

### ЧАСТЬ 1

#### 1. Запрос "Топ-3 аптеки"

- Вывести топ 3 аптеки по объему продаж (GROUP BY, LIMIT)

(название файла: «Ч\_1 п\_1 Топ 3 аптеки.sql»)

Запрос на определение топ-3 аптек по объему продаж помогает выявить лидеров рынка, которые приносят наибольшую прибыль. Это позволяет акцентировать внимание на успешных аптеках для дальнейшего развития, возможного расширения сотрудничества и оптимизации бизнес-процессов.

Сумма продаж сгруппирована по аптекам и выявлены «Топ-3 аптеки» (Столичная, Доктор Айболит, Здравсити).

```
SELECT
    pharmacy_name,
    SUM(price * count) AS total_sales -- Общая сумма продаж для аптеки
FROM
    pharma_orders -- Таблица заказов
GROUP BY
    pharmacy_name -- Группировка по названию аптеки
ORDER BY
    total_sales DESC -- Сортировка по объему продаж по убыванию
LIMIT 3; -- Выводим топ-3 аптеки
```

```
1 SELECT
2     pharmacy_name,
3     SUM(price * COUNT) AS total_sales -- Общая сумма продаж для аптеки
4 FROM
5     pharma_orders -- Таблица заказов
6 GROUP BY
7     pharmacy_name -- Группировка по названию аптеки
8 ORDER BY
9     total_sales DESC -- Сортировка по объему продаж по убыванию
10 LIMIT 3; -- Выводим топ-3 аптеки
11
```

pharmacy_name	total_sales
Столичная	1896483
Доктор Айболит	1893776
Здравсити	1863037

## 2. Запрос "Топ-3 лекарства"

- Вывести топ 3 лекарства по объему продаж

(название файла: «Ч\_1 п\_2 Топ 3 лекарства.sql»)

Запрос определяет топ-3 лекарства по объему продаж. Это позволяет идентифицировать наиболее популярные и прибыльные товары, сосредоточиться на их продвижении, оптимизировать запасы и улучшить стратегическое планирование продаж.

Сумма продаж сгруппирована по лекарству и выявлены «Топ-3 лекарства», наибольшая сумма продаж лекарства «Ибупрофен» - 777 548 ед.

```
SELECT
    drug,
    SUM(price * count) AS total_sales -- Общая сумма продаж для аптеки
FROM
    pharma_orders -- Таблица заказов
GROUP BY
    drug -- Группировка по лекарствам
ORDER BY
    total_sales DESC -- Сортировка по объему продаж по убыванию
LIMIT 3; -- Выводим топ-3 аптеки
```

```
1 SELECT
2     drug,
3     SUM(price * COUNT) AS total_sales -- Общая сумма продаж для аптеки
4 FROM
5     pharma_orders -- Таблица заказов
6 GROUP BY
7     drug -- Группировка по лекарствам
8 ORDER BY
9     total_sales DESC -- Сортировка по объему продаж по убыванию
10 LIMIT 3; -- Выводим топ-3 аптеки
11
```

drug	total_sales
Ибупрофен	777548
Цитрамон	776332
Аквафорте	771579

### 3. Запрос "Аптеки от 1.8 млн оборота"

- Вывести аптеки, имеющие более 1.8 млн оборота (HAVING)

(название файла: «Ч\_1 п\_3 Аптека от 1\_8 млн оборот.sql»)

Запрос выявляет аптеки, которые достигли оборота свыше 1.8 млн единиц валюты. Это помогает сосредоточить внимание на наиболее успешных точках продаж, оптимизировать ресурсы и стратегически планировать дальнейшее развитие и маркетинг.

Выявлены 3 аптеки имеющие оборот более 1.8 млн.руб. (Столичная, Доктор Айболит, Здравсити)

-- 1. Выбираем название аптеки и считаем сумму продаж (price \* count) для каждой аптеки

**SELECT**

pharmacy\_name,

**SUM**(price \* count) **AS** total\_sales

**FROM**

pharma\_orders -- 2. Группируем данные по аптекам, чтобы суммировать продажи каждой аптеки

**GROUP BY**

pharmacy\_name -- 3. Фильтруем только те аптеки, у которых общая сумма продаж превышает 1.8 млн

**HAVING**

**SUM**(price \* count) > 1800000 -- 4. Сортируем результат по названию аптеки

**ORDER BY**

pharmacy\_name;

```
1 -- 1. Выбираем название аптеки и считаем сумму продаж (price * count) для каждой аптеки
2 SELECT
3   pharmacy_name,
4   SUM(price * COUNT) AS total_sales
5 FROM
6   pharma_orders -- 2. Группируем данные по аптекам, чтобы суммировать продажи каждой аптеки
7 GROUP BY
8   pharmacy_name -- 3. Фильтруем только те аптеки, у которых общая сумма продаж превышает 1.8 млн
9 HAVING
10  SUM(price * COUNT) > 1800000 -- 4. Сортируем результат по названию аптеки
11 ORDER BY
12   pharmacy_name;
13
```

pharmacy_name	total_sales
Столичная	1896483
Доктор Айболит	1893776
Здравсити	1863037

#### 4. Запрос "Накопленная сумма продаж по каждой аптеке" (OVER)

(название файла: «Ч\_1 п\_4 Накопленная сумма продаж по аптеке.sql»)

Запрос позволяет определить накопленный объем продаж по каждой аптеке на основе ежедневных данных о продажах. Это позволяет отслеживать, как общий объем продаж в каждой аптеке накапливается с течением времени, и предоставляет информацию о финансовой динамике по каждой аптеке.

-- 1. Выбираем название аптеки и считаем сумму продаж для каждой строки заказа

**SELECT**

pharmacy\_name,

report\_date,

**SUM**(price \* count) **AS** daily\_sales, -- Считаем дневные продажи

-- 2. Используем оконную функцию для вычисления накопленной суммы продаж по каждой аптеке

**SUM**(

**SUM**(price \* count)

) **OVER** (

**PARTITION BY** pharmacy\_name

**ORDER BY**

report\_date **ASC**

) **AS** cumulative\_sales

**FROM**

pharma\_orders -- 3. Группируем данные по аптеке и дате отчета, чтобы получить сумму продаж за день

**GROUP BY**

pharmacy\_name,

report\_date -- 4. Сортируем данные для корректного отображения накопленной суммы

**ORDER BY**

pharmacy\_name,

report\_date;

```
1 -- 1. Выбираем название аптеки и считаем сумму продаж для каждой строки заказа
2 SELECT
3   pharmacy_name,
4   report_date,
5   SUM(price * COUNT) AS daily_sales, -- Считаем дневные продажи
6   -- 2. Используем оконную функцию для вычисления накопленной суммы продаж по каждой аптеке
7   SUM(
8     SUM(price * COUNT)
9   ) OVER (
10    PARTITION BY pharmacy_name
11    ORDER BY
12     report_date ASC
13   ) AS cumulative_sales
14 FROM
15 pharma_orders -- 3. Группируем данные по аптеке и дате отчета, чтобы получить сумму продаж за день
16 GROUP BY
17   pharmacy_name,
18   report_date -- 4. Сортируем данные для корректного отображения накопленной суммы
19 ORDER BY
20   pharmacy_name,
21   report_date;
22
```

pharmacy_name	report_date	daily_sales	cumulative_sales
Столичная	2024-02-01	8218	8218
Столичная	2024-02-02	16092	24310
Столичная	2024-02-03	7160	31470
Столичная	2024-02-04	15542	47012
Столичная	2024-02-05	18098	65110

## 5. Запрос "Количество клиентов в аптеках"

- Соединить таблицы заказов и клиентов (JOIN)
- Посчитать кол-во уникальных клиентов на каждую аптеку (DISTINCT)
- Отсортировать аптеки по убыванию кол-ва клиентов (ORDER BY)

(название файла: «Ч\_1 п\_5 Количество клиентов аптеках.sql»)

Запрос показал аптеки с наибольшим количеством уникальных клиентов. Эти данные помогут идентифицировать аптеки с наиболее широкой клиентской базой, что может быть полезно для оценки их рыночного влияния и планирования будущих стратегий, таких как целевые рекламные кампании или расширение ассортимента.

Наибольшее количество клиентов в аптеке – Столичная.

-- Шаг 1: Объединяем таблицы заказов и клиентов

WITH customer\_counts AS (

SELECT

p.pharmacy\_name,

-- Имя аптеки

COUNT(DISTINCT p.customer\_id) AS unique\_customers -- Количество уникальных клиентов

FROM

pharma\_orders p

INNER JOIN customers c ON p.customer\_id = c.customer\_id -- Соединяем по ID клиента

GROUP BY

p.pharmacy\_name -- Группируем по аптекам

) -- Шаг 2: Выводим результат с количеством уникальных клиентов, отсортированным по убыванию

SELECT

pharmacy\_name,

unique\_customers

FROM

customer\_counts

ORDER BY

unique\_customers DESC;

```
1 -- Шаг 1: Объединяем таблицы заказов и клиентов
2 WITH customer_counts AS (
3     SELECT
4         p.pharmacy_name,
5         -- Имя аптеки
6         COUNT(DISTINCT p.customer_id) AS unique_customers -- Количество уникальных клиентов
7     FROM
8         pharma_orders p
9     INNER JOIN customers c ON p.customer_id = c.customer_id -- Соединяем по ID клиента
10    GROUP BY
11        p.pharmacy_name -- Группируем по аптекам
12    ) -- Шаг 2: Выводим результат с количеством уникальных клиентов, отсортированным по убыванию
13 SELECT
14     pharmacy_name,
15     unique_customers
16 FROM
17     customer_counts
18 ORDER BY
19     unique_customers DESC;
20
```

pharmacy_name	unique_customers
Столичная	441
Аптека.ру	437
Доктор Айболит	430
Здравсити	428
Аптека №1	427
Горздрав	426

## 6. Запрос "Лучшие клиенты"

- Соединить таблицы заказов и клиентов (JOIN)
- Посчитать тотал сумму заказов для каждого клиента
- Проранжировать клиентов по убыванию суммы заказа (row\_number)
- Оставить топ-10 клиентов

(название файла: «Ч\_1 п\_6 Лучшие клиенты.sql»)

Запрос "Лучшие клиенты" позволяет определить клиентов, которые совершили наибольшие покупки. Он выделяет 10 клиентов с наибольшей суммой заказов, что помогает бизнесу сосредоточить внимание на наиболее ценными клиентах. Эти данные могут быть использованы для разработки стратегий лояльности, таргетированных предложений и программ вознаграждений, что в свою очередь может способствовать увеличению продаж и укреплению отношений с ключевыми клиентами.

```
-- Шаг 1: Соединяем таблицы заказов и клиентов
WITH customer_orders AS (
  SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.second_name,
    o.price * o.count AS order_amount -- Вычисляем сумму заказа
  FROM
    pharma_orders o
    INNER JOIN customers c ON o.customer_id = c.customer_id
),
-- Шаг 2: Суммируем заказы для каждого клиента
customer_totals AS (
  SELECT
    customer_id,
    CONCAT(
      last_name, ', ', first_name, ', ', second_name
    ) AS full_name, -- Формируем ФИО
    SUM(order_amount) AS total_amount -- Суммируем сумму заказов для каждого клиента
  FROM
    customer_orders
  GROUP BY
    customer_id,
    full_name
),
-- Шаг 3: Присваиваем ранг каждому клиенту по убыванию суммы заказов
ranked_customers AS (
  SELECT
    customer_id,
    full_name,
    total_amount,
    ROW_NUMBER() OVER (
      ORDER BY
        total_amount DESC
    ) AS rank -- Присваиваем ранг клиентам
  FROM
    customer_totals
) -- Шаг 4: Выбираем топ-10 клиентов
SELECT
  rank,
  full_name,
  total_amount
```

```

FROM
  ranked_customers
WHERE
  rank <= 10 -- Оставляем только топ-10 клиентов
ORDER BY
  rank;

```

```

1 -- Шаг 1: Соединяем таблицы заказов и клиентов
2 WITH customer_orders AS (
3   SELECT
4     c.customer_id,
5     c.first_name,
6     c.last_name,
7     c.second_name,
8     o.price * o.count AS order_amount -- Вычисляем сумму заказа
9   FROM
10    pharma_orders o
11   INNER JOIN customers c ON o.customer_id = c.customer_id
12 ),
13 -- Шаг 2: Суммируем заказы для каждого клиента
14 customer_totals AS (
15   SELECT
16     customer_id,
17     CONCAT(
18       last_name, ', ', first_name, ' ', second_name
19     ) AS full_name,
20     -- Формируем ФИО
21     SUM(order_amount) AS total_amount -- Суммируем сумму заказов для каждого клиента
22   FROM
23    customer_orders
24   GROUP BY
25    customer_id,
26    full_name
27 ),
28 -- Шаг 3: Присваиваем ранг каждому клиенту по убыванию суммы заказов
29 ranked_customers AS (
30   SELECT
31     customer_id,
32     full_name,
33     total_amount,
34     ROW_NUMBER() OVER (
35       ORDER BY
36         total_amount DESC
37     ) AS RANK -- Присваиваем ранг клиентам
38   FROM
39    customer_totals
40 ) -- Шаг 4: Выбираем топ-10 клиентов
41 SELECT
42   RANK,
43   full_name,
44   total_amount
45 FROM
46   ranked_customers
47 WHERE
48   RANK <= 10 -- Оставляем только топ-10 клиентов
49 ORDER BY
50   RANK;
51

```

rank	full_name	total_amount
1	Буров, Бажен Германович	45174
2	Ершов, Велимир Устинович	45100
3	Григорьева, Ульяна Тимофеевна	43870
4	Максимов, Остап Гертрудович	43790
5	Брагина, Акулина Оскаровна	41731
6	Коновалова, Олимпиада Семеновна	40376
7	Воронова, Лукия Афанасьевна	39863
8	Исаков, Исмаил Всеволодович	39753
9	Емельянов, Борис Елисеевич	39030
10	Некрасов, Мстислав Ефремович	38406

## 7. Запрос "Накопленная сумма по клиентам"

- Соединить таблицы заказов и клиентов
- Соединить ФИО в одно поле
- Рассчитать накопленную сумму по каждому клиенту

(название файла: «Ч\_1 п\_7 Накопленная сумма по клиентам.sql»)

Этот запрос позволяет получить список заказов клиентов с их накопленной суммой по каждому заказу, упорядоченным по дате

```
WITH customer_orders AS (  
    -- Соединяем таблицы заказов и клиентов  
    SELECT  
        c.customer_id,  
        c.first_name,  
        c.last_name,  
        c.second_name,  
        CONCAT(  
            COALESCE(c.last_name, ''),  
            '  
            COALESCE(c.first_name, ''),  
            '  
            COALESCE(c.second_name, '')  
        ) AS full_name,  
        -- Объединяем ФИО в одно поле  
        o.report_date,  
        o.price * o.count AS order_amount -- Рассчитываем сумму заказа  
    FROM  
        pharma_orders o  
        INNER JOIN customers c ON o.customer_id = c.customer_id  
),  
cumulative_sales AS (  
    -- Рассчитываем накопленную сумму по каждому клиенту по датам  
    SELECT  
        customer_id,  
        full_name,  
        report_date,  
        order_amount,  
        SUM(order_amount) OVER (  
            PARTITION BY customer_id  
            ORDER BY  
                report_date ASC  
        ) AS cumulative_amount -- Накопленная сумма по клиенту  
    FROM  
        customer_orders  
)  
SELECT  
    ROW_NUMBER() OVER (  
        PARTITION BY customer_id  
        ORDER BY  
            report_date ASC  
    ) AS order_rank,  
    -- Присваиваем порядковый номер заказа клиента по дате  
    customer_id,  
    full_name,  
    report_date,  
    order_amount,  
    cumulative_amount
```



```

FROM
    cumulative_sales
ORDER BY
    customer_id,
    report_date ASC;

```

```

1 WITH customer_orders AS (
2   -- Соединяем таблицы заказов и клиентов
3   SELECT
4     C.customer_id,
5     C.first_name,
6     C.last_name,
7     C.second_name,
8     CONCAT(
9       COALESCE(C.last_name, ''),
10      ' ',
11      COALESCE(C.first_name, ''),
12      ' ',
13      COALESCE(C.second_name, '')
14    ) AS full_name,
15    -- Объединяем ФИО в одно поле
16    O.report_date,
17    O.price * O.count AS order_amount -- Рассчитываем сумму заказа
18  FROM
19    pharma_orders O
20  INNER JOIN customers C ON O.customer_id = C.customer_id
21 ),
22 cumulative_sales AS (
23   -- Рассчитываем накопленную сумму по каждому клиенту по датам
24   SELECT
25     customer_id,
26     full_name,
27     report_date,
28     order_amount,
29     SUM(order_amount) OVER (
30       PARTITION BY customer_id
31       ORDER BY
32         report_date ASC
33     ) AS cumulative_amount -- Накопленная сумма по клиенту
34  FROM
35    customer_orders
36 )
37 SELECT
38   ROW_NUMBER() OVER (
39     PARTITION BY customer_id
40     ORDER BY
41       report_date ASC
42   ) AS order_rank,
43   -- Присваиваем порядковый номер заказа клиента по дате
44   customer_id,
45   full_name,
46   report_date,
47   order_amount,
48   cumulative_amount
49 FROM
50   cumulative_sales
51 ORDER BY
52   customer_id,
53   report_date ASC;
54

```

i	order_rank	customer_id	full_name	report_date	order_amount	cumulative_amount
1	1	32156638	Федосеева Лукия Феликсовна	2024-02-19	636	636
2	2	32156638	Федосеева Лукия Феликсовна	2024-02-26	3552	4188
3	3	32156638	Федосеева Лукия Феликсовна	2024-03-22	2415	6603
4	4	32156638	Федосеева Лукия Феликсовна	2024-04-11	2250	8853
5	5	32156638	Федосеева Лукия Феликсовна	2024-04-28	1768	10621
6	6	32156638	Федосеева Лукия Феликсовна	2024-05-28	786	11407
1	1	32343439	Пахомов Орест Всеволодович	2024-02-13	1350	1350
2	2	32343439	Пахомов Орест Всеволодович	2024-02-18	3592	4942
3	3	32343439	Пахомов Орест Всеволодович	2024-02-19	2800	7742
4	4	32343439	Пахомов Орест Всеволодович	2024-02-22	1494	9236
5	5	32343439	Пахомов Орест Всеволодович	2024-03-22	1640	10876
6	6	32343439	Пахомов Орест Всеволодович	2024-03-30	762	11638
7	7	32343439	Пахомов Орест Всеволодович	2024-04-29	3065	14703
8	8	32343439	Пахомов Орест Всеволодович	2024-04-30	1845	16548
9	9	32343439	Пахомов Орест Всеволодович	2024-06-01	931	17479
1	1	32419148	Никитин Данила Арсеньевич	2024-02-04	2440	2440

## 8. Запрос "Самые частые клиенты аптек Горздрав и Здравсити"

- Сделать две временные таблицы: для аптеки Горздрав и Здравсити (WITH)
- Внутри каждой соединить таблицы заказов и клиентов (JOIN)
- Внутри каждой привести данные в формат "клиент - кол-во заказов в аптеке"
- Внутри каждой оставить топ 10 клиентов каждой аптеки
- Объединить клиентов с помощью UNION

(название файла: «Ч\_1 п\_8 Частые клиенты Горздрав Здравсити.sql»)

Этот запрос позволяет получить топ-10 клиентов для каждой из двух аптек с наибольшим количеством заказов и объединить их в едином результате. Эти данные помогут управлению аптек лучше понимать свою клиентскую базу, улучшать маркетинговые стратегии и повышать клиентскую лояльность.

```
WITH gorzdrav_customers AS (  
  -- Временная таблица для аптеки Горздрав  
  SELECT  
    c.customer_id,  
    CONCAT(  
      COALESCE(c.last_name, ''),  
      '  
'  
      COALESCE(c.first_name, ''),  
      '  
'  
      COALESCE(c.second_name, '')  
    ) AS full_name,  
    COUNT(o.order_id) AS order_count  
  FROM  
    pharma_orders o  
    INNER JOIN customers c ON o.customer_id = c.customer_id  
  WHERE  
    o.pharmacy_name = 'Горздрав'  
  GROUP BY  
    c.customer_id,  
    c.last_name,  
    c.first_name,  
    c.second_name  
  ORDER BY  
    order_count DESC  
  LIMIT  
    10  
, zdravsiti_customers AS (  
  -- Временная таблица для аптеки Здравсити  
  SELECT  
    c.customer_id,  
    CONCAT(  
      COALESCE(c.last_name, ''),  
      '  
'  
      COALESCE(c.first_name, ''),  
      '  
'  
      COALESCE(c.second_name, '')  
    ) AS full_name,  
    COUNT(o.order_id) AS order_count  
  FROM  
    pharma_orders o  
    INNER JOIN customers c ON o.customer_id = c.customer_id  
  WHERE  
    o.pharmacy_name = 'Здравсити'  
  GROUP BY
```

```

        c.customer_id,
        c.last_name,
        c.first_name,
        c.second_name
ORDER BY
    order_count DESC
LIMIT
    10
) -- Объединяем данные из обеих аптек
SELECT
    'Горздрав' AS pharmacy_name,
    full_name,
    order_count
FROM
    gorzdrav_customers
UNION ALL
SELECT
    'Здравсити' AS pharmacy_name,
    full_name,
    order_count
FROM
    zdravsiti_customers
ORDER BY
    pharmacy_name,
    order_count DESC;

```

```

1 WITH gorzdrav_customers AS (
2   -- Временная таблица для аптеки Горздрав
3   SELECT
4     c.customer_id,
5     CONCAT(
6       COALESCE(c.last_name, ''),
7       ' ',
8       COALESCE(c.first_name, ''),
9       ' ',
10      COALESCE(c.second_name, '')
11    ) AS full_name,
12    COUNT(o.order_id) AS order_count
13  FROM
14    pharma_orders o
15    INNER JOIN customers c ON o.customer_id = c.customer_id
16  WHERE
17    o.pharmacy_name = 'Горздрав'
18  GROUP BY
19    c.customer_id,
20    c.last_name,
21    c.first_name,
22    c.second_name
23  ORDER BY
24    order_count DESC
25  LIMIT
26    10
27 ), zdravsiti_customers AS (
28   -- Временная таблица для аптеки Здравсити
29   SELECT
30     c.customer_id,
31     CONCAT(
32       COALESCE(c.last_name, ''),
33       ' ',
34       COALESCE(c.first_name, ''),
35       ' ',
36       COALESCE(c.second_name, '')
37    ) AS full_name,
38    COUNT(o.order_id) AS order_count
39  FROM
40    pharma_orders o
41    INNER JOIN customers c ON o.customer_id = c.customer_id
42  WHERE
43    o.pharmacy_name = 'Здравсити'
44  GROUP BY
45    c.customer_id,
46    c.last_name,
47    c.first_name,
48    c.second_name
49  ORDER BY
50    order_count DESC
51  LIMIT
52    10

```

```

53 ) -- Объединяем данные из обеих аптек
54 SELECT
55     'Горздрав' AS pharmacy_name,
56     full_name,
57     order_count
58 FROM
59     gorzdrav_customers
60 UNION ALL
61 SELECT
62     'Здравсити' AS pharmacy_name,
63     full_name,
64     order_count
65 FROM
66     zdravsiti_customers
67 ORDER BY
68     pharmacy_name,
69     order_count DESC;
70
71

```

pharmacy_name	full_name	order_count
Горздрав	Беляев Клавдий Измаилович	8
Горздрав	Филиппова Майя Максимовна	7
Горздрав	Молчанов Богдан Зиновьевич	6
Горздрав	Гущин Всемир Елисеевич	6
Горздрав	Волков Викторин Гордеевич	6
Горздрав	Симонова Евдокия Ильинична	6
Горздрав	Пестова Майя Мироновна	5
Горздрав	Коновалова Олимпиада Семеновна	5
Горздрав	Дементьев Владимир Всеволодович	5
Горздрав	Смирнов Измаил Вячеславович	5
Здравсити	Мельников Софон Дмитриевич	7
Здравсити	Давыдов Аникей Алексеевич	6
Здравсити	Панова Дарья Егоровна	6
Здравсити	Ефимов Валерий Демидович	6
Здравсити	Некрасов Мстислав Ефремович	5

## ЧАСТЬ 2

### 1. Сравнение динамики продаж между Москвой и Санкт-Петербургом.

- Подсчитываем продажи лекарств по аптекам и месяцам для Москвы и
- проделяваем то же самое для Санкт-Петербурга,
- соединяем таблицы и вычисляем разницу по месяцам в процентах.
- Пишем короткие выводы в комментариях.

(название файла: «Ч\_2 п\_1 Динамика продаж Москва\_С\_Петербург.sql»)

Этот анализ позволяет сравнить динамику продаж между Москвой и Санкт-Петербургом, выявляя различия в объеме продаж по месяцам. Это помогает определить, какие города или аптеки имеют лучшие или худшие результаты и предоставляет информацию для дальнейшего принятия бизнес-решений, таких как оптимизация маркетинговых стратегий или перераспределение ресурсов.

-- Определяем суммарные продажи по месяцам для Москвы

```
WITH sales_moscow AS (  
  SELECT  
    pharmacy_name,  
    TO_CHAR(  
      DATE_TRUNC('day', report_date :: DATE),  
      'YYYY.MM'  
    ) AS day,  
    -- Форматируем дату  
    SUM(price * count) AS total_sales  
  FROM  
    pharma_orders  
  WHERE  
    city = 'Москва'  
  GROUP BY  
    pharmacy_name,  
    day  
)
```

-- Определяем суммарные продажи по месяцам для Санкт-Петербурга

```
sales_spb AS (  
  SELECT  
    pharmacy_name,  
    TO_CHAR(  
      DATE_TRUNC('day', report_date :: DATE),  
      'YYYY.MM'  
    ) AS day,  
    -- Форматируем дату  
    SUM(price * count) AS total_sales  
  FROM  
    pharma_orders  
  WHERE  
    city = 'Санкт-Петербург'  
  GROUP BY  
    pharmacy_name,  
    day  
)
```

-- Объединяем данные по продажам для Москвы и Санкт-Петербурга

```
sales_comparison AS (  
  SELECT  
    m.pharmacy_name,  
    m.day AS m_day,
```

```

m.total_sales AS m_sales,
COALESCE(s.total_sales, 0) AS spb_sales -- Заменяем NULL на 0
FROM
sales_moscow m
LEFT JOIN sales_spb s ON m.pharmacy_name = s.pharmacy_name
AND m.day = s.day
) -- Вычисляем разницу в продажах по месяцам в процентах
SELECT
pharmacy_name,
m_day,
m_sales,
spb_sales,
CASE WHEN spb_sales = 0 THEN NULL ELSE ROUND(
(
(m_sales - spb_sales) / NULLIF(spb_sales, 0):: NUMERIC
) * 100,
2
) END AS sales_difference_percentage
FROM
sales_comparison
ORDER BY
pharmacy_name,
m_day;

```

```

1 -- Определяем суммарные продажи по месяцам для Москвы
2 WITH sales_moscow AS (
3     SELECT
4         pharmacy_name,
5         TO_CHAR(
6             DATE_TRUNC('day', report_date :: DATE),
7             'YYYY.MM'
8         ) AS DAY,
9         -- Форматируем дату
10        SUM(price * COUNT) AS total_sales
11    FROM
12        pharma_orders
13    WHERE
14        city = 'Москва'
15    GROUP BY
16        pharmacy_name,
17        DAY
18 ),
19 -- Определяем суммарные продажи по месяцам для Санкт-Петербурга
20 sales_spb AS (
21     SELECT
22         pharmacy_name,
23         TO_CHAR(
24             DATE_TRUNC('day', report_date :: DATE),
25             'YYYY.MM'
26         ) AS DAY,
27         -- Форматируем дату
28        SUM(price * COUNT) AS total_sales
29    FROM
30        pharma_orders
31    WHERE
32        city = 'Санкт-Петербург'
33    GROUP BY
34        pharmacy_name,
35        DAY
36 ),
37 -- Объединяем данные по продажам для Москвы и Санкт-Петербурга
38 sales_comparison AS (
39     SELECT
40         m.pharmacy_name,
41         m.day AS m_day,
42         m.total_sales AS m_sales,
43         COALESCE(s.total_sales, 0) AS spb_sales -- Заменяем NULL на 0
44    FROM
45        sales_moscow m
46    LEFT JOIN sales_spb s ON m.pharmacy_name = s.pharmacy_name
47    AND m.day = s.day
48 ) -- Вычисляем разницу в продажах по месяцам в процентах
49 SELECT
50     pharmacy_name,
51     m_day,
52     m_sales,
53     spb_sales,
54     CASE WHEN spb_sales = 0 THEN NULL ELSE ROUND(
55         (
56             (m_sales - spb_sales) / NULLIF(spb_sales, 0):: NUMERIC
57         ) * 100,

```

```

58     2
59 ) END AS sales_difference_percentage
60 FROM
61 sales_comparison
62 ORDER BY
63 pharmacy_name,
64 m_day;
65

```

i	pharmacy_name	m_day	m_sales	spb_sales	sales_difference_percentage
	Столичная	2024.02	181765	217820	-16.55
	Столичная	2024.03	251336	219617	14.44
	Столичная	2024.04	259055	233965	10.72
	Столичная	2024.05	254867	249626	2.10
	Столичная	2024.06	15922	12510	27.27
	Аптека №1	2024.02	181081	188197	-3.78
	Аптека №1	2024.03	215738	180926	19.24
	Аптека №1	2024.04	191552	247743	-22.68
	Аптека №1	2024.05	263626	279647	-5.73
	Аптека №1	2024.06	5959	2012	196.17
	Аптека.ру	2024.02	131204	177522	-26.09
	Аптека.ру	2024.03	207523	212734	-2.45
	Аптека.ру	2024.04	243124	242150	0.40

## 2. Лекарства от насморка.

- Выделяем препараты, начинающиеся со слова “аква” (с использованием оператора LIKE).
- Приводим данные к нижнему регистру, группируем и подсчитываем общий объем продаж для каждого препарата,
- ранжируем по убыванию объема продаж и подсчитываем долю продаж каждого лекарства в общем объеме.

(название файла: «Ч\_2 п\_2 Лекарства от насморка.sql»)

Запрос позволяет определить популярность препаратов от насморка, начинающихся со слова "аква". Эти данные помогут эффективно управлять ассортиментом и улучшать маркетинговые стратегии для препаратов от насморка.

Группы лекарств от насморка начинающиеся со слов ‘аква’ отфильтрованы, наибольший уд.вес составили продажи в аптеке «Столичная» с препаратом «Аквафорте» (8.6%).

-- Шаг 1: Создаем временную таблицу для расчета суммы продаж по каждому препарату и аптеке

```

WITH pharma_drug AS (
  SELECT
    pharmacy_name,
    drug,
    SUM(price * count) AS order_amnt
  FROM
    pharma_orders
  WHERE
    LOWER(drug) LIKE '%аква%'
  GROUP BY
    pharmacy_name,
    drug
),

```

-- Шаг 2: Создаем временную таблицу для расчета общей суммы продаж по каждой аптеке

```

pharma_total_sales AS (
  SELECT
    pharmacy_name,
    SUM(price * count) AS total_namepharmacy_sales
  FROM
    pharma_orders
  GROUP BY

```

```

    pharmacy_name
) -- Шаг 3: Основной запрос для объединения данных из временных таблиц и расчета долей
продаж
SELECT
    pharma_drug.pharmacy_name,
    pharma_drug.drug,
    ROW_NUMBER () OVER (
        PARTITION BY pharma_drug.pharmacy_name
        ORDER BY
            pharma_drug.order_amnt DESC
    ) AS drug_rang,
    pharma_drug.order_amnt,
    pharma_total_sales.total_namepharmacy_sales,
    ROUND(
        (
            pharma_drug.order_amnt :: NUMERIC * 100 / pharma_total_sales.total_namepharmacy_sales ::
NUMERIC
        ),
        1
    ) AS drug_in_pharmacy_share
FROM
    pharma_drug
    INNER JOIN pharma_total_sales ON pharma_drug.pharmacy_name =
pharma_total_sales.pharmacy_name
ORDER BY
    pharma_drug.pharmacy_name,
    pharma_drug.order_amnt DESC;

```

```

PostgreSQL PostgreSQL.42
1 -- Шаг 1: Создаем временную таблицу для расчета суммы продаж по каждому препарату и аптеке
2 WITH pharma_drug AS (
3     SELECT
4         pharmacy_name,
5         drug,
6         SUM(price * COUNT) AS order_amnt
7     FROM
8         pharma_orders
9     WHERE
10        LOWER(drug) LIKE '%аква%'
11    GROUP BY
12        pharmacy_name,
13        drug
14 ),
15 -- Шаг 2: Создаем временную таблицу для расчета общей суммы продаж по каждой аптеке
16 pharma_total_sales AS (
17     SELECT
18         pharmacy_name,
19         SUM(price * COUNT) AS total_namepharmacy_sales
20     FROM
21         pharma_orders
22     GROUP BY
23         pharmacy_name
24 ) -- Шаг 3: Основной запрос для объединения данных из временных таблиц и расчета долей продаж
25 SELECT
26     pharma_drug.pharmacy_name,
27     pharma_drug.drug,
28     ROW_NUMBER () OVER (
29         PARTITION BY pharma_drug.pharmacy_name
30         ORDER BY
31             pharma_drug.order_amnt DESC

```



```

32 ) AS drug_rang,
33 pharma_drug.order_amnt,
34 pharma_total_sales.total_namepharmacy_sales,
35 ROUND(
36 (
37 pharma_drug.order_amnt :: NUMERIC * 100 / pharma_total_sales.total_namepharmacy_sales :: NUMERIC
38 ),
39 1
40 ) AS drug_in_pharmacy_share
41 FROM
42 pharma_drug
43 INNER JOIN pharma_total_sales ON pharma_drug.pharmacy_name = pharma_total_sales.pharmacy_name
44 ORDER BY
45 pharma_drug.pharmacy_name,
46 pharma_drug.order_amnt DESC;
47

```

i	pharmacy_...	drug	drug_rang	order_amnt	total_nameph...	drug_in_pharmacy_share
	Столичная	Аквафорте	1	162585	1896483	8.6
	Столичная	Аквафор	2	136633	1896483	7.2
	Столичная	Аквамарис	3	127690	1896483	6.7
	Столичная	Аква-нормикс	4	126463	1896483	6.7
	Аптека №1	Аквамарис	1	124468	1756481	7.1
	Аптека №1	Аквафор	2	118772	1756481	6.8
	Аптека №1	Аквафорте	3	107049	1756481	6.1
	Аптека №1	Аква-нормикс	4	106615	1756481	6.1
	Аптека.ру	Аквафор	1	112460	1743513	6.5

### 3. Вычисляем возраст клиентов

- на основе даты рождения с использованием функции для работы с датами;
- затем используем оператор CASE WHEN для расчета, кто наши клиенты.
- Описываем каждую группу: мужчины младше 30, мужчины от 30 до 45 и так далее. Подсчитываем долю продаж на каждую группу.

(название файла: «Ч\_2 п\_3 Клиенты в группах.sql»)

Анализ возрастных групп клиентов и их доли в продажах помогает лучше понять поведенческие и покупательские привычки различных сегментов. Этот анализ помогает оптимизировать маркетинговые усилия и стратегии продаж, чтобы лучше удовлетворять потребности клиентов и улучшать общие результаты бизнеса.

Клиенты ранжированы по возрастам, в каждой группе подсчитана доля продаж.

i	customer_group	cust_in_group_cnt	costr_group_amnt	total_sales	costr_group_share
	Женщины 30-45	64	1372440	10894725	12.6
	Женщины 45+	126	2710166	10894725	24.9
	Женщины младше 30	65	1298539	10894725	11.9
	Мужчины 30-45	51	1231515	10894725	11.3
	Мужчины 45+	127	2800361	10894725	25.7
	Мужчины младше 30	67	1481704	10894725	13.6

```

-- Временная таблица для распределения по группам покупателей
WITH customer_ages AS (
SELECT
customer_id,
gender,
-- Превращаем в возраст (текущая дата минус дата рождения)
EXTRACT(
YEAR
FROM
AGE(date_of_birth :: DATE)
) AS customer_age,

```

```

-- Группы покупателей
CASE WHEN gender = 'муж'
AND EXTRACT(
    YEAR
    FROM
        AGE(date_of_birth :: DATE)
) < 30 THEN 'Мужчины младше 30' WHEN gender = 'муж'
AND EXTRACT(
    YEAR
    FROM
        AGE(date_of_birth :: DATE)
) BETWEEN 30
AND 45 THEN 'Мужчины 30-45' WHEN gender = 'муж'
AND EXTRACT(
    YEAR
    FROM
        AGE(date_of_birth :: DATE)
) > 45 THEN 'Мужчины 45+' WHEN gender = 'жен'
AND EXTRACT(
    YEAR
    FROM
        AGE(date_of_birth :: DATE)
) < 30 THEN 'Женщины младше 30' WHEN gender = 'жен'
AND EXTRACT(
    YEAR
    FROM
        AGE(date_of_birth :: DATE)
) BETWEEN 30
AND 45 THEN 'Женщины 30-45' WHEN gender = 'жен'
AND EXTRACT(
    YEAR
    FROM
        AGE(date_of_birth :: DATE)
) > 45 THEN 'Женщины 45+' ELSE 'Другая группа' END AS customer_group
FROM
    customers
),
-- количество уникальных клиентов и общая сумма покупок по каждой группе
customer_groups AS (
    SELECT
        customer_group,
        COUNT(DISTINCT c_a.customer_id) AS cust_in_group_cnt,
        SUM(p_o.price * p_o.count) AS costr_group_amnt
    FROM
        customer_ages AS c_a
        INNER JOIN pharma_orders AS p_o USING (customer_id)
    GROUP BY
        customer_group
),
-- общая сумма продаж по всем заказам
total_sales AS (
    SELECT
        SUM(price * count) AS total_sales
    FROM
        pharma_orders
) -- основной запрос
SELECT
    customer_group,
    cust_in_group_cnt,
    costr_group_amnt,

```

```

total_sales.total_sales,
ROUND(
(
    costr_group_amnt :: NUMERIC * 100 / total_sales.total_sales :: NUMERIC
),
1
) AS costr_group_share
FROM
customer_groups
CROSS JOIN total_sales;

```

```

PostgreSQL PostgreSQL.36
1 -- Временная таблица для распределения по группам покупателей
2 WITH customer_ages AS (
3     SELECT
4         customer_id,
5         gender,
6         -- Превращаем в возраст (текущая дата минус дата рождения)
7         EXTRACT(
8             YEAR
9             FROM
10              AGE(date_of_birth :: DATE)
11         ) AS customer_age,
12         -- Группы покупателей
13         CASE WHEN gender = 'муж'
14         AND EXTRACT(
15             YEAR
16             FROM
17              AGE(date_of_birth :: DATE)
18         ) < 30 THEN 'Мужчины младше 30' WHEN gender = 'муж'
19         AND EXTRACT(
20             YEAR
21             FROM
22              AGE(date_of_birth :: DATE)
23         ) BETWEEN 30
24         AND 45 THEN 'Мужчины 30-45' WHEN gender = 'муж'
25         AND EXTRACT(
26             YEAR
27             FROM
28              AGE(date_of_birth :: DATE)
29         ) > 45 THEN 'Мужчины 45+' WHEN gender = 'жен'
30         AND EXTRACT(
31             YEAR
32             FROM
33              AGE(date_of_birth :: DATE)
34         ) < 30 THEN 'Женщины младше 30' WHEN gender = 'жен'
35         AND EXTRACT(
36             YEAR
37             FROM
38              AGE(date_of_birth :: DATE)
39         ) BETWEEN 30
40         AND 45 THEN 'Женщины 30-45' WHEN gender = 'жен'
41         AND EXTRACT(
42             YEAR
43             FROM
44              AGE(date_of_birth :: DATE)
45         ) > 45 THEN 'Женщины 45+' ELSE 'Другая группа' END AS customer_group
46     FROM
47         customers
48 ),
49 -- количество уникальных клиентов и общая сумма покупок по каждой группе
50 customer_groups AS (
51     SELECT
52         customer_group,
53         COUNT(DISTINCT c_a.customer_id) AS cust_in_group_cnt,
54         SUM(p_o.price * p_o.count) AS costr_group_amnt
55     FROM
56         customer_ages AS c_a
57     INNER JOIN pharma_orders AS p_o USING (customer_id)
58     GROUP BY
59         customer_group
60 ),
61 -- общая сумма продаж по всем заказам
62 total_sales AS (
63     SELECT
64         SUM(price * COUNT) AS total_sales
65     FROM
66         pharma_orders
67 ) -- основной запрос
68 SELECT
69     customer_group,
70     cust_in_group_cnt,
71     costr_group_amnt,
72     total_sales.total_sales,
73     ROUND(
74         (
75             costr_group_amnt :: NUMERIC * 100 / total_sales.total_sales :: NUMERIC

```

```
76 ),
77 1
78 ) AS constr_group_share
79 FROM
80 customer_groups
81 CROSS JOIN total_sales;
82
```

customer_group	cust_in_group_cnt	constr_group_amnt	total_sales	constr_group_share
Женщины 30-45	64	1372440	10894725	12.6
Женщины 45+	126	2710166	10894725	24.9
Женщины младше 30	65	1298539	10894725	11.9
Мужчины 30-45	51	1231515	10894725	11.3
Мужчины 45+	127	2800361	10894725	25.7
Мужчины младше 30	67	1481704	10894725	13.6