ПРИМЕНЕНИЕ ЯЗЫКА SQL ДЛЯ УПРАВЛЕНИЯ ДАННЫМИ

Методические указания к лабораторной работе №2

1. ЦЕЛЬ РАБОТЫ

Целью работы является приобретение практических навыков работы с реляционными базами данных, используя язык SQL.

2. КРАТКАЯ ТЕОРЕТИЧЕСКАЯ СПРАВКА

В системах управления базами данных (СУБД) управление данными происходит при помощи специальных языков – *языков управления данными*. В реляционных СУБД таким языком является *язык структурированных запросов* (Structured Query Language – SQL).

Язык SQL является стандартным языком доступа к базам данных, поэтому программы – приложения реляционных баз данных, созданные на одной платформе СУБД, переносимы на другую платформу. Стандарт языка SQL развивался с 1986–го года путем расширения его возможностей. Последняя версия языка SQL относится к 2008-му году. В настоящее время, ни одна реляционная платформа СУБД не реализует стандарт SQL в полном объеме. Однако, поддерживая стандарт в целом, все платформы СУБД реализуют тот или иной *диалект* SQL. Это создает проблемы пари переносе приложений, которые, тем не менее, легко разрешимы.

Стандарт языка SQL основан на теории реляционных баз данных, включающей *реляционную алгебру* и *реляционное исчисление* [4]. Это означает, что любой оператор реляционной алгебры может быть выражен соответствующим оператором языка SQL. Вместе с тем, в языке SQL используется терминология, отличающаяся от терминологии реляционной теории. В таблице 1 приведены некоторые примеры различия терминов, относящихся к таблицам.

Таблица 1. Соответствие терминов, относящихся к таблицам.

Обычный язык	Язык SQL	Реляционная теория
Таблица	Таблица	Отношение
Строка	Запись	Кортеж
Столбец	Поле	Атрибут
Ключ	Первичный ключ Внешний ключ	Ключ

Язык SQL описан в обширной литературе, ссылки на которую можно найти в [1] Спецификацию языка SQL, поддерживаемую сервером Sybase ASA, можно найти в [2]. Изучать язык SQL лучше на примерах и решая задачи [3].

2.1. Структура языка SQL.

Язык SQL представляет собой набор операторов. Язык SQL не является алгоритмическим языком и относится к интерпретируемым языкам. Это значит, что каждый оператор языка SQL выполняется непосредственно при его появлении. Сервер базы данных «понимает» язык SQL и превращает его операторы в *транзакции* – команды непосредственного управления данными базы.

Все множество операторов SQL делится на подмножества, которые считаются подъязыками SQL. В каждом подмножестве используются определенные операторы, заданные их ключевыми словами. Подъязыками SQL являются следующие группы операторов.

Операторы определения данных (*Data Definition Language, DDL*)

- CREATE - создает объект БД (саму базу, таблицу, представление, пользователя);
- ALTER - изменяет объект;
- DROP - удаляет объект.

Операторы манипуляции данными (*Data Manipulation Language, DML*)

- SELECT - считывает данные, удовлетворяющие заданным условиям;
- INSERT - добавляет новые данные;
- UPDATE - изменяет существующие данные;
- DELETE - удаляет данные.

Операторы определения доступа к данным (*Data Control Language, DCL*)

- GRANT - предоставляет пользователю (группе) разрешения на определенные операции с объектом;
- REVOKE - отзывает ранее выданные разрешения;
- DENY - задает запрет, имеющий приоритет над разрешением.

Операторы управления транзакциями (*Transaction Control Language, TCL*)

- COMMIT - применяет транзакцию;
- ROLLBACK - отменяет все изменения, сделанные в контексте текущей транзакции;
- SAVEPOINT - делит транзакцию на более мелкие участки.

2.2. Синтаксис языка SQL

В стандарте ANSI/ISO синтаксис языка SQL описывается с помощью формальной BNF-нотации (форма Бэкуса-Наура). Эта нотация позволяет с высокой степенью детализации описать все возможные варианты синтаксиса операторов SQL, но является трудной для чтения и восприятия. Элементы BNF-нотации приведены в приложении 1. Их знания достаточно для знакомства со спецификацией языка SQL.

Приведем упрощённое описание синтаксиса четырех наиболее востребованных операторов SQL: INSERT, DELETE, UPDATE, SELECT.

Оператор INSERT

Для добавления записей в таблицы используется оператор INSERT, синтаксис которого имеет вид:

INSERT [INTO] table ([column_list] { VALUES ({ DEFAULT | NULL | expression } } [, …])

Здесь table - имя таблицы, column_list - список полей. Ключевое слово VALUES необходимо, за ним следует список значений данных в полях таблицы. Ключевое слово DEFAULT определяет заданное значение по умолчанию. Ключевое слово NULL определяет неопределенные (необязательные) значения данных.

Пример 1. Вставка одной записи в таблицу:

```
INSERT INTO "city"  ( "city_id",  "value", "region" )
VALUES ( 5, 'Томск', 1 ) ;
```

Пример 2. Вставка в таблицу нескольких записей, выбранных из другой таблицы (в таблицу TMP_TABLE вставляются данные о поставщиках из таблицы P, имеющие номера, большие 2):

```
INSERT INTO
  TMP_TABLE (PNUM, PNAME)
  SELECT PNUM, PNAME
    FROM P
    WHERE P.PNUM>2;
```

Оператор DELETE

Для удаления записей из таблиц следует использовать оператор DELETE, синтаксис которого имеет вид:

DELETE FROM table [WHERE criteria]

Внимание! Предложение WHERE не является обязательным, но если его не включить, то из таблицы будут удалены все записи

Пример 3. Удаление всех записей в таблице P:

```
DELETE FROM P;
```

Пример 4. Удаление нескольких записей в таблице:

```
DELETE FROM P
  WHERE P.PNUM = 1;
```

Оператор UPDATE

Для изменения значений в одной или нескольких полях таблицы применяется оператор UPDATE. Синтаксис этого оператора имеет вид:

UPDATE table SET column1 = expression1 [, column2 = expression2] [,…] [WHERE criteria]

Пример 5. Обновление нескольких записей в таблице:

```
UPDATE P
  SET P.NAME = "Иванов"
  WHERE P.PNUM = 1;
```

Оператор SELECT.

Оператор SELECT является самым востребованным в практике управления данными и самым сложным оператором SQL. Он предназначен для выборки данных из таблиц. Оператор SELECT всегда выполняется над некоторыми таблицами, входящими в базу данных. Это могут быть постоянно хранимые таблицы и временные таблицы. Результатом выполнения оператора SELECT всегда является таблица. Эта таблица является виртуальной в том смысле, что она не хранится в базе данных, а конструируется в результате выполнения данного оператора и может содержать данные из нескольких реальных таблиц базы данных.

Синтаксис оператора SELECT. Приведем макро-вариант синтаксиса оператора SELECT, удобный для понимания его структуры:

SELECT <что искать> **FROM** <где искать>

[WHERE <условия поиска>]

[ORDER BY<как упорядочить результаты>]

Операторы SELECT должны содержать слова SELECT и FROM; другие ключевые слова, такие как WHERE или ORDER BY, являются необязательными.

Секция <что искать> оператора содержит *список выборки*. Список выборки включает множество элементов: список имен полей таблиц, их псевдонимы, встроенные функции, выражения, а также комбинации всех этих элементов.

Секция <где искать> оператора содержит *ссылки на таблицы* базы данных. Каждая такая ссылка представляет собой имя таблицы или *имя представления* или *псевдоним*.

Секция < условия поиска > может содержать как простые, так и сложные логические выражения, использующие поля таблиц, константы, сравнения (>, <, = и т.д.), скобки, союзы AND и OR, отрицание NOT.

Познакомиться более подробно с синтаксисом рассмотренных здесь операторов можно по литературе [1-3].

Пример 6. Выбрать все данные из таблицы (ключевые слова SELECT… FROM…):

```
SELECT *
  FROM P;
```

Замечание. В результате получим новую таблицу, содержащую полную копию данных из исходной таблицы P.

Пример 7. Выбрать все строки из таблицы, удовлетворяющие некоторому условию (ключевое слово WHERE…):

```
SELECT *
```

```
FROM P
WHERE P.PNUM > 2;
```

Пример 8. Выбранные строки поместить в создаваемую таблицу test.

```
SELECT * INTO test FROM fac
```

Пример 9. Выбрать строки, но не помещать их в создаваемую таблицу test.

```
SELECT * INTO test FROM fac
WHERE 1 = 0;
```

3. Работа с базами данных в системе PowerBuilder

Система Power Builder работает как с базами данных собственной платформы Sybase формата *.db, так и с базами данных других платформ. Чтобы узнать, как это происходит, рассмотрим некоторые понятия, применяемые в СУБД.

Формат базы данных. Все современные базы данных используют такие понятия как *таблица*, *запись*, *поле* и в рамках применяемой в них модели данных (иерархической, сетевой или реляционной) устроены одинаково. Однако, физические реализации моделей данных могут различаться. Исторически сложилось так, что компании – разработчики инструментальных СУБД создавали свои собственные физические реализации моделей данных, что привело к появлению различных форматов баз данных. Под форматом базы данных понимается внутреннее устройство базы данных - способ хранения данных, соответствующих таблицам и другим объектам, способ доступа к записям таблиц и другие программные решения на уровне выбранной файловой системы. Известны форматы dBASE, FoxPro, Paradox, Oracle, Sybase, Informix, MySQL.

Интерфейсы подключения к базам данных. Технология «клиент - сервер» допускает работу множества клиентов с одной базой данных. В связи с этим в терминологию СУБД вводится понятие *подключение к базе данных*. Подключиться к базе данных означает для клиента открыть возможность работать с данными базы – изменять, удалять или добавлять данные, а также, возможно, изменять структуру базы данных. Последнее, очевидно, требует введения определенных ограничений и диспетчеризации работы клиентов – этим занимается сервер базы данных.

Подключение к базе данных представляет собой набор операций сервера, которые инициируются командой CONNECT языка SQL. Работая с базой данных, клиенты используют такие понятия как *таблица*, *запись*, *поле*, *тип данных в поле*, *ограничения на данные в поле*, *индекс*, *ключ*. При обращении к серверам всех реляционных баз данных используется язык SQL.

На уровне команд языка SQL клиенты работают с базой независимо от ее формата. Поэтому вполне естественным решением представляется создание

унифицированного механизма работы с базами данных разных платформ. В системе Power Builder подключение к базам данных выполняется при помощи единого универсального механизма, основанного на *интерфейсах.* На рис. 1 показаны установленные в системе Power Builder интерфейсы подключения к базам данных.

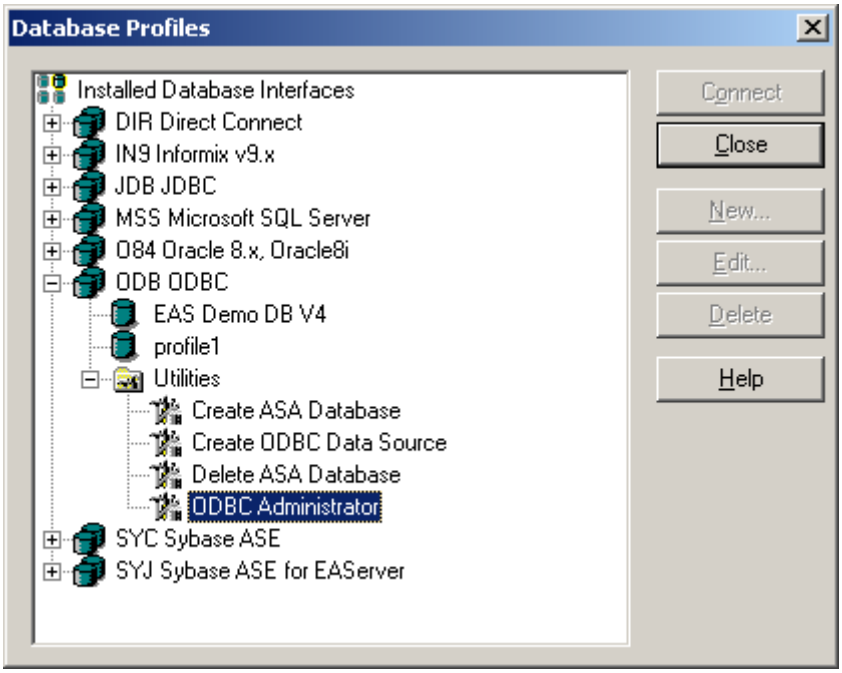


Рис. 1. Пример интерфейсов подключения к базам данных в системе Power Builder. Раскрыт интерфейс ODBC, имеющий два профиля. Показаны утилиты, среди которых Администратор ODBC.

При подключении к базе данных используются понятия *источник данных* и *профиль.*

3.1. Интерфейс ODBC. Описания источников данных. Профили данных (Profiles)

Система ODBC (Open Database Conectivity) — это программный интерфейс (API) доступа к базам данных, используемый в операционной системе Windows. Доступ программ к базам данных в ODBC выполняется при помощи источников данных.

Источник данных – это объект ODBC, представляющий собой настройки драйвера, который используется в Windows для связи с сервером базы данных.

Создание источника данных.

Создать источник данных можно двумя способами: вызвать Администратор источников данных непосредственно в Windows или вызвать его в папке Utilities системы PowerBuilder, как показано на рис. 2.

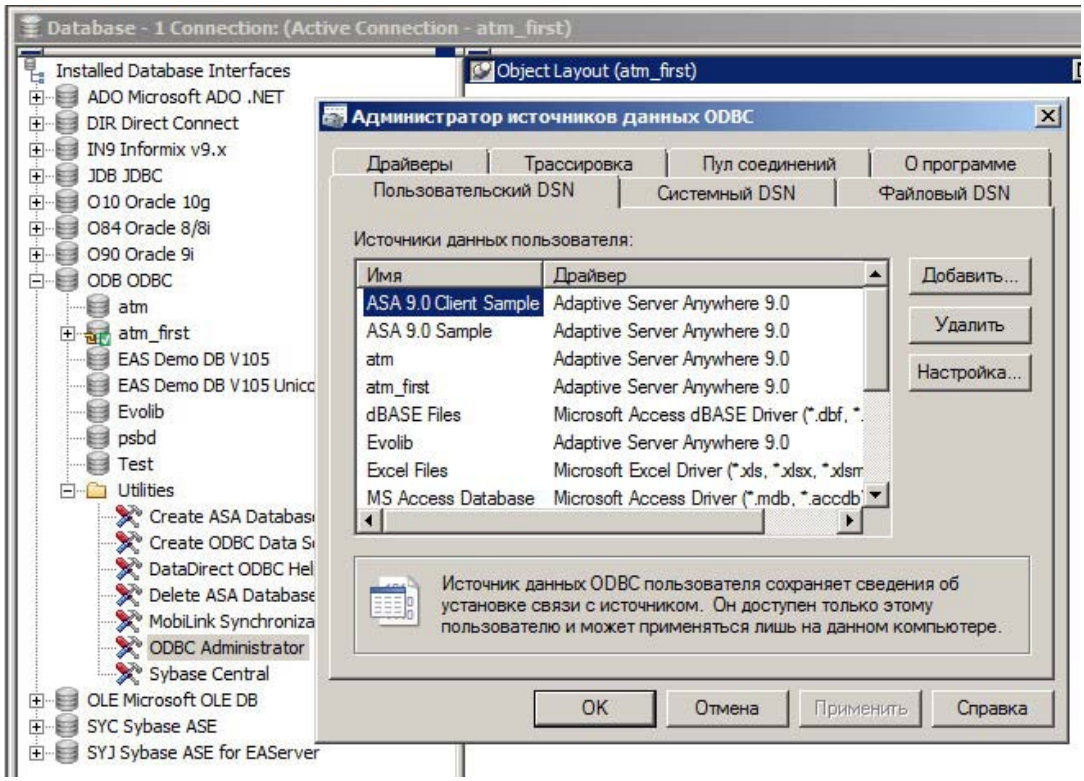


Рис. 2. Вызов Администратора источников данных Windows.

В появившемся диалоговом окне будет выведен список всех драйверов ODBC, установленных на данном компьютере, и имена конфигураций источников данных, существующих для этого драйвера.

Чтобы создать новую конфигурацию источника данных, следует нажать кнопку Add (Добавить), выбрать драйвер и в диалоговом окне ODBC Configuration указать требуемые параметры. Для каждого драйвера существует свое диалоговое окно. На рис. 3 приведен пример окна для источника данных Adaptive Server Anywhere.

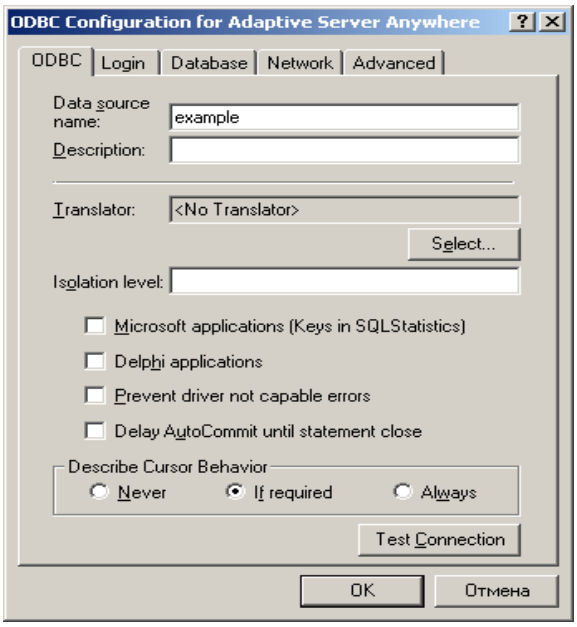


Рис. 3. Диалоговое окно для источника данных Adaptive Server Anywhere.

В закладке ODBC данного окна следует заполнить текстовое поле Data Source Name, указав имя источника данных. Затем в закладке Login в текстовые поля User ID и Password следует ввести имя пользователя базой данных и пароль. При создании базы данных Power Builder по умолчанию задает имя пользователя dba и пароль sql, как показано на рис. 4.

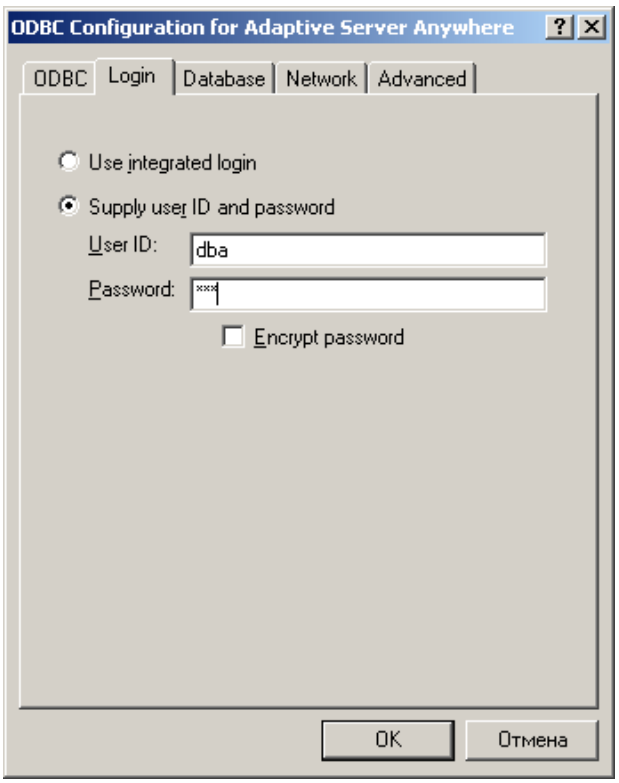


Рис. 4. Ввод имени и пароля пользователя базы данных.

После этого в закладке Database следует заполнить поле Database file, указав путь к файлу базы данных, как показано на рис. 5.

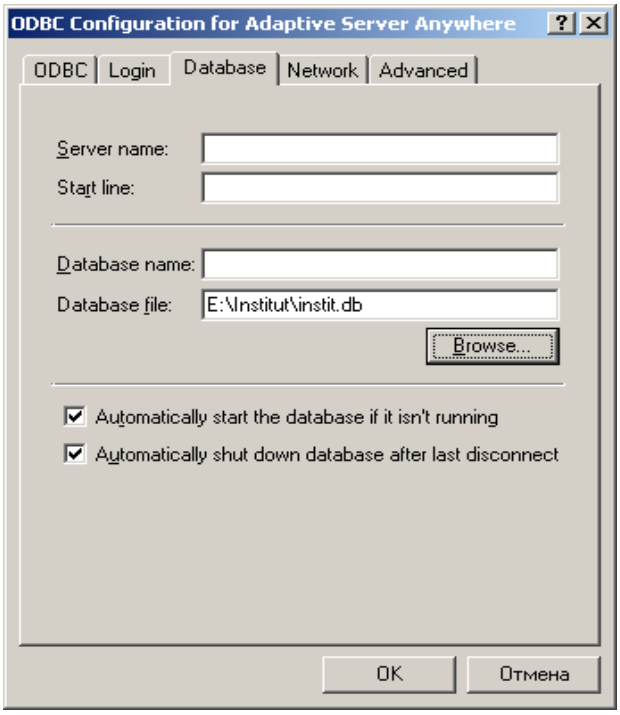


Рис. 5. Задание маршрута к файлу базы данных.

После нажатия кнопки OK к списку существующих источников данных будет добавлен новый (рис.6).

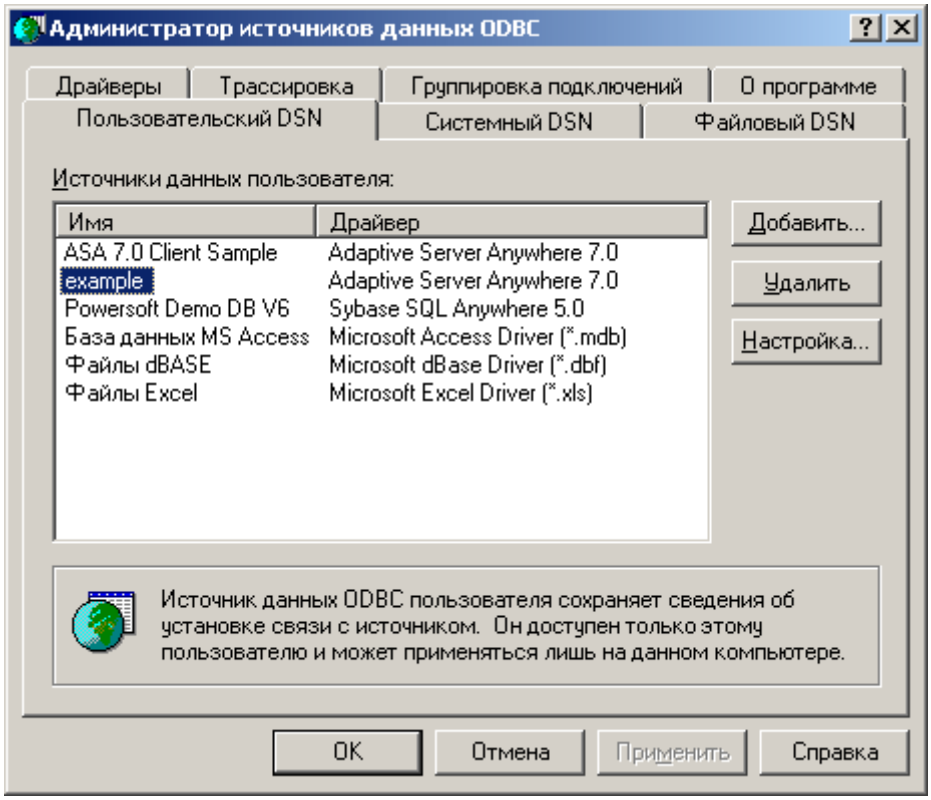
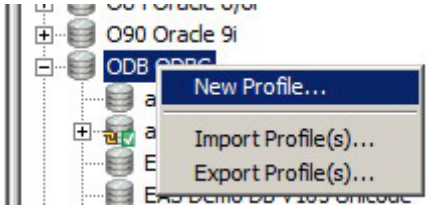


Рис. 6. Список источников данных в системе ODBC.

Создание профиля подключения к базе данных.

Профиль подключения к базе данных необходим в системе PowerBuilder для работы в мастерской баз данных с использованием всех ее возможностей: создание таблиц, индексов, ключей, изменение структуры таблиц, занесение записей в таблицы, выполнение операторов языка SQL.

Для создания профиля в контекстном меню, вызываемом кликированием правой кнопки мыши, выбрать New Profile:



Или в окне Database Profiles (Профили баз данных) (рис. 7) следует нажать кнопку New….

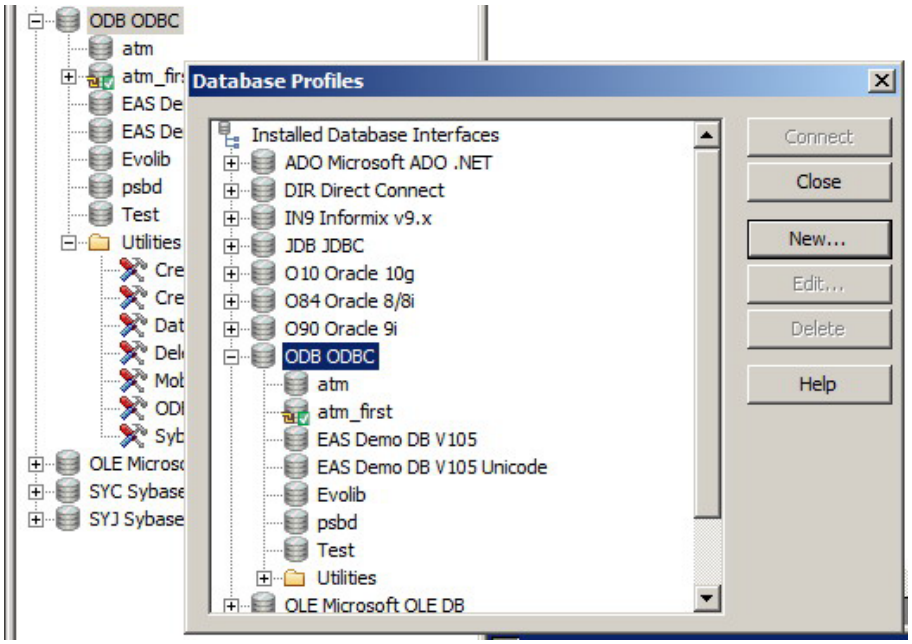


Рис. 7. Окно профилей подключения.

При нажатии кнопки New будет открыто диалоговое окно Database Profile Setup (Установка профиля баз данных), показанное на рис. 8, текстовые поля которого необходимо заполнить:

- текстовое поле Profile Name (Имя профиля) является уникальным идентификатором базы данных для Power Builder и только по этому имени Power Builder будет обращаться к этой базе данных; имя базы данных в профиле необязательно должно соответствовать действительному наименованию базы данных;

- текстовое поле Data Source - имя источника данных ODBC, который используется для работы с базой данных.

Если имя пользователя и пароль уже заданы при определении источника данных, то их можно не задавать в профиле подключения.

Приведенная ниже таблица содержит объяснения параметров, используемых для задания профиля. Не все параметры, указанные в таблице, обязательно должны быть объявлены в профиле: какие параметры необходимы, зависит от типа базы данных.

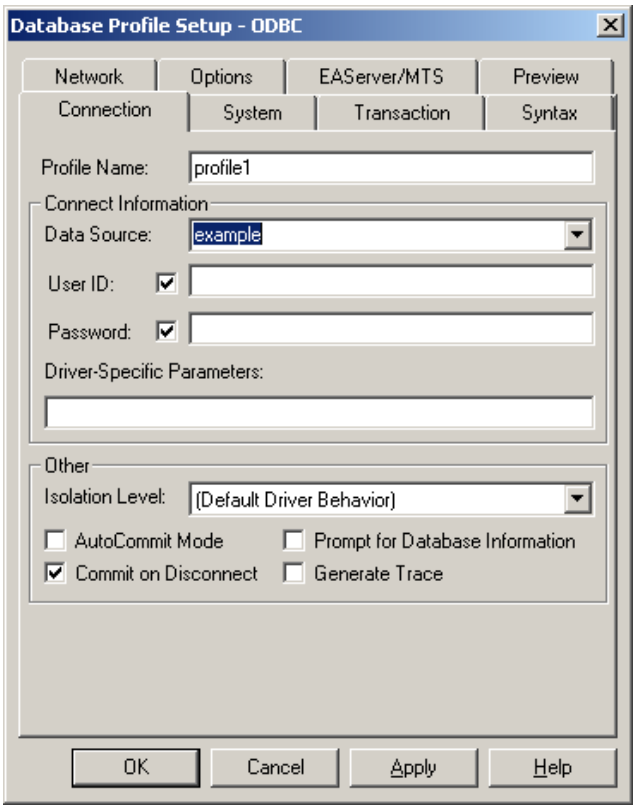


Рис. 8. Окно установки профиля баз данных

Атрибут	Тип	Описание
DBMS	String	Название интерфейса к СУБД, например, ODBC, Oracle или Sybase
Database	String	Имя базы данных. Под этим именем будет создан профиль для базы данных
UserId	String	Имя или идентификатор пользователя базы данных
DBParm	String	Значение этого атрибута зависит от типа СУБД
DBPass	String	Пароль для доступа к базе данных
Lock	String	Тип блокировок
LogId	String	Имя или идентификатор пользователя сервера
LogPass	String	Пароль пользователя сервера
ServerName	String	Имя сервера, на котором находится база данных
AutoCommit	Boolean	Индикатор автоматического завершения транзакции: TRUE – завершать транзакцию после каждой операции над базой данных, FALSE – по умолчанию, не завершать транзакцию автоматически.

Профиль подключения представляет собой текстовые данные, которые записываются либо в файл pb.ini, либо в реестр Windows. Пример текста профиля показан на рис. 9.

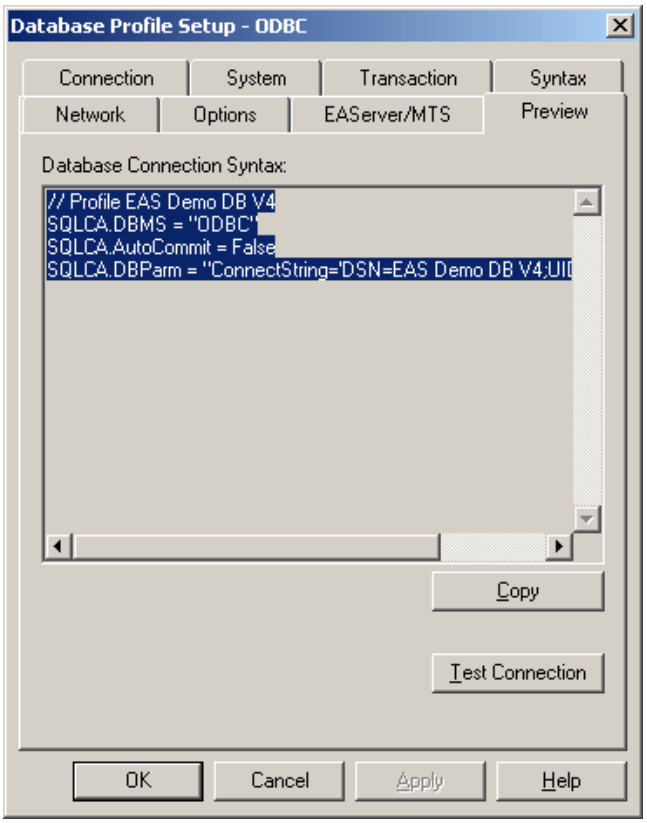


Рис.9. Текст профиля подключения к базе данных.

После закрытия диалогового окна подключиться к базе данных можно с помощью кнопки Connect окна Database Profiles или по щелчку мыши на имени профиля в мастерской баз данных, вызываемой с помощью кнопки  панели инструментов.

3.2. Конструктор операторов языка SQL

В системе PowerBuilder работа с базами данных осуществляется в мастерской баз данных. Порядок работы в мастерской баз данных описан в предыдущей работе. Здесь будет описан конструктор операторов языка SQL, который можно использовать при формировании операторов SQL при обращении к серверу.

На рис. 10 показано окно интерактивной консоли оператора базы данных ISQL Session и контекстное меню выбора операторов SQL.

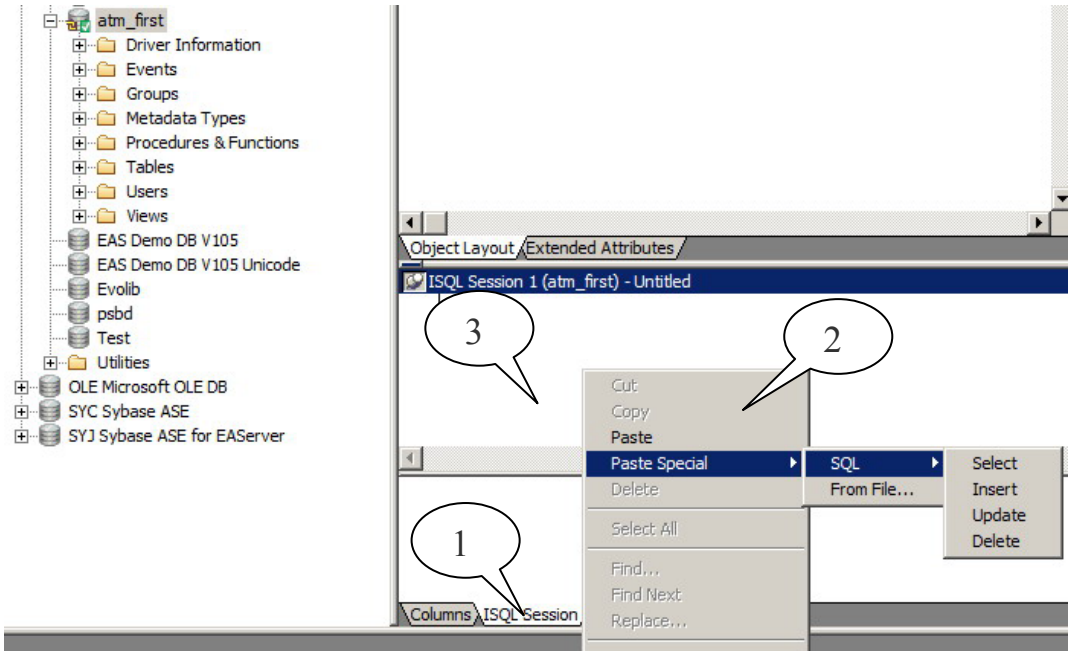
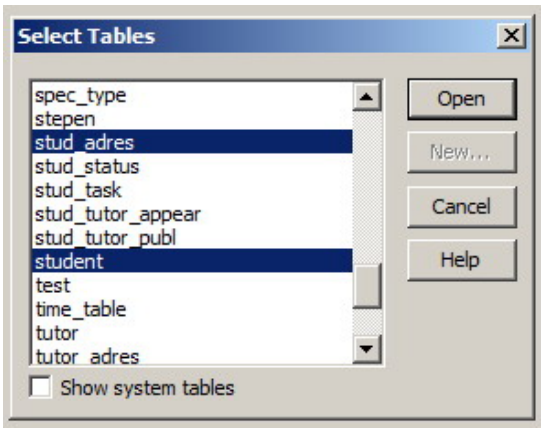


Рис.10. 1- закладка ISQL Session мастерской баз данных; 2 - контекстное меню выбора операторов SQL; 3 - окно интерактивной консоли оператора.

Для работы с конструктором необходимо выбрать закладку ISQL Session в мастерской баз данных и далее выбрать из меню необходимый оператор, как показано на рис. 10.

Рассмотрим работу с конструктором на примере оператора SELECT SQL. Выбрав данный оператор, далее необходимо сделать следующее, в соответствии со структурой оператора SELECT.

1. Выбрать таблицы, из которых будут извлекаться данные, в окне Select Tables:



2. Выбрать необходимые поля в таблицах:

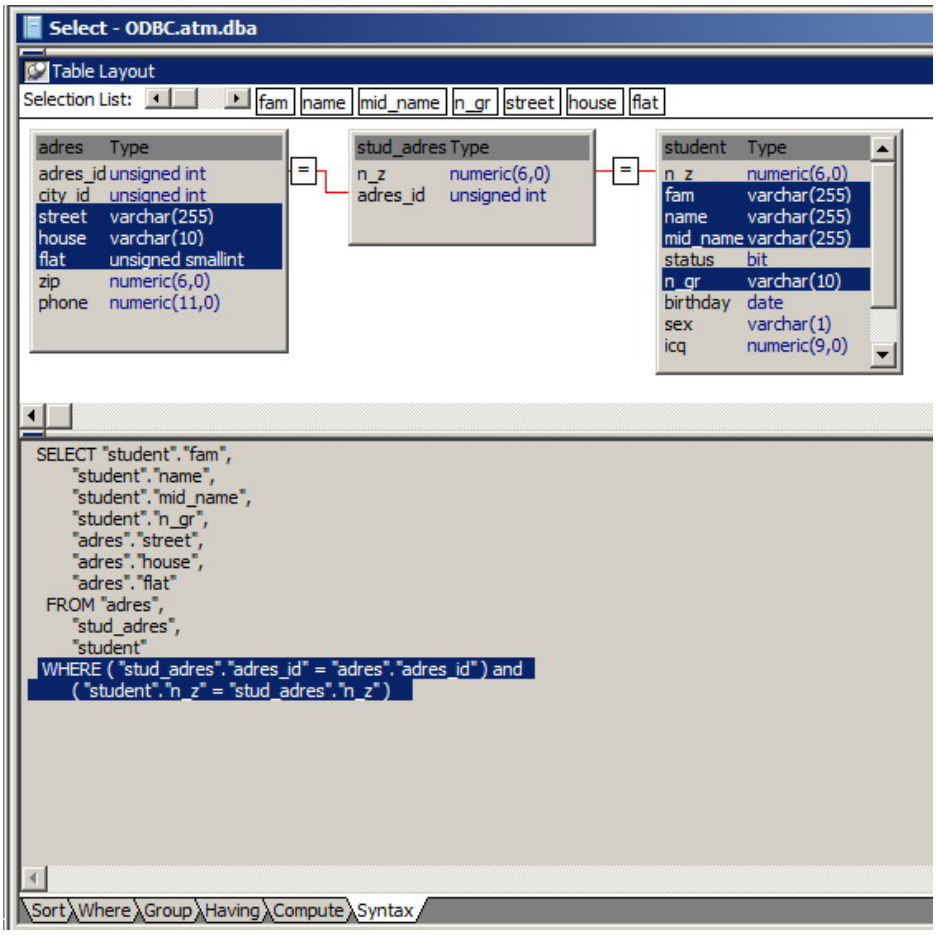


Рис.11. Формирование оператора SELECT

Если таблицы связаны, то конструктор сформирует условия связи автоматически, как показано на рис. 11.

3. Дополнительные условия в опции WHERE формируются в соответствующей закладке конструктора, как показано на рис. 12.

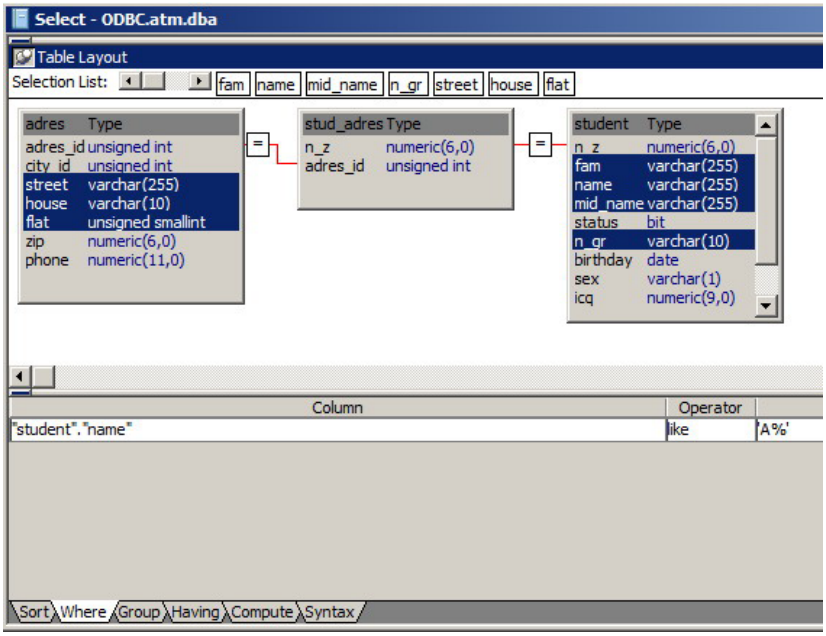


Рис.12. Формирование условий в опции WHERE оператора SELECT

4. Закрыть конструктор кнопкой  в левом верхнем углу и вернуться в консоль ISQL Session. Проверить получившийся оператор (рис. 13) и внести при необходимости в него изменения.

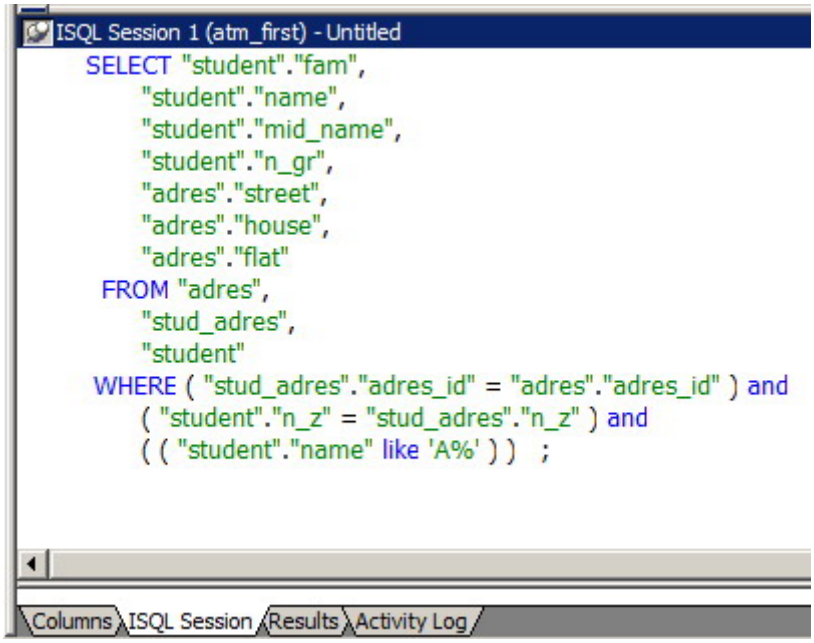


Рис.13. Результирующий оператор SELECT

5. Выполнить полученный оператор, инициировав кнопку  . Результат выполнения оператора на рис. 13 показан в закладке Results на рис. 14

Database - 1 Connection: (Active Connection - atm_first)

Select 1 Results (atm_first) - ISQL Session 1

Student Fam	Student Name	
Андрюшкин	Алексей	Михайлович
Катков	Александр	Михайлович
Кучеров	Андрей	Игоревич
Ромашкин	Андрей	Валериевич
Брыков	Александр	Евгеньевич
Жирнова	Алёна	Владимировна
Ивашкин	Андрей	Александрович
Крылова	Анастасия	Владимировна
Медведев	Александр	Павлович

Рис.14. Результат выполнения оператора SELECT на рис. 13.

Другие операторы языка SQL формируются с помощью конструктора с теми особенностями, которые определяются структурой оператора.

4. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ.

4.1. Изучить синтаксис операторов INSERT, DELETE, UPDATE, SELECT, используя литературные источники, приведенные в списке литературы.

4.2. Познакомиться с системой ODBC, используя ее подсистему помощи. Изучить виды источников данных: файловый, системный, пользовательский.

4.3. Сформировать и выполнить операторы INSERT, DELETE, UPDATE, SELECT на таблицах созданной в лабораторной работе «Проектирование и создание баз данных» базы данных.

4.4. Подключиться к базе данных «Кафедра» по указанию преподавателя.

4.5. Решить задачи, тексты которых приведены в приложении.

5. СОДЕРЖАНИЕ ОТЧЕТА

Отчет по работе должен содержать:

- операторы INSERT, DELETE, UPDATE, SELECT на таблицах созданной в предыдущей лабораторной работе базы данных;
- решения задач из приложения 1.

При защите работы студент должен продемонстрировать умение составлять и выполнять операторы SQL в консоли оператора.

6. КОНТРОЛЬНЫЕ ВОПРОСЫ И УПРАЖНЕНИЯ

1. Можно ли, используя оператор INSERT, занести в таблицу несколько одинаковых записей?
2. К какому типу языков относится язык SQL?
3. Что такое скрипт?
4. Что означает выражение 'A%' в операторе SELECT на рис. 13?
5. В чем разница в применении операторов « = » и Like?
6. Дополните оператор на рис. 13 так, чтобы в таблице на рис. 14 заголовки были на русском языке.

7. В окне ODBC Configuration объяснить, для чего нужны и как работают вкладки Network и Advanced.

8. В окне Database Profile Setup объяснить, для чего нужны и как работают вкладки Network, Options, EAServer, Transaction, Preview, Syntax, System.

9. Должно ли соответствовать имя базы данных в профиле действительному наименованию базы данных?

10. Может ли программа работать с базой данных, не имея профиля подключения?

Литература

1. Электронный ресурс: http://www.sql.ru/
2. Электронный ресурс: http://lis.tula.ru/Data/SQL_ru.pdf
3. Астахова И.Ф. SQL в примерах и задачах; Учеб. пособие / И.Ф. Астахова, А.П. Толстобров, В.М. Мельников.— М.: Издательство Физико-математической литературы, 2007. — 176 с.
4. Марков А.С., Лисовский К.Ю. Базы данных. Введение в теорию и методологию. М.: Финансы и статистика, 2006. - 512 с.
5. Богатырев М.Ю. разработка и программирование систем управления базами данных. - Тула, изд-во ТулГУ, 2009. - 145 с.
6. Богатырев М.Ю. Введение в систему Power Builder. Методические указания к выполнению лабораторных работ. - Тула, изд-во ТулГУ, 1998. - 36 с.
7. Смит Б. Дж., Шаад Г.У. Power Builder 5.0. Библия разработчика. - К.: Диалектика, 1997. - 544 с.
8. Хайес В.Б. Использование Power Builder 6. - Киев: Вильямс, 1998. - 688 с.

Приложение 1

ЗАДАЧИ

Задача №1

Внести в базу данных «Кафедра» Ваши личные данные.

Задача №2

Получить список всех кураторов групп одной специальности

Задача №3

Вывести попарно список всех студентов так, что дата рождения студента слева меньше дата рождения студента справа.

Задача №4

Студентам, родившимся в 1986 – м году, увеличить на 10 баллов оценку за курсовой проект.

Задача №5

Какие оценки за курсовой проект получили студенты с именем «Андрей»?

Задача №6

Удалить записи для всех студентов, номера зачеток которых имеют длину меньше 6-ти символов.