# Задача 1.1

(дані Jupyter-lab в папка Task1\_2\_Python, файл TestTask1\_LTV.ipynb)

На початку роботи очистила дата сет з покупками клієнтів від даних з media**\_**source, яких не було в файлі витрат. Оскільки планувала розраховувати ROMI на основі прогнозованих даних LTV, тому й прогнозувати потрібно було поведінку лише тих клієнтів, що прийшли з рекламних акцій.

Також видалили дублі транзакцій покупок (вважала дублями лише повністю ідентичні строчки по всім колонкам), та очистила дані від строчок зі значенням refunded = True (оскільки з таких транзакцій компанія грошей не отримала).

Додала да дата фрейму дані суми по product\_id та trial.

Далі розділила всіх користувачів на потижневі когорти. Всього 13 когорт від 0-го до 12-го тижнів.

Потім розрахувала середню накопичену суму покупок по клієнту з кожної когорти у розрізі тижня покупки (тиждень покупки для кожного клієнта рахувала як порядковий номер тижня з першої покупки клієнта – іншими словами, розраховувала суму покупок на певний тиждень життя клієнта у компанії).

Щоб звести дані між когортами в одну точку перевела LTV у відсоток, перший тиждень = 100%.

Усереднила дані між когортами та на основі цього ряду даних розрахувала параметри функції прогнозу (як прогнозну функцію використовувала фунцію від натурального логарифму f(x) = a \* ln(x+b) \* c + d).

На основі розрахованих параметрів [a, b, c, d] описала функцію розрахунку прогнозованого LTV - forecast\_LTV.   
Функція отримує на вхід номер когорти, номер тижня прогнозу. Повертає реальне значення LTV, якщо ці дані в нас є, або прогнозоване значення LTV на основі розрахованих параметрів.

Далі для перевірки розрахувала дані по всім когортам до 1 тижня та вивела на графік.

# Задача 1.2

(дані Jupyter-lab в папка Task1\_2\_Python, файл TestTask1\_LTV.ipynb та Book\_ROMI.twbx)

Код знаходиться після розрахунку прогнозу по когортам.

ROMI планувала розраховувати за формулою (відсоток чистого доходу відносно до затрат):

Для завдання 1.2 готувала спеціальний файл з розрізами:

* 'media\_source';
* 'country\_code';
* 'date';
* 'cost' - накопичена сума затрат на маркетинг у розрізі 'media\_source', 'country\_code';
* 'LTV' - накопичена сума покупок клієнтів у розрізі 'media\_source', 'country\_code'.

В даних затрати по Google Ads не розділені по країнам, для розрахунку ROMI всі дані доходів по відомим країнам можемо відкинути (оскільки ми не знаємо на які затрати мають лягти ці доходи), або звести всі дані доходів по країнам в одне значення Organic/Unknown, аналогічно як у затратах. В другому випадку ми отримаємо зведене ROMI по всьому 'media\_source' = Google Ads, та не матимемо деталізації по кожній країні.

Якщо обрати перший варіант, то ROMI по країнам ми не зможемо розрахувати, отже, ці дані доходів потрібно просто відкинути. В такому випадку ми отримаємо ROMI по Google Ads завжди = -100%.

# Задача 2

(дані Jupyter-lab в папка Task1\_2\_Python, файл TestTask2.ipynb та Task2.sql)

Для завдання генерувала дані у Jupyter-lab (файл TestTask2.ipynb). Після генерації заливала ці дані одразу з Jupyter-lab у свою тестову БД PostgreSQL testtask.

Task2.sql – скрипт розділений на декілька частин:

* **Median without Aggregates** – розрахунок медіани;  
  містить опис створення/оновлення функції (median\_from\_arr\_real) та запит з використанням цієї функції.  
  На вхід функція median\_from\_arr\_real отримує масив real (real – тому що з таким типом Jupyter-lab вивантажив мені дані у колонку spend).
* **Median with Aggregates** – розрахунок медіани (варіант 2);  
  містить опис створення агрегату (AGGREGATE) median\_real, який в свою чергу використовує попередню функцію median\_from\_arr\_real та запит з використанням цього агрегату.
* **Mode** – запит на розрахунок моди вибірки;
* **Control sample** – запит на розрахунок Моди та Медіани стандартними функціями PostgreSQL (для перевірки).