетки;
- Screen - отображение результатов выборки в активном окне;
- Graph – запуск Microsoft Graf для создания графиков и диаграмм.

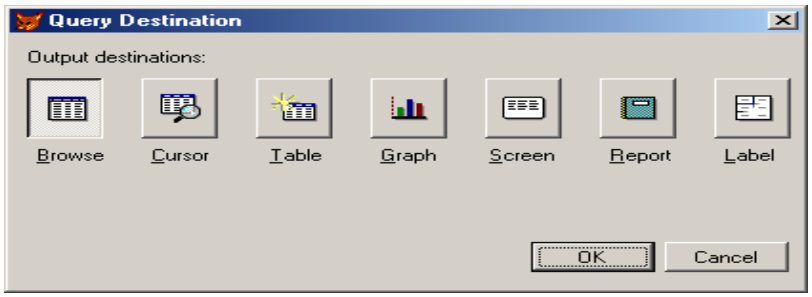


**Рис. 6.** Окно выбора направления вывода результатов

Заметим, что при необходимости конструктор запросов позволяет *изменить названия полей* результирующей таблицы**,** чтобы сделать ее более читабельной. Для изменения наименования полей список выбранных полей **Selected fields** формируется с помощью поля **Function and expression**, в котором за именем поля добавляется ключевое слово **As** и указывается новое имя. Так, результат выборки, представленный на рис.7, повторяет запрос рис.4 с новыми наименованиями полей.

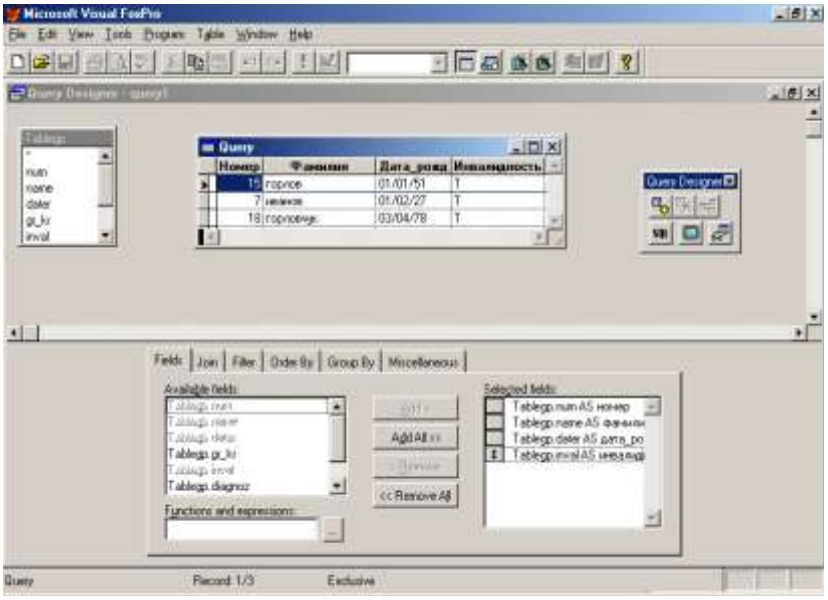

**Рис. 7.** Результаты выборки в таблицу с новыми наименованиями полей

Для формирования запросов в программах на языке Visual FoxPro используется команда **SELECT**, аналогичная конструкции **SELECT** языка запросов высокого уровня SQL.

Упрощенный синтакис команды **SELECT** имеет вид:

**SELECT [DISTINCT] список Выбираемых Полей**
**FROM список Таблиц**
**[WHERE условие Выборки]**
**[GROUP BY Условие Группировки]**
**[ORDER BY условие Упорядочения]**
**[INTO имя Таблицы]**
**[TO FILE имяФайла [ADDITIVE] | [TO PRINT]]**

Опция **DISTINCT** используется для исключения повторяющихся строк запроса.

В качестве иллюстрации приведем примеры команды **SELECT** для выборок, представленных на рис.3,4,5 соответственно.

```
SELECT Tablegp.num, Tablegp.name, Tablegp.dater, Tablegp.inval;
 FROM registrbd!tablegp
SELECT Tablegp.num, Tablegp.name, Tablegp.dater, Tablegp.inval;
 FROM registrbd!tablegp;
 WHERE Tablegp.inval <> .F.;
   AND Tablegp.pol = "мужчина"
SELECT Tablegp.num, Tablegp.name, Tablegp.dater, Tablegp.inval;
 FROM registrbd!tablegp;
 ORDER BY Tablegp.num
```

Оператор **SQL**, эквивалентный созданному запросу, можно просмотреть, нажав кнопку **Show the SQL window** на панели инструментов **Query Designer**. При программировании законченных приложений команда **SELECT** широко используется для формирования разнообразных ведомостей. В связи с этим при создании запросов целесообразно сохранять **SQL**-операторы в качестве заготовок для будущих программ.

При использовании стандартных СУБД достаточно часто, если не всегда, пользователь сам определяет критерий выборки, по которому осуществляется поиск. В этом разделе остановимся на рассмотрении данного случая.

Для примера предположим, что пользователю необходимо осуществить поиск всех больных по определенной группе риска. Во избежание возникновения ошибок, которые могут возникнуть при вводе данных с клавиатуры, поместим выбираемые данные из группы риска во всплывающий список, который разместим на форме диалога (рис. 8).

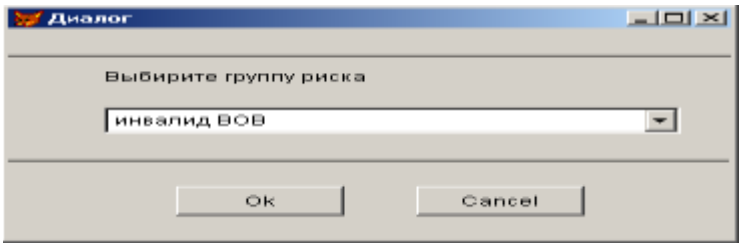


**Рис. 8.** Форма диалога при выборке по текстовому значению

При нажатии кнопки «**Ok**» – выполняется следующий код:

```
SELECT Patienttab.namept, Patienttab.patrpt, Patienttab.surnpt,;
 Patienttab.addresspt, Patienttab.telephone, Patienttab.groupb;
FROM patienttab;
where thisform.CombGrR.value=PATIENTTAB.GROUPRISK
```

Как Вы можете видеть это простой SQL-запрос, но в блоке условия выборки “**where**” – мы пишем, то значение, которое ввел пользователь (*PATIENTTAB.GROUPRISK*).

### 10.2. Выборка по числовому значению

Рассмотрим пример поиска по числовому значению. В окне диалога, рис. 9, пользователем вводится необходимое число, например группа крови пациента, и при нажатии кнопки **Ok** выполняется следующий запрос:

SELECT PatientTab.*;

FROM PatientTab;

Where PatientTab.GroupB= VAL(ThisForm.nGr.Text)

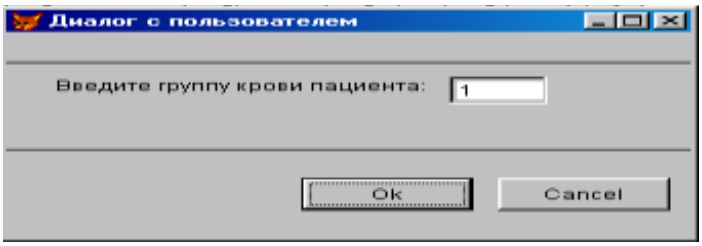


**Рис. 9.** Форма диалога при выборке по числовому значению

Рассмотрим пример поиска по дате. В окне диалога рис. 10 вводится необходимая дата (дата приема пациента) и при нажатии кнопки «**Ok**» выполняется запрос:

MessageBox(thisform.DatT.Value)

SELECT View2.*;

FROM View2;

Where View2.datevisit = CTOD(thisform.DatT.Text)

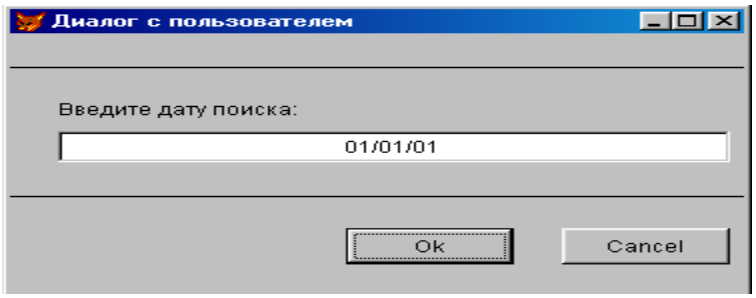


**Рис. 10.** Форма диалога при выборке по дате

10.4. Выборка по интервалу дат

Рассмотрим пример поиска по числовому значению. В окне диалога, рис. 11, пользователем вводится необходимый интервал дат и при нажатии кнопки **Ok** выполняется следующий запрос:

SELECT View2.*;

FROM View2;

Where View2.datevisit >= CTOD(thisform.TDatB.Text);

and View2.datevisit <= CTOD(thisform.TDatE.Text)

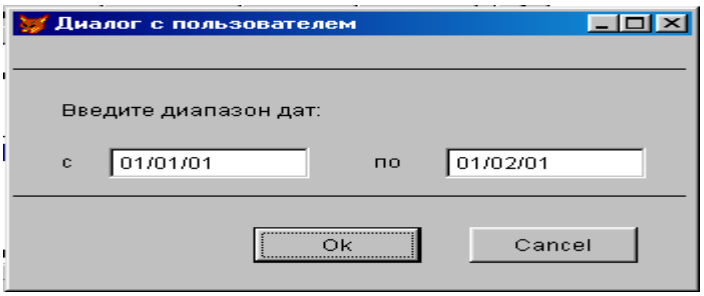


**Рис. 11.** Форма диалога при выборке по интервалу дат

Рассмотрим пример поиска по нескольким параметрам. В окне диалога, рис. 12, пользователем вводятся несколько параметров, например группа риска пациента и дата приема, и при нажатии кнопки **Ok** выполняется следующий запрос:

```
SELECT Patienttab.namept, Patienttab.patrpt, Patienttab.surnpt,;
Patienttab.addresspt, Patienttab.telephone, Patienttab.groupb;
FROM patienttab;
where thisform.CombGrR.value= Patienttab.grouprisk
and View2.datevisit = CTOD(thisform.DatT.Text)
```

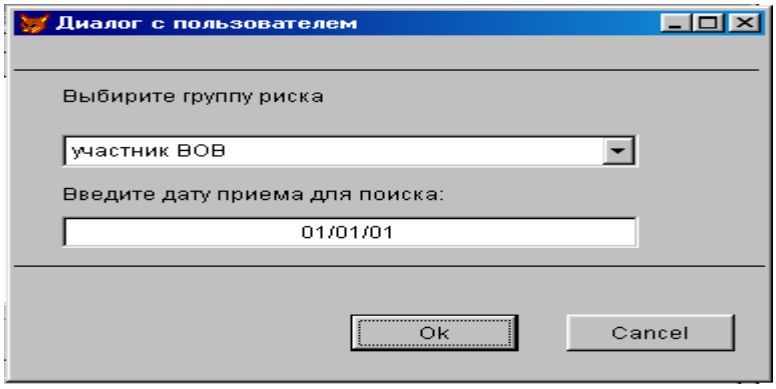


**Рис. 12.** Форма диалога при выборке по нескольким параметрам

Пусть из таблицы **Patienttab** требуется выбрать служащих, удовлетворяющих слудующим условиям:

1. Фамилия начинается с определенных символов;
2. Они имеют определенную группу риска;
3. Дата рождения, начиная с определенного числа.

При этом может быть выбрано одно, два или три условия для формирования запроса.

Для формирования данного запроса создадим форму, представленную на рис. 13.

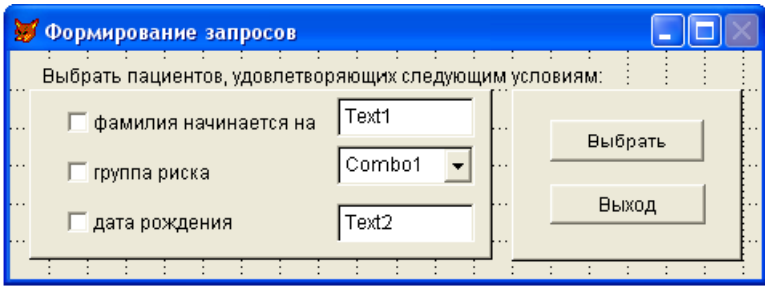


**Рис. 13.** Форма для формирования запроса

Выбор условий формирования запроса определяется тремя переключателями **Check1, Check2, Check3**. В компонент **Text1** вводятся начальные символы фамилии. Из выпадающего списка **Combo1** выбирается профессия служащих, для этого в свойстве **RowSourceType** выбрано значение *1-Value*, а свойству **RowSource** присвоено значение *тюард,пи- лот,борттехник*. Компонент **Text2** предназначен для ввода даты поступления на работу. Текст запроса записан в методе Click кнопки *Выбрать*:

```
public fz,risk,dat                    && объявление переменных
if ThisForm.Check1.Value=1            && прверка выбора первого условия
fz=RTRIM(ThisForm.Text1.Text)  && значение нач. букв фамилии если первое условие выбрано
else fz="  "  && если первое условие не выбрано (начало фамилии с любого символа)
endif
if ThisForm.Check2.Value=1 && прверка выбора второго условия
risk=RTRIM(ThisForm.Combo1.Text)      && выбранная профессия если второе условие выбрано
else
risk="          "     && если второе условие не выбрано (любая профессия)
endif
```

if ThisForm.Check3.Value=1 *&& прверка выбора третьего условия*

dat=ctod(RTRIM(ThisForm.Text2.Text)) *&& дата поступления на работу если третье условие выбрано*

else

dat={^1800-01-01} *&&если третье условие не выбрано (дата, заведомо меньшая любой даты из таблицы)*

endif

SELECT patienttab.surnpt, patienttab.namept, patienttab.patrpt, patienttab.datar, patienttab.grouprisk;

From RegistryBD!patienttab;

WHERE patienttab.surnpt LIKE fz+"%" AND patienttab.grouprisk=risk and patienttab.datar between dat and date();

order by surnpt;

**Пусть в таблицу** PatientTab **занесены следующие данные (рис. 14).**

Patienttab

| Id_pa | Namept | Patrpt | Surnpt | Grouprisk | Datar | Groupb |
|---|---|---|---|---|---|---|
| 49 | Иван | Петрович | Сидоров | нет | / / | |
| 40 | Валентина | Ивановна | Петрова | астма | / / | |
| 15 | Виктор | Петрович | Горлов | инвалид | 01/01/50 | |
| 7 | Владимир | Михайлович | Иванов | инвалид | 02/02/04 | |
| 18 | Галина | Ивановна | Деева | онкология | / / | |
| 9 | Елена | Николаевна | Попова | нет | / / | |
| 19 | Петр | Михайлович | Сикорский | инвалид | / / | |
| 25 | Наталья | Владимировна | Денисова | туберкулез | / / | |
| 17 | Евгений | Михайлович | Горловчук | инвалид | 10/04/99 | |

**Рис. 14.** Содержание таблицы **PatientTab**

Пусть требуется выбрать всех инвалидов, родившихся с 1 января 1950 года и чья фамилия начинается на букву Г (рис. 15).

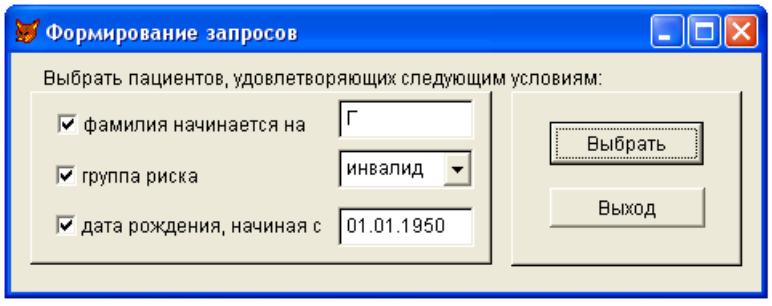


**Рис. 15.** Выбор условий для запроса

Тогда результат работы будет иметь следующий вид (рис. 16):

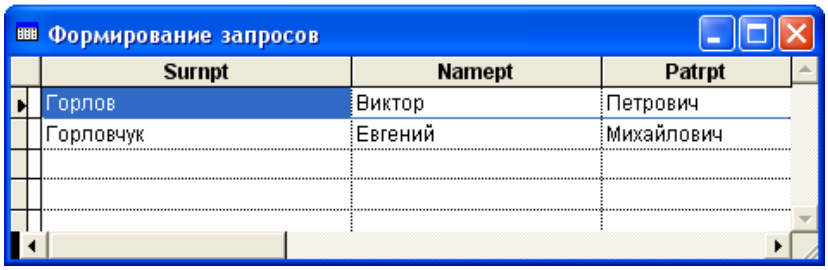


**Рис. 16.** Результат выбора

**Можно выводить результат запроса непосредственно на форму, используя объект** Grid**. Для этого необходимо добавить объект** Grid**, его свойству** RecordSousceType **присвоить значение 0 (Table). Результат выполнения запроса будет записываться в итоговую таблицу Rez, которая будет выводиться в** Grid**, при этом в текст программы необходимо внести следующие изменения (подчеркнутые строки):**

public fz,risk,dat *&& объявление переменных*

if ThisForm.Check1.Value=1 *&& прверка выбора первого условия*

fz=RTRIM(ThisForm.Text1.Text) *&& значение нач. букв фамилии если первое условие выбрано*

else fz=" " *&& если первое условие не выбрано (начало фамилии с любого символа)*

endif

if ThisForm.Check2.Value=1 *&& прверка выбора второго условия*

risk=RTRIM(ThisForm.Combo1.Text) *&& выбранная профессия если второе условие выбрано*

else

risk=" " *&& если второе условие не выбрано (любая профессия)*

endif

if ThisForm.Check3.Value=1 *&& прверка выбора третьего условия*

dat=ctod(RTRIM(ThisForm.Text2.Text)) *&& дата поступления на работу если третье условие выбрано*

else

dat={^1800-01-01} *&&если третье условие не выбрано (дата, заведомо меньшая любой даты из таблицы)*

endif

SELECT patienttab.surnpt, patienttab.namept, patienttab.patrpt, patienttab.datar, patienttab.grouprisk;

From RegistryBD!patienttab;

WHERE patienttab.surnpt LIKE fz+"%" AND patienttab.grouprisk=risk and patienttab.datar between dat and date();

**order by surnpt;**

**into table Rez && запись результата в итоговую таблицу Rez**

ThisForm.Grid1.RecordSource="Rez.dbf" *&& вывод итоговой таблицы Rez в объекте Grid*

Тогда результат выполнения приведенного выше примера будет выведен в объекте Grid (рис. 17).

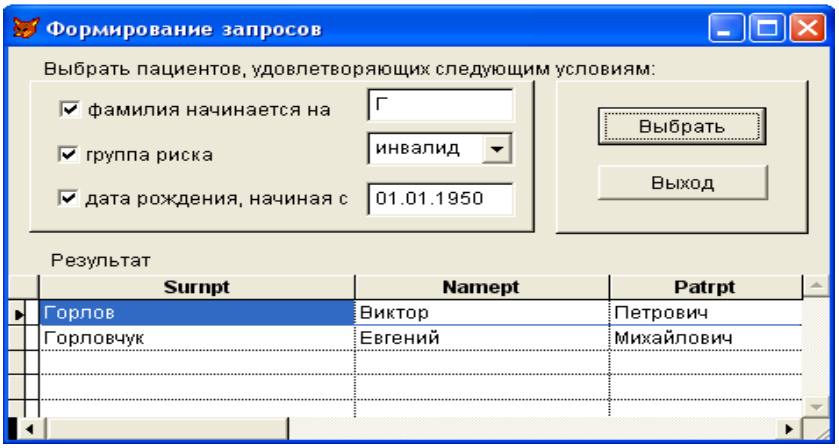


**Рис. 17.** Модифицированная форма

## Порядок выполнения работы

6. Разработать запросы для выборки данных из нескольких таблиц, предусмотрев при их создании следующее:

a. наличие вычисляемых полей;

b. упорядочение данных по различным полям и их комбинациям;

c. фильтрацию выбираемых данных по совпадению (несовпадению) значений полей, по диапазону значений, по списку значений;

d. вычисления итоговых значений.

7. Сохранить SQL-операторы для созданных запросов;

8. Просмотреть результаты запросов в режиме **Browse** и выполнить их вывод в таблицу, в отчет, на принтер и в файл.

**Контрольные задания**

1. Из таблицы DoctorTab выбрать всех хирургов со стажем работы не менее 5 лет.

2. Из таблицы DoctorTab выбрать штатных сотрудников, чья фамилия начинается на «А» или «Б».

3. Из таблицы PatientTab выбрать пациентов с четвертой группой крови и имеющих группу риска.

4. Из таблиц DoctorTab и VisitTab вывести Ф. И. О. и специализацию врачей, принимавших пациента с идентификатором 0000234.

5. Из таблиц DoctorTab и VisitTab вывести Ф. И. О. и специализацию врачей, принимавших пациентов 12.10.2001.

6. Из таблиц PatientTab и VisitTab вывести Ф. И. О. пациентов, посетивших врачей на последней недели.

7. Из таблиц PatientTab и VisitTab вывести Ф. И. О. пациентов, посетивших хирургов в прошедшем году.

8. Из таблицы Chekipag выбрать всех пилотов со стажем работы не менее 5 лет.

9. Из таблицы Chekipag выбрать всех пилотов со стажем работы не менее 5 лет.

10. Из таблицы Chekipag выбрать всех борттехников с окладом менее 200.

11. Из таблицы Reis выбрать все рейсы, с продолжительностью полета более двух часов.

12. Из таблицы Fly выбрать все полеты рейса ВК341 за последние полгода.

13. Из таблицы Models выбрать модели самолетов, вмещающих свыше 100 пассажиров и имеющих дальность полета не менее 500 км.

14. Из таблиц Models и Samolets вывести бортовые номера самолетов модели ТУ-154.

15.Из таблиц Chekipag и Polek выбрать всех служащих, летящих на рейсе ВК341 12.11.2002 года (вывести Ф. И. О., специальность сотрудников).

**Контрольные вопросы**

15. Как вызвать конструктор запросов **Query Designer**?

16. Пояснить назначение вкладок окна **Query Designer** .

17. Как выбираются исходные таблицы и поля для результирующей таблицы запроса?

18.Каков порядок составления выражений для вычисляемых полей?

19.Как используется вкладка **Order By** окна конструктора запросов?

20. Перечислить операторы списка **Criteria** во вкладке **Filter** и пояснить их назначение.

21. Как задать условия для выбора записей в результирующую таблицу ?

22. Как выполнить вычисления итоговых значений в запросе?

23. Как изменить наименования полей в результирующей таблице?

24. Перечислить возможные варианты вывода результатов запроса.

25. Пояснить структуру конструкции **SELECT**.

# ЛАБОРАТОРНАЯ РАБОТА №7
# ПРОЕКТИРОВАНИЕ СИСТЕМЫ МЕНЮ В СУБД

**Цель работы:** Изучение способов организации многоуровневых меню, получение практических навыков проектирования горизонтальных и вертикальных меню в среде СУБД Visual FoxPro.

**Краткие теоретические сведения:**

Основной формой диалогового интерфейса в прикладных системах обработки данных является многоуровневое меню.В соответствии со стандартами Windows в любом приложении рекомендуется иметь строку меню, которая в Visual FoxPro содержит команды, предназначенные для вызова форм, формирования отчетов, запросов и т. д.

При разработке приложения вы можете создать все требуемые объекты (базу данных, входящие в нее таблицы, формы, отчеты, запросы). Затем объединить отдельные объекты с помощью меню. Можно поступить иначе. Сначала разработать и создать меню, а затем по мере создания форм и отчетов включать их запуск в меню. Второй способ более нагляден. Вы в любой момент можете запустить меню и продемонстрировать заказчику, как создаваемая система выглядит, как осуществляется вызов тех или иных программ, опустить уже созданные формы, напечатать подготовленные отчеты. Рассмотрим этапы разработки меню. На начальном этапе разработки необходимо определить требования, предъявляемые к создаваемому приложению, и информацию, которая будет содержаться в проектируемой базе данных.После этого определяется структура таблиц и совпадающие поля для их связывания. Затем создаются сами таблицы, входящие в базу данных, определяются отношения между ними. Одновременно вы должны определить те средства, которые получит в свое распоряжение пользователь при работе с приложением.

Приложение должно содержать эффективную справочную систему, содержащую информацию о приложении, описание его основных функций и инструкцию по работе. В среде Windows предпочтительнее всего создавать справочную систему в принятом стандарте, чтобы пользователю было легко искать информацию в знакомой ему среде.

После того как определена структура данных, спроектированы таблицы, входящие в базу данных, можно приступать к разра-

ботке структуры меню. Прежде чем описывать структуру меню в конструкторе, нарисуйте эскиз меню на бумаге.

Строкой меню называется горизонтальное меню (**Menu**), располагаемое в верхней части экрана. Примерами меню являются основное меню Visual FoxPro и меню программ, работающих в среде Windows. Созданное вами меню в конструкторе может замещать основное меню Visual FoxPro или добавляться к нему. Вертикальным (**Shortcut**) называется меню, пункты которого следуют по вертикали один под другим.

Для создания меню необходимо выполнить следующие действия:

1. Открыть окно конструктора меню.
2. Описать вид меню, текст, пункты меню и его атрибуты.
3. Определить действия, которые будут выполняться при выборе пунктов меню.
4. Сгенерировать меню, используя команду Generate (Генерация) из меню Menu. При этом создается программа, которую вы в результате и запускаете на выполнение.

### 3.1. Запуск конструктора меню

Для открытия окна конструктора меню воспользуйтесь одним из следующих способов:

- В меню **File** (Файл) выберите команду **New** (Новый). В открывшемся диалоговом окне **New** установите опцию **Menu** (Меню) и нажмите кнопку **New File** (Новый файл)
- В окне проекта перейдите на вкладку **Other** (Остальные) и выберите группу **Menus.** Затем нажмите кнопку **New** окна проекта
- Находясь в группе **Menus** окна проекта, нажмите кнопку **New** на стандартной панели инструментов Visual FoxPro. В открывшемся диалоговом окне **New** установите опцию **Menu** и нажмите кнопку **New File**

На экране открывается диалоговое окно **New Menu** (Новое меню), в котором предлагается два варианта меню (рис. 1):

- **Menu –** меню в виде строки
- **Shortcut –** всплывающее меню, в котором основные пункты расположены по вертикали.


**Рис. 1.** Диалоговое окно **New Menu**

Выберите тип создаваемого меню, нажав соответствующую кнопку. В результате запускается конструктор меню, а в основное меню Visual FoxPro убавляется новый пункт **Menu** (рис. 2). Для создания меню в виде строки необходимо выбрать опцию **Menu.**

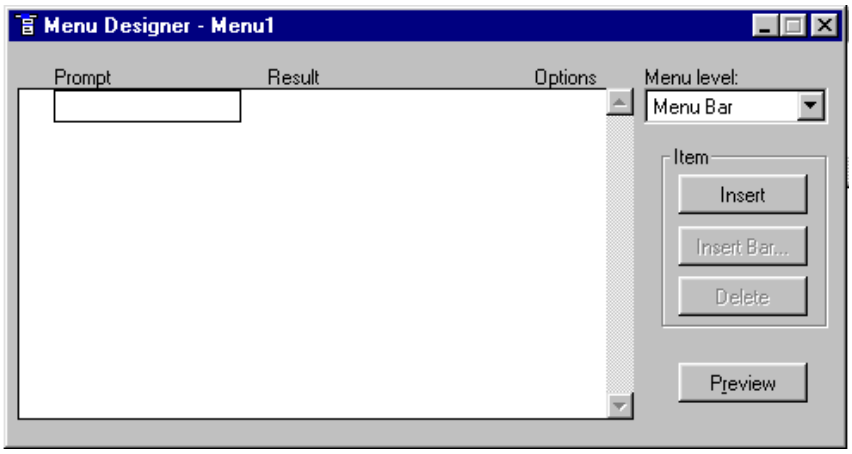


**Рис. 2.** Конструктор меню

Область конструктора меню, над которой размещены надписи **Prompt** (Приглашение), **Result** (Результат) и **Options** (Опции), предназначена для формирования меню.

В поле **Prompt** можно ввести наименования пунктов меню. Раскрывающийся список **Result** используется для указания типа пункта меню. Кнопка **Options** открывает диалоговое окно **Prompt Options** (Опции элемента меню), в котором можно определить дополнительные параметры данного элемента меню ("горячие" клавиши, сообщение, отображаемое в строке состояния при выборе пункта меню, и т. д.). В списке **Menu level** (Уровень меню) указывается уровень текущего меню.

Слева от конструктора меню размещены кнопки:

Таблица 1

| *Кнопка* | Назначение |
|---|---|
| **Insert** (Вставить) | Добавляет в меню новый пункт |
| **Insert Bar** (Вставить команды системного меню) | Открывает диалоговое окно **Insert System Menu Bar**, содержащее команды системного меню Visual FoxPro, позволяя разместить их в создаваемом пользовательском меню |
| **Delete** (Удалить) | Удаляет текущий пункт меню |
| *Preview* (Просмотр) | Размещает создаваемое меню на экране, позволяя просмотреть его внешний вид |

После того как инициализировался конструктор, можно приступать к созданию меню. Для этого выполните следующие действия:

1. В поле **Prompt** (Приглашение) введите наименования первого пункта меню и нажмите клавишу <Enter> или <Tab> для перехода на следующее поле. Курсор оказывается в списке **Result .**
2. Для определения типа пункта меню нажмите кнопку раскрытия списка и выберите необходимое значение из тех, которые предлагает система:

Таблица 2

| Тип меню | **Назначение** |
|---|---|
| **Command** (Команда) | При выборе пункта меню данного типа будет выполняться связанная с ним команда |
| **Pad Name** (Наименование сторки меню) | При выборе пункта меню никаких действии выполняться не будет. Как правило, используется в качестве дополнительного пояснения к меню |
| **Submenu** (Подменю) | При выборе пункта меню раскрывается связанное с данным пунктом ниспадающее меню |
| **Procedure** (Прцедура) | При выборе пункта меню вызывается процедура, определенная для данного пункта меню |

3. Указав тип пункта меню, перейдите в следующую строку и введите информацию о втором пункте меню.

4. Введите наименование остальных пунктов меню и их типы (рис. 3).

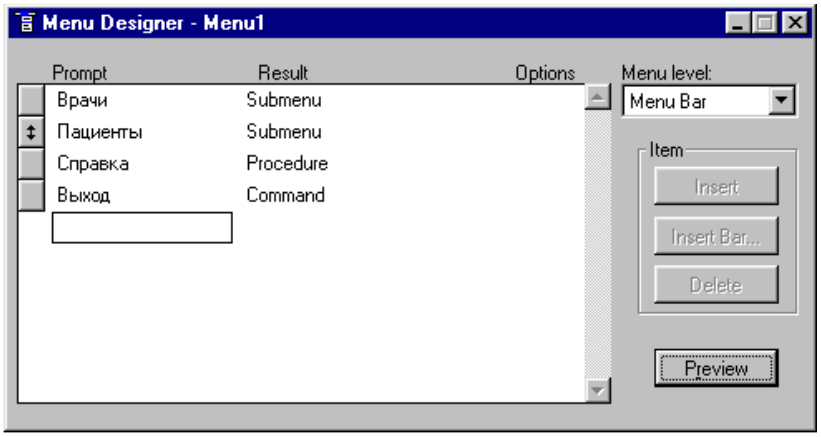


**Рис. 3.** Основное меню приложения

5. Для просмотра созданных пунктов меню нажмите кнопку Preview (Просмотр). Основное меню Visual FoxPro будет заменено созданным меню. Пункты меню отображаются на экране в порядке их описания. На экране также появляется диалоговое окно **Preview,** в котором отображаться текст текущего пункта меню, его тип и выполняемое действие.

Для выбора пункта меню используются клавиши-стрелки или мышь. Помимо этого вы можете определить для пункта меню "горячую" клавишу, нажатие которой вместе с клавишей <Alt> активизирует этот пункт.

Для создания "горячей" клавиши, предоставляющей пользователю возможность ускоренного выбора пункта меню, необходимо включить в его имя перед активизирующим символом следующие символы \<. В качестве "горячей" клавиши можно использовать первый символ имени пункта, что, конечно, более предпочтительно, а также и любой другой символ. Символы, используемые в качестве "горячей" клавиши, выделяются в строке меню подчеркиванием.

**Примечание**

*"Горячие" клавиши для пунктов меню назначаются Visual FoxPro по умолчанию. Для их создания используются первые буквы элементов строки меню. Если два элемента меню начинаются с одинаковой буквы, то обоим элементам строки меню в качестве "горячей" клавиши назначается одинаковый символ. В этом случае вам нужно переопределить "горячую" клавишу для одного из элементов строки меню.*

Для облегчения назначения "горячих" клавиш можно использовать следующий прием: перед именами пунктов меню разместить цифры и их использовать в качестве "горячих" клавиш (рис. 4).

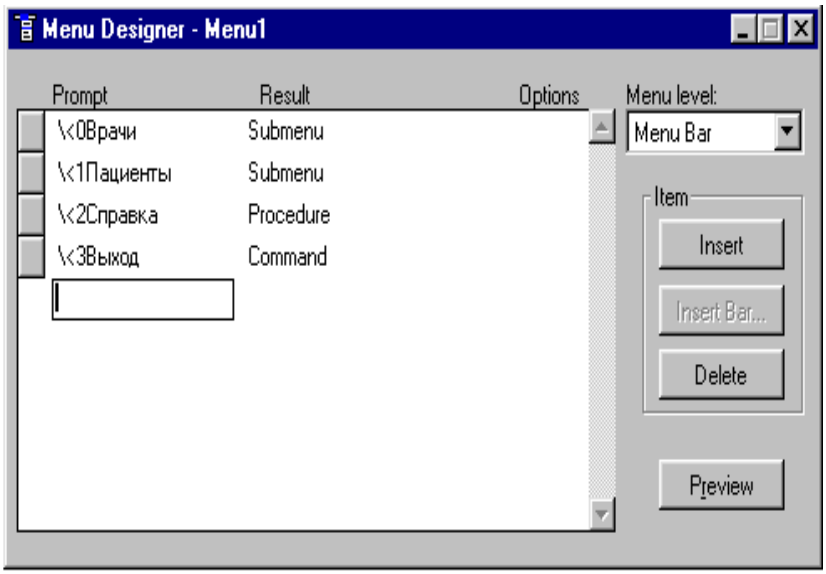


**Рис. 4**. Меню с назначенными “горячими” клавишами

Для задания параметров пункта меню используется диалоговое окно **Prompt Options** (Опции элемента меню) (рис. 5). Чтобы его открыть, установите в конструкторе курсор на пункт меню и нажмите появившуюся в строке кнопку **Options** (Опции).

В диалоговом окне **Prompt Options** содержится область **Shortcut** для задания клавиш быстрого вызова, а также перечисленные ниже поля ввода.

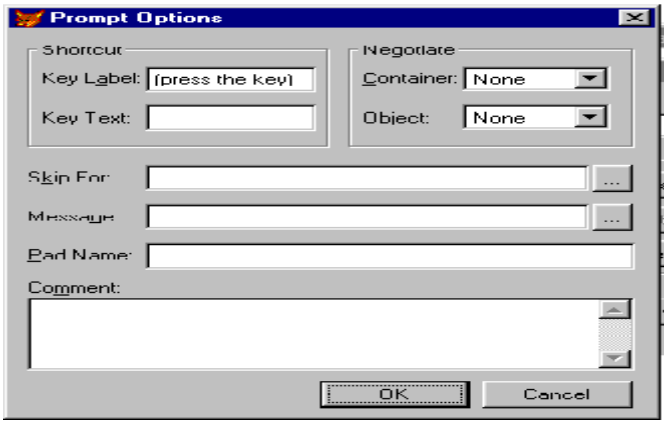


**Рис. 5.** Диалоговое окно Prompt Options

Таблица 3

| **Поле ввода** | Назначение |
|---|---|
| **Skip For** (Пропустить для) | Позволяет заблокировать пункт меню |
| **Message** (Сообщение) | Позволяет задать сообщение, которое будет отображаться |
| **Pad Name** (Имя пункта меню) | Задает имя пункта меню |
| **Comment** (Комментарий) | Задает комментарий к пункту меню |

Область **Negotiate** (Соглашение) содержит два раскрывающихся списка, имеющих следующее назначение:

- **Container** (Контейнер) — определяет расположение меню при редактировании по месту OLE-объектов
- **Object** (Объект) — задает расположение меню при выполнении приложения типа Active Document в Web-браузере

3.5. Действия для пунктов меню и подменю

Основное назначение пункта меню – выполнять определенное действие, Результат выбора пункта меню задается его типом.

Таблица 4

| **Тип пункта меню** | **Действие** |
|---|---|
| **Submenu** (Подменю) | Раскрывается связанное с данным пунктом меню ниспадающее подменю |
| **Procedure** (Процедура) | Выполняется процедура, определенная в конструкторе меню |
| **Command** (Команда) | Выполняется команда, расположенная в поле рядом с типом пункта меню |

Большинство команд меню создаваемого приложения открывает подменю. Например, пункт меню **Врачи** должен содержать команды **Ввод данных**, **Отчеты** и **Запросы**. Для создания данного подменю выполните следующие действия:

1. Нажмите кнопку **Create** (Создать) пункта меню **Врачи.** На экране появляется пустое окно конструктора меню. Список **Menu Level** (Уровень меню) нового окна содержит метку текущего пункта меню.

2. Введите в поле **Prompt** (Приглашение) первой строки команду ***Ввод данных***. Используя список **Result** (Результат), задайте тип созданного пункта меню

3. Во второй строке введите команду ***Отчеты*** и также задайте тип.

4. Аналогичным образом введите третью команду (рис. 6).

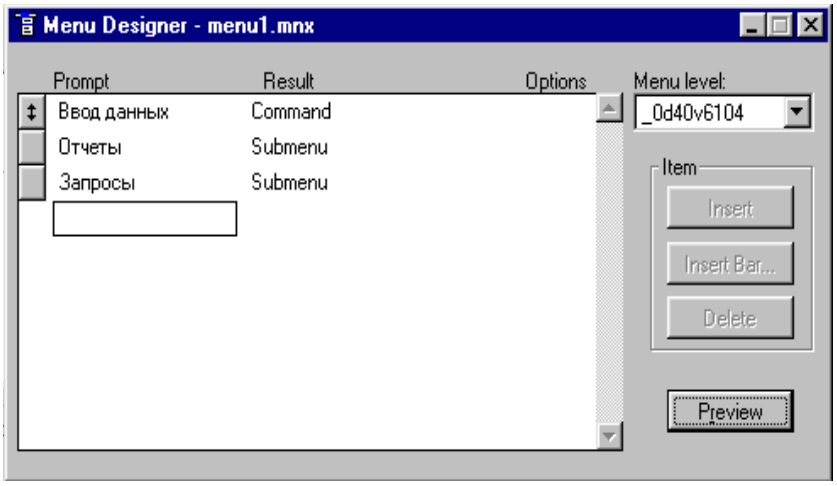


**Рис. 6.** Окно создания подменю

Чтобы определить команду, которая будет выполняться при выборе пункта меню или подменю, необходимо выполнить следующие действия:

1. В столбце **Result** (Результат) выбрать из списка значение **Command** (Команда).

2. Ввести в поле, расположенное справа от списка, команду Visual FoxPro, которая будет выполняться при выборе данного пункта меню.

Обычно команды используются для вызова форм, отчетов, пользовательских процедур. Например, для ввода данных врачей необходимо ввести следующую команду:

DO FORM Doctortab

В том случае, если при выборе пункта меню или подменю должна выполняться некоторая последовательность команд, необходимо в столбце **Result** (Результат) окна конструктора выбрать для пункта меню или подменю тип **Procedure** (Процедура) и определить требуемые команды. Для этого нажмите кнопку **Create** (Создать) и в открывшемся окне определите фрагмент кода, связанный с элементом строки меню. Таким кодом может быть программа некоторого запроса, используемая в качестве заготовки.

Для улучшения внешнего вида, а также для объединения в группы схожих по смыслу команд можно использовать разделительные линии.

Разделительные линии представляют собой пункт меню, в котором в поле ввода **Prompt** (Приглашение) вместо наименования пункта вводятся символы \-. Рассмотрим создание разделительных линий в подменю Врачи. Для этого выполните следующие действия:

1. Откройте файл меню в окне конструктора проекта.
2. Перейдите в режим редактирования пункта меню **Врачи**, нажав кнопку **Create** (Создать).
3. Вставьте новый элемент меню после пункта меню **Ввод данных.** Для этого установите курсор на пункт **Отчеты** и нажмите кнопку **Insert** (Вставить).
4. В поле **Prompt** образованного подпункта меню **New Item** введите \-
5. Аналогично вставьте разделительную линию после пункта меню **Отчеты.**
6. Нажмите кнопку **Preview** (Просмотр). Выберите пункт меню **Врачи**.

На экране появится подменю с разделительными линиями.

Для сохранения созданного меню выберите команду **Save as** (Сохранить как) в меню **File** (Файл). В открывшемся диалоговом окне **Save as** из списка **Папка** выберите папку, в которой вы предполагаете сохранить файл, откройте ее, в поле **Save Menu** (Сохранить меню) введите имя сохраняемого меню. В заключение нажмите кнопку **Сохранить.**

С помощью кнопки **Preview** (Просмотр) окна конструктора меню можно просмотреть внешний вид создаваемого меню, но нельзя его активизировать.

Чтобы можно было использовать меню в приложениях, его необходимо предварительно сгенерировать. Для этого выполните следующие действия:

1. В меню **Menu** выберите команду **Generate** (Генерация). Откроется диалоговое окно **Generate Menu** (Генерация меню) (рис. 7).

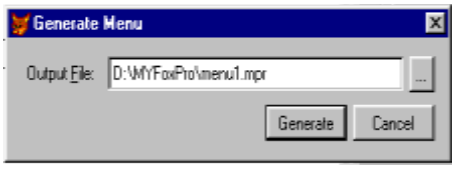


**Рис. 7.** Диалоговое окно **Generate Menu**

2. В поле **Output File** введите имя файла, который будет создан в результате генерации.

3. Для запуска генерации описания меню нажмите кнопку Generate (Генерация).

После завершения генерации можно запустить программу меню на выполнение. Для этого выполните одно из следующих действий:

- В окне проекта установите курсор на наименование созданного меню и нажмите кнопку **Run** (Запустить)
- В меню **Program** (Программа) выберите команду **Do** (Выполнить). В открывшемся диалоговом окне **Do** откройте папку, в котором вы сохранили файл меню, выберите файл с расширением MPR и нажмите кнопку Do

На экране появится созданное вами меню, которое заменит основное меню Visual FoxPro, если в диалоговом окне **General Options** перед генерацией была установлена опция **Replace** (Замещать).

В СУБД имеется возможность создания всплывающего меню средствами конструктора меню.

Способ создания меню данного типа аналогичен созданию горизонтального меню. Для этого меню, как и для обычного, в виде строки, можно определить оперативные клавиши и опции, устанавливаемые в диалоговом окне **Prompt Options** (Опции элемента меню).

Чтобы создать всплывающее меню, выполните следующую последовательность действий:

1. Откройте проект.
2. Для открытия окна конструктора меню в окне проекта перейдите на вкладку **Other** (Остальные) и выберите группу **Menus** (Меню).
3. Нажмите кнопку **New** окна проекта.
4. В открывшемся диалоговом окне **New Menu** (Новое меню) нажмите кнопку **Shortcut** (Всплывающее меню). Откроется окно конструктора меню.
5. В поле **Prompt** (Приглашение) последовательно введите тексты пунктов меню и определите для них выполняемые действия.
6. Для генерации выберите команду **Generate** (Генерация) в меню **Menu**.
7. Запустите меню на выполнение. Вид данного меню при запуске представлен на рис. 8.

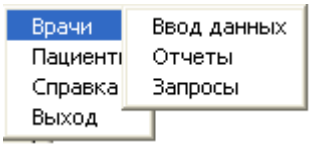


**Рис. 8.** Меню типа **Shortcut**

В процессе проектирования программного приложения может возникнуть вопрос: “Как поместить только что созданное меню в окне своей программы, а не вместо основного меню Visual FoxPro?” Для осуществления задуманного необходимо выполнить следующие действия:

1. В меню **View** строки основного меню Visual FoxPro выбрать **General Options**.

2. В окне General Options установить флажок **Top-Level-Form** (рис. 9).

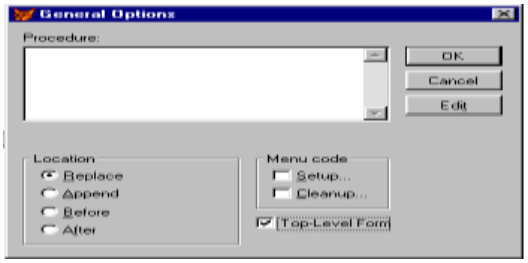


**Рис. 9.** Окно **General Options**

3. Свойство ShowWindow главной формы программы установить в значение 2 – As Top-Level Form (рис. 10).

4. В процедуре **Init** главного окна программы написать следующий код:

**DO** <название меню> **With** This, .F.

В нашем случае, например, следует написать:

DO menu1.mpr With This, .F.

В итоге должно получиться то, что изображено на рис. 11.

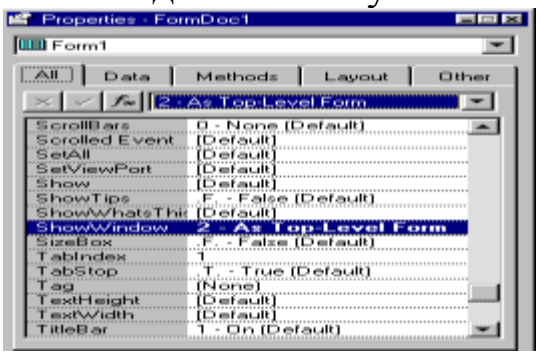


**Рис. 10.** Изменение свойства ShowWindow главной формы программы

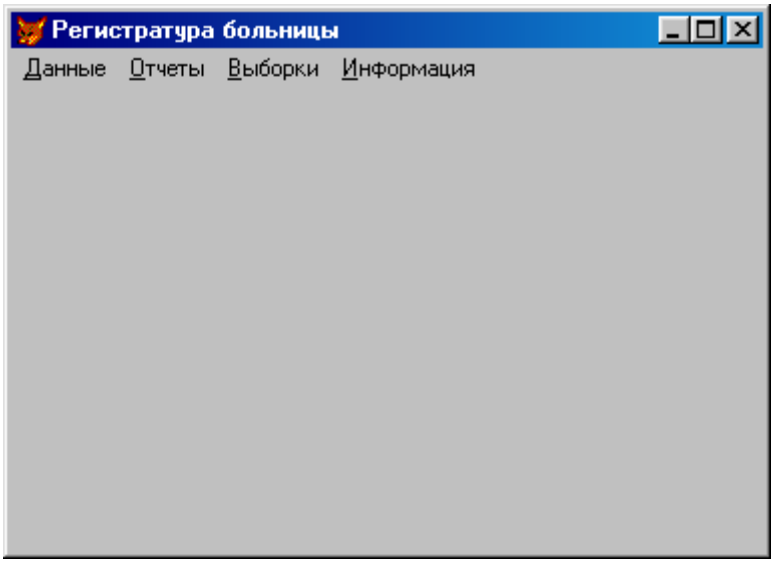


**Рис. 11.** Главное окно программы

Меню любой сложности строится из меню двух типов: горизонтального и вертикального (всплывающего).

Горизонтальное меню, примером которого является основное меню БД, состоит из нескольких горизонтально расположенных пунктов, которые называются **PAD**-пунктами (например, пункты **File, Edit, View**…).

Всплывающее меню – **POPUP** – состоит из нескольких вертикально расположенных пунктов, которые называются **BAR** и появляются только при активизации соответствующего **PAD**-пункта (например, пункты **New, Open,Save**… пункта **File** основного меню).

**POPUP**-меню может использоваться как в составе меню более высокого уровня, так и самостоятельно.

Главное или **POPUP**-меню определяется и активизируется следующим образом:

DEFINE POP / POPUP <имя меню>
<определение составляющих меню (PAD, POPUP, BAR)>
<описание реакции составляющих меню на выбор>
ACTIVATE MENU < имя меню >

Синтаксис команды создания строки меню следующий:

DEFINE MENU <имя меню>
[BAR [AT LINE <номер строки, в которой появится меню>]]
[IN WINDOW <имя окна, определенного заранее>]
[FONT <имя шрифта>]
[STYLE <имя стиля>]
[KEY <имя клавиши, используемой для вызова меню>]
[MESSAGE<сообщение, которое выводится внизу экрана>]

[COLOR SCHEME <список цветовых пар>]

[SCROLL]

Меню с опцией **BAR** имеет свойство системного меню. Опция [SCROLL] используется, если меню не умещается по ширине экрана.

В зависимости от сложности меню проектируется на нескольких уровнях (Menu level). На верхнем уровне определяются независимые меню, к которым относится **MENU** и всплывающее меню **POPUP.**

Пункты **PAD**, составляющие **MENU**, а также пункты **BAR**, составляющие всплывающее меню **POPUP**, зависимы от соответствующего независимого меню. Поэтому для их определения в команде используется конструкция OF <имя независимого меню, включающего этот пункт>.

**PAD**-пункты задаются следующей командой:

DEFINE PAD <имя PAD-пункты> OF <имя BAR-меню, включающего этот PAD>

PROMPT <название PAD-пункта >

[AT <X,Y>]

[BEFORE < имя пункта >/AFTER < имя пункта >]

* определяет положение данного пункта относительно уже включенных

[MESSAGE<текст сообщения>]

[MARK <символ, расположенный слева от пункта меню>]

[KEY <имя клавиши, используемой для вызова меню>]

[SKIP [FOR <вржL, определяющее условие блокировки>]]

[COLOR SCHEME <список цветовых пар>]

Следует различать имя пункта меню и его название (заголовок). Имя задает программный объект и записываться оно должно только латинскими буквами, в то время как заголовок может быть любым.

Например, в следующей команде имя пункта меню – **padname**, а его заголовок задается символьной строкой ‘\<Информация’. При этом пункт может быть активизирован нажатием клавиши “И”, поскольку она назначена клавишей быстрого доступа:

DEFINE PAD padname OF mainmenu PROMPT ‘\< Информация ’ AT 1, 8

DEFINE POPUP <имя всплывающего меню >

[FROM < X1,Y1 > TO < X2,Y2 >]

[PROMPT <вржC> / PROMPT FIELD <врж> / PROMPT FILES [LIKE <шаблон>] / PROMPT STRUCTURE]

[IN WINDOW < имя окна, определенного заранее >]

[KEY <имя клавиши, используемой для вызова меню>]

[FOOTER <>] [TITLE <>]

* заголовки соответственно в нижней и верхней строках всплывающего меню

[MARK < символ, расположенный слева от пункта меню >]

[MESSAGE<текст сообщения>]

[MULTISELECT]

[SHADOW]

* для выделения более темным цветом

[SCROLL]

[COLOR SCHEME <список цветовых пар>]

Опция PROMPT задает заголовки пунктов меню.

При использовании PROMPT FIELD <врж> элементами меню станут значения поля открытой таблицы, если <врж> - имя поля. В общем случае содержит несколько полей, в том числе и из других открытых таблиц.

Например, пункты всплывающего меню **Familii** можно назвать фамилиями пациентов из поля **SurnPt** таблицы **PatientTab.dbf** базы данных регистратуры (рис. 12).

```
USE patienttab
DEFINE POPUP Familii FROM 1,7 TO 15,30;
PROMPT FIELD surnpt
ACTIVATE POPUP Familii
```

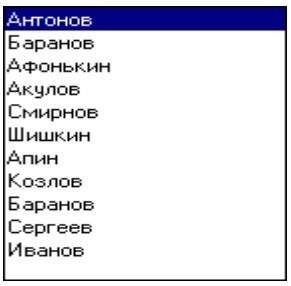


**Рис. 12.** Меню для выбора фамилии

Пункты всплывающего меню могут состоять из нескольких полей разного типа: например, фамилии и имени пациентов из поля **SurnPt** и **NamePt** и даты рождения **Datar** таблицы **PatientTab.dbf** базы данных регистратуры (рис. 13):

```
USE patienttab
DEFINE POPUP Patienti FROM 1,7 TO 15,50;
PROMPT FIELD surnpt + namept + DTOC(datar)
ACTIVATE POPUP Patienti
```

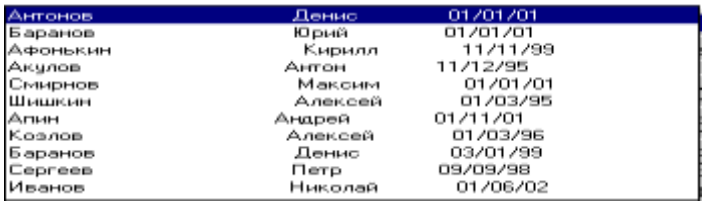


**Рис. 13.** Меню для выбора пациентов

Или специализации врача **SpecDc** и стажа работы **Exper** таблицы **DoctorTab.dbf** базы данных регистратуры (рис. 14):

```
USE DoctorTab
DEFINE POPUP DoctorTab FROM 1,7 TO 5,20;
PROMPT FIELD Specdc + STR(exper)
ACTIVATE POPUP DoctorTab
```

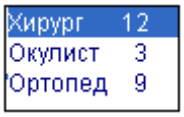


**Рис. 14.** Меню для выбора стажа работы врача

При использовании PROMPT FILES [LIKE <шаблон>] пункты будут называться именами файлов, отобранных в соответствии с шаблоном (рис. 15):

```
DEFINE POPUP FileDBF FROM 1,7 TO 20,50;
PROMPT FILES LIKE *.dbf
ACTIVATE POPUP FileDBF
```

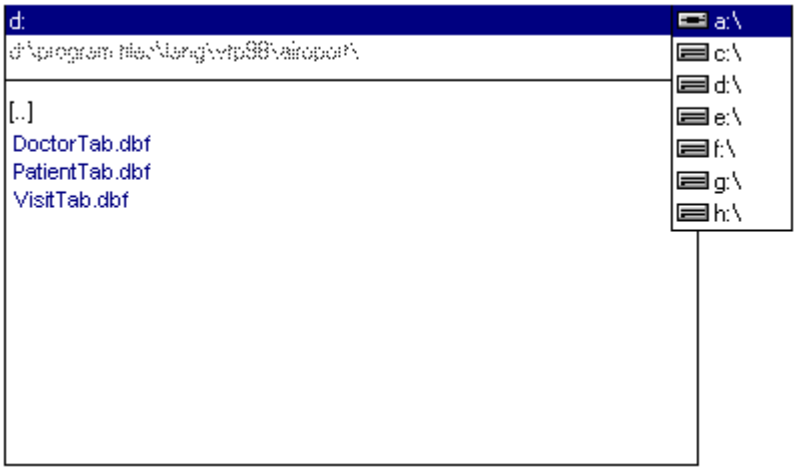


**Рис. 15.** Меню с именами файлов формата dbf

А использование опции PROMPT STRUCTURE обеспечит выбор имен полей открытой таблицы (рис. 16):

```
USE doctortab
DEFINE POPUP patienti FROM 1,7 TO 10,20;
PROMPT STRUCTURE
ACTIVATE POPUP patienti
```

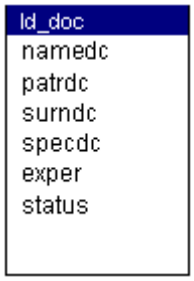


**Рис. 16.** Меню с именами полей открытой таблицы

Для взаимного перемещения пунктов всплывающего меню в команде определения можно использовать опцию MOVER. Для создания, например, всплывающего меню из четырех пунктов с возможностью их перемещения (рис. 17) используйте следующие строки:

```
CLEAR
DEFINE POPUP popDemo MOVER FROM 2,2
DEFINE BAR 1 OF popDemo PROMPT ‘Первый’
DEFINE BAR 2 OF popDemo PROMPT ‘Второй’
DEFINE BAR 3 OF popDemo PROMPT ‘Третий’
DEFINE BAR 4 OF popDemo PROMPT ‘Четвертый’
ACTIVATE POPUP popDemo
```

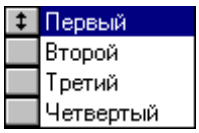


**Рис. 17.** Всплывающее меню с возможностью их перемещения

**DEFINE BAR <номер пункта> OF <имя POPUP-меню> PROMPT <вржС>**
**[BEFORE < имя пункта >/AFTER < имя пункта >]**
**[KEY <имя клавиши, используемой для вызова меню>]**
**[MARK < символ, расположенный слева от пункта меню >]**
**[MESSAGE<текст сообщения>]**
**[SKIP [FOR <вржL, определяющее условие блокировки>]]**
**[COLOR SCHEME <список цветовых пар>]**

Пусть требуется создать меню, со структурой, представленной на рис. 18.

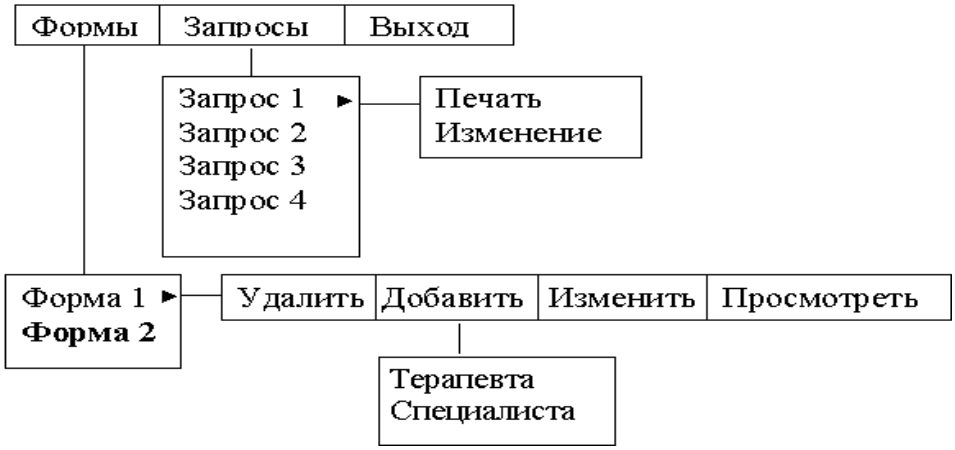


**Рис. 18.** Структура меню

Тогда текст программы будет следующим:

```
DEFINE MENU main
DEFINE PAD padmainOne OF main PROMPT '\<Формы '
DEFINE PAD padmainTwo OF main PROMPT '\<Запросы '
DEFINE PAD padmainThree OF main PROMPT '\<Выход '
ON PAD padmainOne OF main ACTIVATE POPUP popMainOne
ON PAD padmainTwo OF main ACTIVATE POPUP popMainTwo
DEFINE POPUP popmainOne
DEFINE BAR 1 OF popmainOne PROMPT '\<Форма 1 '
DEFINE BAR 2 OF popmainOne PROMPT '\<Форма 2 '
ON BAR 1 OF popmainOne ACTIVATE menu menuline
DEFINE menu menuline
DEFINE PAD padlineOne OF menuline PROMPT '\<Удалить ' at 1.5,7.5
DEFINE PAD padlineTwo OF menuline PROMPT '\<Добавить ' at 1.5,15
DEFINE PAD padlineThree OF menuline PROMPT '\<Изменить ' at
1.5,23.5
DEFINE PAD padlineFour OF menuline PROMPT '\<Просмотреть ' at
1.5,32
ON PAD padlineTwo OF menuline ACTIVATE POPUP popmenuline
DEFINE POPUP popmenuline
DEFINE BAR 1 OF popmenuline PROMPT '\<Терапевта'
DEFINE BAR 2 OF popmenuline PROMPT '\<Специалиста '
DEFINE POPUP popMainTwo
DEFINE BAR 1 OF popMainTwo PROMPT '\<Запрос 1 '
DEFINE BAR 2 OF popMainTwo PROMPT '\<Запрос 2 '
DEFINE BAR 3 OF popMainTwo PROMPT '\<Запрос 3 '
DEFINE BAR 4 OF popMainTwo PROMPT '\<Запрос 4 '
ON BAR 1 OF popMainTwo ACTIVATE POPUP menuzapr
DEFINE POPUP menuzapr
```

```
DEFINE BAR 1 OF menuzapr PROMPT '\<Печать'
DEFINE BAR 2 OF menuzapr PROMPT '\<Изменение '
ACTIVATE MENU main
```

Результат работы представлен на рис. 19.


**Рис. 19.** Сгенерированное меню

Кроме управления доступом к некоторым пунктам меню с помощью опции SKIP рассмотренных команд определения, есть специальные команды разрешения или запрещения доступа, в зависимости от значения <вржL>, соответственно .F. / .T..

SET SKIP OF MENU <имя меню> <вржL>

SET SKIP OF PAD <имя PAD-меню> OF <имя BAR-меню> <вржL>

SET SKIP OF POPUP <имя POPUP-меню> <вржL>

SET SKIP OF BAR <имя BAR-меню> OF <имя POPUP-меню> <вржL>

Эти команды используются следующим образом:

```
IF <условие блокировк>
  SET SKIP OF PAD PadName OF  BarName .T.
  * пункт недоступен
ELSE
  SET SKIP OF PAD PadName OF  BarName .F.
       * пункт доступен
  ENDIF
```

Запуск на выполнение выделенного пункта меню осуществляется нажатием клавиши **Enter** или **Space**, или щелчком на нем основной кнопкой мыши.

При выборе пункта меню может быть:

1. Активизировано всплывающее меню:

**ON BAR <имя BAR-меню> OF <имя POPUP-меню> ACTIVATE POPUP <имя POPUP-меню>**

**ON PAD <имя PAD-меню> OF <имя BAR-меню> ACTIVATE POPUP <имя POPUP-меню>**

или другая линейка меню командами:

**ON BAR <имя BAR-меню> OF <имя POPUP-меню> ACTIVATE MENU <имя меню>**

2. Выполнена заданная команда (подпрограмма).

Команды, определяющие реакцию на выбор элементов меню, следующие:

**ON SELECTION MENU <имя> [<команда>]**

**ON SELECTION BAR<номер>OF <имя POPUP-меню> [<команда>]**

**ON SELECTION PAD <имя> OF <имя BAR-меню > [<команда>]**

**ON SELECTION POPUP <имя>/ALL [<команда>]**

Если параметр <команда> отсутствует, назначение отменяется.

Обычно в качестве параметра <команда> используется вызов процедуры. Например, для просмотра отчета при выборе всплывающего меню rep выполните следующие команды:

```
ON SELECTION POPUP rep DO repproc
……
PROCEDURE repproc
USE <имя таблицы>
REPORT FORM rep.frx PREVIEW
RETURN
```

Но иногда при выборе пункта меню выполняется только одна команда. В частности, для выхода из программы при выборе пункта с именем exit меню необходимо выполнить команду:

ON SELECTION PAD exit OF <имя меню> CANCEL

Для открытия для выполнения окна формы или набора форм, созданных в окне проектирования, необходимо выполнить команду:

ON SELECTION <пункт меню> DO FORM <имя формы>

Для показа или выдачи на печать отчета на основе файла, созданного с помощью окна проектирования отчета:

ON SELECTION <пункт меню> FORM REPORT <имя отчета>

Для выполнения запроса, созданного в окне проектирование, необходимо выполнить команду:

ON SELECTION <пункт меню> DO <имя файла> .qpr

Если не указать расширение .qpr, то Visual FoxPro будет выполнять поиск файла в следующем порядке для расширений: .EXE, .APP, .FXP, .PRG.

Создадим меню, которое дополняет линейку системного меню FoxPro тремя пунктами: **Формы**, **Отчет** и **Выход**. При выборе пункта **Формы** активизируется POPUP-меню из двух пунктов – **Врачи** и **Пациенты**, которые открывают соответствующие формы из базы данных регистратуры. При выборе пункта **Отчет** активизируется POPUP-меню, состоящее из пунктов – **Отчет 1** и **Отчет 2**, которые позволяют просматривать предварительно созданные отчеты. Меню **Выход** возвращает системное меню в обычное состояние. Текст головной программы будет иметь вид:

```
SET DEFA TO home()+'registrybd'  && делаем текущей папку registrybd
&& при дополнении системного меню строчки DEFINE MENU и ACTIVATE MENU не требуются !!!

DEFINE PAD padmainOne OF _MSYSMENU PROMPT '\<Формы '
DEFINE PAD padmainTwo OF _MSYSMENU PROMPT '\<Отчет '
DEFINE PAD padmainThree OF _MSYSMENU PROMPT '\<Выход '
&& добавление пунктов Формы, Отчет и Выход

ON PAD padmainOne OF _MSYSMENU ACTIVATE POPUP popMainOne
&& активизация POPUP-меню  при выборе пункта Формы

ON PAD padmainTwo OF _MSYSMENU ACTIVATE POPUP popMainTwo
&& активизация POPUP-меню  при выборе пункта Отчет

DEFINE POPUP popmainOne
DEFINE BAR 1 OF popmainOne PROMPT '\<Врачи '
DEFINE BAR 2 OF popmainOne PROMPT '\<Пациенты '
&& создание POPUP-меню  пункта Формы

DEFINE POPUP popMainTwo
DEFINE BAR 1 OF popMainTwo PROMPT '\<Отчет 1'
DEFINE BAR 2 OF popMainTwo PROMPT '\<Отчет 2 '
&& создание POPUP-меню  пункта Отчет

ON SELECTION PAD padmainThree OF _MSYSMENU SET SYSMENU TO DEFAULT
```

*&& возврат системного меню в обычное состояние при выборе пункта Выход*

ON SELECTION BAR 1 OF popmainOne DO FORM Doctortab
ON SELECTION BAR 2 OF popmainOne DO FORM Patienttab
*&& открытие соответствующих форм при выборе пунктов Врачи и Пациенты*

ON SELECTION BAR 1 OF popMainTwo DO Rep1
*&& вызов программы Rep1 при выборе пункта Отчет 1*

ON SELECTION BAR 2 OF popMainTwo DO Rep2
*&& вызов программы Rep2 при выборе пункта Отчет 2*

Текст программы Rep1:

```
DEFINE WINDOW Win1 FROM 0,0 TO 150,150 TITLE 'Отчет 1';
CLOSE FLOAT GROW ZOOM  && создаем окно
ACTIVATE WINDOW Win1   && активизируем окно
REPORT FORM Doctortab;
PREVIEW IN window Win1;
NOWAIT
RETURN
```

Текст программы Rep2:

```
DEFINE WINDOW Win2 FROM 0,0 TO 150,150 TITLE 'Отчет 2';
CLOSE FLOAT GROW ZOOM  && создаем окно
ACTIVATE WINDOW Win2   && активизируем окно
REPORT FORM Patienttab;
PREVIEW IN window Win2;
NOWAIT
RETURN
```

Дополненное системное меню представлено на рис. 20.

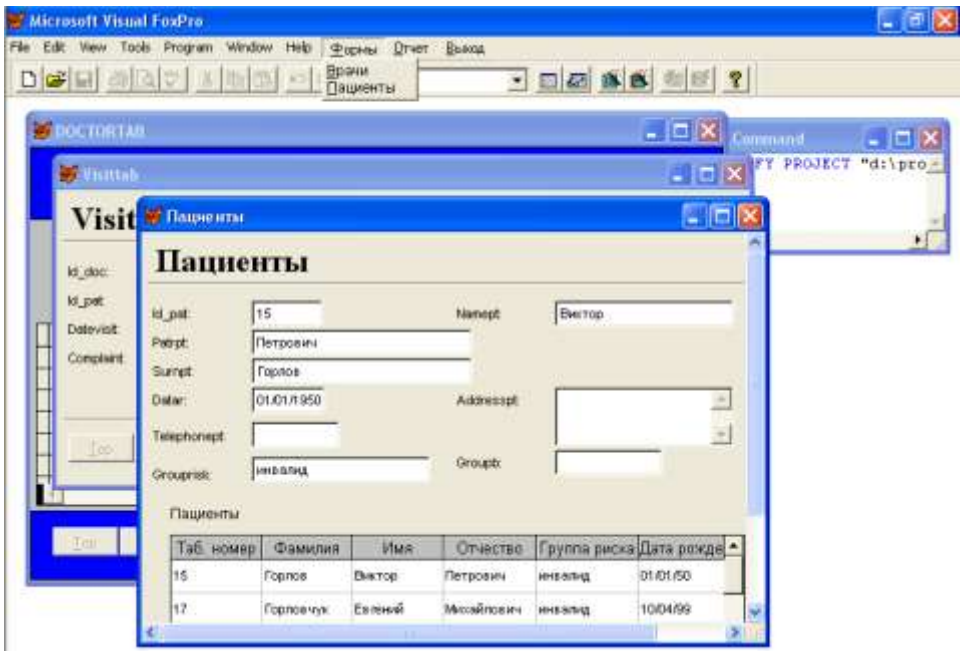


**Рис. 20.** Дополненное системное меню

*7. Создание выполнимого приложения*

После того как будет завершено проектирование и отладка всех необходимых экранных форм и системы меню, можно приступить к созданию выполнимого приложения.

Первоначально в диспетчере проекта следует отметить *главный элемент проекта*, используемый как основной и первый файл при генерации проекта. Главным элементом, как правило, выступает меню. При отсутствии меню главным элементом проекта также может быть окно формы или программный файл. Главный элемент проекта устанавливается из меню **Project → Set Main** и отмечается в списке элементов проекта полужирным начертанием шрифта.

Для построения приложения щелкните на кнопке **Build** в окне Диспетчера проектов. Вы увидите диалоговое окно **Build Options** (рис. 21), в котором выберите один из вариантов построения:

- **Rebuild Project** – обновление проекта;
- **Build Application** – построение файла приложения с расширением .app;
- **Build Executable** – создание исполняемого файла с расширением .exe;
- **Build COM DLL** – создание файла динамической библиотеки с расширением .dll.
-

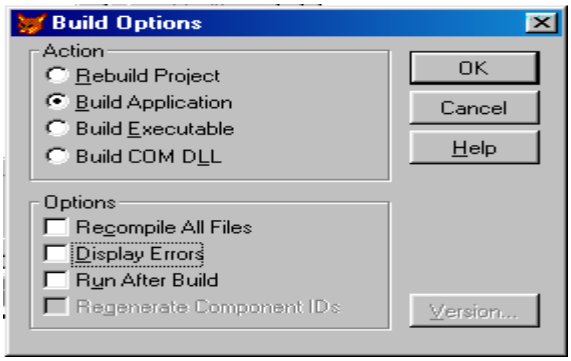


**Рис. 21.** Варианты построения приложения

Запуск .app-файла выполняется командой

DO <имя файла с расширением .app >

Файл с расширением .exe выполняется в среде Windows. При его запуске поверх псевдокомпилированного кода вызывается загрузчик, вызывающий в свою очередь библиотеку поддержки – Runtime-модуль.

Запуск .exe-файла приложения выполняется без входа в среду Visual FoxPro 7.0, а при запуске .APP-файла нужна сама среда Visual FoxPro.

В окне **Build Options** имеются четыре флажка:

- **Recompile All Files** – обновление компонентов проекта, измененных после предыдущего построения;
- **Display Errors** – Отображение всех ошибок, обнаруженных при построении; но даже если этот флажок не установлен, результат диагностирования можно увидеть, выполнив команду **Project → Errors**;
- **Run After Build** – устанавливается, если приложение необходимо запустить сразу после создания;
- **Regenerate Component Ids** – регенерация компонентов; этот флажок доступен только при выборе **Build Executable** или **Build COM DLL**.

**Порядок выполнения работы**

1. Разработать структуру многоуровневого меню приложения.
2. Создать меню с помощью конструктора меню, обеспечив:

- описание пунктов меню;

- назначение на пункты меню команд, процедур, подменю, используя разработанные ранее для БД запросы, отчеты, формы;
- отображение меню на экране;
3. Сохранить и выполнить генерацию меню.
4. Запустить меню на исполнение.

**Контрольные задания**

1. Используя разделители, создать горизонтальное многоуровневое меню для вызова созданных раннее форм.
2. Создать горизонтальное меню для вызова созданных раннее форм. Предусмотреть использование цифр в качестве горячих клавиш.
3. Используя разделители, создать горизонтальное многоуровневое меню для вызова созданных раннее отчетов.
4. Создать горизонтальное меню для вызова созданных раннее отчетов. Предусмотреть использование цифр в качестве горячих клавиш.
5. Используя разделители, создать горизонтальное многоуровневое меню для вызова запросов.
6. Создать горизонтальное меню для вызова запросов. Предусмотреть использование цифр в качестве горячих клавиш.
7. Используя разделители, создать горизонтальное многоуровневое меню для вызова созданных раннее форм и запросов.
8. Создать горизонтальное меню для вызова запросов и форм. Предусмотреть использование цифр в качестве горячих клавиш.
9. Используя разделители, создать вертикальное меню для вызова созданных раннее форм. Предусмотреть наличие пункта *Выход*.
10. Создать вертикальное меню для вызова созданных раннее форм. Предусмотреть наличие горячих клавиш и пункта *Выход*.
11. Используя разделители, создать вертикальное меню для вызова созданных раннее запросов. Предусмотреть наличие пункта *Выход*.
12. Создать вертикальное меню для вызова созданных раннее запросов. Предусмотреть наличие горячих клавиш и пункта *Выход*.
13. Используя разделители, создать вертикальное меню для вызова созданных раннее отчетов. Предусмотреть наличие пункта *Выход*.
14. Создать вертикальное меню для вызова созданных раннее отчетов. Предусмотреть наличие горячих клавиш и пункта *Выход*.
15. Используя разделители, создать вертикальное многоуровневое меню для вызова созданных раннее форм, запросов и отчетов.

**Контрольные вопросы**

1. Что такое горизонтальное меню?
2. Что такое вертикальное меню?
3. Для чего создают меню приложения?

4. В чем заключается суть подготовки к созданию меню?
5. Основные способы запуска конструктора меню?
6. Области конструктора меню и их назначение?
7. Этапы создания горизонтального меню?
8. Этапы создания вертикального меню?
9. Что будет, если два элемента меню начинаются с одинаковой буквы?
10. Каким образом в меню назначаются “горячие” клавиши?
11. Основные функции диалогового окна Prompt Options?
12. Каким образом можно заблокировать пункт меню?
13. Каким образом можно задать имя пункта меню?
14. Каким образом можно задать комментарий к пункту меню?
15. Этапы создания подменю?
16. Что указывается в списке Menu Level конструктора меню?
17. Каким образом можно связывать команды с пунктами меню или подменю?
18. Каким образом можно связывать процедуры с пунктами меню или подменю?
19. Этапы создания разделителей элементов меню?