Оглавление

[**Дашборды** 1](#_Toc129282285)

[**Задание: 1.** 3](#_Toc129282286)

[**МЕТРИКИ** 10](#_Toc129282287)

[**Задание 1** 10](#_Toc129282288)

[**Задача 2.** 12](#_Toc129282289)

[**\*Задание 3** 14](#_Toc129282290)

[**Задача 4.** 16](#_Toc129282291)

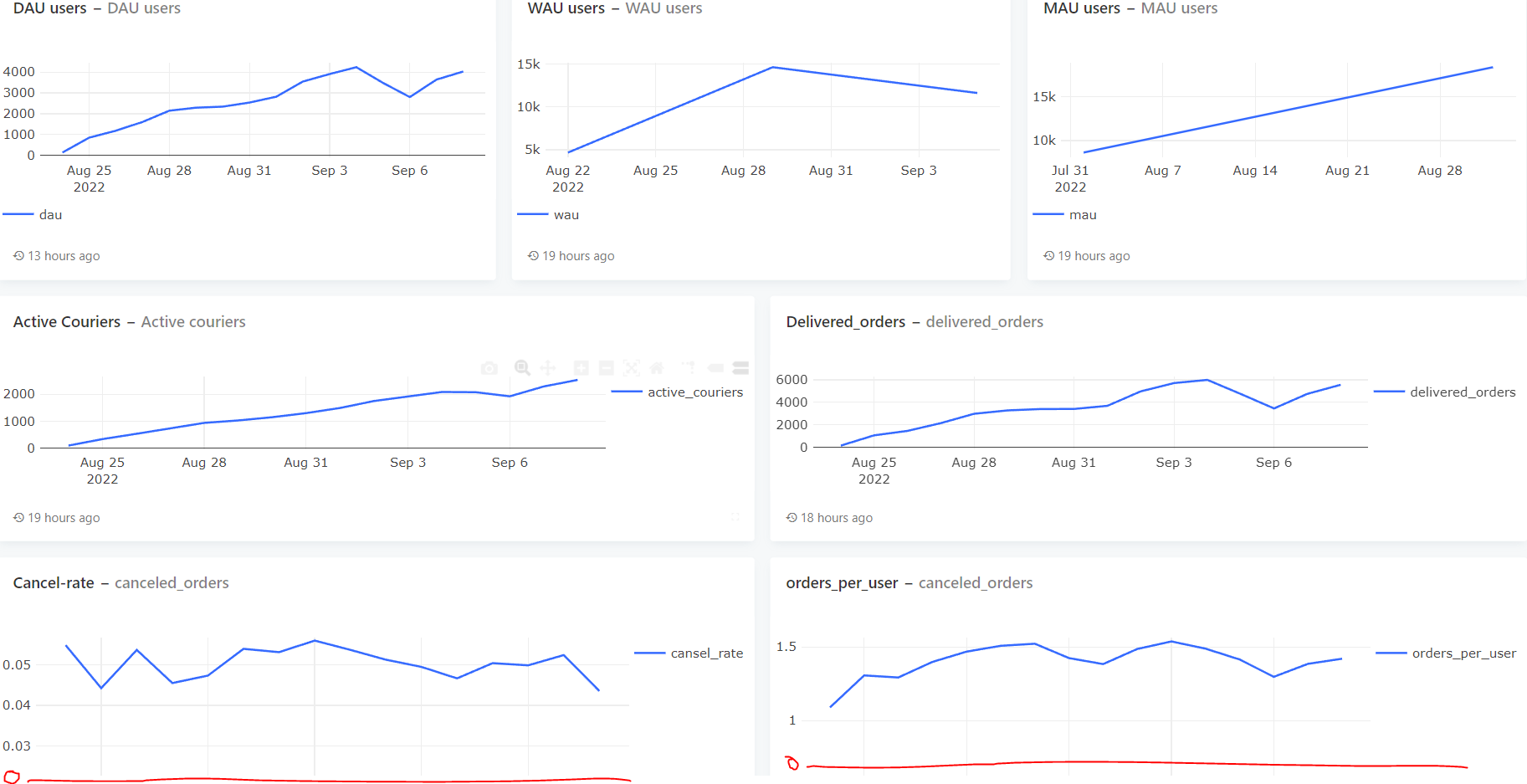
[**\*Задача 5** 18](#_Toc129282292)

[**Задача 6** 20](#_Toc129282293)

**Метрики DAU, WAU, MAU (кол-во уникальных активных пользователей по дням, неделям, месяцам)**

**Дашборды**

Дашборд – <http://redash.public.karpov.courses/public/dashboards/KThsQpcjPJcaR7nMY6CfBxFRXy6OibpHQe7f4Rwb?org_slug=default>



SELECT time::DATE AS date,

COUNT(distinct user\_id) AS DAU

FROM user\_actions

GROUP BY date

-- Округление до первого числа недели

SELECT date\_trunc('week', time) AS week,

COUNT(distinct user\_id) AS WAU

FROM user\_actions

GROUP BY week

-- Округление до первой даты месяца

SELECT date\_trunc('month', time) AS month,

COUNT(distinct user\_id) AS MAU

FROM user\_actions

GROUP BY month

===================================================================

SELECT

cancel\_orders::float / create\_orders as cansel\_rate,

create\_orders::float / active\_users as orders\_per\_user, -- Кол-во заказов на пользователя

cancel\_orders,

create\_orders,

active\_users,

date

from

(select

count(order\_id) filter(where action = 'cancel\_order') as cancel\_orders,

count(order\_id) filter(where action = 'create\_order') as create\_orders,

count(distinct user\_id) as active\_users,

time::date as date

from user\_actions

group by date) t1

**Задание: 1.**

Для начала давайте проанализируем, насколько быстро растёт аудитория нашего сервиса, и посмотрим на динамику числа пользователей и курьеров.

Для каждого дня, представленного в таблицах user\_actions и courier\_actions, рассчитайте следующие показатели:

* Число новых пользователей.
* Число новых курьеров.
* Общее число пользователей на текущий день.
* Общее число курьеров на текущий день.

Колонки с показателями назовите соответственно new\_users, new\_couriers, total\_users, total\_couriers. Колонку с датами назовите date. Проследите за тем, чтобы показатели были выражены целыми числами. Результат должен быть отсортирован по возрастанию даты.

Поля в результирующей таблице: date, new\_users, new\_couriers, total\_users, total\_couriers

Пояснение:

Новыми будем считать тех пользователей и курьеров, которые в данный день совершили своё первое действие в нашем сервисе. Общее число пользователей/курьеров на текущий день — это результат сложения числа новых пользователей/курьеров в текущий день со значениями аналогичного показателя всех предыдущих дней.

После того как составите запрос, попробуйте визуализировать результаты и постройте графики, отражающие динамику рассчитанных показателей.

Предлагаем вам построить два графика: на один поместить динамику новых пользователей и курьеров, а на второй — рост их общего числа.

SELECT min\_date as date, new\_users, new\_couriers,

SUM(new\_users) OVER (ORDER BY min\_date)::int AS total\_users,

SUM(new\_couriers) OVER (ORDER BY min\_date)::int AS total\_couriers

FROM

(SELECT min\_date, count(courier\_id) AS new\_couriers

FROM

(SELECT courier\_id, MIN(time::date) AS min\_date

FROM courier\_actions

GROUP BY courier\_id) t1

GROUP BY min\_date) t3

JOIN

(SELECT min\_date, count(user\_id) AS new\_users

FROM

(SELECT user\_id, MIN(time::date) AS min\_date

FROM user\_actions

GROUP BY user\_id) t2

GROUP BY min\_date) t4

using(min\_date)

ORDER BY date

Проанализируйте построенные графики и попробуйте ответить на следующие вопросы:

* Что растёт быстрее: количество пользователей или количество курьеров?

**Количество пользователей.**

* Насколько стабильны показатели числа новых пользователей и курьеров? Нет ли в данных таких дней, когда показатели сильно выбивались из общей динамики?

**Количество новых пользователей нестабильно, 5-7 сентября падение на 500 человек.**

* Можно ли сказать, что показатель числа новых курьеров более стабилен, чем показатель числа новых пользователей?

**Пока сказать точно нельзя. Сравнивать абсолютные значения не очень удобно. Лучше смотреть динамику показателей в относительных величинах.**

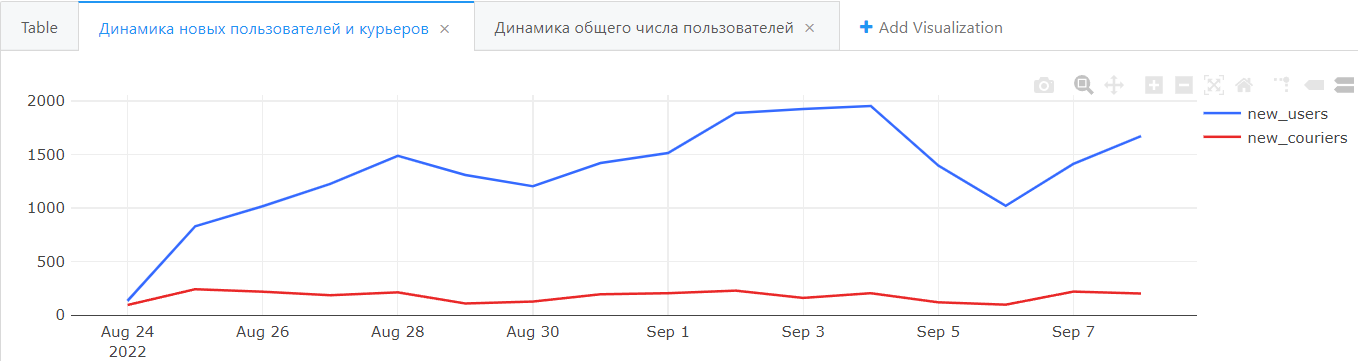


Рисунок 1 - Динамика новых пользователей и курьеров

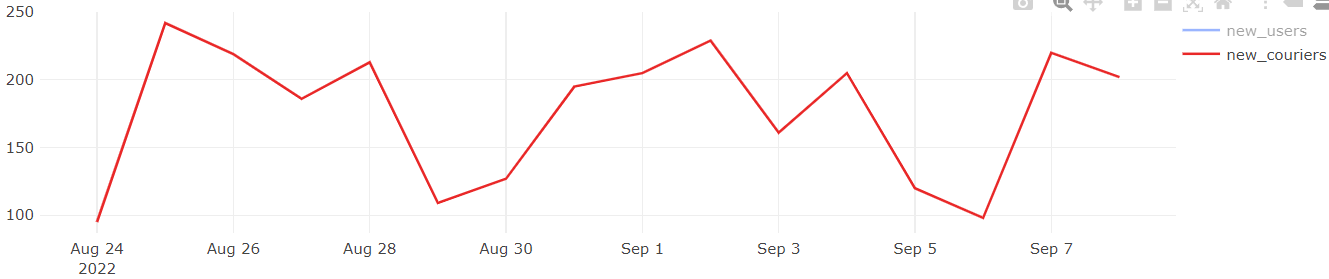


Рисунок 2 - Динамика новых курьеров

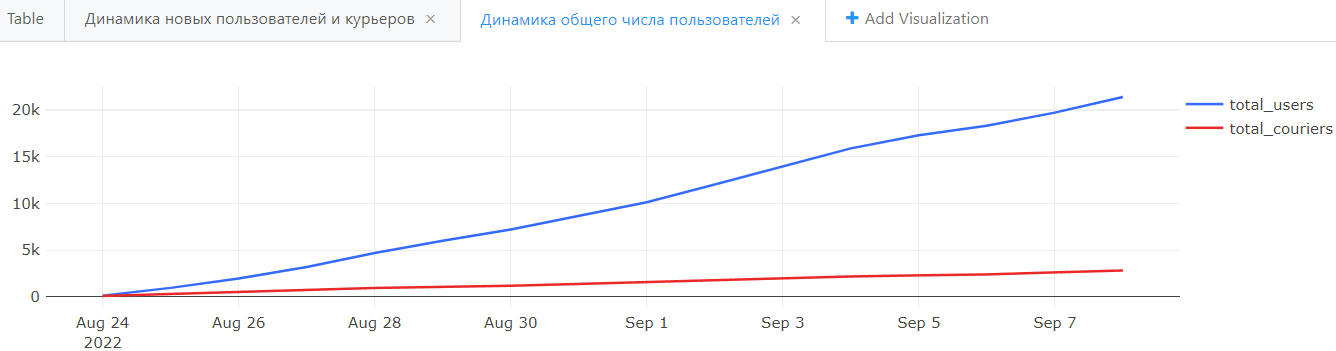


Рисунок 3 - Динамика общего числа пользователей

**Задание 2.**

Дополните запрос из предыдущего задания и теперь для каждого дня, представленного в таблицах user\_actions и courier\_actions, дополнительно рассчитайте следующие показатели:

* Прирост числа новых пользователей.
* Прирост числа новых курьеров.
* Прирост общего числа пользователей.
* Прирост общего числа курьеров.

Показатели, рассчитанные на предыдущем шаге, также включите в результирующую таблицу.

Колонки с новыми показателями назовите соответственно new\_users\_change, new\_couriers\_change, total\_users\_growth, total\_couriers\_growth. Колонку с датами назовите date.

Все показатели прироста считайте в процентах относительно значений в предыдущий день. При расчёте показателей округляйте значения до двух знаков после запятой.

Результирующая таблица должна быть отсортирована по возрастанию даты.

Поля в результирующей таблице:

date, new\_users, new\_couriers, total\_users, total\_couriers, new\_users\_change, new\_couriers\_change, total\_users\_growth, total\_couriers\_growth

Пояснение:

Не забывайте при делении заранее приводить значения к нужному типу данных. Пропущенные значения приростов для самой первой даты не заполняйте — просто оставьте поля в этой строке пустыми.

WITH subquery1 AS (

SELECT min\_date as date, new\_users, new\_couriers,

SUM(new\_users) OVER (ORDER BY min\_date)::int AS total\_users,

SUM(new\_couriers) OVER (ORDER BY min\_date)::int AS total\_couriers

FROM

(SELECT min\_date, count(courier\_id) AS new\_couriers

FROM

(SELECT courier\_id, MIN(time::date) AS min\_date

FROM courier\_actions

GROUP BY courier\_id) t1

GROUP BY min\_date) t3

JOIN

(SELECT min\_date, count(user\_id) AS new\_users

FROM

(SELECT user\_id, MIN(time::date) AS min\_date

FROM user\_actions

GROUP BY user\_id) t2

GROUP BY min\_date) t4

using(min\_date)

ORDER BY date)

SELECT date, new\_users, new\_couriers, total\_users, total\_couriers,

ROUND(new\_users\*100::decimal / lag(new\_users) over (order by date) - 100, 2) AS new\_users\_change,

ROUND(new\_couriers\*100::decimal / lag(new\_couriers) over (order by date) - 100, 2) AS new\_couriers\_change,

ROUND(total\_users\*100::decimal / lag(total\_users) over (order by date) - 100, 2) AS total\_users\_growth,

ROUND(total\_couriers\*100::decimal / lag(total\_couriers) over (order by date) - 100, 2) AS total\_couriers\_growth

FROM subquery1

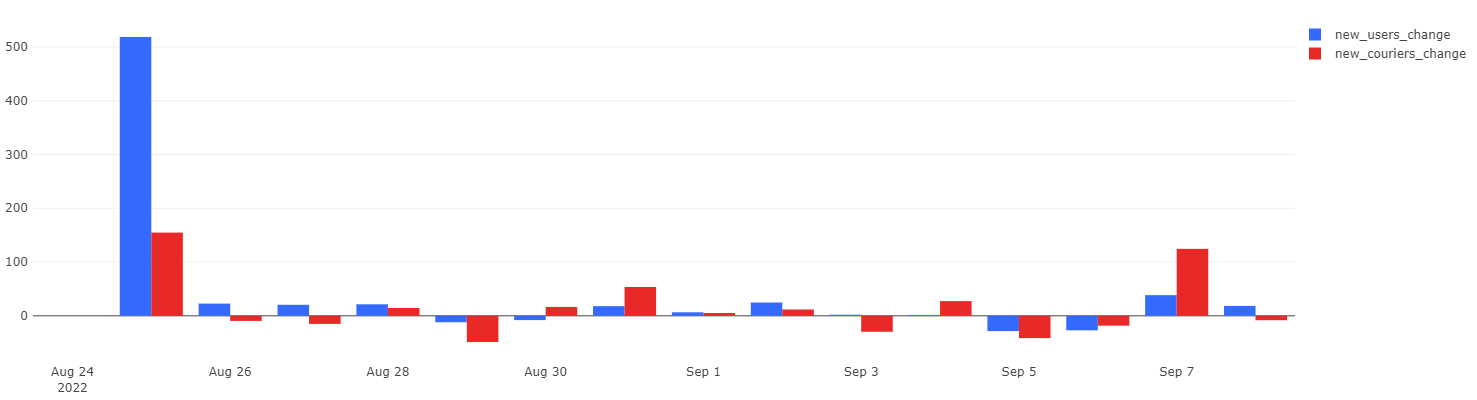


Рисунок 4 - Динамика прироста числа новых пользователей и курьеров

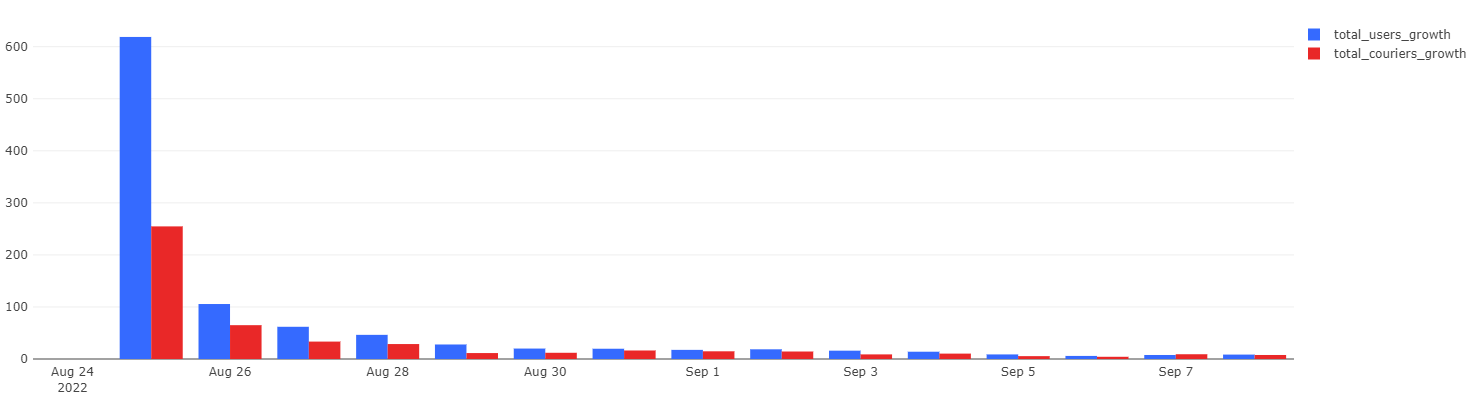


Рисунок 5 - Динамика прироста общего числа пользователей и курьеров

Проанализируйте построенные графики и попробуйте ответить на следующие вопросы:

1. Как изменились темпы прироста общего числа пользователей и курьеров за рассматриваемый промежуток времени? Какая в целом динамика у этих показателей: они растут или, наоборот, затухают?

**Темпы прироста общего числа пользователей и курьеров за рассматриваемый промежуток времени снизились, а показатели затухают.**

1. В какие дни темп прироста числа новых курьеров заметно опережал темп прироста числа новых пользователей?

**30 -31 августа, 4, 7 сентября.**

1. Можно ли, глядя на графики с относительными показателями, сказать, что показатель числа новых курьеров более стабилен, чем показатель числа новых пользователей?

**Нет, если не учитывать выброс числа новых пользователей в первый день.**

**Задача 3.**

Теперь предлагаем вам посмотреть на нашу аудиторию немного под другим углом — давайте посчитаем не просто всех пользователей, а именно ту часть, которая оформляет и оплачивает заказы в нашем сервисе. Заодно выясним, какую долю платящие пользователи составляют от их общего числа.

Для каждого дня, представленного в таблицах user\_actions и courier\_actions, рассчитайте следующие показатели:

* Число платящих пользователей.
* Число активных курьеров.
* Долю платящих пользователей в общем числе пользователей на текущий день.
* Долю активных курьеров в общем числе курьеров на текущий день.

Колонки с показателями назовите соответственно paying\_users, active\_couriers, paying\_users\_share, active\_couriers\_share. Колонку с датами назовите date. Проследите за тем, чтобы абсолютные показатели были выражены целыми числами. Все показатели долей необходимо выразить в процентах. При их расчёте округляйте значения до двух знаков после запятой.

Результат должен быть отсортирован по возрастанию даты.

Поля в результирующей таблице: date, paying\_users, active\_couriers, paying\_users\_share, active\_couriers\_share

Пояснение:

Платящими будем считать тех пользователей, которые в данный день оформили хотя бы один заказ, который в дальнейшем не был отменен.

Курьеров будем считать активными, если в данный день они приняли хотя бы один заказ, который был доставлен (возможно, уже на следующий день), или доставили любой заказ.

Общее число пользователей/курьеров на текущий день — это по-прежнему результат сложения числа новых пользователей/курьеров в текущий день со значениями аналогичного показателя всех предыдущих дней. Мы считали этот показатель на предыдущих шагах.

После того как составите запрос, попробуйте визуализировать результаты и постройте графики, отражающие динамику рассчитанных показателей.

WITH subquery1 AS (

SELECT min\_date as date, new\_users, new\_couriers,

SUM(new\_users) OVER (ORDER BY min\_date)::int AS total\_users,

SUM(new\_couriers) OVER (ORDER BY min\_date)::int AS total\_couriers

FROM

(SELECT min\_date, count(courier\_id) AS new\_couriers

FROM

(SELECT courier\_id, MIN(time::date) AS min\_date

FROM courier\_actions

GROUP BY courier\_id) t1

GROUP BY min\_date) t3

JOIN

(SELECT min\_date, count(user\_id) AS new\_users

FROM

(SELECT user\_id, MIN(time::date) AS min\_date

FROM user\_actions

GROUP BY user\_id) t2

GROUP BY min\_date) t4

using(min\_date)

ORDER BY date),

subquery2 AS (

SELECT date, paying\_users, active\_couriers

FROM

(SELECT time::date AS date, count(courier\_id) AS active\_couriers

FROM courier\_actions

WHERE action = 'deliver\_order' or order\_id IN (SELECT order\_id FROM courier\_actions WHERE action = 'accept\_order')

and order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')

GROUP BY date) t1

JOIN

(SELECT time::date as date, count(distinct user\_id) AS paying\_users

FROM user\_actions

WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')

GROUP BY date) t2

using(date)

)

SELECT date, paying\_users, active\_couriers,

ROUND(paying\_users\*100::decimal / total\_users, 2) AS paying\_users\_share,

ROUND(active\_couriers\*100::decimal / total\_couriers, 2) AS active\_couriers\_share

FROM subquery1 JOIN subquery2

using(date)

**МЕТРИКИ**

Начнём с выручки — наиболее общего показателя, который покажет, какой доход приносит наш сервис.

**Экономика продукта**

**Задание 1**

Для каждого дня в таблице orders рассчитайте следующие показатели:

* Выручку, полученную в этот день.
* Суммарную выручку на текущий день.
* Прирост выручки, полученной в этот день, относительно значения выручки за предыдущий день.

Колонки с показателями назовите соответственно revenue, total\_revenue, revenue\_change. Колонку с датами назовите date.

Прирост выручки рассчитайте в процентах и округлите значения до двух знаков после запятой.

Результат должен быть отсортирован по возрастанию даты.

Поля в результирующей таблице: date, revenue, total\_revenue, revenue\_change

Пояснение:

Будем считать, что оплата за заказ поступает сразу же после его оформления, т.е. **случаи, когда заказ был оформлен в один день, а оплата получена на следующий, возникнуть не могут.**

Суммарная выручка на текущий день — это результат сложения выручки, полученной в текущий день, со значениями аналогичного показателя всех предыдущих дней.

При расчёте выручки помните, что не все заказы были оплачены — некоторые были отменены пользователями.

Не забывайте при делении заранее приводить значения к нужному типу данных. Пропущенные значения прироста для самой первой даты не заполняйте — просто оставьте поля в этой строке пустыми.

*WITH subquery1 AS (*

*SELECT creation\_time::DATE as date, SUM(price) as revenue*

*FROM*

*(SELECT creation\_time, order\_id, product\_ids, unnest(product\_ids) as product\_id*

*FROM orders*

*WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) t1*

*LEFT JOIN products using(product\_id)*

*GROUP BY creation\_time::DATE)*

*SELECT date, revenue,*

*sum(revenue) over (order by date) as total\_revenue,*

*ROUND(revenue \* 100::decimal / lag(revenue) over (order by date) -100, 2) as revenue\_change*

*FROM subquery1*

*ORDER BY date*

После того как составите запрос, попробуйте визуализировать результаты и постройте графики, отражающие динамику рассчитанных показателей.

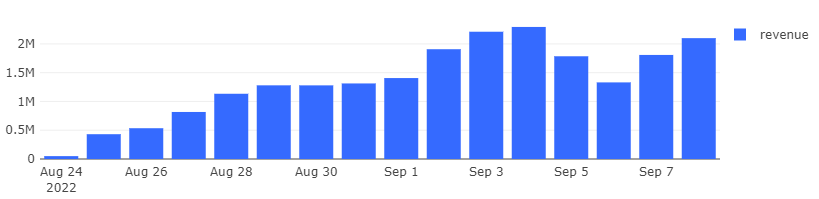


Рисунок 6 - Динамика ежедневной выручки

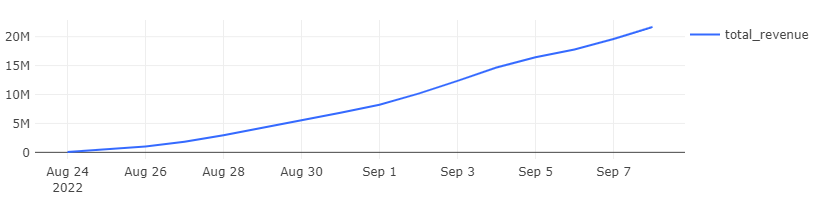


Рисунок 7 - Динамика общей выручки

* В какие дни наблюдалось заметное снижение ежедневной выручки?

Заметное снижение ежедневной выручки наблюдалось 5 и 6 сентября.

* С чем это могло быть связано? (чтобы установить причину падения выручки, можете сопоставить текущие графики с графиками на дашборде из прошлого урока)

Это могло быть связано со снижением числа новых пользователей и заказов.

**Задача 2.**

Теперь на основе данных о выручке рассчитаем несколько относительных показателей, которые покажут, сколько в среднем потребители готовы платить за услуги нашего сервиса доставки. Остановимся на следующих метриках:

1. ARPU (Average Revenue Per User) — средняя выручка на одного пользователя за определённый период.

2. ARPPU (Average Revenue Per Paying User) — средняя выручка на одного платящего пользователя за определённый период.

3. AOV (Average Order Value) — средний чек, или отношение выручки за определённый период к общему количеству заказов за это же время.

Если за рассматриваемый период сервис заработал 100 000 рублей и при этом им пользовались 500 уникальных пользователей, из которых 400 сделали в общей сложности 650 заказов, тогда метрики будут иметь следующие значения:

ARPU =100000/500=200

ARPPU =100000/400=250

AOV=100000/650≈153,85

Задание:

Для каждого дня в таблицах orders и user\_actions рассчитайте следующие показатели:

* Выручку на пользователя (ARPU) за текущий день.
* Выручку на платящего пользователя (ARPPU) за текущий день.
* Выручку с заказа, или средний чек (AOV) за текущий день.

Колонки с показателями назовите соответственно arpu, arppu, aov. Колонку с датами назовите date.

При расчёте всех показателей округляйте значения до двух знаков после запятой.

Результат должен быть отсортирован по возрастанию даты.

Поля в результирующей таблице: date, arpu, arppu, aov

Пояснение:

Будем считать, что оплата за заказ поступает сразу же после его оформления, т.е. **случаи, когда заказ был оформлен в один день, а оплата получена на следующий, возникнуть не могут.**

Платящими будем считать тех пользователей, которые в данный день оформили хотя бы один заказ, который в дальнейшем не был отменен.

При расчёте выручки помните, что не все заказы были оплачены — некоторые были отменены пользователями.

Не забывайте при делении заранее приводить значения к нужному типу данных.

WITH subquery1 AS (

SELECT date, SUM(price) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as revenue,

count(distinct user\_id) as count\_users,

count(distinct user\_id) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as count\_paying\_users,

count(distinct order\_id) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as count\_orders

FROM

(SELECT time::date as date, order\_id, product\_ids, unnest(product\_ids) as product\_id, user\_id

FROM orders JOIN user\_actions using(order\_id)

) t1

LEFT JOIN products using(product\_id)

GROUP BY date)

SELECT date,

ROUND(revenue::decimal / count\_users, 2) as arpu,

ROUND(revenue::decimal / count\_paying\_users, 2) as arppu,

ROUND(revenue::decimal / count\_orders, 2) as aov

FROM subquery1

ORDER BY date

После того как составите запрос, попробуйте визуализировать результаты и постройте график, отражающий динамику рассчитанных показателей.

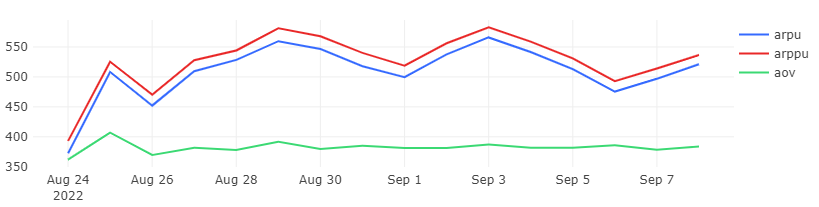


Рисунок 8 - Динамика показателей

Проанализируйте построенный график и попробуйте ответить на следующие вопросы:

* Какие метрики имеют больший разброс значений на протяжении рассматриваемого периода?

Выручка на пользователя (ARPU) и Выручка на платящего пользователя (ARPPU) за день.

* Можно ли сказать, что отдельные метрики имеют аномально высокие или аномально низкие значения в отдельные дни?

Нет, значительных выбросов нет.

* Какой вывод можно сделать о соотношении числа платящих пользователей и всех пользователей сервиса в рассматриваемые дни?

Почти все пользователи что-нибудь заказывают в рассматриваемые дни.

**\*Задание 3**

Дополним наш анализ ещё более интересными расчётами — вычислим все те же метрики, но для каждого дня будем учитывать накопленную выручку и все имеющиеся на текущий момент данные о числе пользователей и заказов. Таким образом, получим динамический ARPU, ARPPU и AOV и сможем проследить, как он менялся на протяжении времени с учётом поступающих нам данных.

По таблицам orders и user\_actions для каждого дня рассчитайте следующие показатели:

* Накопленную выручку на пользователя (Running ARPU).
* Накопленную выручку на платящего пользователя (Running ARPPU).
* Накопленную выручку с заказа, или средний чек (Running AOV).

Колонки с показателями назовите соответственно running\_arpu, running\_arppu, running\_aov. Колонку с датами назовите date.

При расчёте всех показателей округляйте значения до двух знаков после запятой.

Результат должен быть отсортирован по возрастанию даты.

Поля в результирующей таблице: date, running\_arpu, running\_arppu, running\_aov

Пояснение:

При расчёте числа пользователей и платящих пользователей на текущую дату учитывайте соответствующих пользователей за все предыдущие дни, включая текущий.

Платящими будем считать тех пользователей, которые на текущий день оформили хотя бы один заказ, который в дальнейшем не был отменен.

Будем считать, что оплата за заказ поступает сразу же после его оформления, т.е. случаи, когда заказ был оформлен в один день, а оплата получена на следующий, возникнуть не могут.

При расчёте выручки помните, что не все заказы были оплачены — некоторые были отменены пользователями.

Не забывайте при делении заранее приводить значения к нужному типу данных.

*WITH subquery1 AS (*

*SELECT date, SUM(price) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as revenue,*

*count(distinct user\_id) as count\_users,*

*count(distinct user\_id) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as count\_paying\_users,*

*count(distinct order\_id) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as count\_orders*

*FROM*

*(SELECT time::DATE as date, order\_id, product\_ids, unnest(product\_ids) as product\_id, user\_id*

*FROM orders JOIN user\_actions using(order\_id)*

*) t1*

*LEFT JOIN products using(product\_id)*

*GROUP BY date)*

*SELECT date,*

*ROUND(revenue2::decimal / count\_users2, 2) as running\_arpu,*

*ROUND(revenue2::decimal / count\_paying\_users2, 2) as running\_arppu,*

*ROUND(revenue2::decimal / count\_orders2, 2) as running\_aov*

*FROM*

*(SELECT date,*

*sum(revenue) over (order by date) as revenue2,*

*sum(count\_users) over (order by date) as count\_users2,*

*sum(count\_paying\_users) over (order by date) as count\_paying\_users2,*

*sum(count\_orders) over (order by date) as count\_orders2*

*FROM subquery1) t2*

*ORDER BY date*

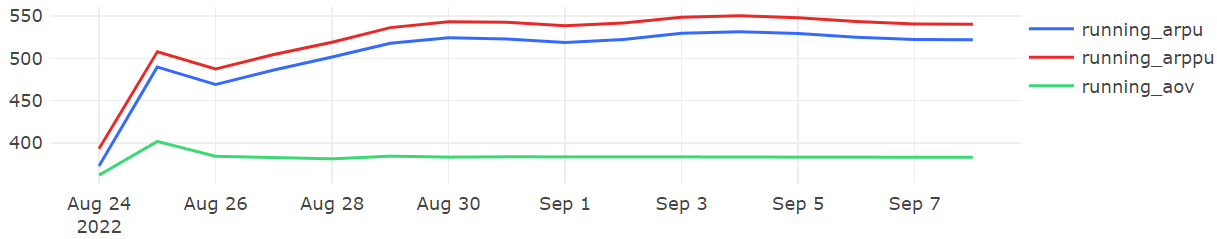


Рисунок 9 - Динамика показателей

Проанализируйте построенный график и попробуйте ответить на следующие вопросы:

* Какая в целом динамика у рассчитанных метрик? Они растут, падают или имеют примерно одинаковое значение в каждый из дней?

Данные метрики имеют примерно одинаковое значение каждый день.

* Можно ли, учитывая динамику рассчитанных метрик, предположить, что со временем растёт число заказов на одного пользователя?

Нельзя.

**Задача 4**

Давайте посчитаем те же показатели, но в другом разрезе — не просто по дням, а по дням недели.

Задание:

Для каждого дня недели в таблицах orders и user\_actions рассчитайте следующие показатели:

* Выручку на пользователя (ARPU).
* Выручку на платящего пользователя (ARPPU).
* Выручку на заказ (AOV).

При расчётах учитывайте данные только за период с 26 августа 2022 года по 8 сентября 2022 года включительно — так, чтобы в анализ попало одинаковое количество всех дней недели (ровно по два дня).

В результирующую таблицу включите как наименования дней недели (например, Monday), так и порядковый номер дня недели (от 1 до 7, где 1 — это Monday, 7 — это Sunday).

Колонки с показателями назовите соответственно arpu, arppu, aov. Колонку с наименованием дня недели назовите weekday, а колонку с порядковым номером дня недели weekday\_number.

При расчёте всех показателей округляйте значения до двух знаков после запятой.

Результат должен быть отсортирован по возрастанию порядкового номера дня недели.

Поля в результирующей таблице:

weekday, weekday\_number, arpu, arppu, aov

Пояснение:

Будем считать, что оплата за заказ поступает сразу же после его оформления, т.е. случаи, когда заказ был оформлен в один день, а оплата получена на следующий, возникнуть не могут.

Платящими будем считать тех пользователей, которые в данный день оформили хотя бы один заказ, который в дальнейшем не был отменен.

При расчёте выручки помните, что не все заказы были оплачены — некоторые были отменены пользователями.

Не забывайте при делении заранее приводить значения к нужному типу данных.

В этой задаче порядковый номер дня недели необходим для того, чтобы дни недели были расположены на графике слева направо в правильном порядке — не по возрастанию наименования, а по возрастанию порядкового номера. Для получения корректной визуализации в настройках оси X необходимо отключить сортировку, установленную по умолчанию.

Получить наименование дня недели из колонки с датой или отметкой времени можно с помощью функции to\_char:

SELECT to\_char(TIMESTAMP '2023-01-13', 'Day')

Результат:

Wednesday

Для выделения порядкового номера для недели можно использовать функцию DATE\_PART с параметром 'isodow'.

*WITH subquery1 as (*

*SELECT to\_char(date, 'Day') as weekday, date\_part('isodow', date) as weekday\_number,*

*SUM(price) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as revenue,*

*count(distinct user\_id) as count\_users,*

*count(distinct user\_id) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as count\_paying\_users,*

*count(distinct order\_id) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as count\_orders*

*FROM*

*(SELECT time::DATE as date, order\_id, product\_ids, unnest(product\_ids) as product\_id, user\_id*

*FROM orders join user\_actions using(order\_id)*

*WHERE time BETWEEN '2022-08-26' AND '2022-09-09') t1*

*LEFT JOIN products using(product\_id)*

*GROUP BY weekday, weekday\_number)*

*SELECT weekday, weekday\_number,*

*ROUND(revenue::decimal / count\_users, 2) as arpu,*

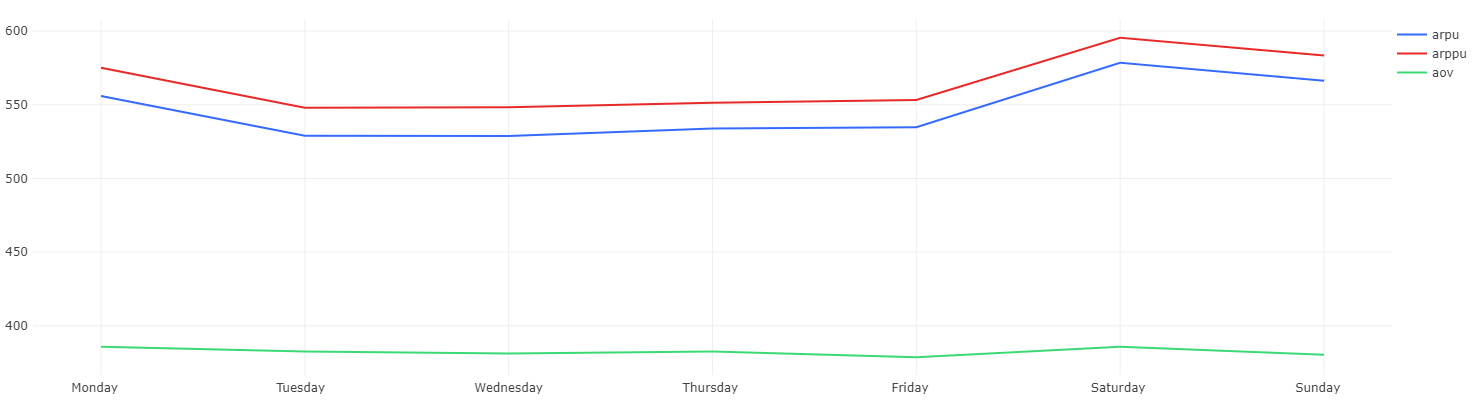
*ROUND(revenue::decimal / count\_paying\_users, 2) as arppu,*

*ROUND(revenue::decimal / count\_orders, 2) as aov*

*FROM subquery1*

*ORDER BY weekday\_number*

После того как составите запрос, попробуйте визуализировать результаты и постройте график, отражающий динамику рассчитанных показателей.



Проанализируйте построенный график и попробуйте ответить на следующие вопросы:

* В какие дни недели метрики ARPU и ARPPU принимали наибольшие значения? Как вы считаете, согласуется ли это в целом со стандартным поведением пользователей сервиса доставки еды?

**В субботу, воскресенье, понедельник.**

* Как вы считаете, почему в те дни, когда метрики ARPU и ARPPU принимали наибольшие значения, метрика AOV оставалась примерно на том же уровне? При каком сценарии такое возможно?

**Когда один покупатель сделал много заказов.**

**\*Задача 5**

Немного усложним наш первоначальный запрос и отдельно посчитаем ежедневную выручку с заказов новых пользователей нашего сервиса. Посмотрим, какую долю она составляет в общей выручке с заказов всех пользователей — и новых, и старых.

Задание:

Для каждого дня в таблицах orders и user\_actions рассчитайте следующие показатели:

Выручку, полученную в этот день.

Выручку с заказов новых пользователей, полученную в этот день.

Долю выручки с заказов новых пользователей в общей выручке, полученной за этот день.

Долю выручки с заказов остальных пользователей в общей выручке, полученной за этот день.

Колонки с показателями назовите соответственно revenue, new\_users\_revenue, new\_users\_revenue\_share, old\_users\_revenue\_share. Колонку с датами назовите date.

Все показатели долей необходимо выразить в процентах. При их расчёте округляйте значения до двух знаков после запятой.

Результат должен быть отсортирован по возрастанию даты.

Поля в результирующей таблице:

date, revenue, new\_users\_revenue, new\_users\_revenue\_share, old\_users\_revenue\_share

Пояснение:

Будем считать, что оплата за заказ поступает сразу же после его оформления, т.е. случаи, когда заказ был оформлен в один день, а оплата получена на следующий, возникнуть не могут.

Новыми будем считать тех пользователей, которые в данный день совершили своё первое действие в нашем сервисе.

При расчёте выручки помните, что не все заказы были оплачены — некоторые были отменены пользователями.

Не забывайте при делении заранее приводить значения к нужному типу данных.

*WITH date\_first\_action as (*

*-- Дата первого действия пользователя*

*SELECT user\_id, MIN(time::date) AS min\_date*

*FROM user\_actions*

*GROUP BY user\_id*

*),*

*order\_cost as (*

*-- Стоимость заказов*

*SELECT order\_id, sum(price)as order\_cost*

*FROM*

*(SELECT order\_id, unnest(product\_ids) as product\_id*

*FROM orders) t1*

*LEFT JOIN products using(product\_id)*

*GROUP BY order\_id*

*),*

*total\_cost\_on\_date as (*

*-- Суммарная стоимость заказов на каждую дату для пользователя*

*SELECT user\_id, time::date as date, sum(order\_cost)*

*FROM user\_actions LEFT JOIN order\_cost using(order\_id)*

*WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')*

*GROUP BY user\_id, date*

*)*

*SELECT date, revenue, new\_users\_revenue,*

*ROUND(new\_users\_revenue \* 100::decimal / revenue, 2) as new\_users\_revenue\_share,*

*100 - ROUND(new\_users\_revenue \* 100::decimal / revenue, 2) as old\_users\_revenue\_share*

*FROM*

*-- Выручка, полученная с каждого пользователя в его первый день(user\_id, date, sum),*

*-- GROUP BY - суммарная выручка с новых пользователей за каждый день*

*(SELECT date, sum(sum) as new\_users\_revenue*

*FROM date\_first\_action LEFT JOIN total\_cost\_on\_date using(user\_id)*

*WHERE min\_date = date*

*GROUP BY date) t2*

*LEFT JOIN*

*-- Выручка, полученная в этот день*

*(SELECT time::date as date, sum(order\_cost) as revenue*

*FROM user\_actions LEFT JOIN order\_cost using(order\_id)*

*WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')*

*GROUP BY date) t3*

*USING(date)*

*ORDER BY date*

После того как составите запрос, попробуйте визуализировать результаты и постройте график, отражающий динамику рассчитанных показателей.

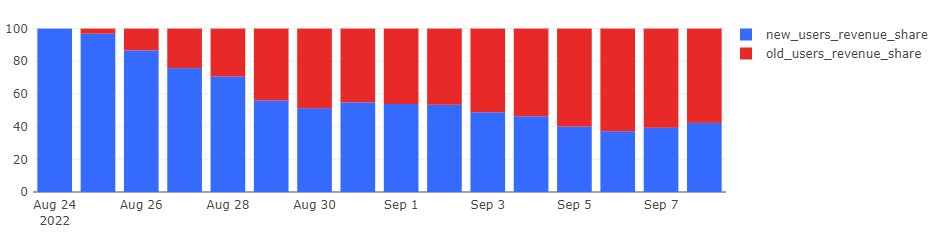


Рисунок 10 - Динамика рассчитанных показателей

Можно ли сказать, что спустя две недели после запуска нашего сервиса показатель выручки от новых пользователей по-прежнему на довольно высоком уровне?

**Да, 42% это хороший результат. Мы довольно хорошо удерживаем наших постоянных клиентов.**

**Задача 6**

Также было бы интересно посмотреть, какие товары пользуются наибольшим спросом и приносят нам основной доход.

Задание:

Для каждого товара, представленного в таблице products, за весь период времени в таблице orders рассчитайте следующие показатели:

1. Суммарную выручку, полученную от продажи этого товара за весь период.
2. Долю выручки от продажи этого товара в общей выручке, полученной за весь период.

Колонки с показателями назовите соответственно revenue и share\_in\_revenue. Колонку с наименованиями товаров назовите product\_name.

Долю выручки с каждого товара необходимо выразить в процентах. При её расчёте округляйте значения до двух знаков после запятой.

Товары, округлённая доля которых в выручке составляет менее 0.5%, объедините в общую группу с названием «ДРУГОЕ» (без кавычек), просуммировав округлённые доли этих товаров.

Результат должен быть отсортирован по убыванию выручки от продажи товара.

Поля в результирующей таблице: product\_name, revenue, share\_in\_revenue

Пояснение:

Будем считать, что оплата за заказ поступает сразу же после его оформления, т.е. случаи, когда заказ был оформлен в один день, а оплата получена на следующий, возникнуть не могут.

При расчёте выручки помните, что не все заказы были оплачены — некоторые были отменены пользователями.

Товары с небольшой долей в выручке необходимо объединить в одну группу, чтобы не выводить на графике абсолютно все товары из таблицы products.

WITH subquery1 AS (

SELECT name as product\_name,

SUM(price) FILTER (WHERE order\_id NOT IN (SELECT order\_id FROM user\_actions WHERE action = 'cancel\_order')) as revenue

FROM

(SELECT order\_id, product\_ids, unnest(product\_ids) as product\_id, user\_id

FROM orders JOIN user\_actions using(order\_id)

) t1

LEFT JOIN products using(product\_id)

GROUP BY product\_name),

subquery2 as (

SELECT product\_name,

revenue,

ROUND(revenue \* 100 / SUM(revenue) OVER(), 2) as share\_in\_revenue

FROM subquery1

),

subquery3 as (

SELECT sum(revenue) as revenue, sum(share\_in\_revenue) as share\_in\_revenue

FROM subquery2

WHERE share\_in\_revenue <= 0.5

)

SELECT \*

FROM subquery2

WHERE share\_in\_revenue > 0.5

UNION

SELECT 'ДРУГОЕ' AS product\_name, revenue, share\_in\_revenue FROM subquery3

ORDER BY revenue DESC

**-- 2 Способ**

SELECT product\_name,

sum(revenue) as revenue,

sum(share\_in\_revenue) as share\_in\_revenue

FROM (SELECT case when round(100 \* revenue / sum(revenue) OVER (), 2) >= 0.5 then name

else 'ДРУГОЕ' end as product\_name,

revenue,

round(100 \* revenue / sum(revenue) OVER (), 2) as share\_in\_revenue

FROM (SELECT name,

sum(price) as revenue

FROM (SELECT order\_id,

unnest(product\_ids) as product\_id

FROM orders

WHERE order\_id not in (SELECT order\_id

FROM user\_actions

WHERE action = 'cancel\_order')) t1

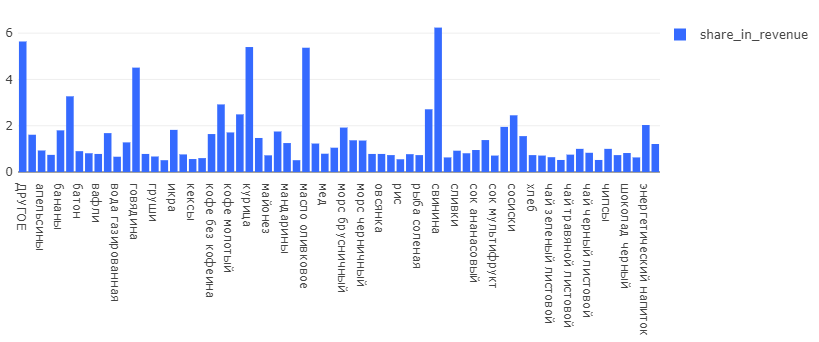
LEFT JOIN products using(product\_id)

GROUP BY name) t2) t3

GROUP BY product\_name

ORDER BY revenue desc

После того как составите запрос, попробуйте визуализировать результаты и постройте график, отражающий динамику рассчитанных показателей.



Посмотрите на построенный график и определите товары с наибольшей долей в выручке. Если бы мы объединили товары в более широкие группы, то какая, по вашему мнению, оказалась бы на первом месте по доле в выручке?

**Мясная продукция**

**Маркетинговые метрики**