Агрегатные функции

1.1 Рассчитайте общий доход от всех операций.
SELECT SUM(price) FROM transactions;

1.2 Найдите средний доход с одной сделки.
SELECT AVG(price) FROM transactions;

1.3 Определите общее количество проданной продукции.
SELECT COUNT() FROM transactions;

1.4 Подсчитайте количество уникальных пользователей, совершивших покупку
SELECT COUNT(DISTINCT user_id) FROM transactions;

Функции для работы с типами данных
2.1. Преобразуйте `transaction_date` в строку формата `YYYY-MM-DD`.
SELECT toYYYYMMDD(transaction_date) FROM transactions;

2.2 Извлеките год и месяц из `transaction_date`
SELECT toYear(transaction_date) FROM transactions;

2.3. Округлите `price` до ближайшего целого числа.
SELECT floor(price) FROM transactions;

2.4. Преобразуйте `transaction_id` в строку.
SELECT toString(transaction_id) FROM transactions;

User-Defined Functions (UDFs)

3.1. Создайте простую UDF для расчета общей стоимости транзакции.

```
* Maximum number of threads is lower than 30000. There could be problems with handling a
 lot of simultaneous queries.

envy :) CREATE FUNCTION transaction_sum AS (a,b) -> a*b;

CREATE FUNCTION transaction_sum AS (a, b) -> (a * b)

Query id: 8da80bcb-cafe-4d1f-9732-b9180f7eafbd

Ok.

0 rows in set. Elapsed: 0.060 sec.

envy :) select transaction_sum(quantity, price) from transactions;

SELECT transaction_sum(quantity, price)
FROM transactions

Query id: d32e661a-e502-4e81-9fde-146d80ed3c64

Ok.
```

3.2. Используйте созданную UDF для расчета общей цены для каждой транзакции.

```
envy :) select transaction_id, sum(transaction_sum(quantity, price)) from transactions
group by transaction_id

SELECT
    transaction_id,
    sum(transaction_sum(quantity, price))
FROM transactions
GROUP BY transaction_id

Query id: e76ecb5b-510f-4db1-aea0-b0cbb0343ce3

Ok.

0 rows in set. Elapsed: 0.036 sec.

envy :)
```

3. 3 — 3.4
Создайте UDF для классификации транзакций на «высокоценные» и «малоценные» на основе порогового значения (например, 100).

Примените UDF для категоризации каждой транзакции.

```
envy :) CREATE FUNCTION transaction_category AS (a,b,c) -> if(a*b>=c, 'high-value', 'low-value')

CREATE FUNCTION transaction_category AS (a, b, c) -> if((a * b) >= c, 'high-value', 'low-value')

Query id: e8263713-53ea-4b18-9c33-5026b010c42c

Ok.

0 rows in set. Elapsed: 0.087 sec.

envy :) SELECT transaction_id, transaction_category(price, quantity, 100) from transactions;

SELECT
    transaction_id,
    transaction_category(price, quantity, 100)
FROM transactions

Query id: fe91d627-cf3e-4143-926f-66c26663b18e

Ok.

0 rows in set. Elapsed: 0.003 sec.

envy :)
```