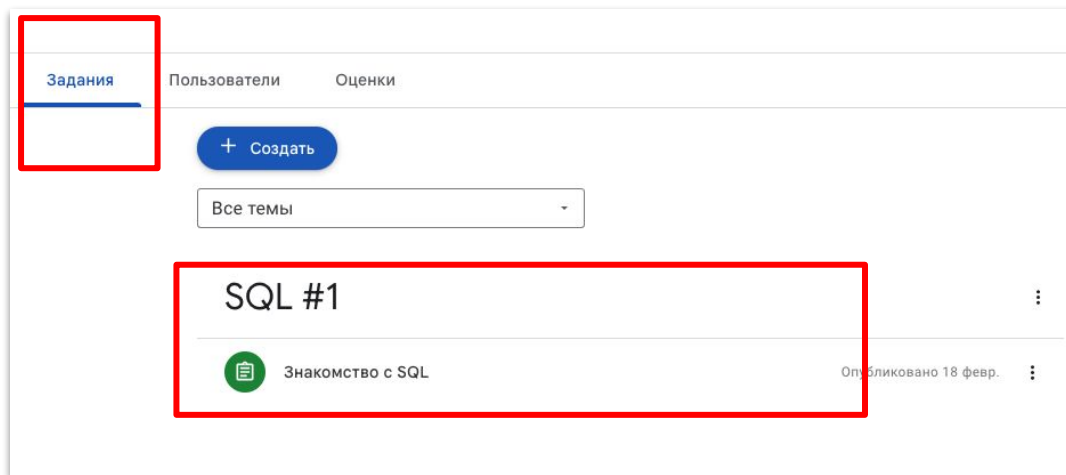


Практика

<https://classroom.google.com/c/NzUxMjQ5ODQxNzAx?cjc=hje7u7q>





Есть ошибки?

ML Basic

ОСНОВЫ SQL

Начнем в 20:01

otus.ru

Общая площадь Арктики и Африки?

```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```

continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country

• REC

Меня хорошо
видно &&
слышно?



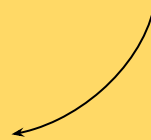
Ставим “—”, если есть проблемы

Общая площадь Арктики и Африки?

```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```

continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country



Тема вебинара

ML Basic

Оконные функции, оптимизация запросов

Катя

Екатерина
Дмитриева

- Telegram: @dmi3eva



Общая площадь Арктики и Африки?

```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```



Разбор

Какой оператор
выполняется **последним**?

Датасет Spider

<https://yale-lily.github.io/spider>

Общая площадь **Арктики** и **Африки**?

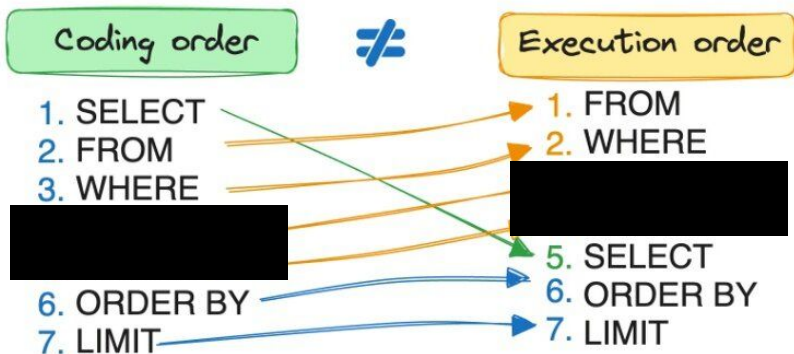
```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```

continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country

Разбор

Какой оператор
выполняется **последним**?



Общая площадь **Арктики** и **Африки**?

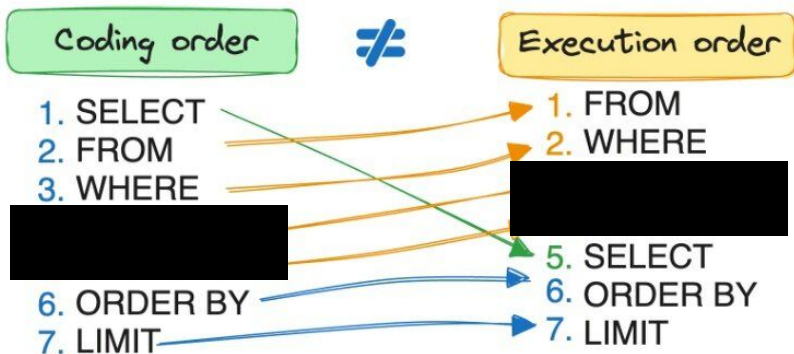
```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```

continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country

Разбор

Какой оператор
выполняется **последним**?



Общая площадь **Арктики** и **Африки**?

```
SELECT  
    SUM(area)
```

```
FROM
```

```
    country
```

```
WHERE
```

```
    continent = "Africa"
```

```
    AND continent = "Arctic"
```

continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country

Разбор

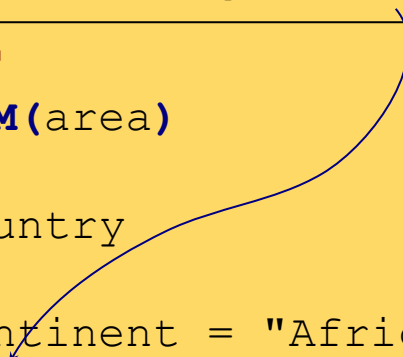
Dataset: Spider

<https://yale-lily.github.io/spider>

Задача: Text-to-SQL

Общая площадь Арктики и Африки?

```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```



continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country



Разбор

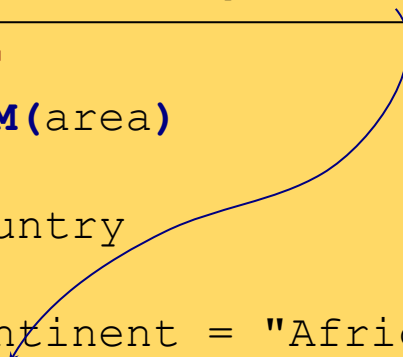
Dataset: Spider

<https://yale-lily.github.io/spider>

Задача: Text-to-SQL

Общая площадь Арктики и Африки?

```
SELECT
    SUM(area)
FROM
    country
WHERE
    continent = "Africa"
    AND continent = "Arctic"
```



continent	area
Arctic	13
Australia	8
Africa	30
Eurasia	54

country



Маршрут вебинара

Еще один SQL-viewer: вспомнить все

Пример DDL-запросов

Работа с датами

Подключение к SQLite БД средствами Python

Новые SQL-конструкции: JOIN, GROUP BY, HAVING, EXISTS

Задача с собеседования

Обобщение

[!] Домашняя практика

- **DDL** = data definition language
Настройка структуры БД
- **DML** = data manipulation language
Управление данными – добавлять, обновлять, удалять
- **DCL** = data control language
Выдача или отзыв прав доступа

Как этим пользоваться?

В чем разница?

- СУБД
- SQL
- Microsoft Office Access

В чем разница?

- **СУБД**

- PostgreSQL
- SQLite ...

Сложное ПО для эффективной работы с потенциально большим объемом данных

- **SQL: ...**

Язык взаимодействия с СУБД

- **DB browser**

- Microsoft Office Access
- SQLite browser
- DBeaver

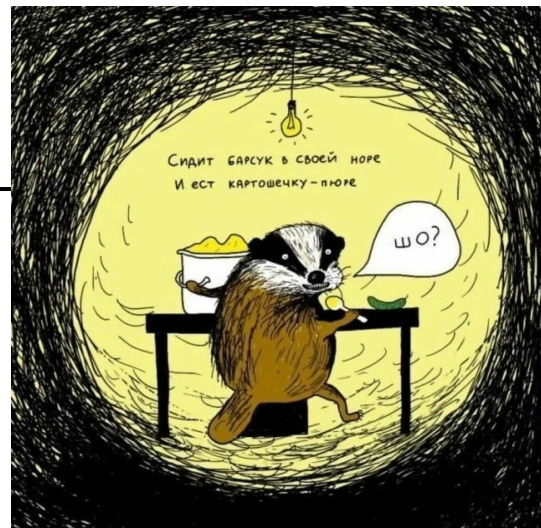
Графический интерфейс для взаимодействия с СУБД

Компоненты

- **Драйвер базы данных** — это компонент системы, который используется для подключения внешних баз данных и дальнейшей работы с ними.

DBeaver

- **Community Edition** и **Enterprise Edition** версии
- Работает с СУБД: SQLite, MySQL, PostgreSQL...
- Поддерживает **основной набор функций**:
Создание БД, SQL-запросы, трансфер данных, ...
- А также:
 - Построение графиков
 - Триггеры
 - Интеграция с Eclipse, IntelliJ IDEA, Visual Studio Code



Будет 3 или 5 строк?

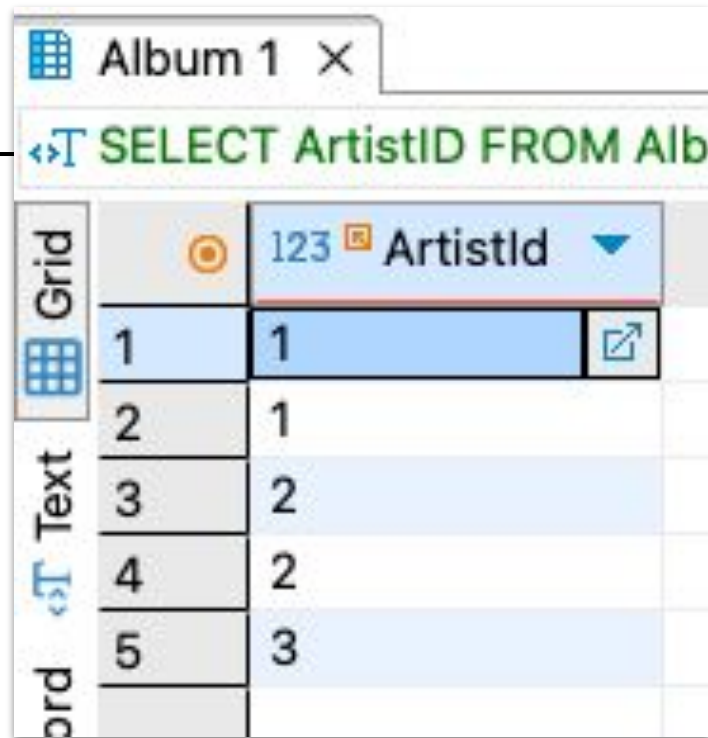
SELECT

DISTINCT (ArtistID)

FROM

Album

LIMIT 5



The screenshot shows a database query editor window titled "Album 1". The query entered is `SELECT ArtistID FROM Alb`. Below the query, there is a table with 5 rows and 2 columns. The first column is labeled "ord" and the second column is labeled "ArtistId". The first row is highlighted in blue and contains the value "1" in the "ord" column and "1" in the "ArtistId" column. The second row contains "2" and "1". The third row contains "3" and "2". The fourth row contains "4" and "2". The fifth row contains "5" and "3".

ord	ArtistId
1	1
2	1
3	2
4	2
5	3

Будет 3 или 5 строк?

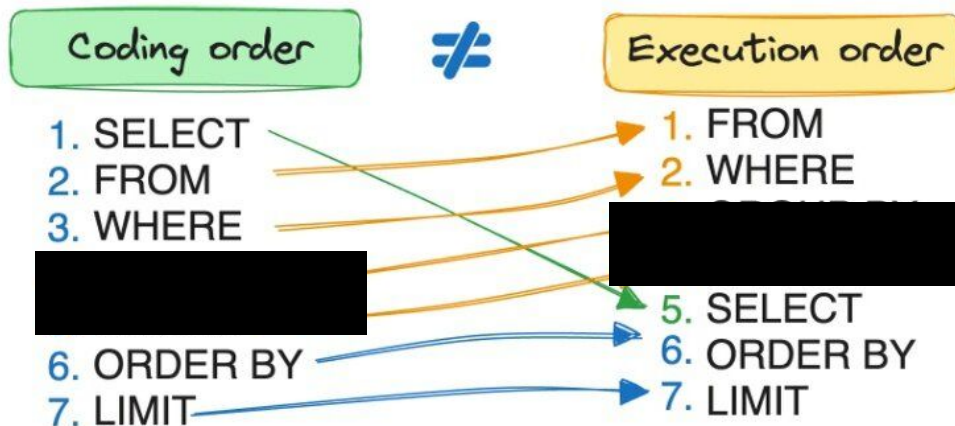
SELECT

DISTINCT (ArtistID)

FROM

Album

LIMIT 5



Будет 3 или 5 строк?

```
SELECT
```

```
    DISTINCT (ArtistID)
```

```
FROM
```

```
    Album
```

```
LIMIT 5
```

The screenshot shows the DBeaver SQL editor interface. The top panel displays the SQL query:

```
SELECT  
    DISTINCT (ArtistID)  
FROM  
    Album  
LIMIT 5
```

The bottom panel shows the results of the query in a table view. The table has two columns: 'ArtistId' and an unnamed column. The results are as follows:

	ArtistId	
1	1	
2	2	
3	3	
4	4	
5	5	

DDL SQL

Памятка

```
CREATE TABLE nutrients (  
  id int,  
  name varchar,  
  calories int  
);
```

id	name	calories
3	tomato	100
4	egg	120
5	biscuit	240

```
INSERT INTO nutrients(id, name, calories) VALUES (3, 'tomato', 100);  
INSERT INTO nutrients(id, name, calories) VALUES (4, 'egg', 120);  
INSERT INTO nutrients(id, name, calories) VALUES (5, 'biscuit', 240);
```

Создание таблицы

```
CREATE TABLE nutrients (  
  id INT PRIMARY KEY,  
  name VARCHAR,  
  calories INT,  
  when_did_i_eat DATE  
);
```

Название таблицы

Ограничитель

Тип колонки

Название колонки



Создание таблицы

```
CREATE TABLE nutrients (  
    id INT PRIMARY KEY,  
    name VARCHAR,  
    calories INTEGER,  
    when_did_i_eat DATE  
);
```

Название таблицы

Ограничитель

Тип колонки

Название колонки

Создание таблицы

```
CREATE TABLE nutrients (  
    id INT PRIMARY KEY,  
    name VARCHAR,  
    calories INT,  
    when_did_i_eat DATE  
);
```

ЧТО ВЫБРАТЬ?



Создание таблицы

```
CREATE TABLE nutrients (  
    id INT PRIMARY KEY,  
    name VARCHAR,  
    calories INT,  
    when_did_i_eat DATE  
);
```

ЧТО ВЫБРАТЬ?



Отличия заключается в
способе хранения и
извлечения данных.

Текстовые типы

TEXT

CHAR (42)

VARCHAR

VARCHAR (42)

ЧТО ВЫБРАТЬ?



Текстовые типы

TEXT

- Строки до 4Гб
- Хранится отдельно от таблицы, в таблице только указатель на место фактического хранения
- Не может быть частью **индекса**

CHAR (42)

VARCHAR

VARCHAR (42)

ЧТО ВЫБРАТЬ?



Индекс в БД — это объект, содержащий упорядоченные значения указанных столбцов таблицы и ссылки на физическое размещение записи с данными значениями.

- Индекс позволяет **ускорить поиск данных в таблице** и упорядочивание данных.

Номер	Запись
2317415	9
2338218	3
→ 2791519	8
3427511	2
4983217	6
→ 4987312	10
5711107	5
5929214	11
5933970	7
6032773	1
6329145	4

Запись	Номер	Фамилия
1	6032773	Аксенов			
2	3427511	Попова			
3	2338218	Рыкова			
4	6329145	Иванов			
5	5711107	Андреев			
6	4983217	Степанов			
7	5933970	Акатьева			
8	2791519	Сергеев			
9	2317415	Смирнов			
10	4987312	Яковлев			
11	5929214	Мухина			

Текстовые типы

TEXT

CHAR (42)

- Для неизменяемых строк (задается при создании)
- Хранятся в самой таблице
- Строки короче заполняются пробелами
- Строки длиннее обрезаются

VARCHAR

VARCHAR (42)

ЧТО ВЫБРАТЬ?



Текстовые типы

TEXT

CHAR (42)

VARCHAR

- Хранит строку переменной длины,
- Выделяется один байт для хранения длины строки.

VARCHAR (42)

- То же самое, но есть ограничение на максимальную длину

ЧТО ВЫБРАТЬ?



Работа с датами

- DATE: даты в формате YYYY-MM-DD
В SQLite `yyy-mm-dd hh:mm:ss.xxxxxx`
- TIME: время в формате HH:MM (без часового пояса)
- DATETIME: дата и время YYYY-MM-DD HH:MM
- TIMESTAMP: точность можно варьировать, можно указать часовой пояс

- TEXT, REAL И INTEGER

Работа с датами

- TEXT: даты и время сохраняются в формате
YYYY-MM-DD HH:MM.SSS или YYYY-MM-DD HH:MM
- REAL: даты хранятся как дневные **числа Джулиана** (число дней с полудня по Гринвичу 24 ноября 4714 года до н. э.)
- INTEGER: дата и время хранятся в виде количества секунд, прошедших с начала эпохи UNIX (1970-01-01 00:00:00 UTC) .

Вставка значений

```
INSERT INTO nutrients(id, name) VALUES (3, 'tomato');  
INSERT INTO nutrients(id, name) VALUES (4, 'egg');  
INSERT INTO nutrients(id, name, calories) VALUES (5, 'biscuit', 240);  
INSERT INTO nutrients(id, when_did_i_eat) VALUES (6, '2023-12-10');
```


Функции, TEXT

He SQLite



```
SELECT EXTRACT(YEAR FROM date_column) FROM table_name
SELECT YEAR(date_column) AS year FROM table_name
SELECT FORMAT('2023-11-15', 'MMMM dd, yyyy', 'en-US')
```

SELECT

`strftime('%Y', 'now')`

FROM

`nutrients`

SELECT

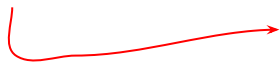
`strftime('%Y', when_did_i_eat)`

FROM

`nutrients`

Функции, TEXT

He SQLite



```
SELECT
    id,
    date('now') - when_did_i_eat
FROM
    nutrients
```

```
SELECT
    id,
    when_did_i_eat
FROM
    nutrients
WHERE
    when_did_i_eat BETWEEN '1950-01-01' AND '1991-12-31'
```



DB Browser for SQLite - C:\sqlite\test.db

File Edit View Tools Help

New Database » Open Project » Attach Database »

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT date_time, datetime(date_time) FROM orders;
```

	date_time	datetime(date_time)
1	2459549.41753472	2021-11-30 22:01:15
2	2459550.08597824	2021-12-01 14:03:48
3	2459549.87764491	2021-12-01 09:03:48

UTF-8

REAL

DB Browser for SQLite - C:\sqlite\test.db

File Edit View Tools Help

New Database » Open Project » Attach Database »

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT date_time, datetime(date_time, 'unixepoch')  
2 FROM orders;
```

	date_time	datetime(date_time, 'unixepoch')
1	1638309675	2021-11-30 22:01:15
2	1644577064	2022-02-11 10:57:44
3	1644559064	2022-02-11 05:57:44

UTF-8

INT

Удаление таблиц

```
DROP TABLE nutrients;
```

```
DROP TABLE nutrients IF EXISTS;
```



```
$result = db_query('  
SELECT *  
FROM users  
WHERE id = 5;DROP TABLE users;  
');
```

Подключение через Python

SQLite3

```
import sqlite3

db_path = "bike_1/bike_1.sqlite"
connector = sqlite3.connect(db_path) # Служит посредником между кодом Python и базой данных SQLite
cursor = connector.cursor()

query_1 = '''
SELECT
    name
FROM
    sqlite_master
WHERE
    type='table'
'''

query_2 = '''
SELECT
    *
FROM
    station
'''

cursor.execute(query_1)
results_1 = cursor.fetchall()
print(results_1)
```

SQLite3

```
import sqlite3

db_path = "bike_1/bike_1.sqlite"
connector = sqlite3.connect(db_path)  # Служит посредником между
кодом Python и базой данных SQLite
cursor = connector.cursor()

connector.commit()  # Только, если есть cursor
cursor.close()
connector.close()
```

SQLite3

```
query_1 = '''
SELECT
    name
FROM
    sqlite_master
WHERE
    type='table'
'''
```

```
cursor.execute(query_1)
results_1 = cursor.fetchall()
print(results_1)
```

```
query_2 = '''
SELECT
    *
FROM
    station
'''
```

```
cursor.execute(query_2)
```



GROUP BY

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    COUNT(id)
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малfoy	Слизерин
5	Люциус	Малfoy	Слизерин
6	Фред	Уизли	Гриффиндор

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    COUNT(id)
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

decision	column 1
Гриффиндор	4
Слизерин	2

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    COUNT(id)
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	Поттер	<u>Гриффиндор</u>
2	Рон	Уизли	<u>Гриффиндор</u>
3	Джинни	Уизли	<u>Гриффиндор</u>
4	Драко	Малфой	<u>Слизерин</u>
5	Люциус	Малфой	<u>Слизерин</u>
6	Фред	Уизли	<u>Гриффиндор</u>

decision	column 1
<u>Гриффиндор</u>	4
<u>Слизерин</u>	2

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    COUNT(id) as number
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

decision	number
Гриффиндор	4
Слизерин	2

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    MAX(id)
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	Поттер	<u>Гриффиндор</u>
2	Рон	Уизли	<u>Гриффиндор</u>
3	Джинни	Уизли	<u>Гриффиндор</u>
4	Драко	Малфой	<u>Слизерин</u>
5	Люциус	Малфой	<u>Слизерин</u>
6	Фред	Уизли	<u>Гриффиндор</u>

decision	column 1
<u>Гриффиндор</u>	?
<u>Слизерин</u>	?

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    MAX(id)
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	Поттер	<u>Гриффиндор</u>
2	Рон	Уизли	<u>Гриффиндор</u>
3	Джинни	Уизли	<u>Гриффиндор</u>
4	Драко	Малфой	<u>Слизерин</u>
5	Люциус	Малфой	<u>Слизерин</u>
6	Фред	Уизли	<u>Гриффиндор</u>

decision	column 1
<u>Гриффиндор</u>	6
<u>Слизерин</u>	5

Группировки

Каково количество героев в разных факультетах?

```
SELECT
    decision,
    AVG(id)
FROM
    heroes
GROUP BY
    decision
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	Поттер	<u>Гриффиндор</u>
2	Рон	Уизли	<u>Гриффиндор</u>
3	Джинни	Уизли	<u>Гриффиндор</u>
4	Драко	Малфой	<u>Слизерин</u>
5	Люциус	Малфой	<u>Слизерин</u>
6	Фред	Уизли	<u>Гриффиндор</u>

decision	column 1
<u>Гриффиндор</u>	3
<u>Слизерин</u>	4.5

Группировки

Так нельзя!

```
SELECT
    decision,
    AVG(id)
FROM
    heroes
GROUP BY
    surname
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

decision	column 1
<u>Поттер</u>	1
<u>Уизли</u>	3.6666
<u>Малфой</u>	4.5

Группировки

Так нельзя!

```
SELECT
    decision,
    AVG(id)
FROM
    heroes
GROUP BY
    surname
```

Должны
совпадать или
агрегация

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

decision	column 1
<u>Поттер</u>	1
<u>Уизли</u>	3.6666
<u>Малфой</u>	4.5

Группировки

Так можно!

```
SELECT
    surname,
    AVG(id)
FROM
    heroes
GROUP BY
    surname
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Уизли</u>	3.6666
<u>Малфой</u>	4.5

Группировки по нескольким столбцам

Каково количество героев в разных факультетах?

```
SELECT
    name,
    surname,
    COUNT(id) AS number
FROM
    heroes
GROUP BY
    name,
    surname
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

Группировки по нескольким столбцам

Каково количество героев в разных факультетах?

```
SELECT
    name,
    surname,
    COUNT(id) AS number
FROM
    heroes
GROUP BY
    name,
    surname
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

name	surname	number
Гарри	Поттер	1
Рон	Уизли	1
Джинни	Уизли	1
Драко	Малфой	1
Люциус	Малфой	1
Фред	Уизли	1

Группировки по нескольким столбцам

Каково количество героев в разных факультетах?

```
SELECT
    name || surname AS full,
    COUNT(id) AS number
FROM
    heroes
GROUP BY
    name || surname
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

full	number
ГарриПоттер	1
РонУизли	1
ДжинниУизли	1
ДракоМалфой	1
ЛюциусМалфой	1
ФредУизли	1

HAVING

Если хотим инфо только по длинным фамилиям

Как это сделать?

```
SELECT
    surname,
    AVG(id)
FROM
    heroes
GROUP BY
    surname
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Уизли</u>	3.6666
<u>Малфой</u>	4.5

Если хотим инфо только по длинным фамилиям

WHERE

SELECT

surname,

AVG (**id**)

FROM

heroes

WHERE

LENGTH (surname) >= 6

GROUP BY

surname

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Малфой</u>	4.5

Если хотим инфо только по длинным фамилиям

```
SELECT
    surname,
    AVG(id)
FROM
    heroes
GROUP BY
    surname
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Уизли</u>	3.6666
<u>Малфой</u>	4.5

Если хотим инфо только по длинным фамилиям

```
SELECT
    *
FROM (
    SELECT
        surname,
        AVG(id)
    FROM
        heroes
    GROUP BY
        surname
)
WHERE
    LENGTH(surname) >= 6
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Малфой</u>	4.5

Если хотим инфо только по длинным фамилиям

```
SELECT
    *
FROM
    heroes
```

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Малфой</u>	4.5

Если хотим инфо только по длинным фамилиям

HAVING

SELECT

surname,

AVG (**id**)

FROM

heroes

GROUP BY

surname

HAVING

LENGTH (surname) >= 6

heroes			
id	name	surname	<u>decision</u>
1	Гарри	<u>Поттер</u>	Гриффиндор
2	Рон	<u>Уизли</u>	Гриффиндор
3	Джинни	<u>Уизли</u>	Гриффиндор
4	Драко	<u>Малфой</u>	Слизерин
5	Люциус	<u>Малфой</u>	Слизерин
6	Фред	<u>Уизли</u>	Гриффиндор

surname	column 1
<u>Поттер</u>	1
<u>Малфой</u>	4.5

В чем разница?

- Сортируем исходные данные VS Сортируем группы
- **WHERE + HAVING** и **WHERE + WHERE** могут отличаться

В чем разница?

- Сортируем исходные данные VS Сортируем группы

WHERE Clause	HAVING Clause
Filters rows before groups are aggregated.	Filters groups after the aggregation process..
WHERE Clause can be used without GROUP BY Clause	HAVING Clause can be used with GROUP BY Clause
WHERE Clause implements in row operations	HAVING Clause implements in column operation
WHERE Clause cannot contain aggregate function	HAVING Clause can contain aggregate function
WHERE Clause can be used with SELECT, UPDATE, DELETE statement.	HAVING Clause can only be used with SELECT statement.
WHERE Clause is used before GROUP BY Clause	HAVING Clause is used after GROUP BY Clause
WHERE Clause is used with single row function like UPPER, LOWER etc.	HAVING Clause is used with multiple row function like SUM, COUNT etc.

JOIN

Джойны

Вывести основателей факультетов
героев.



1	Гарри	Поттер	Гриффиндор	665
2	Рон	Уизли	Гриффиндор	665
3	Джинни	Уизли	Гриффиндор	665
4	Драко	Малфой	Слизерин	666
5	Люциус	Малфой	Слизерин	666
6	Фред	Уизли	Гриффиндор	665

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

school		
faculty	founder	students
Гриффиндор	Годрик	665
Слизерин	Салазар	666
Когтевран	Кандида	333
Пуффендуй	Пенелопа	336

Джойны

Вывести основателей факультетов героев.

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малfoy	Слизерин
5	Люциус	Малfoy	Слизерин
6	Фред	Уизли	Гриффиндор

school		
faculty	founder	students
Гриффиндор	Годрик	665
Слизерин	Салазар	666
Когтевран	Кандида	333
Пуффендуй	Пенелопа	336

Джойны

Вывести основателей факультетов героев.

```
SELECT *  
FROM  
    heroes  
JOIN  
    school  
ON  
    heroes.decision = school.faculty
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

school		
faculty	founder	students
Гриффиндор	Годрик	665
Слизерин	Салазар	666
Когтевран	Кандида	333
Пуффендуй	Пенелопа	336

Джойны

Вывести основателей факультетов
героев.

```
SELECT
    *
FROM
    heroes
JOIN
    school
ON
    heroes.decision = school.faculty
```

1	Гарри	Поттер	Гриффиндор	Годрик	665
2	Рон	Уизли	Гриффиндор	Годрик	665
3	Джинни	Уизли	Гриффиндор	Годрик	665
4	Драко	Малfoy	Слизерин	Салазар	666
5	Люциус	Малfoy	Слизерин	Салазар	666
6	Фред	Уизли	Гриффиндор	Годрик	665

Джойны

Вывести основателей факультетов
героев.

```
SELECT
    *
FROM
    heroes
JOIN
    school
ON
    heroes.decision = school.faculty
```

1	Гарри	Поттер	Гриффиндор	Годрик	665
2	Рон	Уизли	Гриффиндор	Годрик	665
3	Джинни	Уизли	Гриффиндор	Годрик	665
4	Драко	Малfoy	Слизерин	Салазар	666
5	Люциус	Малfoy	Слизерин	Салазар	666
6	Фред	Уизли	Гриффиндор	Годрик	665

Джойны

Вывести основателей факультетов
героев.

```
SELECT
    heroes.*, school.students
FROM
    heroes
JOIN
    school
ON
    heroes.decision = school.faculty
```

1	Гарри	Поттер	Гриффиндор	665
2	Рон	Уизли	Гриффиндор	665
3	Джинни	Уизли	Гриффиндор	665
4	Драко	Малfoy	Слизерин	666
5	Люциус	Малfoy	Слизерин	666
6	Фред	Уизли	Гриффиндор	665

Джойны

Вывести основателей факультетов героев.

```
SELECT
    *
FROM
    heroes
JOIN
    school
ON
    heroes.decision = school.faculty
```

heroes			
id	name	surname	decision
1	Гарри	Поттер	Гриффиндор
2	Рон	Уизли	Гриффиндор
3	Джинни	Уизли	Гриффиндор
4	Драко	Малфой	Слизерин
5	Люциус	Малфой	Слизерин
6	Фред	Уизли	Гриффиндор

school		
faculty	founder	students
Гриффиндор	Годрик	665
Слизерин	Салазар	666
Когтевран	Кандида	333
Пуффендуй	Пенелопа	336

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

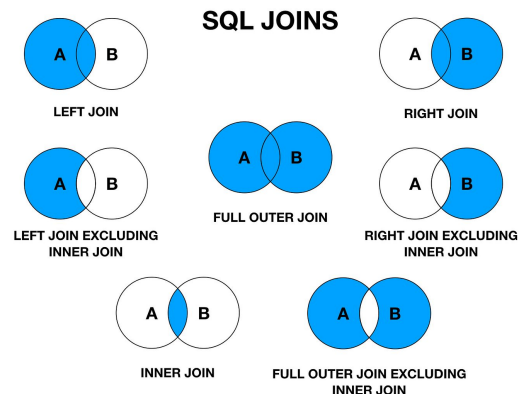
id		students

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

id		students



<table> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	1	2	3	INNER JOIN	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> <tr><td>C</td></tr> </table>	A	B	C	=	<table> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	2	3	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> </table>	A	B		
1																	
2																	
3																	
A																	
B																	
C																	
2																	
3																	
A																	
B																	
<table> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	1	2	3	LEFT JOIN	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> <tr><td>C</td></tr> </table>	A	B	C	=	<table> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	1	2	3	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> </table>	A	B	
1																	
2																	
3																	
A																	
B																	
C																	
1																	
2																	
3																	
A																	
B																	
<table> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	1	2	3	RIGHT JOIN	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> <tr><td>C</td></tr> </table>	A	B	C	=	<table> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	2	3	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> <tr><td>C</td></tr> </table>	A	B	C	
1																	
2																	
3																	
A																	
B																	
C																	
2																	
3																	
A																	
B																	
C																	
<table> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	1	2	3	«FULL» JOIN	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> <tr><td>C</td></tr> </table>	A	B	C	=	<table> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> </table>	1	2	3	<table> <tr><td>A</td></tr> <tr><td>B</td></tr> <tr><td>C</td></tr> </table>	A	B	C
1																	
2																	
3																	
A																	
B																	
C																	
1																	
2																	
3																	
A																	
B																	
C																	

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

id		students



INNER JOIN

1
2
3

A
B
C

=

2
3

A
B

LEFT JOIN

1
2
3

A
B
C

=

1
2
3

A
B

RIGHT JOIN

1
2
3

A
B
C

=

2
3

A
B
C

«FULL» JOIN

1
2
3

A
B
C

=

1
2
3

A
B
C

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

id		students



INNER JOIN

1
2
3

A
B
C

=

2
3

A
B

LEFT JOIN

1
2
3

A
B
C

=

1
2
3

A
B

RIGHT JOIN

1
2
3

A
B
C

=

2
3

A
B
C

«FULL» JOIN

1
2
3

A
B
C

=

1
2
3

A
B
C

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

id		students
2	B	666
2	B	777



INNER JOIN

1
2
3

A
B
C

=

2
3

A
B

LEFT JOIN

1
2
3

A
B
C

=

1
2
3

A
B

RIGHT JOIN

1
2
3

A
B
C

=

2
3

A
B
C

«FULL» JOIN

1
2
3

A
B
C

=

1
2
3

A
B
C

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

id		students
2	B	666
2	B	777



INNER JOIN

1
2
3

A
B
C

=

2
3

A
B

LEFT JOIN

1
2
3

A
B
C

=

1
2
3

A
B

RIGHT JOIN

1
2
3

A
B
C

=

2
3

A
B
C

«FULL» JOIN

1
2
3

A
B
C

=

1
2
3

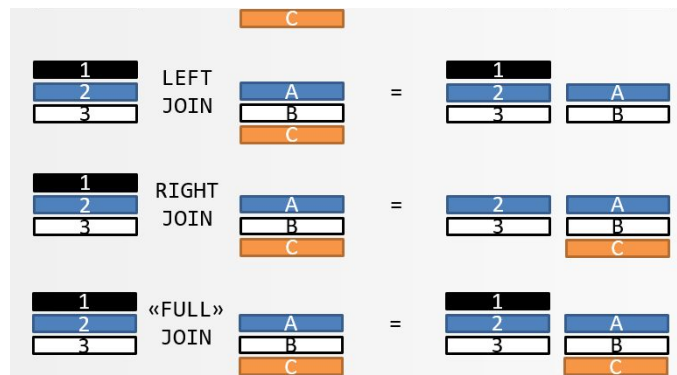
A
B
C

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
D	888

id		students
2	B	666
2	B	777
3	B	666
3	B	777



Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
B	999
D	888

id		students
2	B	666
2	B	777
3	B	666
3	B	777

Джойны

heroes	
id	decision
1	A
2	B
3	B
4	C

school	
faculty	students
B	666
B	777
B	999
D	888

id		students
2	B	666
2	B	777
2	B	999
3	B	666
3	B	777
3	B	999

Про JOIN



В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

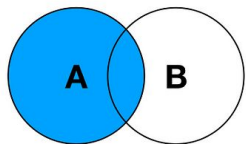
table_1	
name	age
	7
	8
	9

table_2	
alias	weight
	10
	200

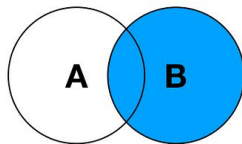
```
SELECT
    *
FROM
    table_1
JOIN
    table_2
ON
    table_1.name = table_2.alias
```

INNER JOIN

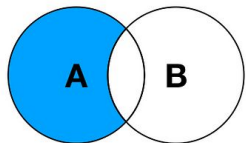
Προ JOIN



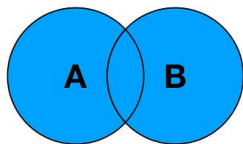
LEFT JOIN



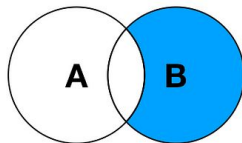
RIGHT JOIN



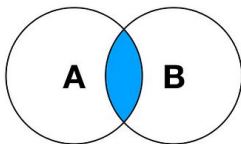
LEFT JOIN EXCLUDING
INNER JOIN



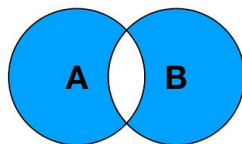
FULL OUTER JOIN



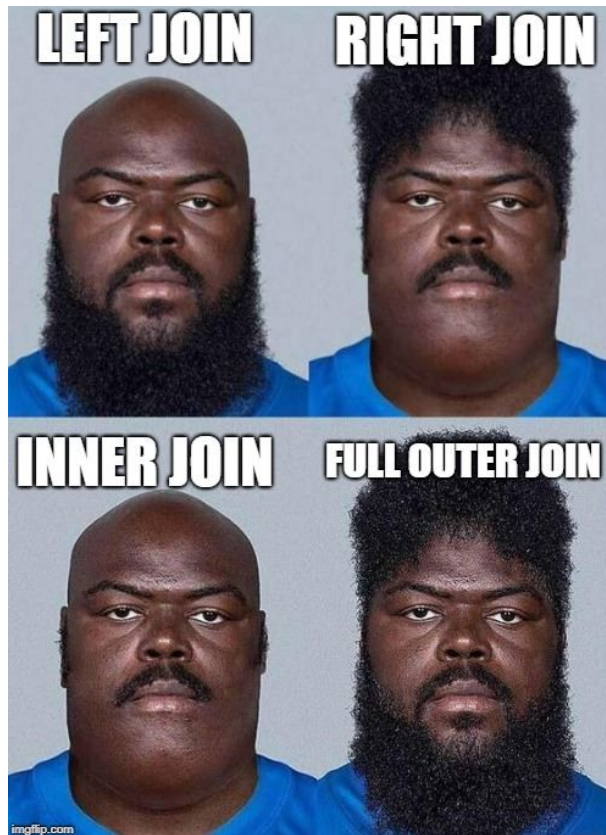
RIGHT JOIN EXCLUDING
INNER JOIN



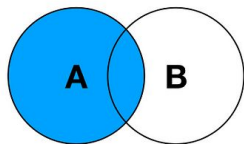
INNER JOIN



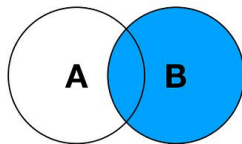
FULL OUTER JOIN EXCLUDING
INNER JOIN



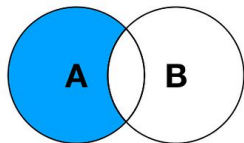
Про JOIN



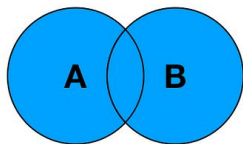
LEFT JOIN



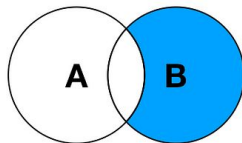
RIGHT JOIN



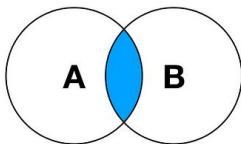
LEFT JOIN EXCLUDING
INNER JOIN



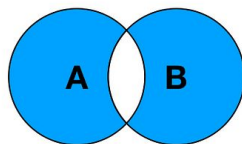
FULL OUTER JOIN



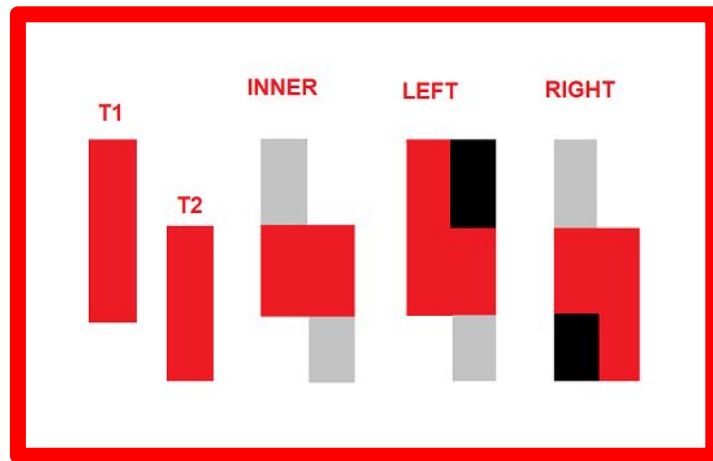
RIGHT JOIN EXCLUDING
INNER JOIN



INNER JOIN

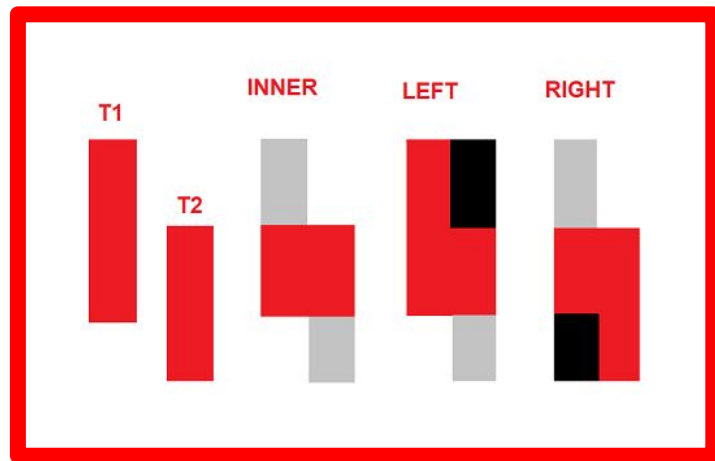


FULL OUTER JOIN EXCLUDING
INNER JOIN



Про JOIN

<table><tr><td>1</td></tr><tr><td>2</td></tr><tr><td>3</td></tr></table>	1	2	3	INNER JOIN	<table><tr><td>A</td></tr><tr><td>B</td></tr><tr><td>C</td></tr></table>	A	B	C	=	<table><tr><td>2</td></tr><tr><td>3</td></tr></table>	2	3	<table><tr><td>A</td></tr><tr><td>B</td></tr></table>	A	B		
1																	
2																	
3																	
A																	
B																	
C																	
2																	
3																	
A																	
B																	
<table><tr><td>1</td></tr><tr><td>2</td></tr><tr><td>3</td></tr></table>	1	2	3	LEFT JOIN	<table><tr><td>A</td></tr><tr><td>B</td></tr><tr><td>C</td></tr></table>	A	B	C	=	<table><tr><td>1</td></tr><tr><td>2</td></tr><tr><td>3</td></tr></table>	1	2	3	<table><tr><td>A</td></tr><tr><td>B</td></tr></table>	A	B	
1																	
2																	
3																	
A																	
B																	
C																	
1																	
2																	
3																	
A																	
B																	
<table><tr><td>1</td></tr><tr><td>2</td></tr><tr><td>3</td></tr></table>	1	2	3	RIGHT JOIN	<table><tr><td>A</td></tr><tr><td>B</td></tr><tr><td>C</td></tr></table>	A	B	C	=	<table><tr><td>2</td></tr><tr><td>3</td></tr></table>	2	3	<table><tr><td>A</td></tr><tr><td>B</td></tr><tr><td>C</td></tr></table>	A	B	C	
1																	
2																	
3																	
A																	
B																	
C																	
2																	
3																	
A																	
B																	
C																	
<table><tr><td>1</td></tr><tr><td>2</td></tr><tr><td>3</td></tr></table>	1	2	3	«FULL» JOIN	<table><tr><td>A</td></tr><tr><td>B</td></tr><tr><td>C</td></tr></table>	A	B	C	=	<table><tr><td>1</td></tr><tr><td>2</td></tr><tr><td>3</td></tr></table>	1	2	3	<table><tr><td>A</td></tr><tr><td>B</td></tr><tr><td>C</td></tr></table>	A	B	C
1																	
2																	
3																	
A																	
B																	
C																	
1																	
2																	
3																	
A																	
B																	
C																	



Максимальное количество [?]

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
	7
	8
	9

table_2	
alias	weight
	10
	200

```
SELECT
    *
FROM
    table_1
JOIN
    table_2
ON
    table_1.name = table_2.alias
```

Максимальное количество

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
a	8
a	9

table_2	
alias	weight
a	10
a	200

INNER JOIN	

Максимальное количество

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
a	8
a	9

table_2	
alias	weight
a	10
a	200

INNER JOIN			
name	age	alias	weight

Максимальное количество

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
a	8
a	9

table_2	
alias	weight
a	10
a	200

INNER JOIN			
name	age	alias	weight

Максимальное количество

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
a	8
a	9



table_2	
alias	weight
a	10
a	200

INNER JOIN			
name	age	alias	weight
a	7	a	10
a	7	a	200

Максимальное количество

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
a	8
a	9

table_2	
alias	weight
a	10
a	200



INNER JOIN			
name	age	alias	weight
a	7	a	10
a	7	a	200
a	8	a	10
a	8	a	200

Максимальное количество

В одной таблице 3 строки, в другой – 2.

Какое **максимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
a	8
a	9



table_2	
alias	weight
a	10
a	200

INNER JOIN			
name	age	alias	weight
a	7	a	10
a	7	a	200
a	8	a	10
a	8	a	200
a	9	a	10
a	9	a	200

Минимальное количество [?]

Минимальное количество [?]

В одной таблице 3 строки, в другой – 2.

Какое **минимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
	7
	8
	9

table_2	
alias	weight
	10
	200

```
SELECT
    *
FROM
    table_1
JOIN
    table_2
ON
    table_1.name = table_2.alias
```

Минимальное количество [?]

В одной таблице 3 строки, в другой – 2.

Какое **минимальное** количество строк может быть в их **INNER JOIN**?

table_1	
name	age
a	7
b	8
c	9

table_2	
alias	weight
d	10
e	200

```
SELECT
    *
FROM
    table_1
JOIN
    table_2
ON
    table_1.name = table_2.alias
```

Минимальное количество = 0

В одной таблице 3 строки, в другой – 2.

Какое **минимальное** количество строк может быть в их **INNER JOIN**?

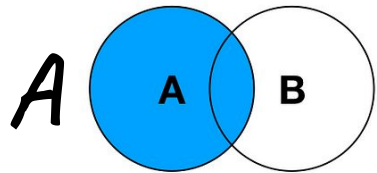
table_1	
name	age
a	7
b	8
c	9

table_2	
alias	weight
d	10
e	200

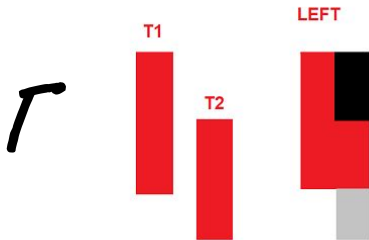
```
SELECT
    *
FROM
    table_1
JOIN
    table_2
ON
    table_1.name = table_2.alias
```

Другие JOIN

LEFT JOIN



LEFT JOIN



Null



LEFT JOIN

1	A
2	B
3	

=

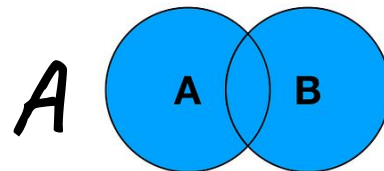
1	A
2	B
3	

Diagram illustrating a LEFT JOIN operation. The first table (left) has 3 rows. The second table (right) has 2 rows. The result of the LEFT JOIN is a table with 3 rows, where the first two rows are joined and the third row is null.

$$\mathcal{B}$$

- Возвращает все записи первой (левой) таблицы,
- ... даже если они не соответствуют записям во второй (правой) таблице.
- ... там Null

FULL OUTER JOIN



FULL OUTER JOIN

Null



Diagram illustrating a join operation on a full table:

1
2
3

«FULL»
JOIN

A
B
C

=

1
2
3

A
B
C

The diagram shows a join operation between a table with 3 rows (labeled 1, 2, 3) and a table with 3 rows (labeled A, B, C). The result is a table with 3 rows (labeled 1, 2, 3) and a table with 3 rows (labeled A, B, C). The result is a table with 3 rows (labeled 1, 2, 3) and a table with 3 rows (labeled A, B, C).

$$\mathcal{B}$$

- **FULL JOIN = FULL OUTER JOIN**
выполняют одну и ту же функцию.
- Возвращает все строки из обеих таблиц
- Если нет соответствующих строк в другой таблице - NULL

LEFT JOIN, минимум

LEFT JOIN

Минимальное количество

table_1	
name	age
a	7
b	8
c	9

table_2	
alias	weight
d	10
e	200

LEFT JOIN

Минимальное количество = 3

table_1	
name	age
a	7
b	8
c	9

table_2	
alias	weight
d	10
e	200

LEFT JOIN			
name	age	alias	weight
a	7	Null	Null
b	8	Null	Null
c	9	Null	Null

LEFT JOIN, максимум

LEFT JOIN

Максимальное количество

table_1	
name	age
a	7
a	8
a	9

table_2	
alias	weight
a	10
a	200

LEFT JOIN = INNER JOIN

Максимальное количество = 6

table_1	
name	age
a	7
a	8
a	9

table_2	
alias	weight
a	10
a	200

FULL OUTER JOIN			
name	age	alias	weight
a	7	a	10
a	7	a	200
a	8	a	10
a	8	a	200
a	9	a	10
a	9	a	200

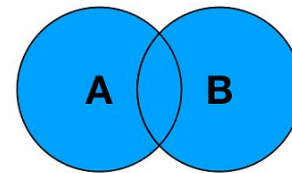
FULL OUTER JOIN

FULL OUTER JOIN

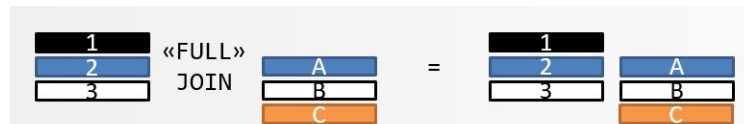
Минимальное количество

table_1	
name	age

table_2	
alias	weight



FULL OUTER JOIN

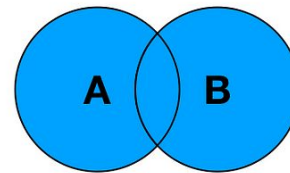


FULL OUTER JOIN

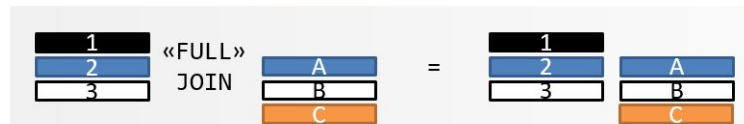
Минимальное количество

table_1	
name	age
a	7
b	8
c	9

table_2	
alias	weight
a	10
b	200



FULL OUTER JOIN



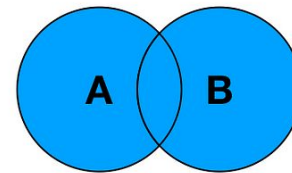
FULL OUTER JOIN

Минимальное количество = 3

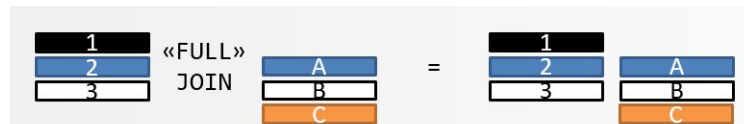
table_1	
name	age
a	7
b	8
c	9

table_2	
alias	weight
a	10
b	200

FULL OUTER JOIN			
name	age	alias	weight
a	7	a	10
b	8	b	200
c	9	Null	Null



FULL OUTER JOIN



Итого

Итого

	Минимальное кол-во строк	Максимальное кол-во строк
INNER JOIN		
LEFT JOIN		
RIGHT JOIN		
FULL OUTER JOIN		

n {

table_1	
name	age

m {

table_2	
alias	weight

Итого

	Минимальное кол-во строк	Максимальное кол-во строк
INNER JOIN		
LEFT JOIN		
RIGHT JOIN		
FULL OUTER JOIN		

n {

table_1	
name	age

m {

table_2	
alias	weight

Какое
минимальное
кол-во?

Итого

	Минимальное кол-во строк	Максимальное кол-во строк
INNER JOIN	0	
LEFT JOIN		
RIGHT JOIN		
FULL OUTER JOIN		

n {

table_1	
name	age

m {

table_2	
alias	weight

Какое
минимальное
кол-во?

Итого

	Минимальное кол-во строк	Максимальное кол-во строк
INNER JOIN	0	nm
LEFT JOIN		nm
RIGHT JOIN		nm
FULL OUTER JOIN		nm

n {

table_1	
name	age

m {

table_2	
alias	weight

Итого

	Минимальное кол-во строк	Максимальное кол-во строк
INNER JOIN	0	nm
LEFT JOIN	n	nm
RIGHT JOIN	m	nm
FULL OUTER JOIN		nm

n {

table_1	
name	age

m {

table_2	
alias	weight

Итого

	Минимальное кол-во строк	Максимальное кол-во строк
INNER JOIN	0	nm
LEFT JOIN	n	nm
RIGHT JOIN	m	nm
FULL OUTER JOIN	$\max(m, n)$	nm

n {

table_1	
name	age

m {

table_2	
alias	weight

EXISTS

EXISTS

SELECT

*

FROM

customers

WHERE EXISTS (

SELECT

*

FROM

orders

WHERE

customers.customer_id = orders.customer_id

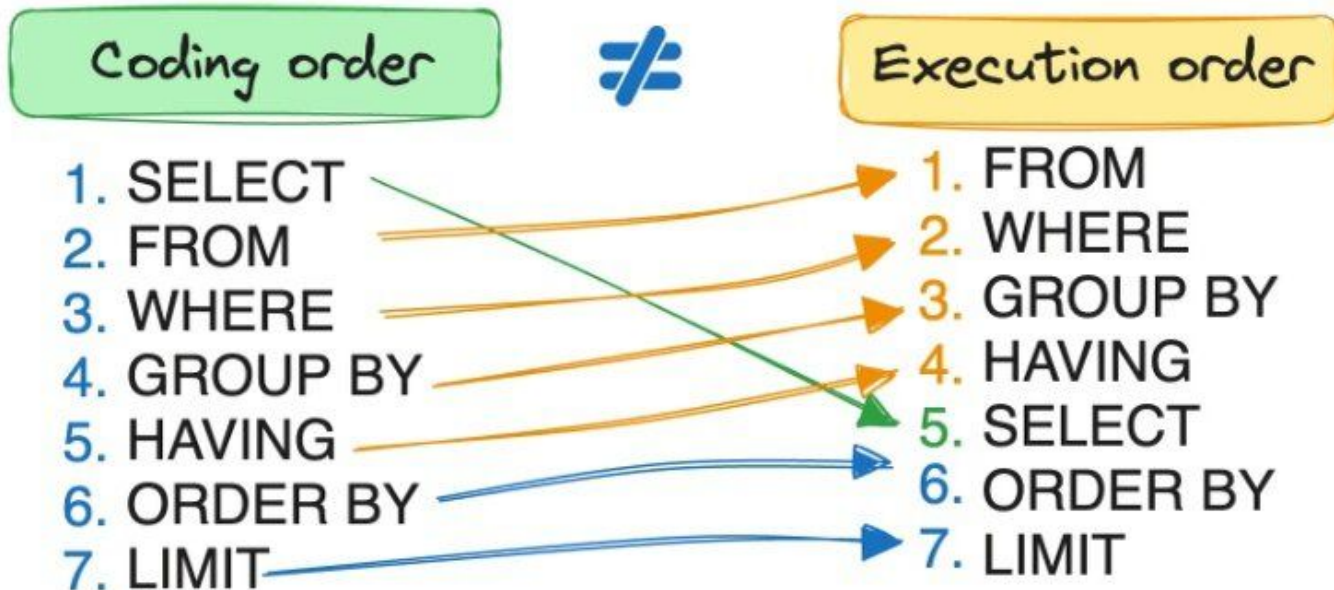
);

order_id	customer_id	order_date
1	7000	2019/06/18
2	5000	2019/06/18
3	8000	2019/06/19
4	4000	2019/06/20
5	NULL	2019/07/01

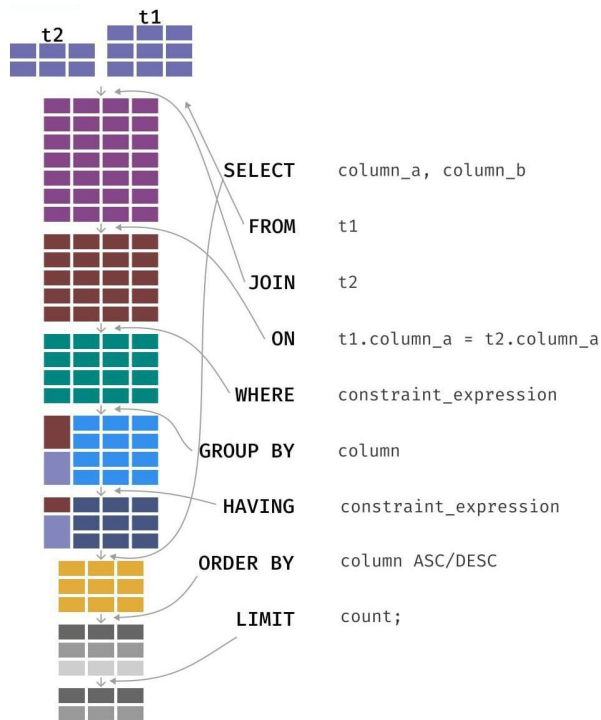
customer_id	first_name	last_name	favorite_website
4000	Justin	Bieber	google.com
5000	Selena	Gomez	bing.com
6000	Mila	Kunis	yahoo.com
7000	Tom	Cruise	oracle.com
8000	Johnny	Depp	NULL
9000	Russell	Crowe	google.com

Подведение итогов

Порядок выполнения

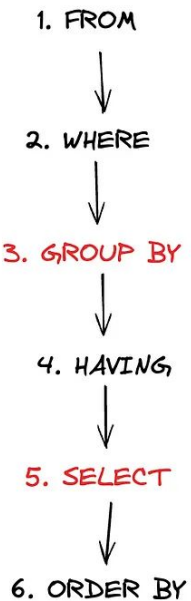


Порядок выполнения

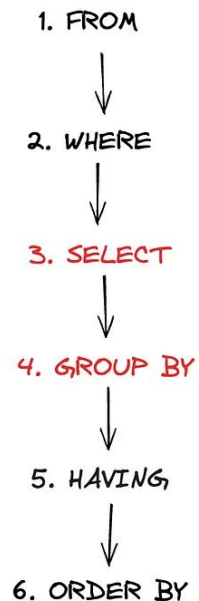


Диалекты SQL

SQL Server



MySQL



Прошлый семинар

Занятие «Обзор про базы данных, SQL и теорию множеств. Таблицы, представления, простые выборки»

Цели занятия

- ✓ понять область применения языка запросов SQL;
- создать подключение к SQL базе данных на языке Python;

[Показать еще](#)

Краткое содержание

- ✓ поймем какие базы данных бывают;
- ✓ поймем чем реляционные SQL базы отличаются от noSQL;
- посмотрим какие есть инструменты для доступа к базе данных и выполнения команд;
- создадим подключение на Python и с использованием инструмента DBeaver;
- ✓ познакомимся с конструкциями запросов и ~~работой с датами~~;
- ✓ выполним несложные запросы с условиями фильтрации.

[Свернуть](#)

Результаты

- создаст подключение к SQL базе данных (MySQL, PostgreSQL, SQLite) на языке Python;
- ✓ выполнит простые запросы на выборку данных из базы данных с условиями фильтрации по условию на значение полей и ~~условию на дату/время~~;
- ✓ создаст простые таблицы и связи между ними (SQLite).

[Свернуть](#)

Текущий семинар

Занятие «Join, exists, вложенные запросы, group by, having»

Цели занятия

научиться делать более сложные выборки из реляционных баз, которые включают данные из нескольких таблиц;
объединение данных в виде подзапросов и использование структуры JOIN.

Краткое содержание

объединение данных с использованием подзапросов;
объединение данных из нескольких таблиц с использованием конструкции JOIN;
различие в применении конструкции WHERE и HAVING.

[Свернуть](#)

Ваше мнение

- <https://otus.ru/polls/112655/>

Практика:

<https://classroom.google.com/c/NzUxMjQ5ODQxNzAx?cjc=hje7u7q>

Дополнительные материалы

Ссылки

- Типы данных (MySQL): <https://timeweb.cloud/blog/typy-dannyh-v-mysql>
- Работа с датами: <https://metanit.com/sql/sqlite/6.3.php>