

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт цифрового развития

Кафедра инфокоммуникаций.

Дисциплина: Кроссплатформенное программирование

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Основы SQLite в Pandas

Выполнила: студентка 3 курса
09.03.01 «Информатика и вычислительная
техника» группы ИВТ-б-о-19-1
Бондаренко В.В.

Проверил:
Воронкин Роман Александрович

Работа защищена с оценкой:

Ставрополь, 2021г

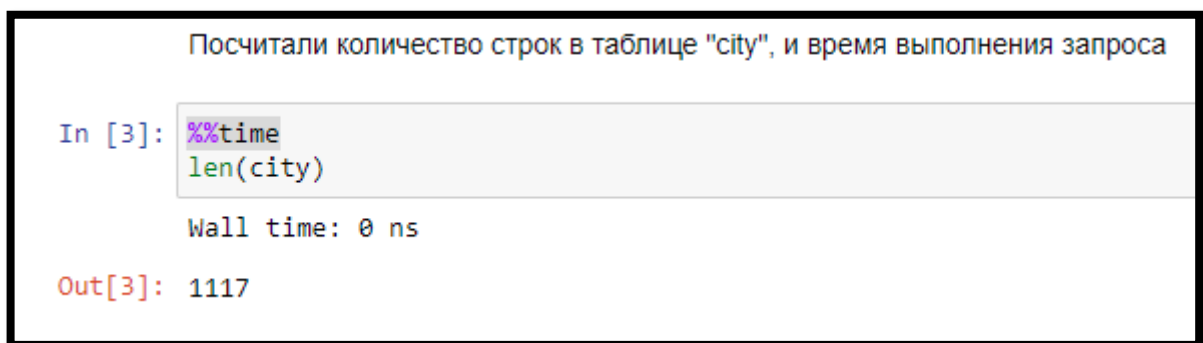
Лабораторная работа №1

«Основы SQLite».

Цель работы: исследовать базовые возможности системы управления базами данных SQLite.

Выполнение:

1. Использовали команду “%%time”, которая отвечает за вывод времени выполнения запроса. Включили её и в результатах запроса добавилась строчка. (Рисунок 1).



```
Посчитали количество строк в таблице "city", и время выполнения запроса

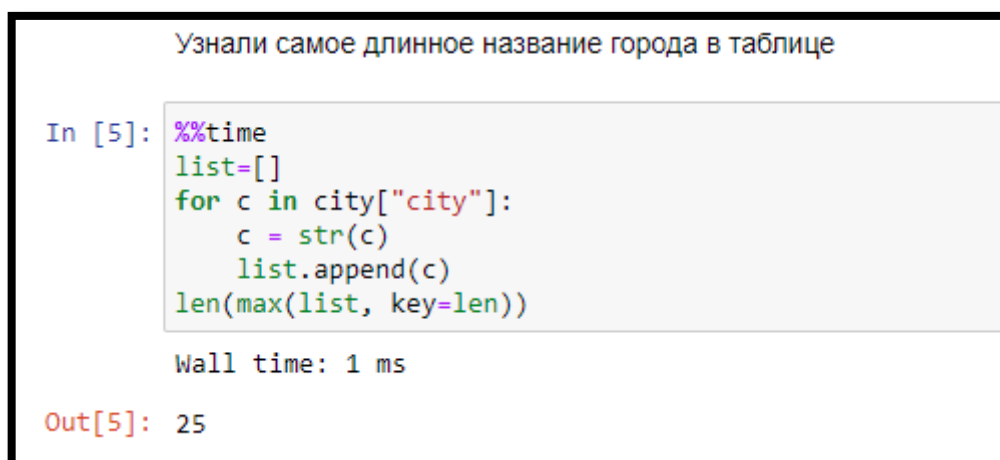
In [3]: %%time
        len(city)

        Wall time: 0 ns

Out[3]: 1117
```

Рисунок 1 – Применение команды

2. Затем, загрузили файл city.csv в Jupiter Notebook. Импортировали его, а затем выполнили запрос аналогичный этому (при помощи Pandas): `select max(length(city)) from city;` После выполнения этого запроса, вывелось число: 25 (Рисунок 2).



```
Узнали самое длинное название города в таблице

In [5]: %%time
        list=[]
        for c in city["city"]:
            c = str(c)
            list.append(c)
        len(max(list, key=len))

        Wall time: 1 ms

Out[5]: 25
```

Рисунок 2 – Выполнение запроса

3. Написали в Jupiter Notebook запрос, который посчитал кол-во городов для каждого часового пояса в Сибирском и Приволжском федеральных округах. Значение timezone (UTC +5) для city_count = 58 (Рисунок 3)

```
In [22]: k = city[city['federal_district'].isin(['Приволжский', 'Сибирский'])]
k[['timezone', 'address']].groupby('timezone').count().rename({'address': 'city_count'}, axis=1)

Out[22]:
```

timezone	count
UTC+3	101
UTC+4	41
UTC+5	58
UTC+6	6
UTC+7	86
UTC+8	22

Рисунок 3 – Выполнение задания

4. Написали в песочнице запрос, который посчитал кол-во городов в каждом часовом поясе. Отсортировали по кол-ву городов по убыванию. (Рисунок 4)

```
In [55]: k = city[['timezone', 'address']].groupby('timezone').count().sort_values('address', ascending=False)
k.rename({'address': 'city_count'}, axis=1)

Out[55]:
```

timezone	city_count
UTC+3	660
UTC+5	173
UTC+7	86
UTC+4	66
UTC+9	31
UTC+8	28
UTC+10	22
UTC+2	22
UTC+11	17
UTC+12	6
UTC+6	6

Рисунок 4 – Выполнение 1-ого задания

5. Затем, загрузили в SQLite выбранный нами датасет в формате CSV. Сформировали более пяти запросов к таблицам БД. Выгрузили результат выполнения запросов в формате CSV и JSON.

```
B [11]: audi['model'].value_counts()
Out[11]:
A3      1929
Q3      1417
A4      1381
A1      1347
A5       882
Q5       877
Q2       822
A6       748
Q7       397
TT       336
A7       122
A8       118
Q8        69
RS6        39
RS3        33
RS4        31
RS5        29
R8         28
S3         18
SQ5        16
S4         12
SQ7         8
S8          4
S5          3
A2          1
RS7          1
Name: model, dtype: int64
```

Рисунок 5 – Первый запрос

```
Сумма продаж каждой модели

B [20]: audi[['model', 'price']].groupby('model').sum().sort_values('price', ascending=False)
Out[20]:
```

	price
A3	33581039
Q3	32589954
A4	27972777
Q5	26700869
A5	20795015
A1	19299480
Q2	18508954
Q7	17780963
A6	16976148
TT	7319576
Q8	4147936
A8	4127858
A7	3521593
R8	2734262
RS6	2182591
RS4	1554700
RS5	1486691
RS3	1123667
SQ5	502653
SQ7	394152
S4	374977

Рисунок 6 – Второй запрос

```
Типы топлива

B [21]: audi['fuelType'].value_counts()

Out[21]: Diesel      5577
         Petrol      5063
         Hybrid       28
         Name: fuelType, dtype: int64
```

Рисунок 7 – Третий запрос

```
5 самых дешевых машин

B [25]: audi[['model', 'year', 'price']].sort_values('price').head(5)

Out[25]:
```

	model	year	price
10588	A3	2003	1490
10552	A4	2004	1699
7795	A3	2005	1975
10108	TT	2002	1990
7404	A3	2009	2490

Рисунок 8 – Четвертый запрос

```
5 машин с наименьшим расходом

B [26]: audi[['model', 'year', 'mpg']].sort_values('mpg').head(5)

Out[26]:
```

	model	year	mpg
10427	A8	1997	18.9
8941	A6	2004	19.3
1869	R8	2013	19.6
10171	RS4	2006	20.3
9829	S4	2007	20.3

Рисунок 9 – Пятый запрос

Вывод: Таким образом, после выполнения данной лабораторной работы, исследовали базовые возможности системы управления базами данных SQLite при помощи Pandas.

Ответы на контрольные вопросы:

1. Какие существуют средства для импорта данных в SQLite?

Команда `.import` автоматически создала таблицу `city` со всеми столбцами из `city.csv` и загрузила данные из файла.

2. В чем недостатки локальных и централизованных СКВ?

Локальные СКВ:

Недостатки:

- возможность потери данных вследствие возникновения физических поломок оборудования;
- отсутствие возможности совместной разработки.

Централизованные СКВ:

Недостатки:

- отсутствие доступа к данным при сбое работы сервера;
- довольно низкая скорость работы (из-за возникновения сетевых задержек).

3. Каково назначение команды `.schema` ?

`.schema` – это специальная команда SQLite, не часть стандарта SQL. Она показывает шаблон сопоставления инструкций CREATE.

4. Как выполняется группировка и сортировка данных в запросах SQLite?

`groupby 1` и `orderby 2` — это короткое обращение к столбцам из `select` по порядковому номеру.

5. Каково назначение "табличных выражений" в SQLite?

Выражение `withhistoryas (...)` создает именованный запрос. Название — `history`, а содержание — селект в скобках (век основания для каждого

города). К history можно обращаться по имени в остальном запросе, что мы и делаем.

Строго говоря, селект в блоке with называют «табличным выражением» (common table expression, CTE). Так что, если встретите в документации — не удивляйтесь. Запомните главное: это обычный селект, к которому можно для краткости обращаться по имени, как к таблице.

6. Как осуществляется экспорт данных из SQLite в форматы CSV и JSON?

```
.mode csv
```

```
.mode json
```

7. Какие еще форматы для экспорта данных Вам известны?

Всякий раз, когда вы экспортируете в файл или буфер обмена, используется определенный формат экспорта. Часто это CSV, но можно использовать и другие, например: JSON, XML, HTML и Markdown.