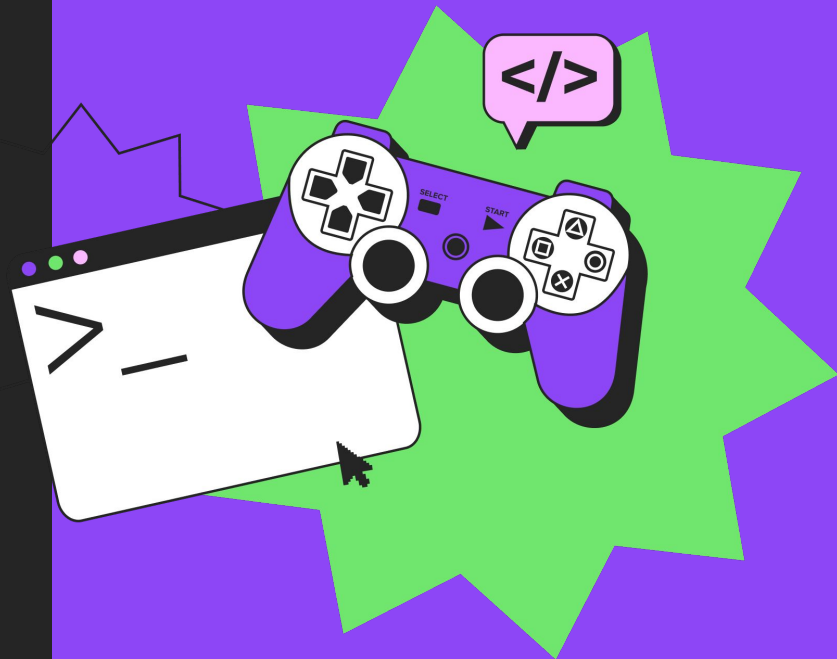


# Базы данных и SQL

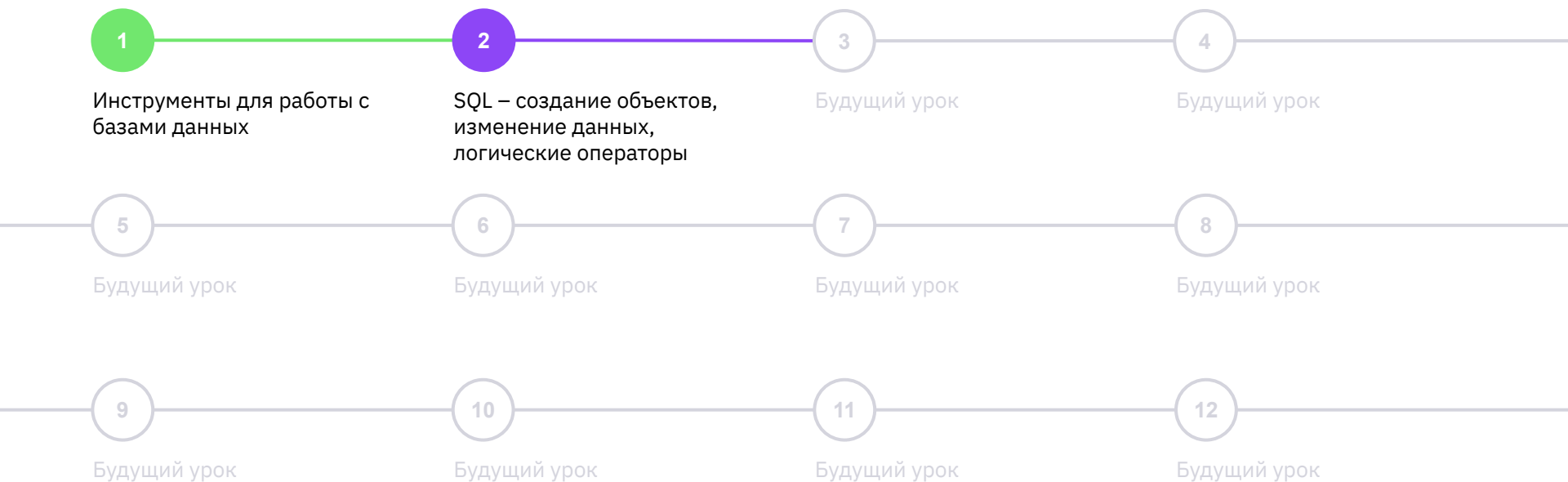
## Урок 2

SQL – создание объектов, изменение данных, логические операторы





## План курса (горизонтальный)












# Содержание урока



## Что будет на уроке сегодня

-  Типы данных, значения NULL, create table, PK, FK, index
-  Комментарии
-  Арифметических операции
-  Логические операторы (and, or, between, not, in)
-  Приоритет выполнения операторов, порядок выполнения запроса
-  Оператор CASE, IF
-  Запросы изменения данных (insert, update, delete)



## Создание базы данных

```
1 CREATE DATABASE MySampleDB;
```



## Просмотр созданных баз данных

1 `show databases;`

```
MySQL 8.0 Command Line Client
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.




mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysampledb |
| mysql |
| performance_schema |
| sakila |
| sys |
| users |
| world |
+-----+
8 rows in set (0.00 sec)
```



## Просмотр созданных баз данных

5 • `show databases;`

---

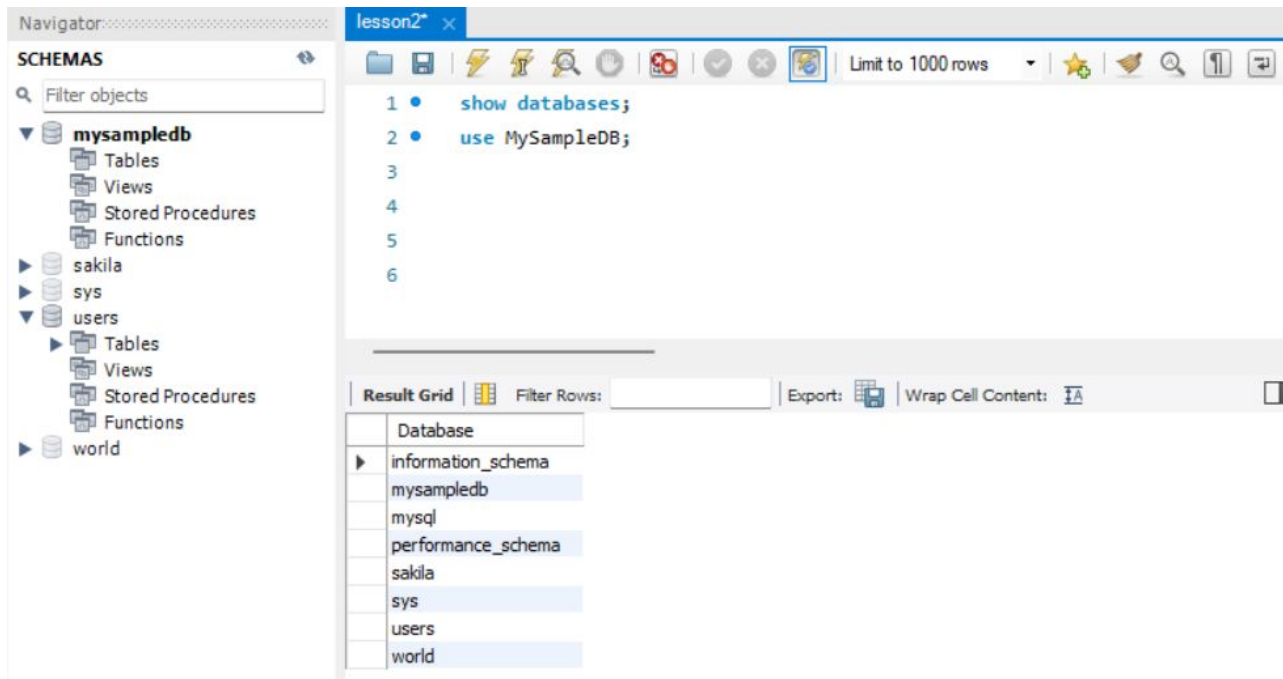
Result Grid |  Filter Rows:  | Export:  | Wrap Cell Content: 

	Database
▶	information_schema
	mysampledб
	mysql
	performance_schema
	sakila
	sys
	users
	world



## Подключение к конкретной БД

Подключенная БД выделяется черным цветом





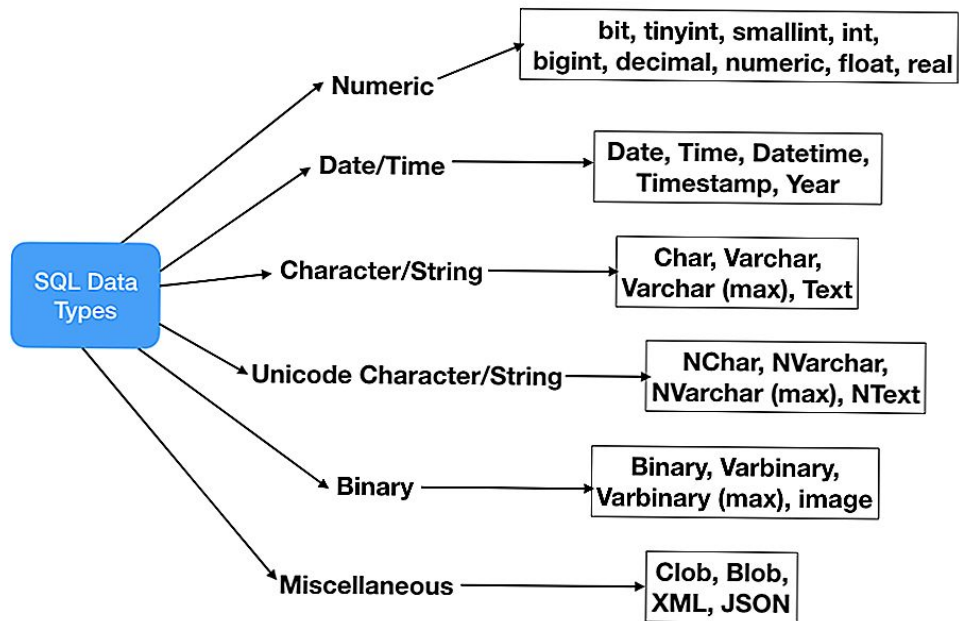


## Создание сущности в MySQL

```
1 CREATE TABLE table_name
2 (
3     column_name_1 column_type_1,
4     column_name_2 column_type_2,
5     ... ,
6     column_name_N column_type_N
7 );
8 -- Комментарий
9 -- table_name — имя таблицы;
10 -- column_name — имя столбца;
11 -- column_type — тип данных столбца.
```



## Типы данных в MySQL





## Основные типы данных: числовые



### **INT**

целочисленные значения от  $-2147483648$  до  $2147483647$ , 4 байта.



### **DECIMAL**

хранит числа с заданной точностью.



### **BOOL**

0 или 1. Однозначный ответ на однозначный вопрос — false или true.



## Основные типы данных: символьные



### **VARCHAR(N)**

N определяет максимально возможную длину строки.



### **TEXT**

подходит для хранения большого объема текста до 65 KB, например, целой статьи.



## Основные типы данных: дата и время



### **DATE**

только дата. Диапазон от 1000-01-01 по 9999-12-31.



### **TIME**

только время — часы, минуты, секунды — «hh:mm:ss».  
Память хранения — 3 байта.



### **DATETIME**

соединяет оба предыдущих типа — дату и время.  
Использует 8 байтов памяти.



### **TIMESTAMP**

хранит дату и время начиная с 1970 года.



## Основные типы данных: бинарные



### **BLOB**

до 65 КБ бинарных данных



### **LARGEBLOB**

до 4 ГБ.



## Первичный ключ - primary key (pk)

```
1 CREATE TABLE table_name
2 (
3     column1 column_definition,
4     column2 column_definition,
5     CONSTRAINT [constraint_name]
6     PRIMARY KEY [ USING BTREE | HASH ] (column_1, column_2, ...
7     column_n)
8 );
```



## Внешний ключ - foreign key (fk)

```
1 [CONSTRAINT имя_ограничения]
2 FOREIGN KEY (столбец1, столбец2, ... столбецN)
3 REFERENCES главная_таблица (столбец_главной_таблицы1,
4   столбец_главной_таблицы2, ... столбец_главной_таблицыN)
5 [ON DELETE действие]
6 [ON UPDATE действие]
```





## Создание таблиц и связей между ними

```
1 CREATE TABLE Customers
2 (
3     Id INT PRIMARY KEY AUTO_INCREMENT,
4     Age INT,
5     FirstName VARCHAR(20) NOT NULL,
6     LastName VARCHAR(20) NOT NULL,
7     Phone VARCHAR(20) NOT NULL UNIQUE
8 );
9 CREATE TABLE Orders
10 (
11     Id INT PRIMARY KEY AUTO_INCREMENT,
12     CustomerId INT,
13     CreatedAt Date,
14     FOREIGN KEY (CustomerId) REFERENCES Customers (Id)
15 );
```



## Комментарии

```
1  -- Это комментарий
2  # Это тоже комментарий
3  /*
4  Многосторочный
5  комментарий
6  */
```



## Арифметические операции: сложение

```
1 SELECT 3+5;
```

## Арифметические операции: вычитание

```
1 SELECT 3-5;
```



## Арифметические операции: умножение

```
1 SELECT 3*5;
```

## Арифметические операции: умножение

```
1 SELECT 18014398509481984*18014398509481984;  
2 #произведение умножения целых чисел выходит за границы 64-  
   #битового диапазона для вычислений с точностью BIGINT
```



## Арифметические операции: деление

```
1 SELECT 3/5;
```

## Арифметические операции: деление

```
1 SELECT 102/0;
```

```
2 # Деление на ноль дает NULL
```



## Логические операторы



### AND

операция логического И. Она объединяет два выражения. Синтаксис: **выражение1 AND выражение2**.



### OR

операция логического ИЛИ. Она также объединяет два выражения: **выражение1 OR выражение2**.



### NOT

операция логического отрицания. Если выражение в этой операции ложно, то общее условие истинно: **NOT выражение**.



## Таблицы, для которой будут применяться логические операторы






```
CREATE TABLE Products
(
    Id INT AUTO_INCREMENT PRIMARY KEY,
    ProductName VARCHAR(30) NOT NULL,
    Manufacturer VARCHAR(20) NOT NULL,
    ProductCount INT DEFAULT 0,
    Price DECIMAL
);
```



## Таблицы, для которой будут применяться логические операторы

```
1 • USE productsdb;
2
3 • SELECT * FROM Products;
```

<

Result Grid   Filter Rows:  Edit:   

	Id	ProductName	Manufacturer	ProductCount	Price
	1	iPhone X	Apple	3	76000
	2	iPhone 8	Apple	2	51000
	3	Galaxy S9	Samsung	2	56000
	4	Galaxy S8	Samsung	1	41000
	5	P20 Pro	Huawei	5	36000
	NULL	NULL	NULL	NULL	NULL





## Логические операторы: “AND”

```
1 • USE productsdb;
2
3 • SELECT * FROM Products
4   WHERE Manufacturer = 'Samsung' AND Price > 50000
```

<

Result Grid Filter Rows:  Edit: Export/

	Id	ProductName	Manufacturer	ProductCount	Price
	3	Galaxy S9	Samsung	2	56000
	NULL	NULL	NULL	NULL	NULL



## Логические операторы: “OR”

```
1 • USE productsdb;
2 |
3 • SELECT * FROM Products
4 WHERE Manufacturer = 'Samsung' OR Price > 50000
```

<

Result Grid Filter Rows:  Edit: Export/

	Id	ProductName	Manufacturer	ProductCount	Price
	1	iPhone X	Apple	3	76000
	2	iPhone 8	Apple	2	51000
	3	Galaxy S9	Samsung	2	56000
	4	Galaxy S8	Samsung	1	41000
	NULL	NULL	NULL	NULL	NULL



## Логические операторы: “NOT”

```
1 • USE productsdb;
2
3 • SELECT * FROM Products
4   WHERE NOT Manufacturer = 'Samsung';
```

<

Result Grid Filter Rows:  Edit:

	Id	ProductName	Manufacturer	ProductCount	Price
	1	iPhone X	Apple	3	76000
	2	iPhone 8	Apple	2	51000
	5	P20 Pro	Huawei	5	36000
	NULL	NULL	NULL	NULL	NULL



## Приоритет операций

```
1 SELECT * FROM Products
2 WHERE Manufacturer = 'Samsung' OR NOT Price > 30000 AND ProductCount > 2;
```



## Переопределить приоритет операций

```
1 SELECT * FROM Products
2 WHERE Manufacturer = 'Samsung' OR NOT (Price > 30000 AND ProductCount > 2)
```



## Оператор CASE

```
1 CASE
2     WHEN условие_1 THEN результат_1
3     WHEN условие_2 THEN результат_2
4     .....
5     WHEN условие_N THEN условие_N
6     [ELSE альтернативный_результат]
7 END
```



## Оператор CASE: пример

```
1 USE productsdb;
2
3 SELECT ProductName, ProductCount,
4 CASE
5     WHEN ProductCount = 1
6     THEN 'Товар заканчивается'
7     WHEN ProductCount = 2
8     THEN 'Мало товара'
9     WHEN ProductCount = 3
10    THEN 'Есть в наличии'
11    ELSE 'Много товара'
12 END AS Category
13 FROM Products;
```

< Result Grid Filter Rows:  Export:

	ProductName	ProductCount	Category
	iPhone X	2	Мало товара
	iPhone 8	2	Мало товара
	iPhone 7	5	Много товара
	Galaxy S9	2	Мало товара
	Galaxy S8	1	Товар заканчивается
	Honor 10	2	Мало товара
	Nokia 8	6	Много товара



## Оператор IF

```
1 IF(условие, значение_1, значение_2)
2 #Если условие, передаваемое в качестве первого параметра, верно,
  #то возвращается первое значение, иначе возвращается второе
  #значение
```





## Запросы изменения данных



### **INSERT**

вставка новых данных в таблицу



### **UPDATE**

обновление данных из таблицы



### **DELETE**

удаление данных



## Запросы изменения данных: INSERT

Данный оператор имеет 2 основные формы:

**1. INSERT INTO таблица(перечень\_полей)**

**VALUES(перечень\_значений)** – вставка в таблицу новой строки значения полей которой формируются из перечисленных значений

**2. INSERT INTO таблица(перечень\_полей) SELECT**

**перечень\_значений FROM ...** – вставка в таблицу новых строк, значения которых формируются из значений строк возвращенных запросом.



## Запросы изменения данных: INSERT

```
1 CREATE TABLE Products
2 (
3     Id INT AUTO_INCREMENT PRIMARY KEY,
4     ProductName VARCHAR(30) NOT NULL,
5     Manufacturer VARCHAR(20) NOT NULL,
6     ProductCount INT DEFAULT 0,
7     Price DECIMAL
8 );
9 INSERT INTO Products (ProductName, Manufacturer, ProductCount,
10    Price)
11 VALUES
12 ('iPhone X', 'Apple', 3, 76000),
13 ('iPhone 8', 'Apple', 2, 51000),
14 ('Galaxy S9', 'Samsung', 2, 56000),
15 ('Galaxy S8', 'Samsung', 1, 41000),
16 ('P20 Pro', 'Huawei', 5, 36000);
```



## Запросы изменения данных: UPDATE

```
1 UPDATE имя_таблицы
2 SET столбец1 = значение1, столбец2 = значение2, ... столбецN =
  значениеN
3 [WHERE условие_обновления]
```

Например, увеличим у всех товаров цену на 3000:

```
1 UPDATE Products
2 SET Price = Price + 3000;
```



## Запросы изменения данных: UPDATE

```
mysql> UPDATE Products SET Price = Price + 3000;  
Query OK, 5 rows affected (0.01 sec)  
Rows matched: 5  Changed: 5  Warnings: 0
```

```
mysql> SELECT * FROM Products;
```

Id	ProductName	Manufacturer	ProductCount	Price
1	iPhone X	Apple	3	79000
2	iPhone 8	Apple	2	54000
3	Galaxy S9	Samsung	2	59000
4	Galaxy S8	Samsung	1	44000
5	P20 Pro	Huawei	5	39000

5 rows in set (0.00 sec)



## Запросы изменения данных: DELETE

```
1 DELETE FROM имя_таблицы  
2 [WHERE условие_удаления]
```

Например, удалим строки, у которых производитель - Huawei:

```
1 DELETE FROM Products  
2 WHERE Manufacturer='Huawei';
```



## Запросы изменения данных: DELETE

```
1 • USE productsdb;
2
3 • DELETE FROM Products
4   WHERE Manufacturer='Huawei';
5
6 • SELECT * FROM Products;
```









<

Result Grid | Filter Rows:  | Edit:

	Id	ProductName	Manufacturer	ProductCount	Price
	1	iPhone X	Apple	3	79000
	2	iPhone 8	Apple	2	54000
	3	Galaxy S9	Samsung	5	59000
	4	Galaxy S8	Samsung	4	44000
	NULL	NULL	NULL	NULL	NULL



## Итоги занятия:

-  Изучили команды для создания таблицы данных.
-  Узнали, как создавать первичный и внешние ключи
-  Научились ставить комментарии
-  Разобрались с арифметическими операциями и порядком выполнения
-  Узнали, как выполняется запрос
-  Поработали с операторами CASE, IF
-  Разобрались с запросами изменения данных (insert, update, delete)
-  Подготовились к дальнейшей работе по курсу.





**Спасибо за внимание!**