Пояснения к заданию: в ходе выполнения задания нужно будет написать код + создать специальные файлы с ответами res1.txt res2.txt и тд. которые будут содержать одно или несколько чисел - ответов на задания. После выполнения задания нужно отправить код решающий поставленные задачи (можно в одном текстовом файле jupyter notebook - е или как-нибудь еще) + видоизмененную базу данных DB.db + приложить файлы res1.txt, res2.txt и тд.

**Задание 1**

[DB.db](https://drive.google.com/file/d/1SpjA39youfcWhIg24yPstg6cpsfL5nfp/view?usp=sharing)

В базе "DB.db" (SQLite) есть таблица изменения цен на товары “prices”. В таблице три колонки: product - название продукта, price - цена продукта, date\_start - дата, начиная с которой соответствующая цена на товар вступает в силу:

|  |  |  |
| --- | --- | --- |
| **date\_start** | **product** | **price** |
| ‘1970-01-01’ | A | 20 |
| ‘2019-02-04’ | B | 45 |
| ‘2018-05-11’ | A | 27 |
| ... | ... | ... |

\*Важно понимать, что столбец date\_start содержит только те даты, когда цена на товар менялась.

* Напишите запрос, возвращающий строки с актуальными ценами (с наибольшей датой - date\_start) для каждого товара (product). Посчитайте среднюю цену и запишите ответ-число в файл **res1.txt**
* Напишите функцию на любом языке программирования (предпочтительно Python) принимающую название продукта (поле product) и возвращающую tuple вида ([date\_start1, date\_start2,... ],[price1, price2,...]), из таблицы “prices”. Где date\_start(n) - это даты, отсортированные по возрастанию, а price(n) - соответствующая дате цена на продукт. При каждом вызове функция должна обращаться к БД, выбирая необходимые данные. Формат дат особого значения не имеет.

Пример работы функции:

>>>my\_func(‘A’)

>>>([‘2018-05-11’,’2019-01-01’, … ], [27, 20, … ])

**Задание 2**

Еще есть таблица “sales” в нашей базе.

|  |  |  |
| --- | --- | --- |
| **product** | **date** | **amount** |
| A | ‘2019-02-15’ | 92 |
| B | ‘2019-05-06’ | 113 |
| B | ‘2019-06-20’ | 12 |
| ... | ... | ... |

Каждая строка в такой таблице характеризует продажи (amount) определенного товара за определенную дату.

Напишите запрос, результатом которого станет создание таблицы “revenue” вида:

|  |  |  |  |
| --- | --- | --- | --- |
| **product** | **date** | **amount** | **revenue** |
| ... | ... | ... | ... |

Где revenue вычисляется по формуле: revenue = sales.amount \* prices.price. Естественно, price из таблицы prices должен соответствовать date из таблицы sales. Следует учитывать, что таблица sales содержит большое кол-во строк.

Выполните запрос и создайте таблицу в revenue "DB.db".

Запишите величину выборочного стандартного отклонения revenue для продукта "B" в файл res2.txt.

**Задание 3**

Напишите функцию, которая принимает название товара product и две даты: date\_start,

date\_end, тренирует (находит c МНК коэффициенты k, b) линейную модель вида y = kx+b и возвращает коэффициенты, k и b.

В модели **x** - это номер дня от начала заданного периода (0,1,2,3...), а **y** - кол-во проданных единиц продукции (amount из таблицы sales или revenue). При каждом вызове функция должна обращаться к таблице “revenue” или "sales" из задания 2, выбирая необходимые данные.

**Запишите коэффициенты, возвращаемые функцией при начальных аргументах: lin\_reg(product='Q', date\_start='2019-01-01', date\_end='2019-03-31') в res3.txt через пробел.**