Привет!

Рады видеть тебя на втором этапе.
Мы стали чуть ближе друг к другу, и теперь нам хочется проверить твои аналитические способности, мышление и узнать больше про боевой опыт.

Описание: у тебя есть 100 минут для прохождения теста из 5 заданий. Постарайся решить как можно больше заданий за отведенное время.

Рекомендация: вначале прочти внимательно все задачи и начни с наиболее простых, далее можно приступить к оставшимся.

Также постарайся поменьше гуглить, т.к. это отнимает драгоценное время =)
По истечении отведенного времени отправь решение в удобном для тебя виде.
Например: **file.py** или **file.ipynb**.

Желаем тебе удачи!

Логика

1. Есть игровой автомат. При нажатии на кнопку он показывает игроку случайное число из равномерного распределения на интервале [0, 1].

Какое число показал автомат – столько миллионов рублей выигрывает игрок.

У игрока есть два возможных действия: он может забрать деньги сразу после первого нажатия кнопки, видя результат, или же сыграть еще раз и забрать деньги после второй попытки.

Задача:

- Какая стратегия гарантирует игроку максимальный выигрыш?
- Найти мат. ожидание выигрыша этой стратегии.

2. Алиса и Боб играют в игру: Алиса записывает число (1 или 0), а Боб пытается назвать записанное число. Из 15 попыток он 14 раз назвал число правильно.

Задача:

- Проверить гипотезу H_0, что Боб выбирает число случайно.

SQL

1. Какой запрос вернет самое большое число? Почему?

Для справки: (A right join B) эквивалентно (B left join A).

A.

```
select count(distinct first.counter_column)
from first
left join second on first.join_key = second.join_key
where second.filter_column >= 42;
```

B.

```
select count(distinct first.counter_column)
from first
left join second on first.join_key = second.join_key and second.filter_column >= 42;
```

C.

```
select count(distinct first.counter_column)
from first
right join second on first.join_key = second.join_key
where second.filter_column >= 42;
```

D.

```
select count(distinct first.counter_column)
from first
right join second on first.join_key = second.join_key and second.filter_column >= 42;
```

2. Для этого задания нам потребуются несколько дополнительных таблиц**:**

- **ClientOrder** – информация о заказах

Название столбца	**Тип данных**	**Описание**
ID	bigint	ID заказа
Date	datetime	Время оформления заказа
ClientID	bigint	ID пользователя
ClientOrderStateID	int	ID статуса заказа (1 – оформлен, 2 – получен, 3 – отменен)

- **ClientOrderItems** – информация о товарах в заказе

Название столбца	**Тип данных**	**Описание**
ClientOrderID	bigint	ID заказа
ItemID	bigint	ID товара
categoryLvl1	varchar	Коммерческая категория 1-го уровня
price	float	Цена за одну единицу товара
qty	int	Единиц товара в заказе

- **ClientOrderAdditionalInfo** – дополнительные атрибуты заказа

Название столбца	**Тип данных**	**Описание**
ClientOrderID	bigint	ID заказа
code	varchar	Название атрибута (OrderType – тип заказа, Platform – платформа, с которой совершен заказ)
value	varchar	Значение атрибута (OrderType – test или regular, Platform – ios, android, site или mobile)

Задача:

1. Найти среднее время между первым и вторым заказом у пользователей. В запросе запрещается использование JOIN’ов. Тестовые заказы фильтровать не нужно.

2. Для каждой пары платформы и категории товара найти топ-3 пользователей, у которых наименьшее количество дней между первым и последним не тестовым заказом товаров из этой категории.

Python

Для выполнения этого задания требуется сгенерировать DataFrame с синтетическими данными.

DataFrame должен состоять из 10000 строк и 5 колонок.

Каждую из колонок мы предлагаем тебе создать и наполнить следующим образом:

1-я колонка – **user_id** – идентификатор пользователя. Длина user_id должна равняться 15-ти символам. Идентификатор состоит из случайной комбинации следующих символов: "1234567890abcdefghijk". Для каждой строки в DataFrame значение user_id формируются случайным образом.

2-я колонка – **order_number** – номер заказа. Столбец необходимо заполнить случайными значениями в диапазоне от 1 до 10.

3-я колонка – **click2delivery** – время, прошедшее с момента оформления заказа до вручения клиенту. Столбец необходимо заполнить случайными значениями из нормального распределения со средним 1440 и стандартным отклонением 200.

4-я колонка – **order_items_sum** – общая стоимость заказа. Значения для этого столбца необходимо взять из экспоненциального распределения с параметром λ = 1, смещённого на +1.

5-я колонка – **retention** – день жизни покупателя, в который он совершил заказ. Необходимо сгенерировать значения 1, 2, 3, 4, 5 с вероятностями 0.35, 0.25, 0.2, 0.15 и 0.05 соответственно.

В случае, если в колонке **user_id** встречаются дублирующиеся значения, оставь только первое из них.

Задача:

- ☑ Для всех строк исходного датасета, сгруппированных по номеру заказа, посчитать среднее значение времени доставки по группе. Результат необходимо добавить в новый столбец датафрейма.
- ☑ Отдельной колонкой добавить значения последовательности, начинающейся с 0,1, где каждый следующий элемент является суммой двух предыдущих, умноженных на 0.5
- ☑ Напиши функцию, которая принимает на вход значение **user_id** и возвращает строку следующего вида: все буквы в той последовательности, в которой они встречаются в **user_id**, затем квадрат числа, полученного из всех цифр в **user_id** в той последовательности, в которой они встречаются в **user_id**.
- ☑ Добавь результат применения этой функции к **user_id** в новый столбец. Использование циклов вне функции **запрещено**.
- ☑ Вычисли моду, медиану, среднее, дисперсию и стандартное отклонение для столбцов **click2delivery**, **order_items_sum** и **retention**. Построй гистограммы распределения значений в столбцах
- ☑ Построй график, который наиболее полно, на твой взгляд, описывает зависимость времени доставки заказа от его номера. Обоснуй, почему ты так считаешь?