

# Основы SQL. Практическое применение.

**Джавадов Бахтияр**

Аналитик

Финансовый департамент ВТБ

# Структура

1. SELECT запросы
2. Добавление записей (INSERT INTO)
3. Обновление записей (UPDATE)
4. Удаление записей (DELETE)
5. Создание таблиц (CREATE TABLE)
6. Изменение таблиц (ALTER TABLE)
7. Удаление таблиц (DROP TABLE)

# 1. SELECT запросы

Важно соблюдать порядок прописывания запроса:

```
SELECT <column_name1>, aggregate_function(<column_name2>), ...  
FROM <table_name1>  
INNER JOIN / LEFT OUTER JOIN <table_name2>  
ON <table_name1.id> = <table_name2.id>  
WHERE <conditions>  
GROUP BY <column_name1>, ...  
HAVING <conditions>  
ORDER BY <column_name1> ASC/DESC, ...  
LIMIT <number_of_records_in> OFFSET <number_of_records_not_in>;
```

# 1. SELECT запросы

1. С помощью запроса вывести наименование трека, наименование альбома, наименование музыкальной группы, выпустивший этот альбом. Группы: AC/DC, Black Sabbath, Aerosmith, Led Zeppelin.
2. С помощью запроса вывести наименование музыкальной группы, у которых несколько альбомов. Отсортировать по убыванию количества альбомов.
3. С помощью запроса вывести топ 10 наименований музыкальной группы, их альбомов. Указать длительность альбомов. Отсортировать по убыванию по длительности.
4. С помощью запроса вывести формат аудиофайла, указать количество треков в данном формате, средний размер в Мб.

## 2. Добавление записей

```
INSERT INTO <table_name>  
VALUES (<value11>, <value12>, ... )  
      (<value21>, <value22>, ... )  
      ... ;
```

```
INSERT INTO <table_name> (<column_name1>, <column_name2>, ... )  
VALUES (<value11>, <value12>, ... )  
      (<value21>, <value22>, ... )  
      ... ;
```

Значения, которые не указаны в INSERT, принимаются по умолчанию.

## 2. Добавление записей

Примеры:

```
INSERT INTO artist (name)
VALUES
('Bud Powell');
```

```
INSERT INTO artists (name)
VALUES
('Buddy Rich'),
('Candido'),
('Charlie Byrd');
```

```
INSERT INTO artists
DEFAULT VALUES;
```

# 3. Обновление записей

```
UPDATE <table_name>  
SET <column_name1> = <value1>,  
<column_name1> = <value2>,  
...  
WHERE <conditions>;
```

```
UPDATE <table_name>  
SET <column_name1> = <expression1>,  
<column_name1> = < expression2>,  
...  
WHERE <conditions>;
```

# 3. Обновление записей

## Примеры:

```
UPDATE employee  
SET lastname = 'Smith'  
WHERE employeeid = 3;
```

```
UPDATE employee  
SET city = 'Toronto',  
state = 'ON',  
postalcode = 'M5P 2N7'  
WHERE employeeid = 4;
```

```
UPDATE employee  
SET email = LOWER( firstname || '.' || lastname || '@msu.ef.com' )  
WHERE email IS NULL;
```



# 4. Удаление записей

```
DELETE FROM <table_name>  
WHERE <conditions>;
```

При прописывании таких запросов лучше заранее убедиться, что удаляется те записи, которые действительно больше не нужны.

# 4. Удаление записей

Примеры:

```
DELETE FROM artist  
WHERE name = 'Bud Powell';
```

```
DELETE FROM artist  
WHERE name IN ('Buddy Rich', 'Candido', 'Charlie Byrd')  
OR name IS NULL;
```

# 5. Создание таблиц

```
CREATE TABLE <table_name>
(
  <column_name1> <data_type1> <column_type1> DEFAULT <default_value1>,
  <column_name1> <data_type1> <column_type2> DEFAULT <default_value2>,
  ...
);
```

```
CREATE TABLE <table_name> AS

SELECT <column_name1>, ...
FROM <table_name1>
WHERE <condition>;
```

При создании таблицы через уже существующую таблицу тип данных принимает значение существующей таблицы.

# 5. Создание таблиц

## Типы данных и колонок для создания таблиц

| Тип данных          | Определение / условие  |
|---------------------|--|
| INTEGER             | Тип целого числового значения  |
| FLOAT, DECIMAL(n,k) | Тип дробного числового значения (n – количество чисел до запятой, k – после запятой) |
| TEXT, VARCHAR(n)    | Тип текстового значения (n – количество символов)                                    |
| DATE, DATETIME      | Тип значения даты, даты-времени  |

| Тип колонки | Определение / условие   |
|-------------|---|
| PRIMARY KEY | Колонка уникальна. Каждое значение в колонке используется для идентификации строки в таблице. |
| NOT NULL    | Колонка не может иметь значения NULL.   |

# 5. Создание таблиц

## Примеры:

```
CREATE TABLE table_name  
(  
  column_1 INTEGER PRIMARY KEY,  
  column_2 VARCHAR(10) NOT NULL,  
  column_3 INTEGER DEFAULT 0  
);
```

```
CREATE TABLE movies  
(  
  id INTEGER PRIMARY KEY,  
  title TEXT,  
  director TEXT,  
  year INTEGER,  
  length_minutes INTEGER  
);
```

# 6. Изменение таблиц

## Добавление колонок

```
ALTER TABLE <table_name>  
ADD <column_name> <column_type> DEFAULT <default_value>;
```

Новая колонка НЕ может быть **PRIMARY KEY**, так как в таблице он уже есть. Если есть условие **NOT NULL**, тогда нужно указать значение по умолчанию, отличный от **NULL** (DEFAULT <default\_value>)

## Удаление колонок

```
ALTER TABLE <table_name>  
DROP <column_name>;
```

# 6. Изменение таблиц

## Примеры:

```
CREATE TABLE devices  
(  
  name TEXT NOT NULL,  
  model TEXT NOT NULL,  
  serial INTEGER NOT NULL  
);
```

```
INSERT INTO devices (name, model, serial)  
VALUES  
( 'HP ZBook 17 G3 Mobile Workstation', 'ZBook', 'SN-2015');
```

```
ALTER TABLE devices  
ADD COLUMN location TEXT;
```

```
ALTER TABLE devices  
DROP COLUMN location;
```

# 7. Удаление таблиц

```
DROP TABLE <table_name>;
```

При прописывании таких запросов лучше заранее убедиться (несколько раз), что удаляется именно та таблица, которая больше не нужна.

Примеры:

```
DROP TABLE devices;
```

```
DROP TABLE foo;
```



**СПАСИБО ЗА ВНИМАНИЕ!**