Тестове завдання для Data Analyst

Виконав: Олександр Мучак

Завдання 1:

Дані:

https://docs.google.com/spreadsheets/d/1kVu76PhHKNY__sJDza6w3KuBfxs9r1zX3261jDxXdEs/edit?usp=sharing

Платіжна воронка

Мій підхід

1. Створюю CTE для визначення всіх транзакцій, за якими було повернення.
2. У другому CTE використовую віконні функції FIRST_VALUE та ROW_NUMBER для знаходження оригінальної транзакції та номера поновлення для кожної підписки користувача.
3. У фінальному запиті об'єдную дані та коригую дохід за допомогою CASE.

-- Завдання 1: **Платіжна воронка**
-- Автор: Олександр
-- Дата: 12.09.2025

SQL-запит

```
WITH
-- Крок 1: Створюємо список всіх транзакцій, за якими було повернення коштів
refunded_transactions AS (
    SELECT DISTINCT
        refunded_transaction_id
    FROM
        subscription_events
    WHERE
        refunded_transaction_id IS NOT NULL
),

-- Крок 2: Обираємо всі платіжні події та розраховуємо номер поновлення
numbered_purchases AS (
```

```
    SELECT
        uuid,
        product_id,
        transaction_id,
        event_timestamp,
        revenue_usd,
        FIRST_VALUE(transaction_id) OVER (PARTITION BY uuid, product_id ORDER BY
event_timestamp) AS original_transaction_id,
        ROW_NUMBER() OVER (PARTITION BY uuid, product_id ORDER BY
event_timestamp) AS renewal_number
    FROM
        subscription_events
    WHERE
        event_name IN ('purchase', 'trial') AND transaction_id IS NOT NULL
)

-- Крок 3: Об'єднуємо дані та коригуємо дохід
SELECT
    np.uuid,
    np.product_id,
    np.transaction_id,
    np.original_transaction_id,
    CASE
        WHEN rt.refunded_transaction_id IS NOT NULL THEN 0
        ELSE np.revenue_usd
    END AS revenue_usd,
    np.renewal_number
FROM
    numbered_purchases np
LEFT JOIN
    refunded_transactions rt ON np.transaction_id = rt.refunded_transaction_id
ORDER BY
    np.uuid,
    np.product_id,
    np.renewal_number;
```

Покупки користувача

Мій підхід

Створюю зведений звіт для кожного унікального користувача (`uuid`), агрегуючи всю історію його подій в один рядок.

Для вирішення я знову використав CTE:

- Підготовка даних: Спочатку я створив кілька допоміжних CTE.
    - Перша (`refunded_ids`) — для визначення всіх транзакцій, за якими було повернення. Це необхідно для коректного розрахунку загального доходу.
    - Друга (`last_purchase_info`) — для знаходження дати та періоду *останньої покупки* кожного користувача. Ця інформація є ключовою для обчислення дати закінчення підписки (`expiration_time`).
    - Третя (`current_product_info`) — для визначення *поточного продукту*, базуючись на останній за часом події користувача.
- Фінальна агрегація: В основному запиті я згрупував усі дані за `uuid`.
    - Для знаходження дат (початок тріалу, перша покупка, скасування) я застосував умовну агрегацію з `MIN` та `MAX` у поєднанні з `CASE WHEN`.
    - Загальний дохід (`total_revenue_usd`) розраховано як суму всіх платежів, ID яких *не* потрапили до списку повернутих.
    - Підготовлені дані з CTE були приєднані через `LEFT JOIN`, щоб додати до фінального звіту `current_product_id` та розрахувати `expiration_time`.

-- Завдання 2: **Агрегована інформація по покупках кожного користувача**
-- Автор: Олександр
-- Дата: 12.09.2025

SQL-запит

```
WITH
-- 1. Знаходимо ID всіх транзакцій, за якими було повернення
refunded_ids AS (
    SELECT DISTINCT
      refunded_transaction_id
    FROM
      subscription_events
    WHERE
      refunded_transaction_id IS NOT NULL
),
```

```
-- 2. Для кожного користувача знаходимо час та період його останньої покупки
last_purchase_info AS (
    SELECT
        uuid,
        event_timestamp AS last_purchase_time,
        period
    FROM (
        SELECT
            uuid,
            event_timestamp,
            period,
            ROW_NUMBER() OVER(PARTITION BY uuid ORDER BY event_timestamp DESC)
as rn
        FROM
            subscription_events
        WHERE
            event_name = 'purchase'
    ) AS ranked_purchases
    WHERE
      rn = 1
),

-- 3. Знаходимо поточний product_id, базуючись на останній події користувача
current_product_info AS (
    SELECT
        uuid,
        product_id AS current_product_id
    FROM (
        SELECT
            uuid,
            product_id,
            ROW_NUMBER() OVER(PARTITION BY uuid ORDER BY event_timestamp DESC)
as rn
        FROM
            subscription_events
    ) AS ranked_events
    WHERE
      rn = 1
)

-- 4. Агрегуємо всі дані, групуючи по користувачу (uuid)
SELECT
    s.uuid,
    cpi.current_product_id,
    MIN(CASE WHEN s.event_name = 'trial' THEN s.event_timestamp END) AS
trial_started_time,
```

```
    MIN(CASE WHEN s.event_name = 'purchase' THEN s.event_timestamp END) AS
first_purchase_time,
    MAX(CASE WHEN s.event_name = 'purchase' THEN s.event_timestamp END) AS
last_purchase_time,
    COUNT(DISTINCT CASE WHEN s.event_name = 'purchase' THEN s.transaction_id
END) AS total_purchases,
    SUM(
        CASE
            WHEN s.event_name IN ('purchase', 'trial')
            AND s.transaction_id NOT IN (SELECT refunded_transaction_id FROM
refunded_ids)
            THEN s.revenue_usd
            ELSE 0
        END
    ) AS total_revenue_usd,
    -- Додаємо кількість днів (period) до дати останньої покупки
    lpi.last_purchase_time + (lpi.period || ' days')::interval AS expiration_time,
    MAX(CASE WHEN s.event_name = 'cancellation' THEN s.event_timestamp END) AS
cancelation_time,
    MAX(CASE WHEN s.event_name = 'refund' THEN s.event_timestamp END) AS
refund_time
FROM
    subscription_events s
-- Приєднуємо підготовлені дані
LEFT JOIN
    last_purchase_info lpi ON s.uuid = lpi.uuid
LEFT JOIN
    current_product_info cpi ON s.uuid = cpi.uuid
GROUP BY
    s.uuid,
    cpi.current_product_id,
    lpi.last_purchase_time,
    lpi.period;
```

Завдання 2:

User LTV

Дані:

https://drive.google.com/file/d/1GorPiMNdyqAV_jplWVQhEfgQ49UVeRZG/view?usp=sharing

Розрахунок:

https://colab.research.google.com/drive/1ckNn3XyAW7fJVFT3f30BTBBr7-c-clPo?usp=sharing

Що було зроблено:

Для розрахунку та прогнозування LTV було застосовано **когортний аналіз** з подальшою **математичною екстраполяцією**.

- **Когорти:** Користувачі були згруповані в місячні когорти за датою їхнього **першого платежу**.
- **Фактичний LTV:** Ми розрахували **середній накопичений дохід (кумулятивний LTV)** для кожної когорти на кожен місяць її життя, врахувавши повернення коштів.
- **Прогноз LTV:** Існуюча крива LTV для кожної когорти була продовжена на 6 місяців уперед за допомогою **апроксимації логарифмічною функцією**, яка добре моделює уповільнення росту доходу з часом.

2. Оцінка якості розрахунку

Обраний метод є швидким та наочним, але має свої обмеження.

- **Сильні сторони:** *Простота, швидкість реалізації та легкість в інтерпретації результатів.*
- **Слабкі сторони:** *Низька точність на рівні окремого користувача, чутливість до аномалій та неможливість пояснити причини змін у поведінці.*

3. Рекомендації щодо покращення

Для підвищення точності прогнозу були запропоновані три основні напрямки:

- **Імовірнісні моделі:** Використання індустріального стандарту — моделей **BG/NBD** (для прогнозування кількості транзакцій) та **Gamma-Gamma** (для прогнозування їхньої вартості).
- **Машинне навчання:** Побудова **ML-моделей** (наприклад, Gradient Boosting) на основі характеристик користувачів (тип першої підписки, країна тощо) для створення персоналізованих прогнозів.
- **Поглиблена сегментація:** Побудова окремих моделей для різних сегментів користувачів, які поводяться по-різному.

Завдання 3:

Пошук точок зростання

Дані:

https://drive.google.com/file/d/1BMVxVlkLuwHQG4imW8BCOoVOQ9BG17S5/view?usp=sharing

Розрахунок:

https://colab.research.google.com/drive/1oAMJ0g7yjn3XDgI73DdoPgDVwj-HAvus?usp=sharing

Що було зроблено:

- **Проаналізовано перформанс:** Ми розрахували ключові показники **CPA** (вартість залучення) та **ROAS** (окупність витрат) для кожного медіа-джерела та кожної країни.
- **Результати в таблицях:** Ми створили зведені таблиці для аналізу загальної ефективності та, найголовніше, списки конкретних кампаній для оптимізації.
- **Візуалізації:** Ми побудували графік, який наочно продемонстрував аномальну ефективність `ms_3` та порівняв `ms_1` і `ms_2`.
- **Пропозиції щодо закупівлі:** Ми сформулювали чіткі, засновані на даних рекомендації:
    - **Зупинити :** *Виявлено критично неефективні кампанії (особливо в **США**), які генерують величезні збитки.*
    - **Масштабувати :** *Знайдено кампанії-лідери (в **Японії, на Філіппінах** та ін.), які мають високий потенціал для зростання при обережному збільшенні бюджету.*