Практика №1

Базовый SQL

```
!pip install ipython-sql #не обязательно запускать в colab
```

Показать скрытые выходные данные

```
%load_ext sql
%sql sqlite://
%config SqlMagic.style = '_DEPRECATED_DEFAULT'
```

```
%%sql
SELECT 'Hello, world';
```

```
 * sqlite://
Done.
```

'Hello, world'
Hello, world

Пара замечаний:

- Мы используем веб-интерфейс для python, поэтому для запуска SQL запросов необходимо применить ряд вещей:
  - Необходимо подключить расширение SQL через так называемые magic command. Более подробно здесь
  - Для работы с SQL надо вызвать либо %sql для однострочной команды, либо %%sql - для многострочной
  - При использовании выражения SELECT результатом вывода является таблица, но при этом в notebook выводится уже внутреннее представление языка python (объект класса sql.run.resultset, подробности далее), что приводит к ряду несоответствий (например, None вместо NULL)

Давайте создадим таблицу, заполним ее и сделаем какой-нибудь запрос!

CREATE TABLE, INSERT

```
%%sql

DROP TABLE IF EXISTS Product;

CREATE TABLE Product(
        pname        varchar(20) PRIMARY KEY,  -- имя продукта
        price        money DEFAULT 0,          -- цена продукта; money = decimal(n,2)
        category     char(20),                 -- категория товара
        manufacturer varchar(20) NOT NULL      -- производитель
);
```

```
 * sqlite://
Done.
Done.
[]
```

Особенности SQLite: https://www.sqlite.org/datatype3.html

Типы данных:

- TEXT
- NUMERIC
- INTEGER
- REAL
- BLOB

```
%%sql

INSERT INTO Product (pname, price, category, manufacturer) values
('Тетрадь', 39.99, 'Канцелярия', 'Академия холдинг');

INSERT INTO Product values('Клавиатура', 949.99, 'Техника', 'Sven');

INSERT INTO Product
    values ('Степлер', 129.99, 'Канцелярия', 'Brauberg'),
           ('Батарейка', 39.99, 'Для дома', 'Krona'),
           ('Лампочка', 89.70, 'Для дома', 'Energolux');
```

```
 * sqlite://
1 rows affected.
1 rows affected.
3 rows affected.
[]
```

```
%%sql
insert into product (pname, category) values ('Веб-камера', 'Техника');
```

```
 * sqlite://
(sqlite3.IntegrityError) NOT NULL constraint failed: Product.manufacturer
[SQL: insert into product (pname, category) values ('Веб-камера', 'Техника');]
(Background on this error at: https://sqlalche.me/e/20/gkpj)
```

```
%%sql
insert into product (pname, manufacturer) values ('Веб-камера', 'Sven');
```

```
 * sqlite://
1 rows affected.
[]
```

Посмотрим на полученную таблицу.

```
%sql select * from product;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Тетрадь	39.99	Канцелярия	Академия холдинг
Клавиатура	949.99	Техника	Sven
Степлер	129.99	Канцелярия	Brauberg
Батарейка	39.99	Для дома	Krona
Лампочка	89.7	Для дома	Energolux
Веб-камера	0	None	Sven

```
%%sql
INSERT INTO Product
    values ('Батарейка', 50, 'Для дома', 'Energizer');
```

```
 * sqlite://
(sqlite3.IntegrityError) UNIQUE constraint failed: Product.pname
[SQL: INSERT INTO Product
    values ('Батарейка', 50, 'Для дома', 'Energizer');]
(Background on this error at: https://sqlalche.me/e/20/gkpj)
```

Иногда бывает полезно применять UNIQUE при создании атрибута

Немного SQL терминологии

- *имя* таблицы - product.
- Каждая строка таблицы называется *строкой* или *кортежем*.
- Заметьте, что все кортежи имеют *поля* или *атрибуты*.
- Количество строк называет *мощностью*, в то время как количество атрибутов *арностью*

Простые запросы

- Рассмотрим SQL-запросы на примерах

> SELECT <Множество атрибутов>
> FROM <список таблиц и условие на соединение>
> WHERE <список условий>

Это простейший SELECT-FROM-WHERE (SFW) блок

```
%%sql
SELECT *
FROM Product
WHERE category = 'Канцелярия' AND manufacturer = 'Brauberg';
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Степлер	129.99	Канцелярия	Brauberg

* - обозначает вывод всех полей, которые были описаны в поле FROM

Посмотрим на примеры *проекции*, то есть получим только несколько атрибутов запроса.

```
%%sql
SELECT Pname,
       Price,
       Manufacturer
  FROM Product;
```

```
 * sqlite://
Done.
```

pname	price	manufacturer
Тетрадь	39.99	Академия холдинг
Клавиатура	949.99	Sven
Степлер	129.99	Brauberg
Батарейка	39.99	Krona
Лампочка	89.7	Energolux
Веб-камера	0	Sven

На выходе *все еще* таблица и ее схема -

> Answer(pname, price, manufacturer)

Можно объединять выборку и проекцию + менять результат

```
%%sql
SELECT Pname, Price * 2 as newname, Manufacturer
FROM Product
WHERE category in ('Техника', 'Канцелярия');
```

```
 * sqlite://
Done.
```

pname	newname	manufacturer
Тетрадь	79.98	Академия холдинг
Клавиатура	1899.98	Sven
Степлер	259.98	Brauberg

```
%%sql
SELECT pname || '->' || price || '->' || manufacturer as product_concat
FROM Product
WHERE category = 'Техника';
```

```
 * sqlite://
Done.
```

product_concat
Клавиатура->949.99->Sven

На выходе запроса к таблице - снова таблица

```
%%sql
SELECT manufacturer, price
FROM Product p0
WHERE p0.price < 100.00;
```

```
 * sqlite://
Done.
```

manufacturer	price
Академия холдинг	39.99
Krona	39.99
Energolux	89.7
Sven	0

Вложенный запрос

```
%%sql
SELECT *
FROM
    (SELECT p0.manufacturer, price
     FROM Product p0
     WHERE p0.price < 100.00) p1 -- вложенный запрос
WHERE manufacturer = "Krona"
```

```
 * sqlite://
Done.
```

manufacturer	price
Krona	39.99

Небольшие детали

- Некоторые элементы регистро-независимые:
  - Одно и то же: SELECT, Select, select
  - Одно и то же: Product, product
  - Разные: "Техника", "техника" (Здесь это строковая константа)

LIKE

Регулярные выражения (упрощенный вариант)

Опертор LIKE нужен для поиска строк:

> SELECT * FROM Products WHERE pname like '%подстрока%'

- % - сколько угодно символов
- _ ровно один символ
- оператор LIKE - регистрозависимый

```
%%sql
SELECT * FROM product
where category LIKE '%ка%';
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Клавиатура	949.99	Техника	Sven

```
%%sql
SELECT * FROM product
where pname LIKE '_a%';
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Батарейка	39.99	Для дома	Krona
Лампочка	89.7	Для дома	Energolux

Убрать дубли

Дубли не всегда хорошо, и иногда их стоит убирать

- Помните, что таблицы - *мультимножества*!

```
%%sql
SELECT category from product;
```

```
 * sqlite://
Done.
```

category
Канцелярия
Техника
Канцелярия
Для дома
Для дома
None

```
%%sql
-- чтобы убрать дубли используйте слово DISTINCT
SELECT DISTINCT category from product;
```

```
 * sqlite://
Done.
```

category
Канцелярия
Техника
Для дома
None

```
%%sql
-- чтобы убрать дубли используйте слово DISTINCT
SELECT DISTINCT category, price from product;
```

```
 * sqlite://
Done.
```

category	price
Канцелярия	39.99
Техника	949.99
Канцелярия	129.99
Для дома	39.99
Для дома	89.7
None	0

Сортировка результатов

Так как таблица в SQL - это мультимножество строк, то порядок вывода строк не гарантирован. Иногда необходимо выводить строки в определенном порядке

```
%%sql
-- сортировка результатов
-- сортировка по умолчанию - ascending
SELECT   pname, price, manufacturer
FROM     Product
WHERE    price < 400
ORDER BY price, manufacturer;
```

```
 * sqlite://
Done.
```

pname	price	manufacturer
Веб-камера	0	Sven
Батарейка	39.99	Krona
Тетрадь	39.99	Академия холдинг
Лампочка	89.7	Energolux
Степлер	129.99	Brauberg

```
%%sql
-- сортировка результатов
-- тип сортировки каждого компонента определяется индивидуально
SELECT   price, manufacturer
```

```
5 FROM     Product
6 ORDER BY manufacturer ASC, price DESC;
```

```
 * sqlite://
Done.
```

price	manufacturer
129.99	Brauberg
89.7	Energolux
39.99	Krona
949.99	Sven
0	Sven
39.99	Академия холдинг

Можно делать сортировку по порядковому номеру, но довольно часто это считается bad practice

```
1 %%sql
2 SELECT   price, manufacturer
3 FROM     Product
4 ORDER BY 2 ASC, 1 DESC;
```

```
 * sqlite://
Done.
```

price	manufacturer
129.99	Brauberg
89.7	Energolux
39.99	Krona
949.99	Sven
0	Sven
39.99	Академия холдинг

Ограничение и смещение

Вывести топ 3 товара по дороговизне

```
1 %%sql
2 SELECT * FROM Product
3 ORDER BY price DESC
4 LIMIT 3;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Клавиатура	949.99	Техника	Sven
Степлер	129.99	Канцелярия	Brauberg
Лампочка	89.7	Для дома	Energolux

Вывести 2, 3 и 4 товары по алфавиту

```
%%sql
SELECT * FROM Product
ORDER BY pname
LIMIT 3 OFFSET 1;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Веб-камера	0	None	Sven
Клавиатура	949.99	Техника	Sven
Лампочка	89.7	Для дома	Energolux

Группировка

Можно осуществлять схлопывание строк в рамках каких-нибудь групп

После схлопывания к группам можно применять агрегатные функции

AVG(<поле>)

SUM(<поле>)

MIN(<поле>)

MAX(<поле>)

COUNT(<поле>)

COUNT(*)

AVG(distinct <поле>)

SUM(distinct <поле>)

COUNT(distinct <поле>)

```
%%sql
select category, avg(price), min(price), max(price), sum(price)
from Product
group by category;
```

```
 * sqlite://
Done.
```

category	avg(price)	min(price)	max(price)	sum(price)
None	0.0	0	0	0
Для дома	64.845	39.99	89.7	129.69
Канцелярия	84.99000000000001	39.99	129.99	169.98000000000002
Техника	949.99	949.99	949.99	949.99

Существует возможность фильтровать не только по строкам, но и по группам. Для этого можно использовать инструкцию HAVING

```
%%sql
select category, avg(price), min(price), max(price), count(price)
```

```
  3 from Product
  4 group by category
  5 having min(price) > 100;
```

```
 * sqlite://
Done.
```

category	avg(price)	min(price)	max(price)	count(price)
Техника	949.99	949.99	949.99	1

```
  1 %%sql
  2 select category, count(*) as star, count(category) as cat, count(distinct category) as dist
  3 from Product
  4 group by category;
```

```
 * sqlite://
Done.
```

category	star	cat	dist
None	1	0	0
Для дома	2	2	1
Канцелярия	2	2	1
Техника	1	1	1

count(*) - кол-во строк

count(<поле>) - кол-во непустых строк

count(distinct <поле>) - кол-во уникальных непустых строк

Работа с NULL

```
  1 %%sql
  2 select *
  3 from Product
  4 where category != NULL;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer

```
  1 %%sql
  2 select *
  3 from Product
  4 where category = NULL;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer

```
  1 %%sql
  2 select *
```

```
  3 from Product
  4 where category is NULL;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Веб-камера	0	None	Sven

```
  1 %%sql
  2 select *
  3 from Product
  4 where category is not NULL;
```

```
 * sqlite://
Done.
```

pname	price	category	manufacturer
Тетрадь	39.99	Канцелярия	Академия холдинг
Клавиатура	949.99	Техника	Sven
Степлер	129.99	Канцелярия	Brauberg
Батарейка	39.99	Для дома	Krona
Лампочка	89.7	Для дома	Energolux

```
  1 %%sql
  2 drop table if exists numbers;
  3
  4 create table numbers(a int, b int);
  5
  6 insert into numbers
  7   values (1,2), (2,3), (4,null), (null,null), (null,5);
```

Показать скрытые выходные данные

```
  1 %%sql
  2 select * from numbers;
```

```
 * sqlite://
Done.
```

a	b
1	2
2	3
4	None
None	None
None	5

```
  1 %%sql
  2 select sum(a+b) as sum1,
  3        sum(a) + sum(b) as sum2
  4        from numbers;
```

```
 * sqlite://
```

```
%%sql
select a || '+' || b as concat
       from numbers;
```

```
 * sqlite://
Done.
```

concat
1+2
2+3
None
None
None

```
%%sql
select *
  from numbers
where a>0;
```

```
 * sqlite://
Done.
```

a	b
1	2
2	3
4	None