МИНОБРНАУКИ РОССИИ
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**
**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**
**БОРИСОГЛЕБСКИЙ ФИЛИАЛ**
**(БФ ФГБОУ ВО «ВГУ»)**

# МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ
# Информационные системы в машиностроении

**1. Код и наименование направления подготовки:**

15.03.01 Машиностроение

**2. Профиль подготовки:**

Технологии, оборудование и автоматизация машиностроительных производств

**3. Квалификация (степень) выпускника:**

Бакалавр

**4. Форма обучения:**

Очная, заочная

**5. Кафедра, отвечающая за реализацию дисциплины:**

кафедра прикладной математики, информатики, физики и методики их преподавания

**6. Составитель(и):**

Хвостов М.Н., кандидат физико-математических наук

## 7. Методические указания для обучающихся по освоению дисциплины

Приступая к изучению учебной дисциплины, прежде всего обучающиеся должны ознакомиться с учебной программой дисциплины. Электронный вариант рабочей программы размещён на сайте БФ ВГУ.

Обучающиеся должны иметь четкое представление о:

- перечне и содержании компетенций, на формирование которых направлена дисциплина;
- основных целях и задачах дисциплины;
- планируемых результатах, представленных в виде знаний, умений и навыков, которые должны быть сформированы в процессе изучения дисциплины;
- количестве часов, предусмотренных учебным планом на изучение дисциплины, форму промежуточной аттестации;
- количестве часов, отведенных на контактную и на самостоятельную работу;
- формах контактной и самостоятельной работы;
- структуре дисциплины, основных разделах и темах;
- системе оценивания ваших учебных достижений;
- учебно-методическом и информационном обеспечении дисциплины.

Знание основных положений, отраженных в рабочей программе дисциплины, поможет обучающимся ориентироваться в изучаемом курсе, осознавать место и роль изучаемой дисциплины, строить свою работу в соответствии с требованиями, заложенными в программе.

Основными формами контактной работы по дисциплине являются лекции и лабораторные работы, посещение которых обязательно для всех студентов.

В ходе лекционных занятий следует не только слушать излагаемый материал и кратко его конспектировать, но очень важно участвовать в анализе примеров, предлагаемых преподавателем, в рассмотрении и решении проблемных вопросов, выносимых на обсуждение. Необходимо критически осмысливать предлагаемый материал, задавать вопросы как уточняющего характера, помогающие уяснить отдельные излагаемые положения, так и вопросы продуктивного типа, направленные на расширение и углубление сведений по изучаемой теме, на выявление недостаточно освещенных вопросов, слабых мест в аргументации и т.п.

В ходе выполнения лабораторных работ студент выполняет задания, содержащиеся в методическом пособии дисциплины в соответствии с имеющимися указаниями. Далее студент самостоятельно выполняет индивидуальное задание.

Обязательно следует познакомиться с критериями оценивания каждой формы контроля – это поможет избежать недочетов, снижающих оценку за работу.

При подготовке к промежуточной аттестации необходимо повторить пройденный материал в соответствии с учебной программой, примерным перечнем вопросов, выносящихся на зачет. Рекомендуется использовать конспекты лекций и источники, перечисленные в списке литературы в рабочей программе дисциплины, а также ресурсы электронно-библиотечных систем. Необходимо обратить особое внимание на темы учебных занятий, пропущенных по разным причинам. При необходимости можно обратиться за консультацией и методической помощью к преподавателю.

## 8. Методические материалы для обучающихся по освоению теоретических вопросов дисциплины

| № | Тема лекции | Рассматриваемые вопросы |
|---|---|---|
| 1 | Основные понятия. | Данные и модели данных. Формальное определение модели данных. Модель плоских файлов.<br>Структуры данных. Понятие знака и типа. Абстракция – как основной способ структуризации данных. Обобщение и агрегация, как способы абстракции. Классификация и обобщение в узком смысле. Экземпляризация и специализация. |

| | | |
|---|---|---|
| | | Декомпозиция. Формы хранения данных: множество, комплекс, кортеж, отношение. Понятия домена и атрибута. Экстенсионал и интенсионал множества, отношения и БД. Табличное представление данных.<br>Процедуры баз данных. Транзакции, триггеры. |
| 2 | Классификация моделей данных. | Понятие модели данных. Типы структур данных. Операции над данными. Ограничения целостности. Сетевая модель данных (СМД). Иерархическая модель данных (ИМД). |
| 3 | Технология клиент-сервер. | Преимущества и недостатки. FS, RDA, DBS,AS модели. |
| 4 | Реляционная модель. | Структуры данных. Определение отношения в 1НФ. Отношения, атрибуты, первичные ключи. Назначение реляционной модели. Правила трансформации данных из ER-модели в реляционную. 2-я и 3-я НФ. НФ Бойса-Кодда. Нормализация баз. |
| 5 | Проектирование ИС. | Способы проектирования ИС.<br>Последовательность создания информационной модели в классической методике. Концептуальная и логическая модели предметной области.<br>Этапы проектирования БД. Выбор СУБД, анализ предметной области. Деловая модель.<br>Инфологическое моделирование. Описание предметной области. Получение логической модели предметной области в нужной модели данных.<br>Физическое проектирование БД. Создание адекватного состояния БД. Обучение персонала.<br>Объектно-ориентированный подход к проектированию ИС и предпосылки к созданию объектно-ориентированных БД.<br>Понятие о CASE-средствах. |
| 6 | Структура языка запросов SQL. | Операторы языка: CREATE, INSERT, UPDATE, DELETE, ALTER, SELECT. Использование SQL для выборки данных из таблицы: операторы в условиях IN, BETWEEN, LIKE, IS NULL; определение выборки – предложение WHERE; создание SQL- запросов. Выборка данных из одной и нескольких таблиц. Функции агрегирования. |
| 7 | Администрирование баз данных. Иерархия прав доступа. | Обзор возможностей и особенностей различных СУБД. Методы хранения и доступа к данным. Работа с внешними данными с помощью технологии ODBC (BDE, ADO). |
| 8 | Создание БД в среде Delphi | Этапы разработки приложений для работы с информационными системы средствами объектно-ориентированного языка программирования, на примере Delphi. Проектирование логической модели данных. Определение входных – выходных данных, проектирование интерфейса пользователя. выделение интерфейсных элементов. |
| 9 | Основные операции над набором данных. | Навигация, поиск, добавление, удаление.<br>События, свойства, методы компонентов DataBase, Table, Query. |
| 10 | Компоненты доступа и отображения данных. | Палитры:, DataControl. События, свойства, методы компонентов, DataSource, DBEdit, DBComboBox, DBListBox, DBGrid, DBNavigator. |
| 11 | Особенности языка SQL в Delphi. | Создание статических и параметрических запросов. Динамичесое создание запросов. Процедуры и методы: FieldByName, ParamByName, Open, Close, Execute. |
| 12 | Перспективы развития Баз данных. | Интеллектуальный анализ данных. Объектно-ориентированные базы данных. Темпоральные баз данных. Дедуктивные баз данных. Взаимодействие Web-технологии и баз данных. |

## 9. Методические материалы для обучающихся по подготовке к практическим/лабораторным занятиям

| № | Тема занятия | Рассматриваемые вопросы |
|---|---|---|
| 1 | Администрирование баз данных. Иерархия прав доступа. | Создание псевдонима и таблиц базы данных. |
| 2 | Создание БД в среде Delphi | Создание псевдонима и таблиц базы данных. |
| 3 | Основные операции над набором данных. | Создание главной формы. Создание справочной формы. |
| 4 | Компоненты доступа и отображения данных. | Создание справочной формы с использованием запроса. |
| 5 | Особенности языки SQL в Delphi. | Создание учетной формы. |

## 10. Тематика рефератов/докладов/эссе, методические рекомендации по выполнению контрольных и курсовых работ, иные материалы

**Примерный перечень вопросов к дифференцированному зачету по дисциплине «Информационные системы»**Данные и модели данных. Формальное определение модели данных. Модель плоских файлов.

2. Структуры данных. Понятие знака и типа. Абстракция – как основной способ структуризации данных. Обобщение и агрегация, как способы абстракции.
3. Классификация и обобщение в узком смысле. Экземпляризация и специализация. Декомпозиция.
4. Формы хранения данных: множество, комплекс, кортеж, отношение.
5. Понятия домена и атрибута. Экстенсионал и интенсионал множества, отношения и БД.
6. Табличное представление данных.
7. Процедуры баз данных. Транзакции, триггеры.
8. Понятие модели данных. Типы структур данных. Операции над данными. Ограничения целостности. Сетевая модель данных (СМД). Иерархическая модель данных (ИМД).
9. Преимущества и недостатки. FS, RDA, DBS,AS модели.
10. Структуры данных. Определение отношения в 1НФ. Отношения, атрибуты, первичные ключи.
11. Назначение реляционной модели. Правила трансформации данных из ER-модели в реляционную.
12. 2-я и 3-я НФ. НФ Бойса-Кодда. Нормализация баз.
13. Способы проектирования ИС.
14. Последовательность создания информационной модели в классической методике. Концептуальная и логическая модели предметной области.
15. Этапы проектирования БД. Выбор СУБД, анализ предметной области. Деловая модель.
16. Инфологическое моделирование. Описание предметной области. Получение логической модели предметной области в нужной модели данных.
17. Физическое проектирование БД. Создание адекватного состояния БД. Обучение персонала.
18. Объектно-ориентированный подход к проектированию ИС и предпосылки к созданию объектно-ориентированных БД.
19. Понятие о CASE-средствах.

20. Операторы языка: CREATE, INSERT, UPDATE, DELETE, ALTER, SELECT. Использование SQL для выборки данных из таблицы: операторы в условиях IN, BETWEEN, LIKE, IS NULL; определение выборки – предложение WHERE; создание SQL- запросов.
21. Выборка данных из одной и нескольких таблиц. Функции агрегирования.
22. Обзор возможностей и особенностей различных СУБД. Методы хранения и доступа к данным. Работа с внешними данными с помощью технологии ODBC (BDE, ADO).
23. Этапы разработки приложений для работы с информационными системы средствами объектно-ориентированного языка программирования, на примере Delphi.
24. Проектирование логической модели данных. Определение входных – выходных данных, проектирование интерфейса пользователя. выделение интерфейсных элементов.
25. Навигация, поиск, добавление, удаление.
26. События, свойства, методы компонентов DataBase, Table, Query.
27. Палитры:, DataControl. События, свойства, методы компонентов, DataSource, DBEdit, DBComboBox, DBListBox, DBGrid, DBNavigator.
28. Создание статических и параметрических запросов. Динамичесое создание запросов. Процедуры и методы: FieldByName, ParamByName, Open, Close, Execute.
29. Интеллектуальный анализ данных. Объектно-ориентированные базы данных. Темпоральные баз данных. Дедуктивные баз данных. Взаимодействие Web-технологии и баз данных.

**Методические рекомендации по выполнению лабораторных работ по дисциплине «Информационные системы»**

**Создание псевдонима и таблиц базы данных**

Перед работой необходимо создать папку Sklad, в ней еще две папки Form и BD. При работе с таблицами база данных (БД) считается каталог на диске, в котором хранятся файлы таблиц, индексов, и т.д. Обращение к БД из программы осуществляется по псевдониму (алиасу). Псевдоним БД создается с помощью утилититы BDE Administrator.

Для создания псевдонима БД необходимо:

- запустить BDE Administrator (Пуск - Программы - Borland Delphi 7 - BDE Administrator);
- выбрать в главном меню пункт Objekt - New;
- в появившемся окне указать тип БД (по умолчанию STANDART) и нажать кнопку Ok;
- появится окно администратора БД. В левой части окна находятся имена имеющихся БД и имя новой БД (по умолчанию STANDART 1). Переименуйте БД в Sklad;
- в строке PATH укажите путь к БД (путь до папки BD);
- сохраните созданный псевдоним.

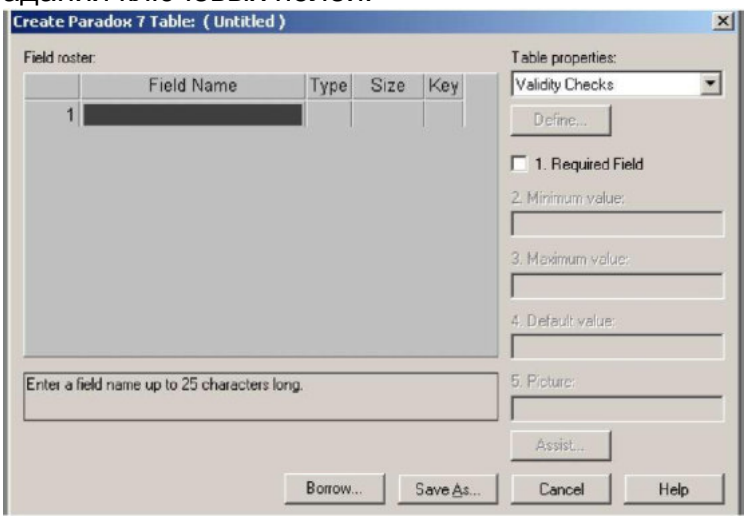
Для создания таблиц БД используется утилита Database Desktop. После запуска утилиты необходимо установить рабочий псевдоним. Для этого надо выбрать пункт главного меню File - Working Directory и в списке Aliases выбрать имя своего псевдонима. После чего можно приступать к созданию таблиц. Для создания таблицы БД необходимо:

- запустить Database Desktop (Пуск - Программы - Borland Delphi 7 - Database Desktop или если среда Delphi уже загружена, то командой Database Desktop из пункта меню Tools);

установите рабочий псевдоним. Обратитесь в пункт меню File - Working Directory и в списке Aliases выберите имя своего псевдонима. Щелкните по кнопке Ok;

- для создания таблиц выберите пункт меню File - New - Table... (по умолчанию таблиц создается в формате Paradox);
- в появившемся окне задать структуру таблицы.

В столбце Field Name задается имя поля, в Type - тип, Size - размер поля, Key - используется для задания ключевых полей.




Для таблицы tip

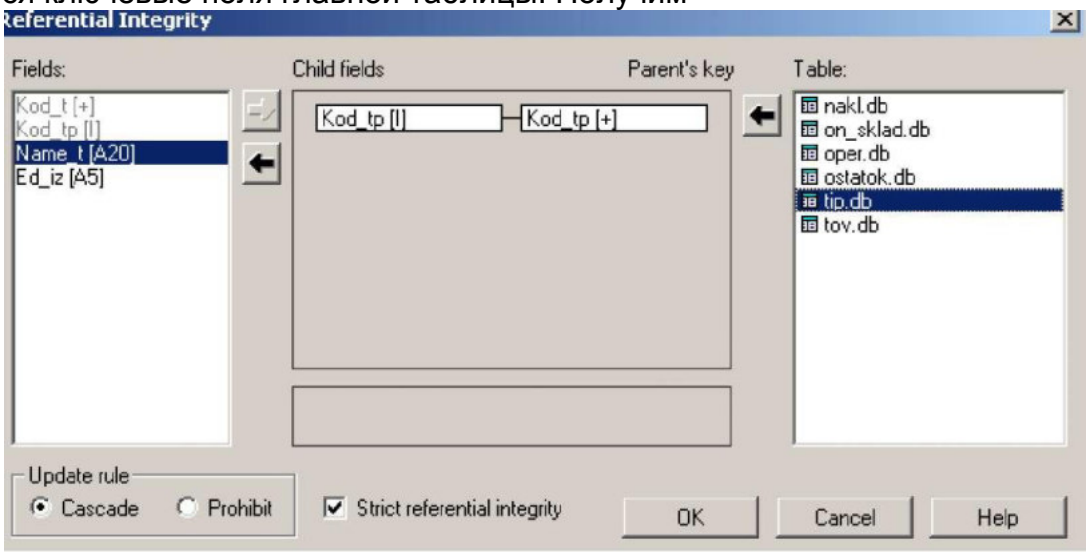
| | Field Name | Type | Size | Key |
|---|---|---|---|---|
| 1 | kod tp | + | | * |
| 2 | name_tp | A | 20 | |

Сохраните таблицу в папку BD, используя кнопку Save as..., назовите таблицу tip. Первыми идут поля, составляющие первичный ключ. **Эти поля всегда располагаются в начале таблицы.** Чтобы указать, что поле входит в первичный ключ надо в свойстве **Key** этого поля поставить Создайте таблицу tov (товары). Она имеет следующие поля:

| | Field Name | Type | Size | Key |
|---|---|---|---|---|
| 1 | kod t | + | | * |
| 2 | kod tp | I | | |
| 3 | name t | A | 25 | |
| 4 | ed iz | A | 5 | |

Сохраните таблицу, дав ей имя tov. Для создания связей необходимо *открыть подчиненную таблицу,* в нашем случае это таблица товары.

- откройте таблицу tov (пункт меню File - Open - Table...);
- чтобы изменить структуру таблицы, обратитесь в пункт меню Table - Restructure., или щелкнуть по кнопке Restructure на панели инструментов;
- с помощью списка Table Properties (свойства таблицы) связи таблицы. Выберите из списка Referential Integrity и щелкните по кнопке Define;
- *слева* приведен *список полей подчиненной таблицы, справа список таблиц БД.* Т.к. связываем таблицы tip и tov по полю kod_tp, то из списка полей выберите kod_tp, и нажатием кнопки со стрелкой вправо перенесите его в список Child fields. Из списка таблиц выберите таблицу tip (которая является "родительской" по отношению к tov), и после нажатием кнопки со стрелкой влево в список Parent's Key автоматически заносятся ключевые поля главной таблицы. Получим




Переключатели Update rules определяют вид каскадных воздействий на родительскую таблицу при изменении значения поля связи в дочерней таблице.

- щелкните по кнопке Ok, задайте *имя* связи;
- сохраните произведенные изменения;
- аналогичным образом создайте и свяжите остальные таблицы.
- **Типы полей формата Paradox**

| | |
|---|---|
| Alpha | строка длиной 1-255 байт, содержащая любые печатаемые символы |
| Number | числовое поле длиной 8 байт, значение которого может быть положительным и отрицательным. Диапазон чисел - от 10-308 до 10308 с 15 значащими цифрами |

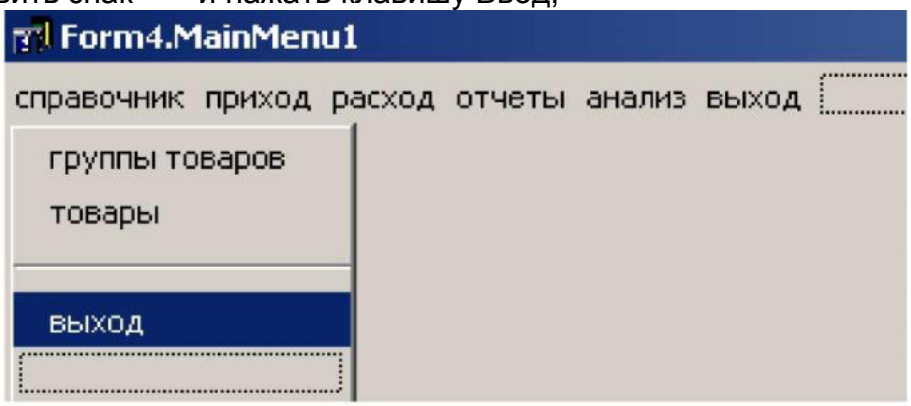
| | |
|---|---|
| $ (Money) | числовое поле, значение которого может быть положительным и |
| | которую удобней всего применять в триггерах) |
| Binary | поле, содержащее любую двоичную информацию. Может иметь любую длину. При этом размер, указываемый при создании таблицы, означает количество символов, сохраняемых в таблице (0-240) - остальные символы сохраняются в отдельном файле с расширением .MB. Это полнейший аналог поля BLOb в InterBase |
| Bytes | строка цифр длиной 1-255 байт, содержащая любые данные |
| | отрицательным. По умолчанию, является форматированным для отображения десятичной точки и денежного знака |
| Short | числовое поле длиной 2 байта, которое может содержать только целые числа в диапазоне от -32768 до 32767 |
| Long Integer | числовое поле длиной 4 байта, которое может содержать целые числа в диапазоне от -2147483648 до 2147483648 |
| # (BCD) | числовое поле, содержащее данные в формате BCD (Binary Coded Decimal). Скорость вычислений немного меньше, чем в других числовых форматах, однако точность - гораздо выше. Может иметь 0-32 цифр после десятичной точки |
| Date | поле даты длиной 4 байта, которое может содержать дату от 1 января 9999 г. до нашей эры - до 31 декабря 9999 г. нашей эры. Корректно обрабатывает високосные года и имеет встроенный механизм проверки правильности даты |
| Time | поле времени длиной 4 байта, содержит время в миллисекундах от полуночи и ограничено 24 часами |
| @ (Timestamp) | обобщенное поле даты длиной 8 байт - содержит и дату и время |
| Memo | поле для хранения символов, суммарная длина которых более 255 байт. Может иметь любую длину. При этом размер, указываемый при создании таблицы, означает количество символов, сохраняемых в таблице (1-240) - остальные символы сохраняются в отдельном файле с расширением .MB |
| Formatted Memo | поле, аналогичное Memo, с добавлением возможности задавать шрифт текста. Также может иметь любую длину. При этом размер, указываемый при создании таблицы, означает количество символов, сохраняемых в таблице (0-240) - остальные символы сохраняются в отдельном файле с расширением .MB. Однако, Delphi в стандартной поставке не обладает возможностью работать с полями типа Formatted Memo |
| Graphic | поле, содержащее графическую информацию. Может иметь любую длину. Смысл размера - такой же, как и в Formatted Memo. Database Desktop "умеет" создавать поля типа Graphic, однако наполнять их можно только в приложении |
| OLE | поле, содержащее OLE-данные (Object Linking and Embedding) - образы, звук, видео, документы - которые для своей обработки вызывают создавшее их приложение. Может иметь любую длину. Смысл размера - такой же, как и в Formatted Memo. Database Desktop |

| | |
|---|---|
| | "умеет" создавать поля типа OLE, однако наполнять их можно только в приложении. Delphi "напрямую" не умеет работать с OLE-полями, но это легко обходится путем использования потоков |
| Logical | поле длиной 1 байт, которое может содержать только два значения - T(true, истина) или F (false, ложь). Допускаются строчные и прописные буквы |
| +<br>(Autoincrement) | поле длиной 4 байта, содержащее нередактируемое (read-only) значение типа long integer. Значение этого поля автоматически увеличивается (начиная с 1) с шагом 1 - это очень удобно для создания уникального идентификатора записи (физический номер записи не может служить ее идентификатором, поскольку в Парадоксе таковой отсутствует. В InterBase также отсутствуют физические номера записей, но отсутствует и поле Autoincrement. Его с успехом заменяет встроенная функция Gen id, |

**Создание главной формы**

Эта форма будет раскрываться при запуске БД.

- создайте новую форму (при запуске среды Delphi автоматически будет создана новая форма);
- сохраните модуль и проект в папку Form, дав соответственно имена UMain и Sklad;
- в свойстве Caption укажите название формы «Учет товаров на складе», в свойстве WindowState укажите wsMaximized (при запуске окно формы будет раскрываться во весь
экран);
- расположите на форме компонент MainMenu с панели Standard. Этот компонент позволяет поместить главное меню в программу. При помещении MainMenu на форму это выглядит, как просто иконка. Иконки данного типа называют "невидимыми компонентом", поскольку они невидимы во время выполнения программы;
- свойство Items позволяет создать компоненты меню, названия вводятся в свойство Caption. Введите следующее меню разделитель можно сделать, если в свойстве Caption поставить знак <-> и нажать клавишу Ввод;




- щелкните по кнопке выход (в созданном меню);
- в раскрывшемся окне модуля напишите close;
- выход в подменю можно сделать аналогичным образом или перейти на закладку события (Events) и в событие OnClick указать *Click (* - имя «первого Выхода»);
- сохраните результаты работы;
- откомпилируйте проект и запустите его (Ctrl+F9).

На экране у вас появится меню, в котором будут работать только кнопки Выход. Проверьте работу кнопок.

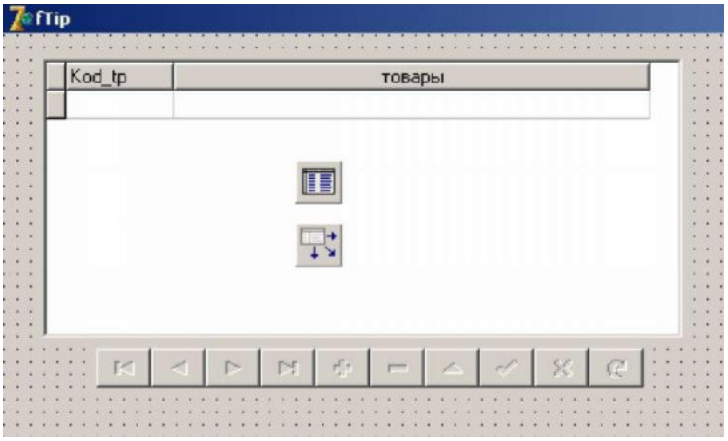
**Создание справочной формы**

**Форма "группы товаров"**

- создайте новую форму и сохраните модуль под именем UTip;
- в свойстве Caption укажите название формы - Группы товаров;
- в свойстве Name - FTip;
- для того чтобы присоединить форму FTip к главной форме, откройте главную форму;
- в объекте MainMenu выберите пункт группы товаров, щелкните по нему два раза. В появившемся окне модуля напишите FTip.showmodal;
- сохраните результаты работы;
- откомпилируйте и запустите проект. (Появиться сообщение, в котором спрашивают присоединить форму тип к главной форме, нажмите Да).

Рассмотрим работу с данными на основе компонента **Table** - обращается ко всей таблице для считывания, добавления, изменения или удаления данных, причем только к одной таблице (это невизуальный компонент, при запуске проекта его видно не будет).

- расположите на форме компонент Table с панели BDE;
- в свойстве DatabaseName укажите расположение БД. Это имя каталога, в котором расположены файлы БД или псевдоним, ссылающийся на этот каталог, т.е. в нашем случае Sklad.
- в свойстве TableName выберите таблицу БД tip.db;
- в свойстве Name укажите имя компонента tTip;
- «щелкните» два раза мышкой по компоненту TTable появиться окно редактора полей fTip.tTip;
- «щелкните» правой клавишей мыши в окне редактора полей flip.tlip и выберите пункт Add fields.. .(добавить все поля...);
- расположите на форме компонент TDataSource с панели инструментов Data Access. Класс TDataSource используется в качестве проводника между TTable и компонентами, визуализирующими данные, типа TDBGrid, TDBEdit и TDBComboBox;
- в свойстве DataSet укажите соответствующий TTable, т.е. в нашем случае это tTip;
- в свойстве Name укажите имя компонента dsTip;
- расположите на форме компонент TDBGrid с панели инструментов Data Controls, этот компонент служит для визуализации данных находящихся в таблице tip;
- отметьте свойство DataSource, указав в нем имя связующего компонента dsTip;
- откройте редактор полей два раза «щелкнув» мышкой по TDBGrid. Нажмите кнопку Add all fields...;
- выделите поле name_tp. В свойстве Alignment установите выравнивание по центру, «щелкните» по свойству Title раскроется дополнительный набор свойств. В пункте Caption напишите товары;
- в событие OnActivate для формы fTip «щелкнит» два раза мышкой, в раскрывшемся окне модуля напишите tTip.Open (или fTip.Active:=true);
- в событие OnClose для формы fTip «щелкнит» два раза мышкой, в раскрывшемся окне модуля напишите tTip.Close (или fTip.Active:=false);
- расположите на форме компонент DBNavigator с панели инструментов Data Controls;
- отметься свойство DataSource - dsTip;



Форма FTip

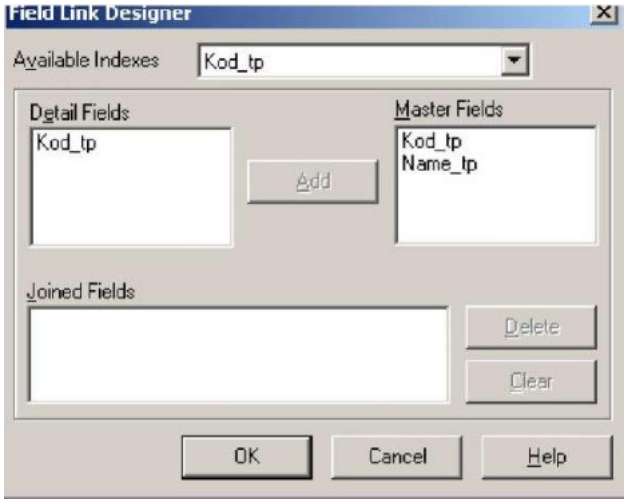
- сохраните результаты работы;
- откомпилируйте и запустите проект;
- внесите несколько записей в таблицу типы товаров, например соки-воды, овощи и фрукты и.т.д.

**Форма "товары"**

- создайте новую форму. Сохраните модуль под именем UTov;
- присоедините форму товары к главной форме (см. присоединение формы fTip);
- расположите на форме по два компонента TTable, TDataSource, TDBGrid;
- отметьте следующие свойства

выберите из левого и правого списка Kod_tp и нажмите кнопку Add, щелкните по кнопке Ok;

| **№ п/п** | **Свойство** | **TTable первый** | **TTable второй** |
|---|---|---|---|
| 1 | DatabaseName | Sklad | Sklad |
| 2 | TableName | tip.db | tov.db |
| 3 | Name | ttip | ttov |
| **№ п/п** | **Свойство** | **TDataSource первый** | **TDataSource второй** |
| 1 | DataSet | ttip | ttov |
| 2 | Name | dstip | dstov |
| **№ п/п** | **Свойство** | **TDBGrid первый** | **TDBGrid второй** |
| 1 | DateSource | dstip | dstov |

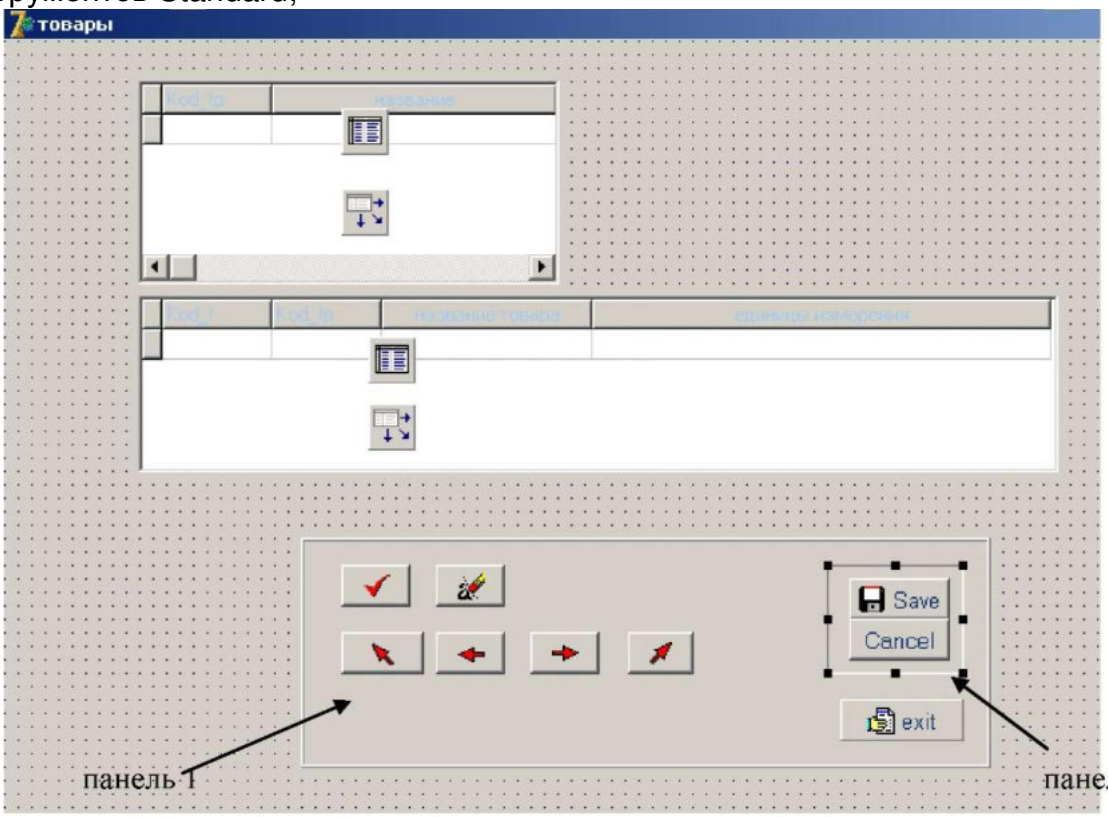


- добавьте все поля через редактор полей (для компонентов TTable, TDBGrid);

- для компонента ttov отметьте еще свойство MasterSource - dstip, обратившись к свойству MasterFields откроется окно
- • при активации формы в модуле откройте и закройте ttov и ttip

| | при активации формы | при закрытии формы |
|---|---|---|
| запись в модуле | ttip.Open;<br>ttov.Open; | ttip.Close; ttov.Close; |

- сохраните и запустите проект, внесите несколько записей в таблицу товары;
- расположите на форме компоненты Button (кнопка) с панели инструментов Standard или TBitBtn (кнопка с рисунком, его можно добавить, обратившись к свойству Glyph) с панели инструментов Additional и два компонента Panel с панели инструментов Standard;




у панелей уберите название, оставив свойство Caption пустым, у панели 2 в свойстве Name укажите имя pi;

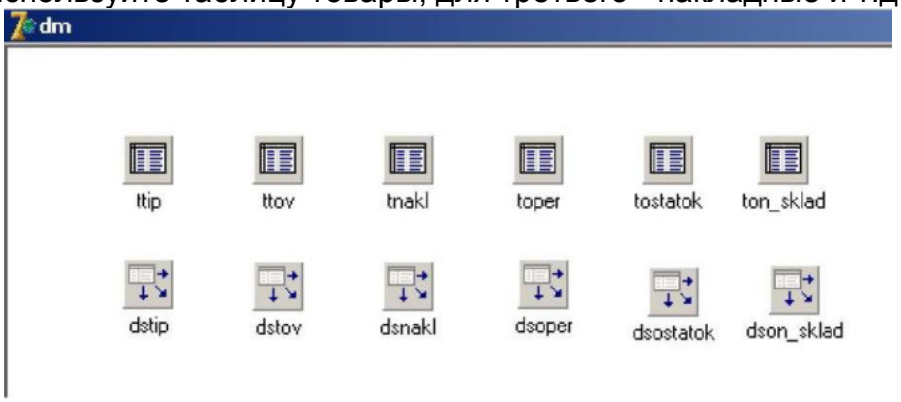
• отметьте необходимые свойства и сделайте записи в модуле

| Название компонента (Caption) | действие, которое он выполняет | Name | Запись в модуле |
|---|---|---|---|
| + | добавляет новую запись в таблицу товары | badd | ttov.Append;<br>pl.Visible:=true;<br>bcancel.visible:=true;<br>bpost.visible:=true; |
| - | удаляет запись из таблицы товары | bdel | ttov.Delete; |
| first | переходит на первую запись в таблице товары | bfirst | ttov.First; |
| last | переходит на последнюю запись в таблице товары | blast | ttov.Last; |

| | | | |
|---|---|---|---|
| next | переходит на одну запись вниз в таблице товары | bnext | ttov.next; |
| prior | переходит на одну запись вверх в таблице товары | bprior | ttov.prior; |
| save | сохраняет изменения в таблице товары | bpost | ttov.Post; |
| cancel | отменяет изменения в таблице товары | bcancel | ttov.Cancel; |
| exit | закрывает форму | bexit | close |

• для панели 2, поставьте свойство Visible - False и запишите в модуле pl.Visible:=false;
bcancel.visible:=false;
bsave.visible:=false;
• сохраните и запустите проект, проверьте работу кнопок.
*Примечание:* в дальнейшем при использование модуля данных такой способ использовать не целесообразно.

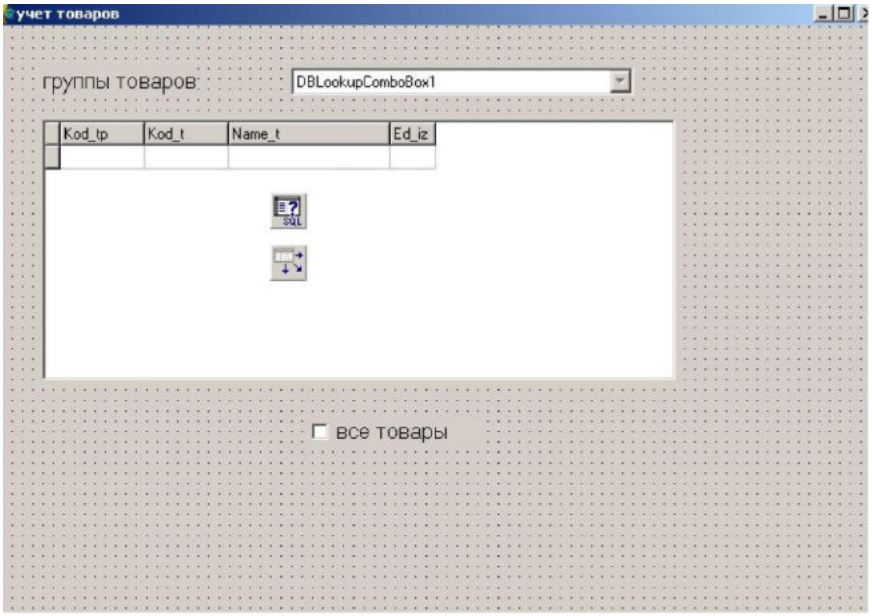
**Создание формы товары через запрос с использованием модуля данных**
**Создание модуля данных**
• обратитесь в пункт меню File - New - Other... - DataModule;
• в свойстве Name укажите имя модуля dm. Сохраните модуль под именем Udm;
• расположите в модуле 6 компонентов TTable и 6 компонентов TDataSource (для всех объектов БД);
• для первого компонента TTable укажите DatabaseName - Sklad, Name -, TableName - tip.db;
• для первого компонента TDataSource укажите DataSet- tTip, Name - dsTip;
• проделайте аналогичные действия для оставшихся компонентов, для второго компонента используйте таблицу товары, для третьего - накладные и т.д.;



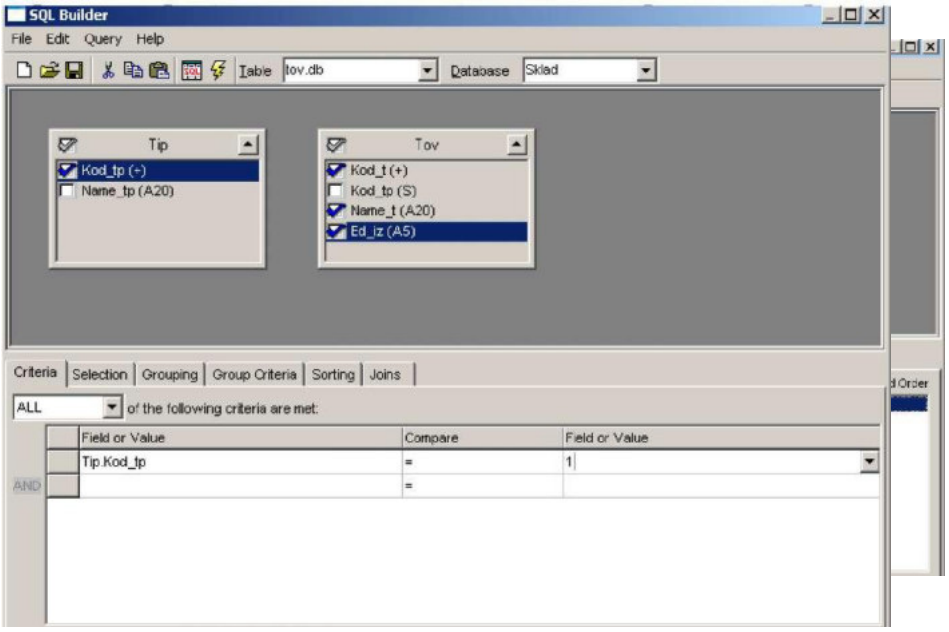

• сохраните модуль данных.
• создайте новую форму и сохраните модуль под именем UTov2, в свойстве Caption укажите название формы - Товары2;
• в свойстве Name - fTov2;
• присоедините форму товары2 к главной форме (см. присоединение формы fTip);
• подключите к форме модуль данных. Обратитесь в пункт меню File - Use Unit... - Udm;
• расположите на форме следующие компоненты и отметьте соответствующие свойства

| **Label**<br>(панель инструментов Standard) | |
|---|---|
| Caption | группы товаров |
| **Queri**<br>(панель инструментов BDE) | |
| DatabaseName | Sklad |
| Name | Qtov |
| **TDateSource**<br>(пвнель инструментов Data Access) | |
| Name | dsQtov |
| DataSet | Qtov |

| **TDBGrit**<br>(панель инструментов Data Controls) | |
|---|---|
| DateSource | dsQtov |
| **TCheckBox**<br>(панель инструментов Standard) | |
| Caption | все товары |
| Name | Ch1 |
| **DBLookupComboBox**<br>(панель инструментов Data Controls) | |




- щелкните по Qtov правой кнопкой мыши и выберите пункт SQL Builder. Откроется
- окно построителя запроса;
- укажите имя БД в поле DataBase - Sklad;
- в поле Table укажите таблицы по которым будет строиться запрос Tip, Tov; "галочками" укажите поля которые будут выводиться на экран - все кроме Name_tp; укажите критерий отбора (закладка Criteria) - код типа равен 1 (Tip.Kod_tp=l)




- перейдите на закладку Joins (связи) и укажите связь между таблицами тип и товары. Для этого выберите в левом поле Tip.Kod_tp а в правом Tov.Kod_tp
- закройте окно построителя запросов (сохраните); для QTov и DBGrit добавьте все поля;
- проверьте работу запроса. Для этого в свойстве Active объекта QTov установите True,
- в объекте DBGrit появятся записи товаров с кодом группы 1 (Kod_tp=1);
- после проверки в свойстве Active объекта QTov установите False;
- переделаем запрос в параметрический. Выберите объект QTov;
- щелкните в свойстве SQL, откроется окно с текстом запроса;
  • в строке WHERE Tip.Kod_tp = 1, поменяйте условие отбора (1 на n) указав WHERE Tip.Kod_tp =:n (без пробелов);
- скопируйте текст запроса;
- обратитесь в свойство Params;
- выделите 0-n;
- измените следующие свойства: Param Type - ptInput, DataType - ftInteger, Value - 1, свойство Value имеет еще "подсвойство" Type - Integer;
- проверьте работу запроса через свойство Active;
- перейдите на закладку Sorting, установите сортировку названий товаров по алфавиту. Для этого выберите tov.Name t и нажмите кнопку Add
- перейдите на закладку Sorting, установите сортировку названий товаров по алфавиту. Для этого выберите tov.Name t и нажмите кнопку Add
- у объекта **DBLookupComboBox** есть возможность только отображать данные из одной таблицы и сохранять в другую. Отметьте для него свойства

| | | |
|---|---|---|
| **ListSource** | dm.dstip | источник списка |
| **ListField** | Name_tp | поле, которое отображается |
| **Key Field** | Kod tp | ключевое поле |
| **Data Source** | | куда сохраняем |
| **Data Field** | | что сохраняем |

- щелкните два раза по объекту DBLookupComboBox;
- в открывшемся окне модуля запишите Qtov.Close;

Qtov.Params[0].Value:=dm.ttipKod_tp.Value; Qtov.Open;

- при активации формы откройте таблицы товары, тип и запрос. Для этого щелкните в событии OnActivate для формы FTov2, и запишите в модуле

dm.ttip.Open;
dm.ttov.Open;
Qtov.open;
DBLookupComboBox 1.keyvalue:=1;
последняя строка отвечает за то, чтобы при активации формы отображались товары с кодом группы l(Tip.Kod_tp = 1);

- при закрытии формы в событие OnClose для формы FTov2, и запишите в модуле

dm.ttip.Close;
dm.ttov.Close; Qtov.Close;

- сохраните результаты работы и запустите проект, проверьте правильность работы запроса;
- кнопка Все товары позволяет отображать все товары одновременно. Для ее создания преобразуйте запись в модуле для объекта DBLookupComboBox следующим образом

```
Qtov.Close; Qtov.SQL.Clear;
Qtov.SQL.Add('SELECT Tip.Kod_tp, Tov.Kod_t, Tov.Name_t, Tov.Ed_iz');
Qtov.SQL.Add('FROM "tip.db" Tip');
Qtov.SQL .Add('INNER JOIN "tov.db" Tov');
Qtov.SQL.Add(' ON (Tip.Kod_tp = Tov.Kod_tp)');
Qtov.SQL.Add('WHERE Tip.Kod_tp =:n');
Qtov.Params[0].Value:=dm.ttipKod_tp.Value;
Qtov.Open;
```

- для объекта chl запишите в модуле

```
if ch1.Checked=true then
begin
Qtov.Close; Qtov.SQL.Clear;
Qtov.SQL.Add('SELECT Tip.Kod_tp, Tov.Kod_t, Tov.Name_t, Tov.Ed_iz');
Qtov.SQL.Add('FROM "tip.db" Tip');
Qtov.SQL .Add('INNER JOIN "tov.db" Tov');
Qtov.SQL.Add(' ON (Tip.Kod_tp = Tov.Kod_tp)');
Qtov.SQL.Add('WHERE Tip.Kod_tp>0');
Qtov.Open;
end
else
begin
Qtov.Close; Qtov.SQL.Clear;
Qtov.SQL.Add('SELECT Tip.Kod_tp, Tov.Kod_t, Tov.Name_t, Tov.Ed_iz');
Qtov.SQL.Add('FROM "tip.db" Tip');
Qtov.SQL .Add('INNER JOIN "tov.db" Tov');
Qtov.SQL.Add(' ON (Tip.Kod_tp = Tov.Kod_tp)');
Qtov.SQL.Add('WHERE Tip.Kod_tp =:n');
Qtov.Params[0].Value:=dm.ttipKod_tp.Value;
Qtov.Open;
end;
```

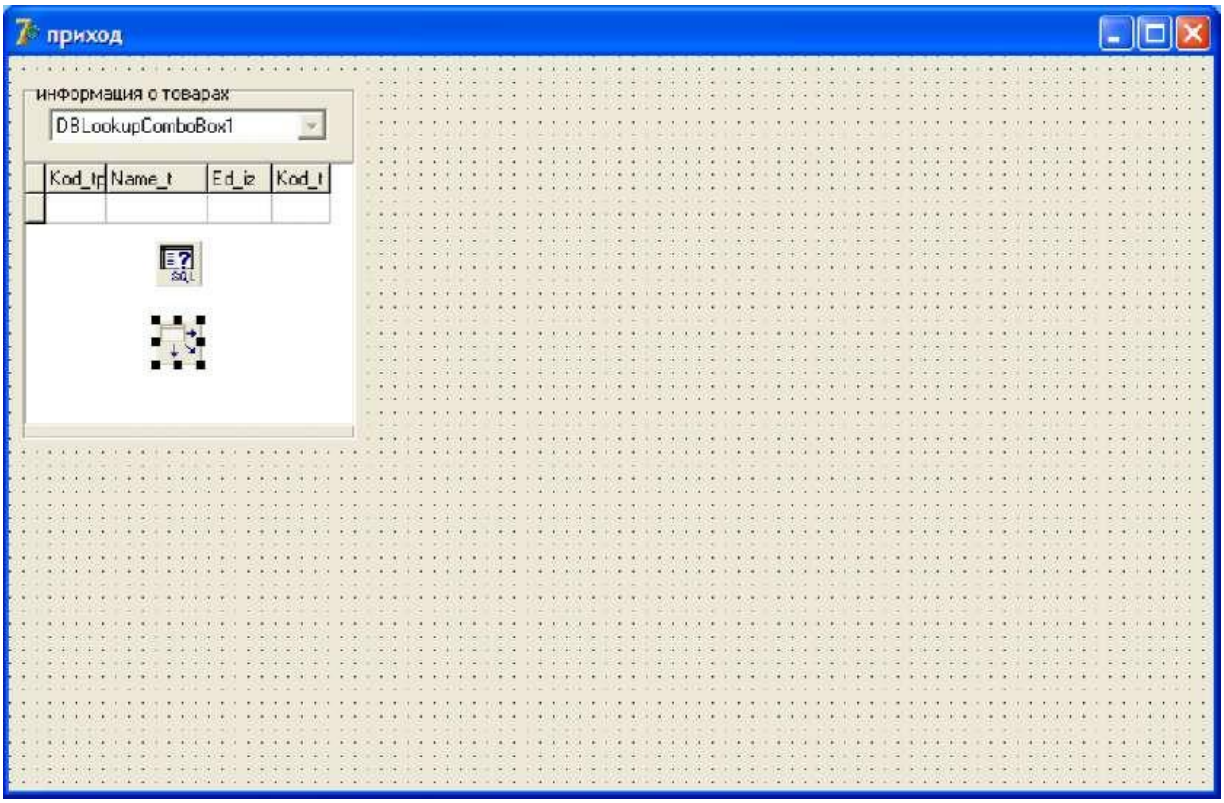
• сохраните результаты работы и запустите проект.

**Форма "Приход"**

- создайте новую форму. Сохраните модуль под именем Upr;
- в свойстве Caption укажите название формы - Приход;
- в свойстве Name - fpr;
- присоедините форму приход к главной форме (см. присоединение формы fTip);
- расположите на форме следующие компоненты и отметьте соответствующие свойства

| **GroupBoxI**<br>(панель инструментов Standard) |
|---|

| | |
|---|---|
| Caption | Информация о товарах |
| **TQueri**<br>(панель инструментов BDE) | |
| DatabaseName | Sklad |
| Name | Q1 |
| **TDateSource**<br>(панель инструментов Data Access) | |
| DataSet | Q1 |
| **TDBGrit**<br>(панель инструментов Data Controls) | |
| DateSource | dsQ1 |
| **DBLookupComboBox**<br>(панель инструментов Data Controls) | |



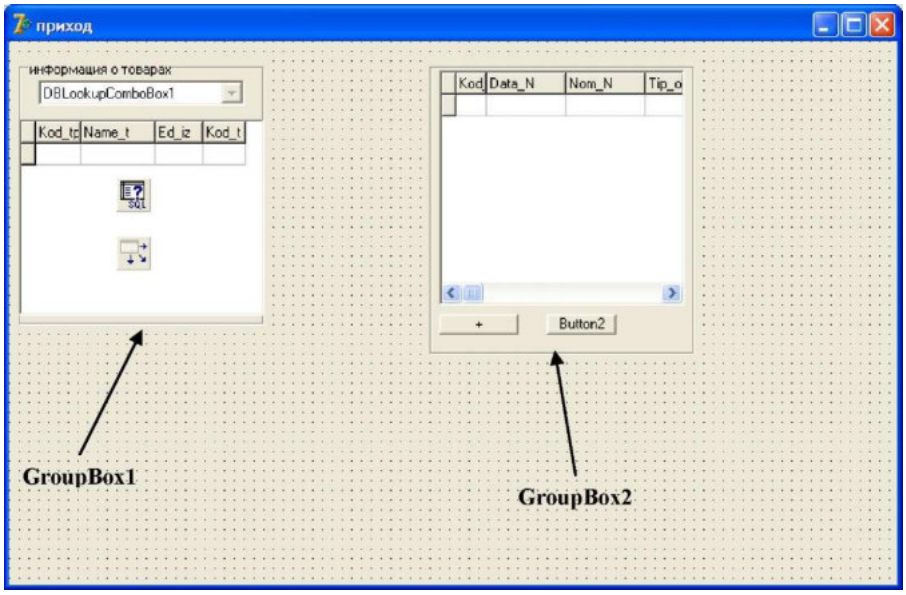

- в событие OnActivate для формы fpr «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите:
  q1.Open;
  dm.ttip.Open;
  dm.ttov.Open;
  dm.tnakl.Open;
  dm.tnakl.Filter:='Tip_o=true';
  dm.tnakl.Filtered:=true;
  dm.toper.Open;
  dm.tostatok.Open;
  dm.ton_sklad.Open;
- в событие OnClose для формы fpr «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите:

q1.close;
dm.ttip.close;
dm.ttov.close;
dm.tnakl.close;
dm.toper.close;
dm.tostatok.close;
dm.tonsklad.close;

- нажмите на Q1 правой кнопкой мыши и выберите пункт SQL Builder. Откроется окно построителя запроса;
- постройте запрос (см. построитель запросов для формы Товары2);
- проверьте работу запроса (см. форма Товары2);
- для Q1 и DBGrit добавьте все поля;
- проверьте работу запроса. Для этого в свойстве Active объекта Q1 установите True, в объекте DBGrit появятся записи товаров с кодом группы 1 (Kod_tp=l);
- после проверки в свойстве Active объекта Q1 установите False;
- переделаем запрос в параметрический. Выберите объект Q1;
- щелкните в свойстве SQL, откроется окно с текстом запроса;
- в строке WHERE Tip.Kod_tp=1, поменяйте условие отбора (1 на n) указав WHERE Tip.Kod_tp=:n (без пробелов);
- скопируйте текст запроса;
- обратитесь в свойство Params;
  - выделите 0-n;измените следующие свойства: Param Type - ptInput, DataType - ftInteger, Value- 1, свойство Value имеет еще "подсвойство" Type - Integer;
- проверьте работу запроса через свойство Active;
- у объекта **DBLookupComboBox** есть возможность только отображать данные из одной таблицы и сохранять в другую. Отметьте для него свойства

| | | |
|---|---|---|
| **ListSource** | dm.dstip | источник списка |
| **ListField** | Name tp | поле, которое отображается |
| **Key Field** | Kod tp | ключевое поле |
| **Data Source** | | куда сохраняем |
| **Data Field** | | что сохраняем |

- «щелкните» два раза по объекту DBLookupComboBox; в открывшемся окне модуля запишите
  q1.close;
  q1.Params[0].Value :=dm.ttipKod_tp.Value; ql.Open;
- сохраните результаты работы;
- откомпилируйте и запустите проект;




• расположите на форме следующие компоненты и отметьте соответствующие свойства

| | |
|---|---|
| **GroupBox2** (панель инструментов Standard) | |
| **TButton1** (панель инструментов Standard) | |
| Caption | + |
| Name | BNakladd |
| **TButton2** (панель инструментов Standard) | |
| Caption | - |
| Name | del |
| **TDBGrit2** (панель инструментов Data Controls) | |
| DateSource | dm.dsnakl |

• для компонента TDBGrid2 добавьте все поля через редактор полей;
• в событие OnClick (для кнопки BNakladd) «щелкните» два раза мышкой, в
• раскрывшемся окне модуля напишите

```
dm.Tnakl.Append;
GroupBox3.Visible := true;
DateTimePicker1.DateTime:= now();
```

• в событие OnClick (для кнопки Button2) «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите dm.tnakl.delete; сохраните результаты работы; откомпилируйте и запустите проект;
• расположите на форме следующие компоненты и отметьте соответствующие свойства:

| | |
|---|---|
| **GroupBox3** (панель инструментов Standard) | |
| **Button3** (панель инструментов Standard) | |
| Caption | save |
| **Button4** (панель инструментов Standard) | |

| | |
|---|---|
| Caption | cancel |
| **DataTimePickerI**<br>(панель инструментов Data Controls) | |
| **Edit1**<br>(панель инструментов Standard) | |
| Text | Edit1 |
| **Label1**<br>(панель инструментов Standard) | |
| Caption | номер |
| **Label2** | |
| (панель инструментов Standard) | |
| Caption | дата |

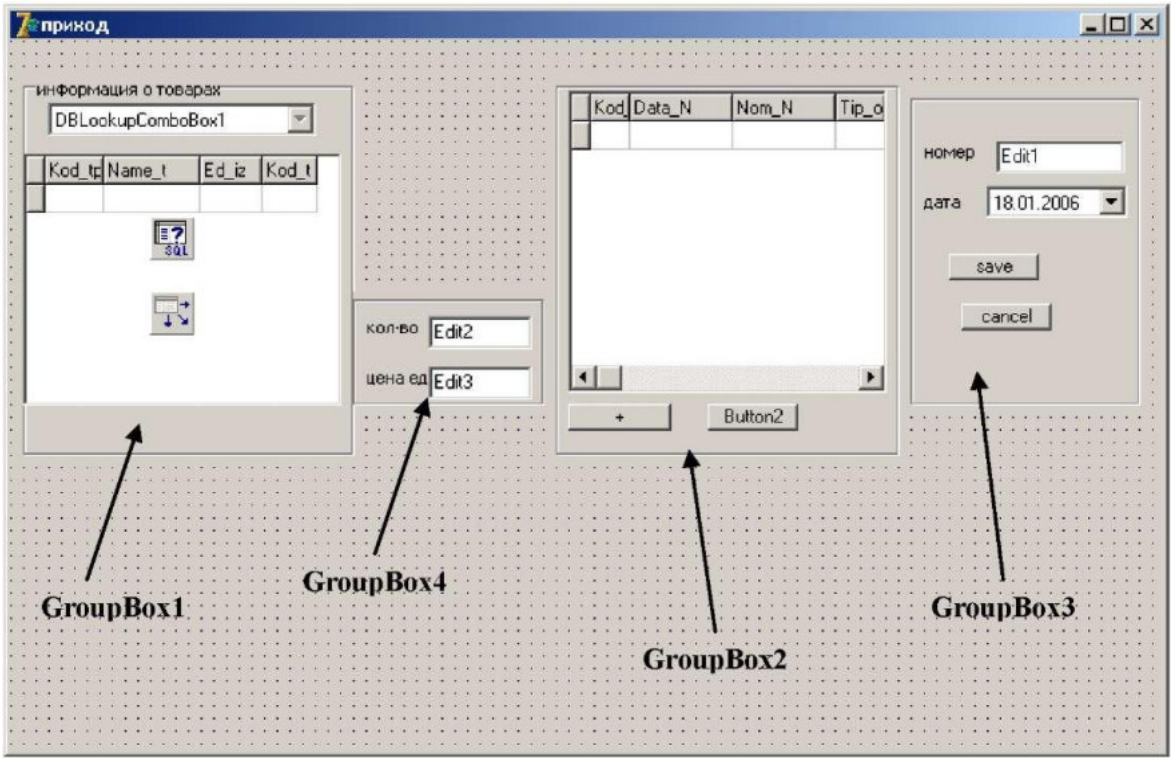
- в событие OnClick (для кнопки Button3) «щелкните» два раза мышкой, в
- раскрывшемся окне модуля напишите:

```
dm.tnaklNom_N.Value :=StrToInt(edit1.Text);
dm.tnaklData_N.Value:=DateTimePicker1.Date;
dm.tnaklTip_o.Value := True;
dm.Tnakl.Post;
GroupBox3.Visible := false;
```

- в событие OnClick (для кнопки Button4) «щелкните» два раза мышкой, в
- раскрывшемся окне модуля напишите:

```
dm.tnakl.Cancel;
groupbox3 .Visible :=false;
```

- сохраните результаты работы;
- откомпилируйте и запустите проект;
- расположите на форме следующие компоненты и отметьте соответствующие свойства:

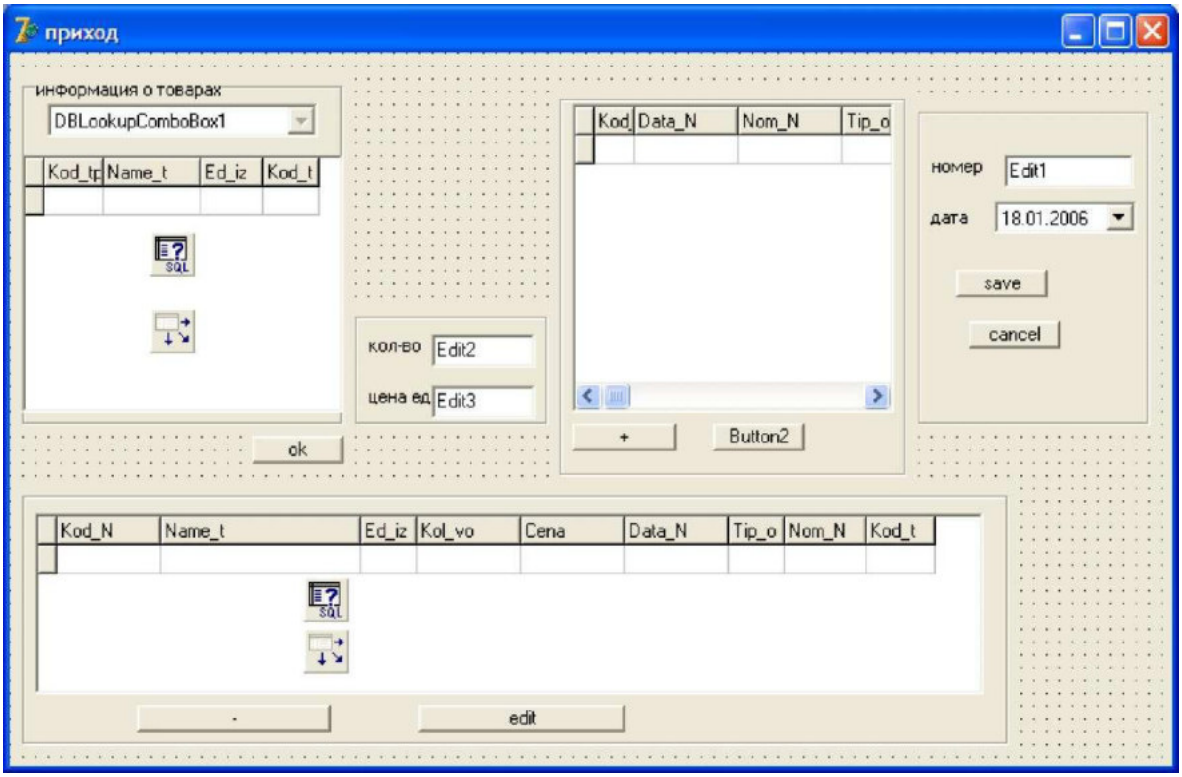



| |
|---|
| **GroupBox4**<br>(панель инструментов Standard) |

| **Edit2** (панель инструментов Standard) | |
|---|---|
| Text | Edit2 |
| **Edit3** (панель инструментов Standard) | |
| **Label3** (панель инструментов Standard) | |
| Caption | Кол-во |
| **Label4** (панель инструментов Standard) | |
| Caption | Цена ед. |

- расположите на форме следующие компоненты и отметьте соответствующие свойства:

| **GroupBox5** (панель инструментов Standard) | |
|---|---|
| **Queri** (панель инструментов BDE) | |
| DatabaseName | Sklad |
| Name | Qsod n |
| **TDateSource** (панель инструментов Data Access) | |
| Name | DateSource2 |
| DataSet | Qsod n |
| **TDBGrit3** (панель инструментов Data Controls) | |
| DateSource | Qsod n |
| **Button5** (панель инструментов Standard) | |
| Caption | ok |
| **Button6** (панель инструментов Standard) | |
| Caption | - |
| **Button7** (панель инструментов Standard) | |
| Caption | edit |




- для компонента TDBGrid3 добавьте все поля через редактор полей;
  постройте запрос с помощью которого будут отображаться товары на складе (см. построитель запросов для формы Товары2), в поле Table укажите таблицы по которым будет строиться запрос, Tov, Nakl, Oper, выделив поля Kod_N, Name_t, Ediz, Kolvo, Cena, DataN, Tipo, NomN, Kod t;
- проверьте работу запроса (см. форма Товары2);
- в событие OnCellClick (для компонента TDBGrid2) «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите
  qsod_n.Close;
  qsod_n.Params[0].Value:=dm.tnaklKod_N.Value; qsodn.Open;
- в событие OnKeyUp (для компонента TDBGrid2) «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите:
  qsod_N.Close;
  qsod_N.Params[0].Value := dm.TnaklKod_n.Value; qыod_N.Open;
- для добавления товара на склад в событие OnClick (для кнопки Button5 (ok)) «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите:
  Val(edit2.Text,c,k);

Val(edit3.Text,c,k1); if (k<>0) or(k1<>0)
then MessageDlg('Не верные данные в полях ЦЕНА или КОЛИЧЕСТВО!', mtInformation, [mbOk], 0) else begin
/// Добавление данных в табл. Операции dm.Toper.Append;
dm.toperKod_N.Value:= dm.tnaklKod_N.Value;
dm.toperKod_t.Value:= q1Kod_t.Value;
dm.TоperCena.Value := StrToFloat(Edit3.Text);
dm.Toper.Post; ///добавление данных в таблицу на складе dm.ton_sklad.First;
if dm.ton_sklad.Locate('kod_t',q1Kod_t.Value,[]) = true
Then begin dm.ton_sklad.Edit;
dm.ton_skladKol_vo.Value :=
dm.ton_skladKol_vo.Value + StrTOInt(Edit2.text);
dm.ton_sklad.post;
end
Else begin

```
                dm.ton_sklad.Append;
                dm.ton_skladKod_t.Value :=q1Kod_t.Value;
                 dm.ton_skladKol_vo.Value: = StrToInt(Edit2.text);
                 dm.ton_sklad.Post;
                end;
        /// Добавление в табл. "Остатки"
        If dm.tostatok.Locate('kod_t;data_o',VarArrayOf([q1Kod_t.Value,
        dm.tnaklData_N.value]),[])=true then
        begin
                dm.tostatok.Edit;
                dm.tostatokKol_vo.Value :=
                        dm.tostatokKol_vo.Value+StrToInt(Edit2.Text);
                dm.tostatok.post;
                end
        Else begin
                dm.tostatok.Edit; dm.tostatok.Append;
                dm.tostatokKod_t.Value :=q1Kod_t.Value;
                dm.tostatokKol_vo.Value := StrToint(Edit2.Text);
                dm.tostatokData_o.Value := dm.tnaklData_N.Value; dm.tostatok.post;
                end;
        // Смотрим содержание накладной qsod_n.Close;
        qsod_n.Params[0].Value:=dm.tnaklKod_N.Value;
        qsod_n.Open; end;
        end;
```

• в событие OnClick (для кнопки Button6) «щелкните» два раза мышкой, в раскрывшемся окне модуля напишите:

```
k:=qsod_nKod_t.Value; dt := qsod_NData_N.Value;
            //У даляем данные из табл. Остаток
            if dm.tostatok.Locate('kod_t;data_o',VarArrayOf([k,Dt]),[])=true then begin
            if dm.tostatokKol_vo.Value= qsod_nKol_vo.Value then
            begin
                 dm.tostatok.Edit;
                 dm.tostatok.Delete;
            end;
 if dm.tostatokKol_vo.Value > qSod_NKol_vo.Value then
 begin
        dm.tostatok.Edit;
        dm.tostatokKol_vo.Value:=
           dm.tostatokKol_vo.ValueqSod_NKol_vo.Value; dm.tostatok.Post;
end;
 if dm.tostatokKol_vo.Value < qSod_NKol_vo.Value then
 begin
            MessageDlg('Такого количества НЕТ в ост!', mtInformation, [mbOk], 0)
end;
end;
 qsod_N.Close;
 qsod_N.Params[0].Value := dm.TnaklKod_N.Value;
 qsod_N.Open;
 //удаляем данные из таблицы на складе
 if dm.ton_sklad.Locate('kod_t',qsod_nKod_t.Value,[]) = true then begin
 if dm.ton_skladKol_vo.Value= qsod_nKol_vo.Value then
 begin
        dm.ton_skl ad.Edit;
```

```
        dm.ton_sklad.Delete;
end;
if dm.ton_skladKol_vo.Value > qSod_NKol_vo.Value then
begin
        dm.ton_skl ad.Edit;
        dm.ton_skladKol_vo.Value:=
                dm.ton_skladKol_vo.Value-qSod_NKol_vo.Value;
        dm.ton_sklad.Post;
end;
if dm.ton_skladKol_vo.Value < qSod_NKol_vo.Value then
begin
    MessageDlg('Такого количества НЕТ на скл!', mtInformation, [mbOk], 0)
end;
end;
qsod_N.Close;
qsod_N.Params[0].Value := dm.TnaklKod_N.Value;
qsod_N.Open;
//удаляем данные из таблицы операции
if dm.toper.Locate('kod_t;kod_n',VarArrayOf([k,qsod_nKod_N.Value]),[])=true then
begin
        if dm.toperKol_vo.Value= qsod_nKol_vo.Value then
        begin
           dm.toper.Edit;
           dm.toper.Delete;
        end;
        if dm.toperKol_vo.Value > qSod_NKol_vo.Value then
        begin
           dm.toper.Edit;
           dm.toperKol_vo.Value:=dm.toperkol_vo.Value-qSod_NKol_vo.Value;
           dm.toper.Post;
        end;
    if dm.toperKol_vo.Value < qSod_NKol_vo.Value then
    begin
        MessageDlg('Такого количества НЕТ !', mtInformation, [mbOk], 0)
    end;
    end;
    qsod_N.Close;
    qsod_N.Params[0].Value := dm.TnaklKod_N.Value;
    qsod_N.Open;
```

- сохраните результаты работы; откомпилируйте и запустите проект.