ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ.

Факультет информационных технологий

Кафедра Программное обеспечение информационных технологий

Базы данных

Методические указания к выполнению лабораторных работ для студентов специальности 5521900 ИТ

Ташкент 2007

Составители: ТУИТ кафедра «Программное обеспечение информационных технологий» доц. Салахутдинов В. Х.

АНРУЗ институт «Информатики» старший научный сотрудник д.т.н. Нурмухамедов Т.Р.,

ТУИТ кафедра «ИТ» доц. д.т.н. Зайнутдинов Х.

Данные методические указания рассмотрены и утверждены на заседании кафедры ПОИТ. (___ _____2007 год, протокол №____)

Заведующий кафедрой_____акад. Камилов М.М.

(подпись) (Ф.И.О.)

Методическое указание рассмотрено и утверждено научно-методическом советом факультета ИТ.(___ _____2007год, протокол №____)

Председатель научно-методическим совета _____проф. Раджапов Б.

(подпись) (Ф.И.О.)

ВВЕДЕНИЕ

Решение широкого круга задач, связанных с выполнением выпускных работ, а в дальнейшем научных или производственных, требует умения разрабатывать и создавать базы данных, подбирать соответствующие им системы управления, разрабатывать прикладные программы поиска и обработки информации.

Задачей лабораторных работ является закрепление теоретического материала дисциплины "Базы данных и получение практических навыков по созданию и использованию баз данных.

Лабораторная Работа 1.

Тема: Анализ предметной области. Разработка модели "СУЩНОСТЬ-СВЯЗЬ".

Цель: Получить навыки по обследованию предметной области для разработки модели информационной базы данных.

Анализ предметной области (ПО) заключается в выявлении следующих составляющих : объектов, свойств объектов, связей (объектных отношений), временных интервалов (времени, в течение которого объекты могут иметь определенные состояния(Табл.1.1 по 1.6))

Пример ПО. " Склад "

Таблица 1.1

объекты	Количество
1. Склад	n1
2. Материалы	n2
3. Поставщики	n3
4. Потребитель	n4
5. Город	n5
6. Детали	n6

Объекты, свойства объектов

Таблица 1.2

Свойства	Кому принадлежит	Адрес склада	Наименование	Количество	
Склад				Приход	Расход

Таблица 1.3

|Свойства	Наименование	Потребитель	Где хранится
Материалы			

Таблица1.4

|Свойства	Почтовый индекс	Наиме-нование	Что постав-ляется	Что выво-зится	Потре-бители	Постав- щики
Город						

Таблица1.5

|Свойства	Расходуемый материал	Количество на 1 деталь	Цвет	Из какого склада	Наимен. детали	Вес детали	Кто изго-товляет
Деталь							

Таблица1.6

|Свойства	Имя	Материалы		Адрес потребителя	Из склада
Потребитель					

С в я з и (объектные отношения)

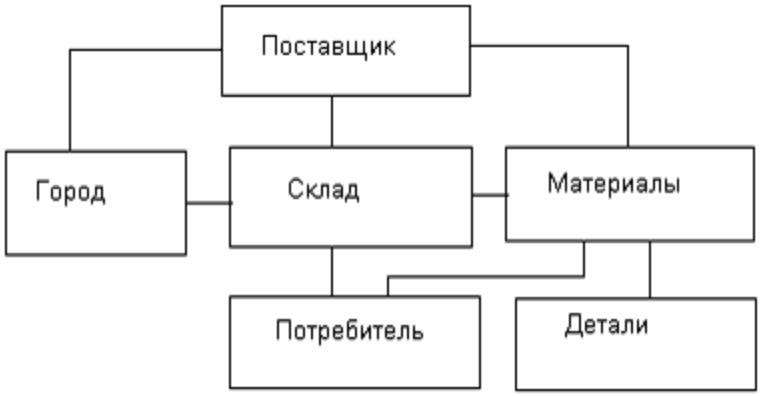

Рис. 1.1. Объектные отношения между элементами модели объекта «Склад».

Модель "Сущность - Связь" строится с использованием трех основных компонент, составляющих ПО : СУЩНОСТЬ, АТРИБУТ , СВЯЗЬ. Составляющая "ВРЕМЯ" в составе конструктивных элементов может присутствовать только в неявном виде. В модели время представлено атрибутами: год, дата и тому подобным.

При построении модели СВЯЗЬ выступает как абстракция, существующего объекта, процесса или явления. АТРИБУТ представляет характеристику с именем, которая принимает значения из некоторого множества значений.

К связям в модели "Сущность - Связь" необходимо добавить отношения, присущие для каждого типа связи между двумя сущностями(бинарные, тернарные, ... , n-нарные).

Информация о проекте оформляется в виде диаграммы, для этого обозначают: типы сущностей - прямоугольниками, атрибуты - овалами, соединяя их соответствующими сущностями - ненаправленными ребрами. Связи(отношения) - ромбами, соединяя их, в общем случае, ненаправленными ребрами с типами сущностей, а при бинарных связях направленными ребрами.

Модель "Сущность - Связь" может отражать, только определенную часть ПО, в этом случае ее называют локальной. Для более полной информации о ПО

необходимо ее дообследование и построение локальных моделей, дополняющих предыдущую. После чего локальные модели объединяются в композиционно-единое представление о ПО (Рис.1.2).

Пример

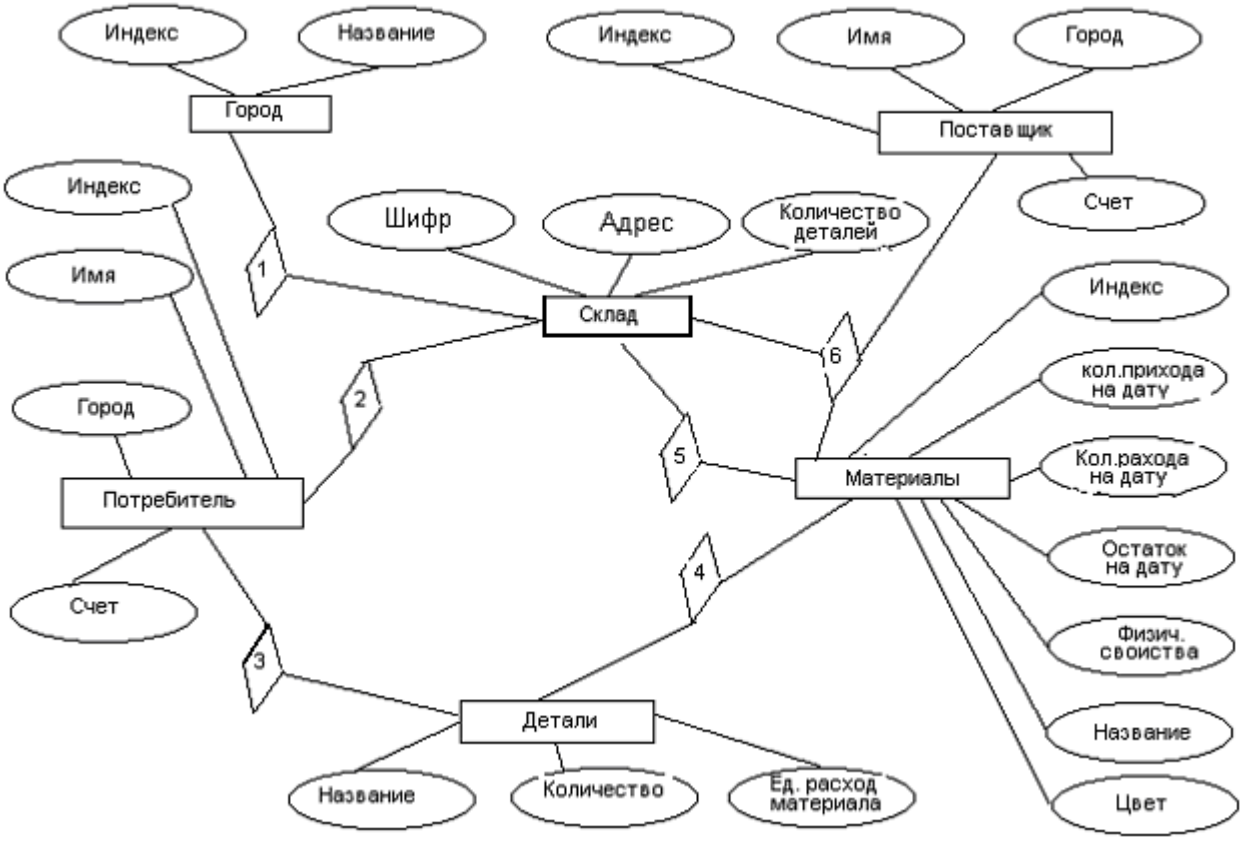

Рис.1.2 Пример модели "Сущность – Связь для предметной области «Склад».

Данный пример модели "Сущность - Связь" объединяет в себе пять локальных моделей.

Контрольные вопросы.

1. Дайте определение понятию предметная область.
2. Что понимается под моделью "Сущность-связь"?
3. Определите понятие "Базовые элементы модели "Сущность-связь"".
4. Каким образом отображается модель "Сущность-связь" в структуры базы данных?

5. Что представляют собой диаграммы модели «Сущность - связь»?

Лабораторная работа № 2.

Тема: Создание реляционной модели базы данных для заданной предметной области.

Цель: научиться разрабатывать систему таблиц реляционной модели базы данных.

При разработке таблиц необходимо учесть, что система таблиц базы данных должна соответствовать разработанной Вами модели «Сущность - Связь»

Таблица 2.1

№	Понятия	События	Характеристики
1	Город	Располагаться	Расстояние
2	Поставщик	Снабжать	Количество, вес
3	Склад	Поставлять, находиться	Цвет, размер
4	Потребитель	Получать, расходовать	
5	Материалы	Изготавлять	
6	Детали		

В таблице 2.1 выделены понятия, отвечающие сущностям, события – связям, а характеристики – атрибутам сущностей модели «Сущность – связь».

Понятия:

"Город" Таблица 2.2

Индекс	Название города
700000	Ташкент

"Поставщик " Таблица 2.3

Инд	Имя	Город	Счет
ПС1	Кураев И.С.	Ташкент	31

"Склад" Таблица 2.4

Шифр	Наименование
С1	Склад лакокрасочных изделий
С2	Склад лакокрасочных изделий

"Потребитель" Таблица 2.6

Инд	Имя	Город	Счет
ПТ1	Таксопарк	Ташкент	231
ПТ2	МП "Шарк"	Самарканд	39

События.

Располагаться Таблица 2.8

Город	Склад Шифр
Ташкент Ташкент	С1 С2

"Снабжать" Таблица 2.9

Постав-щик	Материалы Шифр	Склад Шифр
ПС1 ПС2	ЭК- 1Б ЭК- 2з	С1 С2

"Расходовать" Таблица 2.10

Деталь Шифр	Материалы Шифр	Потреби-тель
Д1 Д2	ЭК- 1Б ЭК- 2з	ПТ1 ПТ2

"Получать" Таблица 2.11

Склад Шифр	Потребитель	Материалы Шифр
С1 С2	ТТ3 Таксопарк	ЭК- 1Б ЭК- 2з

"Изготовлять" Таблица 2.12

Потреби-тель	Наименование детали
ПТ1 ПТ2	Консоль Клапан

"Имеется" Таблица 2.13

Склад Шифр	Материалы Шифр	Количеств тонн	Дата
С1 С2	ЭК- 1Б ЭК- 2з	3 4	01.01.01 12.31.01

Характеристики

Колич. расхода Таблица 2.14

Детали	Колич.расх. материала г/детль	материал Шифр
Консоль Клапан	300 200	ЭК - 1б ЭК - 2з

Колич. поставки Таблица 2.15

материал Шифр	Колич. постав.мате-риала в тоннах
ЭК - 1б ЭК - 2з	5 6

Цвет Таблица 2.16

материал Шифр	Цвет
ЭК - 1б ЭК - 2з	Белая Зеленая

Колич. рахода-прихода Таблица 2.17

материал Шифр	Расход тонн	Приход тонн	Дата	Склад Шифр
ЭК - 1б ЭК - 2з	0.5 0.1	1 1	20.01.01 20.02.01	С1 С2

Остаток Таблица 2.18

материал Шифр	Остаток Тонн	Дата	Склад Шифр
ЭК - 1б ЭК - 2з	3 4	20.01.01 20.02.01	С1 С2

Контрольные вопросы.

1. Что понимается под понятием «Реляционная база данных»?
2. Дайте определение понятиям: «Кортеж», «Домен».
3. Какие операции Вы знаете над таблицами реляционной базы данных?
4. Дайте определение первой, второй и третьей нормальным формам.
5. Можно ли свести таблицу, находящейся в первой нормальной форме к третьей, минуя вторую?

Лабораторная работа 3.

Тема: Создание системы запросов к базе данных. Выбор СУБД.

Цель работы: Приобретение навыков в составление запросов к базе данных .

Содержание задания.

1. Совместно с заказчиком базы данных (БД) разработчику необходимо определить систему запросов к БД , максимально удовлетворяющую требованиям заказчика и релевантной созданной модели "Сущность - связь".

Если при заданных требованиях заказчика данная модель "Сущность - связь" не может полностью удовлетворить системе запросов, то необходимо дополнительно обследовать предметную область и дополнить модель "Сущность - связь" необходимыми элементами, связями и отношениями.

2. Классифицировать систему запросов по выдаваемой пользователю БД информации на стандартные запросы и нестандартные (нерегламентированные).

3.

Нерегламентированные запросы к БД - запросы, которые могут менять свою формулировку в зависимости от текущих потребностей пользователя, но только в пределах возможностей построенной модели и ее релевантности этих запросов к ней.

Пример системы запросов для предметной области "Склад":

1) Найти внешнее описание всех материалов, находящихся на складе.

2) Найти названия и даты приема материалов.

3) Найти название материалов, не имеющихся на складах.

4) Найти название материалов и потребителей.

5) Найти название складов, получивших материалы.

6) Для каждого материала найти номер склада, количество на складе, приход, расход.

7) Найти потребителей и количество требуемого материала с номером М1.

8) Найти количество прихода и расхода материалов с весом > 100.

9) Найти название материалов и их количество для склада с номером С1.

10) Найти название материалов красного цвета с максимальным весом.

11) Найти потребность в материале (количество и потребителя) для деталей с минимальным весом.

12) Найти номера материалов, принятых 31.4 числа.

13) Найти все склады, на которых хранятся все материалы, необходимые потребителю П1.

14) Найти всех потребителей, которым нужны материалы красного цвета.

15) Найти всех потребителей и требуемое количество для всех материалов с весом больше 40.

16) Найти все склады, на которых хранятся материалы М1 и М2.

17) Найти всех потребителей, которым нужны какие-нибудь материалы, полученные 31.4 числа.

18) Найти название материалов и их количество для склада с номером С1.

19) Найти общий вес материалов, необходимых потребителю П1.

20) Найти общее число материалов, необходимых потребителю П1.

21) Для каждого материала и каждого склада подсчитать новое состояние склада по формуле

кол = кол0 + кол1 - кол2

| | |

остаток приход расход

22) Найти общее количество прихода по каждому материалу.

23) Для каждого склада и каждого материала найти общее количество расхода.

24) Найти материалы с наибольшим требуемым количеством.

25) В отношение "Приход" добавить выборки:

26) Изменить текущее состояние склада (учесть приход и расход).

27) Во всех отношениях уничтожить выборки.

28) Вычислить нехватку для каждого материала.

29) Вычислить избыток белого материала.

30) Выбрать материалы с максимальным избытком.

31) Найти все склады, на которых хранятся материалы М1 и М2, создав дополнительные отношения.

4. Представление БД в виде реляционной схемы. Выбор СУБД.

Необходимо выполнить работы:

а) По модели "Сущность - связь" выделить три типа отношений:

- понятия,
- отношения "событие",
- отношения " характеристика".

Пример.

Для модели "Сущность - связь" все типы отношений сведем в таблицу 2.1.

Порядков. номер	Номер детали	Колич. приход	Колич. остаток	Номер потребителя
10 17	Д1 Д2	160 170	349 352	ПТ1 ПТ2

Выбор СУБД.

Выбор СУБД в большей степени зависит от типа приложений. Тип приложений определяется методами доступа к Б.Д., которые, в свою очередь, определяются планируемыми запросами.

Различают следующие типы:

Тип 1. Получить все или многие записи. Под этот тип попадают чаще всего последовательная обработка, генерация больших отчетов и пакетная обработка.

Тип 2. Получить уникальную запись. Доступ осуществляется только к одной записи. В этот тип попадают: метод прямого доступа, метод произвольного доступа, индексные методы, бинарное дерево и т.д.

Тип 3. Получить некоторые записи. Для такого типа запросов наибольший интерес представляет поиск по вторичному ключу. Здесь лучше всего использовать инвертированный метод доступа.

Учет подобной классификации позволит наилучшим образом реализовать СУБД, удовлетворить требованиям к времени ответа на запросы и ограничениям на объем памяти.

Реляционная схема наилучшим образом удовлетворяет требованиям к языку запросов к Б.Д., а именно, язык запросов должен быть одновременно простым в изучении и легким в пользовании.

Контрольные вопросы.

1. Какие формы запросов к базе данных Вы знаете?
2. Какие языки используются при составлении запросов к базе данных?
3. Что получает пользователь с помощью языка запросов при обращении к базе данных;
4. Какие типы функций СУБД Вы знаете;
5. Назовите основные составляющие организации СУБД.
6. Какими критериями определяется выбор СУБД для разработки базы данных?

Лабораторная работа 4

Тема: Создание программ ведения баз данных.

Цель работы : Приобретение навыков работы с операторами языка SQL. Научиться создавать программы ведения баз данных в выбранной СУБД.

Задание к работе.

1. Изучить конструкции языка SQL для создания базы данных
2. Создать базу данных для табличных структур полученных при разработке модели «Сущность - связь», заданной предметной области.
3. В среде WISQL или SQL-Explorer создать физическую структуру разработанной базы данных
4. Внести в базу данных некоторое количество информации, на примере которой можно продемонстрировать работу всех используемых структур.

Теоретические сведения

Преимущества архитектуры клиент-сервер:

- Большинство вычислительных процессов происходит на сервере, что снижает требования к вычислительным мощностям компьютеров клиентов
- Снижается сетевой трафик за счет посылки клиенту только тех данных, которые он запрашивал, а не всей базы данных

- Возрастает безопасность всей системы за счет перенесения большей части бизнес-правил на сервер, существенного увеличения защищенности центральной базы данных от несанкционированного доступа, централизованного управления транзакциями.

SQL-сервер InterBase (IBDataBase) предназначен для хранения и обработки больших объемов информации в условиях одновременной работы с БД множества клиентских приложений. Ниже представлены основные возможности сервера:

Для задания целостности БД определяются:

- Отношения подчиненности между таблицами путем задания первичных и внешних ключей
- Проверка ограничений на вводимые значения для столбцов — проверка диапазона, соответствие маске, требуемое отношение с записями в других таблицах
- Триггеры — автоматически выполняются до или после модификации записей в таблицах
- Генераторы используются для задания уникальных значений полей

Для ускорения работы клиентских приложений используются

- хранимые процедуры и
- виртуальные таблицы (или просмотры)

Для введения отсутствующих в ядре сервера функций, InterBase позволяет использовать подключаемые с помощью DLL

- функции, определяемые пользователем

При наступлении определенных событий InterBase может посылать клиентским приложениям *уведомления*, кроме того, работающие приложения могут обмениваться через сервер сообщениями.

Для доступа к БД используется утилита Windows Interactive SQL (WISQL), входящая в комплект InterBase и SQL Explorer, входящий в поставку Delphi. Для просмотра и анализа процессов, проходящих на сервере при реализации пользовательского запроса, используется утилита SQL Monitor.

Ниже дается краткое описание и примеры определения большинства структур, необходимых для успешного выполнения лабораторной работы. Для получения более подробной информации обращайтесь к литературе по работе с базами данных в среде Delphi.

Создание новой базы данных.

Предполагается, что сервер InterBase уже запущен

0. Запустить WISQL (Пуск | Программы | Interbase Client 5.1 | WISQL
1. File | Create Database
2. Отметить *Local Engine*, в *Database* ввести путь и имя создаваемого файла базы данных. Стандартное расширение — .gdb, В поле *User Name* ввести *SYSDBA*, в поле *Password* ввести *masterkey*. В поле *Databasse Options* ввести *DEFAULT CHARACTER SET WIN1251*. Нажать *OK*.
3. Если сообщений об ошибках не было, база данных создана. Теперь необходимо ее наполнить. Об этом — в следующих разделах.

База данных может также быть создана и средствами языка SQL, например:

```
CREATE DATABASE "C:\Work\PV51\MiD\Walmart.gdb" PAGE_SIZE 1024
USER 'SYSDBA' PASSWORD 'masterkey'  DEFAULT CHARACTER SET WIN1251;
```

Открытие существующей базы данных.

Предполагается, что сервер InterBase уже запущен

0. Запустить WISQL (Пуск | Программы | Interbase Client 5.1 | WISQL
1. File | Connect to DataBase
2. В появившемся окне ввести те же параметры, что и при создании БД

3. Нажать *OK*. Если сообщений об ошибках не поступало, то открытие базы данных прошло успешно.

Создание доменов

Домены определяют множества значений, которые могут храниться в соответствующем домену столбце таблицы. Пример определения домена:

```
CREATE DOMAIN TPHONE AS CHAR(8) CHECK (VALUE LIKE "___-____");
```

Создание таблиц

После создания или открытия базы данных требуется создать в ней таблицы. Создание таблиц может производиться через WISQL или с помощью SQL Explorer'a. В последнем случае все происходит гораздо проще и нагляднее, поэтому разберем сначала его. После двойного щелчка на псевдониме базы данных и ввода имени пользователя и пароля, щелкаем правой кнопкой на Tables, выбираем New, вводим название таблицы. После этого два раза щелкаем на новой таблице, щелкаем правой кнопкой на Columns, вводим название столбца, задаем его тип или домен, соответствующий ему. Аналогичным образом назначаем первичные ключи, определяем ссылочную целостность. После произведенных изменений выбираем Apply (синяя стрелка на верхней панели инструментов) для сохранения изменений.

Создание других компонентов (триггеров, генераторов, индексов) в программе SQL Explorer осуществляется аналогичным образом, поэтому дальше будут рассматриваться приемы создания структур с использованием WISQL.

Создание таблиц в WISQL

В окне SQL Statement необходимо ввести оператор создания таблицы. Пример:

```
CREATE TABLE MAIN (
 SITE        CHAR(30) CHARACTER SET WIN1251 NOT NULL,
 ZIP         CHAR(5) CHARACTER SET WIN1251,
```

```
ADDR            CHAR(50) CHARACTER SET WIN1251,
PHONE     TPHONE NOT NULL,
STORENUM      INTEGER NOT NULL,
PHARMACY      TBOOLEAN,
PHOTO     TBOOLEAN,
TIRES     TBOOLEAN,
CONSTRAINT CPHONE UNIQUE (PHONE),
CONSTRAINT ISTORENUMM PRIMARY KEY (STORENUM));
```

Создание индексов

Для создания индексов в окне WISQL можно ввести оператор определения индекса. Пример:

```
CREATE INDEX ISITE ON MAIN(SITE);
```

Создание генераторов

Генераторы используются для присвоения уникальных значений ключевым полям. В этом смысле они аналогичны полям автоинкремента в таблицах Paradox. Создание генераторов можно проиллюстрировать примером:

```
CREATE GENERATOR GSTORENUM;
```

Создание просмотров

(отображений, представлений, виртуальных таблиц)

Ядром представления является оператор Select, который и формирует содержимое виртуальной таблицы. Он записывается в теле представления. В заголовке описываются выводимые поля. Пример:

```
CREATE VIEW PHARMACY (SITE, GRIDLOC, ZIP, ADDR, STATE, AREACODE,
PHONE) AS
SELECT M.SITE, S.GRIDLOC, M.ZIP, M.ADDR, S.STATE
FROM MAIN M, SITEDETAIL S
WHERE (M.PHARMACY="Present") AND (M.SITE=S.SITE);
```

Создание хранимых процедур

Хранимые процедуры компилируются, хранятся и выполняются на сервере, посылая клиенту по сети лишь результаты исполнения. Пример определения хранимой процедуры:

```
SET TERM ^ ;
CREATE PROCEDURE GET_INFO
RETURNS (NUM_SITES INTEGER, NUM_STATES INTEGER, SUM_INCOME
INTEGER,
NUM_EMPLOYEES INTEGER, AVG_RATING FLOAT, OLDEST_EMPLOYEE
DATE) AS
BEGIN
SELECT COUNT(DISTINCT M.SITE), COUNT(MD.STATE),
SUM(S.ANNUALINCOME), COUNT(E.EMPNAME), AVG(E.RATING),
MIN(E.BIRTHDATE)
FROM MAIN M, MONEYDETAIL MD, STOREDETAIL S, EMPLOYEEDETAIL E
INTO :NUM_SITES, :NUM_STATES, :SUM_INCOME, :NUM_EMPLOYEES,
:AVG_RATING,:OLDEST_EMPLOYEE;
 SUSPEND;
END ^
```

Определение функций пользователя.

Функции пользователя позволяют расширить функции базы данных, наделяя ее такими свойствами, которыми она ранее не обладала. Для подключения функций необходимо заранее создать и откомпилировать DLL- библиотеку, содержащую эту функцию. Если библиотека пишется на Delphi, то после заголовка функции должны быть указаны ключевые слова CDECL;EXPORT;, а сам модуль должен оформляться не как unit, а как library.

Пример описания функции пользователей в СУБД Interbase:

```
DECLARE EXTERNAL FUNCTION DELS CSTRING(256) CHARACTER SET
WIN1251
RETURNS CSTRING(256) CHARACTER SET WIN1251
ENTRY_POINT "DelSpaces" MODULE_NAME "C:\Work\PV51\MiD\WalDll.dll";
```

Создание триггеров

Триггеры — это процедуры, которые автоматически запускаются при внесении изменений в таблицы базы данных. С помощью них можно поддерживать ссылочную целостность, производить каскадные изменения, подсчитывать агрегатные функции и выполнять любые другие действия над таблицами. Следующий пример иллюстрирует применение триггера для обновления информации в двух таблицах, получения значения от генератора и вызова функции, определяемой пользователем:

```
CREATE TRIGGER BI_MAIN FOR MAIN
ACTIVE BEFORE INSERT POSITION 0
AS
BEGIN
NEW.STORENUM=GEN_ID(GSTORENUM,1);
NEW.SITE=DELS(NEW.SITE);
INSERT INTO STOREDETAIL
(STORENUM,SQFEET,EMPLOYEES,ANNUALINCOME)
 VALUES (NEW.STORENUM,0,0,0);
IF (NOT EXISTS (SELECT * FROM SITEDETAIL WHERE SITE=NEW.SITE)) THEN
 INSERT INTO SITEDETAIL (SITE,GRIDLOC,STATE,AREACODE)
      VALUES (NEW.SITE, "?-??","??","????");
END ^
```

Контрольные вопросы.

1. Определите три основные составляющие языка SQL.
2. Назовите основные функции языка SQL.
3. Что Вы понимаете под стандартом SQL?
4. Опишите возможности **SQL СУБД InterBase.**
5. Опишите возможности **SQL СУБД ORACLE.**

Лабораторная работа № 5

Тема: Создание программ обработки данных по SQL запросам.

Цель работы: Приобретение навыков работы по составлению запросов к базе данных на SQL.

Задание к работе

Создать клиентское приложение для работы с базой данных, созданной в первой лабораторной работе. Это клиентское приложение должно отвечать следующим требованиям:

1. Приложение должно представлять собой главную форму, из которой будут вызываться другие формы. Допускается использование компонента TPageControl (закладок)
2. Для каждой таблицы или просмотра базы данных должна быть своя форма (или страница), в которой можно было бы осуществлять ее просмотр и редактирование
3. Для управления базой данных должны использоваться все необходимые элементы управления (TDBMemo, TDBEdit, TDBRichText, TDBGrid, TDBNavigator, и т.д.)
4. В проекте должна присутствовать хотя бы одна форма, которая бы отражала две таблицы, связанные в отношении Master-Detail с помощью свойств Master Source, Master Fields.
5. В проекте должна присутствовать хотя бы одна форма, которая отражала бы использование компонентов TDBLookup (list box или combo box)

6. В проекте должен присутствовать Visual Query Builder, с помощью которого можно было бы создавать запросы не только с переменными параметрами, но и с гибкой настройкой структуры, связывания таблиц, выбора условий, полей, способа сортировки и т.д.
7. В проекте должны быть два отчета — простой и сложный (содержание которого формируется на основе нескольких таблиц, применяется группировка, подсчет итогов, и т.д.)
8. В проекте должны быть графики, отображающие какую-то статистику по базе данных
9. В программе должен быть реализован более-менее понятный интерфейс, заголовки полей в компонентах DBGrid должны быть нормальными русскими, немного нужно подумать о дизайне и эстетике программы

Теоретические сведения

Для того, чтобы более детально узнать о построении приложений, работающих с базами данных, обращайтесь к соответствующей литературе по Delphi или к справочной системе самой среды Delphi.

Создание программ в архитектуре клиент-сервер имеет следующую специфику:

- Не рекомендуется использовать компонент TTable, поскольку он требует передачи всех данных результата выполнения запроса к серверу; в отличие от этого, компонент TQuery получает от сервера только ту их часть, которая должна быть визуализирована;
- Изменение записей БД следует производить не методами Insert, Edit, Delete, Post, Cances, которые оперируют с одной (текущей) записью, а при помощи SQL-операторов INSERT, UPDATE, DELETE, которые оперируют сразу множеством записей;

- Необходимо особое внимание уделять управлению транзакциями и в первую очередь — выбору адекватного потребностям программы уровня изоляции транзакций;
- Бизнес-правила, где это возможно, нужно переносить на сервер, разгружая от них клиентское приложение;
- Следует как можно чаще использовать хранимые процедуры, выполняющиеся быстрее обычных SQL-запросов и уменьшающие загрузку сети;
- Следует везде. где это необходимо, явно вызывать методы TDataBase.StartTransaction и TDataBase.Commit для старта и подтверждения транзакций; подтверждение единичных изменений БД неявно стартуемыми и завершаемыми (в режиме SQLPASSTHRU=SHARED AUTOCOMMIT) транзакциями ведет к возрастанию загрузки сети и, как следствие, к замедлению работы, часто весьма существенному.
- Необходимо уделять существенное внимание оптимизации запросов к БД, особенно при чтении данных (SELECT), поскольку оптимально построенный запрос может выполняться в несколько раз быстрее и требовать меньшего количества ресурсов.

Соединение с БД из программы производится с помощью компонента TDataBase.

В параметре *Alias name* выбираем псевдоним ранее созданной базы данных, в окне *Parameter Overrides* вводим

USER NAME=SYSDBA

PASSWORD=masterkey

Снимаем галочку с опции *Login prompt* для отмены запроса пароля в диалоге. Нажимаем *OK*.

Для соединения с БД устанавливаем свойство *Connected* в *True*.

Работа с таблицами БД в клиентском приложении

Для каждой из таблиц создаем объект TTable

В свойстве DataBase каждого из них ставим ссылку на псевдоним нашей БД

В свойствах TableName выбираем соответствующее имя таблицы для каждого компонента

Для каждого компонента TTable создаем свой компонент TDataSource, и дальше действуем как при работе с обыкновенными локальными таблицами.

Просмотры или виртуальные таблицы для клиентского приложения выглядят как обыкновенные реальные таблицы, поэтому для них тоже можно использовать компоненты TTable.

Описание использования других компонент достаточно объемно и выходит за рамки методических указаний. Поэтому для получения этой информации обращайтесь к соответствующей литературе.

Создание отчетов

Для создания отчета используется компонент TQuickRep, на котором с помощью компонентов TQRBand, TQRSubDetail, TQRGroup задаются области отчета, на которых размещаются компоненты TQRLabel, TQRDBText,TQRExpr, TQRSysData и другие, которые отображают информацию в отчете.

Далее будут перечислены компоненты и их назначение, для более подробной информации обращайтесь к литературе по Delphi и встроенной помощи Delphi.

- TQRBand — область для расположения данных, заголовков, титула отчета и др. Свойство BandType позволяет более конкретно определить функцию этого компонента в отчете.
- TQRSubDetail — область отчета, в которой располагаются данные подчиненной таблицы при реализации в отчете связи Master-Detail.
- TQRGroup — применяется для группировки данных в отчете согласно значению выражения, которое задается в свойстве Expression данного компонента

- TQRLabel — позволяет разместить в отчете статический текст, надписи и т.д.
- TQRDBText — позволяет разместить в отчете содержимое полей наборов данных.
- TQRExpr — позволяет разместить в отчете значения. являющиеся результатом вычисления выражений. Алгоритм вычисления выражений строится с помощью редактора формул этого компонента.
- TQRMemo — для вывода значений полей комментариев
- TQRRichText — вывод полей RTF
- TQRShape — вывод графических фигур, например рамок, линий
- TQRImage — вывод графической информации из баз данных
- TQRChart — вывод в отчет графиков, построенных, например, по информации, содержащейся в базе данных.
- TQRSysData — позволяет встроить в отчет вывод некоторой специальной информации. например текущей страницы, колонки, даты, времени, количества информации в наборе данных и т.д.

Простой отчет должен демонстрировать применение основных компонентов типа TQRLabel. TQRShape, TQRDBText.

Сложный отчет должен включать в себя использование колонтитулов, выборку данных из нескольких таблиц, группировку данных, подсчет агрегатной информации.

Контрольные вопросы.

1. Перечислите функции СУБД при обращении к базе данных из приложения.
2. Нужна ли поддержка базы данных для откомпилированного приложения.
3. Какие языки используются для поддержки базы данных?
4. Что Вы понимаете под хранимыми процедурами
5. Отличаются ли языки SQL-типа современных СУБД?

Варианты заданий к лабораторным работам

Таблица 1

Вид информац.системы	Предметная бласть	Точка зрения
к	j	n

Таблица 2

Наименование предметной области (j)

1 Поставщик - поставка

2 Библиотека

3 Учеба по специальности ПОИТ

5 Компьютер

6 Автомобиль

7 Дорожное движение

8 Вождение автомобиля

9 Деканат дневного отделения

10 Операционная система MSDOS

Язык программирования (ПАСКАЛЬ, Си, Basic и т.д.)

11 Средняя школа

12 ВУЗ

13 Кафедра (по выбору)

14 Цех

15 Отделение поликлинники

16 ВЦ

17 Бухгалтерия предприятия

18 Малое предприятие (по выбору)

Таблица 3

	Вид информационной системы (k)
1	Обучающая
2	Диагностирующая
3	Консультирующая
4	Прогнозирующая
5	Советующая
6	Анализирующая
7	Поисковая
8	Планирующая
9	Составляющая расписание
10	Составляющая календарный план
11	Тестирующая

Таблица 4

№	n	Точка зрения пользователя БД
	1 2 3 4	Преподаватель кафедры
	5 6 7 8	Студент, Поставщик, Директор
	9 10	Поставщик, Директор, Рабочий
	11 12	Бухгалтер, Начальник, ОТК
	13 14	Начальник отдела, Оператор ВЦ
	15 16	Преподаватель школы, Зам.
	17	Проректора, Абонент библиотеки, Системный

		программист, Прикладной программист, Врач

ЛИТЕРАТУРА

1. Четвериков В.Н. и другие
"Базы и банки данных". Москва, ВШ, 1987г.

2. Кливерт Ч. Энциклопедия пользователя Delphi2,DiaSoft,Киев,1996г.

3. Т. Коннолли, К Брегг. Базы данных. проектирование, реализация и сопровождение теория и практика, Университет Пейсли, Шотландия, изд М • СПБ •Киев, 2003.

4. А.Я. Архангельский Delphi 7, Москва, изд. «Бином», 2003.