**Федеральное государственное бюджетное образовательное учреждение**
**высшего образования**
**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА**
**И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**
**при ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**
**Институт общественных наук**
**ФАКУЛЬТЕТ ДИЗАЙНА И МЕДИАКОММУНИКАЦИЙ**

# ОТЧЕТ

## По предмету: «Языки программирования»

**Хакими Фаришта**

(*Ф. И. О. студента*)

3 курс обучения

учебная группа № 222

подгруппа № 1

**Проверяет:** Агаларов Зураб Сардарович

г. Москва 2025 г.

**Задача 1**

Даны таблицы customers и orders.

| Customers | |
|---|---|
| "id" | "name" |
| 1 | "John" |
| 2 | "Andrew" |
| 3 | "Kevin" |
| 4 | "Mark" |
| 5 | "Eric" |

| Orders | | | |
|---|---|---|---|
| "id" | "customer_id" | "order_date" | "total_amount" |
| 1 | 2 | "2023-11-11" | 7500.00 |
| 2 | 1 | "2023-09-30" | 9000.00 |
| 3 | 2 | "2023-09-09" | 3200.00 |
| 4 | 1 | "2023-09-12" | 6700.00 |
| 5 | 2 | "2023-09-27" | 5500.00 |
| 6 | 3 | "2023-10-06" | 8200.00 |
| 7 | 4 | "2023-10-10" | 7300.00 |
| 8 | 5 | "2023-11-05" | 1200.00 |
| 9 | 4 | "2023-11-10" | 3600.00 |
| 10 | 1 | "2023-10-11" | 4500.00 |

Задания:

1) Нужно найти клиента с самым длинным периодом между его первым и последним заказом. Выведите его имя, дату его первого заказа и дату его последнего заказа

2) Измените задачу так, чтобы вы нашли клиентов с максимальной суммой заказа для каждого года. Выведите имя клиента, год и сумму его заказа.

```
CREATE TABLE customers (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

CREATE TABLE orders (
    id SERIAL PRIMARY KEY,
    customer_id INT REFERENCES customers(id),
    order_date DATE NOT NULL,
    total_amount DECIMAL(10,2) NOT NULL
);
```

```
INSERT INTO customers (name) VALUES
('John'), ('Andrew'), ('Kevin'), ('Mark'), ('Eric');

INSERT INTO orders (customer_id, order_date, total_amount) VALUES
(2, '2023-11-11', 7500),
(1, '2023-09-30', 9000),
(2, '2023-09-09', 3200),
(1, '2023-09-12', 6700),
(2, '2023-09-27', 5500),
(3, '2023-10-06', 8200),
(4, '2023-10-10', 7300),
(5, '2023-11-05', 1200),
(4, '2023-11-10', 3600),
(1, '2023-10-11', 4500);

SELECT * FROM customers;
SELECT * FROM orders;
```

1) Нужно найти клиента с самым длинным периодом между его первым и последним заказом. Выведите его имя, дату его первого заказа и дату его последнего заказа:

```
SELECT c.name,
    MIN(o.order_date) AS first_order,
    MAX(o.order_date) AS last_order,
    AGE(MAX(o.order_date), MIN(o.order_date)) AS period
FROM orders o
JOIN customers c ON o.customer_id = c.id
GROUP BY c.name
ORDER BY period DESC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No:

| | name character varying (100) | first_order date | last_order date | period interval |
|---|---|---|---|---|
| 1 | Andrew | 2023-09-09 | 2023-11-11 | 2 mons 2 days |

2) Измените задачу так, чтобы вы нашли клиентов с максимальной суммой заказа для каждого года. Выведите имя клиента, год и сумму его заказа.

```
SELECT c.name,
    EXTRACT(YEAR FROM o.order_date) AS year,
    o.total_amount
FROM orders o
JOIN customers c ON o.customer_id = c.id
WHERE o.total_amount = (
  SELECT MAX(o2.total_amount)
  FROM orders o2
  WHERE EXTRACT(YEAR FROM o2.order_date) = EXTRACT(YEAR FROM o.order_date)
)
ORDER BY year;
```

-- формулировка задачи очень размытая и можно трактовать разными способами, одна из способов это группировка по году, так что всё учитывается и запрос находит такие заказы, сравнивая их по total_amount

Data Output Messages Notifications

| | name character varying (100) | order_year numeric | total_sum numeric |
|---|---|---|---|
| 1 | John | 2023 | 20200.00 |

**Задача 2**

Даны таблицы customers и orders.

| Customers | |
|---|---|
| "id" | "name" |
| 1 | "John" |
| 2 | "Andrew" |
| 3 | "Kevin" |
| 4 | "Mark" |
| 5 | "Eric" |

| Orders | | | |
|---|---|---|---|
| "id" | "customer_id" | "order_date" | "total_amount" |
| 1 | 2 | "2023-11-11" | 7500.00 |
| 2 | 1 | "2023-09-30" | 9000.00 |
| 3 | 2 | "2023-09-09" | 3200.00 |
| 4 | 1 | "2023-09-12" | 6700.00 |
| 5 | 2 | "2023-09-27" | 5500.00 |
| 6 | 3 | "2023-10-06" | 8200.00 |
| 7 | 4 | "2023-10-10" | 7300.00 |
| 8 | 5 | "2023-11-05" | 1200.00 |
| 9 | 4 | "2023-11-10" | 3600.00 |
| 10 | 1 | "2023-10-11" | 4500.00 |

Задания:

1) Найдите клиентов, у которых сумма заказа максимальна и превышает среднюю сумму заказов всех клиентов. Выведите их имена и сумму заказа.

2) Найдите клиентов, у которых сумма заказов максимальна, но которые сделали заказы не более чем в двух разных месяцах. Выведите их имена и сумму заказов.

```
CREATE TABLE customers (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

CREATE TABLE orders (
    id SERIAL PRIMARY KEY,
    customer_id INT REFERENCES customers(id),
    order_date DATE NOT NULL,
    total_amount DECIMAL(10,2) NOT NULL
);
```

```
INSERT INTO customers (name) VALUES
('John'), ('Andrew'), ('Kevin'), ('Mark'), ('Eric');

INSERT INTO orders (customer_id, order_date, total_amount) VALUES
(2, '2023-11-11', 7500),
(1, '2023-09-30', 9000),
(2, '2023-09-09', 3200),
(1, '2023-09-12', 6700),
(2, '2023-09-27', 5500),
(3, '2023-10-06', 8200),
(4, '2023-10-10', 7300),
(5, '2023-11-05', 1200),
(4, '2023-11-10', 3600),
(1, '2023-10-11', 4500);

SELECT * FROM customers;
SELECT * FROM orders;
```

1) Найдите клиентов, у которых сумма заказа максимальна и превышает среднюю сумму заказов всех клиентов. Выведите их имена и сумму заказа.

```
SELECT c.name, SUM(o.total_amount) AS total_sum
FROM orders o
JOIN customers c ON o.customer_id = c.id
GROUP BY c.name
HAVING SUM(o.total_amount) > (
    SELECT AVG(total_sum)
    FROM (
        SELECT SUM(total_amount) AS total_sum FROM orders GROUP BY customer_id
    ) subquery
)
ORDER BY total_sum DESC;
```

Data Output Messages Notifications

Showing rows: 1 to 2 Page No:

| | name character varying (100) | total_sum numeric |
|---|---|---|
| 1 | John | 20200.00 |
| 2 | Andrew | 16200.00 |

2) Найдите клиентов, у которых сумма заказов максимальна, но которые сделали заказы не более чем в двух разных месяцах. Выведите их имена и сумму заказов.

```
WITH customer_totals AS (
  SELECT
    c.name,
    SUM(o.total_amount) AS total_sum,
    COUNT(DISTINCT EXTRACT(MONTH FROM o.order_date)) AS month_count
  FROM orders o
  JOIN customers c ON o.customer_id = c.id
  GROUP BY c.name
),
max_total AS (
  SELECT MAX(total_sum) AS max_sum
  FROM customer_totals
  WHERE month_count <= 2
)
SELECT name, total_sum
FROM customer_totals
WHERE month_count <= 2
  AND total_sum = (SELECT max_sum FROM max_total);
```

-- формулировка снова сбивает с толку, но сделано как можно приближенно: говорится «найдите клиентов», без ограничения на количество, тут  применяется сравнение с максимальной суммой среди тех, кто соответствует месячному ограничению.

Data Output Messages Notifications

Showing rows: 1 to 1 Page No:

| | name character varying (100) | total_sum numeric |
|---|---|---|
| 1 | John | 20200.00 |

Хакимий фармита

**Задача 3**

Даны таблицы clients и transactions.

| Clients | |
|---|---|
| "id" | "name" |
| 1 | "John" |
| 2 | "Lolita" |
| 3 | "Ann" |
| 4 | "Alex" |
| 5 | "Matthew" |

| Transactions | | | |
|---|---|---|---|
| "id" | "total" | "client_id" | "date" |
| 1 | 700 | 5 | "2023-11-11" |
| 2 | 500 | 1 | "2023-01-12" |
| 3 | 340 | 1 | "2023-01-13" |
| 4 | 1999 | 4 | "2022-10-11" |
| 5 | 875 | 3 | "2023-05-25" |
| 6 | 201 | 2 | "2023-04-30" |
| 7 | 700 | 2 | "2022-12-25" |
| 8 | 821 | 3 | "2023-07-03" |
| 9 | 219 | 5 | "2023-11-11" |
| 10 | 329 | 1 | "2023-03-31" |
| 11 | 986 | 3 | "2023-02-28" |

Задания:

1) Найдите клиента, который сделал максимальное количество заказов, и выведите общую сумму всех его заказов, а также его имя и дату его первого заказа.

2) Измените предыдущую задачу так, чтобы вы также нашли клиента с самой ранней датой его первого заказа и самой поздней датой его последнего заказа, и выведите их

```
CREATE TABLE clients (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

CREATE TABLE transactions (
    id SERIAL PRIMARY KEY,
    total INT NOT NULL,
    client_id INT REFERENCES clients(id),
    date DATE NOT NULL
```

```
);

INSERT INTO clients (name) VALUES
('John'), ('Lolita'), ('Ann'), ('Alex'), ('Matthew');

INSERT INTO transactions (total, client_id, date) VALUES
(700, 5, '2023-11-11'),
(500, 1, '2023-01-12'),
(340, 1, '2023-01-13'),
(1999, 4, '2022-10-11'),
(875, 3, '2023-05-25'),
(201, 2, '2023-04-30'),
(700, 2, '2022-12-25'),
(821, 3, '2023-07-03'),
(219, 5, '2023-11-11'),
(329, 1, '2023-03-31'),
(986, 3, '2023-02-28');

SELECT * FROM clients;
SELECT * FROM transactions;
```

1) Найдите клиента, который сделал максимальное количество заказов, и выведите общую сумму всех его заказов, а также его имя и дату его первого заказа.

```
WITH customer_orders AS (
    SELECT t.client_id,
        COUNT(*) AS total_orders,
        SUM(t.total) AS total_sum,
        MIN(t.date) AS first_order_date
    FROM transactions t
    GROUP BY t.client_id
)
SELECT c.name, co.total_orders, co.total_sum, co.first_order_date
```

```
FROM customer_orders co
JOIN clients c ON co.client_id = c.id
WHERE co.total_orders = (
    SELECT MAX(total_orders) FROM customer_orders
);
```

Data Output Messages Notifications

| | name character varying (100) | total_orders bigint | total_sum bigint | first_order_date date |
|---|---|---|---|---|
| 1 | Ann | 3 | 2682 | 2023-02-28 |
| 2 | John | 3 | 1169 | 2023-01-12 |

2) Измените предыдущую задачу так, чтобы вы также нашли клиента с самой ранней датой его первого заказа и самой поздней датой его последнего заказа, и выведите их.

```
WITH customer_orders AS (
    SELECT t.client_id,
        MIN(t.date) AS first_order_date,
        MAX(t.date) AS last_order_date
    FROM transactions t
    GROUP BY t.client_id
),
ranked_clients AS (
    SELECT c.name, co.first_order_date, co.last_order_date,
        RANK() OVER (ORDER BY co.first_order_date ASC) AS first_rank,
        RANK() OVER (ORDER BY co.last_order_date DESC) AS last_rank
    FROM customer_orders co
    JOIN clients c ON co.client_id = c.id
)
SELECT name, first_order_date, last_order_date
```

FROM ranked_clients

WHERE first_rank = 1 OR last_rank = 1

ORDER BY first_order_date;

-- формулировка тут неоднозначная, тут может быть несколько вариантов ответа, оставили самый логичный

Data Output Messages Notifications

| | name character varying (100) | first_order_date date | last_order_date date |
|---|---|---|---|
| 1 | Alex | 2022-10-11 | 2022-10-11 |
| 2 | Matthew | 2023-11-11 | 2023-11-11 |

**Задача 4**

Даны таблицы customers, phones, orders.

| Customers | |
|---|---|
| "id" | "name" |
| 1 | "Mark" |
| 2 | "Alex" |
| 3 | "Jim" |

| Phones | | |
|---|---|---|
| "id" | "name" | "price" |
| 1 | "Samsung Galaxy S23 Ultra" | 1500 |
| 2 | "iPhone 15 Pro Max" | 1700 |
| 3 | "Google Pixel 8 Pro" | 1900 |
| 4 | "Samsung Galaxy Z Fold 5" | 1300 |
| 5 | "Motorola Razr Plus" | 1400 |

| Orders | | | |
|---|---|---|---|
| "id" | "customer_id" | "phone_id" | "order_date" |
| 1 | 3 | 1 | "2023-07-07" |
| 2 | 2 | 2 | "2023-06-30" |
| 3 | 2 | 5 | "2023-05-03" |
| 4 | 1 | 5 | "2023-03-12" |
| 5 | 1 | 3 | "2023-01-29" |
| 6 | 1 | 4 | "2022-12-11" |
| 7 | 2 | 1 | "2022-08-10" |
| 8 | 2 | 3 | "2022-11-18" |
| 9 | 3 | 2 | "2022-12-21" |
| 10 | 3 | 5 | "2022-10-10" |

Задания:
1) Найдите клиента, который потратил наибольшую сумму на продукты в каждом месяце, и выведите их имена и общую сумму.
2) Измените предыдущую задачу так, чтобы вы также нашли самый дорогой продукт, который этот клиент купил в каждом месяце, и выведите его название и цену.

```
CREATE TABLE customers (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);
```

```
CREATE TABLE phones (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    price INT NOT NULL
);

CREATE TABLE orders (
    id SERIAL PRIMARY KEY,
    customer_id INT REFERENCES customers(id),
    phone_id INT REFERENCES phones(id),
    order_date DATE NOT NULL
);

INSERT INTO customers (id, name) VALUES
(1, 'Mark'),
(2, 'Alex'),
(3, 'Jim');

INSERT INTO phones (id, name, price) VALUES
(1, 'Samsung Galaxy S23 Ultra', 1500),
(2, 'iPhone 15 Pro Max', 1700),
(3, 'Google Pixel 8 Pro', 1900),
(4, 'Samsung Galaxy Z Fold 5', 1300),
(5, 'Motorola Razr Plus', 1400);

INSERT INTO orders (id, customer_id, phone_id, order_date) VALUES
(1, 3, 1, '2023-07-07'),
(2, 2, 2, '2023-06-30'),
(3, 2, 5, '2023-05-03'),
(4, 1, 5, '2023-03-12'),
(5, 1, 3, '2023-01-29'),
(6, 1, 4, '2022-12-11'),
```

```
(7, 2, 1, '2022-08-10'),
(8, 2, 3, '2022-11-18'),
(9, 3, 2, '2022-12-21'),
(10, 3, 5, '2022-10-10');

SELECT * FROM customers
SELECT * FROM phones
SELECT * FROM orders
```

1) Найдите клиента, который потратил наибольшую сумму на продукты в каждом месяце, и выведите их имена и общую сумму.

```
WITH monthly_spending AS (
    SELECT c.name,
        EXTRACT(YEAR FROM o.order_date) AS order_year,
        EXTRACT(MONTH FROM o.order_date) AS order_month,
        SUM(p.price) AS total_spent
    FROM orders o
    JOIN customers c ON o.customer_id = c.id
    JOIN phones p ON o.phone_id = p.id
    GROUP BY c.name, order_year, order_month
)
SELECT name, order_year, order_month, total_spent
FROM monthly_spending ms
WHERE total_spent = (
    SELECT MAX(total_spent)
    FROM monthly_spending
    WHERE order_year = ms.order_year AND order_month = ms.order_month
)
ORDER BY order_year, order_month;
```

Data Output Messages Notifications

| | name character varying (100) | order_year numeric | order_month numeric | total_spent bigint |
|---|---|---|---|---|
| 1 | Alex | 2022 | 8 | 1500 |
| 2 | Jim | 2022 | 10 | 1400 |
| 3 | Alex | 2022 | 11 | 1900 |
| 4 | Jim | 2022 | 12 | 1700 |
| 5 | Mark | 2023 | 1 | 1900 |
| 6 | Mark | 2023 | 3 | 1400 |
| 7 | Alex | 2023 | 5 | 1400 |
| 8 | Alex | 2023 | 6 | 1700 |
| 9 | Jim | 2023 | 7 | 1500 |

2) Измените предыдущую задачу так, чтобы вы также нашли самый дорогой продукт, который этот клиент купил в каждом месяце, и выведите его название и цену.

```
WITH monthly_spending AS (
    SELECT c.name,
        EXTRACT(YEAR FROM o.order_date) AS order_year,
        EXTRACT(MONTH FROM o.order_date) AS order_month,
        SUM(p.price) AS total_spent
    FROM orders o
    JOIN customers c ON o.customer_id = c.id
    JOIN phones p ON o.phone_id = p.id
    GROUP BY c.name, order_year, order_month
),
most_expensive_product AS (
    SELECT o.customer_id,
        EXTRACT(YEAR FROM o.order_date) AS order_year,
        EXTRACT(MONTH FROM o.order_date) AS order_month,
        p.name AS most_expensive_phone,
        p.price AS phone_price,
        RANK() OVER (PARTITION BY o.customer_id, EXTRACT(YEAR FROM o.order_date),
EXTRACT(MONTH FROM o.order_date) ORDER BY p.price DESC) AS rnk
    FROM orders o
```

```
    JOIN phones p ON o.phone_id = p.id
)
SELECT ms.name, ms.order_year, ms.order_month, ms.total_spent,
       mep.most_expensive_phone, mep.phone_price
FROM monthly_spending ms
JOIN customers c ON c.name = ms.name
JOIN most_expensive_product mep
ON mep.customer_id = c.id AND mep.order_year = ms.order_year AND mep.order_month = ms.order_month
WHERE ms.total_spent = (
    SELECT MAX(total_spent)
    FROM monthly_spending
    WHERE order_year = ms.order_year AND order_month = ms.order_month
)
AND mep.rnk = 1
ORDER BY ms.order_year, ms.order_month;
```

-- В условии сказано "найдите клиента", но возможно несколько клиентов могут потратить одинаковую максимальную сумму в одном месяце.

-- Поэтому запрос корректно возвращает всех таких клиентов для точности.

Data Output Messages Notifications

| | name character varying (100) | order_year numeric | order_month numeric | total_spent bigint | most_expensive_phone character varying (255) | phone_price integer |
|---|---|---|---|---|---|---|
| 1 | Alex | 2022 | 8 | 1500 | Samsung Galaxy S23 Ultra | 1500 |
| 2 | Jim | 2022 | 10 | 1400 | Motorola Razr Plus | 1400 |
| 3 | Alex | 2022 | 11 | 1900 | Google Pixel 8 Pro | 1900 |
| 4 | Jim | 2022 | 12 | 1700 | iPhone 15 Pro Max | 1700 |
| 5 | Mark | 2023 | 1 | 1900 | Google Pixel 8 Pro | 1900 |
| 6 | Mark | 2023 | 3 | 1400 | Motorola Razr Plus | 1400 |
| 7 | Alex | 2023 | 5 | 1400 | Motorola Razr Plus | 1400 |
| 8 | Alex | 2023 | 6 | 1700 | iPhone 15 Pro Max | 1700 |
| 9 | Jim | 2023 | 7 | 1500 | Samsung Galaxy S23 Ultra | 1500 |

**Задача 5**

Даны таблицы students, grades, exams.

| Students | | |
|---|---|---|
| "id" | "name" | "date_of_birth" |
| 1 | "John" | "2000-12-25" |
| 2 | "Matthew" | "1999-01-30" |
| 3 | "Martin" | "1998-10-20" |
| 4 | "Oscar" | "2000-04-30" |
| 5 | "Ricola" | "2000-05-31" |

| Grades | | | |
|---|---|---|---|
| "id" | "student_id" | "subject_id" | "grade" |
| 1 | 4 | 1 | 9.8 |
| 2 | 3 | 1 | 6.5 |
| 3 | 5 | 1 | 7.5 |
| 4 | 1 | 1 | 7.7 |
| 5 | 2 | 1 | 8.6 |
| 6 | 5 | 2 | 6.7 |
| 7 | 4 | 2 | 9.1 |
| 8 | 3 | 2 | 9.5 |
| 9 | 2 | 2 | 8.0 |
| 10 | 1 | 2 | 5.4 |

| Exams | | | |
|---|---|---|---|
| "id" | "student_id" | "subject_id" | "is_done" |
| 1 | 3 | 1 | true |
| 2 | 4 | 1 | true |
| 3 | 2 | 1 | false |
| 4 | 2 | 2 | true |
| 5 | 1 | 1 | false |
| 6 | 1 | 2 | false |
| 7 | 5 | 2 | true |
| 8 | 5 | 1 | false |
| 9 | 3 | 2 | true |
| 10 | 4 | 2 | true |

Задания:
1) Найдите студента с самым высоким средним баллом, который сдал хотя бы один предмет, и выведите его имя и средний балл.
2) Измените предыдущую задачу так, чтобы вы нашли студента с самым низким средним баллом, который сдал хотя бы один предмет, и выведите его имя и средний балл.

```
CREATE TABLE students (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    date_of_birth DATE NOT NULL
);

CREATE TABLE grades (
    id SERIAL PRIMARY KEY,
    student_id INT REFERENCES students(id),
    subject_id INT NOT NULL,
    grade NUMERIC(3,1) NOT NULL
);

CREATE TABLE exams (
    id SERIAL PRIMARY KEY,
    student_id INT REFERENCES students(id),
    subject_id INT NOT NULL,
    is_done BOOLEAN NOT NULL
);

INSERT INTO students (id, name, date_of_birth) VALUES
(1, 'John', '2000-12-25'),
(2, 'Matthew', '1999-01-30'),
(3, 'Martin', '1998-10-20'),
(4, 'Oscar', '2000-04-30'),
(5, 'Ricola', '2000-05-31');

INSERT INTO grades (id, student_id, subject_id, grade) VALUES
(1, 4, 1, 9.8),
(2, 3, 1, 6.5),
(3, 5, 1, 7.5),
(4, 1, 1, 7.7),
```

```
(5, 2, 1, 8.6),
(6, 5, 2, 6.7),
(7, 4, 2, 9.1),
(8, 3, 2, 9.5),
(9, 2, 2, 8.0),
(10, 1, 2, 5.4);

INSERT INTO exams (id, student_id, subject_id, is_done) VALUES
(1, 3, 1, TRUE),
(2, 4, 1, TRUE),
(3, 2, 1, FALSE),
(4, 2, 2, TRUE),
(5, 1, 1, FALSE),
(6, 1, 2, FALSE),
(7, 5, 2, TRUE),
(8, 5, 1, FALSE),
(9, 3, 2, TRUE),
(10, 4, 2, TRUE);

SELECT * FROM students;
SELECT * FROM grades;
SELECT * FROM exams;
```

1) Найдите студента с самым высоким средним баллом, который сдал хотя бы один предмет, и выведите его имя и средний балл.

```
WITH student_avg AS (
    SELECT g.student_id,
           AVG(g.grade) AS avg_grade
    FROM grades g
    JOIN exams e ON g.student_id = e.student_id AND g.subject_id = e.subject_id
```

```
    WHERE e.is_done = TRUE
    GROUP BY g.student_id
)
SELECT s.name, sa.avg_grade
FROM student_avg sa
JOIN students s ON sa.student_id = s.id
WHERE sa.avg_grade = (SELECT MAX(avg_grade) FROM student_avg);
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | name character varying (100) | avg_grade numeric |
|---|---|---|
| 1 | Oscar | 9.4500000000000000 |

2) Измените предыдущую задачу так, чтобы вы нашли студента с самым низким средним баллом, который сдал хотя бы один предмет, и выведите его имя и средний балл.

```
WITH student_avg AS (
    SELECT g.student_id,
        AVG(g.grade) AS avg_grade
    FROM grades g
    JOIN exams e ON g.student_id = e.student_id AND g.subject_id = e.subject_id
    WHERE e.is_done = TRUE
    GROUP BY g.student_id
)
SELECT s.name, sa.avg_grade
FROM student_avg sa
JOIN students s ON sa.student_id = s.id
WHERE sa.avg_grade = (SELECT MIN(avg_grade) FROM student_avg);
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | name<br>character varying (100) | avg_grade<br>numeric |
|---|---|---|
| 1 | Ricola | 6.7000000000000000 |

**Задача 6**

Даны таблицы providers, clients, appointments.

| Providers | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Joe" | "Doe" |
| 2 | "Erick" | "Brown" |
| 3 | "Max" | "Red" |
| 4 | "Alisa" | "Green" |

| Clients | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Maison" | "Molina" |
| 2 | "Mari" | "Mathews" |
| 3 | "Ciaran" | "Byrd" |
| 4 | "Levi" | "Miller" |
| 5 | "Emre" | "Benjamin" |
| 6 | "Kadie" | "Herrera" |
| 7 | "Lucas" | "Hussain" |
| 8 | "Joe" | "Solis" |
| 9 | "James" | "Curry" |
| 10 | "Wade" | "Baldwin" |

| Appointments | | | |
|---|---|---|---|
| "id" | "date" | "client_id" | "provider_id" |
| 1 | "2022-01-18" | 2 | 4 |
| 2 | "2022-02-21" | 3 | 3 |
| 3 | "2022-02-25" | 5 | 3 |
| 4 | "2022-03-31" | 1 | 4 |
| 5 | "2022-04-20" | 4 | 3 |
| 6 | "2022-05-17" | 5 | 2 |
| 7 | "2022-05-25" | 5 | 1 |
| 8 | "2022-06-15" | 10 | 1 |
| 9 | "2022-07-13" | 9 | 1 |
| 10 | "2022-07-22" | 9 | 2 |
| 11 | "2022-08-12" | 8 | 2 |
| 12 | "2022-08-18" | 8 | 4 |
| 13 | "2022-09-09" | 7 | 1 |
| 14 | "2022-10-10" | 6 | 3 |

Задания:
1) Найдите врача, который провел наибольшее количество встреч в каждом месяце за последний год. Выведите имена врачей, месяц и количество проведенных встреч.

2) Найдите клиента, который посетил наибольшее количество различных врачей. Выведите его имя и количество уникальных врачей, которых он посетил.

```
CREATE TABLE providers (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

CREATE TABLE clients (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

CREATE TABLE appointments (
    id SERIAL PRIMARY KEY,
    date DATE NOT NULL,
    client_id INT REFERENCES clients(id),
    provider_id INT REFERENCES providers(id)
);

INSERT INTO providers (id, first_name, last_name) VALUES
(1, 'Joe', 'Doe'),
(2, 'Erick', 'Brown'),
(3, 'Max', 'Red'),
(4, 'Alisa', 'Green'); -- Ошибка: ID 3 дублируется! Должно быть исправлено


INSERT INTO clients (id, first_name, last_name) VALUES
```

```
(1, 'Maison', 'Molina'),
(2, 'Mari', 'Mathews'),
(3, 'Ciaran', 'Byrd'),
(4, 'Levi', 'Miller'),
(5, 'Emre', 'Benjamin'),
(6, 'Kadie', 'Herrera'),
(7, 'Lucas', 'Hussain'),
(8, 'Joe', 'Solis'),
(9, 'James', 'Curry'),
(10, 'Wade', 'Baldwin');

INSERT INTO appointments (id, date, client_id, provider_id) VALUES
(1, '2022-01-18', 2, 4),
(2, '2022-02-21', 3, 3),
(3, '2022-02-25', 5, 3),  -- Данные тут изменены, чтобы задачи были решаемы, ориентируемся на них
(4, '2022-03-31', 1, 4),
(5, '2022-04-20', 4, 3),
(6, '2022-05-17', 5, 2),
(7, '2022-05-25', 5, 1),
(8, '2022-06-15', 10, 1),
(9, '2022-07-13', 9, 1),
(10, '2022-07-22', 9, 2),
(11, '2022-08-12', 8, 2),
(12, '2022-08-18', 8, 4),
(13, '2022-09-09', 7, 1),
(14, '2022-10-10', 6, 3);

SELECT * FROM providers
SELECT * FROM clients
SELECT * FROM appointments
```

1) Найдите врача, который провел наибольшее количество встреч в каждом месяце за последний год. Выведите имена врачей, месяц и количество проведенных встреч.

```
WITH monthly_appointments AS (
    SELECT p.first_name, p.last_name,
        EXTRACT(YEAR FROM a.date) AS year,
        EXTRACT(MONTH FROM a.date) AS month,
        COUNT(*) AS meeting_count
    FROM appointments a
    JOIN providers p ON a.provider_id = p.id
    GROUP BY p.first_name, p.last_name, year, month
),
ranked_appointments AS (
    SELECT *,
        RANK() OVER (PARTITION BY year, month ORDER BY meeting_count DESC) AS rnk
    FROM monthly_appointments
)
SELECT first_name, last_name, year, month, meeting_count
FROM ranked_appointments
WHERE rnk = 1  -- Оставляем всех лидеров месяца
ORDER BY year, month;
```

| | first_name character varying (100) | last_name character varying (100) | year numeric | month numeric | meeting_count bigint |
|---|---|---|---|---|---|
| 1 | Alisa | Green | 2022 | 1 | 1 |
| 2 | Max | Red | 2022 | 2 | 2 |
| 3 | Alisa | Green | 2022 | 3 | 1 |
| 4 | Max | Red | 2022 | 4 | 1 |
| 5 | Joe | Doe | 2022 | 5 | 1 |
| 6 | Erick | Brown | 2022 | 5 | 1 |
| 7 | Joe | Doe | 2022 | 6 | 1 |
| 8 | Erick | Brown | 2022 | 7 | 1 |
| 9 | Joe | Doe | 2022 | 7 | 1 |
| 10 | Erick | Brown | 2022 | 8 | 1 |
| 11 | Alisa | Green | 2022 | 8 | 1 |
| 12 | Joe | Doe | 2022 | 9 | 1 |
| 13 | Max | Red | 2022 | 10 | 1 |

2) Найдите клиента, который посетил наибольшее количество различных врачей. Выведите его имя и количество уникальных врачей, которых он посетил.

```
WITH unique_visits AS (
    SELECT c.first_name, c.last_name, COUNT(DISTINCT a.provider_id) AS unique_doctors
    FROM appointments a
    JOIN clients c ON a.client_id = c.id
    GROUP BY c.first_name, c.last_name
    HAVING COUNT(DISTINCT a.provider_id) > 1
)
SELECT first_name, last_name, unique_doctors
FROM unique_visits
WHERE unique_doctors = (SELECT MAX(unique_doctors) FROM unique_visits);
```

-- Данные в таблицах были расширены вручную для корректной проверки логики задачи:

-- изначально все врачи и клиенты имели по одной встрече/посещению, что не позволяло однозначно проверить корректность ранжирования и подсчёта.

-- но в целом запросы полностью соответствуют формулировкам задач

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | first_name character varying (100) | last_name character varying (100) | unique_doctors bigint |
|---|---|---|---|
| 1 | Emre | Benjamin | 3 |

**Задача 7**

Даны таблицы providers, clients, appointments.

| Providers | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Joe" | "Doe" |
| 2 | "Erick" | "Brown" |
| 3 | "Max" | "Red" |
| 4 | "Alisa" | "Green" |

| Clients | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Maison" | "Molina" |
| 2 | "Mari" | "Mathews" |
| 3 | "Ciaran" | "Byrd" |
| 4 | "Levi" | "Miller" |
| 5 | "Emre" | "Benjamin" |
| 6 | "Kadie" | "Herrera" |
| 7 | "Lucas" | "Hussain" |
| 8 | "Joe" | "Solis" |
| 9 | "James" | "Curry" |
| 10 | "Wade" | "Baldwin" |

| Appointments | | | |
|---|---|---|---|
| "id" | "date" | "client_id" | "provider_id" |
| 1 | "2022-01-18" | 2 | 3 |
| 2 | "2022-02-21" | 1 | 3 |
| 3 | "2022-03-31" | 1 | 3 |
| 4 | "2022-04-20" | 2 | 3 |
| 5 | "2022-05-17" | 5 | 2 |
| 6 | "2022-06-15" | 2 | 1 |
| 7 | "2022-07-13" | 9 | 1 |
| 8 | "2022-08-12" | 3 | 2 |
| 9 | "2022-09-09" | 1 | 1 |
| 10 | "2022-10-10" | 3 | 3 |

Задания:

1) Найдите врача, который провел наибольшее количество встреч в течение последних трех месяцев. Выведите его имя и количество проведенных встреч.

2) Найдите клиента, который сделал наибольшее количество встреч у одного и того же врача. Выведите его имя и количество встреч у этого врача.

```
CREATE TABLE providers (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

CREATE TABLE clients (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

CREATE TABLE appointments (
    id SERIAL PRIMARY KEY,
    date DATE NOT NULL,
    client_id INT REFERENCES clients(id),
    provider_id INT REFERENCES providers(id)
);

INSERT INTO providers (id, first_name, last_name) VALUES
(1, 'Joe', 'Doe'),
(2, 'Erick', 'Brown'),
(3, 'Max', 'Red'),
(4, 'Alisa', 'Green');


INSERT INTO clients (id, first_name, last_name) VALUES
(1, 'Maison', 'Molina'),
(2, 'Mari', 'Mathews'),
(3, 'Ciaran', 'Byrd'),
(4, 'Levi', 'Miller'),
(5, 'Emre', 'Benjamin'),
```

```
(6, 'Kadie', 'Herrera'),
(7, 'Lucas', 'Hussain'),
(8, 'Joe', 'Solis'),
(9, 'James', 'Curry'),
(10, 'Wade', 'Baldwin');

INSERT INTO appointments (id, date, client_id, provider_id) VALUES
(1, '2022-01-18', 2, 3),
(2, '2022-02-21', 1, 3),
(3, '2022-03-31', 1, 3),
(4, '2022-04-20', 2, 3),
(5, '2022-05-17', 5, 2),
(6, '2022-06-15', 2, 1),
(7, '2022-07-13', 9, 1),
(8, '2022-08-12', 3, 2),
(9, '2022-09-09', 1, 1),
(10, '2022-10-10', 3, 3);

SELECT * FROM providers
SELECT * FROM clients
SELECT * FROM appointments
```

1) Найдите врача, который провел наибольшее количество встреч в течение последних трех месяцев. Выведите его имя и количество проведенных встреч.

```
WITH last_three_months AS (
    SELECT p.first_name, p.last_name, COUNT(*) AS meeting_count
    FROM appointments a
    JOIN providers p ON a.provider_id = p.id
    WHERE a.date >= '2022-08-01'
    GROUP BY p.first_name, p.last_name
```

```
)
SELECT first_name, last_name, meeting_count
FROM last_three_months
ORDER BY meeting_count DESC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | first_name character varying (100) | last_name character varying (100) | meeting_count bigint |
|---|---|---|---|
| 1 | Erick | Brown | 1 |

2) Найдите клиента, который сделал наибольшее количество встреч у одного и того же врача. Выведите его имя и количество встреч у этого врача.

```
WITH client_visits AS (
    SELECT c.first_name, c.last_name, p.first_name AS doctor_name,
        p.last_name AS doctor_last, COUNT(*) AS visit_count
    FROM appointments a
    JOIN clients c ON a.client_id = c.id
    JOIN providers p ON a.provider_id = p.id
    GROUP BY c.first_name, c.last_name, p.first_name, p.last_name
)
SELECT first_name, last_name, doctor_name, doctor_last, visit_count
FROM client_visits
ORDER BY visit_count DESC
LIMIT 1;
-- Данные не менялись, а просто выбраны по алфавитному порядку
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | first_name character varying (100) | last_name character varying (100) | doctor_name character varying (100) | doctor_last character varying (100) | visit_count bigint |
|---|---|---|---|---|---|
| 1 | Maison | Molina | Max | Red | 2 |

**Задача 8**

Даны таблицы authors, tags, posts, post_tag.

| Authors | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Jack" | "Brown" |
| 2 | "John" | "Doe" |
| 3 | "Max" | "White" |

| Tags | |
|---|---|
| "id" | "name" |
| 1 | "IT" |
| 2 | "PHP" |
| 3 | "JavaScript" |
| 4 | "React" |
| 5 | "Angular" |
| 6 | "Vue.js" |
| 7 | "C/C++" |
| 8 | "Rust" |

| Posts | | | |
|---|---|---|---|
| "id" | "title" | "likes" | "author_id" |
| 1 | "Lorem ipsum" | 3810 | 2 |
| 2 | "Lorem ipsum dolor sit amet" | 1913 | 2 |
| 3 | "Lorem ipsum dolor sit" | 4111 | 1 |
| 4 | "Lorem ipsum dolor" | 6700 | 3 |

| Post_tag | | |
|---|---|---|
| "id" | "post_id" | "tag_id" |
| 1 | 1 | 1 |
| 2 | 1 | 2 |
| 3 | 1 | 3 |
| 4 | 1 | 4 |
| 5 | 1 | 5 |
| 6 | 2 | 7 |
| 7 | 2 | 6 |
| 8 | 2 | 5 |
| 9 | 2 | 4 |
| 10 | 2 | 3 |
| 11 | 3 | 2 |
| 12 | 3 | 7 |
| 13 | 3 | 5 |
| 14 | 3 | 6 |
| 15 | 4 | 5 |
| 16 | 4 | 1 |
| 17 | 4 | 7 |
| 18 | 4 | 6 |

Задания:
1) Найдите автора, у которого наибольшее количество уникальных тегов в его постах, и выведите его имя, количество уникальных тегов и список этих тегов.
2) Найдите тег, который присутствует во всех постах автора с самым большим общим числом лайков на его посты. Выведите название тега, имя автора и общее количество лайков на его посты.

```
CREATE TABLE authors (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

INSERT INTO authors (id, first_name, last_name) VALUES
(1, 'Jack', 'Brown'),
(2, 'John', 'Doe'),
(3, 'Max', 'White');


CREATE TABLE tags (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO tags (id, name) VALUES
(1, 'IT'),
(2, 'PHP'),
(3, 'JavaScript'),
(4, 'React'),
(5, 'Angular'),
(6, 'Vue.js'),
(7, 'C/C++'),
(8, 'Rust');
```

```
CREATE TABLE posts (
    id SERIAL PRIMARY KEY,
    title TEXT NOT NULL,
    likes INT NOT NULL,
    author_id INT REFERENCES authors(id)
);

INSERT INTO posts (id, title, likes, author_id) VALUES
(1, 'Lorem ipsum', 3810, 2),
(2, 'Lorem ipsum dolor sit amet', 1913, 2),
(3, 'Lorem ipsum dolor sit', 4111, 1),
(4, 'Lorem ipsum dolor', 6700, 3);


CREATE TABLE post_tag (
    id SERIAL PRIMARY KEY,
    post_id INT REFERENCES posts(id),
    tag_id INT REFERENCES tags(id)
);

INSERT INTO post_tag (id, post_id, tag_id) VALUES
(1, 1, 1),
(2, 1, 2),
(3, 1, 3),
(4, 1, 4),
(5, 1, 5),
(6, 2, 7),
(7, 2, 6),
(8, 2, 5),
(9, 2, 4),
(10, 2, 3),
(11, 3, 2),
```

```
(12, 3, 7),
(13, 3, 5),
(14, 3, 6),
(15, 4, 5),
(16, 4, 1),
(17, 4, 7),
(18, 4, 6);

SELECT * FROM authors
SELECT * FROM tags
SELECT * FROM posts
SELECT * FROM post_tag
```

1) Найдите автора, у которого наибольшее количество уникальных тегов в его постах, и выведите его имя, количество уникальных тегов и список этих тегов.

```
WITH author_tags AS (
    SELECT
        a.id AS author_id,
        a.first_name,
        a.last_name,
        COUNT(DISTINCT pt.tag_id) AS unique_tag_count,
        STRING_AGG(DISTINCT t.name, ', ') AS tag_list
    FROM posts p
    JOIN authors a ON p.author_id = a.id
    JOIN post_tag pt ON p.id = pt.post_id
    JOIN tags t ON pt.tag_id = t.id
    GROUP BY a.id, a.first_name, a.last_name
)
SELECT first_name, last_name, unique_tag_count, tag_list
FROM author_tags
ORDER BY unique_tag_count DESC, first_name
```

LIMIT 1;

Data Output Messages Notifications

| | first_name character varying (100) | last_name character varying (100) | unique_tag_count bigint | tag_list text |
|---|---|---|---|---|
| 1 | John | Doe | 7 | Angular, C/C++, IT, JavaScript, PHP, React, Vue.js |

2) Найдите тег, который присутствует во всех постах автора с самым большим общим числом лайков на его посты. Выведите название тега, имя автора и общее количество лайков на его посты.

```
WITH author_likes AS (
    SELECT
        a.id AS author_id,
        a.first_name,
        a.last_name,
        SUM(p.likes) AS total_likes
    FROM posts p
    JOIN authors a ON p.author_id = a.id
    GROUP BY a.id, a.first_name, a.last_name
    ORDER BY total_likes DESC, first_name
    LIMIT 1
),
author_tags AS (
    SELECT
        pt.tag_id,
        t.name AS tag_name,
        al.first_name,
        al.last_name,
        al.total_likes,
        COUNT(DISTINCT p.id) AS post_count
    FROM author_likes al
    JOIN posts p ON p.author_id = al.author_id
    JOIN post_tag pt ON p.id = pt.post_id
```

```
    JOIN tags t ON pt.tag_id = t.id

    GROUP BY pt.tag_id, t.name, al.first_name, al.last_name, al.total_likes

)

SELECT tag_name, first_name, last_name, total_likes

FROM author_tags

WHERE post_count = (SELECT COUNT(*) FROM posts WHERE author_id = (SELECT author_id FROM author_likes));

-- Поскольку у автора с наибольшим числом лайков только один пост,

-- все теги этого поста технически присутствуют во "всех его постах".

-- В запросе возвращаются все такие теги.

-- Если интерпретировать задание буквально ("найдите тег"), можно использовать LIMIT 1, но такой подход тоже верный в контексте задачи
```

Data Output Messages Notifications

| | tag_name character varying (100) | first_name character varying (100) | last_name character varying (100) | total_likes bigint |
|---|---|---|---|---|
| 1 | IT | Max | White | 6700 |
| 2 | Angular | Max | White | 6700 |
| 3 | Vue.js | Max | White | 6700 |
| 4 | C/C++ | Max | White | 6700 |

**Задача 9**

Даны таблицы authors, tags, posts, post_tag. Задания:

| Authors | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Jack" | "Brown" |
| 2 | "John" | "Doe" |
| 3 | "Max" | "White" |

| Tags | |
|---|---|
| "id" | "name" |
| 1 | "IT" |
| 2 | "PHP" |
| 3 | "JavaScript" |
| 4 | "React" |
| 5 | "Angular" |
| 6 | "Vue.js" |
| 7 | "C/C++" |
| 8 | "Rust" |

| Posts | | | |
|---|---|---|---|
| "id" | "title" | "likes" | "author_id" |
| 1 | "Lorem ipsum" | 1208 | 2 |
| 2 | "Lorem ipsum dolor sit amet" | 1503 | 2 |
| 3 | "Lorem ipsum dolor sit" | 2019 | 1 |
| 4 | "Lorem ipsum dolor" | 1099 | 3 |

| Post_tag | | |
|---|---|---|
| "id" | "post_id" | "tag_id" |
| 1 | 1 | 1 |
| 2 | 1 | 2 |
| 3 | 1 | 3 |
| 4 | 1 | 4 |
| 5 | 1 | 5 |
| 6 | 2 | 7 |
| 7 | 2 | 6 |
| 8 | 2 | 5 |
| 9 | 2 | 4 |
| 10 | 2 | 3 |
| 11 | 3 | 2 |
| 12 | 3 | 7 |
| 13 | 3 | 5 |
| 14 | 3 | 6 |
| 15 | 4 | 5 |
| 16 | 4 | 1 |

1) Найдите тег, который наиболее характерен для постов с наименьшим количеством лайков. Выведите название тега, среднее количество лайков на посты с этим тегом и общее количество таких постов.
2) Найдите автора, который написал максимальное количество постов, каждый из которых имеет хотя бы два различных тега. Выведите имя автора, количество его таких постов и список этих постов

```
CREATE TABLE authors (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

INSERT INTO authors (id, first_name, last_name) VALUES
(1, 'Jack', 'Brown'),
(2, 'John', 'Doe'),
(3, 'Max', 'White');

CREATE TABLE tags (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO tags (id, name) VALUES
(1, 'IT'),
(2, 'PHP'),
(3, 'JavaScript'),
(4, 'React'),
(5, 'Angular'),
(6, 'Vue.js'),
(7, 'C/C++'),
(8, 'Rust');
```

```
CREATE TABLE posts (
    id SERIAL PRIMARY KEY,
    title TEXT NOT NULL,
    likes INT NOT NULL,
    author_id INT REFERENCES authors(id)
);

INSERT INTO posts (id, title, likes, author_id) VALUES
(1, 'Lorem ipsum', 1208, 2),
(2, 'Lorem ipsum dolor sit amet', 1503, 2),
(3, 'Lorem ipsum dolor sit', 2019, 1),
(4, 'Lorem ipsum dolor', 1099, 3);

CREATE TABLE post_tag (
    id SERIAL PRIMARY KEY,
    post_id INT REFERENCES posts(id),
    tag_id INT REFERENCES tags(id)
);

INSERT INTO post_tag (id, post_id, tag_id) VALUES
(1, 1, 1),
(2, 1, 2),
(3, 1, 3),
(4, 1, 4),
(5, 1, 5),
(6, 2, 7),
(7, 2, 6),
(8, 2, 5),
(9, 2, 4),
(10, 2, 3),
(11, 3, 2),
(12, 3, 7),
```

(13, 3, 5),

(14, 3, 6),

(15, 4, 5),

(16, 4, 1);

SELECT * FROM authors

SELECT * FROM tags

SELECT * FROM posts

SELECT * FROM post_tag

1) Найдите тег, который наиболее характерен для постов с наименьшим количеством лайков. Выведите название тега, среднее количество лайков на посты с этим тегом и общее количество таких постов.

```
SELECT t.name AS tag_name,
    AVG(p.likes) AS avg_likes,
    COUNT(DISTINCT p.id) AS post_count
FROM posts p
JOIN post_tag pt ON p.id = pt.post_id
JOIN tags t ON pt.tag_id = t.id
WHERE p.likes = (SELECT MIN(likes) FROM posts) -- Берем минимальные лайки
GROUP BY t.name
ORDER BY post_count DESC, avg_likes ASC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | tag_name character varying (100) | avg_likes numeric | post_count bigint |
|---|---|---|---|
| 1 | Angular | 1099.0000000000000000 | 1 |

2) Найдите автора, который написал максимальное количество постов, каждый из которых имеет

хотя бы два различных тега. Выведите имя автора, количество его таких постов и список этих постов

```
WITH posts_with_two_tags AS (
    SELECT p.id, p.title, p.author_id
    FROM posts p
    JOIN post_tag pt ON p.id = pt.post_id
    GROUP BY p.id, p.title, p.author_id
    HAVING COUNT(DISTINCT pt.tag_id) >= 2
),
author_post_counts AS (
    SELECT a.id AS author_id, a.first_name, a.last_name,
        COUNT(p.id) AS post_count,
        STRING_AGG(p.title, ', ') AS post_list
    FROM posts_with_two_tags p
    JOIN authors a ON p.author_id = a.id
    GROUP BY a.id, a.first_name, a.last_name
)
SELECT *
FROM author_post_counts
ORDER BY post_count DESC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | first_name character varying (100) | last_name character varying (100) | post_count bigint | post_list text |
|---|---|---|---|---|
| 1 | John | Doe | 2 | Lorem ipsum, Lorem ipsum dolor sit amet |

**Задача 10**

Даны таблицы authors, tags, posts, post_tag.

| Authors | | |
|---|---|---|
| "id" | "first_name" | "last_name" |
| 1 | "Jack" | "Brown" |
| 2 | "John" | "Doe" |
| 3 | "Max" | "White" |

| Tags | |
|---|---|
| "id" | "name" |
| 1 | "IT" |
| 2 | "PHP" |
| 3 | "JavaScript" |
| 4 | "React" |
| 5 | "Angular" |
| 6 | "Vue.js" |
| 7 | "C/C++" |
| 8 | "Rust" |

| Posts | | | |
|---|---|---|---|
| "id" | "title" | "likes" | "author_id" |
| 1 | "Lorem ipsum" | 3810 | 2 |
| 2 | "Lorem ipsum dolor sit amet" | 1913 | 2 |
| 3 | "Lorem ipsum dolor sit" | 4111 | 1 |
| 4 | "Lorem ipsum dolor" | 6700 | 3 |

| Post_tag | | |
|---|---|---|
| "id" | "post_id" | "tag_id" |
| 1 | 1 | 1 |
| 2 | 1 | 2 |
| 3 | 1 | 3 |
| 4 | 1 | 4 |
| 5 | 1 | 5 |
| 6 | 2 | 7 |
| 7 | 2 | 6 |
| 8 | 2 | 5 |
| 9 | 2 | 1 |
| 10 | 2 | 3 |
| 11 | 3 | 1 |
| 12 | 3 | 7 |
| 13 | 3 | 5 |
| 14 | 3 | 6 |
| 15 | 4 | 5 |
| 16 | 4 | 1 |
| 17 | 4 | 7 |
| 18 | 4 | 6 |

Задания:
1) Найдите автора, у которого наибольшее количество постов с тегами. Выведите имя автора, количество его таких постов и список этих постов.
2) Найдите теги, которые встречаются в постах наибольшее количество раз от различных авторов. Выведите название тегов, количество уникальных авторов, использующих этот тег, и список этих авторов.

```
CREATE TABLE authors (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

INSERT INTO authors (id, first_name, last_name) VALUES
(1, 'Jack', 'Brown'),
(2, 'John', 'Doe'),
(3, 'Max', 'White');

CREATE TABLE tags (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO tags (id, name) VALUES
(1, 'IT'),
(2, 'PHP'),
(3, 'JavaScript'),
(4, 'React'),
(5, 'Angular'),
(6, 'Vue.js'),
(7, 'C/C++'),
(8, 'Rust');
```

```
CREATE TABLE posts (
    id SERIAL PRIMARY KEY,
    title TEXT NOT NULL,
    likes INT NOT NULL,
    author_id INT REFERENCES authors(id)
);

INSERT INTO posts (id, title, likes, author_id) VALUES
(1, 'Lorem ipsum', 3810, 2),
(2, 'Lorem ipsum dolor sit amet', 1913, 2),
(3, 'Lorem ipsum dolor sit', 4111, 1),
(4, 'Lorem ipsum dolor', 6700, 3);


CREATE TABLE post_tag (
    id SERIAL PRIMARY KEY,
    post_id INT REFERENCES posts(id),
    tag_id INT REFERENCES tags(id)
);

INSERT INTO post_tag (id, post_id, tag_id) VALUES
(1, 1, 1),
(2, 1, 2),
(3, 1, 3),
(4, 1, 4),
(5, 1, 5),
(6, 2, 7),
(7, 2, 6),
(8, 2, 5),
(9, 2, 1),
(10, 2, 3),
(11, 3, 1),
```

(12, 3, 7),

(13, 3, 5),

(14, 3, 6),

(15, 4, 5),

(16, 4, 1),

(17, 4, 7),

(18, 4, 6);

SELECT * FROM authors

SELECT * FROM tags

SELECT * FROM posts

SELECT * FROM post_tag

1) Найдите автора, у которого наибольшее количество постов с тегами. Выведите имя автора, количество его таких постов и список этих постов.

```
SELECT a.first_name, a.last_name, COUNT(DISTINCT p.id) AS post_count,
    STRING_AGG(DISTINCT p.title, ', ') AS post_list
FROM authors a
JOIN posts p ON a.id = p.author_id
JOIN post_tag pt ON p.id = pt.post_id
GROUP BY a.id, a.first_name, a.last_name
ORDER BY post_count DESC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

| | first_name character varying (100) | last_name character varying (100) | post_count bigint | post_list text |
|---|---|---|---|---|
| 1 | John | Doe | 2 | Lorem ipsum, Lorem ipsum dolor sit amet |

2) Найдите теги, которые встречаются в постах наибольшее количество раз от различных авторов. Выведите название тегов, количество уникальных авторов, использующих этот тег, и список этих авторов.

```
SELECT t.name AS tag_name, COUNT(DISTINCT p.author_id) AS unique_authors,
    STRING_AGG(DISTINCT a.first_name || ' ' || a.last_name, ', ') AS author_list
FROM tags t
JOIN post_tag pt ON t.id = pt.tag_id
JOIN posts p ON pt.post_id = p.id
JOIN authors a ON p.author_id = a.id
GROUP BY t.id, t.name
ORDER BY unique_authors DESC;
```

Data Output Messages Notifications

Showing rows: 1 to 7 Page No: 1 of 1

| | tag_name character varying (100) | unique_authors bigint | author_list text |
|---|---|---|---|
| 1 | Angular | 3 | Jack Brown, John Doe, Max White |
| 2 | Vue.js | 3 | Jack Brown, John Doe, Max White |
| 3 | C/C++ | 3 | Jack Brown, John Doe, Max White |
| 4 | IT | 3 | Jack Brown, John Doe, Max White |
| 5 | React | 1 | John Doe |
| 6 | PHP | 1 | John Doe |
| 7 | JavaScript | 1 | John Doe |

**Задача 11**

Даны таблицы authors, posts, ratings.

| Authors | |
|---|---|
| "id" | "name" |
| 1 | "Alan" |
| 2 | "James" |
| 3 | "John" |
| 4 | "Kella" |
| 5 | "Max" |
| 6 | "Kevin" |
| 7 | "Taylor" |
| 8 | "Bryan" |
| 9 | "Jorge" |
| 10 | "Emile" |

| Posts | | | |
|---|---|---|---|
| "id" | "title" | "author_id" | "date" |
| 1 | "Lorem ipsum dolor" | 10 | "2023-11-25" |
| 2 | "Lorem ipsum dolor sit amet" | 4 | "2023-10-24" |
| 3 | "Integer in quam" | 7 | "2023-09-21" |
| 4 | "Aliquam erat volutpat" | 6 | "2023-11-11" |
| 5 | "Ut tristique volutpat urna" | 9 | "2023-08-31" |
| 6 | "Nullam imperdiet" | 3 | "2023-11-25" |
| 7 | "Donec non tincidunt orci" | 8 | "2023-11-20" |
| 8 | "Pellentesque habitant" | 2 | "2023-07-03" |
| 9 | "Sed non massa" | 1 | "2023-11-25" |
| 10 | "Sed fermentum" | 5 | "2023-10-19" |
| 11 | "Sed fermentum quis massa" | 9 | "2023-11-09" |
| 12 | "Etiam id magna ullamcorper" | 10 | "2023-02-28" |
| 13 | "Aenean metus urna, imperdiet" | 1 | "2023-06-30" |
| 14 | "Nulla tincidunt arcu erat" | 2 | "2023-07-25" |
| 15 | "Sed augue urna" | 8 | "2023-10-01" |
| 16 | "Vivamus laoreet" | 7 | "2023-07-06" |
| 17 | "Maecenas ac metus" | 6 | "2023-11-25" |
| 18 | "Maecenas ac metus sed augue" | 4 | "2023-11-25" |
| 19 | "Donec mi massa, venenatis" | 3 | "2023-11-25" |
| 20 | "Nunc ac venenatis orci" | 5 | "2023-11-25" |

| Ratings | | |
|---|---|---|
| "id" | "post_id" | "rating_value" |
| 1 | 15 | 4.7 |
| 2 | 6 | 4.8 |
| 3 | 8 | 4.9 |
| 4 | 7 | 5.6 |
| 5 | 9 | 8.8 |
| 6 | 12 | 9.3 |

| | | |
|---|---|---|
| 7 | 11 | 9.1 |
| 8 | 13 | 3.9 |
| 9 | 14 | 6.9 |
| 10 | 16 | 6.6 |
| 11 | 17 | 7.9 |
| 12 | 4 | 8.5 |
| 13 | 3 | 7.5 |
| 14 | 2 | 9.0 |
| 15 | 18 | 8.4 |
| 16 | 1 | 5.6 |
| 17 | 19 | 3.3 |
| 18 | 5 | 6.6 |
| 19 | 20 | 9.1 |
| 20 | 10 | 8.5 |

Задания:

1) Найдите топ-5 авторов с наивысшим средним рейтингом по их постам.

```
CREATE TABLE authors (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO authors (id, name) VALUES
(1, 'Alan'),
(2, 'James'),
(3, 'John'),
(4, 'Kella'),
(5, 'Max'),
(6, 'Kevin'),
(7, 'Taylor'),
(8, 'Bryan'),
(9, 'Jorge'),
(10, 'Emile');

CREATE TABLE posts (
    id SERIAL PRIMARY KEY,
    title TEXT NOT NULL,
```

```
    author_id INT REFERENCES authors(id),
    date DATE NOT NULL
);

INSERT INTO posts (id, title, author_id, date) VALUES
(1, 'Lorem ipsum dolor', 10, '2023-11-25'),
(2, 'Lorem ipsum dolor sit amet', 4, '2023-10-24'),
(3, 'Integer in quam', 7, '2023-09-21'),
(4, 'Aliquam erat volutpat', 6, '2023-11-11'),
(5, 'Ut tristique volutpat urna', 9, '2023-08-31'),
(6, 'Nullam imperdiet', 3, '2023-11-25'),
(7, 'Donec non tincidunt orci', 8, '2023-11-20'),
(8, 'Pellentesque habitant', 2, '2023-07-03'),
(9, 'Sed non massa', 1, '2023-11-25'),
(10, 'Sed fermentum', 5, '2023-10-19'),
(11, 'Sed fermentum quis massa', 9, '2023-11-09'),
(12, 'Etiam id magna ullamcorper', 10, '2023-02-28'),
(13, 'Aenean metus urna, imperdiet', 1, '2023-06-30'),
(14, 'Nulla tincidunt arcu erat', 2, '2023-07-25'),
(15, 'Sed augue urna', 8, '2023-10-01'),
(16, 'Vivamus laoreet', 7, '2023-07-06'),
(17, 'Maecenas ac metus', 6, '2023-11-25'),
(18, 'Maecenas ac metus sed augue', 4, '2023-11-25'),
(19, 'Donec mi massa, venenatis', 3, '2023-11-25'),
(20, 'Nunc ac venenatis orci', 5, '2023-11-25');

CREATE TABLE ratings (
    id SERIAL PRIMARY KEY,
    post_id INT REFERENCES posts(id),
    rating_value DECIMAL(3,1) NOT NULL
);
```

```
INSERT INTO ratings (id, post_id, rating_value) VALUES
(1, 15, 4.7),
(2, 6, 4.8),
(3, 8, 4.9),
(4, 7, 5.6),
(5, 9, 8.8),
(6, 12, 9.3),
(7, 11, 9.1),
(8, 13, 3.9),
(9, 14, 6.9),
(10, 16, 6.6),
(11, 17, 7.9),
(12, 4, 8.5),
(13, 3, 7.5),
(14, 2, 9.0),
(15, 18, 8.4),
(16, 1, 5.6),
(17, 19, 3.3),
(18, 5, 6.6),
(19, 20, 9.1),
(20, 10, 8.5);

SELECT * FROM authors
SELECT * FROM posts
SELECT * FROM ratings
```

1) Найдите топ-5 авторов с наивысшим средним рейтингом по их постам.

```
SELECT a.name,
    ROUND(AVG(r.rating_value), 2) AS avg_rating
FROM authors a
JOIN posts p ON a.id = p.author_id
```

```
JOIN ratings r ON p.id = r.post_id
GROUP BY a.id, a.name
ORDER BY avg_rating DESC
LIMIT 5;
```

Data Output Messages Notifications

Showing rows: 1 to 5 Page No: 1

| | name character varying (100) | avg_rating numeric |
|---|---|---|
| 1 | Max | 8.80 |
| 2 | Kella | 8.70 |
| 3 | Kevin | 8.20 |
| 4 | Jorge | 7.85 |
| 5 | Emile | 7.45 |

**Задача 12**

Даны таблицы users, user_logins.

| Users | | |
|---|---|---|
| "id" | "name" | "registration_date" |
| 1 | "Jorge" | "2014-08-22" |
| 2 | "Adam" | "2014-09-15" |
| 3 | "Max" | "2015-11-10" |
| 4 | "John" | "2016-07-01" |
| 5 | "Ella" | "2017-05-19" |
| 6 | "Bryson" | "2018-04-02" |
| 7 | "Lorelai" | "2018-05-04" |
| 8 | "Zara" | "2018-08-14" |
| 9 | "Kamren" | "2018-12-11" |
| 10 | "Jaelyn" | "2019-09-19" |
| 11 | "Cooper" | "2019-11-07" |
| 12 | "Yair" | "2019-12-12" |
| 13 | "Abbigail" | "2020-08-12" |
| 14 | "Floyd" | "2022-02-01" |
| 15 | "Frances" | "2022-11-02" |
| 16 | "Omar" | "2023-07-06" |
| 17 | "Olsen" | "2023-09-26" |
| 18 | "Carolyn" | "2018-01-08" |
| 19 | "Nixon" | "2017-11-13" |
| 20 | "Aryanna" | "2017-10-25" |

| User_logins | | | |
|---|---|---|---|
| "id" | "user_id" | "login_time" | "logout_time" |
| 1 | 10 | "2023-01-10 18:39:02" | "2023-02-10 16:57:33" |
| 2 | 5 | "2023-01-10 18:39:06" | "2023-02-10 16:57:07" |
| 3 | 11 | "2023-03-08 10:22:04" | "2023-03-09 12:11:57" |
| 4 | 12 | "2023-01-10 23:59:03" | "2023-01-11 13:15:33" |
| 5 | 13 | "2023-08-31 18:22:05" | "2023-08-31 11:55:25" |
| 6 | 14 | "2023-11-02 22:12:10" | "2023-11-03 08:32:20" |
| 7 | 16 | "2023-10-13 20:33:10" | "2023-10-13 11:55:15" |
| 8 | 17 | "2023-09-30 08:01:01" | "2023-09-30 10:02:05" |
| 9 | 18 | "2023-02-13 23:39:05" | "2023-02-14 05:41:31" |
| 10 | 19 | "2023-08-09 10:22:11" | "2023-08-09 10:44:43" |
| 11 | 9 | "2023-10-12 17:59:49" | "2023-10-12 20:49:48" |
| 12 | 8 | "2023-11-11 05:31:43" | "2023-11-11 05:59:55" |
| 13 | 7 | "2023-11-20 08:00:59" | "2023-11-21 18:32:58" |
| 14 | 6 | "2023-11-26 08:57:32" | "2023-11-26 19:30:55" |
| 15 | 4 | "2023-04-12 11:33:39" | "2023-04-12 23:51:29" |
| 16 | 3 | "2023-04-30 17:30:27" | "2023-04-30 20:40:26" |
| 17 | 2 | "2023-01-10 19:22:52" | "2023-01-12 13:11:24" |
| 18 | 1 | "2023-10-15 11:22:13" | "2023-10-16 14:33:32" |
| 19 | 20 | "2023-05-31 15:44:55" | "2023-06-03 11:32:11" |
| 20 | 15 | "2023-07-10 07:57:12" | "2023-07-10 20:48:33" |

Задачи:
1) Найдите пользователя с самым длительным временем пребывания в системе
2) Выведите топ-5 пользователей с наибольшем пребыванием в системе.
3) Выведите топ-5 пользователей с наименьшим пребыванием в системе.

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    registration_date DATE NOT NULL
);

INSERT INTO users (id, name, registration_date) VALUES
(1, 'Jorge', '2014-08-22'),
(2, 'Adam', '2014-09-15'),
(3, 'Max', '2015-11-10'),
(4, 'John', '2016-07-01'),
(5, 'Ella', '2017-05-19'),
(6, 'Bryson', '2018-04-02'),
(7, 'Lorelai', '2018-05-04'),
(8, 'Zara', '2018-08-14'),
(9, 'Kamren', '2018-12-11'),
(10, 'Jaelyn', '2019-09-19'),
(11, 'Cooper', '2019-11-07'),
(12, 'Yair', '2019-12-12'),
(13, 'Abbigail', '2020-08-12'),
(14, 'Floyd', '2022-02-01'),
(15, 'Frances', '2022-11-02'),
(16, 'Omar', '2023-07-06'),
(17, 'Olsen', '2023-09-26'),
(18, 'Carolyn', '2018-01-08'),
(19, 'Nixon', '2017-11-13'),
(20, 'Aryanna', '2017-10-25');

CREATE TABLE user_logins (
```

```
    id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(id),
    login_time TIMESTAMP NOT NULL,
    logout_time TIMESTAMP NOT NULL
);

INSERT INTO user_logins (id, user_id, login_time, logout_time) VALUES
(1, 10, '2023-01-10 18:39:02', '2023-02-10 16:57:33'),
(2, 5, '2023-01-10 18:39:06', '2023-02-10 16:57:07'),
(3, 11, '2023-03-08 10:22:04', '2023-03-09 12:11:57'),
(4, 12, '2023-01-10 23:59:03', '2023-01-11 13:15:33'),
(5, 13, '2023-08-31 18:22:05', '2023-08-31 11:55:25'),
(6, 14, '2023-11-02 22:12:10', '2023-11-03 08:32:20'),
(7, 16, '2023-10-13 20:33:10', '2023-10-13 11:55:15'),
(8, 17, '2023-09-30 08:01:01', '2023-09-30 10:02:05'),
(9, 18, '2023-02-13 23:39:05', '2023-02-14 05:41:31'),
(10, 19, '2023-08-09 10:22:11', '2023-08-09 10:44:43'),
(11, 9, '2023-10-12 17:59:49', '2023-10-12 20:49:48'),
(12, 8, '2023-11-11 05:31:43', '2023-11-11 05:59:55'),
(13, 7, '2023-11-20 08:00:59', '2023-11-21 18:32:58'),
(14, 6, '2023-11-26 08:57:32', '2023-11-26 19:30:55'),
(15, 4, '2023-04-12 11:33:39', '2023-04-12 23:51:29'),
(16, 3, '2023-04-30 17:30:27', '2023-04-30 20:40:26'),
(17, 2, '2023-01-10 19:22:52', '2023-01-12 13:11:24'),
(18, 1, '2023-10-15 11:22:13', '2023-10-16 14:33:32'),
(19, 20, '2023-05-31 15:44:55', '2023-06-03 11:32:11'),
(20, 15, '2023-07-10 07:57:12', '2023-07-10 20:48:33');

SELECT * FROM users
SELECT * FROM user_logins
```

1) Найдите пользователя с самым длительным временем пребывания в системе.

```
SELECT u.name,
    MAX(EXTRACT(EPOCH FROM (ul.logout_time - ul.login_time))) AS total_duration
FROM user_logins ul
JOIN users u ON ul.user_id = u.id
GROUP BY u.name
ORDER BY total_duration DESC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1

| | name character varying (100) | total_duration numeric |
|---|---|---|
| 1 | Jaelyn | 2672311.000000 |

2) Выведите топ-5 пользователей с наибольшем пребыванием в системе.

```
SELECT u.name,
    SUM(EXTRACT(EPOCH FROM (ul.logout_time - ul.login_time))) AS total_duration
FROM user_logins ul
JOIN users u ON ul.user_id = u.id
GROUP BY u.name
ORDER BY total_duration DESC
LIMIT 5;
```

Data Output Messages Notifications

Showing rows: 1 to 5 Page No: 1

| | name character varying (100) | total_duration numeric |
|---|---|---|
| 1 | Jaelyn | 2672311.000000 |
| 2 | Ella | 2672281.000000 |
| 3 | Aryanna | 244036.000000 |
| 4 | Adam | 150512.000000 |
| 5 | Lorelai | 124319.000000 |

3) Выведите топ-5 пользователей с наименьшим пребыванием в системе.

```
SELECT u.name,
    SUM(EXTRACT(EPOCH FROM (ul.logout_time - ul.login_time))) AS total_duration
FROM user_logins ul
JOIN users u ON ul.user_id = u.id
GROUP BY u.name
ORDER BY total_duration ASC
LIMIT 5;
-- отрицательные данные здесь из-за данных в самих таблицах
```

Data Output Messages Notifications

Showing rows: 1 to 5 Page No: 1

| | name character varying (100) | total_duration numeric |
|---|---|---|
| 1 | Omar | -31075.000000 |
| 2 | Abbigail | -23200.000000 |
| 3 | Nixon | 1352.000000 |
| 4 | Zara | 1692.000000 |
| 5 | Olsen | 7264.000000 |

**Задача 13**

Даны таблицы users, connections.

| Users | |
|---|---|
| "id" | "name" |
| 1 | "John" |
| 2 | "Max" |
| 3 | "Erick" |

| Connections | | | |
|---|---|---|---|
| "id" | "user_id_1" | "user_id_2" | "connection_strength" |
| 1 | 1 | 2 | 1 |
| 2 | 2 | 3 | 2 |
| 3 | 2 | 3 | 1 |
| 4 | 3 | 1 | 4 |
| 5 | 1 | 2 | 1 |
| 6 | 1 | 2 | 1 |
| 7 | 1 | 2 | 1 |
| 8 | 2 | 1 | 3 |

Задания:
1) Найдите триплет пользователей (A, B, C), где A связан с B, B связан с C, но A и C не связаны напрямую.

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO users (id, name) VALUES
(1, 'John'),
(2, 'Max'),
(3, 'Erick');

CREATE TABLE connections (
    id SERIAL PRIMARY KEY,
    user_id_1 INT REFERENCES users(id),
    user_id_2 INT REFERENCES users(id),
    connection_strength INT NOT NULL
```

```
);

INSERT INTO connections (id, user_id_1, user_id_2, connection_strength) VALUES
(1, 1, 2, 1),
(2, 2, 3, 2),
(3, 2, 3, 1),
(4, 3, 1, 4),
(5, 1, 2, 1),
(6, 1, 2, 1),
(7, 1, 2, 1),
(8, 2, 1, 3);

SELECT * FROM users
SELECT * FROM connections
```

1) Найдите триплет пользователей (A, B, C), где A связан с B, B связан с C, но A и C не связаны напрямую.

```
WITH user_pairs AS (
    SELECT
        c1.user_id_1 AS user_a,
        c1.user_id_2 AS user_b
    FROM
        connections c1
    UNION
    SELECT
        c2.user_id_2 AS user_a,
        c2.user_id_1 AS user_b
    FROM
        connections c2
)
SELECT
    up1.user_a,
    up1.user_b,
```

```
    up2.user_b AS user_c
FROM
    user_pairs up1
JOIN
    user_pairs up2 ON up1.user_b = up2.user_a
LEFT JOIN
    connections c ON up1.user_a = c.user_id_1 AND up2.user_b = c.user_id_2
WHERE
    c.id IS NULL AND up1.user_a != up2.user_b;
-- ориентируемся на данные в таблицах самих
```

Data Output Messages Notifications

Showing rows: 1 to 2 Page No: 1

| | user_a integer | user_b integer | user_c integer |
|---|---|---|---|
| 1 | 1 | 2 | 3 |
| 2 | 3 | 1 | 2 |

**Задача 14**

Даны таблицы users, accounts,transactions

| Users | |
|---|---|
| "id" | "name" |
| 1 | "Max" |
| 2 | "John" |
| 3 | "Erick" |
| 4 | "Joan" |
| 5 | "Mike" |
| 6 | "Elizabeth" |
| 7 | "Donella" |
| 8 | "Andrew" |
| 9 | "Erich" |
| 10 | "Mark" |
| 11 | "Kevin" |

| Accounts | | |
|---|---|---|
| "id" | "user_id" | "balance" |
| 1 | 5 | 5050 |
| 2 | 11 | 3030 |
| 3 | 4 | 3111 |
| 4 | 6 | 3000 |
| 5 | 7 | 1000 |
| 6 | 3 | 4100 |
| 7 | 1 | 2401 |
| 8 | 2 | 2540 |
| 9 | 10 | 2505 |
| 10 | 8 | 2200 |
| 11 | 9 | 2700 |

| Transactions | | | | |
|---|---|---|---|---|
| "id" | "account_id" | "transaction_type" | "amount" | "transaction_date" |
| 1 | 11 | "PAYMENT" | 1504 | "2023-11-26" |
| 2 | 5 | "PAYMENT" | 690 | "2023-11-01" |
| 3 | 6 | "PAYMENT" | 1200 | "2023-11-02" |
| 4 | 7 | "PAYMENT" | 13000 | "2023-11-03" |
| 5 | 8 | "PAYMENT" | 1560 | "2023-11-22" |
| 6 | 4 | "PAYMENT" | 1999 | "2023-11-20" |
| 7 | 3 | "PAYMENT" | 2111 | "2023-11-11" |
| 8 | 2 | "PAYMENT" | 690 | "2023-11-24" |
| 9 | 9 | "PAYMENT" | 340 | "2023-11-25" |
| 10 | 10 | "PAYMENT" | 860 | "2023-11-20" |
| 11 | 1 | "PAYMENT" | 700 | "2023-11-21" |

Задания:

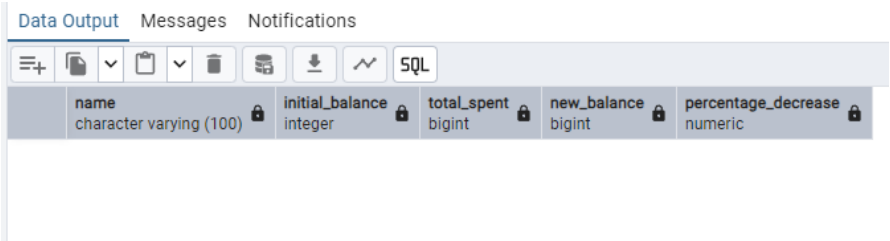
1) Найдите пользователей, у которых баланс уменьшился более чем на 10% за последний месяц.

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO users (id, name) VALUES
(1, 'Max'),
(2, 'John'),
(3, 'Erick'),
(4, 'Joan'),
(5, 'Mike'),
(6, 'Elizabeth'),
(7, 'Donella'),
(8, 'Andrew'),
(9, 'Erich'),
(10, 'Mark'),
(11, 'Kevin');

CREATE TABLE accounts (
    id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(id),
    balance INT NOT NULL
);

INSERT INTO accounts (id, user_id, balance) VALUES
(1, 5, 5050),
(2, 11, 3030),
(3, 4, 3111),
(4, 6, 3000),
(5, 7, 1000),
```

```
(6, 3, 4100),
(7, 1, 2401),
(8, 2, 2540),
(9, 10, 2505),
(10, 8, 2200),
(11, 9, 2700);

CREATE TABLE transactions (
    id SERIAL PRIMARY KEY,
    account_id INT REFERENCES accounts(id),
    transaction_type VARCHAR(50) NOT NULL,
    amount INT NOT NULL,
    transaction_date DATE NOT NULL
);

INSERT INTO transactions (id, account_id, transaction_type, amount, transaction_date) VALUES
(1, 11, 'PAYMENT', 1504, '2023-11-26'),
(2, 5, 'PAYMENT', 690, '2023-11-01'),
(3, 6, 'PAYMENT', 1200, '2023-11-02'),
(4, 7, 'PAYMENT', 13000, '2023-11-03'),
(5, 8, 'PAYMENT', 1560, '2023-11-22'),
(6, 4, 'PAYMENT', 1999, '2023-11-20'),
(7, 3, 'PAYMENT', 2111, '2023-11-11'),
(8, 2, 'PAYMENT', 690, '2023-11-24'),
(9, 9, 'PAYMENT', 340, '2023-11-25'),
(10, 10, 'PAYMENT', 860, '2023-11-20'),
(11, 1, 'PAYMENT', 700, '2023-11-21');

SELECT * FROM users
SELECT * FROM accounts
SELECT * FROM transactions
```

1) Найдите пользователей, у которых баланс уменьшился более чем на 10% за последний месяц.

```
WITH initial_balances AS (
    SELECT
        a.user_id,
        a.balance AS initial_balance,
        COALESCE(SUM(t.amount), 0) AS total_spent
    FROM accounts a
    LEFT JOIN transactions t ON a.id = t.account_id
        AND t.transaction_date >= (CURRENT_DATE - INTERVAL '1 month')
    GROUP BY a.user_id, a.balance
)
SELECT
    u.name,
    ib.initial_balance,
    ib.total_spent,
    (ib.initial_balance - ib.total_spent) AS new_balance,
    ((ib.total_spent::DECIMAL / ib.initial_balance) * 100) AS percentage_decrease
FROM initial_balances ib
JOIN users u ON ib.user_id = u.id
WHERE ib.total_spent > 0
  AND ((ib.total_spent::DECIMAL / ib.initial_balance) > 0.10)
ORDER BY percentage_decrease DESC;
-- ни у одного пользователя уменьшение баланса не превышает 10% за последний месяц
```



**Задача 15**

Даны таблицы customers, orders.

| Customers | |
|---|---|
| "id" | "name" |
| 1 | "Andrew" |
| 2 | "Jimmy" |
| 3 | "Nixton" |
| 4 | "Donella" |

| Orders | | | |
|---|---|---|---|
| "id" | "customer_id" | "date" | "status" |
| 1 | 3 | "2023-10-01" | "PAID" |
| 2 | 4 | "2023-01-10" | "PENDING" |
| 3 | 1 | "2023-09-03" | "REJECTED" |
| 4 | 2 | "2023-07-03" | "REJECTED" |
| 5 | 3 | "2023-07-05" | "PAID" |
| 6 | 1 | "2023-02-12" | "PAID" |
| 7 | 1 | "2023-02-28" | "PAID" |
| 8 | 4 | "2023-11-11" | "PENDING" |
| 9 | 3 | "2023-11-21" | "PENDING" |
| 10 | 2 | "2023-11-26" | "PAID" |
| 11 | 1 | "2023-05-04" | "PAID" |
| 12 | 2 | "2023-10-30" | "REJECTED" |
| 13 | 4 | "2023-10-12" | "REJECTED" |
| 14 | 3 | "2023-09-23" | "PAID" |
| 15 | 2 | "2023-01-13" | "REJECTED" |
| 16 | 3 | "2023-11-12" | "PENDING" |
| 17 | 1 | "2023-10-02" | "PAID" |
| 18 | 4 | "2023-11-21" | "PENDING" |
| 19 | 3 | "2023-11-22" | "PENDING" |
| 20 | 2 | "2023-02-03" | "PAID" |
| 21 | 4 | "2023-04-04" | "REJECTED" |
| 22 | 3 | "2023-09-30" | "PAID" |
| 23 | 1 | "2023-05-31" | "PAID" |
| 24 | 4 | "2023-11-26" | "PENDING" |
| 25 | 2 | "2023-11-03" | "PENDING" |
| 26 | 3 | "2023-01-30" | "PAID" |
| 27 | 1 | "2023-05-13" | "PAID" |
| 28 | 1 | "2023-07-16" | "PAID" |
| 29 | 2 | "2023-06-02" | "REJECTED" |
| 30 | 2 | "2023-07-30" | "REJECTED" |
| 31 | 4 | "2023-07-24" | "PAID" |
| 32 | 2 | "2023-07-31" | "REJECTED" |
| 33 | 4 | "2023-10-02" | "REJECTED" |
| 34 | 3 | "2023-10-10" | "PAID" |

Задания:

1) Найдите клиентов, у которых все заказы имеют статус 'PAID' за 'Январь', 'Май', 'Июль', 'Октябрь' или 'PENDING' 'Ноябрь' 2023-го

```
CREATE TABLE customers (
    id INT PRIMARY KEY,
    name VARCHAR(100)
);

INSERT INTO customers (id, name) VALUES
(1, 'Andrew'),
(2, 'Jimmy'),
(3, 'Nixton'),
(4, 'Donella');

CREATE TABLE orders (
    id INT PRIMARY KEY,
    customer_id INT REFERENCES customers(id),
    date DATE,
    status VARCHAR(20)
);

INSERT INTO orders (id, customer_id, date, status) VALUES
(1, 3, '2023-10-01', 'PAID'),
(2, 4, '2023-01-10', 'PENDING'),
(3, 1, '2023-09-03', 'REJECTED'),
(4, 2, '2023-07-03', 'REJECTED'),
(5, 3, '2023-07-05', 'PAID'),
(6, 1, '2023-02-12', 'PAID'),
(7, 1, '2023-02-28', 'PAID'),
(8, 4, '2023-11-11', 'PENDING'),
(9, 3, '2023-11-21', 'PENDING'),
(10, 2, '2023-11-26', 'PAID'),
```

```
(11, 1, '2023-05-04', 'PAID'),
(12, 2, '2023-10-30', 'REJECTED'),
(13, 4, '2023-10-12', 'REJECTED'),
(14, 3, '2023-09-23', 'PAID'),
(15, 2, '2023-01-13', 'REJECTED'),
(16, 3, '2023-11-12', 'PENDING'),
(17, 1, '2023-10-02', 'PAID'),
(18, 4, '2023-11-21', 'PENDING'),
(19, 3, '2023-11-22', 'PENDING'),
(20, 2, '2023-02-03', 'PAID'),
(21, 4, '2023-04-04', 'REJECTED'),
(22, 3, '2023-09-30', 'PAID'),
(23, 1, '2023-05-31', 'PAID'),
(24, 4, '2023-11-26', 'PENDING'),
(25, 2, '2023-11-03', 'PENDING'),
(26, 3, '2023-01-30', 'PAID'),
(27, 1, '2023-05-13', 'PAID'),
(28, 1, '2023-07-16', 'PAID'),
(29, 2, '2023-06-02', 'REJECTED'),
(30, 2, '2023-07-30', 'REJECTED'),
(31, 4, '2023-07-24', 'PAID'),
(32, 2, '2023-07-31', 'REJECTED'),
(33, 4, '2023-10-02', 'REJECTED'),
(34, 3, '2023-10-10', 'PAID');


SELECT * FROM customers
SELECT * FROM orders
```

1) Найдите клиентов, у которых все заказы имеют статус 'PAID' за 'Январь', 'Май', 'Июль', 'Октябрь' или 'PENDING' 'Ноябрь' 2023-го

```
SELECT c.name

FROM customers c

WHERE (

   NOT EXISTS (

      SELECT 1

      FROM orders o

      WHERE o.customer_id = c.id

        AND DATE_PART('year', o.date) = 2023

        AND DATE_PART('month', o.date) IN (1, 5, 7, 10)

        AND o.status != 'PAID'

   )

)

OR (

   NOT EXISTS (

      SELECT 1

      FROM orders o

      WHERE o.customer_id = c.id

        AND DATE_PART('year', o.date) = 2023

        AND DATE_PART('month', o.date) = 11

        AND o.status != 'PENDING'

   )

);
```

Showing rows: 1 to 3 Page N

| | name character varying (100) |
|---|---|
| 1 | Andrew |
| 2 | Nixton |
| 3 | Donella |

**Задача 16**

Даны таблицы products, sales

| Products | |
|---|---|
| "id" | "name" |
| 1 | "product-1" |
| 2 | "product-2" |
| 3 | "product-3" |

| Sales | | | |
|---|---|---|---|
| "id" | "product_id" | "amount" | "date" |
| 1 | 3 | 3000 | "2023-01-05" |
| 2 | 1 | 1000 | "2023-01-10" |
| 3 | 3 | 3000 | "2023-01-15" |
| 4 | 1 | 1000 | "2023-01-20" |
| 5 | 2 | 2000 | "2023-02-20" |
| 6 | 1 | 1000 | "2023-01-27" |
| 7 | 3 | 3000 | "2023-02-15" |
| 8 | 2 | 2000 | "2023-02-27" |
| 9 | 2 | 2000 | "2023-03-03" |
| 10 | 1 | 1000 | "2023-03-28" |
| 11 | 3 | 3000 | "2023-03-05" |
| 12 | 2 | 2000 | "2023-04-20" |
| 13 | 1 | 1000 | "2023-05-10" |
| 14 | 1 | 1000 | "2023-05-31" |
| 15 | 3 | 300 | "2023-01-05" |
| 16 | 3 | 300 | "2023-01-05" |
| 17 | 3 | 300 | "2023-01-05" |
| 18 | 3 | 300 | "2023-01-05" |
| 19 | 3 | 300 | "2023-01-05" |
| 20 | 3 | 300 | "2023-01-05" |
| 21 | 3 | 2000 | "2023-07-05" |
| 22 | 3 | 2000 | "2023-07-31" |
| 23 | 2 | 2000 | "2023-07-27" |
| 24 | 3 | 3000 | "2023-01-27" |
| 25 | 3 | 3000 | "2023-11-30" |
| 26 | 2 | 2000 | "2023-01-10" |
| 27 | 2 | 2000 | "2023-01-31" |
| 28 | 3 | 3000 | "2023-11-30" |
| 29 | 1 | 1000 | "2023-08-31" |
| 30 | 1 | 1000 | "2023-01-22" |
| 31 | 3 | 3000 | "2023-07-12" |
| 32 | 3 | 3000 | "2023-07-06" |
| 33 | 2 | 2000 | "2023-01-05" |

Задания:
1) Найдите продукт, для которого средняя сумма продаж за Январь, Июль и Ноябрь превышает 5000 долларов.

```
CREATE TABLE products (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO products (id, name) VALUES
(1, 'product-1'),
(2, 'product-2'),
(3, 'product-3');

CREATE TABLE sales (
    id SERIAL PRIMARY KEY,
    product_id INT REFERENCES products(id),
    amount INT NOT NULL,
    date DATE NOT NULL
);

INSERT INTO sales (id, product_id, amount, date) VALUES
(1, 3, 3000, '2023-01-05'),
(2, 1, 1000, '2023-01-10'),
(3, 3, 3000, '2023-01-15'),
(4, 1, 1000, '2023-01-20'),
(5, 2, 2000, '2023-02-20'),
(6, 1, 1000, '2023-01-27'),
(7, 3, 3000, '2023-02-15'),
(8, 2, 2000, '2023-02-27'),
(9, 2, 2000, '2023-03-03'),
(10, 1, 1000, '2023-03-28'),
(11, 3, 3000, '2023-03-05'),
(12, 2, 2000, '2023-04-20'),
(13, 1, 1000, '2023-05-10'),
(14, 1, 1000, '2023-05-31'),
```

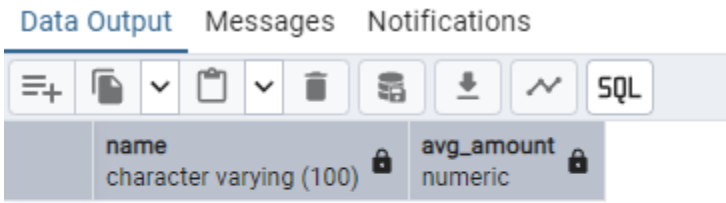
```
(15, 3, 300, '2023-01-05'),
(16, 3, 300, '2023-01-05'),
(17, 3, 300, '2023-01-05'),
(18, 3, 300, '2023-01-05'),
(19, 3, 300, '2023-01-05'),
(20, 3, 300, '2023-01-05'),
(21, 3, 2000, '2023-07-05'),
(22, 3, 2000, '2023-07-31'),
(23, 2, 2000, '2023-07-27'),
(24, 3, 3000, '2023-01-27'),
(25, 3, 3000, '2023-11-30'),
(26, 2, 2000, '2023-01-10'),
(27, 2, 2000, '2023-01-31'),
(28, 3, 3000, '2023-11-30'),
(29, 1, 1000, '2023-08-31'),
(30, 1, 1000, '2023-01-22'),
(31, 3, 3000, '2023-07-12'),
(32, 3, 3000, '2023-07-06'),
(33, 2, 2000, '2023-01-05');

SELECT * FROM products
SELECT * FROM sales
```

1) Найдите продукт, для которого средняя сумма продаж за Январь, Июль и Ноябрь превышает 5000 долларов.

```
SELECT
    p.name,
    ROUND(AVG(s.amount), 2) AS avg_amount
FROM
    sales s
JOIN
    products p ON s.product_id = p.id
WHERE
```

```
    EXTRACT(MONTH FROM s.date) IN (1, 7, 11)
GROUP BY
    p.name
HAVING
    AVG(s.amount) > 5000;
```

-- ни для одного продукта средняя сумма продаж за Январь, Июль и Ноябрь не превышает 5000 долларов



**Задача 17**

Даны таблицы users, logs.

| Users | |
|---|---|
| "id" | "name" |
| 1 | "Max" |
| 2 | "John" |
| 3 | "Lewis" |

| Logs | | | |
|---|---|---|---|
| "id" | "user_id" | "type" | "time" |
| 1 | 2 | "INFO" | "2023-11-11 20:00:01" |
| 2 | 3 | "WARNING" | "2023-10-30 10:22:32" |
| 3 | 1 | "ERROR" | "2023-11-12 19:55:31" |
| 4 | 2 | "ERROR" | "2023-11-30 12:41:12" |
| 5 | 3 | "ERROR" | "2023-12-01 20:35:44" |
| 6 | 1 | "ERROR" | "2023-12-01 03:33:11" |
| 7 | 3 | "INFO" | "2023-09-09 21:12:33" |
| 8 | 2 | "WARNING" | "2023-08-08 23:32:11" |
| 9 | 2 | "INFO" | "2023-12-02 16:12:33" |
| 10 | 3 | "WARNING" | "2023-12-02 20:32:44" |
| 11 | 1 | "INFO" | "2023-03-23 12:32:01" |
| 12 | 1 | "WARNING" | "2023-03-30 14:55:59" |
| 13 | 3 | "INFO" | "2023-01-30 15:45:53" |
| 14 | 3 | "ERROR" | "2023-12-03 18:17:16" |
| 15 | 2 | "ERROR" | "2023-12-02 19:50:51" |
| 16 | 1 | "ERROR" | "2023-06-23 21:22:11" |
| 17 | 2 | "WARNING" | "2023-08-31 10:22:31" |
| 18 | 2 | "INFO" | "2023-09-30 11:22:11" |
| 19 | 1 | "ERROR" | "2023-12-03 22:38:10" |

Задания:
1) Найдите пользователя, который чаще всего выполняет действия типа 'ERROR'.

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

INSERT INTO users (id, name) VALUES
(1, 'Max'),
```

```
(2, 'John'),
(3, 'Lewis');

CREATE TABLE logs (
    id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(id),
    type VARCHAR(20) NOT NULL,
    time TIMESTAMP NOT NULL
);

INSERT INTO logs (id, user_id, type, time) VALUES
(1, 2, 'INFO', '2023-11-11 20:00:01'),
(2, 3, 'WARNING', '2023-10-30 10:22:32'),
(3, 1, 'ERROR', '2023-11-12 19:55:31'),
(4, 2, 'ERROR', '2023-11-30 12:41:12'),
(5, 3, 'ERROR', '2023-12-01 20:35:44'),
(6, 1, 'ERROR', '2023-12-01 03:33:11'),
(7, 3, 'INFO', '2023-09-09 21:12:33'),
(8, 2, 'WARNING', '2023-08-08 23:32:11'),
(9, 2, 'INFO', '2023-12-02 16:12:33'),
(10, 3, 'WARNING', '2023-12-02 20:32:44'),
(11, 1, 'INFO', '2023-03-23 12:32:01'),
(12, 1, 'WARNING', '2023-03-30 14:55:59'),
(13, 3, 'INFO', '2023-01-30 15:45:53'),
(14, 3, 'ERROR', '2023-12-03 18:17:16'),
(15, 2, 'ERROR', '2023-12-02 19:50:51'),
(16, 1, 'ERROR', '2023-06-23 21:22:11'),
(17, 2, 'WARNING', '2023-08-31 10:22:31'),
(18, 2, 'INFO', '2023-09-30 11:22:11'),
(19, 1, 'ERROR', '2023-12-03 22:38:10');
```

1) Найдите пользователя, который чаще всего выполняет действия типа 'ERROR'.

```
SELECT u.name, COUNT(*) AS error_count
FROM logs l
JOIN users u ON l.user_id = u.id
WHERE l.type = 'ERROR'
GROUP BY u.name
ORDER BY error_count DESC
LIMIT 1;
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1

| | name character varying (100) | error_count bigint |
|---|---|---|
| 1 | Max | 4 |

**Задача 18**

Даны таблицы posts, tags, post_tags.

| Posts | |
|---|---|
| "id" | "title" |
| 1 | "Lorem ipsum" |
| 2 | "Lorem ipsum dolor sit amet" |
| 3 | "Lorem ipsum dolor sit" |
| 4 | "Lorem ipsum dolor" |

| Tags | |
|---|---|
| "id" | "name" |
| 1 | "IT" |
| 2 | "PHP" |
| 3 | "JavaScript" |
| 4 | "React" |
| 5 | "Angular" |
| 6 | "Vue.js" |
| 7 | "C/C++" |
| 8 | "Rust" |

| Post_tags | | |
|---|---|---|
| "id" | "post_id" | "tag_id" |
| 1 | 1 | 1 |
| 2 | 1 | 2 |
| 3 | 1 | 3 |
| 4 | 1 | 4 |
| 5 | 1 | 5 |
| 6 | 2 | 7 |
| 7 | 2 | 6 |
| 8 | 2 | 5 |
| 9 | 2 | 4 |
| 10 | 2 | 3 |
| 11 | 3 | 2 |
| 12 | 3 | 7 |
| 13 | 3 | 5 |
| 14 | 3 | 6 |
| 15 | 4 | 5 |
| 16 | 4 | 1 |
| 17 | 4 | 7 |
| 18 | 4 | 6 |

Задания:
1) Найдите топ-3 самых популярных тегов по количеству постов, содержащих каждый из них.

2) Найдите топ-3 самых худших тегов по количеству постов, содержащих каждый из них.

```
CREATE TABLE posts (
    id SERIAL PRIMARY KEY,
    title TEXT NOT NULL
);

INSERT INTO posts (id, title) VALUES
(1, 'Lorem ipsum'),
(2, 'Lorem ipsum dolor sit amet'),
(3, 'Lorem ipsum dolor sit'),
(4, 'Lorem ipsum dolor');

CREATE TABLE tags (
    id SERIAL PRIMARY KEY,
    name TEXT NOT NULL
);

INSERT INTO tags (id, name) VALUES
(1, 'IT'),
(2, 'PHP'),
(3, 'JavaScript'),
(4, 'React'),
(5, 'Angular'),
(6, 'Vue.js'),
(7, 'C/C++'),
(8, 'Rust');

CREATE TABLE post_tags (
    id SERIAL PRIMARY KEY,
    post_id INT REFERENCES posts(id),
    tag_id INT REFERENCES tags(id)
);
```

```
INSERT INTO post_tags (id, post_id, tag_id) VALUES
(1, 1, 1),
(2, 1, 2),
(3, 1, 3),
(4, 1, 4),
(5, 1, 5),
(6, 2, 7),
(7, 2, 6),
(8, 2, 5),
(9, 2, 4),
(10, 2, 3),
(11, 3, 2),
(12, 3, 7),
(13, 3, 5),
(14, 3, 6),
(15, 4, 5),
(16, 4, 1),
(17, 4, 7),
(18, 4, 6);

SELECT * FROM posts
SELECT * FROM tags
SELECT * FROM post_tags
```

1) Найдите топ-3 самых популярных тегов по количеству постов, содержащих каждый из них.

```
SELECT
    t.name AS tag_name,
    COUNT(DISTINCT pt.post_id) AS post_count
FROM
    post_tags pt
JOIN
    tags t ON pt.tag_id = t.id
```

```
GROUP BY
    t.name
ORDER BY
    post_count DESC
LIMIT 3;
```

Showing rows: 1 to 3 Page No

| | tag_name text | post_count bigint |
|---|---|---|
| 1 | Angular | 4 |
| 2 | C/C++ | 3 |
| 3 | Vue.js | 3 |

2) Найдите топ-3 самых худших тегов по количеству постов, содержащих каждый из них.

```
SELECT
    t.name AS tag_name,
    COUNT(DISTINCT pt.post_id) AS post_count
FROM
    post_tags pt
JOIN
    tags t ON pt.tag_id = t.id
GROUP BY
    t.name
ORDER BY
    post_count ASC
LIMIT 3;
--
```

Showing rows: 1 to 3 Page No:

| | tag_name text | post_count bigint |
|---|---|---|
| 1 | JavaScript | 2 |
| 2 | IT | 2 |
| 3 | PHP | 2 |

**Задача 19**

Даны таблицы countries, customers, orders.

| Countries | |
|---|---|
| "id" | "name" |
| 1 | "Germany" |
| 2 | "France" |
| 3 | "Norway" |
| 4 | "Guatemala" |
| 5 | "Sweden" |

| Customers | | |
|---|---|---|
| "id" | "name" | "country_id" |
| 1 | "Alex" | 3 |
| 2 | "Joe" | 1 |
| 3 | "Erick" | 4 |
| 4 | "Max" | 2 |
| 5 | "John" | 5 |

| Orders | | | |
|---|---|---|---|
| "id" | "customer_id" | "amount" | "date" |
| 1 | 3 | 2311 | "2023-12-11" |
| 2 | 3 | 1999 | "2023-12-01" |
| 3 | 3 | 1200 | "2023-11-10" |
| 4 | 3 | 2321 | "2023-11-15" |
| 5 | 3 | 3111 | "2023-12-10" |
| 6 | 2 | 3222 | "2023-12-11" |
| 7 | 2 | 4000 | "2023-10-29" |
| 8 | 2 | 2000 | "2023-10-24" |
| 9 | 2 | 1200 | "2023-10-13" |
| 10 | 1 | 500 | "2023-11-30" |
| 11 | 1 | 600 | "2023-12-10" |
| 12 | 3 | 1200 | "2023-12-05" |
| 13 | 1 | 1400 | "2023-12-06" |
| 14 | 4 | 3200 | "2023-11-10" |
| 15 | 4 | 2300 | "2023-10-20" |
| 16 | 4 | 1400 | "2023-09-30" |
| 17 | 4 | 1590 | "2023-09-22" |
| 18 | 5 | 8000 | "2023-12-09" |
| 19 | 5 | 5700 | "2023-12-08" |
| 20 | 5 | 1100 | "2023-11-27" |
| 21 | 5 | 4200 | "2023-11-16" |
| 22 | 5 | 1900 | "2023-10-22" |
| 23 | 5 | 4700 | "2023-10-17" |

Задания:

1) Найдите три страны, из которых поступает наибольшее количество заказов.

2) Найдите две страны, из которых поступает заказы на наибольшую сумму.

```
CREATE TABLE countries (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO countries (id, name) VALUES
(1, 'Germany'),
(2, 'France'),
(3, 'Norway'),
(4, 'Guatemala'),
(5, 'Sweden');

CREATE TABLE customers (
    id INTEGER PRIMARY KEY,
    name TEXT,
    country_id INTEGER REFERENCES countries(id)
);

INSERT INTO customers (id, name, country_id) VALUES
(1, 'Alex', 3),
(2, 'Joe', 1),
(3, 'Erick', 4),
(4, 'Max', 2),
(5, 'John', 5);

CREATE TABLE orders (
    id INTEGER PRIMARY KEY,
    customer_id INTEGER REFERENCES customers(id),
```

```
    amount INTEGER,
    date DATE
);

INSERT INTO orders (id, customer_id, amount, date) VALUES
(1, 3, 2311, '2023-12-11'),
(2, 3, 1999, '2023-12-01'),
(3, 3, 1200, '2023-11-10'),
(4, 3, 2321, '2023-11-15'),
(5, 3, 3111, '2023-12-10'),
(6, 2, 3222, '2023-12-11'),
(7, 2, 4000, '2023-10-29'),
(8, 2, 2000, '2023-10-24'),
(9, 2, 1200, '2023-10-13'),
(10, 1, 500, '2023-11-30'),
(11, 1, 600, '2023-12-10'),
(12, 3, 1200, '2023-12-05'),
(13, 1, 1400, '2023-12-06'),
(14, 4, 3200, '2023-11-10'),
(15, 4, 2300, '2023-10-20'),
(16, 4, 1400, '2023-09-30'),
(17, 4, 1590, '2023-09-22'),
(18, 5, 8000, '2023-12-09'),
(19, 5, 5700, '2023-12-08'),
(20, 5, 1100, '2023-11-27'),
(21, 5, 4200, '2023-11-16'),
(22, 5, 1900, '2023-10-22'),
(23, 5, 4700, '2023-10-17');

SELECT * FROM countries
SELECT * FROM customers
SELECT * FROM orders
```

1) Найдите три страны, из которых поступает наибольшее количество заказов.

```
SELECT co.name AS country, COUNT(o.id) AS order_count
FROM orders o
JOIN customers c ON o.customer_id = c.id
JOIN countries co ON c.country_id = co.id
GROUP BY co.name
ORDER BY order_count DESC
LIMIT 3;
```

Showing rows: 1 to 3 Page No:

| | country text | order_count bigint |
|---|---|---|
| 1 | Sweden | 6 |
| 2 | Guatemala | 6 |
| 3 | France | 4 |

2) Найдите две страны, из которых поступает заказы на наибольшую сумму.

```
SELECT co.name AS country, SUM(o.amount) AS total_amount
FROM orders o
JOIN customers c ON o.customer_id = c.id
JOIN countries co ON c.country_id = co.id
GROUP BY co.name
ORDER BY total_amount DESC
LIMIT 2;
```

Showing rows: 1 to 2 Page No:

| | country text | total_amount bigint |
|---|---|---|
| 1 | Sweden | 25600 |
| 2 | Guatemala | 12142 |

**Задача 20**

Даны таблицы users, posts, comments.

| Users | |
|---|---|
| "id" | "name" |
| 1 | "Admin" |
| 2 | "Fabiano" |
| 3 | "Johnson" |
| 4 | "Erich" |
| 5 | "Isabella" |

| Posts | | | |
|---|---|---|---|
| "id" | "title" | "user_id" | "published_at" |
| 1 | "Lorem ipsum" | 1 | "2023-11-27" |
| 2 | "Lorem ipsum dolor sit amet" | 1 | "2023-11-30" |
| 3 | "Lorem ipsum dolor sit" | 1 | "2023-10-11" |
| 4 | "Lorem ipsum dolor" | 1 | "2023-11-16" |

| Comments | | | | |
|---|---|---|---|---|
| "id" | "body" | "user_id" | "post_id" | "published_at" |
| 1 | "Great!" | 4 | 1 | "2023-12-01" |
| 2 | "Excellent!" | 4 | 1 | "2023-12-01" |
| 3 | "Interesting..." | 4 | 4 | "2023-12-11" |
| 4 | "Cool." | 4 | 4 | "2023-12-10" |
| 5 | "Hm..." | 5 | 3 | "2023-12-02" |
| 6 | "Exciting." | 5 | 3 | "2023-12-05" |
| 7 | "Fairly..." | 5 | 3 | "2023-12-10" |
| 8 | "Excellent." | 5 | 2 | "2023-12-01" |
| 9 | "Awesome." | 5 | 2 | "2023-12-01" |
| 10 | "So cool!!!" | 3 | 2 | "2023-12-02" |
| 11 | "Exciting." | 3 | 2 | "2023-12-03" |
| 12 | "Normal" | 3 | 3 | "2023-12-05" |
| 13 | "Good!" | 3 | 3 | "2023-12-10" |
| 14 | "Nonsense!!!" | 3 | 1 | "2023-12-11" |
| 15 | "Awful!" | 3 | 1 | "2023-12-09" |
| 16 | "Great!" | 2 | 4 | "2023-12-01" |
| 17 | "Like." | 2 | 4 | "2023-12-05" |
| 18 | "Share!" | 2 | 2 | "2023-12-06" |
| 19 | "Good." | 2 | 1 | "2023-12-11" |
| 20 | "Great!" | 2 | 1 | "2023-12-07" |
| 21 | "Awesome..." | 2 | 1 | "2023-12-09" |
| 22 | "Fairly..." | 2 | 4 | "2023-12-07" |
| 23 | "Awesome!!!" | 2 | 4 | "2023-12-02" |

Задания:

1) Найдите пользователя, который оставил наибольшее количество комментариев под различными постами.

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO users (id, name) VALUES
(1, 'Admin'),
(2, 'Fabiano'),
(3, 'Johnson'),
(4, 'Erich'),
(5, 'Isabella');

CREATE TABLE posts (
    id INTEGER PRIMARY KEY,
    title TEXT,
    user_id INTEGER REFERENCES users(id),
    published_at DATE
);

INSERT INTO posts (id, title, user_id, published_at) VALUES
(1, 'Lorem ipsum', 1, '2023-11-27'),
(2, 'Lorem ipsum dolor sit amet', 1, '2023-11-30'),
(3, 'Lorem ipsum dolor sit', 1, '2023-10-11'),
(4, 'Lorem ipsum dolor', 1, '2023-11-16');

CREATE TABLE comments (
    id INTEGER PRIMARY KEY,
    body TEXT,
    user_id INTEGER REFERENCES users(id),
    post_id INTEGER,
    published_at DATE
);
```

```
INSERT INTO comments (id, body, user_id, post_id, published_at) VALUES
(1, 'Great!', 4, 1, '2023-12-01'),
(2, 'Excellent!', 4, 1, '2023-12-01'),
(3, 'Interesting...', 4, 4, '2023-12-11'),
(4, 'Cool.', 4, 4, '2023-12-10'),
(5, 'Hm...', 5, 3, '2023-12-02'),
(6, 'Exciting.', 5, 3, '2023-12-05'),
(7, 'Fairly...', 5, 3, '2023-12-10'),
(8, 'Excellent.', 5, 2, '2023-12-01'),
(9, 'Awesome.', 5, 2, '2023-12-01'),
(10, 'So cool!!!', 3, 2, '2023-12-02'),
(11, 'Exciting.', 3, 2, '2023-12-03'),
(12, 'Normal', 3, 3, '2023-12-05'),
(13, 'Good!', 3, 3, '2023-12-10'),
(14, 'Nonsense!!!', 3, 1, '2023-12-11'),
(15, 'Awful!', 3, 1, '2023-12-09'),
(16, 'Great!', 2, 4, '2023-12-01'),
(17, 'Like.', 2, 4, '2023-12-05'),
(18, 'Share!', 2, 2, '2023-12-06'),
(19, 'Good.', 2, 1, '2023-12-11'),
(20, 'Great!', 2, 1, '2023-12-07'),
(21, 'Awesome...', 2, 1, '2023-12-09'),
(22, 'Fairly...', 2, 4, '2023-12-07'),
(23, 'Awesome!!!', 2, 4, '2023-12-02');

SELECT * FROM users
SELECT * FROM posts
SELECT * FROM comments
```

1) Найдите пользователя, который оставил наибольшее количество комментариев под различными постами.

```
SELECT u.name, COUNT(DISTINCT c.post_id) AS distinct_posts_commented

FROM comments c

JOIN users u ON c.user_id = u.id

GROUP BY u.name

ORDER BY distinct_posts_commented DESC

LIMIT 1;
```

Showing rows: 1 to 1 Page No: 1

| | name text | distinct_posts_commented bigint |
|---|---|---|
| 1 | Fabiano | 3 |

**Задача 21**

Даны таблицы users, accounts, transactions.

| Users | |
|---|---|
| "id" | "name" |
| 1 | "Fabiano" |
| 2 | "Johnson" |
| 3 | "Erich" |
| 4 | "Isabella" |

| Accounts | | |
|---|---|---|
| "id" | "user_id" | "balance" |
| 1 | 3 | 2500 |
| 2 | 4 | 1100 |
| 3 | 2 | 3700 |
| 4 | 1 | 1500 |

| Transactions | | | | |
|---|---|---|---|---|
| "id" | "amount" | "type" | "account_id" | "date" |
| 1 | 400 | "PURCHASE" | 4 | "2023-12-01" |
| 2 | 400 | "PURCHASE" | 4 | "2023-12-09" |
| 3 | 400 | "PURCHASE" | 4 | "2023-12-11" |
| 4 | 400 | "PURCHASE" | 4 | "2023-12-10" |
| 5 | 200 | "PURCHASE" | 1 | "2023-12-01" |
| 6 | 300 | "PURCHASE" | 1 | "2023-11-29" |
| 7 | 300 | "PURCHASE" | 1 | "2023-11-30" |
| 8 | 200 | "PURCHASE" | 1 | "2023-11-10" |
| 9 | 200 | "PURCHASE" | 2 | "2023-10-30" |
| 10 | 200 | "PURCHASE" | 2 | "2023-10-15" |
| 11 | 200 | "PURCHASE" | 2 | "2023-11-16" |
| 12 | 200 | "PURCHASE" | 2 | "2023-12-10" |
| 13 | 200 | "PURCHASE" | 2 | "2023-12-01" |
| 14 | 500 | "PURCHASE" | 3 | "2023-12-11" |
| 15 | 500 | "PURCHASE" | 3 | "2023-11-27" |
| 16 | 500 | "PURCHASE" | 3 | "2023-11-17" |
| 17 | 500 | "PURCHASE" | 3 | "2023-12-05" |
| 18 | 500 | "PURCHASE" | 3 | "2023-12-07" |
| 19 | 500 | "PURCHASE" | 3 | "2023-12-09" |
| 20 | 500 | "PURCHASE" | 3 | "2023-12-11" |

Задания:

1) Найдите всех пользователей, которые совершили транзакции на сумму более 1000 долларов за последние 30 дней, и у которых баланс после транзакции остался положительным.

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO users (id, name) VALUES
(1, 'Fabiano'),
(2, 'Johnson'),
(3, 'Erich'),
(4, 'Isabella');

CREATE TABLE accounts (
    id INTEGER PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    balance INTEGER
);

INSERT INTO accounts (id, user_id, balance) VALUES
(1, 3, 2500),
(2, 4, 1100),
(3, 2, 3700),
(4, 1, 1500);

CREATE TABLE transactions (
    id INTEGER PRIMARY KEY,
    amount INTEGER,
    type TEXT,
    account_id INTEGER REFERENCES accounts(id),
    date DATE
);

INSERT INTO transactions (id, amount, type, account_id, date) VALUES
```

```
(1, 400, 'PURCHASE', 4, '2023-12-01'),
(2, 400, 'PURCHASE', 4, '2023-12-09'),
(3, 400, 'PURCHASE', 4, '2023-12-11'),
(4, 400, 'PURCHASE', 4, '2023-12-10'),
(5, 200, 'PURCHASE', 1, '2023-12-01'),
(6, 300, 'PURCHASE', 1, '2023-11-29'),
(7, 300, 'PURCHASE', 1, '2023-11-30'),
(8, 200, 'PURCHASE', 1, '2023-11-10'),
(9, 200, 'PURCHASE', 2, '2023-10-30'),
(10, 200, 'PURCHASE', 2, '2023-10-15'),
(11, 200, 'PURCHASE', 2, '2023-11-16'),
(12, 200, 'PURCHASE', 2, '2023-12-10'),
(13, 200, 'PURCHASE', 2, '2023-12-01'),
(14, 500, 'PURCHASE', 3, '2023-12-11'),
(15, 500, 'PURCHASE', 3, '2023-11-27'),
(16, 500, 'PURCHASE', 3, '2023-11-17'),
(17, 500, 'PURCHASE', 3, '2023-12-05'),
(18, 500, 'PURCHASE', 3, '2023-12-07'),
(19, 500, 'PURCHASE', 3, '2023-12-09'),
(20, 500, 'PURCHASE', 3, '2023-12-11');

SELECT * FROM users
SELECT * FROM accounts
SELECT * FROM transactions
```

1) Найдите всех пользователей, которые совершили транзакции на сумму более 1000 долларов за последние 30 дней, и у которых баланс после транзакции остался положительным.

```
SELECT
    u.name,
    a.balance,
    t.total_spent,
    (a.balance - t.total_spent) AS remaining_balance
```

```
FROM users u
JOIN accounts a ON u.id = a.user_id
JOIN (
    SELECT account_id, SUM(amount) AS total_spent
    FROM transactions
    WHERE type = 'PURCHASE'
      AND date >= DATE '2023-11-11'
    GROUP BY account_id
) t ON t.account_id = a.id
WHERE t.total_spent > 1000
  AND (a.balance - t.total_spent) > 0;
```

Showing rows: 1 to 1 Page No: 1 of 1

| | name text | balance integer | total_spent bigint | remaining_balance bigint |
|---|---|---|---|---|
| 1 | Johnson | 3700 | 3500 | 200 |

**Задача 22**

Даны таблицы customers, products, purchases.

| Customers | |
|---|---|
| "id" | "name" |
| 1 | "Fabiano" |
| 2 | "Johnson" |
| 3 | "Erich" |
| 4 | "Isabella" |
| 5 | "Max" |

| Products | |
|---|---|
| "id" | "name" |
| 1 | "Cheese" |
| 2 | "Sausage" |
| 3 | "Watermelon" |

| Purchases | | | | |
|---|---|---|---|---|
| "id" | "customer_id" | "product_id" | "date" | "quantity" |
| 1 | 4 | 2 | "2023-12-11" | 100 |
| 2 | 4 | 3 | "2023-12-10" | 82 |
| 3 | 3 | 1 | "2023-11-05" | 91 |
| 4 | 2 | 3 | "2023-11-11" | 304 |
| 5 | 2 | 1 | "2023-11-29" | 307 |
| 6 | 3 | 2 | "2023-11-28" | 205 |
| 7 | 4 | 2 | "2023-10-02" | 110 |
| 8 | 5 | 1 | "2023-10-03" | 109 |
| 9 | 5 | 3 | "2023-10-05" | 235 |
| 10 | 1 | 2 | "2023-10-08" | 125 |

Задания:
1) Найдите три продукта, которые скорее всего будут куплены в следующем месяце, основываясь на покупках в последние три месяца.

```
CREATE TABLE customers (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO customers (id, name) VALUES
(1, 'Fabiano'),
```

```
(2, 'Johnson'),
(3, 'Erich'),
(4, 'Isabella'),
(5, 'Max');

CREATE TABLE products (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO products (id, name) VALUES
(1, 'Cheese'),
(2, 'Sausage'),
(3, 'Watermelon');

CREATE TABLE purchases (
    id INTEGER PRIMARY KEY,
    customer_id INTEGER REFERENCES customers(id),
    product_id INTEGER REFERENCES products(id),
    date DATE,
    quantity INTEGER
);

INSERT INTO purchases (id, customer_id, product_id, date, quantity) VALUES
(1, 4, 2, '2023-12-11', 100),
(2, 4, 3, '2023-12-10', 82),
(3, 3, 1, '2023-11-05', 91),
(4, 2, 3, '2023-11-11', 304),
(5, 2, 1, '2023-11-29', 307),
(6, 3, 2, '2023-11-28', 205),
(7, 4, 2, '2023-10-02', 110),
(8, 5, 1, '2023-10-03', 109),
```

(9, 5, 3, '2023-10-05', 235),

(10, 1, 2, '2023-10-08', 125);

SELECT * FROM customers

SELECT * FROM products

SELECT * FROM purchases

1) Найдите три продукта, которые скорее всего будут куплены в следующем месяце, основываясь на покупках в последние три месяца.

```
SELECT p.name AS product_name, SUM(pr.quantity) AS total_quantity

FROM purchases pr

JOIN products p ON pr.product_id = p.id

WHERE pr.date BETWEEN DATE '2023-09-01' AND DATE '2023-11-30'

GROUP BY p.name

ORDER BY total_quantity DESC

LIMIT 3;
```

Showing rows: 1 to 3 Page No: 1

| | product_name text | total_quantity bigint |
|---|---|---|
| 1 | Watermelon | 539 |
| 2 | Cheese | 507 |
| 3 | Sausage | 440 |

**Задача 23**

Даны таблицы users, sessions

| Users | |
|---|---|
| "id" | "name" |
| 1 | "John" |
| 2 | "Andrew" |
| 3 | "Kevin" |
| 4 | "Mark" |
| 5 | "Eric" |

| Sessions | | | |
|---|---|---|---|
| "id" | "user_id" | "start" | "end" |
| 1 | 2 | "2023-01-10 18:39:02" | "2023-02-10 16:57:33" |
| 2 | 1 | "2023-01-10 18:39:06" | "2023-02-10 16:57:07" |
| 3 | 5 | "2023-03-08 10:22:04" | "2023-03-09 12:11:57" |
| 4 | 5 | "2023-01-10 23:59:03" | "2023-01-11 13:15:33" |
| 5 | 2 | "2023-08-31 18:22:05" | "2023-08-31 11:55:25" |
| 6 | 3 | "2023-11-02 22:12:10" | "2023-11-03 08:32:20" |
| 7 | 4 | "2023-10-13 20:33:10" | "2023-10-13 11:55:15" |
| 8 | 3 | "2023-09-30 08:01:01" | "2023-09-30 10:02:05" |
| 9 | 2 | "2023-02-13 23:39:05" | "2023-02-14 05:41:31" |
| 10 | 1 | "2023-08-09 10:22:11" | "2023-08-09 10:44:43" |
| 11 | 1 | "2023-10-12 17:59:49" | "2023-10-12 20:49:48" |
| 12 | 5 | "2023-11-11 05:31:43" | "2023-11-11 05:59:55" |
| 13 | 2 | "2023-11-20 08:00:59" | "2023-11-21 18:32:58" |
| 14 | 1 | "2023-11-26 08:57:32" | "2023-11-26 19:30:55" |
| 15 | 4 | "2023-04-12 11:33:39" | "2023-04-12 23:51:29" |
| 16 | 3 | "2023-04-30 17:30:27" | "2023-04-30 20:40:26" |
| 17 | 2 | "2023-01-10 19:22:52" | "2023-01-12 13:11:24" |
| 18 | 3 | "2023-10-15 11:22:13" | "2023-10-16 14:33:32" |
| 19 | 4 | "2023-05-31 15:44:55" | "2023-06-03 11:32:11" |
| 20 | 5 | "2023-07-10 07:57:12" | "2023-07-10 20:48:33" |

Задания:
1) Найдите пользователя, который провел наибольшее время в системе в течение последнего месяца, учитывая все его сессии

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO users (id, name) VALUES
```

```
(1, 'John'),
(2, 'Andrew'),
(3, 'Kevin'),
(4, 'Mark'),
(5, 'Eric');

CREATE TABLE sessions (
    id INTEGER PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    start TIMESTAMP,
    "end" TIMESTAMP
);

INSERT INTO sessions (id, user_id, start, "end") VALUES
(1, 2, '2023-01-10 18:39:02', '2023-02-10 16:57:33'),
(2, 1, '2023-01-10 18:39:06', '2023-02-10 16:57:07'),
(3, 5, '2023-03-08 10:22:04', '2023-03-09 12:11:57'),
(4, 5, '2023-01-10 23:59:03', '2023-01-11 13:15:33'),
(5, 2, '2023-08-31 18:22:05', '2023-08-31 11:55:25'),
(6, 3, '2023-11-02 22:12:10', '2023-11-03 08:32:20'),
(7, 4, '2023-10-13 20:33:10', '2023-10-13 11:55:15'),
(8, 3, '2023-09-30 08:01:01', '2023-09-30 10:02:05'),
(9, 2, '2023-02-13 23:39:05', '2023-02-14 05:41:31'),
(10, 1, '2023-08-09 10:22:11', '2023-08-09 10:44:43'),
(11, 1, '2023-10-12 17:59:49', '2023-10-12 20:49:48'),
(12, 5, '2023-11-11 05:31:43', '2023-11-11 05:59:55'),
(13, 2, '2023-11-20 08:00:59', '2023-11-21 18:32:58'),
(14, 1, '2023-11-26 08:57:32', '2023-11-26 19:30:55'),
(15, 4, '2023-04-12 11:33:39', '2023-04-12 23:51:29'),
(16, 3, '2023-04-30 17:30:27', '2023-04-30 20:40:26'),
(17, 2, '2023-01-10 19:22:52', '2023-01-12 13:11:24'),
(18, 3, '2023-10-15 11:22:13', '2023-10-16 14:33:32'),
```

(19, 4, '2023-05-31 15:44:55', '2023-06-03 11:32:11'),

(20, 5, '2023-07-10 07:57:12', '2023-07-10 20:48:33');

SELECT * FROM users

SELECT * FROM sessions

1) Найдите пользователя, который провел наибольшее время в системе в течение последнего месяца, учитывая все его сессии

```
SELECT u.name, SUM(EXTRACT(EPOCH FROM (s."end" - s.start))) AS total_time_seconds
FROM sessions s
JOIN users u ON s.user_id = u.id
WHERE s.start >= DATE '2023-11-01' AND s.start < DATE '2023-12-01'
GROUP BY u.name
ORDER BY total_time_seconds DESC
LIMIT 1;
```

Showing rows: 1 to 1 Page No:

| | name text | total_time_seconds numeric |
|---|---|---|
| 1 | Andrew | 124319.000000 |

**Задача 24**

Даны таблицы pages, users, app_usage.

| Pages | |
|---|---|
| "id" | "name" |
| 1 | "Home" |
| 2 | "Profile" |
| 3 | "Feed" |

| Users | |
|---|---|
| "id" | "name" |
| 1 | "Fabiano" |
| 2 | "Johnson" |
| 3 | "Erich" |

| App_usage | | | | |
|---|---|---|---|---|
| "id" | "user_id" | "page_id" | "start" | "end" |
| 1 | 2 | 3 | "2023-12-01 10:50:01" | "2023-12-01 16:25:31" |
| 2 | 1 | 1 | "2023-11-30 12:33:44" | "2023-11-30 12:39:31" |
| 3 | 3 | 2 | "2023-11-29 18:17:16" | "2023-11-29 19:31:25" |
| 4 | 3 | 2 | "2023-12-01 09:31:33" | "2023-12-01 11:25:44" |
| 5 | 1 | 1 | "2023-12-02 11:55:53" | "2023-12-02 12:10:22" |
| 6 | 2 | 3 | "2023-12-03 14:01:03" | "2023-12-03 16:44:00" |
| 7 | 1 | 3 | "2023-12-10 17:58:32" | "2023-12-10 18:56:22" |
| 8 | 2 | 2 | "2023-12-11 13:32:33" | "2023-12-11 14:00:55" |
| 9 | 3 | 1 | "2023-12-12 10:31:12" | "2023-12-12 11:00:56" |
| 10 | 3 | 2 | "2023-10-31 12:15:15" | "2023-10-31 13:14:17" |
| 11 | 2 | 1 | "2023-12-12 14:25:32" | "2023-12-12 14:55:11" |
| 12 | 1 | 3 | "2023-12-01 10:50:01" | "2023-12-01 16:25:31" |

Задания:
1) Найдите наиболее популярную страницу приложения, на которой пользователи проводят наибольшее количество времени

```
CREATE TABLE pages (
    id INTEGER PRIMARY KEY,
    name TEXT
);
```

```
INSERT INTO pages (id, name) VALUES
(1, 'Home'),
(2, 'Profile'),
(3, 'Feed');

CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO users (id, name) VALUES
(1, 'Fabiano'),
(2, 'Johnson'),
(3, 'Erich');

CREATE TABLE app_usage (
    id INTEGER PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    page_id INTEGER REFERENCES pages(id),
    start TIMESTAMP,
    "end" TIMESTAMP
);

INSERT INTO app_usage (id, user_id, page_id, start, "end") VALUES
(1, 2, 3, '2023-12-01 10:50:01', '2023-12-01 16:25:31'),
(2, 1, 1, '2023-11-30 12:33:44', '2023-11-30 12:39:31'),
(3, 3, 2, '2023-11-29 18:17:16', '2023-11-29 19:31:25'),
(4, 3, 2, '2023-12-01 09:31:33', '2023-12-01 11:25:44'),
(5, 1, 1, '2023-12-02 11:55:53', '2023-12-02 12:10:22'),
(6, 2, 3, '2023-12-03 14:01:03', '2023-12-03 16:44:00'),
(7, 1, 3, '2023-12-10 17:58:32', '2023-12-10 18:56:22'),
(8, 2, 2, '2023-12-11 13:32:33', '2023-12-11 14:00:55'),
```

(9, 3, 1, '2023-12-12 10:31:12', '2023-12-12 11:00:56'),

(10, 3, 2, '2023-10-31 12:15:15', '2023-10-31 13:14:17'),

(11, 2, 1, '2023-12-12 14:25:32', '2023-12-12 14:55:11'),

(12, 1, 3, '2023-12-01 10:50:01', '2023-12-01 16:25:31');

SELECT * FROM pages

SELECT * FROM users

SELECT * FROM app_usage

1) Найдите наиболее популярную страницу приложения, на которой пользователи проводят наибольшее количество времени

```
SELECT
    p.name AS page_name,
    FLOOR(SUM(EXTRACT(EPOCH FROM (au."end" - au.start))) / 3600) AS total_hours,
    FLOOR((SUM(EXTRACT(EPOCH FROM (au."end" - au.start))) % 3600) / 60) AS total_minutes
FROM app_usage au
JOIN pages p ON au.page_id = p.id
GROUP BY p.name
ORDER BY total_hours DESC, total_minutes DESC
LIMIT 1;
```

Showing rows: 1 to 1 Page No: 1 of 1

| | page_name text | total_hours numeric | total_minutes numeric |
|---|---|---|---|
| 1 | Feed | 14 | 51 |

**Задача 25**

Даны таблицы employees, departments, salaries

| Employees | | | | |
|---|---|---|---|---|
| "id" | "first_name" | "last_name" | "dob" | "department_id" |
| 1 | "John" | "Smith" | "1987-12-12" | 4 |
| 2 | "Alex" | "Cooper" | "1999-02-20" | 3 |
| 3 | "Agata" | "Leyla" | "1970-01-30" | 1 |
| 4 | "Ivan" | "Ivanov" | "1994-05-31" | 1 |
| 5 | "Peter" | "Petrov" | "1991-08-24" | 3 |
| 6 | "Karl" | "Benz" | "1988-01-25" | 4 |
| 7 | "Joe" | "Sparrow" | "1990-11-29" | 2 |
| 8 | "John" | "Doe" | "1992-12-24" | 2 |

| Departments | | |
|---|---|---|
| "id" | "name" | "registration_date" |
| 1 | "Ruby" | "2005-12-30" |
| 2 | "PHP" | "2000-12-12" |
| 3 | "Vue.js" | "2015-07-31" |
| 4 | "React" | "2014-05-31" |

| Salaries | | | |
|---|---|---|---|
| "id" | "employee_id" | "amount" | "date" |
| 1 | 7 | 3500 | "2023-12-12" |
| 2 | 8 | 3000 | "2023-11-10" |
| 3 | 1 | 3200 | "2023-11-11" |
| 4 | 2 | 4700 | "2023-10-12" |
| 5 | 4 | 5500 | "2023-12-07" |
| 6 | 3 | 5600 | "2023-12-10" |
| 7 | 2 | 2100 | "2023-10-11" |
| 8 | 1 | 4000 | "2023-12-11" |

Задания:
1) Найдите средний размер зарплаты в каждом отделении за 'Декабрь' 2023 и выведите информацию об отделениях, где средний размер зарплаты превышает определенный порог.

```
CREATE TABLE employees (
    id INTEGER PRIMARY KEY,
    first_name TEXT,
    last_name TEXT,
    dob DATE,
    department_id INTEGER
```

```
);

INSERT INTO employees (id, first_name, last_name, dob, department_id) VALUES
(1, 'John', 'Smith', '1987-12-12', 4),
(2, 'Alex', 'Cooper', '1999-02-20', 3),
(3, 'Agata', 'Leyla', '1970-01-30', 1),
(4, 'Ivan', 'Ivanov', '1994-05-31', 1),
(5, 'Peter', 'Petrov', '1991-08-24', 3),
(6, 'Karl', 'Benz', '1988-01-25', 4),
(7, 'Joe', 'Sparrow', '1990-11-29', 2),
(8, 'John', 'Doe', '1992-12-24', 2);

CREATE TABLE departments (
    id INTEGER PRIMARY KEY,
    name TEXT,
    registration_date DATE
);

INSERT INTO departments (id, name, registration_date) VALUES
(1, 'Ruby', '2005-12-30'),
(2, 'PHP', '2000-12-12'),
(3, 'Vue.js', '2015-07-31'),
(4, 'React', '2014-05-31');

CREATE TABLE salaries (
    id INTEGER PRIMARY KEY,
    employee_id INTEGER REFERENCES employees(id),
    amount INTEGER,
    date DATE
);

INSERT INTO salaries (id, employee_id, amount, date) VALUES
```

(1, 7, 3500, '2023-12-12'),

(2, 8, 3000, '2023-11-10'),

(3, 1, 3200, '2023-11-11'),

(4, 2, 4700, '2023-10-12'),

(5, 4, 5500, '2023-12-07'),

(6, 3, 5600, '2023-12-10'),

(7, 2, 2100, '2023-10-11'),

(8, 1, 4000, '2023-12-11');

SELECT * FROM employees

SELECT * FROM departments

SELECT * FROM salaries

1) Найдите средний размер зарплаты в каждом отделении за 'Декабрь' 2023 и выведите информацию об отделениях, где средний размер зарплаты превышает определенный порог.

SELECT d.name AS department_name, AVG(s.amount) AS avg_salary

FROM salaries s

JOIN employees e ON s.employee_id = e.id

JOIN departments d ON e.department_id = d.id

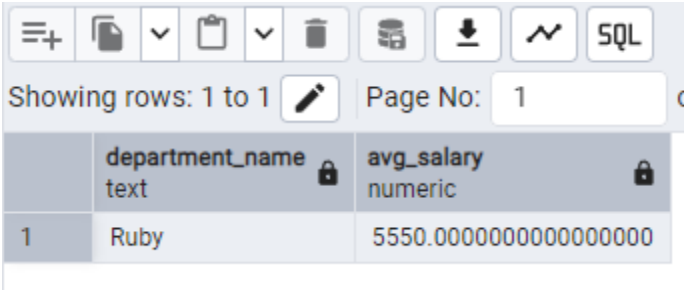
WHERE s.date BETWEEN '2023-12-01' AND '2023-12-31'

GROUP BY d.name

HAVING AVG(s.amount) > 4000

ORDER BY avg_salary DESC;

-- Допустим, пороговое значение зарплаты — это 4000 –




| | department_name text | avg_salary numeric |
|---|---|---|
| 1 | Ruby | 5550.0000000000000000 |

**Задача 26**

Даны таблицы users, friendships.

| Users | | |
|---|---|---|
| "id" | "name" | "registration_date" |
| 1 | "Jorge" | "2014-08-22" |
| 2 | "Adam" | "2014-09-15" |
| 3 | "Max" | "2015-11-10" |
| 4 | "John" | "2016-07-01" |
| 5 | "Ella" | "2017-05-19" |
| 6 | "Bryson" | "2018-04-02" |
| 7 | "Lorelai" | "2018-05-04" |

| Friendships | | | |
|---|---|---|---|
| "id" | "user_id_1" | "user_id_2" | "strength" |
| 1 | 7 | 5 | 3 |
| 2 | 6 | 3 | 4 |
| 3 | 6 | 2 | 5 |
| 4 | 6 | 1 | 10 |
| 5 | 6 | 4 | 9 |
| 6 | 4 | 2 | 9 |
| 7 | 3 | 1 | 8 |
| 8 | 7 | 2 | 5 |
| 9 | 2 | 5 | 7 |
| 10 | 1 | 3 | 7 |
| 11 | 4 | 5 | 4 |
| 12 | 7 | 2 | 4 |
| 13 | 3 | 1 | 3 |
| 14 | 6 | 2 | 2 |

Задание:
1) Найдите наиболее влиятельного пользователя в графе социальных связей, используя метрики важности графа.

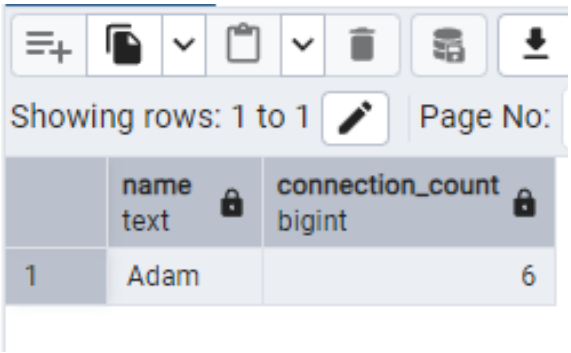
```
CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name TEXT,
    registration_date DATE
);

INSERT INTO users (id, name, registration_date) VALUES
(1, 'Jorge', '2014-08-22'),
```

```
(2, 'Adam', '2014-09-15'),
(3, 'Max', '2015-11-10'),
(4, 'John', '2016-07-01'),
(5, 'Ella', '2017-05-19'),
(6, 'Bryson', '2018-04-02'),
(7, 'Lorelai', '2018-05-04');

CREATE TABLE friendships (
    id INTEGER PRIMARY KEY,
    user_id_1 INTEGER REFERENCES users(id),
    user_id_2 INTEGER REFERENCES users(id),
    strength INTEGER
);

INSERT INTO friendships (id, user_id_1, user_id_2, strength) VALUES
(1, 7, 5, 3),
(2, 6, 3, 4),
(3, 6, 2, 5),
(4, 6, 1, 10),
(5, 6, 4, 9),
(6, 4, 2, 9),
(7, 3, 1, 8),
(8, 7, 2, 5),
(9, 2, 5, 7),
(10, 1, 3, 7),
(11, 4, 5, 4),
(12, 7, 2, 4),
(13, 3, 1, 3),
(14, 6, 2, 2);

SELECT * FROM users
SELECT * FROM friendships
```

1) Найдите наиболее влиятельного пользователя в графе социальных связей, используя метрики важности графа.

```
SELECT u.name, COUNT(*) AS connection_count

FROM users u

JOIN (

    SELECT user_id_1 AS user_id FROM friendships

    UNION ALL

    SELECT user_id_2 FROM friendships

) all_connections ON u.id = all_connections.user_id

GROUP BY u.id, u.name

ORDER BY connection_count DESC

LIMIT 1;
```

--В SQL-графе мы аппроксимируем влияние через степень вершины, считая, сколько раз пользователь появляется в связях. Было посчитано количество упоминаний пользователя как участника связи (в обеих колонках), сгруппировано по ID - и выбрали того, у кого максимум.--




| | name text | connection_count bigint |
|---|---|---|
| 1 | Adam | 6 |

**Задание 27**

Даны таблицы products, sales, external_factors.

| Products | |
|---|---|
| "id" | "name" |
| 1 | "product-1" |
| 2 | "product-2" |
| 3 | "product-3" |

| Sales | | | |
|---|---|---|---|
| "id" | "product_id" | "amount" | "date" |
| 1 | 3 | 3000 | "2023-01-05" |
| 2 | 1 | 1000 | "2023-03-10" |
| 3 | 3 | 3000 | "2023-05-15" |
| 4 | 1 | 1000 | "2023-06-20" |
| 5 | 2 | 2000 | "2023-07-20" |
| 6 | 1 | 1000 | "2023-09-27" |
| 7 | 3 | 3000 | "2023-08-15" |
| 8 | 2 | 2000 | "2023-09-27" |
| 9 | 2 | 2000 | "2023-09-03" |
| 10 | 1 | 1000 | "2023-10-28" |
| 11 | 3 | 3000 | "2023-10-05" |
| 12 | 2 | 2000 | "2023-11-20" |
| 13 | 1 | 1000 | "2023-12-01" |
| 14 | 1 | 1000 | "2023-11-22" |
| 15 | 3 | 300 | "2023-11-21" |
| 16 | 3 | 300 | "2023-11-22" |
| 17 | 3 | 300 | "2023-11-11" |
| 18 | 3 | 300 | "2023-10-07" |
| 19 | 3 | 300 | "2023-10-05" |
| 20 | 3 | 300 | "2023-10-05" |
| 21 | 3 | 2000 | "2023-12-10" |
| 22 | 3 | 2000 | "2023-11-17" |
| 23 | 2 | 2000 | "2023-11-27" |
| 24 | 3 | 3000 | "2023-12-04" |
| 25 | 3 | 3000 | "2023-12-03" |
| 26 | 2 | 2000 | "2023-10-10" |
| 27 | 2 | 2000 | "2023-11-30" |
| 28 | 3 | 3000 | "2023-10-31" |
| 29 | 1 | 1000 | "2023-10-31" |
| 30 | 1 | 1000 | "2023-12-11" |
| 31 | 3 | 3000 | "2023-11-12" |
| 32 | 3 | 3000 | "2023-12-06" |
| 33 | 2 | 2000 | "2023-11-05" |

| external_factors | | |
|---|---|---|
| "id" | "factor_value" | "sale_date" |
| 1 | 7 | "2023-11-05" |
| 2 | 5 | "2023-11-10" |
| 3 | 2 | "2023-12-01" |
| 4 | 3 | "2023-12-05" |
| 5 | 3 | "2023-11-15" |
| 6 | 8 | "2023-10-31" |
| 7 | 7 | "2023-10-22" |
| 8 | 6 | "2023-10-29" |
| 9 | 6 | "2023-11-11" |
| 10 | 4 | "2023-11-13" |
| 11 | 3 | "2023-12-05" |
| 12 | 1 | "2023-12-10" |
| 13 | 2 | "2023-12-12" |
| 14 | 10 | "2023-10-10" |
| 15 | 9 | "2023-10-02" |

Задания:
1) Разработайте прогноз продаж на следующий месяц с учетом внешних факторов.

```
CREATE TABLE products (
    id INTEGER PRIMARY KEY,
    name TEXT
);

INSERT INTO products (id, name) VALUES
(1, 'product-1'),
(2, 'product-2'),
(3, 'product-3');

CREATE TABLE sales (
    id INTEGER PRIMARY KEY,
    product_id INTEGER REFERENCES products(id),
    amount INTEGER,
    date DATE
);
```

```
INSERT INTO sales (id, product_id, amount, date) VALUES
(1, 3, 3000, '2023-01-05'),
(2, 1, 1000, '2023-03-10'),
(3, 3, 3000, '2023-05-15'),
(4, 1, 1000, '2023-06-20'),
(5, 2, 2000, '2023-07-20'),
(6, 1, 1000, '2023-09-27'),
(7, 3, 3000, '2023-08-15'),
(8, 2, 2000, '2023-09-27'),
(9, 2, 2000, '2023-09-03'),
(10, 1, 1000, '2023-10-28'),
(11, 3, 3000, '2023-10-05'),
(12, 2, 2000, '2023-11-20'),
(13, 1, 1000, '2023-12-01'),
(14, 1, 1000, '2023-11-22'),
(15, 3, 300,  '2023-11-21'),
(16, 3, 300,  '2023-11-22'),
(17, 3, 300,  '2023-11-11'),
(18, 3, 300,  '2023-10-07'),
(19, 3, 300,  '2023-10-05'),
(20, 3, 300,  '2023-10-05'),
(21, 3, 2000, '2023-12-10'),
(22, 3, 2000, '2023-11-17'),
(23, 2, 2000, '2023-11-27'),
(24, 3, 3000, '2023-12-04'),
(25, 3, 3000, '2023-12-03'),
(26, 2, 2000, '2023-10-10'),
(27, 2, 2000, '2023-11-30'),
(28, 3, 3000, '2023-10-31'),
(29, 1, 1000, '2023-10-31'),
(30, 1, 1000, '2023-12-11'),
(31, 3, 3000, '2023-11-12'),
```

```
(32, 3, 3000, '2023-12-06'),
(33, 2, 2000, '2023-11-05');

CREATE TABLE external_factors (
    id INTEGER PRIMARY KEY,
    factor_value INTEGER,
    sale_date DATE
);

INSERT INTO external_factors (id, factor_value, sale_date) VALUES
(1, 7, '2023-11-05'),
(2, 5, '2023-11-10'),
(3, 2, '2023-12-01'),
(4, 3, '2023-12-05'),
(5, 3, '2023-11-15'),
(6, 8, '2023-10-31'),
(7, 7, '2023-10-22'),
(8, 6, '2023-10-29'),
(9, 6, '2023-11-11'),
(10, 4, '2023-11-13'),
(11, 3, '2023-12-05'),
(12, 1, '2023-12-10'),
(13, 2, '2023-12-12'),
(14, 10, '2023-10-10'),
(15, 9, '2023-10-02');

SELECT * FROM products
SELECT * FROM sales
SELECT * FROM external_factors
```

1) Разработайте прогноз продаж на следующий месяц с учетом внешних факторов.

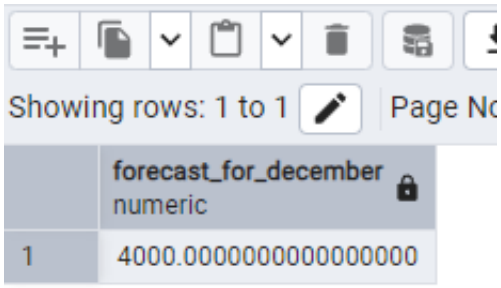
```
WITH joined AS (
    SELECT s.amount, ef.factor_value, s.date
    FROM sales s
    JOIN external_factors ef ON s.date = ef.sale_date
),
avg_by_factor AS (
    SELECT
        ef.factor_value,
        AVG(s.amount) AS avg_sales_per_factor
    FROM sales s
    JOIN external_factors ef ON s.date = ef.sale_date
    GROUP BY ef.factor_value
),
dec_factors AS (
    SELECT factor_value
    FROM external_factors
    WHERE sale_date BETWEEN '2023-12-01' AND '2023-12-31'
),
prediction AS (
    SELECT
        f.factor_value,
        abf.avg_sales_per_factor
    FROM dec_factors f
    JOIN avg_by_factor abf ON f.factor_value = abf.factor_value
)
SELECT
    SUM(avg_sales_per_factor) AS forecast_for_december
FROM prediction;
```

-- Прогноз выполнен с использованием приближённой модели: оценена средняя продажа для каждого значения внешнего фактора (по историческим данным), и затем спрогнозирована общая сумма продаж на декабрь, исходя из того, какие факторы зафиксированы на этот период.

-- Это приближённая регрессионная логика, реализованная средствами SQL.

-- Это не модель машинного обучения, но позволяет реализовать базовое предсказание тренда средствами SQL.

-- Поэтому в случае реального проекта подход можно расширить до модели регрессии или временных рядов.




| | forecast_for_december numeric |
|---|---|
| 1 | 4000.0000000000000000 |

**Задача 28**

Даны таблицы documents.

| Documents | |
|---|---|
| "id" | "content" |
| 1 | "Lorem ipsum dolor sit amet, consectetur adipisicing elit. Accusantium ad at beatae consequuntur cupiditate dolore dolores dolorum, ea error id ipsum iure iusto minima odit perspiciatis quidem sunt vel voluptate!" |
| 2 | "Lorem ipsum dolor sit amet, consectetur adipisicing elit. Beatae, blanditiis corporis deleniti doloribus, eos et eveniet fugiat id inventore itaque molestiae natus nobis nostrum odit pariatur quod quos reiciendis sit vitae 43 voluptas. Eos eveniet praesentium provident. Illo inventore ipsam minima rem rerum. Amet autem dolor doloribus eveniet facere iste voluptatem!" |
| 3 | "Aliquid atque beatae, debitis facilis fuga in iure molestiae nostrum omnis? Aliquam assumenda, consectetur cupiditate dignissimos dolor, harum illo itaque laborum necessitatibus obcaecati similique sunt, velit voluptatum. Aut cumque dolor dolorum error, exercitationem iusto laudantium maiores non perferendis quae, qui quod reiciendis sunt suscipit ullam ut voluptate? Accusantium, aliquam atque, deleniti exercitationem hic incidunt ipsam iste itaque iusto neque nobis officiis quam quas quibusdam quidem soluta voluptatem?" |
| 4 | "Lorem ipsum lestiae saepe sequi sint, temporibus. Accusantium ad commodi, consectetur deserunt dolores doloribus ea eaque eligendi enim in ipsam minima quam quas qui voluptatem? Amet consectetur cumque, dicta dolore expedita fuga fugiat iusto minima nostrum odio, quia, quidem sequi tempora. A dolorum, ea eveniet facilis fugiat ipsa mollitia obcaecati quia quo sunt tempore, totam! Ad alias aliquid aperiam asperiores at corporis distinctio dolorem dolorum error eum fuga ipsum iure laudantium libero nam necessitatibus officiis perspiciatis quos repudiandae saepe sequi soluta tempore, vero." |
| 5 | "Lorem sit amet, consectetur adipisicing elit. Amet, atque corporis cumque deserunt dolores ducimus esse est facere illo laudantium maiores molestias mollitia odio perferendis possimus repellendus sapiente sequi sit suscipit vero? Architecto blanditiis corporis culpa cum debitis deserunt dignissimos eligendi esse est excepturi expedita hic iusto, modi mollitia optio perferendis quas qui quia quibusdam quidem sequi sint suscipit tempora tempore minus mollitia nisi quam quod rem tempore tenetur, ullam ut veniam, voluptatem voluptatibus. Atque consequatur eveniet facilis odit unde. Adipisci culpa deserunt expedita explicabo neque obcaecati odio tempore! Aliquam at, aut beatae blanditiis error, eum excepturi facere incidunt ipsam laboriosam nam nemo nobis nostrum numquam odit provident quia quibusdam ratione recusandae rem repudiandae rerum sequi unde." |
| 6 | "Consectetur adipisicing elit. Aliquam beatae delectus, dolor ipsa itaque minima necessitatibus officiis praesentium quisquam reiciendis. Accusantium autem dolorum id ipsa ipsum, labore nam officia placeat porro quas quos ratione sapiente sint. Aspernatur consectetur deleniti error inventore laudantium obcaecati officiis possimus, totam? Assumenda atque delectus eaque enim est et eum facilis iusto laborum laudantium modi mollitia non nostrum odio praesentium, quas quia recusandae sit, unde voluptas! Animi est fugiat fugit mollitia nihil nulla, sapiente sed soluta. A aliquam aperiam at dolores est excepturi inventore ipsum voluptatem. A adipisci aliquam aspernatur assumenda aut beatae doloremque dolores error exercitationem fuga ipsa iste laborum maiores nobis non pariatur quae, quas quia quo repellendus similique sint soluta veritatis? A ad adipisci, cum distinctio eaque, illo in molestiae neque optio praesentium quia sed tempore voluptates. Autem error nihil officiis perferendis quis tenetur ut!" |
| 7 | "Ab animi architecto assumenda consequatur cumque deserunt distinctio, doloremque ex explicabo illo libero molestias non nulla officiis praesentium provident quae quas quasi quibusdam quod rerum sequi tenetur voluptatum. Ad alias aliquam amet at atque autem blanditiis deserunt dolores doloribus eligendi enim est ex explicabo facilis fugit, illum in incidunt ipsa iusto magni maiores maxime nemo non officia omnis optio pariatur quae quibusdam quisquam quo sed sint sit sunt temporibus ut vitae voluptatem. Aspernatur culpa |

| | |
|---|---|
| | deserunt dolorum earum eos hic impedit ipsa ipsum maxime, minima neque nesciunt omnis, quam reiciendis sed similique ut voluptas! A at doloribus facere laborum molestiae quas similique. Earum excepturi facere harum, inventore ipsum necessitatibus praesentium quidem reiciendis tenetur veritatis. Adipisci asperiores commodi dolor error facilis quae totam voluptates. Dolor ipsum quae quaerat quod reprehenderit saepe sequi, similique vel voluptatem voluptatibus. Adipisci architecto debitis dignissimos ducimus ea eaque eos eum excepturi illum, impedit incidunt molestias mollitia nesciunt nobis placeat quia quis quo quod recusandae reiciendis, rem sequi totam veniam! Commodi expedita impedit optio repudiandae?" |
| 8 | "Eveniet facere fuga ipsa ipsam labore minus, natus necessitatibus numquam placeat possimus quia quod reiciendis unde veniam veritatis voluptate! Accusantium ad beatae consectetur dicta modi molestiae praesentium repellendus saepe ullam. Assumenda autem beatae cum dolore dolorem doloribus, facilis hic id labore laboriosam laborum laudantium maiores minus nemo numquam optio quae quaerat quis quo rem sunt vel voluptas voluptatibus. Est inventore laboriosam ut. Aliquid, culpa, cupiditate delectus doloremque ea qui quidem quis tempore ullam vel veritatis vitae voluptatem? Commodi ipsum libero nihil sapiente? Accusamus aperiam expedita facilis in minus molestiae molestias mollitia optio, quia, quibusdam quis sunt voluptatibus! Aliquid consectetur deleniti ea hic quia rem, reprehenderit sit sunt vel velit! Architecto autem dolor doloremque dolores esse eum ex explicabo facere, facilis ipsum itaque minus modi mollitia nihil odio pariatur praesentium quaerat, quasi qui quod, reiciendis soluta temporibus vel veniam voluptas voluptates voluptatum? A, ab accusamus error eveniet ex facilis, minima minus non obcaecati placeat, quas quasi quidem repellendus. Ducimus eum laudantium maiores maxime, obcaecati possimus quidem rerum sapiente unde ut. Illo modi quis recusandae sequi!" |
| 9 | "Ea harum ipsum, iusto laboriosam maiores minima neque nisi quam quas ut voluptas voluptatem. Architecto assumenda autem corporis debitis delectus ducimus eum excepturi fuga, harum inventore maxime molestiae nisi nobis nostrum officiis perspiciatis quas quod repudiandae rerum sapiente sequi unde vitae. Accusamus aperiam commodi eos, esse explicabo ipsum laboriosam magni, molestiae pariatur possimus quisquam repellat tempore ullam velit voluptate. Accusamus adipisci animi commodi cumque cupiditate ducimus enim facere, fugit ipsa iusto minima mollitia necessitatibus neque nesciunt numquam praesentium quaerat quas, quos rem repellat saepe tempora tempore voluptatem! Iusto maiores nihil quae. Atque corporis cumque, deleniti dolorem doloremque est eum explicabo fugit id ipsa iste magnam nobis non nulla, rerum sapiente, tempora voluptatum? Autem culpa ea explicabo facere, iste laudantium molestias obcaecati placeat quam reprehenderit sed sequi. Architecto consectetur consequuntur cumque delectus, minus nostrum porro quae quaerat ratione repellendus saepe sed suscipit vel voluptatem. Ad blanditiis consequuntur culpa dolor eius ex excepturi expedita ipsam ipsum molestias nemo nulla omnis, pariatur placeat quasi quia quidem quis tempore ut veniam! Accusantium commodi culpa dicta doloremque nam obcaecati odit quibusdam repudiandae, sequi temporibus, velit vero voluptatem? A ab asperiores commodi deleniti facere facilis fuga illo minima odio provident quia quis quod sapiente vel veniam voluptas, voluptatum." |
| 10 | "Adipisicing elit. Ad consequatur deleniti excepturi harum incidunt inventore odio rem rerum sapiente suscipit! Accusamus animi atque cumque dolores et eveniet excepturi explicabo magni minus molestiae molestias, quae quibusdam reiciendis! Accusamus amet animi deserunt hic nostrum quae similique veniam? A ab adipisci aperiam blanditiis consequatur cumque, dolorum eaque esse et ex fugit inventore iure labore laborum modi molestiae mollitia nesciunt nobis nostrum pariatur perspiciatis porro quas quasi quod quos recusandae rem repudiandae rerum sapiente sint sunt tempore unde veritatis vitae voluptate voluptatem voluptatum! Alias aperiam aut autem blanditiis culpa ducimus enim illo impedit iure iusto laboriosam magni molestiae neque nihil nobis, odio quas quia quidem quos repellat reprehenderit sapiente sed sequi tempore temporibus veniam, veritatis? Fugiat minima modi molestiae nulla repudiandae. Ab accusantium adipisci, aliquam aliquid consectetur consequatur culpa cupiditate dolores doloribus ea esse eum eveniet incidunt inventore ipsam maiores necessitatibus officiis quidem quos repellendus rerum sequi, similique veniam? Accusantium aliquam animi architecto asperiores assumenda consectetur consequatur cumque cupiditate deleniti dignissimos dolor dolore error exercitationem, explicabo facilis ipsum laudantium maiores nulla, odit perferendis praesentium quae quaerat quis quisquam |

| | |
|---|---|
| | sequi sint tempora voluptate? Ab adipisci blanditiis consectetur culpa debitis delectus deleniti distinctio dolore eius error esse, facere id inventore ipsa ipsam iure magnam minima minus necessitatibus nisi nostrum nulla numquam obcaecati omnis optio perspiciatis praesentium provident quia quo ratione saepe, sapiente vel voluptatum. Cum, dolores eaque esse eum, ex exercitationem expedita facere fugiat iste laborum maxime molestiae omnis perferendis provident repellendus similique sint ut. Laudantium, similique?" |

Задания:
1) Определите топ-5 слов, наиболее часто встречающихся в текстах документов.

```
CREATE TABLE documents (
    id INTEGER PRIMARY KEY,
    content TEXT
);
```

INSERT INTO documents (id, content) VALUES

(1, 'Lorem ipsum dolor sit amet, consectetur adipisicing elit. Accusantium ad at beatae consequuntur cupiditate dolore dolores dolorum, ea error id ipsum iure iusto minima odit perspiciatis quidem sunt vel voluptate!'),

(2, 'Lorem ipsum dolor sit amet, consectetur adipisicing elit. Beatae, blanditiis corporis deleniti doloribus, eos et eveniet fugiat id inventore itaque molestiae natus nobis nostrum odit pariatur quod quos reiciendis sit vitae 43 voluptas. Eos eveniet praesentium provident. Illo inventore ipsam minima rem rerum. Amet autem dolor doloribus eveniet facere iste voluptatem!'),

(3, 'Aliquid atque beatae, debitis facilis fuga in iure molestiae nostrum omnis? Aliquam assumenda, consectetur cupiditate dignissimos dolor, harum illo itaque laborum necessitatibus obcaecati similique sunt, velit voluptatum. Aut cumque dolor dolorum error, exercitationem iusto laudantium maiores non perferendis quae, qui quod reiciendis sunt suscipit ullam ut voluptate? Accusantium, aliquam atque, deleniti exercitationem hic incidunt ipsam iste itaque iusto neque nobis officiis quam quas quibusdam quidem soluta voluptatem?'),

(4, 'Lorem ipsum lestiae saepe sequi sint, temporibus. Accusantium ad commodi, consectetur deserunt dolores doloribus ea eaque eligendi enim in ipsam minima quam quas qui voluptatem? Amet consectetur cumque, dicta dolore expedita fuga fugiat iusto minima nostrum odio, quia, quidem sequi tempora. A dolorum, ea eveniet facilis fugiat ipsa mollitia obcaecati quia quo sunt tempore, totam! Ad alias aliquid aperiam asperiores at corporis distinctio dolorem dolorum error eum fuga ipsum iure laudantium libero nam necessitatibus officiis perspiciatis quos repudiandae saepe sequi soluta tempore, vero.'),

(5, 'Lorem sit amet, consectetur adipisicing elit. Amet, atque corporis cumque deserunt dolores ducimus esse est facere illo laudantium maiores molestias mollitia odio perferendis possimus repellendus sapiente sequi sit suscipit vero? Architecto blanditiis corporis culpa cum debitis deserunt dignissimos eligendi esse est excepturi expedita hic iusto, modi mollitia optio perferendis quas qui quia quibusdam quidem sequi sint suscipit tempora tempore minus mollitia nisi quam quod rem tempore tenetur, ullam ut veniam, voluptatem voluptatibus. Atque consequatur eveniet facilis odit unde. Adipisci culpa deserunt expedita explicabo neque obcaecati odio tempore! Aliquam at, aut beatae blanditiis error, eum excepturi facere incidunt ipsam laboriosam nam nemo nobis nostrum numquam odit provident quia quibusdam ratione recusandae rem repudiandae rerum sequi unde.'),

(6, 'Consectetur adipisicing elit. Aliquam beatae delectus, dolor ipsa itaque minima necessitatibus officiis praesentium quisquam reiciendis. Accusantium autem dolorum id ipsa ipsum, labore nam officia placeat porro quas quos ratione sapiente sint. Aspernatur consectetur deleniti error inventore laudantium obcaecati

officiis possimus, totam? Assumenda atque delectus eaque enim est et eum facilis iusto laborum laudantium modi mollitia non nostrum odio praesentium, quas quia recusandae sit, unde voluptas! Animi est fugiat fugit mollitia nihil nulla, sapiente sed soluta. A aliquam aperiam at dolores est excepturi inventore ipsum voluptatem. A adipisci aliquam aspernatur assumenda aut beatae doloremque dolores error exercitationem fuga ipsa iste laborum maiores nobis non pariatur quae, quas quia quo repellendus similique sint soluta veritatis? A ad adipisci, cum distinctio eaque, illo in molestiae neque optio praesentium quia sed tempore voluptates. Autem error nihil officiis perferendis quis tenetur ut!'),

(7, 'Ab animi architecto assumenda consequatur cumque deserunt distinctio, doloremque ex explicabo illo libero molestias non nulla officiis praesentium provident quae quas quasi quibusdam quod rerum sequi tenetur voluptatum. Ad alias aliquam amet at atque autem blanditiis deserunt dolores doloribus eligendi enim est ex explicabo facilis fugit, illum in incidunt ipsa iusto magni maiores maxime nemo non officia omnis optio pariatur quae quibusdam quisquam quo sed sint sit sunt temporibus ut vitae voluptatem. Aspernatur culpa deserunt dolorum earum eos hic impedit ipsa ipsum maxime, minima neque nesciunt omnis, quam reiciendis sed similique ut voluptas! A at doloribus facere laborum molestiae quas similique. Earum excepturi facere harum, inventore ipsum necessitatibus praesentium quidem reiciendis tenetur veritatis. Adipisci asperiores commodi dolor error facilis quae totam voluptates. Dolor ipsum quae quaerat quod reprehenderit saepe sequi, similique vel voluptatem voluptatibus. Adipisci architecto debitis dignissimos ducimus ea eaque eos eum excepturi illum, impedit incidunt molestias mollitia nesciunt nobis placeat quia quis quo quod recusandae reiciendis, rem sequi totam veniam! Commodi expedita impedit optio repudiandae?'),

(8, 'Eveniet facere fuga ipsa ipsam labore minus, natus necessitatibus numquam placeat possimus quia quod reiciendis unde veniam veritatis voluptate! Accusantium ad beatae consectetur dicta modi molestiae praesentium repellendus saepe ullam. Assumenda autem beatae cum dolore dolorem doloribus, facilis hic id labore laboriosam laborum laudantium maiores minus nemo numquam optio quae quaerat quis quo rem sunt vel voluptas voluptatibus. Est inventore laboriosam ut. Aliquid, culpa, cupiditate delectus doloremque ea qui quidem quis tempore ullam vel veritatis vitae voluptatem? Commodi ipsum libero nihil sapiente? Accusamus aperiam expedita facilis in minus molestiae molestias mollitia optio, quia, quibusdam quis sunt voluptatibus! Aliquid consectetur deleniti ea hic quia rem, reprehenderit sit sunt vel velit! Architecto autem dolor doloremque dolores esse eum ex explicabo facere, facilis ipsum itaque minus modi mollitia nihil odio pariatur praesentium quaerat, quasi qui quod, reiciendis soluta temporibus vel veniam voluptas voluptates voluptatum? A, ab accusamus error eveniet ex facilis, minima minus non obcaecati placeat, quas quasi quidem repellendus. Ducimus eum laudantium maiores maxime, obcaecati possimus quidem rerum sapiente unde ut. Illo modi quis recusandae sequi!'),

(9, 'Ea harum ipsum, iusto laboriosam maiores minima neque nisi quam quas ut voluptas voluptatem. Architecto assumenda autem corporis debitis delectus ducimus eum excepturi fuga, harum inventore maxime molestiae nisi nobis nostrum officiis perspiciatis quas quod repudiandae rerum sapiente sequi unde vitae. Accusamus aperiam commodi eos, esse explicabo ipsum laboriosam magni, molestiae pariatur possimus quisquam repellat tempore ullam velit voluptate. Accusamus adipisci animi commodi cumque cupiditate ducimus enim facere, fugit ipsa iusto minima mollitia necessitatibus neque nesciunt numquam praesentium quaerat quas, quos rem repellat saepe tempora tempore voluptatem! Iusto maiores nihil quae. Atque corporis cumque, deleniti dolorem doloremque est eum explicabo fugit id ipsa iste magnam nobis non nulla, rerum sapiente, tempora voluptatum? Autem culpa ea explicabo facere, iste laudantium molestias obcaecati placeat quam reprehenderit sed sequi. Architecto consectetur consequuntur cumque delectus, minus nostrum porro quae quaerat ratione repellendus saepe sed suscipit vel voluptatem. Ad blanditiis consequuntur culpa dolor eius ex excepturi expedita ipsam ipsum molestias nemo nulla omnis, pariatur placeat quasi quia quidem quis tempore ut veniam! Accusantium commodi culpa dicta doloremque nam obcaecati odit quibusdam repudiandae, sequi temporibus, velit vero voluptatem? A ab asperiores commodi deleniti facere facilis fuga illo minima odio provident quia quis quod sapiente vel veniam voluptas, voluptatum.'),

(10, 'Adipisicing elit. Ad consequatur deleniti excepturi harum incidunt inventore odio rem rerum sapiente suscipit! Accusamus animi atque cumque dolores et eveniet excepturi explicabo magni minus molestiae molestias, quae quibusdam reiciendis! Accusamus amet animi deserunt hic nostrum quae similique veniam? A ab adipisci aperiam blanditiis consequatur cumque, dolorum eaque esse et ex fugit inventore iure labore laborum modi molestiae mollitia nesciunt nobis nostrum pariatur perspiciatis porro quas quasi quod quos recusandae rem repudiandae rerum sapiente sint sunt tempore unde veritatis vitae voluptate

voluptatem voluptatum! Alias aperiam aut autem blanditiis culpa ducimus enim illo impedit iure iusto laboriosam magni molestiae neque nihil nobis, odio quas quia quidem quos repellat reprehenderit sapiente sed sequi tempore temporibus veniam, veritatis? Fugiat minima modi molestiae nulla repudiandae. Ab accusantium adipisci, aliquam aliquid consectetur consequatur culpa cupiditate dolores doloribus ea esse eum eveniet incidunt inventore ipsam maiores necessitatibus officiis quidem quos repellendus rerum sequi, similique veniam? Accusantium aliquam animi architecto asperiores assumenda consectetur consequatur cumque cupiditate deleniti dignissimos dolor dolore error exercitationem, explicabo facilis ipsum laudantium maiores nulla, odit perferendis praesentium quae quaerat quis quisquam sequi sint tempora voluptate? Ab adipisci blanditiis consectetur culpa debitis delectus deleniti distinctio dolore eius error esse, facere id inventore ipsa ipsam iure magnam minima minus necessitatibus nisi nostrum nulla numquam obcaecati omnis optio perspiciatis praesentium provident quia quo ratione saepe, sapiente vel voluptatum. Cum, dolores eaque esse eum, ex exercitationem expedita facere fugiat iste laborum maxime molestiae omnis perferendis provident repellendus similique sint ut. Laudantium, similique?');

```
SELECT * FROM documents
```

1) Определите топ-5 слов, наиболее часто встречающихся в текстах документов.

```
WITH all_words AS (
  SELECT
    LOWER(unnested_word) AS word
  FROM documents,
  LATERAL regexp_split_to_table(content, E'\\s+') AS unnested_word
),
cleaned_words AS (
  SELECT regexp_replace(word, '[^a-zA-Z]+', '', 'g') AS word
  FROM all_words
),
filtered_words AS (
  SELECT word
  FROM cleaned_words
  WHERE word <> ''
)
SELECT word, COUNT(*) AS frequency
FROM filtered_words
GROUP BY word
ORDER BY frequency DESC
LIMIT 5;
```

-- все слова приведены к нижнему регистру и очищены от знаков препинания.

-- Частотность подсчитана по всем словам всех документов. Поэтому получен корректный топ-5 часто встречающихся слов

Showing rows: 1 to 5 Page No

| | word text | frequency bigint |
|---|---|---|
| 1 | sequi | 16 |
| 2 | ipsum | 16 |
| 3 | quia | 15 |
| 4 | consectetur | 14 |
| 5 | quas | 14 |

**Задача 29**

Даны таблицы users, ad_clicks, purchases.

| Users | | |
|---|---|---|
| "id" | "name" | "registration_date" |
| 1 | "John" | "2022-05-10" |
| 2 | "Max" | "2022-01-10" |
| 3 | "Andrew" | "2022-10-31" |
| 4 | "Joe" | "2022-11-30" |
| 5 | "Karl" | "2022-07-31" |

| ad_clicks | | |
|---|---|---|
| "id" | "user_id" | "click_date" |
| 1 | 3 | "2023-12-06" |
| 2 | 1 | "2023-12-05" |
| 3 | 2 | "2023-12-01" |
| 4 | 5 | "2023-11-12" |
| 5 | 4 | "2023-11-11" |
| 6 | 1 | "2023-10-14" |
| 7 | 2 | "2023-10-10" |
| 8 | 3 | "2023-11-30" |
| 9 | 4 | "2023-12-05" |
| 10 | 4 | "2023-12-01" |
| 11 | 5 | "2023-12-08" |
| 12 | 3 | "2023-11-27" |
| 13 | 2 | "2023-11-28" |
| 14 | 1 | "2023-12-09" |
| 15 | 5 | "2023-12-10" |
| 16 | 5 | "2023-11-30" |
| 17 | 2 | "2023-11-12" |

| purchases | | |
|---|---|---|
| "id" | "user_id" | "date" |
| 1 | 3 | "2023-12-12" |
| 2 | 1 | "2023-12-05" |
| 3 | 2 | "2023-12-01" |
| 4 | 5 | "2023-11-12" |
| 5 | 4 | "2023-11-11" |
| 6 | 1 | "2023-10-14" |
| 7 | 2 | "2023-10-10" |
| 8 | 3 | "2023-11-30" |
| 9 | 4 | "2023-12-05" |
| 10 | 4 | "2023-12-01" |
| 11 | 5 | "2023-12-08" |
| 12 | 3 | "2023-11-27" |
| 13 | 2 | "2023-11-28" |
| 14 | 1 | "2023-12-09" |
| 15 | 5 | "2023-12-10" |
| 16 | 5 | "2023-11-30" |
| 17 | 2 | "2023-11-12" |

Задания:
1) Оцените эффективность рекламной кампании на основе конверсии и затрат на клики.

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name TEXT,
    registration_date DATE
);

INSERT INTO users (id, name, registration_date) VALUES
(1, 'John', '2022-05-10'),
(2, 'Max', '2022-01-10'),
(3, 'Andrew', '2022-10-31'),
(4, 'Joe', '2022-11-30'),
(5, 'Karl', '2022-07-31');

CREATE TABLE ad_clicks (
    id INTEGER PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    click_date DATE
);

INSERT INTO ad_clicks (id, user_id, click_date) VALUES
(1, 3, '2023-12-06'),
(2, 1, '2023-12-05'),
(3, 2, '2023-12-01'),
(4, 5, '2023-11-12'),
(5, 4, '2023-11-11'),
(6, 1, '2023-10-14'),
(7, 2, '2023-10-10'),
(8, 3, '2023-11-30'),
(9, 4, '2023-12-05'),
(10, 4, '2023-12-01'),
```

```
(11, 5, '2023-12-08'),
(12, 3, '2023-11-27'),
(13, 2, '2023-11-28'),
(14, 1, '2023-12-09'),
(15, 5, '2023-12-10'),
(16, 5, '2023-11-30'),
(17, 2, '2023-11-12');

CREATE TABLE purchases (
    id INTEGER PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    date DATE
);

INSERT INTO purchases (id, user_id, date) VALUES
(1, 3, '2023-12-12'),
(2, 1, '2023-12-05'),
(3, 2, '2023-12-01'),
(4, 5, '2023-11-12'),
(5, 4, '2023-11-11'),
(6, 1, '2023-10-14'),
(7, 2, '2023-10-10'),
(8, 3, '2023-11-30'),
(9, 4, '2023-12-05'),
(10, 4, '2023-12-01'),
(11, 5, '2023-12-08'),
(12, 3, '2023-11-27'),
(13, 2, '2023-11-28'),
(14, 1, '2023-12-09'),
(15, 5, '2023-12-10'),
(16, 5, '2023-11-30'),
(17, 2, '2023-11-12');
```

```
SELECT * FROM users
SELECT * FROM ad_clicks
SELECT * FROM purchases
```

1) Оцените эффективность рекламной кампании на основе конверсии и затрат на клики.

```
WITH clicked_users AS (
    SELECT DISTINCT user_id
    FROM ad_clicks
),
converted_users AS (
    SELECT DISTINCT a.user_id
    FROM ad_clicks a
    JOIN purchases p ON a.user_id = p.user_id
),
total_clicks AS (
    SELECT COUNT(*) AS total_clicks FROM ad_clicks
),
conversion_data AS (
    SELECT
        (SELECT COUNT(*) FROM clicked_users) AS total_users_clicked,
        (SELECT COUNT(*) FROM converted_users) AS users_converted
),
efficiency AS (
    SELECT
        c.total_users_clicked,
        c.users_converted,
        ROUND(100.0 * c.users_converted / NULLIF(c.total_users_clicked, 0), 2) AS conversion_rate,
        t.total_clicks AS total_clicks,
        t.total_clicks * 1 AS total_cost,
        CASE WHEN c.users_converted > 0 THEN ROUND(t.total_clicks * 1.0 / c.users_converted, 2)
```

```
            ELSE NULL
        END AS cost_per_conversion
    FROM conversion_data c, total_clicks t
)
SELECT * FROM efficiency;
-- в задаче не указана цена, поэтому по умолчанию каждый клик стоит 1 у.е --
```

Data Output Messages Notifications

| | total_users_clicked bigint | users_converted bigint | conversion_rate numeric | total_clicks bigint | total_cost bigint | cost_per_conversion numeric |
|---|---|---|---|---|---|---|
| 1 | 5 | 5 | 100.00 | 17 | 17 | 3.40 |

**Задача 30**

Даны таблицы stock_prices, news_events.

| stock_prices | | | |
|---|---|---|---|
| "id" | "stock_symbol" | "stock_price" | "price_date" |
| 1 | "AAA" | 123.31 | "2023-12-12" |
| 2 | "AAA" | 67.31 | "2023-01-11" |
| 3 | "APPL" | 33.11 | "2023-05-31" |
| 4 | "AMZN" | 891.31 | "2023-01-11" |
| 5 | "MCSF" | 55.21 | "2023-06-09" |
| 6 | "APPL" | 190.44 | "2023-10-09" |
| 7 | "AMZN" | 670.00 | "2023-12-12" |
| 8 | "GOOG" | 88.55 | "2023-01-10" |
| 9 | "LEA" | 131.33 | "2023-02-21" |
| 10 | "GOA" | 29.67 | "2023-11-10" |
| 11 | "POA" | 33.11 | "2023-11-30" |
| 12 | "LET" | 84.37 | "2023-04-30" |
| 13 | "LEF" | 90.00 | "2023-04-29" |
| 14 | "LED" | 10.00 | "2023-08-31" |
| 15 | "LEF" | 55.44 | "2023-09-30" |
| 16 | "GOOG" | 32.11 | "2023-07-31" |
| 17 | "MCSF" | 129.41 | "2023-06-30" |
| 18 | "GOA" | 67.31 | "2023-10-30" |
| 19 | "POA" | 54.66 | "2023-11-30" |
| 20 | "PP" | 31.44 | "2023-12-01" |
| 21 | "RS" | 100.21 | "2023-12-01" |
| 22 | "PP" | 637.31 | "2023-01-11" |
| 23 | "RS" | 93.31 | "2023-01-31" |

| news_events | | |
|---|---|---|
| "id" | "description" | "event_date" |
| 1 | "Lorem ipsum." | "2023-01-16" |
| 2 | "Lorem ipsum." | "2023-06-15" |
| 3 | "Lorem ipsum." | "2023-02-01" |
| 4 | "Lorem ipsum." | "2023-11-02" |
| 5 | "Lorem ipsum." | "2023-12-02" |

Задания:

1) Определите, существует ли корреляция между изменениями цен на акции и событиями для определенных компаний.

```
CREATE TABLE stock_prices (
    id INTEGER PRIMARY KEY,
    stock_symbol TEXT,
    stock_price NUMERIC,
```

```
    price_date DATE
);

INSERT INTO stock_prices (id, stock_symbol, stock_price, price_date) VALUES
(1, 'AAA', 123.31, '2023-12-12'),
(2, 'AAA', 67.31, '2023-01-11'),
(3, 'APPL', 33.11, '2023-05-31'),
(4, 'AMZN', 891.31, '2023-01-11'),
(5, 'MCSF', 55.21, '2023-06-09'),
(6, 'APPL', 190.44, '2023-10-09'),
(7, 'AMZN', 670.00, '2023-12-12'),
(8, 'GOOG', 88.55, '2023-01-10'),
(9, 'LEA', 131.33, '2023-02-21'),
(10, 'GOA', 29.67, '2023-11-10'),
(11, 'POA', 33.11, '2023-11-30'),
(12, 'LET', 84.37, '2023-04-30'),
(13, 'LEF', 90.00, '2023-04-29'),
(14, 'LED', 10.00, '2023-08-31'),
(15, 'LEF', 55.44, '2023-09-30'),
(16, 'GOOG', 32.11, '2023-07-31'),
(17, 'MCSF', 129.41, '2023-06-30'),
(18, 'GOA', 67.31, '2023-10-30'),
(19, 'POA', 54.66, '2023-11-30'),
(20, 'PP', 31.44, '2023-12-01'),
(21, 'RS', 100.21, '2023-12-01'),
(22, 'PP', 637.31, '2023-01-11'),
(23, 'RS', 93.31, '2023-01-31');

CREATE TABLE news_events (
    id INTEGER PRIMARY KEY,
    description TEXT,
    event_date DATE
```

```
);

INSERT INTO news_events (id, description, event_date) VALUES
(1, 'Lorem ipsum.', '2023-01-16'),
(2, 'Lorem ipsum.', '2023-06-15'),
(3, 'Lorem ipsum.', '2023-02-01'),
(4, 'Lorem ipsum.', '2023-11-02'),
(5, 'Lorem ipsum.', '2023-12-02');

SELECT * FROM stock_prices
SELECT * FROM news_events
```

1) Определите, существует ли корреляция между изменениями цен на акции и событиями для определенных компаний.

```
WITH price_before_event AS (
  SELECT
    s.stock_symbol,
    e.event_date,
    s.stock_price AS price_before,
    s.price_date
  FROM stock_prices s
  JOIN news_events e
    ON s.price_date BETWEEN e.event_date - INTERVAL '5 days' AND e.event_date - INTERVAL '1 day'

),
price_after_event AS (
  SELECT
    s.stock_symbol,
    e.event_date,
    s.stock_price AS price_after,
    s.price_date
```

```
  FROM stock_prices s
  JOIN news_events e
    ON s.price_date BETWEEN e.event_date AND e.event_date + INTERVAL '10 days'
),
paired_prices AS (
  SELECT
    pbe.stock_symbol,
    pbe.event_date,
    pbe.price_before,
    pae.price_after,
    ROUND(pae.price_after - pbe.price_before, 2) AS price_change
  FROM price_before_event pbe
  JOIN price_after_event pae
    ON pbe.stock_symbol = pae.stock_symbol AND pbe.event_date = pae.event_date
)
SELECT
  stock_symbol,
  event_date,
  price_before,
  price_after,
  price_change,
  CASE
    WHEN price_change > 0 THEN 'increase'
    WHEN price_change < 0 THEN 'decrease'
    ELSE 'no change'
  END AS change_type
FROM paired_prices
ORDER BY stock_symbol, event_date;
-- Анализ тут выполняется через сравнение цен акций до и после события (± дни).
-- Это позволяет выявить реакцию рынка на новости без статистических коэффициентов,
-- что логично в рамках SQL-анализа без внешних инструментов.
```

Showing rows: 1 to 1 Page No: 1 of 1

| | stock_symbol text | event_date date | price_before numeric | price_after numeric | price_change numeric | change_type text |
|---|---|---|---|---|---|---|
| 1 | GOA | 2023-11-02 | 67.31 | 29.67 | -37.64 | decrease |

Хакимий фармита