МИНОБРНАУКИ РОССИИ

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

Кафедра информационной безопасности

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_____»_____2017 г.

РАЗРАБОТКА КЛИЕНТСКОГО ИНТЕРФЕЙСА ДЛЯ БД И СОЗДАНИЕ ОТЧЕТОВ В КЛИЕНТСКОМ ПРИЛОЖЕНИИ

Методические указания по выполнению лабораторной работы №4
для студентов направления подготовки бакалавриата
10.03.01«Информационная безопасность»

Курск 2017

УДК 621.(076.1)

Составители: А.Г. Спеваков

Рецензент

Кандидат технических наук, доцент кафедры
«Информационная безопасность» И.В. Калуцкий

Разработка клиентского интерфейса [Текст] : методические указания по выполнению лабораторной работы / Юго-Зап. Гос. ун-т; сост.: А.Г. Спеваков. – Курск, 2017. – 35с.: ил. 29. – Библиогр.: с. 35.

Содержат сведения по вопросам проектирования баз данных. Указывается порядок выполнения лабораторной работы, правила содержание отчета.

Методические указания соответствуют требованиям программы, утвержденной учебно-методическим объединением по специальности.

Предназначены для студентов направления подготовки бакалавриата 10.03.01 «Информационная безопасность».

Текст печатается в авторской редакции

Подписано в печать . Формат 60х84 1/16.
Усл.печ. л. 2,03. Уч.-изд. л. 1,84.Тираж 100 экз. Заказ. Бесплатно.
Юго-Западный государственный университет.
305040, г.Курск, ул. 50 лет Октября, 94.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Клиентский интерфейс базы данных предназначен для удаленного использования и администрирования. В данной методичке для создания клиентского интерфейса используется следующее ПО: Delphi 7 (с дополнительно поставленным компонентом ehlibrus) и MSSQL Server 2008.

Ehlibrus.rar

Установка:

1. Распакуйте содержимое архива в любую папку, далее архив можно удалить. Создайте пустую папку для установки Ehlibrus.

Например: C:\Program Files\Borland\Delphi7\eh

2. Скопируйте все файлы из папки Common (распакованного архива) в новую папку C:\Program Files\Borland\Delphi7\eh
3. Скопируйте все файлы из папки Delphi7(распакованного архива) в новую папку C:\Program Files\Borland\Delphi7\eh
4. Запускаем Delphi7, заходим Инструменты=> Опции среды (Рис. 1)

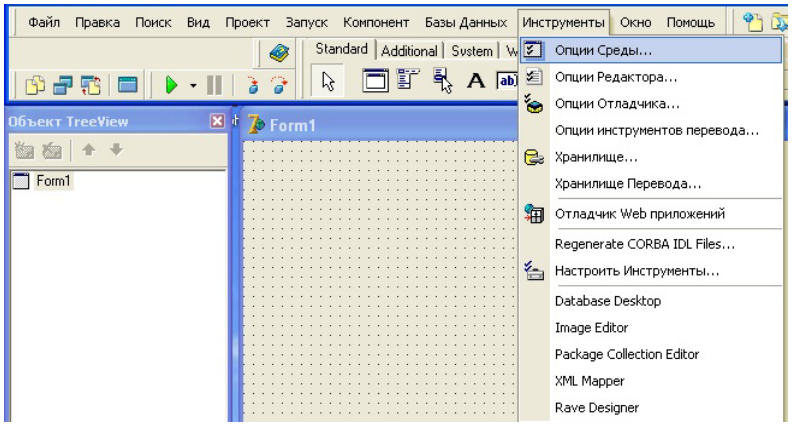


Рис. 1 - Окно Delphi7, вкладка Инструменты

5. Нажимаем на вкладку “Библиотека” => Путь библиотеки. Добавляем в путь папку, которую мы создали ранее.(Рис. 2)

6. Открываем нашу папку “eh” и запускаем файл “DclEhLib70.dpk”. В окне установки файла DclEhLib70.dpk нажимаем установить (Install).

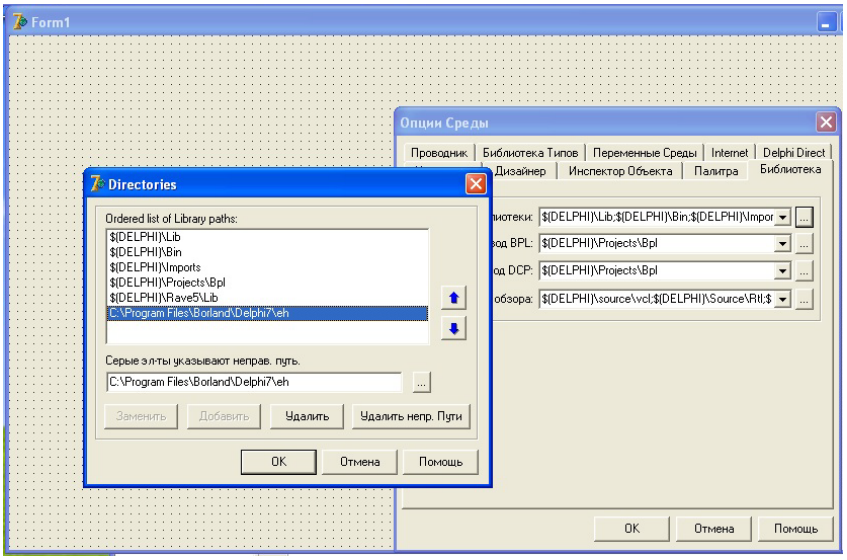


Рис. 2 - Добавление в путь созданную папку

7. Если всё сделано правильно, будет получено сообщение, “has been installed” - проект установлен.

После установки компонента в Delphi должна появиться следующая вкладка:

На этом установка компонента Ehlibrus завершена.

ЦЕЛЬ РАБОТЫ

Получить доступ к данным СУБД, разработать клиентское приложение с удобным пользовательским интерфейсом, позволяющим осуществлять просмотр, выборку и изменение содержимого БД. Формирование отчетов средствами клиентского приложения.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Получить задание.
2. Изучить теоретическую часть.
3. Установить необходимое ПО и дополнительные компоненты.
4. Разработать интерфейс клиентского приложения.
5. Разработать текст SQL-запросов для выборки, добавления, изменения и удаления субъектов базы данных.
6. Создать процедуру формирования отчетов в клиентском приложении.

СОДЕРЖАНИЕ ОТЧЕТА

1. Индивидуальное задание.
2. Создание проекта.
3. Установка подключения к базе данных.
4. Создание интерфейса.
5. Отображение таблиц БД в клиенте.
6. Поиск по таблицам БД.
7. Добавление/удаление/изменение в таблицах БД.
8. Формирование отчетов.
9. Выводы.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Компоненты Delphi для работы с базами данных были созданы в расчете на работу с SQL и архитектурой клиент/сервер. При работе с ними вы можете воспользоваться характеристиками расширенной поддержки удаленных серверов. Delphi осуществляет эту поддержку двумя способами. Во-первых, непосредственные команды из Delphi позволяют разработчику управлять таблицами, устанавливать пределы, удалять, вставлять и редактировать существующие записи. Второй способ заключается в использовании запросов на языке SQL, где строка запроса передается на сервер для ее разбора, оптимизации, выполнения и передачи обратно результатов.

Для доступа к данным SQL Server из клиентских приложений, написанных на языке Delphi для платформы Win32 можно применять хорошо известные универсальные технологии Borland Database Engine (BDE) и DBExpress, разработанные фирмой Borland. Однако рекомендуемый подход – использование технологии Microsoft ActiveX Data Objects (ADO), оптимизированной для SQL Server.

Технология ADO основана на возможностях COM, а именно интерфейсов OLE DB. Базовый набор интерфейсов OLE DB предустановлен во всех версиях Microsoft Windows, поэтому при переносе приложения на другой компьютер для его работоспособности достаточно лишь правильно настроить провайдер OLE DB. Провайдер OLE DB представляет собой COM-сервер, предоставляющий набор интерфейсов для доступа к данным, и «скрывающий» особенности конкретных источников данных. Провайдеры OLE DB разработаны для большинства СУБД (Oracle, Interbase и др.) и многих других источников данных. Часть провайдеров (например, OLE DB Provider for SQL Server) уже установлена в системе, другие доступны для скачивания в Internet. Технология ADO – надстройка над интерфейсом OLE DB, облегчающая его использование прикладными программистами.

Технология ADO и интерфейсы OLE DB предоставляют приложениям единый способ доступа к источникам данных различных типов. Приложение, использующее ADO, может однотипно работать с данными, хранящимися на сервере SQL, с

электронным таблицами и локальными СУБД. Согласно терминологии ADO, любой источник данных (база данных, электронная таблица, файл) называется хранилищем данных, с которым при помощи провайдера взаимодействует приложение.

В результате приложение обращается не напрямую к источнику данных, а к объекту OLE DB, который представляет данные в виде таблицы БД или результата выполнения запроса SQL. Такая архитектура позволяет сделать набор объектов и интерфейсов открытым и расширяемым. Набор объектов и соответствующий провайдер могут быть созданы для любого хранилища данных без изменения исходной структуры ADO. При этом существенно расширяется само понятие данных. Можно разработать набор объектов и интерфейсов и для не табличных данных, например, графических данных, древовидных структур, данных CASE-инструментов и др.

В Delphi на странице ADO палитры компонентов расположены компоненты доступа к данным, инкапсулирующие технологию ADO. Общая методика их использования построена по тем же принципам, что и у остальных компонентов доступа к данным (BDE, IBExpress, DBExpress и др.), однако внутренняя организация совсем другая. Это удобно, так как программист может с успехом использовать ранее имевшиеся навыки и опыт работы с другими СУБД. Например, компоненты ADO поддерживают навигацию, работу с наборами данных, кэшируемые изменения (здесь они называются пакетными обновлениями), управление транзакциями. Наиболее серьезное препятствие здесь – научиться мыслить категориями архитектуры клиент-сервер, и не пытаться переносить методы и приемы создания персональных баз, данных в многопользовательскую клиент серверную среду.

С другой стороны, с помощью свойств и методов упомянутых компонентов при необходимости легко обратиться к дополнительным возможностям ADO для более «тонкой» настройки приложения.

Если в ваших приложениях вы собираетесь использовать SQL, то вам непременно придется познакомиться с компонентом TQuery. Компоненты TQuery и TTable наследуются от TDataset. TDataset обеспечивает необходимую функциональность для

получения доступа к базам данных. Как таковые, компоненты TQuery и TTable имеют много общих признаков. Для подготовки данных для показа в визуальных компонентах используется все тот же TDatasource. Также, для определения к какому серверу и базе данных необходимо получить доступ, необходимо задать имя псевдонима. Это должно выполняться установкой свойства aliasName объекта TQuery.

Все же TQuery имеет некоторую уникальную функциональность. Например, у TQuery имеется свойство с именем SQL. Свойство SQL используется для хранения SQL-запроса.

Компонент TDatabase обеспечивает функциональность, которой не хватает TQuery и TStoredProc. В частности, TDatabase позволяет создавать локальные псевдонимы BDE, так что приложению не потребуются псевдонимы, содержащиеся в конфигурационном файле BDE. Этим локальным псевдонимом в приложении могут воспользоваться все имеющиеся TTable, TQuery и TStoredProc. TDatabase также позволяет разработчику настраивать процесс подключения, подавляя диалог ввода имени и пароля пользователя, или заполняя необходимые параметры. И, наконец, самое главное, TDatabase может обеспечивать единственную связь с базой данных, суммируя все операции с базой данных через один компонент. Это позволяет элементам управления для работы с БД иметь возможность управления транзакциями.

ВЫПОЛНЕНИЕ РАБОТЫ

Итак, у нас есть готовая база данных на сервере MSSQL Server 2008 и можно приступить непосредственно к разработке клиентского интерфейса.

1. Открываем Delphi и создаем новый проект. Для удобства, чтобы не создавать много форм в проекте, используется компонента PageControl из вкладки Win32. Таблиц в нашей базе всего 3, поэтому страниц на PageControl у нас тоже 3. Добавление новой страницы достигается при помощи контекстного меню. (Рис. 3)

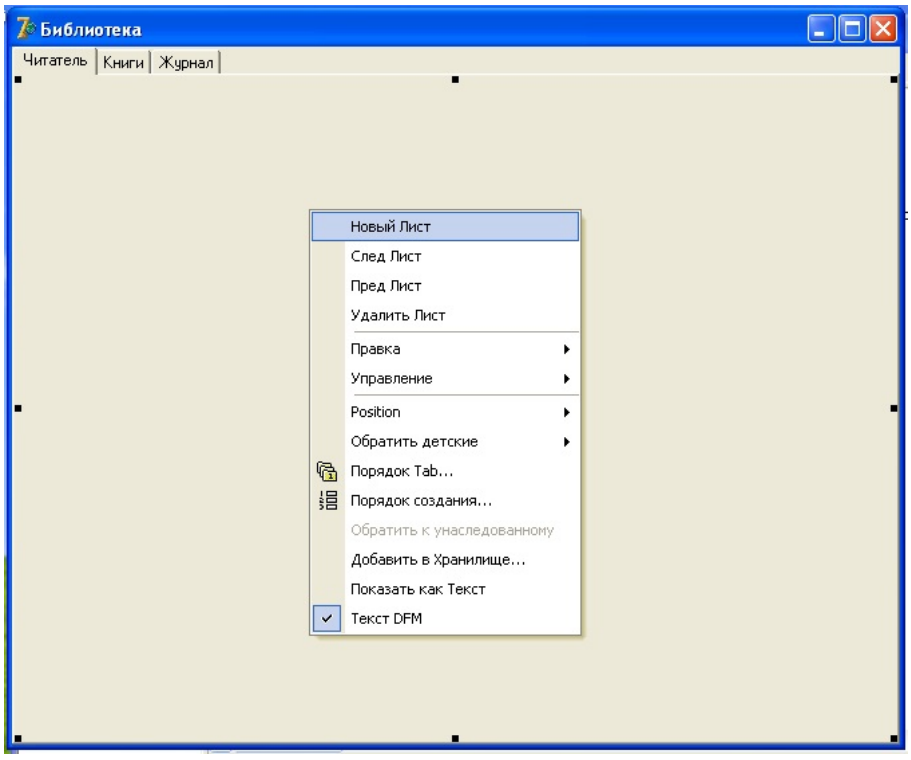


Рис. 3 - Добавление новой страницы

2. Создаём модуль данных (Data Module): Файл=> Новый=> Модуль данных (Рис. 4)
3. На наш Модуль Данных «ставим» следующие компоненты:

- TDataSource
- TADOConnection
- TADOQuery

TDataSource находится на вкладке Data Access, предназначен для связи нашей сетки отображения данных, с самой БД.

TADOConnection находится на вкладке ADO, предназначен для подключения нашей БД по определенному провайдеру.

TADOQuery находится также на вкладке ADO, предназначен для получение нужных результатов из нашей БД. (Рис. 5). Компоненты можно переименовать с помощью инспектора объектов изменяя поле Name.

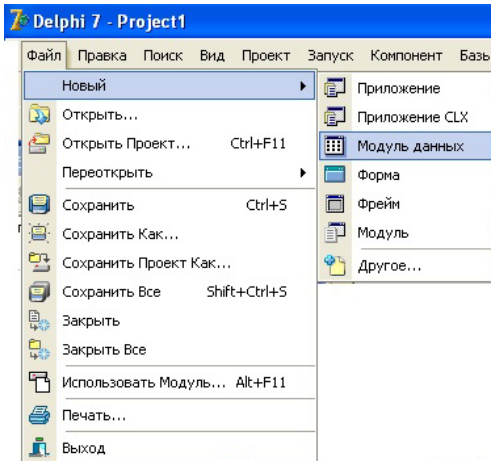


Рис. 4 - Создание модуля данных

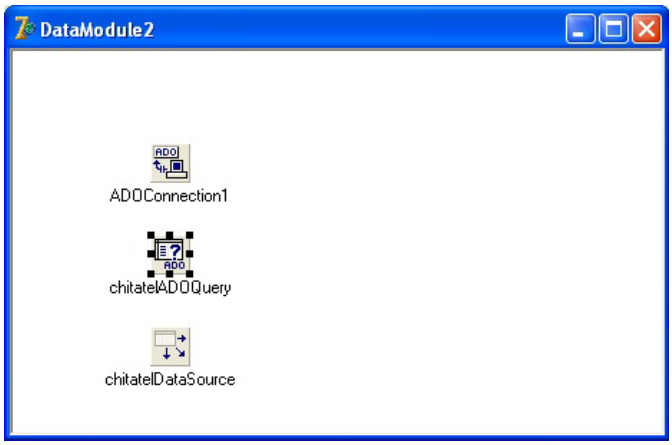


Рис. 5 - Добавление компонентов на модуль Данных

4. Связываем TADOQuery в свойстве Connection с TADOConnection. Из выпадающего списка выбираем имя данного компонента.
5. Связываем TDataSource в свойстве DataSet с TADOQuery из выпадающего списка выбираем имя данного компонента.

6. Выделяем компонент TADOConnection в свойстве ConnectionString нажимаем на кнопку с «…», появится окно следующего вида (Рис. 6). В данном окне нажимаем на кнопку «Build…», появится окно Свойство связи с данными (Рис. 7). В данном окне мы выбираем провайдера, а именно Microsoft OLE DB Provider for SQL Server и нажимаем кнопку «Далее».

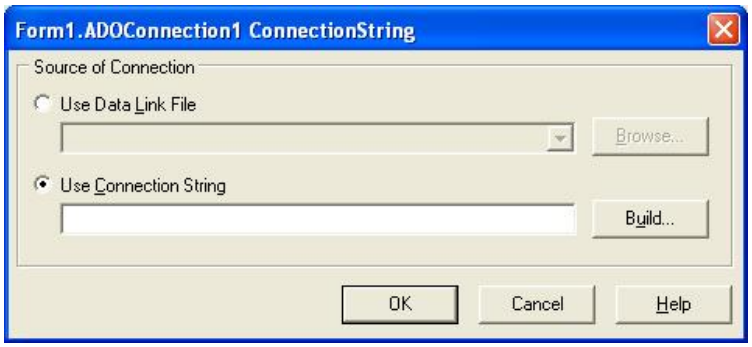


Рис. 6 - Окно компонента TADOConnection- ConnectionString

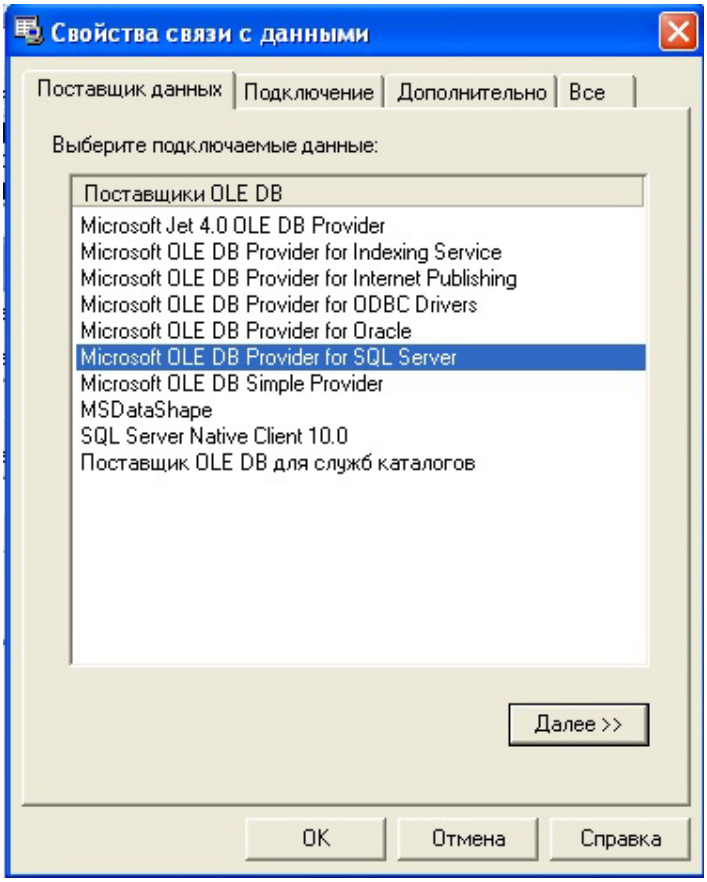


Рис. 7 - Окно Свойства связи с данными (Поставщик данных)

7. Во вкладке Подключение указываем имя сервера; имя пользователя; пароль (который мы создавали в sql server management studio для нашей базы данных) и выбираем нашу базу данных (Рис. 8). Нажимаем проверить подключение, если всё введено правильно, выскакивает окно, что проверка подключения выполнена, нажимаем ОК. (Рис. 9)

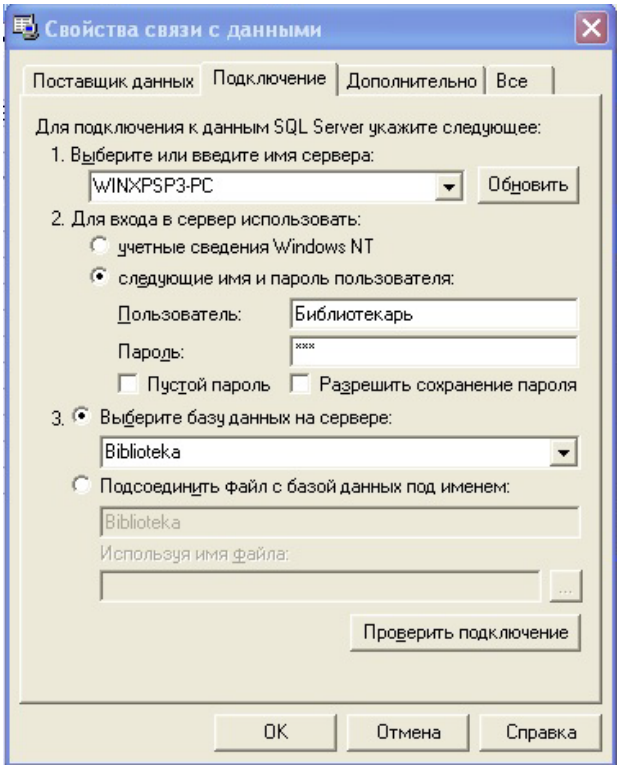


Рис. 8 - Окно Свойства связи с данными (Подключение)

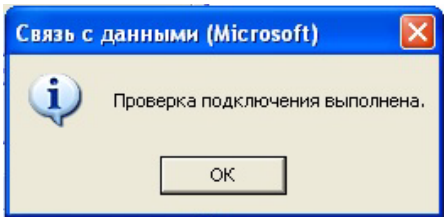


Рис. 9 - Проверка подключения

8. В свойстве ACTIVE TADOConnection ставим True. Заходим в свойства TADOQuery выбираем свойство SQL нажимаем на кнопку с «…» и появляется окно следующего вида (Рис. 10). Пишем в этом окне запрос на выбор всех столбцов из таблицы chitatel нашей базы данных (select * from chitatel) и нажимаем ОК. Ставим свойство ACTIVE в положение TRUE.

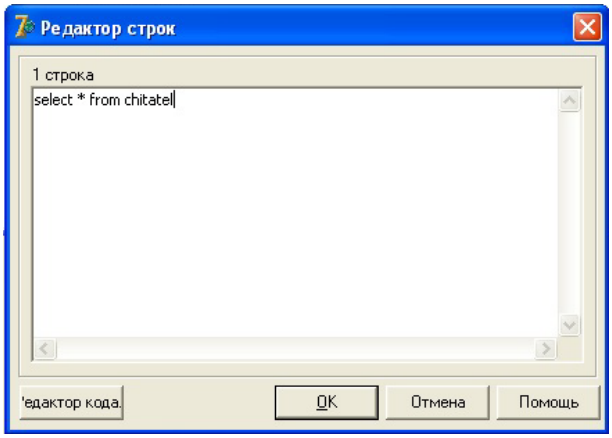


Рис. 10 - Запрос на выбор всех столбцов из таблиц

Итак, у нас есть контакт с сервером, переходим к созданию интерфейса

1. Поместим на форму (на активную страницу компонента PageControl) одну компоненту GroupBox из вкладки Standard и одну компоненту PageControl из вкладки Win32. В Object Inspector в GroupBox в поле Caption напишем “Поиск”. В поле Align соответственно выбираем “alTop” и “alBottom”. Между компонентами GroupBox и PageControl вставим из вкладки EhLib компоненту DBGridEh. В компоненте DBGridEh в поле Align выбираем “alClient” (Рис. 11).

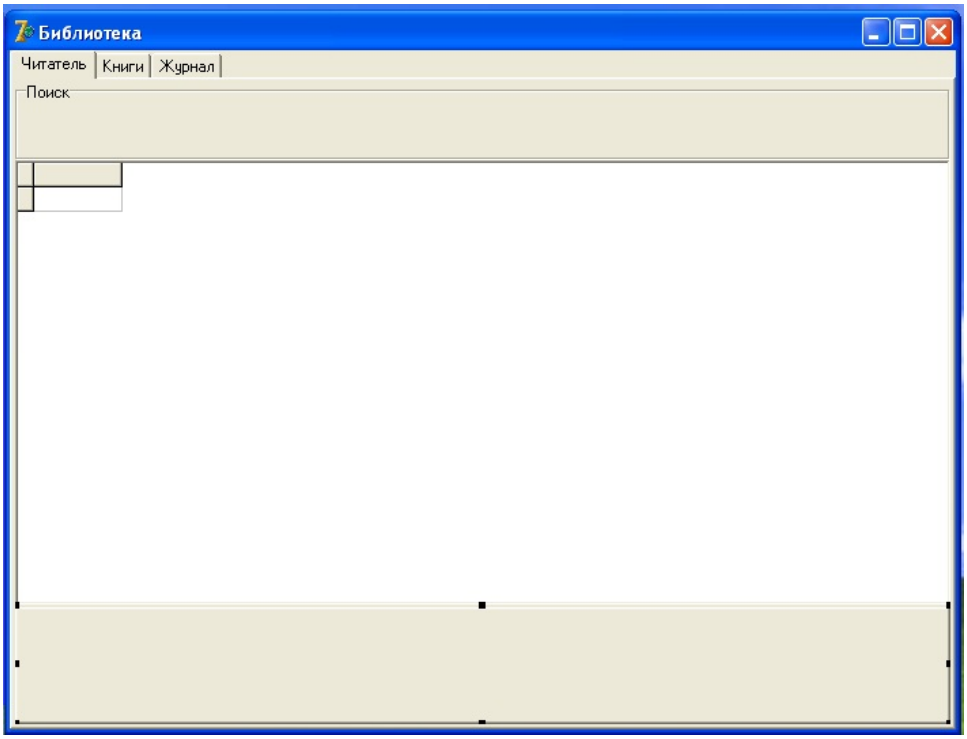


Рис. 11 - Расстановка компонентов

Параметры поля Align мы меняли для того, чтобы при развертывании окна приложения, компоненты также меняли свои размеры.

2. Заполним GroupBox “Поиск”, поместим на него компоненты ComboBox, Edit и Button (все компоненты с вкладки Standard); кнопку назовем “Искать”, текст компоненты Edit очистим, а компоненту ComboBox трогать пока не будем (она понадобится нам для поиска по отдельным столбцам таблицы). Добавляем новые листы на компоненте PageControl (Рис. 12)

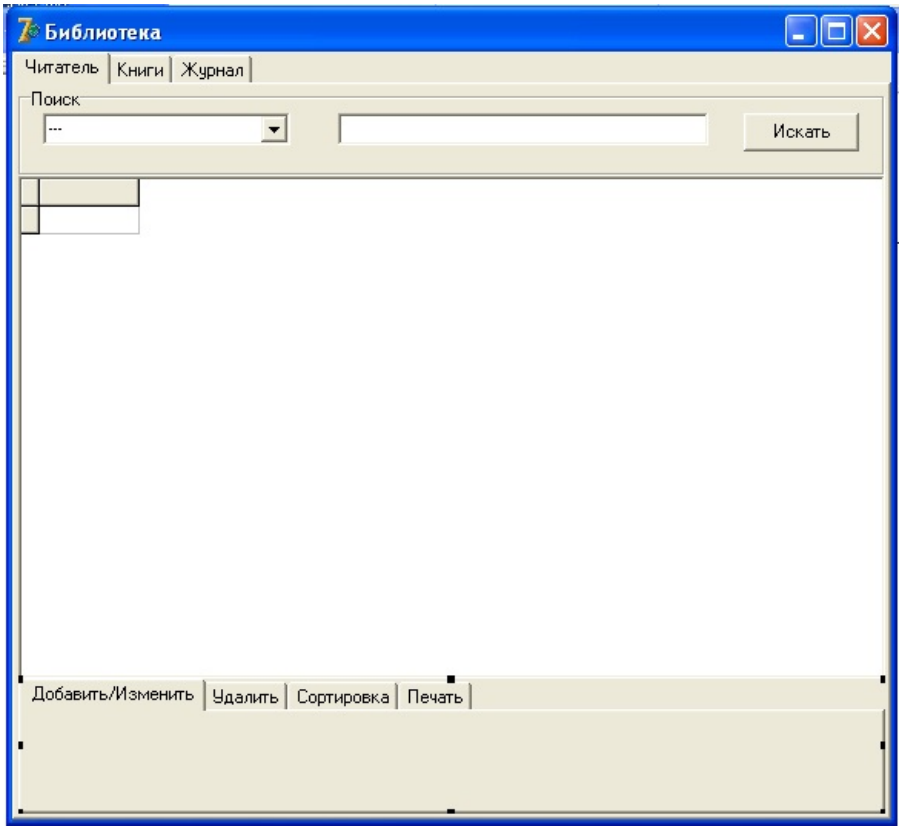


Рис. 12 - Добавление компонентов

Интерфейс для одной из таблиц базы данных создан. Данная последовательность шагов повторяется для всех таблиц, которые необходимо отобразить в клиентском приложении.

Из вкладки Ehlib помещаем на любую область формы компонент:

1. PrintDBGridEh – компонента для печати содержимого DBGridEh.

2. Компонент PrintDBGridEh в параметре DBGridEh выберем нужный нам DBGridEh.

Связываем нашу базу данных с DBGridEh: в параметре DataSource компоненты DBGridEh выбираем нужный нам DataSource – в конкретном примере “chitatelDataSource” (Рис. 14), но перед этим в Unit1.pas в uses подключаем (прописываем) Unit2 (Рис. 13)

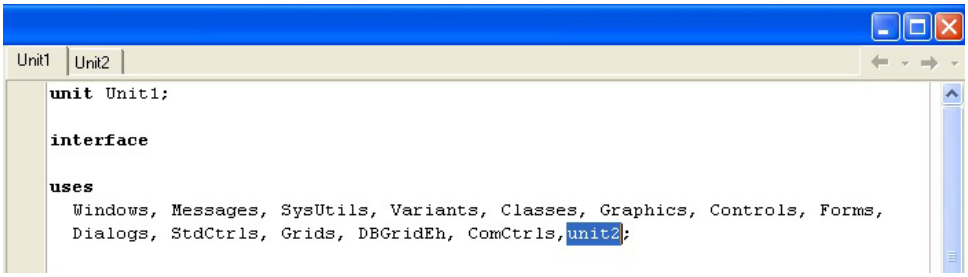


Рис. 13 - Подключение Unit2

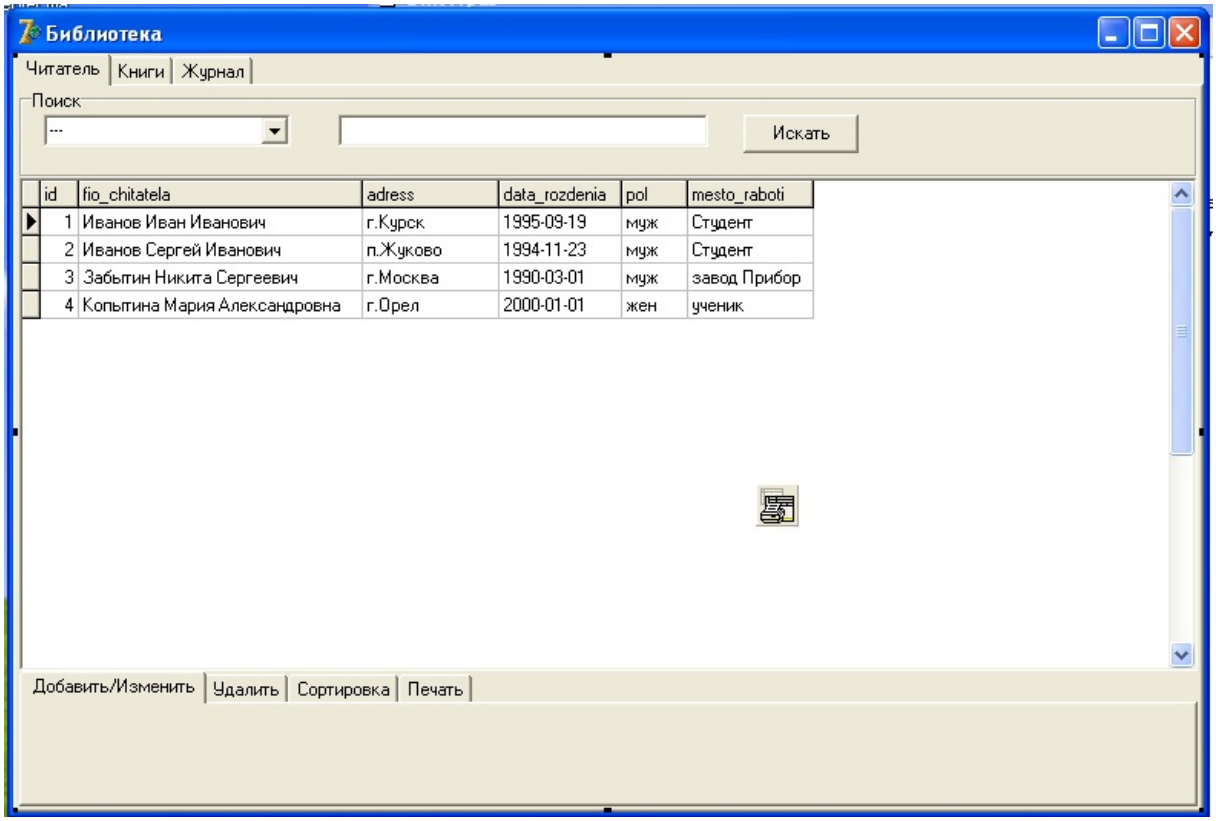


Рис. 14 - Связь базы данных с DBGridEh

Таким образом, мы видим отображение таблицы базы данных в нашем клиентском приложении (Рис. 14), но требуется немного

отредактировать это отображение. Выполним двойной клик на компоненте DBGridEh, появится окно Editing DBGridEh1.Columns в нем выполним левый клик и выберем опцию “Add all fields” (если опция “Add all fields” неактивна, значит TADOQuery не выставлена в состояние true).

Рис. 15 - Настройка полей таблицы

В окне Editing DBGridEh1.Columns (Рис. 15) мы видим поля таблицы, ими мы можем управлять в инспекторе объектов. Нас интересует изменение полей Title.Caption (заголовок поля в DBGridEh), Visible (показывать/не показывать поле), Width (ширина поля). Изменив параметры этих полей приведем DBGridEh в оформленный вид (Рис. 16)

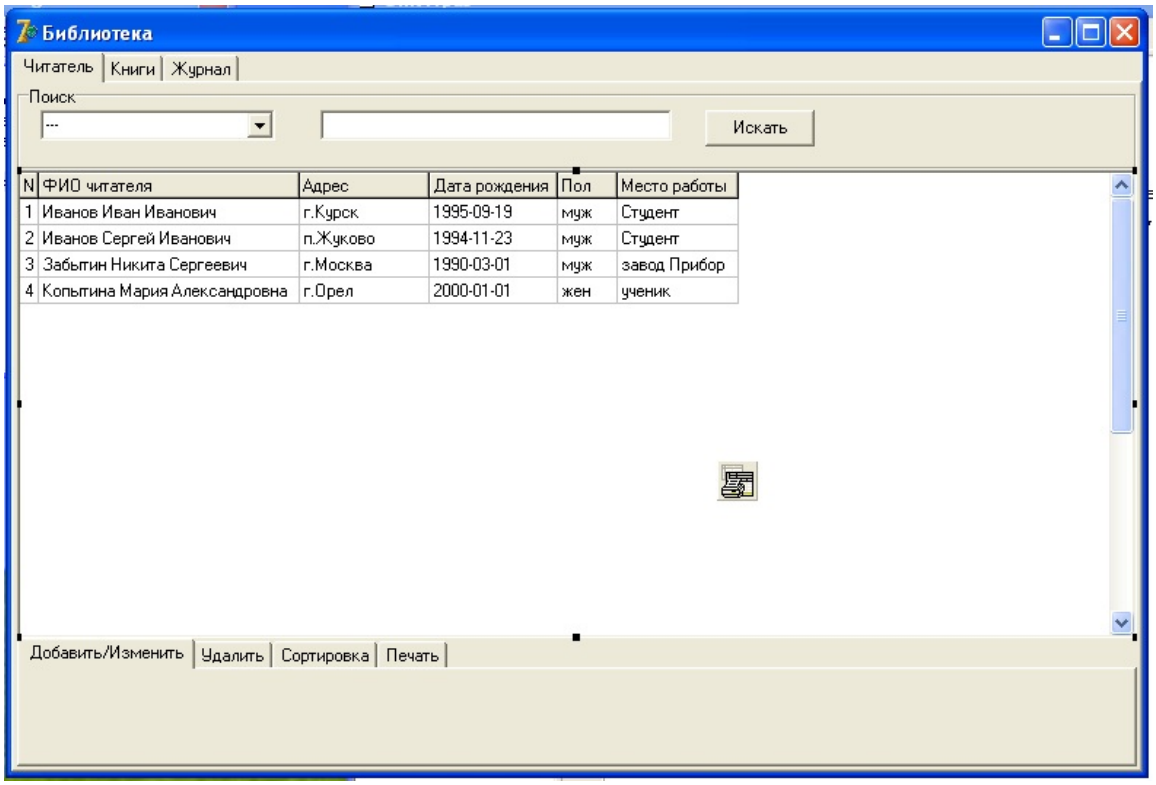


Рис. 16 - Полученный интерфейс

Итак, мы получили отображение в приложении одной таблицы из базы данных, проделываем тоже самое с другими таблицами.

Теперь сделаем поиск по таблице базе данных, которая отображается у нас в DBGridEh. Поиск у нас будет проходить как по каждому полю таблицы, так и по всем ее полям, для начала добавим в ComboBox1.Items нужные нам поля (Рис. 17)

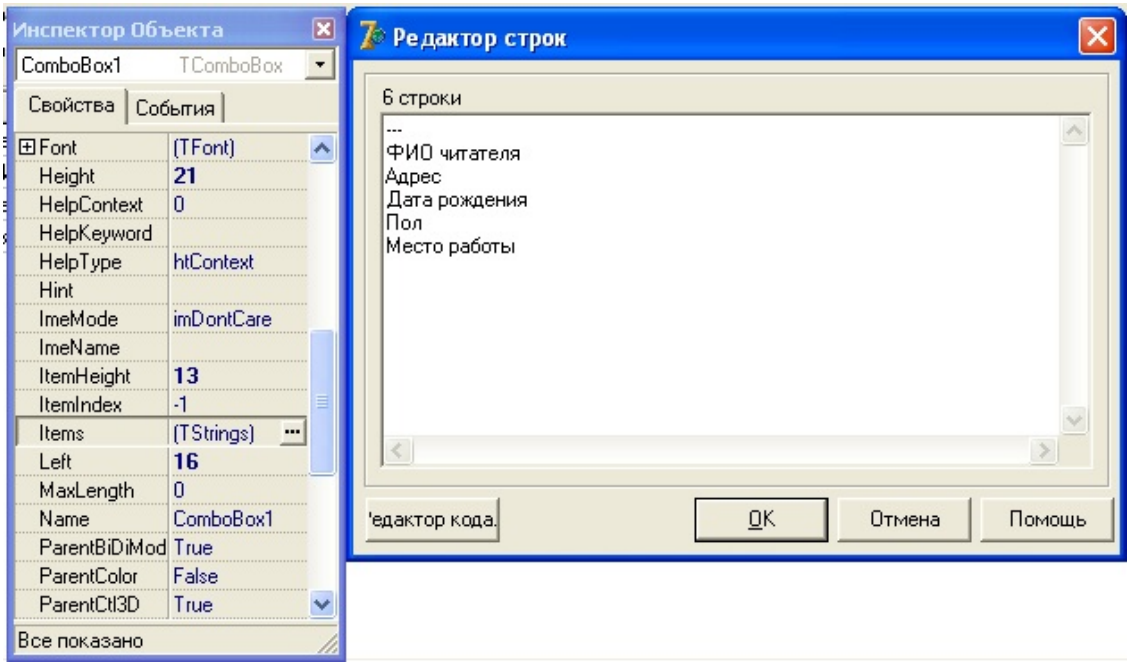


Рис. 17 - Добавление в ComboBox1.Items нужные нам поля

Сделаем двойной клик на кнопке “Искать” и вставим в процедуру обработки нажатия кнопки следующий код:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
if edit1.Text='' then
begin
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel';
DataModule2.chitatelADOQuery.Active:=true;
end else
begin
If ComboBox1.ItemIndex=1 then
begin
```

```
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel where fio_chitatela like "'+'%'+edit1.text+'%'+"";
DataModule2.chitatelADOQuery.Active:=true;
end else
If ComboBox1.ItemIndex=2 then
begin
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel where adress like "'+'%'+edit1.text+'%'+"";
DataModule2.chitatelADOQuery.Active:=true;
end else
If ComboBox1.ItemIndex=3 then
begin
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel where data_rozdenia like "'+'%'+edit1.text+'%'+"";
DataModule2.chitatelADOQuery.Active:=true;
end else
If ComboBox1.ItemIndex=4 then
begin
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel where pol like "'+'%'+edit1.text+'%'+"";
DataModule2.chitatelADOQuery.Active:=true;
end else
If ComboBox1.ItemIndex=5 then
begin
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel where mesto_raboti like "'+'%'+edit1.text+'%'+"";
DataModule2.chitatelADOQuery.Active:=true;
end else
begin
DataModule2.chitatelADOQuery.Active:=false;
DataModule2.chitatelADOQuery.SQL.Text:='select * from chitatel where (fio_chetatela like "'+'%'+edit1.text+'%'+"") or (adress like
```

'"+'%'+edit1.text+'%'+'") or (data_rozdenia like '"+'%'+edit1.text+'%'+'") or (pol like '"+'%'+edit1.text+'%'+'") or (mesto_raboti like '"+'%'+edit1.text+'%'+'")';
DataModule2.chitatelADOQuery.Active:=true;
end;
end;
end;

Поиск по всей таблице и по каждому полю готов и описывается вышеописанным кодом. Аналогичным образом данный шаг повторяется для выполнения поиска для других таблиц.

Перейдем к процедуре добавления в базу данных. Для этого добавим на форму на компоненту PageControl (лист Добавить/Изменить) 5 DBEditEh из вкладки Ehlib и 5 Label из вкладки Standart и одну кнопку Button (Рис. 18)

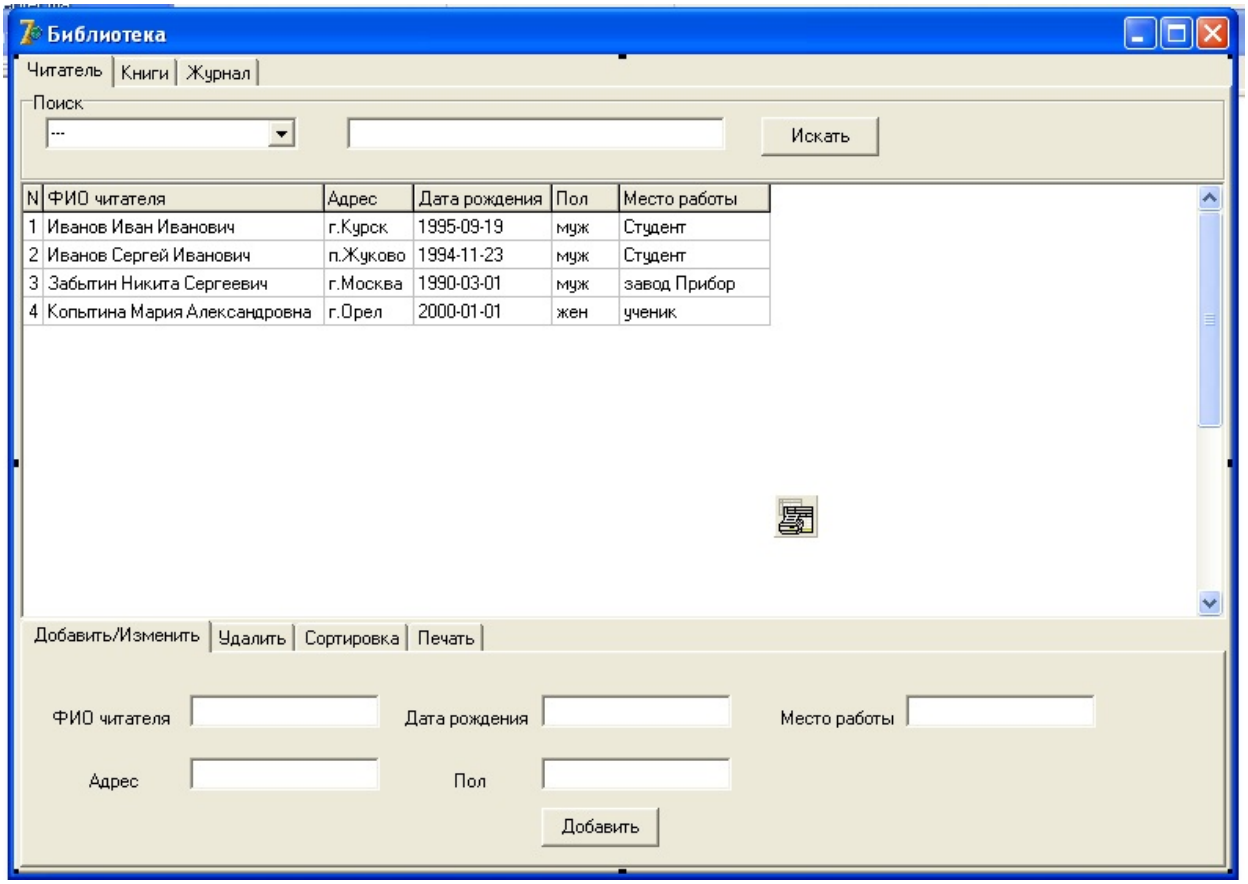


Рис. 18 - Добавление компонентов на лист “Добавить/Изменить”

Далее в свойствах DBEditEh1: DataSource выставляем DataModule2.chitatelDataSource, а в DataField - fio_chitatela.(Рис. 19)

Проделываем эту же операцию для оставшихся DBEditEh, с тем же значением поля DataSource, ставя соответствующими значениями поля DataField:

DBEditEh1=>DataField=>fio_chitatela(название столбца в таблице)
DBEditEh2=>DataField=>adress
DBEditEh3=>DataField=>data_rozdenia
DBEditEh4=>DataField=>pol
DBEditEh5=>DataField=>mesto_raboti

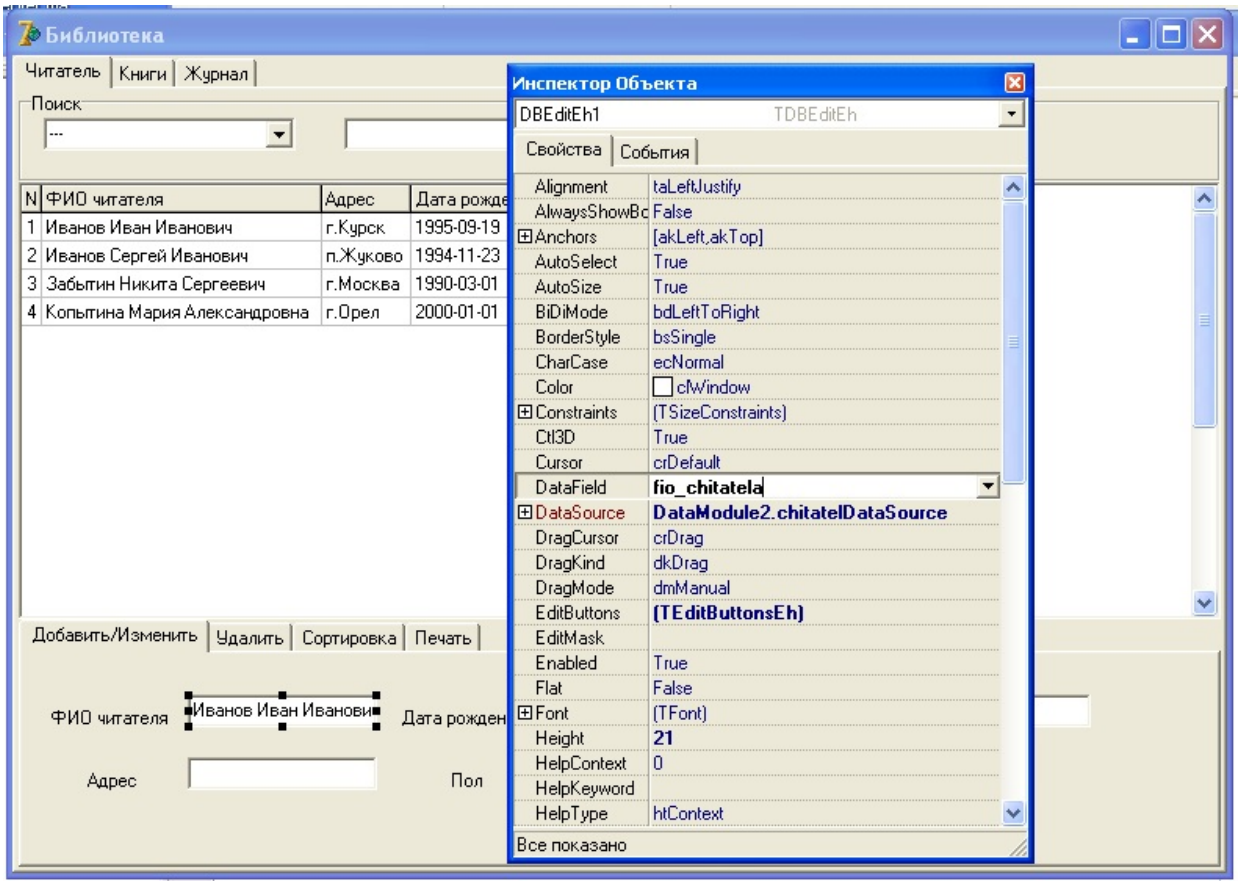


Рис. 19 - Свойства DBEditEh1

Далее заходим в SQL Server Management Studio и нажимаем создать запрос, в открывшемся окне создаём процедуру. Процедуру мы создаём для упрощения кода в Delphi:

```
create procedure new_chit (
@fio_chitatela      NVARCHAR(50),
@adress             NVARCHAR(50),
```

@data_rozdenia date,
@pol NVARCHAR(50),
@mesto_raboti NVARCHAR(50))
as
begin
insert into chitatel values ((select ISNULL(MAX(id)+1,1) from chitatel),
@fio_chitatela,@adress,
@data_rozdenia, @pol,
@mesto_raboti); End

При успешном выполнении данной процедуры нам выведет сообщение «Выполнение команд успешно завершено» (Рис. 20)

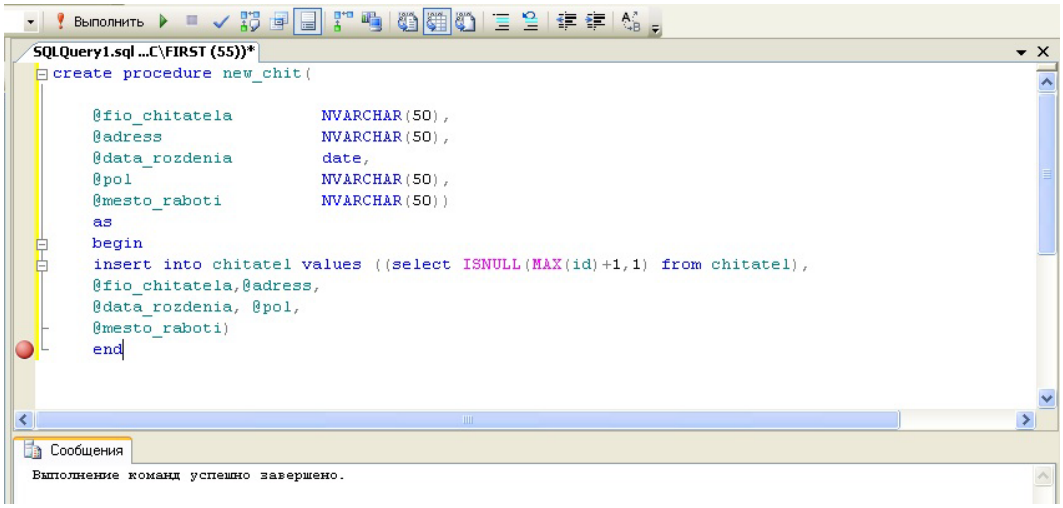


Рис. 20 - Успешное выполнение команд

Процедура создана. Закрываем sql server management studio. Переходим к Delphi. Открываем DataModule: Вид=> Формы=> DataModule2. Жмём ОК (Рис. 21)

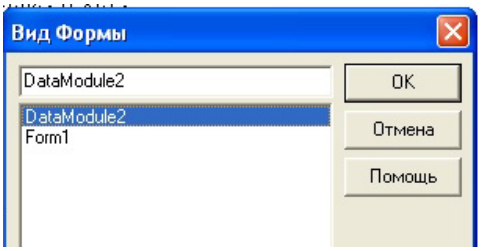


Рис. 21 - Открытие DataModule2

Добавляем на форму DataModule ещё один компонент TADOQuery сразу переименовываем его в HelpQuery (будет вспомогательным ADOQuery). В свойстве Connection выбираем ADOConnection1. Выбираем свойство SQL нажимаем на кнопку с «…». Пишем в этом окне любой запрос, т.к. это вспомогательной компонент (Например: select * from chitatel) и нажимаем ОК (Рис. 22). Ставим свойство ACTIVE в положение TRUE

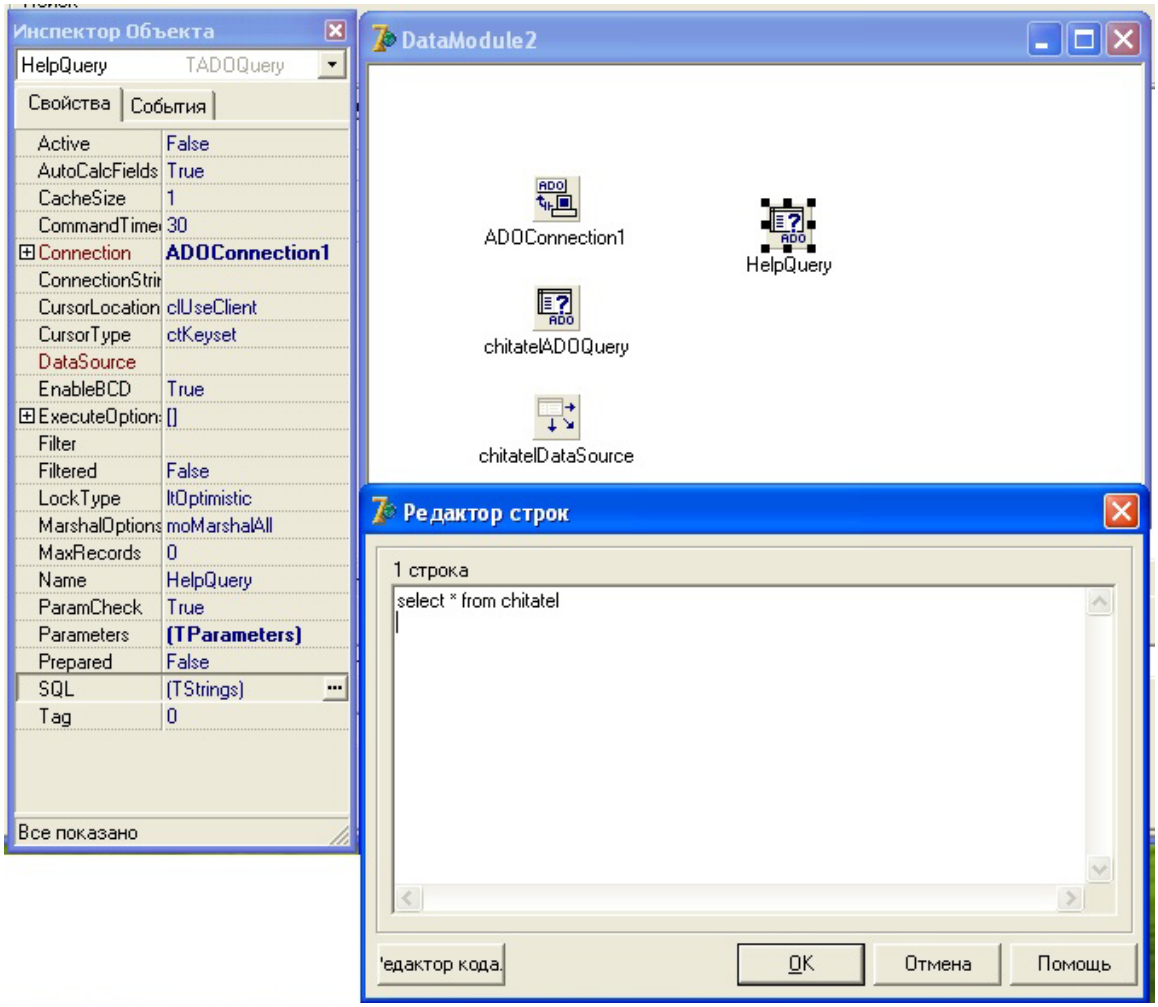


Рис. 22 - Свойства HelpQuery

Сделаем двойной клик на кнопке “Добавить” и вставим в процедуру обработки нажатия кнопки следующий код:

```
procedure TForm1.Button2Click(Sender: TObject);
var
fio_chitatela,adress,data_rozdenia,pol,mesto_raboti:string;
begin
```

```
fio_chitatela:=quotedstr(unit1.Form1.DBEditEh1.Text);
adress:=quotedstr(unit1.Form1.DBEditEh2.Text);
data_rozdenia:=quotedstr(unit1.Form1.DBEditEh3.Text);
pol:=quotedstr(unit1.Form1.DBEditEh4.Text);
mesto_raboti:=quotedstr(unit1.Form1.DBEditEh5.Text);
with unit2.DataModule2.HelpQuery do
begin close;
SQL.Clear;
SQL.ADD
('exec new_chit'+fio_chitatela+','+adress+',
'+data_rozdenia+','+pol+','+mesto_raboti);
ExecSQL;
Unit2.DataModule2.chitatelADOQuery.Active:=False;
Unit2.DataModule2.chitatelADOQuery.Active:=True;
end; end;
```

Для изменения записей таблицы достаточно просто выбрать нужную строку таблицы, значения столбцов выбранной строки отобразятся в DBEditEh записываем другие значения, после нажатия на любую часть формы значения поменяются, также значения можно менять простым нажатием на нужное поле в DBGridEh.

Приступаем к созданию процедуры удаления. Нажимаем на лист Удалить компоненты PageControl. Выставляем 2 элемента: кнопку Button из вкладки Standart и DBComboBoxEh из вкладки ADO.

В свойствах DBComboBoxEh DataSource выбираем DataModule2.chitatelDataSource DataField – fio_chitatela. (Рис. 24)

Для удаления нужно создать дополнительную форму для подтверждения удаления. Форма может выглядеть примерно вот так (Рис. 23)

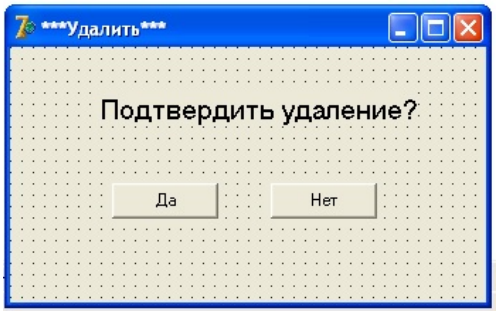


Рис. 23 - Форма подтверждения удаления

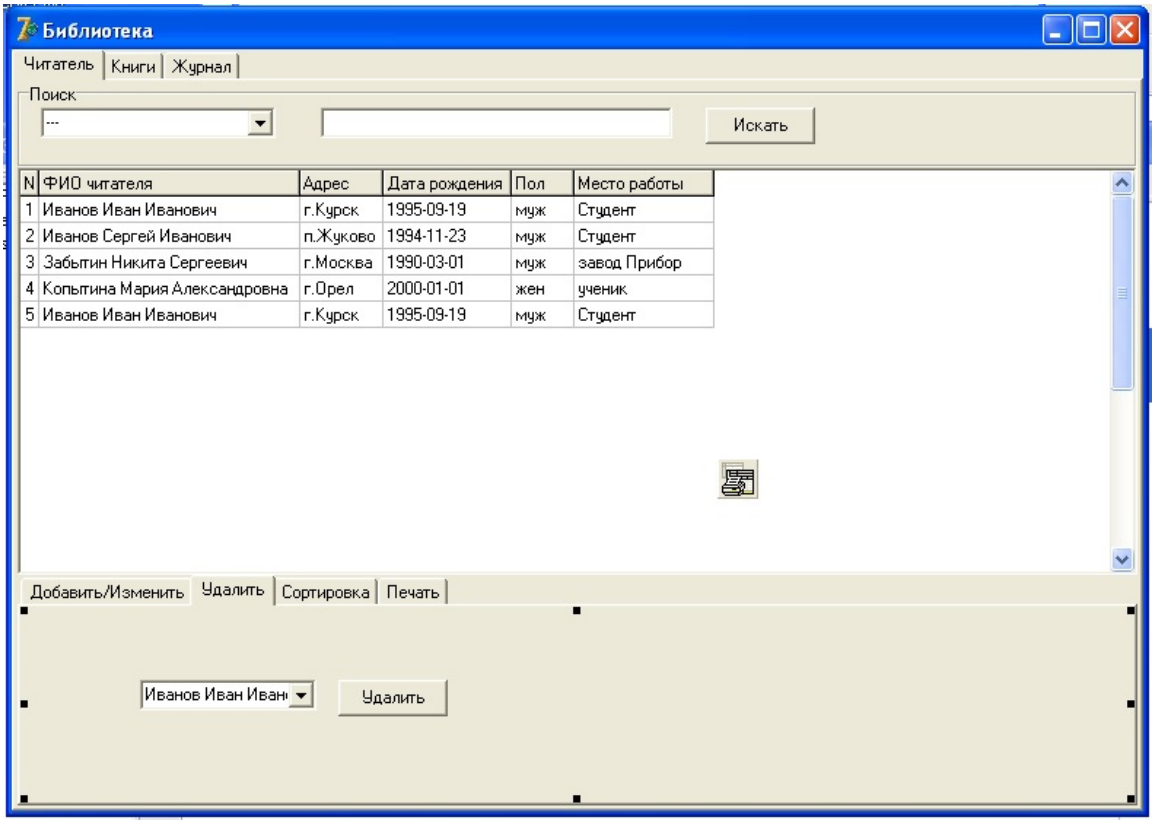


Рис. 24 - Добавление компонентов на лист “Удалить”

Кнопка “Нет” закрывает форму и имеет следующий код:

```
procedure TForm3.Button2Click(Sender: TObject);
begin
close;
end;
```

Кнопка “Да” имеет следующий код:

```
procedure TForm3.Button1Click(Sender: TObject);
begin
DataModule2.chitatelADOQuery.Delete;
close;
end;
```

Сделаем двойной клик на кнопке “Удалить” и вставим в процедуру обработки нажатия кнопки следующий код:

```
procedure TForm1.Button3Click(Sender: TObject);
begin
Form3.Showmodal;
end;
```

Перейдем к процедуре сортировки. В качестве примера сделаем сортировку по ФИО читателя, на основе примера можно будет сделать все сортировки какие необходимы. Итак, на Форме компонента PageControl (лист Сортировка) помещаем компонент из вкладки Ehlib DBCheckBoxEh и в его свойствах DataSource и DataField выбираем соответствующие значения DataModule2.chitatelDataSource и fio_chitatela. Также меняем Caption (Рис. 25)

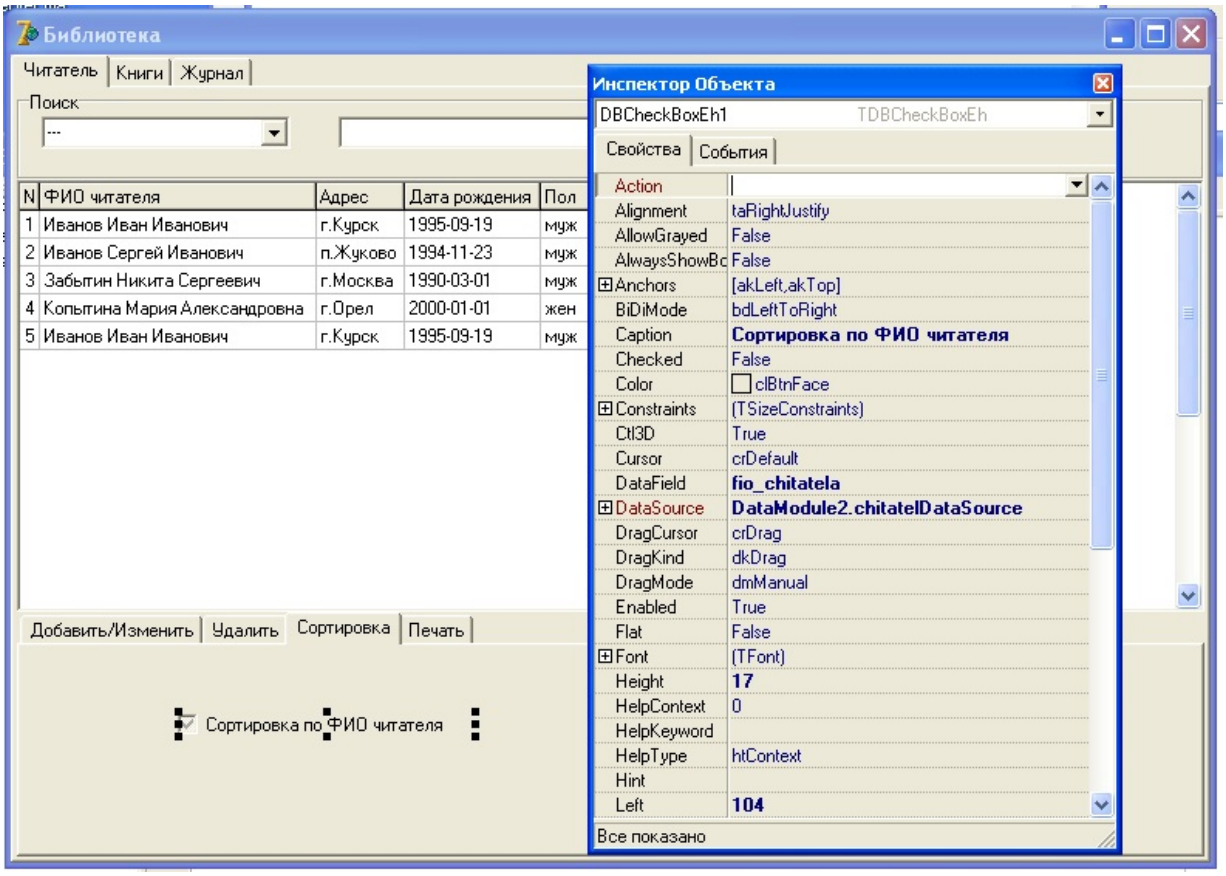


Рис. 25 - DBCheckBoxEh1 и его свойства

Сделаем двойной клик на DBCheckBoxEh “Сортировать по ФИО читателя” и вставим в процедуру обработки нажатия кнопки следующий код:

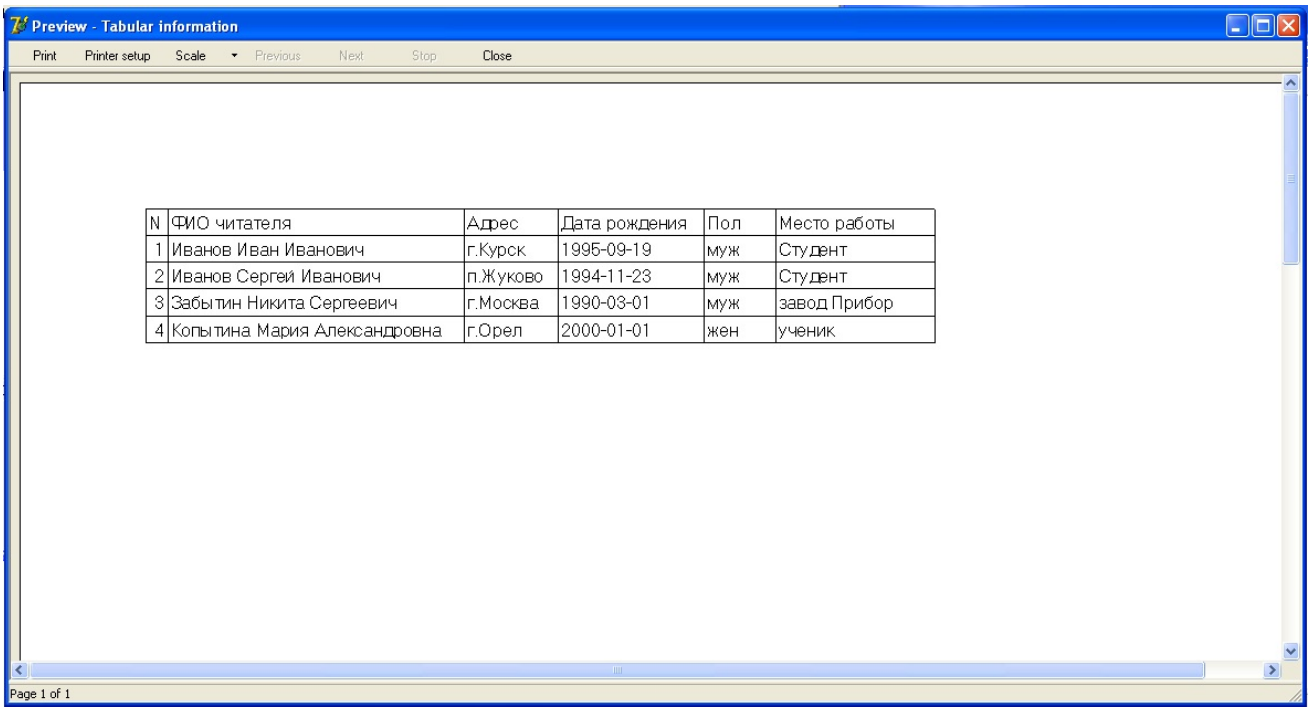
```
procedure TForm1.DBCheckBoxEh1Click(Sender: TObject);
begin
if unit1.Form1.DBCheckBoxEh1.Checked=true then
begin
with unit2.DataModule2.chitatelADOQuery do
begin
```

```
close;
SQL.Clear;
SQL.ADD('select * from chitatel order by fio_chitatela');
open;
end;
end;
end;
```

При помощи компонента PrintDBGridEh опишем процесс вывода результата запроса из базы данных на принтер. В параметре DBGridEh у нас уже стоит нужный нам DBGridEh. Просто создадим процедуру для кнопки “РАСПЕЧАТАТЬ”, вставив туда нижеследующий код:

```
procedure TForm1.Button4Click(Sender: TObject);
begin
PrintDBGridEh1.SetSubstitutes(['%(Today)',DateToStr(Now)]);
PrintDBGridEh1.Preview;
end;
```

Результатом работы кода является появление окна с preview-видом таблицы, где можно произвести необходимые настройки принтера (Рис. 26)



N	ФИО читателя	Адрес	Дата рождения	Пол	Место работы
1	Иванов Иван Иванович	г.Курск	1995-09-19	муж	Студент
2	Иванов Сергей Иванович	п.Жуково	1994-11-23	муж	Студент
3	Забытин Никита Сергеевич	г.Москва	1990-03-01	муж	завод Прибор
4	Копытина Мария Александровна	г.Орел	2000-01-01	жен	ученик

Рис. 26 - Результат работы кода

Данный шаг нужно повторить необходимое число раз для необходимого числа компонент DBGridEh и создать необходимое число компонент PrintDBGridEh.

Следующая процедура - создание произвольных запросов. Для этого открываем DataModule2 и помещаем 2 вспомогательных компонента ADOQuery1(в свойстве Connection выбираем ADOConnection1 в свойстве SQL пишем любой запрос, ставим значение ACTIVE- True) и DataSource1(выставляем в свойстве DataSet: ADOQuery1). (Рис. 27)

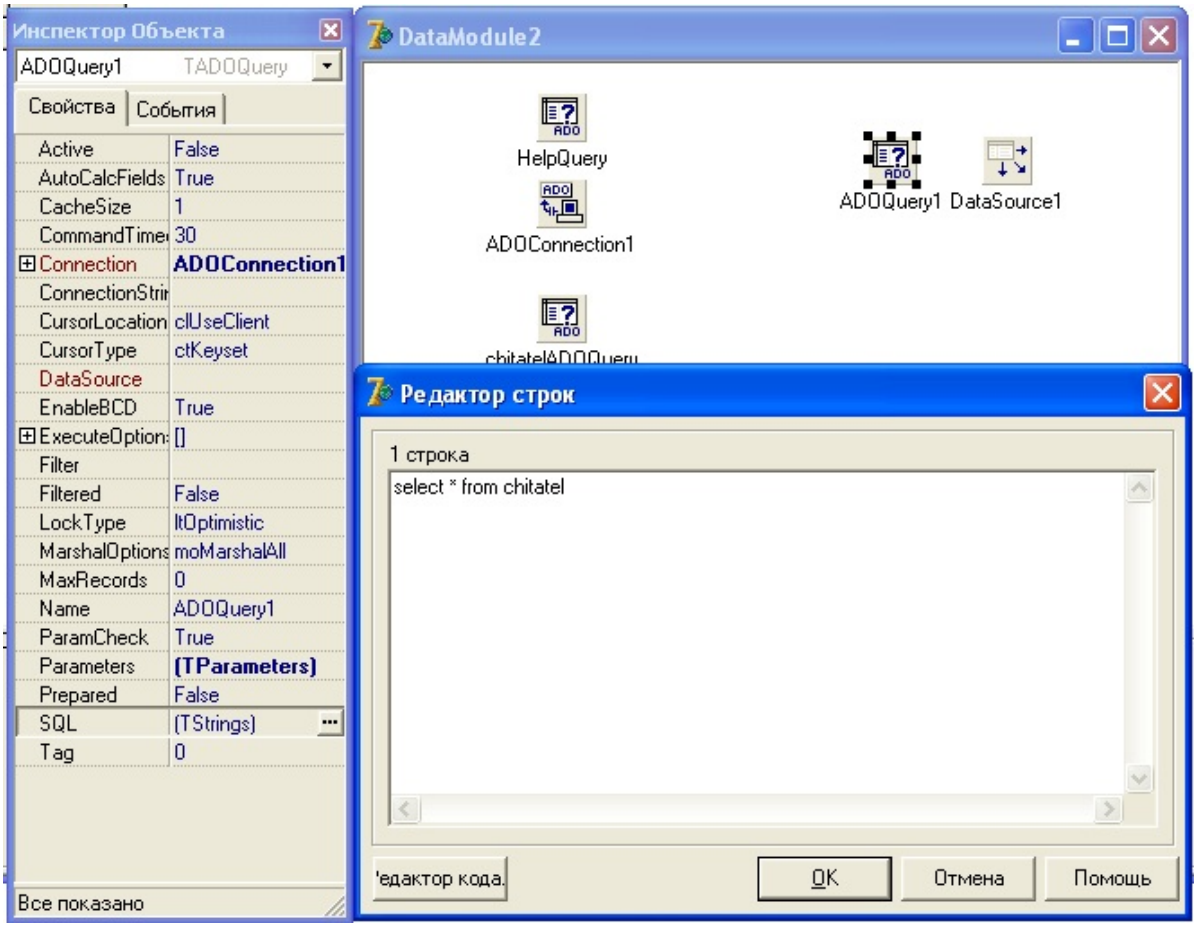


Рис. 27 - Добавление и настройка компонентов ADOQuery1 и DataSource1

Выделим главный PageConrol из пункта и создадим новый лист (Произвольная таблица). Ставим на форму GroupBox из вкладки Standart и меняем Caption “Произвольный запрос”. Помещаем на компонент GroupBox, компонент Memo из вкладки Standart и в свойстве Memo ScrollBars выставляем значение: ssVertical. Ставим 2 BUTTON из вкладки Standart и меняем Caption.

Кнопки будут “OK” и “РАСПЕЧАТАТЬ” выставляем на форму DBGridEh из вкладки Ehlib (в свойствах ставим DataSOurce: DataModule2.DataSource1, Align: alClient).

Из вкладки Ehlib помещаем на любую область формы компонент: PrintDBGridEh в параметре DBGridEh выберем нужный нам DBGridEh. (Рис. 28)

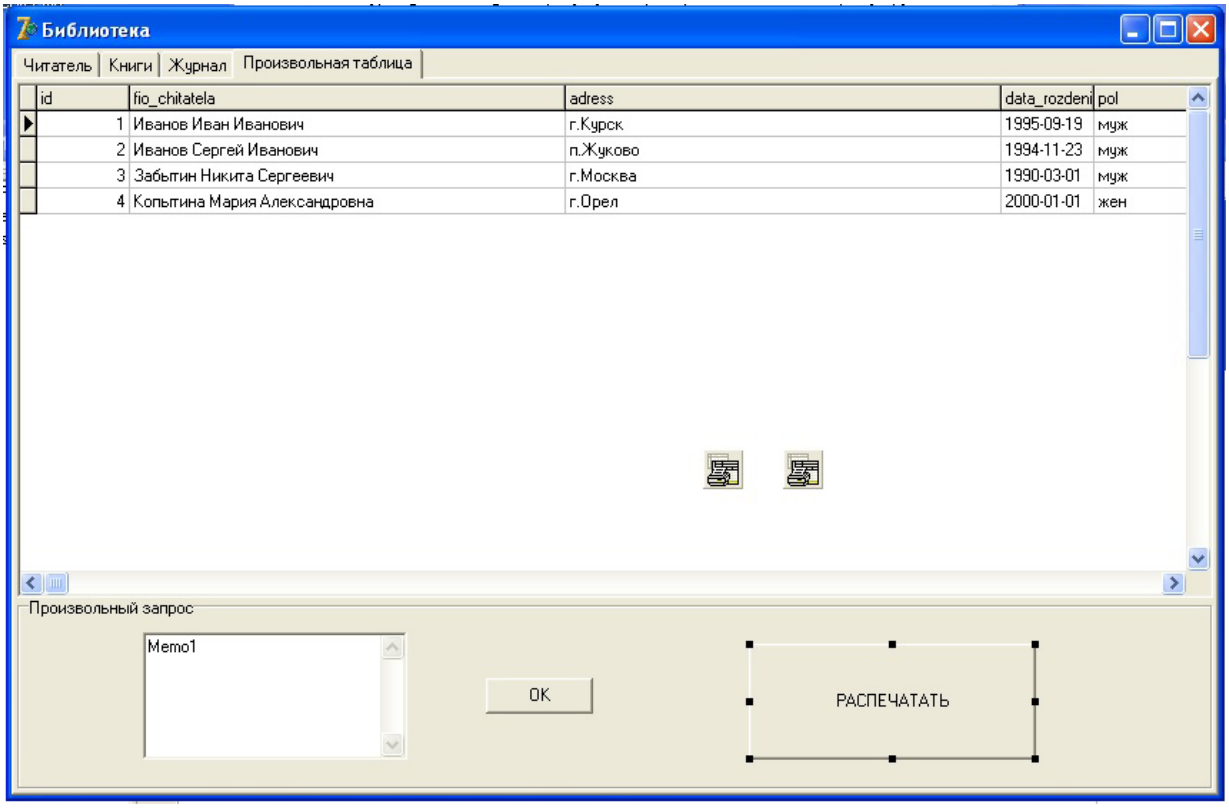


Рис. 28 - Добавление компонентов на лист “Произвольная таблица”

Сделаем двойной клик на кнопку “РАСПЕЧАТАТЬ” и вставим в процедуру обработки нажатия кнопки следующий код:

```
procedure TForm1.Button6Click(Sender: TObject);
begin
PrintDBGridEh2.SetSubstitutes(['%(Today)',DateToStr(Now)]);
PrintDBGridEh2.Preview;
end;
```

Сделаем двойной клик на кнопку “OK” и вставим в процедуру обработки нажатия кнопки следующий код:

```
procedure TForm1.Button5Click(Sender: TObject);
begin
```

```
DataModule2.ADOQuery1.close;
DataModule2.ADOQuery1.SQL.Clear;
DataModule2.ADOQuery1.Active:=false;
DataModule2.ADOQuery1.SQL.ADD("+Memo1.text+");
DataModule2.ADOQuery1.Active:=true; end;
```

Данная процедура позволит нам вводить любой запрос в Memo результат которого будет отображаться в PageControl (лист Произвольная таблица).

Таким образом мы соединили базу данных sql server management studio с Delphi 7. Связали базу данных с делфи при помощи компонентов ADO.

Для того чтобы в автоматическом режиме открывалось окно Свойство связи с данными (где мы выбираем провайдера, а именно Microsoft OLE DB Provider for SQL Server и указываем имя сервера; имя пользователя и пароль (который мы создавали в sql server management studio для нашей базы данных), выбираем нашу базу данных при запуске базы данных с помощью Delphi. (Рис. 29)

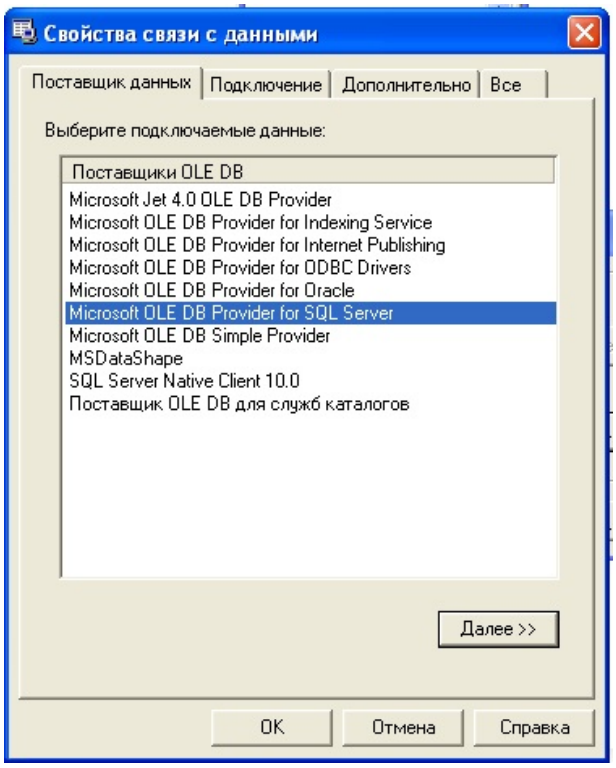


Рис. 29 - Настройка соединения с базой данных

Открываем DataModule2 выбираем компонент ADOConnection1=> События=> OnWillConnect жмём дважды по пустому полю и записываем в процедуру обработки нажатия следующий код:

```
procedure TDataModule2.ADOConnection1WillConnect(
Connection: TADOConnection;
var ConnectionString, UserID,
Password: WideString;
var ConnectOptions: TConnectOption;
var EventStatus: TEventStatus);
begin
ADODB.PromptDataSource(0,'');
end;
```

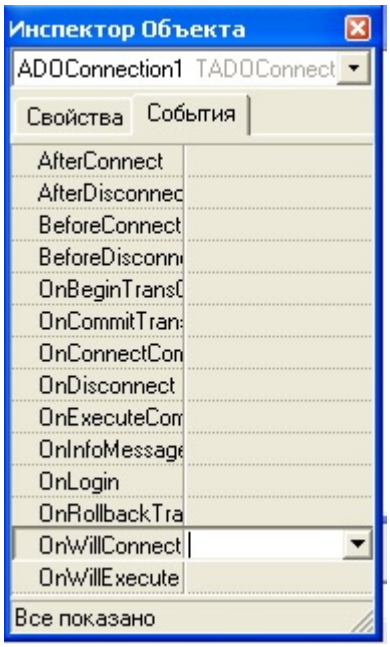


Рис.30 – События компонента ADOConnection1

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. С помощью каких компонент можно получить прямой доступ к базе данных через среду разработки?
2. Каким образом и какими средствами языка SQL можно выполнить поиск в базе данных?
3. Каким образом и какими средствами языка SQL можно добавить новую запись в базу данных?
4. Каким образом и какими средствами языка SQL можно изменить существующую запись в таблице базы данных?
5. Каким образом и какими средствами языка SQL можно удалить существующую запись в таблице базы данных?
6. С помощью библиотеки каких компонент можно получить отчет таблиц БД?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Все о SQL и клиент/серверных технологиях [электронный ресурс]: /Internet. - http://sql.ru, 2006.
2. Все о Delphi [электронный ресурс]: /Internet. - http://delphisource.ru, 2007.
3. Все о Delphi от Чертенка [электронный ресурс]: /Internet. - http://delphi.chertenok.ru, 2007.
4. Мастера Delphi (исходники, документация) [электронный ресурс]: /Internet. - http://delphi.master.ru, 2006.
5. RuLooper – YouTube http://www.youtube.com/user/RuLooper/videos
6. SQL-запросы в Delphi http://www.codenet.ru/progr/delphi/stat/SQL-Delphi.php
7. SQL в Delphi - Delphi Sources FAQ http://www.delphisources.ru/pages/faq/base/sql_in_delphi.html