

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Локтионова Оксана Геннадьевна

Должность: проректор по учебной работе

Дата подписания: 08.02.2021 16:45:45

Уникальный программный ключ:

0b817ca911e6668abb13a5d426d39e5f1c11eabbf73e943df4a4851fda56d089

МИНОБРНАУКИ РОССИИ

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

Кафедра информационной безопасности

УТВЕРЖДАЮ

Проректор по учебной работе

_____ **О.Г. Локтионова**

« _____ » _____ **2017 г.**

ЗАПОЛНЕНИЕ БД В СРЕДЕ MS SQL SERVER 2008 R2. ВЫБОРКА ДАННЫХ С ПОМОЩЬЮ ЗАПРОСОВ

**Методические указания по выполнению лабораторной работы
№2**

**для студентов направления подготовки бакалавриата
10.03.01 «Информационная безопасность»**

Курск 2017

УДК 621.(076.1)

Составитель: А.Г. Спеваков

Рецензент

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

Заполнение БД. Выборка данных с помощью запросов
[Текст] : методические указания по выполнению лабораторной работы / Юго-Зап. Гос. ун-т; сост.: А.Г. Спеваков. – Курск, 2017. – 31с.: ил. 28, табл. 7. – Библиогр.: с. 31.

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

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

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

Текст печатается в авторской редакции

Подписано в печать . Формат 60x84 1/16.
Усл.печ. л. 1,80. Уч.-изд. л. 1,63. Тираж 100 экз. Заказ. Бесплатно.
Юго-Западный государственный университет.
305040, г.Курск, ул. 50 лет Октября, 94.

СОДЕРЖАНИЕ

| | |
|---------------------------------|----|
| Введение | 4 |
| Цель работы..... | 5 |
| Порядок выполнения работы | 6 |
| Содержание отчета | 7 |
| Теоретическая часть | 8 |
| Выполнение работы | 14 |
| Контрольные вопросы..... | 30 |
| Библиографический список..... | 31 |

ВВЕДЕНИЕ

В настоящее время для языка SQL существуют международные стандарты, формально определяющие его как стандартный язык создания и манипулирования реляционными базами данных, каковым он фактически и является.

Структура команд задается набором ключевых слов, представляющих собой обычные слова английского языка, такие как CREATE TABLE (Создать таблицу), INSERT (Вставить), SELECT (Выбрать), UPDATE (Обновить), DELETE (удалить).

ЦЕЛЬ РАБОТЫ

Заполнить полученную базу данных в SQL Management Studio и сформировать SQL запросы на выборку данных из базы данных.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Получить задание.
2. Изучить теоретическую часть.
3. Заполнить базу данных в SQL Management Studio.
4. С помощью SQL запросов произвести выборку данных из БД.

СОДЕРЖАНИЕ ОТЧЕТА

1. Содержание.
2. Индивидуальное задание.
3. Результаты заполнения БД в среде MS SQL Server 2008 (не менее 5 записей в каждой таблице).
4. Текст SQL-скриптов на выборку данных (не менее 2).
5. Результаты работы скриптов на выборку из БД.
6. Контрольные вопросы.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Язык SQL – это непроцедурный язык, поэтому в нем необходимо указывать, какая информация должна быть получена, а не как ее можно получить. Иначе говоря, язык SQL не требует указания методов доступа к данным.

Структура команд задается набором ключевых слов, представляющих собой обычные слова английского языка, такие как CREATE TABLE (Создать таблицу), INSERT (Вставить), SELECT (Выбрать), UPDATE (Обновить), Delete (удалить).

В настоящее время для языка SQL существуют международные стандарты, формально определяющие его как стандартный язык создания и манипулирования реляционными базами данных, каковым он фактически и является.

Для определения формата операторов SQL мы будем применять следующую расширенную форму системы обозначений BNF (Backus Naur Form—форма Бэкуса-Наура)

1. Прописные буквы будут использоваться для записи зарезервированных слов и должны указываться в операторах точно так же, как это будет показано.
2. Строчные буквы будут использоваться для записи слов, определяемых пользователем.
3. Вертикальная черта (|) указывает на необходимость выбора одного из нескольких приведенных значений, например a | b | c.
4. Фигурные скобки определяют обязательный элемент, например {a}.
5. Квадратные скобки определяют необязательный элемент, например [a].

Структура SQL

В отличие от реляционной алгебры, где были представлены только операции запросов к БД, SQL является полным языком, в нем присутствуют не только операции запросов, но и операторы, соответствующие Data Definition Language (DDL) — языку описания данных. Кроме того, язык содержит операторы, предназначенные для управления (администрирования) БД.

Таблица1 - Операторы определения данных DDL

| Оператор | Действие |
|-------------|---|
| CREATETABLE | Создает новую таблицу в БД |
| DROPTABLE | Удаляет таблицу из БД |
| ALTERTABLE | Изменяет структуру существующей таблицы или ограничения целостности, задаваемые для данной таблицы |
| CREATEVIEW | Создает виртуальную таблицу, соответствующую некоторому SQL-запросу |
| ALTERVIEW | Изменяет ранее созданное представление |
| DROP VIEW | Удаляет ранее созданное представление |
| CREATEINDEX | Создает индекс для некоторой таблицы для обеспечения быстрого доступа по атрибутам, входящим в индекс |
| DROPINDEX | Удаляет ранее созданный индекс |
| DELETE | Удаляет одну или несколько строк, соответствующих условиям фильтрации, из базовой таблицы. Применение оператора согласуется с принципами поддержки целостности, поэтому этот оператор не всегда может быть выполнен корректно, даже если синтаксически он записан правильно |
| INSERT | Вставляет одну строку в базовую таблицу. Допустимы модификации оператора, при которых сразу несколько строк могут быть перенесены из одной таблицы или запроса в базовую таблицу |
| UPDATE | Обновляет значения одного или нескольких столбцов в одной или нескольких строках, соответствующих условиям фильтрации |
| SELECT | Оператор, заменяющий все операторы реляционной алгебры и позволяющий сформировать результирующее отношение, соответствующее запросу |

Таблица2 - Средства администрирования данных

| Оператор | Смысл | Действие |
|----------------|------------------------------|---|
| ALTERDATABASE | Изменить БД | Изменить набор основных объектов в базе данных, ограничений, касающихся всей базы данных |
| ALTERDBAREA | Изменить область хранения БД | Изменить ранее созданную область хранения |
| ALTERPASSWORD | Изменить пароль | Изменить пароль для всей базы данных |
| CREATEDATABASE | Создать БД | Создать новую базу данных, определив основные параметры для нее |
| CREATEDBAREA | Создать область хранения | Создать новую область хранения и сделать ее доступной для размещения данных |
| DROPDATABASE | Удалить БД | Удалить существующую базу данных (только в том случае, когда вы имеете право выполнить это действие) |
| DROP DBAREA | Удалить область хранения БД | Удалить существующую область хранения (если в ней на настоящий момент не располагаются активные данные) |
| GRANT | Предоставить права | Предоставить права доступа на ряд действий над некоторым объектом БД |
| REVOKE | Лишить прав | Лишить прав доступа к некоторому объекту или некоторым действиям над объектом |

Таблица3 - Средства управления транзакциями

| Оператор | Смысл | Действие |
|-----------|---|---|
| COMMIT | Завершить транзакцию | Завершить комплексную взаимосвязанную обработку информации, объединенную в транзакцию |
| ROLLBACK | Откатить транзакцию | Отменить изменения, проведенные в ходе выполнения транзакции |
| SAVEPOINT | Сохранить промежуточную точку выполнения транзакции | Сохранить промежуточное состояние БД, пометить его для того, чтобы можно было в дальнейшем к нему вернуться |

Таблица4 - Программный SQL

| Оператор | Смысл | Действие |
|----------|--|---|
| DECLARE | Определяет курсор для запроса | Задаёт некоторое имя и определяет связанный с ним запрос к БД |
| OPEN | Открыть курсор | Формирует виртуальный набор данных, соответствующий описанию указанного курсора и текущему состоянию БД |
| FETCH | Считать строку из множества строк, определенных курсором | Считывает очередную строку, заданную параметром команды из виртуального набора данных, соответствующего открытому курсору |
| CLOSE | Закрывает курсор | Прекращает доступ к виртуальному набору данных, соответствующему указанному курсору |
| PREPARE | Подготовить оператор SQL к динамическому выполнению | Сгенерировать план выполнения запроса, соответствующего заданному оператору SQL |

Обработка элементов оператора SELECT выполняется в следующей последовательности.

1. FROM. Определяются имена используемой таблицы или нескольких таблиц.
2. WHERE. Выполняется фильтрация строк объекта в соответствии с заданными условиями.
3. GROUP BY. Образуются группы строк, имеющих одно и то же значение в указанном столбце.
4. HAVING. Фильтруются группы строк объекта в соответствии с указанным условием.
5. SELECT. Устанавливается, какие столбцы должны присутствовать в выходных данных.
6. ORDER BY. Определяется упорядоченность результатов выполнения оператора.

Порядок конструкций в операторе SELECT не может быть изменен. Только две конструкции оператора — SELECT и FROM — являются обязательными, все остальные конструкции могут быть опущены

Оператор Insert предназначен для добавления новых данных в таблицу. Он имеет следующий формат:

```
INSERT INTO TableName [(columnList)] VALUES (dataValueList);
```

Здесь параметр TableName (Имя таблицы) может представлять имя таблицы базы данных. Параметр columnList (Список столбцов) представляет собой список, состоящий из имен одного или более столбцов, разделенных запятыми. Параметр columnList является необязательным. Если он опущен, то предполагается использование списка из имен всех столбцов таблицы, указанных в том порядке, в котором они были описаны в операторе CREATE TABLE.

Оператор Update позволяет изменять содержимое уже существующих строк указанной таблицы. Этот оператор имеет следующий формат:

```
UPDATE tableName
```

```
SET columnName1 = dataValue1 [, columnName2 = dataValue2 ... ]  
[WHERE searchCondition];
```

Здесь параметр `tableName` представляет имя таблицы базы данных. В конструкции `SET` указываются имена одного или более столбцов, данные в которых необходимо изменить. Конструкция `WHERE` является необязательной. Если она опущена, значения указанных столбцов будут изменены во всех строках таблицы. Если конструкция `WHERE` присутствует, то обновлены будут только те строки, которые удовлетворяют условию поиска, заданному в параметре `searchCondition`.

Оператор `Delete` позволяет удалять строки данных из указанной таблицы. Этот оператор имеет следующий формат:

```
DELETE FROM tableName  
[WHERE searchCondition];
```

Как и в случае операторов `INSERT` и `UPDATE`, параметр `tableName` представляет собой таблицы базы данных. Параметр `searchCondition` является необязательным — если он опущен, из таблицы будут удалены все существующие в ней строки. Однако сама по себе таблица удалена не будет. Если необходимо удалить не только содержимое таблицы, но и ее определение, следует использовать оператор `DROP TABLE`. Если конструкция `WHERE` присутствует, из таблицы будут удалены только те строки, которые удовлетворяют условию отбора, заданному параметром `searchCondition`.

ВЫПОЛНЕНИЕ РАБОТЫ

Приступим к заполнению строк в наших таблицах. Для этого щелкаем правой кнопкой мыши по нужной нам таблице и выбираем Изменить первые 200 строк. (Рис. 1)

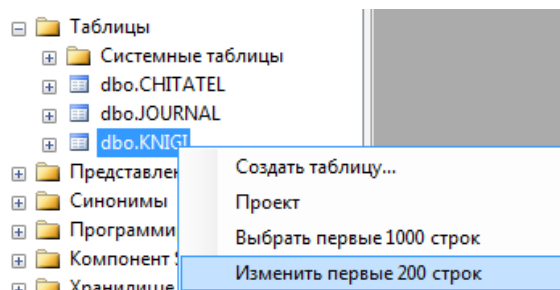


Рис.1 – Вызов окна заполнения таблиц

В открывшемся окне заполняем строки нужными данными. В нашем случае столбец ID – счётчик и заполняется автоматически.

| ВИКТОРИЯ-ПК\НО...ка - dbo.KNIGI | | | | | | | |
|---------------------------------|------------|------------------|------------------|----------|--------------|-------------|--|
| ID | Шифр_книги | Название | ФИО_автора | Жанр | Издательство | Год_издания | |
| 1 | 111 | Евгений Онегин | Александр Сер... | Классика | Перо | 2002 | |
| 2 | 222 | Преступление ... | Федор Михайло... | Классика | Москва | 2006 | |
| 3 | 333 | Горе от ума | Александр Сер... | Классика | Москва | 2008 | |

Рис.2 – Заполненная таблица KNIGI

Таким же образом заполняем остальные таблицы.

| ID | ФИО_читателя | Адрес | Дата_рождения | Пол | Место_работы |
|----|-------------------|-------------------|---------------|-----|---------------|
| 1 | Иванов Иван И... | Курск 50 лет О... | 1998-12-11 | муж | Такси |
| 2 | Кривцова Анна ... | Курск улица Ле... | 1993-07-14 | жен | ТРЦ Европа 40 |

Рис.3 – Заполненная таблица ЧИТАТЕЛ

| ID | Шифр_книги | ФИО_читателя | Дата_выдачи | Срок_сдачи |
|----|------------|--------------|-------------|------------|
| 1 | 1 | 2 | 2014-03-23 | 2014-08-23 |
| 2 | 3 | 1 | 2014-11-12 | 2014-12-12 |

Рис.4 – Заполненная таблица JOURNAL

Важно иметь в виду что столбцы Шифр_книги и Фιο_читателя таблицы JOURNAL заполняются только внесенными данными в столбец ID KNIGI и ID ЧИТАТЕЛ соответственно. В противном случае SQL выдаст ошибку (Рис. 5)

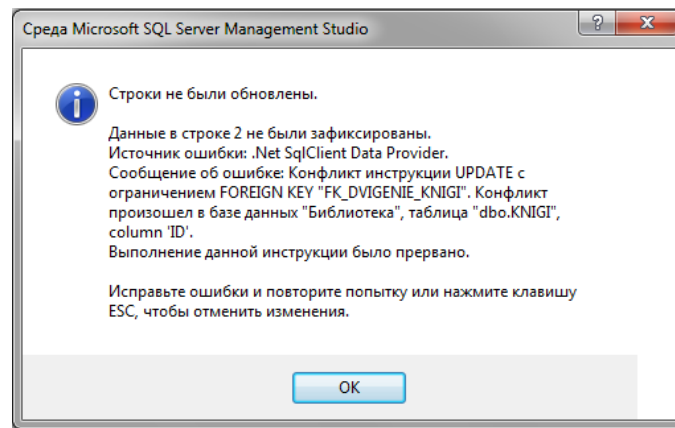


Рис.5 – Окно ошибки «Строки не были обновлены»

Теперь приступим к выборке данных с помощью запросов. Для начала рассмотрим 4 основных запроса SELECT, INSERT, UPDATE, DELETE. Будем работать с таблицей KNIGI.

SELECT – наиболее часто используемый SQL оператор. Он предназначен для выборки информации из таблиц. Чтобы при помощи оператора SELECT извлечь данные из таблицы, нужно указать как минимум две вещи — что вы хотите выбрать и откуда.

Щелкаем правой кнопкой мыши по нужной таблице, выбираем Создать сценарий для таблицы Используя SELECT Новое окно редактора запросов. (Рис. 6)

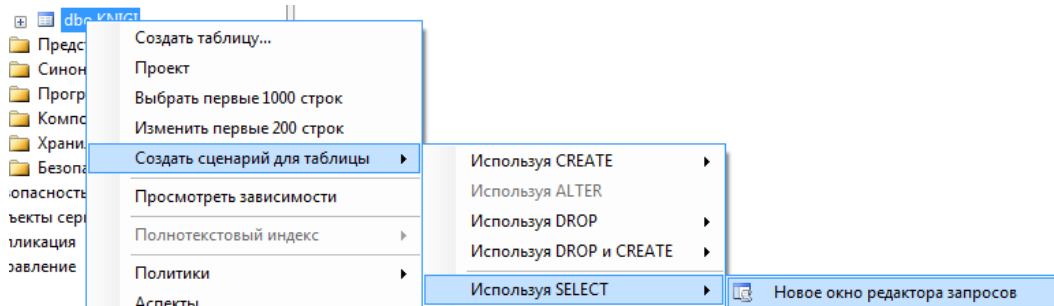
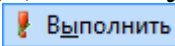


Рис.6 – Вызов окна редактора запросов

Сделаем выборку отдельных столбцов:

```
SELECT [Имя столбца]  
FROM Имя таблицы
```

Для создания и тестирования данного запроса в Management Studio выполните следующие шаги:

1. В открывшемся окне создания нового запроса введите представленные выше инструкции SQL.
2. Для запуска запроса на выполнение щелкните кнопку  на панели инструментов или нажмите клавишу F5. В нижней части экрана должны появиться результаты.

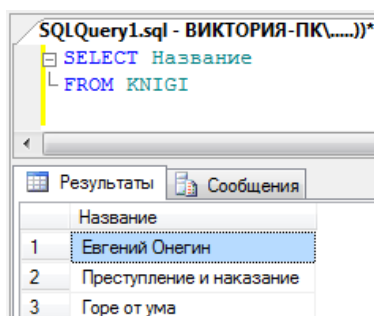


Рис.7 – Результат выполнения запроса

Сделаем выборку нескольких столбцов:

Для выборки из таблицы нескольких столбцов используется тот же оператор SELECT. Отличие состоит в том, что после ключевого слова SELECT необходимо через запятую указать несколько имен столбцов. (Рис. 8)

```
SELECT Имя столбца1, Имя столбца2  
FROM Имя таблицы
```

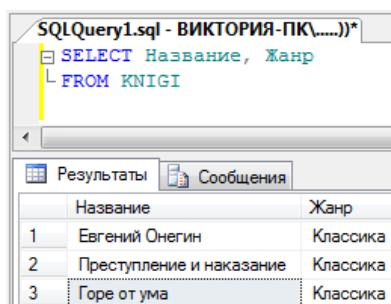


Рис.8 – Результат выполнения запроса

Выборка всех столбцов:

Помимо возможности осуществлять выборку определенных столбцов (одного или нескольких), при помощи оператора SELECT можно запросить все столбцы, не перечисляя каждый из них. Для этого вместо имен столбцов вставляется групповой символ “звездочка” (*). Это делается следующим образом. (Рис. 9)

```
SELECT * FROM Имя таблиц
```

| ID | Шифр_книги | Название | ФИО_автора | Жанр | Издательство | Год_издания | Страницы | Количество |
|----|------------|--------------------------|-------------------------------|----------|--------------|-------------|----------|------------|
| 1 | 111 | Руслан и Людмила | Александр Сергеевич Пушкин | Поэма | Перо | 2002 | 240 | 2 |
| 2 | 222 | Преступление и наказание | Федор Михайлович Достоевский | Классика | Москва | 2006 | 650 | 4 |
| 3 | 333 | Горе от ума | Александр Сергеевич Грибоедов | Классика | Москва | 2008 | 280 | 3 |

Рис.9 – Результат выполнения запроса

Сортировка данных:

В результате выполнения запроса на выборку данные выводятся в том порядке, в котором они находятся в таблице. Для точной сортировки выбранных при помощи оператора SELECT данных используется предложение ORDER BY. В этом предложении указывается имя одного или нескольких столбцов, по которым необходимо отсортировать результаты. (Рис. 10). Взгляните на следующий пример

```
SELECT Имя столбца1, Имя столбца2, Имя столбца3
FROM Название
ORDER BY Имя столбца3
```

| Название | ФИО_автора | Жанр |
|----------------------------|-------------------------------|----------|
| 1 Горь от ума | Александр Сергеевич Грибоедов | Классика |
| 2 Евгений Онегин | Александр Сергеевич Пушкин | Классика |
| 3 Преступление и наказание | Федор Михайлович Достоевский | Классика |

Рис.10 – Результат выполнения запроса

Указание направления сортировки:

В предложении ORDER BY можно также использовать порядок сортировки по убыванию. Для этого необходимо указать ключевое слово DESC. (Рис. 11)

```
SELECT Имя столбца1, Имя столбца2, Имя столбца3  
FROM Название  
ORDER BY Имя столбца3 DESC
```

Ключевое слово DESC применяется только к тому столбцу, после которого оно указано. Таким образом, столбец Имя столбца3 отсортирован в порядке убывания.

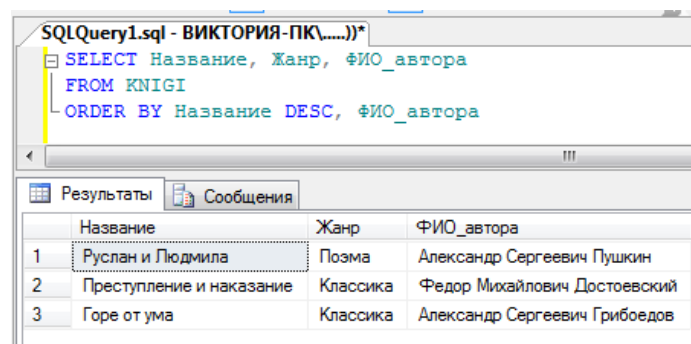


Рис.11 – Результат выполнения запроса

Фильтрация данных:

В таблицах баз данных обычно содержится много информации и довольно редко возникает необходимость выбирать все строки таблицы. Гораздо чаще бывает нужно извлечь какую-то часть данных таблицы для каких-либо действий или отчетов. Выборка только необходимых данных включает в себя критерий поиска, также известный под названием предложение фильтрации. В операторе SELECT данные фильтруются путем указания критерия поиска в предложении WHERE. Предложение WHERE указывается сразу после названия таблицы (предложения FROM) следующим образом (Рис. 12):

```
SELECT Имя столбца1, Имя столбца2, Имя столбца3  
FROM Название таблицы  
WHERE Имя столбца= 'значение'
```

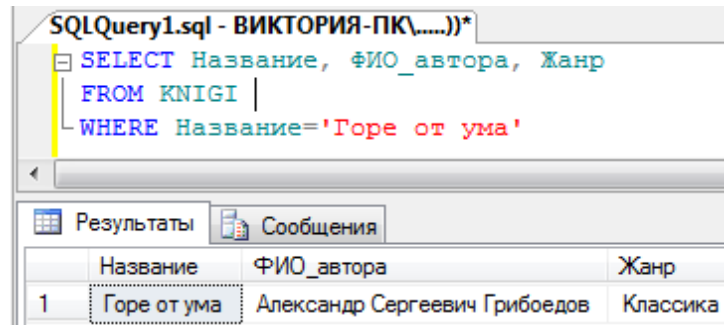


Рис.12 – Результат выполнения запроса

При совместном использовании предложений ORDER BY и WHERE, предложение ORDER BY должно следовать после WHERE.

SQL поддерживает весь спектр условных (логических) операций, которые приведены в следующей таблице.

Таблица5 – Условные операторы SQL

| Операция | Описание |
|----------|-----------------------------------|
| = | Равенство |
| <> | Неравенство |
| != | Неравенство |
| < | Меньше |
| <= | Меньше или равно |
| !< | Не меньше |
| > | Больше |
| >= | Больше или равно |
| !> | Не больше |
| BETWEEN | Между двумя указанными значениями |
| IS NULL | Значение NULL |

Для фильтрации данных по критерию соответствия определенной символьной строки заданному шаблону используется оператор LIKE. Шаблон может включать обычные символы и символы-шаблоны. Во время сравнения с шаблоном необходимо, чтобы его обычные символы в точности совпадали с символами, указанными в строке. Символы-шаблоны могут

совпадать с произвольными элементами символьной строки. Использование символов-шаблонов с оператором LIKE предоставляет больше возможностей, чем использование обычных операторов сравнения. Шаблон может включать в себя следующие символы-шаблоны.

Таблицаб – Символы-шаблоны оператора LIKE

| Символ-шаблон | Описание |
|---------------|--|
| % | Любое количество символов |
| _ | Любой одиночный символ |
| [] | Любой символ, указанный в квадратных скобках |
| [^] | Любой символ, кроме перечисленных в квадратных скобках |

Отбираются значения, соответствующие образцу (Рис. 13)

SELECT * FROM Название таблицы WHERE Имя столбца LIKE 'знач%'

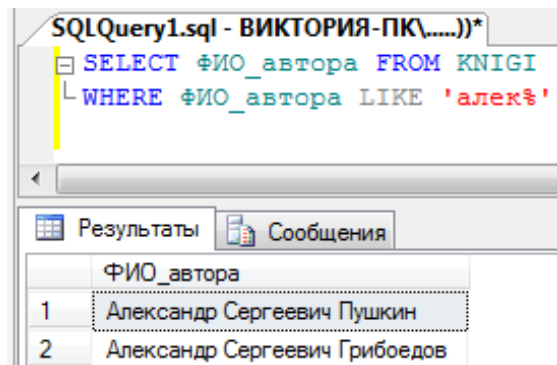


Рис.13 – Результат выполнения запроса

Исключение дублирующих записей:

Для исключения из результата выборки повторяющихся строк используется ключевое слово DISTINCT, которое указывается сразу после SELECT.

SELECT DISTINCT Столбец1

FROM Таблица1

Использование агрегатных функций:

В SQL определено множество встроенных функций различных категорий, среди которых особое место занимают агрегатные функции, оперирующие значениями столбцов множества строк и возвращающие одно значение. Аргументами агрегатных функций могут быть как столбцы таблиц, так и результаты выражений над ними. В следующей таблице приведены наиболее часто используемые стандартные унарные агрегатные функции.

Таблица7 - Стандартные унарные агрегатные функции

| Функция | Возвращаемое значение |
|---------|---|
| COUNT | Количество значений в столбце или строк в таблице |
| SUM | Сумма |
| AVG | Среднее |
| MIN | Минимум |
| MAX | Максимум |

Общий формат унарной агрегатной функции следующий:

имя_функции ([ALL | DISTINCT] выражение)

где DISTINCT указывает, что функция должна рассматривать только различные значения аргумента, а ALL — все значения, включая повторяющиеся (этот вариант используется по умолчанию). Например, функция AVG с ключевым словом DISTINCT для строк столбца со значениями 1, 1, 1 и 3 вернет 2, а при наличии ключевого слова ALL вернет 1,5.

Агрегатные функции применяются во фразах SELECT и HAVING. Здесь мы рассмотрим их использование во фразе SELECT. В этом случае выражение в аргументе функции применяется ко всем строкам входной таблицы фразы SELECT.

Функция COUNT имеет два формата. В первом случае возвращается количество строк входной таблицы, во втором случае — количество значений аргумента во входной таблице:

- COUNT(*)

- COUNT([DISTINCT | ALL] выражение)

Простейший способ использования этой функции — подсчет количества строк в таблице (всех или удовлетворяющих указанному условию). Для этого используется первый вариант синтаксиса. (Рис. 14)

```
SELECT COUNT(*) AS 'ИМЯ НОВОЙ ТАБЛИЦЫ'  
FROM Название таблицы
```

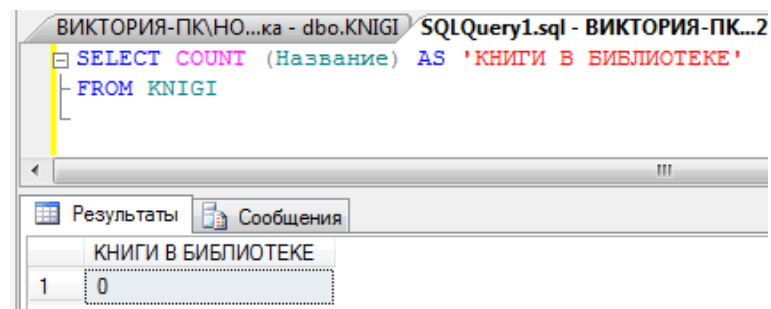


Рис.14 – Результат выполнения запроса

Использование остальных унарных агрегатных функции аналогично COUNT за тем исключением, что для функций MIN и MAX использование ключевых слов DISTINCT и ALL не имеет смысла. С функциями COUNT, MAX и MIN кроме числовых могут использоваться и символьные поля. Если аргумент агрегатной функции не содержит значений, функция COUNT возвращает 0, а все остальные - значение NULL. (Рис. 15)

```
SELECT MAX(ИМЯ СТОЛБЦА)  
FROM [ТАБЛИЦА]  
WHERE ИМЯ СТОЛБЦА<'1.09.2010'
```

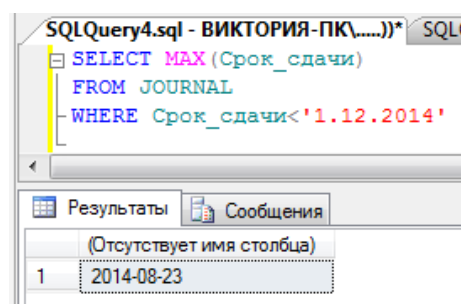


Рис.15 – Результат выполнения запроса

Инструкция INSERT добавляет в таблицу одну или несколько новых строк. В простейшем виде инструкция INSERT имеет следующий вид:

```
INSERT [INTO] table_or_view [(column_list)] data_values
```

Инструкция INSERT выполняет вставку в указанную таблицу или представление значений data_values в виде одной или нескольких строк. Параметр column_list представляет собой разделяемый запятыми список имен столбцов, для которых представляются данные. Если аргумент column_list не задается, данные получают все столбцы таблицы или представления.

Если в аргументе column_list указаны не все столбцы таблицы или представления, то пропущенные столбцы вставляются либо значение по умолчанию (если для столбца оно задано), или значение NULL. Для всех пропущенных столбцов должно быть либо определено значение по умолчанию, либо допускаться значение NULL.

Щелкаем правой кнопкой мыши по нужной таблице, выбираем Создать сценарий для таблицы Используя INSERT Новое окно редактора запросов. (Рис. 16)

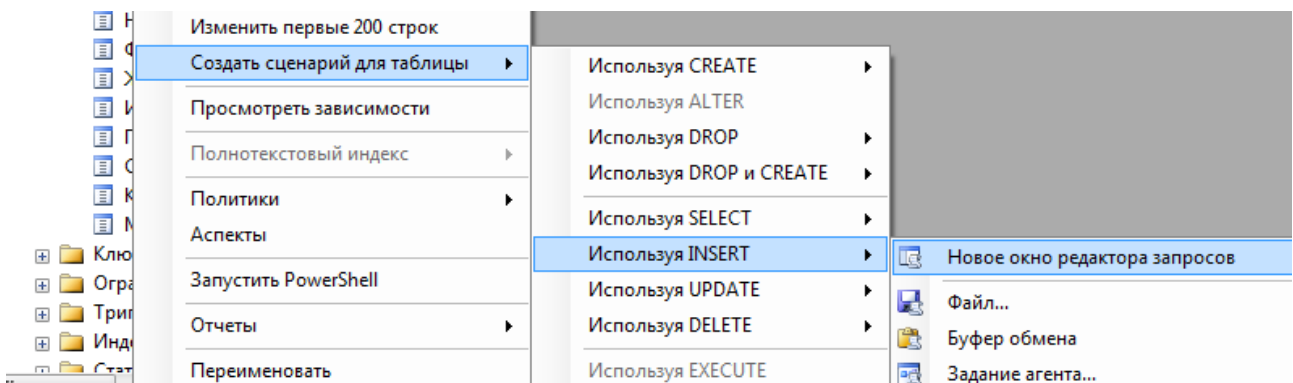
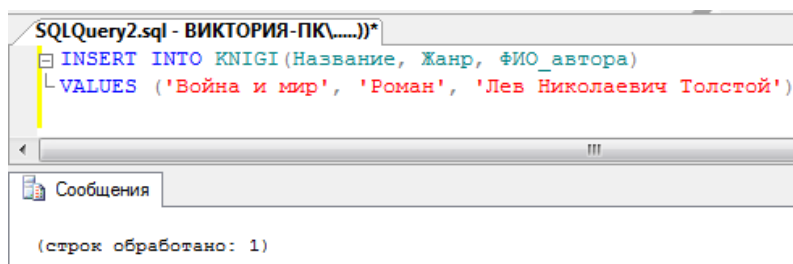


Рис.16 – Вызов окна редактора запросов

```
INSERT INTO table_name (column1, column2, column3)
VALUES ('data1', 'data2', 'data3')
```

Как видно из SQL запроса, сначала перечисляются столбцы с которые будет производиться запись, затем параметром VALUES вводится информация в соответствии со столбцами. Если запись производиться во все столбцы и по порядку, то перечислять их не обязательно, но желательно. (Рис 17, 18)

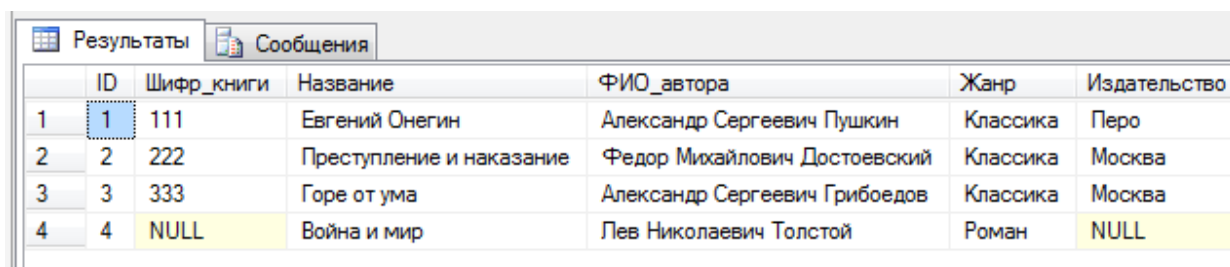


```

SQLQuery2.sql - ВИКТОРИЯ-ПК\.....)*
INSERT INTO KNIGI (Название, Жанр, ФИО_автора)
VALUES ('Война и мир', 'Роман', 'Лев Николаевич Толстой')
    
```

Сообщения
(строк обработано: 1)

Рис.17 – Выполнение запроса



| ID | Шифр_книги | Название | ФИО_автора | Жанр | Издательство |
|----|------------|--------------------------|-------------------------------|----------|--------------|
| 1 | 111 | Евгений Онегин | Александр Сергеевич Пушкин | Классика | Перо |
| 2 | 222 | Преступление и наказание | Федор Михайлович Достоевский | Классика | Москва |
| 3 | 333 | Горе от ума | Александр Сергеевич Грибоедов | Классика | Москва |
| 4 | NULL | Война и мир | Лев Николаевич Толстой | Роман | NULL |

Рис.18 – Результат выполнения запроса

Инструкция UPDATE – SQL запрос обновления существующих данных. Щелкаем правой кнопкой мыши по нужной таблице, выбираем Создать сценарий для таблицы Используя UPDATE Новое окно редактора запросов. (Рис. 19)

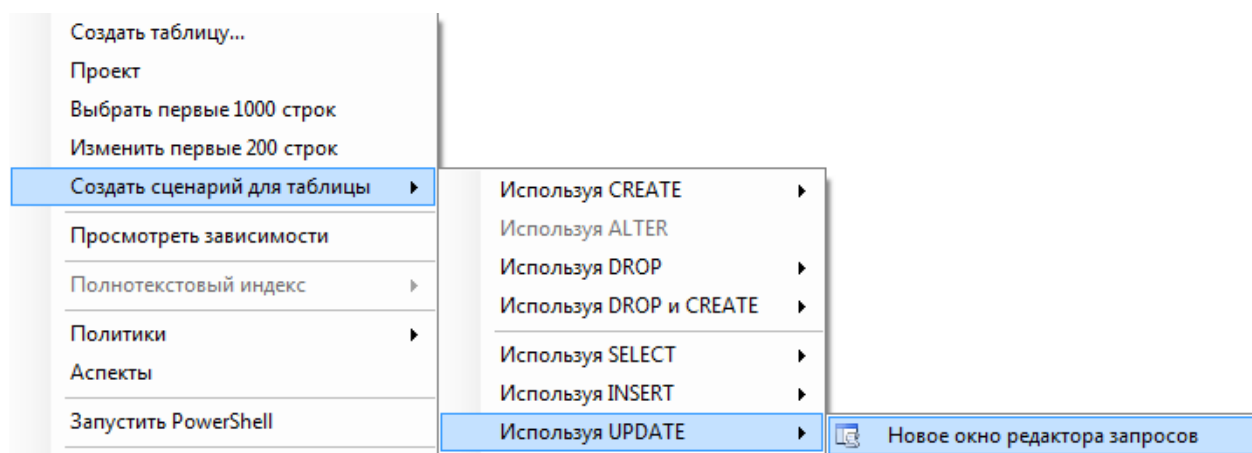


Рис.19 – Вызов окна редактора запросов

Запрос UPDATE используется для изменения существующих данных на новые значения. Чаще всего данный запрос применяется с условием WHERE и выглядит так:

```
UPDATE table_name SET column1 = 'data1', column2 = 'data2'
WHERE column3 = 'data3'
```

Данный запрос обновляет строки в столбцах column1, column2, в которых выполняется условие column3 = 'data3'. (Рис. 20, 21)

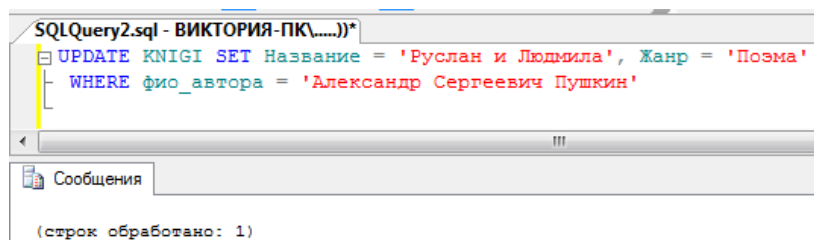


Рис.20 – Выполнение запроса

| ID | Шифр_книги | Название | ФИО_автора | Жанр | Издательство | Год_издания | Страницы | Количество |
|----|------------|--------------------------|-------------------------------|----------|--------------|-------------|----------|------------|
| 1 | 111 | Руслан и Людмила | Александр Сергеевич Пушкин | Поэма | Перо | 2002 | 240 | 2 |
| 2 | 222 | Преступление и наказание | Федор Михайлович Достоевский | Классика | Москва | 2006 | 650 | 4 |
| 3 | 333 | Горе от ума | Александр Сергеевич Грибоедов | Классика | Москва | 2008 | 280 | 3 |
| 4 | NULL | Война и мир | Лев Николаевич Толстой | Роман | NULL | NULL | NULL | NULL |

Рис.21 – Результат выполнения запроса

Инструкция DELETE – SQL запрос удаления записи. Щелкаем правой кнопкой мыши по нужной таблице, выбираем Создать сценарий для таблицы Используя DELETE Новое окно редактора запросов. Запрос DELETE удаляет ненужные вам строки (Рис. 22, 23). Будьте внимательны запрос удаляет все строку целиком:

```
DELETE FROM table_name WHERE column1 = 'data1'
```

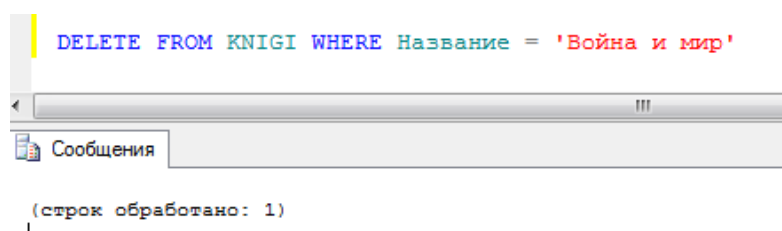
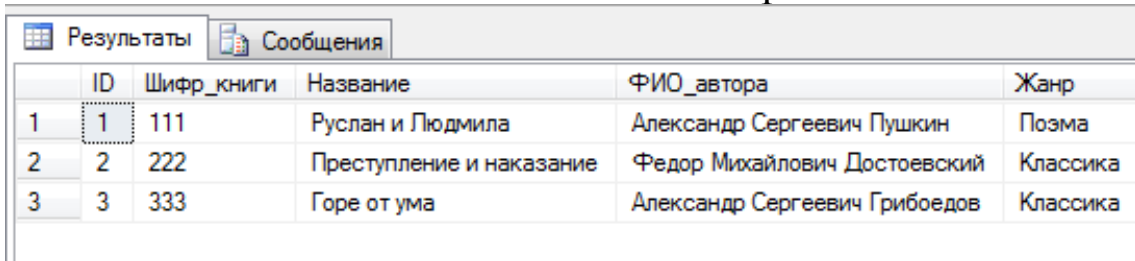


Рис.22 – Выполнение запроса



| | ID | Шифр_книги | Название | ФИО_автора | Жанр |
|---|----|------------|--------------------------|-------------------------------|----------|
| 1 | 1 | 111 | Руслан и Людмила | Александр Сергеевич Пушкин | Поэма |
| 2 | 2 | 222 | Преступление и наказание | Федор Михайлович Достоевский | Классика |
| 3 | 3 | 333 | Горе от ума | Александр Сергеевич Грибоедов | Классика |

Рис.23 – Результат выполнения запроса

Этот SQL запрос удаляет все строки где выполняется условие `column1 = 'data1'`. Помните после выполнения запроса, удаленная строка не подлежит восстановлению.

Оператор `INNER JOIN` возвращает все записи из таблиц `table_01` и `table_02`, связанные посредством `primary/foreign` ключей, и соответствующие условию `WHERE` для таблицы `table_01`. Если в какой-либо из таблиц отсутствует запись, соответствующая соседней, то в выдачу такая пара включена не будет. Иными словами, выдадутся только те записи, которые есть и в первой, и во второй таблице. То есть выборка идет фактически по связи (ключу), выдадутся только те записи, которые связаны между собой. «Одинокие» записи, для которых нет пары в связи, выданы не будут. (Рис. 24)

```
SELECT * FROM table_01  
INNER JOIN table_02  
ON table_01.primary_key = table_02.foreign_key
```

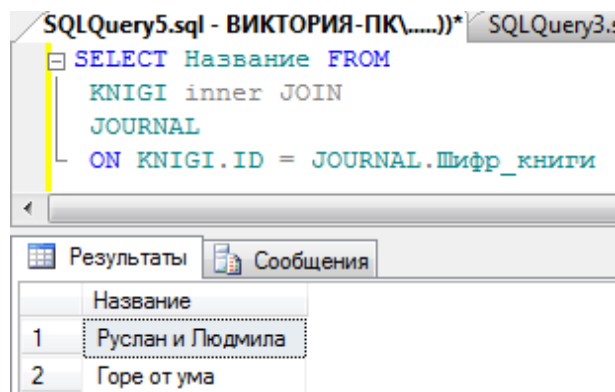


Рис.24 – Результат выполнения запроса

Оператор LEFT JOIN возвращает все данные из «левой» таблицы, даже если не найдено соответствий в «правой» таблице («левая» таблица в SQL-запросе стоит левее знака равно, «правая» — правее, то есть обычная логика правой и левой руки). Иными словами, если мы присоединяем к «левой» таблице «правую», то выберутся все записи в соответствии с условиями WHERE для левой таблицы. Если в «правой» таблице не было соответствий по ключам, они будут возвращены как NULL. Таким образом, здесь главной выступает «левая» таблица, и относительно нее идет выдача. В условии ON «левая» таблица прописывается первой по порядку (table_01), а «правая» – второй (table_02), (Рис. 25, 26):

```
SELECT * FROM table_01
LEFT JOIN table_02
ON table_01.primary_key = table_02.foreign_key
```

| ID | Шифр_книги | ФИО_читателя | Дата_выдачи | Срок_сдачи |
|----|------------|--------------|-------------|------------|
| 1 | 1 | 2 | 2014-03-23 | 2014-08-23 |
| 2 | 3 | 1 | 2014-11-12 | 2014-12-12 |
| 3 | 2 | NULL | NULL | NULL |
| 4 | 1 | NULL | NULL | NULL |
| 5 | 2 | NULL | NULL | NULL |
| 6 | 3 | NULL | NULL | NULL |
| 7 | 1 | NULL | NULL | NULL |

Рис.25 – Таблица Journal

ВИКТОРИЯ-ПК\НО... - dbo.JOURNAL SQLQuery5.sq

```
SELECT Название FROM
KNIGI LEFT JOIN
JOURNAL
ON KNIGI.ID = JOURNAL.Шифр_книги
```

Результаты Сообщения

| Название |
|----------------------------|
| 1 Руслан и Людмила |
| 2 Руслан и Людмила |
| 3 Руслан и Людмила |
| 4 Преступление и наказание |
| 5 Преступление и наказание |
| 6 Горе от ума |
| 7 Горе от ума |

Рис.26 – Результат выполнения запроса

Оператор RIGHT JOIN возвращает все данные из «правой» таблицы, даже если не найдено соответствий в «левой» таблице. То есть примерно также, как и в LEFT JOIN, только NULL вернется для полей «левой» таблицы. Грубо говоря, эта выборка ставит во главу угла правую «таблицу», относительно нее идет выдача. Обратите внимание на WHERE в следующем примере, условие выборки затрагивает «правую» таблицу (Рис. 27):

```
SELECT * FROM table_01  
RIGHT JOIN table_02  
ON table_01.primary_key = table_02.foreign_key  
WHERE table_02.column_01 = 'value'
```

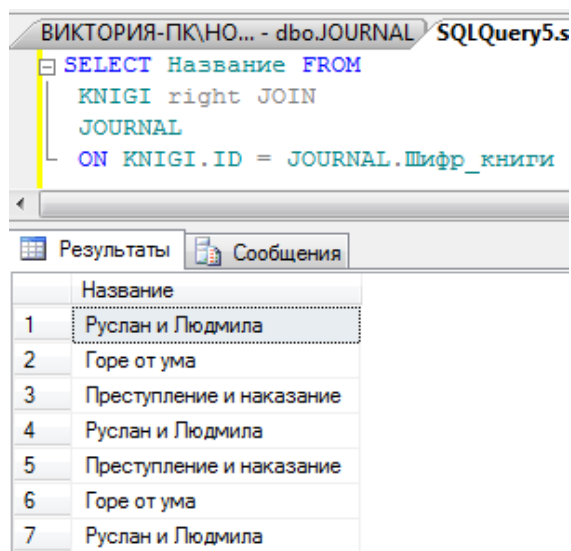


Рис.27 – Результат выполнения запроса

Запросы, отделенные круглыми скобками и входящие в состав конструкций HAVING, WHERE, FROM, SELECT и WITH внешнего запроса SELECT или каких-либо других перечисленных запросов, заключающих в себе эти конструкции, называются вложенными запросами.

Вложенные подзапросы могут быть простыми и связанными. Если результат вложенного запроса не зависит от результата внешнего, то такой вложенный запрос называется простым.

К примеру, данные из таблиц, перечисленных после ключевого слова FROM внешнего и вложенного запроса, находятся независимо, что не требует вводить полный адрес столбцов.

В одном запросе может быть несколько подзапросов, синтаксис у такого запроса следующий (Рис. 28)

```
SELECT имя_столбца FROM имя_таблицы
WHERE часть условия IN
(SELECT имя_столбца FROM имя_таблицы
WHERE часть условия IN
(SELECT имя_столбца FROM имя_таблицы WHERE
условие))
```

Обратите внимание, что подзапросы могут выбирать только один столбец, значения которого они будут возвращать внешнему запросу. Попытка выбрать несколько столбцов приведет к ошибке.

Не рекомендуется создавать запросы со степенью вложения больше трех. Это приводит к увеличению времени выполнения и к сложности восприятия кода.

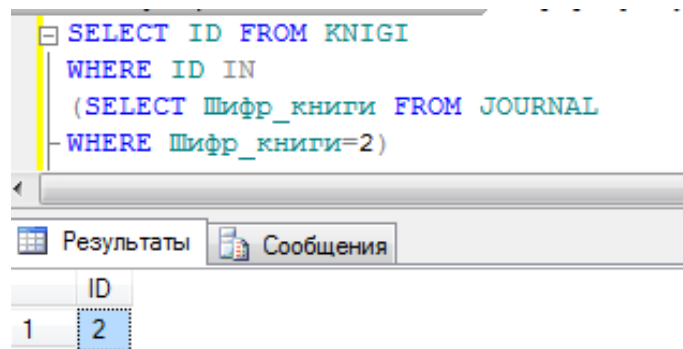


Рис.28 – Результат выполнения запроса

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие таблицы есть в базе данных, которую вы создали? Опишите процесс получения списка таблиц.
2. Опишите последовательность действий, необходимых для выполнения запроса к SQL серверу. Запрос уточните у преподавателя. Опишите что вернул сервер в ответ на ваш запрос.
3. Опишите структуру любой из таблиц в базе данных, к которой вы подключились. Каким образом можно узнать структуру таблицы?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. SQL Like <http://dimonchik.com/sql-like.html>
2. SQL - Урок 4. Выборка данных - оператор SELECT
<http://www.site-do.ru/db/sql4.php>
3. Изучаем inner/left/right join в запросах SQL http://nprk-kaluga.ru/CompareTwoList_MSSQL.htm