

Урок №3

Теория баз данных

Содержание

Введение в SQL.	4
Типы запросов SQL	8
Подготовка платформы	11
Запросы с использованием одной таблицы.....	14
Выборка с использованием оператора WHERE.....	19
Сортировка данных.....	29
Многотабличные запросы на выборку данных.	31
Домашнее задание	37

Введение в SQL.

Что это такое? Категории команд SQL: DDL, DML, DCL

На прошлых парах Вы научились создавать, проектировать и заполнять базу данных, изучили все тонкости и нюансы данного процесса. Но после создания с базой данных, а именно с ее объектами, нужно как-то работать, а именно получать данные из БД в том виде, который необходим в тот или иной момент времени. Для осуществления действий по получению данных используется язык структурированных запросов **SQL**.

SQL (*Structured Query Language*) — это универсальный компьютерный язык, который используется для создания, модификации и управления данными в реляционных базах данных. Стоит заметить, что SQL не является языком программирования. Уникальность данного языка заключается в том, что он является единственным стандартным языком баз данных. Язык SQL поддерживают более сотни СУБД. И самое главное: несмотря на то, что каждая СУБД по-своему интерпретирует стандарт (вносит свои дополнения), набор стандартных инструкций SQL поддерживают все базы данных.

Что касается истории, то начало языка SQL было заложено в **1974 году**, когда компанией IBM для экспериментальной реляционной СУБД **System R** был разработан специальный язык **SEQUEL** (*Structured English QUery Language*,

структурированный английский язык запросов), который позволял очень просто управлять данными в базе данных. Разработкой занимались Дональд Чемберлин и Рей Бойс. Целью разработки SEQUEL было создание языка, которым мог бы пользоваться обычный пользователь, не имеющий опыта программирования.

Позже язык SEQUEL был переименован в SQL и в 1986 г. вышел его первый стандарт **ANSI** (*American National Standards Institute*), а в 1987 г. стандарт **ISO** (*International Organization for Standardization, Международная организация по стандартизации*). Но он не был единственным языком для баз данных. В то же время Калифорнийский университет Беркли разработал собственный язык **QUEL** для некоммерческой реляционной СУБД **Ingres** (*прародитель сегодняшней PostgreSQL*). Но, к сожалению, он не выдержал конкуренции с SQL.

Таким образом, в 1987 г. SQL стал международным языком баз данных (SQL-86), в 1992 г. вышла вторая расширенная версия данного стандарта (ANSI SQL-92 или SQL2), а в 1999 г. третья версия (SQL 1999, SQL-99 или SQL3). Сегодня действует стандарт, принятый в 2003 г. (SQL 2003) с небольшими изменениями, которые были внесены в 2006 и 2008 годах. Кстати, в новый стандарт SQL была добавлена поддержка работы и с XML данными.

Работа с SQL не сложная. В случае, если пользователю необходимо получить данные из базы данных, он обращается с запросом к СУБД за получением этой информации. В результате, СУБД обрабатывает этот запрос и возвра-

щает пользователю результирующие данные. Фактически, запрос — это команда, написанная на языке SQL.

С помощью SQL можно:

- создавать, модифицировать или удалять объекты базы данных и саму базу данных;
- устанавливать связи между объектами базы данных;
- осуществлять выборку данных;
- вставлять, модифицировать и выделять данные из базы данных;
- обеспечивать целостность данных;
- другое.

Вышеописанные действия и еще много других, осуществляются с помощью инструкций (команд) SQL, которые мы будем с Вами изучать в течение следующих уроков.

Все команды языка SQL делятся на четыре категории:

- 1) **DDL** (*Data Definition Language* — Язык Определения Данных) — включает в себя инструкции, которые предназначены для создания, модификации и выделения объектов базы данных (таблицы, представления, индексы и т.д.).
- 2) **DML** (*Data Manipulation Language* — Язык Манипуляции Данными) — это набор инструкций, которые предназначены для работы с данными в базе данных, то есть их вставкой, удалением, изменением и выборкой.
- 3) **DCL** (*Data Control Language* — Язык Управления Данными) — содержит набор инструкций, которые управляют правами доступа пользователей к объектам базы данных.

- 4) **TCL** (*Transaction Control Language* — Язык Управления Транзакциями). В него вносят набор инструкций, позволяющих управлять транзакциями.

А теперь от коротенькой теории перейдем к практике. Знакомство с командами SQL начнем с рассмотрения DML команд, а именно тех, которые отвечают за выборку данных на экран.

Типы запросов SQL

Основой работы с языком SQL является запрос. Запрос — это команда, которая сообщает о необходимости получения указанных данных. Например, можно построить запрос, который будет выводить список всех клиентов магазина, данные о наиболее активных покупателях или сформировать алфавитный перечень товаров, пользующихся наибольшим спросом, или которые были списаны в течение определенного периода. Как правило, эта информация получается из основного источника реляционной базы данных — таблиц, а результат выводится на экран компьютера. Хотя ее можно вывести на принтер, сохранить в файле или другом объекте базы данных и тому подобное.

Существуют следующие основные *типы запросов SQL*:

1. **Запросы на выборку.** Осуществляют выборку данных, которая отвечает указанным критериям запроса, из одной или нескольких таблиц. Результат выполнения данного запроса — набор записей, отображаемых как правило, в режиме таблицы.
2. **Запросы на изменение.** С помощью таких запросов можно осуществлять модификацию данных в базе данных. Существует 4 подтипа запросов на изменение:
 - **Запросы на обновление.** Позволяют обновить данные согласно указанным условиям. Например, установить новые цены на товары определенного

типа, снизив их на 10%, в связи с сезонной распродажей.

- **Запросы на добавление.** Позволяют установить новые данные в таблицу. Новые данные добавляются в конце указанной таблицы.
- **Запросы на удаление** — это запросы, которые в результате своих действий удаляют записи, соответствующие определенному критерию, из указанной таблицы или группы таблиц. Например, могут удалить из базы данных товары, дата поставки которых была более года. Это может быть обусловлено тем, что эти товары предварительно были списаны.
- **Запросы на создание таблицы.** В результате работы данных запросов, осуществляется выборка данных из одной или нескольких таблиц и помещается в новую таблицу определенной структуры. Стоит отметить, что такой тип запросов поддерживается не всеми СУБД и каждая из них осуществляет такой запрос различными средствами.

СУБД MS Access поддерживает еще два типа запросов:

1. **Перекрестные запросы** — запросы, результат работы которых — это организованные в специальный формат данные, сгруппированные по двум критериям. Как правило, такой тип запросов используется для формирования итоговых месячных, квартальных или годовых отчетов по продажам или поставкам товаров.
2. **Запросы с параметрами** — это специальный интерактивный тип запроса, который перед выполнением

выводит диалоговое окно с просьбой ввести один или несколько параметров, необходимых для его работы. Эти параметры фактически позволяют пользователю накладывать условия отбора записей. Например, отбирать товары, которые были поставлены в промежутке определенных дат и т.д. В других СУБД этот вид запросов представлен отдельными объектам базы данных, таких как сохраняемые процедуры и функции.

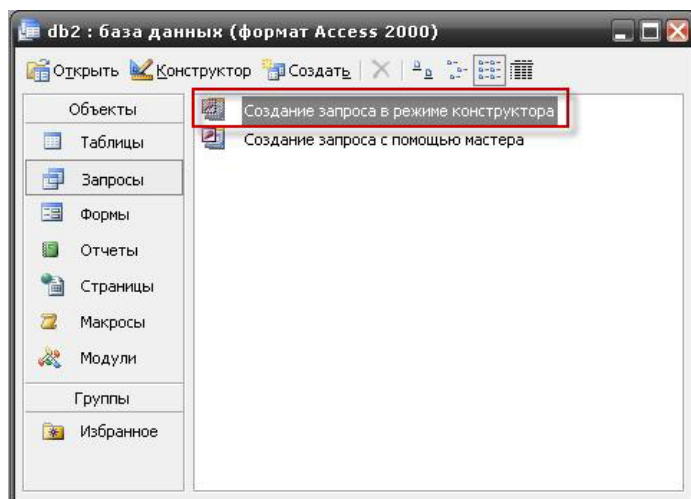
Подготовка платформы

В MS Access существует два *метода создания запросов*:

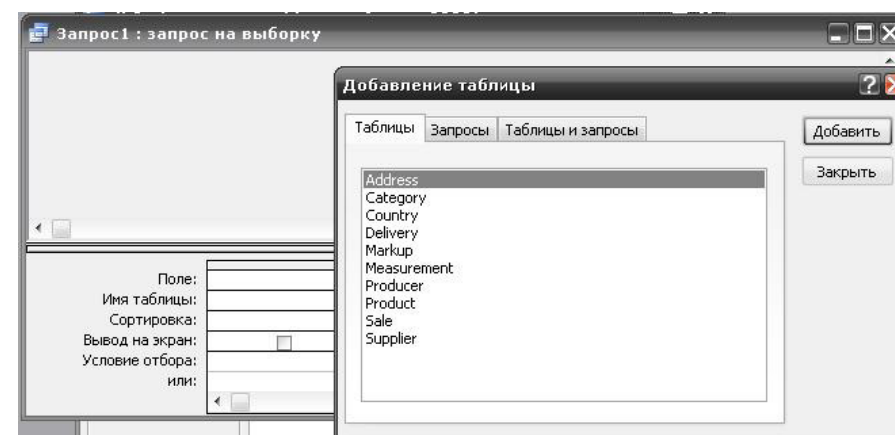
1. С помощью **QBE** (*Query By Example*) — на основе шаблона;
2. С помощью языка структурированных запросов **SQL** (*Structure Query Language*).

Мы будем рассматривать второй вариант создания запросов, то есть с помощью инструкций языка SQL, поскольку этот способ гораздо более гибкий и позволит изучить язык SQL.

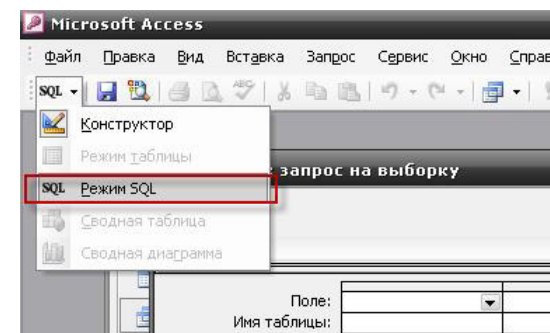
Итак, для того, чтобы написать запрос на языке SQL необходимо выбрать объект «**Запросы**» и создать его в произвольном режиме. Самый удобный и быстрый режим создания — с помощью конструктора, поэтому его и выберем.



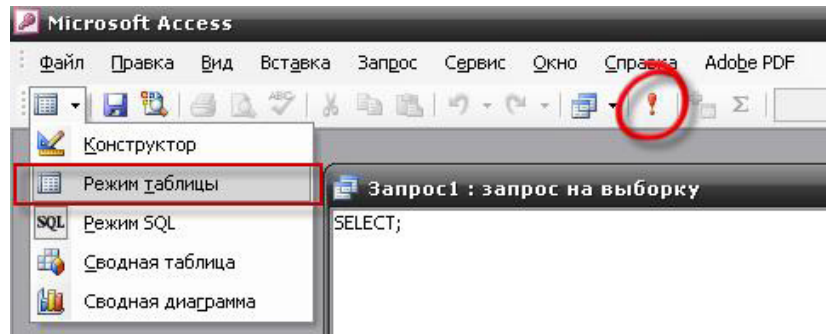
После этого появится окно добавления таблиц, в котором Вам предложат выбрать таблицы для запроса. На основе этих таблиц будет осуществлена выборка данных из базы данных, то есть с них будет вытягиваться необходимая информация. Это нужно только в случае, если Вы планируете создавать запрос в режиме QBE. Поскольку у нас цель другая, мы отказываемся от предложенного (кнопка «Заккрыть»).



Следующий шаг — это *переход в режим SQL*. Для этого на панели инструментов из выпадающего списка «Вид» выбираем нужный нам режим.



Это позволит открыть окно с редактором для SQL запросов. Когда запрос написан, чтобы увидеть результат его работы, то есть запустить его на выполнение, достаточно или перейти в режим таблицы, или воспользоваться кнопкой на панели инструментов «Запуск».



Запросы с использованием одной таблицы.

Оператор SELECT

Все запросы на получение любого количества данных из одной или нескольких таблиц выполняются с помощью оператора **SELECT**, который имеет следующий синтаксис:

```
SELECT [ALL | DISTINCT] { * | поле_для_выборки
                                [ , поле_для_выборки] }
FROM { таблица | представление } [as] [псевдоним]
     [ , {таблица | представление} [as] [псевдоним] ]
[WHERE условие]
[GROUP BY [ALL] выражение_группирования]
[HAVING условие_на_группу]
[ORDER BY имя_поля | номер_поля [ASC | DESC]];
```

Расшифруем:

- **SELECT** — задает перечень полей, откуда необходимо выбрать данные. Оператор ALL определен стандартом, но поддерживается не всеми СУБД;
- **FROM** — из каких таблиц или таблицы нужно осуществить выборку (где содержатся указанные для выборки поля);
- **WHERE** — условие на выборку;
- **GROUP BY** — задает перечень полей, по которому необходимо группировать выборку, чтобы получить

для каждой группы единое агрегированное значение. Для этого в операторе SELECT нужно ОБЯЗАТЕЛЬНО использовать SQL-функции агрегирования;

- **HAVING** — накладывает условие, которое позволяет получить в результате только те группы, которые удовлетворяют указанному критерию;
- **ORDER BY** — позволяет отсортировать результирующую совокупность по указанным полям.

Точка с запятой является обязательной в конце каждого SELECT запроса, но некоторые СУБД позволяют ее упускать. Также стоит отметить, что допускается вложенность операторов SELECT. Но делать ее необходимо с умом и только в случае необходимости.

Для того, чтобы лучше разобраться с работой данного оператора, попробуем попрактиковаться и создать для начала *простые выборки данных*:

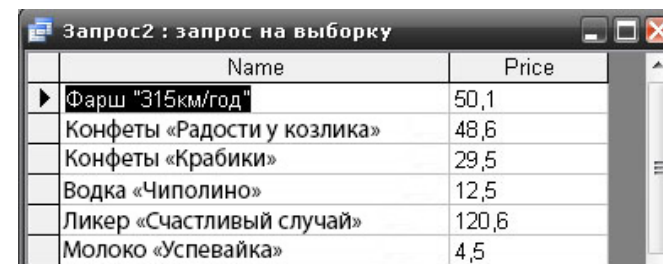
1. Создать запрос на выборку всей информации о товарах:

```
SELECT *
FROM Product;
```

2. Если мы хотим получить только название и цену товара, то нам нужно возле оператора SELECT написать перечень полей для выборки:

```
SELECT Name, Price
FROM Product;
```

Результат:

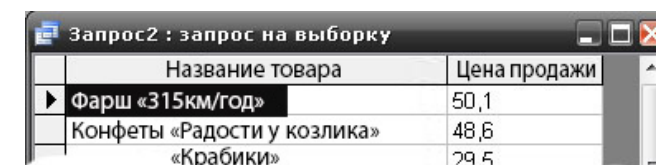


Name	Price
Фарш "315км/год"	50,1
Конфеты «Радости у козлика»	48,6
Конфеты «Крабики»	29,5
Водка «Чиполино»	12,5
Ликер «Счастливый случай»	120,6
Молоко «Успевайка»	4,5

При этом можно одновременно отформатировать вывод информации на экран, то есть задать названия полей (псевдонимы полей) в результирующем запросе. Для этого запрос нужно переписать следующим образом:

```
SELECT Name as [Название товара],
       Price as [Цена продажи]
FROM Product;
```

Результат:



Название товара	Цена продажи
Фарш «315км/год»	50,1
Конфеты «Радости у козлика»	48,6
«Крабики»	29,5

Чтобы сделать результат запроса еще более наглядным, используют литералы. *Литерал* — это строка, которая записывается в двойных или одинарных кавычках и включается в параметр <поле_для_выборки>. Он выводится в результирующем запросе как отдельное поле. В результирующей совокупности данных литерал выступает в роли текста, объясняет значение следующего за ним поля.

Синтаксис добавления литерала следующий:

```
SELECT 'літерал' [, 'літерал']
```

С использованием литералов наш запрос примет вид:

```
SELECT Name as [Назва товару], Price as [Ціна продажу],
       'грн.' as [Грошова одиниця]
FROM Product;
```

Результат:

Название товара	Цена продажи	Денежная ед.
Фарш "315км/год"	50,1	грн.
Конфеты «Радости у козлика»	48,6	грн.
Конфеты «Крабики»	29,5	грн.
Водка «Чиполино»	12,5	грн.
Пикер "Шоколад" "падок"	12,6	грн.

3. Также можно включать в выборку результаты расчетов. Например, добавим к нашей выборке цену продажи товаров, увеличенную в два раза.

```
SELECT Name as [Наименование товара]
       Price as [Цена продажи],
       'грн.' as [Денежная единица],
       Price * 2 as [Двойная цена продажи]
FROM Product;
```

4. Для того, чтобы избежать дублирования, нужно дополнить запрос ключевым словом **DISTINCT** (разный).

```
SELECT DISTINCT Name as [Наименование товара]
FROM Product;
```

Результат:

Название товара
Моршинская 0,5л
Хлеб чорний
Бананы
Бананы

Название товара
Апельсины "Накопоти"
Бананы
Водка «Чиполино»
Картошка «Зеленое чудо»
Колбаса «Роспазнай»

5. К средствам форматирования результирующей выборки можно отнести операцию конкатенации — **&**. С помощью оператора конкатенации мы можем вывести объединенные данные нескольких полей в одном. Например, необходимо вывести информацию про наценки в одном поле, то есть название наценки и ее размер:

```
SELECT Markup.Name & "-" & Markup.Percent & '%' AS [Наценка]
FROM Markup;
```

Результат:

Наценка
Базовая - 15 %
Праздничная - 7 %
Супер-наценка - 1 %
Оптовая - 5 %
Розничная - 7 %

В основном, такое форматирование используется для вывода полного имени лиц или адреса. Чтобы лучше продемонстрировать использование оператора конкатенации, напомним еще один пример. Допустим, в нашей базе данных существует таблица, которая хранит информацию о работниках магазина и необходимо вывести о них информацию. Имя и фамилия должно храниться в отдельных полях, но для лучшего визуального восприятия их уместно вывести в отдельном поле. Такой запрос получит следующий вид:

```
SELECT FName & ' ' & LName as [Имя работника],
       Address as [Адрес]
FROM Employee;
```


Выборка с использованием оператора WHERE

Простые выборки данных редкое явление. Как правило, отбирается набор данных, которые удовлетворяют определенному критерию, условию или их набору. Для этого в операторе SELECT используется конструкция WHERE, после которой указывается собственно само условие отбора в виде выражения. В теле условия могут использоваться:

1. **Операторы**, которые позволяют выполнять набор действий над одним или несколькими компонентами выражения.
 - **Арифметические:** сложение (+), вычитание (-), умножение (*), деление (/), целочисленное деление одного операнда на второй (\), оператор деления по модулю (Mod), возведение в степень (^). В качестве операндов могут быть как числа, так и значения полей.
 - **Сравнение:** >, <, >=, <=, =, <>. Если один из операндов имеет значение NULL, то результатом сравнения также будет NULL.
 - **Логические (булевские)**, результатом работы которых является логическое значение True (-1), False (0) или NULL. Их используют для комбинирования результатов выполнения двух и более операций.

Это: логическое И (And), включающее ИЛИ (Or), логическое отрицание (Not) и исключающее ИЛИ (Xor).

- **Конкатенация** — для объединения нескольких строк (&). О нем мы уже говорили выше.
 - Операторы SQL: BETWEEN..AND .., IN, LIKE, IS NULL (проверка на равенство нулю).
2. **Литералы** — это значение в явном их представлении.
 - **Числовые**, которые могут содержать знак разделения (в десятичных числах) и знак минус (-) для отрицательных значений, символы е или Е. Например: 3,4567E-01 12000, 25.
 - **Текстовые** (строчные) — любые печатные символы (А-Я, 0-9, знаки пунктуации и т.д.). Их следует писать в одинарных или двойных кавычках. Например: "Мелодрама", "Киев", "Гостиница" Украина "".
 - **Дата и время.** В большинстве СУБД дата и время пишутся в одинарных кавычках, но это может быть и другой произвольный символ. В MS Access этим символом является символ хэш (#). Но не забывайте, что это требование только для выражений SQL и при вводе значения даты или времени через интерфейс данный символ не указывается. Например: # 01.03.99 #, # 15-января-2001 #.
 3. **Функции** позволяют упростить процесс установки условия для выборки данных. В MS Access можно использовать как встроенные функции, так и пользовательские (написанные на VBA). Приведем краткий перечень часто используемых встроенных функций:

■ *Даты и времени:*

Date () — возвращает текущую дату;

DateAdd ("d", -15, [ДатаПоставки]) — возвращает дату, которая на 15 дней предшествует дате, заданной значением поля «ДатаПоставки»;

DateDiff ("d" [ДатаПоставки], [ДатаПродажи]) — возвращает значение, является разницей значений полей «ДатаПоставки» и «ДатаПродажи»;

Year (# 12.06.01 #) — возвращает год с указанной даты, то есть 2001.

■ *Строки:*

Format (Date, # dd-mm-yyuu #) — возвращает отформатированную дату;

InStr ("Город", "C") — возвращает число, указывающее позицию первого вхождения одной строки в другую, то есть 3;

LCase ("ГОРОД") — переводит строку в нижний регистр;

Left ([Город], 2) — отражает два первых символа значения поля «Город»;

Right ([Город], 3) — отражает три последних символа значения поля «Город»;

Trim ([Название]) — возвращает значение поля «Название» без пропусков (space).

■ *Приведение типа данных:*

Val ("12.35") — конвертирует текст в число, то есть возвращает 12,35;

Str (12,35) — конвертирует число в текст — "12,35".

Приведем несколько примеров создания выборки с условиями:

I. Запрос на выборку информации о товаре с названием «Бананы», в котором кроме названия товара нужно указать его категорию, производителя и цену:

```
SELECT Name, Id Category, Id Producer, Price
FROM Product
WHERE Name = 'Бананы';
```

Результат:



Name	IdCategory	IdProducer	Price
Бананы	Фрукты	ЗАТ "Яблоко"	8,5
Бананы	Фрукты	Banana Republica	9

Следует отметить, что на самом деле, вместо категории и производителя СУБД выводит их соответствующий идентификатор, но при работе с MS Access эти данные скрываются. Для вывода значения по идентификатору используются многотабличные запросы, которые будут рассмотрены позже (п.7. Многотабличные запросы на выборку. Декартово произведение множеств).

6. Создать запрос на выборку информации о товарах, цены которых находятся в пределах от 10 до 50 грн.

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Price >= 10 AND Price <= 50;
```

Результат:

Запрос4 : запрос на выборку

Name	IdCategory	IdProducer	Price
Фарш "Байкер"	Мясные	ЗАТ "ДПС"	42,00
Фарш "315км/год"	Мясные	ВАТ «Сало Украины»	50,00
Конфеты «Радости у козлика»	Кондитерские	ВАТ «Конфеты»	49,00
Конфеты «Крабики»	Кондитерские	ВАТ «Молдавские сладости»	30,00
Водка «Чиполино»	Ликеро-водочные	АТ "Русская водка"	12,00

Но есть и более простой способ написания такого рода запроса, когда критерием отбора служит диапазон значений. Заключается он в использовании оператора SQL BETWEEN..AND ... С использованием данного оператора запрос переписывается следующим образом:

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Price BETWEEN 10 AND 50;
```

Результат будет аналогичен, за исключением того, что последняя включенная цена будет 49, то есть 50 не включается.

В случае, если необходимо осуществить полностью противоположную операцию, то есть вывести товары, цены которых не входят в указанный промежуток, следует использовать оператор логического отрицания NOT:

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Price NOT BETWEEN 10 AND 50;
```

7. Оператор BETWEEN..AND .. можно использовать для проверки попадания в промежуток дат, букв и тому подобное.

ПРИМЕЧАНИЕ! Не забывайте, что дата записывается в формате # 00/00/0000 #. Для получения текущей даты используется функция Date ().

Рассмотрим пример: создать запрос на выборку информации о поставке товаров в промежутке от 01/06/2009 до текущей даты.

```
SELECT IdProduct, DateDelivery
FROM Delivery
WHERE DateDelivery BETWEEN #01/06/2009# AND Date();
```

Результат:

Запрос4 : запрос на выборку

IdProduct	DateDelivery
Фарш "315км/год"	01/05/2009
Фарш «315км/год»	25/06/2009
Водка «Чиполино»	07/07/2008
Молоко «Успевайка»	25/06/2009
Водка «Чиполино»	01/07/2009
Бананы	03/02/2009
Сухарики «Дубовые дровишки»	05/06/2009
Колбаса «Распознай»	17/08/2008

Запрос4 : запрос на выборку

IdProduct	DateDelivery
Фарш "315км/год"	25.06.2009
Молоко «Успевайка»	25.06.2009
Водка «Чиполино»	01.07.2009
Сухарики «Дубовые дровишки»	05.06.2009

8. Создадим запрос на выборку информации о фильмах, цена которых 10 и 50 грн.

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Price=10 OR Price=50;
```

Результат:

Запрос4 : запрос на выборку

Name	IdCategory	IdProducer	Price
Фарш "315км/год"	Мясные	ВАТ «Сальце Украины»	50,00
Апельсины «Наколотые»	Фрукты	ЗАТ «Яблоко»	10,00

Неплохо, но и этот запрос можно сократить, используя оператор IN, который позволяет проверить

принадлежность данных поля множеству определенных значений.

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Price IN(10, 50);
```

9. Для поиска значений по шаблону используется оператор LIKE. Шаблон выражения может включать в себя:

- * или % — в данной позиции может присутствовать 0 или более символов;
- ? или _ — в данной позиции обязательно присутствует один произвольный символ;
- # — в данной позиции присутствует одна цифра;
- [a-z] — в данной позиции обязательно присутствует один символ из указанного диапазона;
- [abc] — в данной позиции обязательно присутствует один символ из указанного диапазона значений;
- [! a-z] — в данной позиции обязательно присутствует один символ, не входящий в указанный диапазон;
- [! abc] — в данной позиции обязательно присутствует один символ, не входящий в указанный диапазон значений.

Итак, создадим запрос на выборку информации о товарах, в названии которых не менее двух букв "о":

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Name LIKE '*o*o*';
```

Результат:

Name	IdCategory	IdProducer	Price
Конфетка «Радости у козлика»	Кондитерские	BAT «Конфеты»	49,00
Водка «Чиполино»	Ликеро-водочные	АТ «Русская водка»	12,00
Молоко «Успевайка»	Молочные	ПП «Коровка»	4,00
Сушари «Дубовые дровишки»	Кондитерские	ПП «Молдавские сладости»	5,00
Апельсины «Наколотые»	Фрукты	ЗАТ «Яблоко»	10,00
Картошка «Зеленое Чудо»	Овощи	ЗАТ «Картошка»	6,00
	Колбасы	ВАТ «Сальце Украины»	4,100

10. Используя маску, напишем запрос на выборку информации о товарах, названия которых начинаются с букв, которые лежат в промежутке от А до К:

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Name BETWEEN "A%" and "K%";
```

Результат:

Name	IdCategory	IdProducer	Price
Бананы	Фрукты	ЗАТ «Яблоко»	8,00
Бананы	Фрукты	Banana Republica	9,00
Водка «Чиполино»	Ликеро-водочные	АТ «Русская водка»	12,00
Апельсины «Наколотые»	Фрукты	ЗАТ «Яблоко»	10,00

Но результирующая выборка не будет выводить на экран товары, которые начинаются с буквы «К», поскольку оператор BETWEEN..AND .. не включает конечное указанное значение и об этом мы уже сегодня говорили. Если необходимо вывести товары, которые начинают с буквы А по К включительно, тогда условие следует перестроить:

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Name BETWEEN "A%" and "Л%";
```


11. В SQL существует понятие NULL поля. Значение NULL не равносильно пропуску (space) или нулю в данных числового типа. Поле данных имеет значение NULL, если ему не было присвоено значение, то есть оно пустое. Фактически NULL является недействительным значением. Найти записи, содержащие NULL значение можно с помощью ключевых слов IS NULL или IS NOT NULL (в зависимости от сути выборки).

Создадим запрос на выборку информации о товарах, имеющих цену, то есть значение цены которых не равно NULL:

```
SELECT Name, IdCategory, IdProducer, Price
FROM Product
WHERE Price IS NOT NULL;
```

Результатом будут записи о товарах, для которых установлена цена.

12. Очень часто возникает необходимость указать несколько критериев отбора. В таком случае их можно совместить с помощью логического оператора AND или OR (выбор зависит от необходимого результата). Например, напишем запрос на выборку данных о товарах, цена которых более 40 грн. и имеющееся количество находится в диапазоне от 50 до 100.

```
SELECT Name, Price, Quantity
FROM Product
WHERE Price >=40 AND Quantity BETWEEN 50 AND 100;
```

Результат:

Name	Price	Quantity
Фарш "Байкер"	42,00	50
Колбаса «Роспознай»	54,00	50

Если вместо оператора AND указать оператор OR, тогда на экран будут выведены данные, которые удовлетворяют или одному, или второму условию, или же их сочетанию.

Name	Price	Quantity
Фарш «Байкер»	42,00	50
Фарш «315км/год»	50,00	45
Конфеты «Радости у козлика»	49,00	23
Конфеты «Крабики»	30,00	56
Водка «Чиполино»	12,00	56

Сортировка данных

Для сортировки результирующего набора в операторе SELECT используется ключевое слово ORDER BY с дополнительным параметром ASC (по умолчанию) — сортировка по возрастанию, или DESC — сортировка по убыванию.

1. Отобразить информацию о товарах, отсортированных в возрастающем порядке по их названиям:

```
SELECT Name, Price, Quantity
FROM Product
WHERE Price >=40 AND Quantity BETWEEN 50 AND 100
ORDER BY Name ASC;
```

Результатом будет следующая отсортированная совокупность:

Name	Price	Quantity
Колбаса «Распознай»	54,00	50
Фарш «Байкер»	42,00	50

2. Кроме имен полей в списке при ORDER BY можно использовать их порядковые номера в списке SELECT.

```
SELECT Name, Price, Quantity
FROM Product
WHERE Price >=40 AND Quantity BETWEEN 50 AND 100
ORDER BY 1;
```

Результат будет аналогичен предыдущему.

3. Сортировать можно и по нескольким полям.

```
SELECT Name, Price, Quantity
FROM Product
WHERE Price <= 20 OR Quantity BETWEEN 50 AND 100
ORDER BY 1, 2 DESC;
```

В таком случае выборка будет отсортирована по названию продукта в возрастающем порядке, а затем по цене в порядке убывания.

Name	Price	Quantity
Апельсины «Наколотые»	10,00	10
Бананы	9,00	10
Бананы	8,00	10
Водка «Чиполино»	12,00	56
Картошка «Зеленое Чудо»	6,00	100
Колбаса «Распознай»	54,00	50
Молоко «Успевайка»	4,00	75

Многотабличные запросы на выборку данных.

Декартово произведение множеств

В предыдущих главах мы осуществляли выборку данных о товарах и получали информацию только из одной таблицы. Если нам необходимо было вывести информацию о производителе или поставщике продукции, то мы указывали поле внешнего ключа и данные были выведены на экран. Построим еще раз такой запрос: выведем информацию о товаре и его производителе.

```
SELECT Name as [Наименование товара],
       IdProducer as [Производитель]
FROM Product;
```

Результат:

Запрос5 : запрос на выборку		
	Название товара	Производитель
▶	Фарш "315км/год"	БАТ «Сальце Украины»
	Конфеты «Радости у козлика»	БАТ «Конфеты»
	Конфеты «Крабики»	ПП «Молдавские сладости»
	Водка «Чиполино»	АТ «Русская водка»

На первый взгляд, никаких проблем, но на самом деле, вместо производителя СУБД выводит их соответствующий идентификатор, то есть значение внешних ключей. MS Access эти данные скрывает, но при работе

с СУБД результат Вас не удовлетворит. Кроме того, если Вам необходимо будет наложить условие на выборку по производителю, то у Вас могут возникнуть трудности. Например, выведем информацию о товарах производителя ОАО «Росинка»:

```
SELECT Name as [Наименование товара],
       IdProducer as [Производитель]
FROM Product
WHERE IdProducer='БАТ "Росинка";
```

После запуска на выполнение такого запроса, сгенерируется ошибка о несоответствии типов, поскольку поле «IdProducer» является числовым, а условие — символьного типа.

Выходов из такой ситуации два:

1. Получить информацию о том, какое значение первичного ключа соответствует производителю ОАО «Росинка» и переписать запрос:

```
SELECT Name as [Наименование товара],
       IdProducer as [Производитель]
FROM Product
WHERE IdProducer=34;
```

2. Построить запрос и сделать его многотабличным

Конечно, первый способ не очень удобен, ведь мы не можем всегда искать соответствующие первичные ключи в связанных таблицах и подставлять их значения. Итак, рассмотрим второй способ более подробно.

Для того, чтобы создать многотабличный запрос, необходимо в списке таблиц при конструкции FROM перечислить список необходимых таблиц. После этого в списке SELECT (или в местах, где идет обращение к полям таблиц) необходимо указать, какие именно поля и из каких таблиц необходимо получить. Доступ к полям таблиц организуется с помощью операторов идентификации точка (.) и восклицательный знак (!) следующим образом:

КлассОбъекта!ИмяОбъекта.Свойство_или_Метод

Например, если необходимо получить доступ к полю *Name* таблицы *Student*, то нужно написать следующее выражение:

Student.Name или: Tables!Student.Name

Если это понятно, тогда перейдем к практике:

1. Создать запрос на выборку данных о товарах, с указанием их категории и производителя:

```
SELECT Product.Name as [Наименование товара],
       Category.Name as [Категория],
       Producer.Name as [Производитель]
FROM Product, Category, Producer;
```

Результат:

Название товара	Категория	Производитель
Моршинська 0,5л	Бакалея	БАТ «Росинка»
Хлеб черный	Хлебо-булочные	ЗАТ «Ровно-Хлеб»
Бананы	Фрукты	ЗАТ «Яблоко»
Бананы	Фрукты	Banana Republica
Фарш «Байкер»	Мясные	ЗАТ «ДПС»
Фарш «315км/год»	Мясные	БАТ «Сальце Украины»

Как видите, в результате работы запроса образовалось количество записей, больше существующих. В таком случае говорят, что образовалось «**декартово произведение множеств**», то есть все возможные комбинации записей. Чтобы такого не было, нужно прописать используемые таблицы связанные между собой в теле оператора WHERE.

```
SELECT Product.Name as [Наименование товара],
       Category.Name as [Категория],
       Producer.Name as [Производитель]
FROM Product, Category, Producer
WHERE Product.IdCategory=Category.IdCategory
      AND Product.IdProducer=Producer.IdProducer;
```

Результат:

Название товара	Категория	Производитель
Моршинська 0,5л	Бакалея	БАТ «Росинка»
Хлеб черный	Хлебо-булочные	ЗАТ «Ровно-Хлеб»
Бананы	Фрукты	ЗАТ «Яблоко»
Бананы	Фрукты	Banana Republica
Фарш «Байкер»	Мясные	ЗАТ «ДПС»
Фарш «315км/год»	Мясные	БАТ «Сальце Украины»

2. Теперь дополним наш запрос условием: отразить только те товары, категория которых «Фрукты»:

```
SELECT Product.Name as [Наименование товара],
       Category.Name as [Категория],
       Producer.Name as [Производитель]
FROM Product, Category, Producer
WHERE Product.IdCategory=Category.IdCategory
      AND Product.IdProducer=Producer.IdProducer
      AND Category.Name='Фрукты';
```

Результат:

Название товара	Категория	Производитель
Бананы	Фрукты	ЗАТ «Яблоко»
Бананы	Фрукты	Banana Republica
Апельсины «Наколотые»	Фрукты	ЗАТ «Яблоко»
Яблоко «Червячок»	Фрукты	ЗАТ «Яблоко»

Запись: 1 из 4

3. Отсортируем данную выборку по названиям производителей в возрастающем порядке:

```
SELECT Product.Name as [Наименование товара],
       Category.Name as [Категория],
       Producer.Name as [Производитель]
FROM Product, Category, Producer
WHERE Product.IdCategory=Category.IdCategory
      AND Product.IdProducer=Producer.IdProducer
      AND Category.Name='Фрукты'
ORDER BY 3;
```

Хотя названия таблиц перед именами полей достаточно хорошо информируют пользователя о процессе выборки, постоянно переписывать их (иногда достаточно громоздкие названия) довольно трудно. Для облегчения жизни, но не понимания, в SQL существует понятие *алиасов* (*Alias*) или *псевдонимов* имен таблиц. Псевдонимы используются в основном для визуального упрощения текста запросов. Например, перепишем предыдущий запрос с использованием псевдонимов таблиц:

```
SELECT pr.Name as [Наименование товара],
       c.Name as [Категория],
       Producer.Name as [Производитель]
```

```
FROM Product as pr, Category as c, Producer
WHERE pr.IdCategory=c.IdCategory
      AND pr.IdProducer=Producer.IdProducer
      AND c.Name='Фрукты'
ORDER BY 3;
```

Кстати, при создании псевдонима допускается упустить ключевое слово as:

```
SELECT pr.Name as [Наименование товара],
       c.Name as [Категория],
       Producer.Name as [Производитель]
FROM Product pr, Category c, Producer
WHERE pr.IdCategory=c.IdCategory
      AND pr.IdProducer=Producer.IdProducer
      AND c.Name='Фрукты'
ORDER BY 3;
```

Домашнее задание

1. Создать запрос на выборку всей информации о поставках товаров в магазин.
2. Вывести названия товаров, имеющееся количество которых не указано.
3. Создать запрос на выборку информации о поставщиках магазина, а именно их названия и полный адрес. Адрес вывести в одном поле (отформатировать вывод).
4. Вывести информацию о том, каких именно товаров было продано более 10 в промежутке от 01/12/2008 до 01/03/2009.
5. Вывести названия товаров, которые продавались в течение последнего месяца (без повторений). **Примечание!** Для получения информации о текущем месяце следует воспользоваться необходимыми функциями.
6. Вывести список товаров, который сопровождается названиями производителей, категория которых "Бакалея".
7. Создать запрос на выборку информации о средней стоимости продажи на указанную в условии дату.
8. Вывести список всех поставщиков, с указанием стран их происхождения, названия которых содержат букву К, а город расположения начинается с буквы М.
9. Вывести все товары, поставляемые поставщиками ЧП Петров и ЧП Иванов.
10. Вывести информацию обо всех товарах, названия которых начинаются с В по Л включительно.

11. Выбрать все товары с указанием поставщика, название производителя которых почкается с К или М и категория не "Соки и воды".
12. Выбрать все товары с указанием имени и страны их производителя. Условие: страна производителя "Украина", "Россия" или "Польша", цена поставки чтобы была <50 грн., А даты поставки чтобы были в пределах 01/01/2006 и до сегодняшнего дня.
13. Найти товары, жкатегория которых "Соки и воды", количество продаж которых более 100. Вывести такие товары, их производителей, поставщиков и категорию.
14. Создать многотабличный запрос на выборку информации о поставке товаров в следующем виде: название поставленных товаров, их поставщиков, категории, даты поставки и общую стоимость их поставки. **Условие:** только трех выбранных поставщиков. Отсортировать выборку по названию товара в алфавитном порядке.
15. Создать многотабличный запрос на выборку информации о продаже товаров в следующем виде: название проданных товаров, их производителей, категорий, даты продажи, стоимости продажи, с указанием полного адреса производителей. **Условие:** выведенная информация не должна касаться двух определенных производителей (например, кроме ЧП Иванова и ЧП Смирнова). Предусмотреть вывод полного адреса производителей в одном поле. Отсортировать выборку по названиям товаров в возрастающем порядке и по их стоимости в убывающем порядке.