Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа, обработки и интерпретации больших данных**

О Т Ч Е Т

по лабораторной работе № 3

Название: Запросы PostgreSQL

Дисциплина: Технология параллельных систем баз данных

Студент ИУ6-12М (Группа) _____ (Подпись, дата) С.В. Астахов (И.О. Фамилия)

Преподаватель _____ (Подпись, дата) _____ (И.О. Фамилия)

Москва, 2023

Введение

1. Цель работы: научиться разрабатывать сложные запросы SELECT.

Ход выполнения

2. Создание БД.

Создадим в файловой системе папку “lab3” и поместим туда файл с дампом БД. Создадим из дампа, подключимся к ней и просмотрим список схем, где обнаружим схему bookings (рисунок 1).

```
postgres@Ubuntu1:~$ psql -f /home/lab3/demo-small.sql -U postgres > demo.log 2>demo.err
postgres@Ubuntu1:~$ psql -d demo -U postgres
psql (9.6.24)
Type "help" for help.

demo=# /dn
demo-# \dn
   List of schemas
   Name    |  Owner
-----------+----------
 bookings | postgres
 public   | postgres
(2 rows)
```

Рисунок 1 — создание БД из дампа

Затем, переключимся на схему bookings и посмотрим список таблиц в ней.

```
demo=# SET search_path TO bookings;
SET
demo=# \dt+
                        List of relations
  Schema  |      Name       | Type  |  Owner   |    Size    | Description
----------+-----------------+-------+----------+------------+-------------
 bookings | aircrafts       | table | postgres | 16 kB      |
 bookings | airports        | table | postgres | 8192 bytes |
 bookings | boarding_passes | table | postgres | 33 MB      |
 bookings | bookings        | table | postgres | 13 MB      |
 bookings | flights         | table | postgres | 3160 kB    |
 bookings | seats           | table | postgres | 88 kB      |
 bookings | ticket_flights  | table | postgres | 68 MB      |
 bookings | tickets         | table | postgres | 48 MB      |
(8 rows)
```

Рисунок 2 — список таблиц в БД

3. Запросы

Просмотрим содержимое таблиц самолетов и аэропортов (рисунок 3).

Примечание: здесь и далее многие запросы из методических указаний дополнены ограничением “LIMIT” для удобства просмотра.

Выполним запросы, позволяющий понять работу опраторов “LIKE” и “~”, сравнивающих строку с заданным шаблоном. В операторе LIKE: “%” —

любое число любых символов, “_” — строго один символ. В операторе “~”: “^” — начало строки, “$” — конец строки, “|” — дизъюнкция.

```
demo=# SELECT * FROM airports limit 10;
 airport_code |  airport_name   |    city     | longitude  | latitude  |      timezone
--------------+-----------------+-------------+------------+-----------+--------------------
 MJZ          | Мирный          | Мирный      | 114.038928 | 62.534689 | Asia/Yakutsk
 NBC          | Бегишево        | Нижнекамск  |      52.06 |     55.34 | Europe/Moscow
 NOZ          | Спиченково      | Новокузнецк |    86.8772 |   53.8114 | Asia/Novokuznetsk
 NAL          | Нальчик         | Нальчик     |    43.6366 |   43.5129 | Europe/Moscow
 OGZ          | Беслан          | Владикавказ |    44.6066 |   43.2051 | Europe/Moscow
 CSY          | Чебоксары       | Чебоксары   |    47.3473 |   56.0903 | Europe/Moscow
 NYM          | Надым           | Надым       |    72.6989 |   65.4809 | Asia/Yekaterinburg
 NYA          | Нягань          | Нягань      |     65.615 |     62.11 | Asia/Yekaterinburg
 URS          | Курск-Восточный | Курск       |    36.2956 |   51.7506 | Europe/Moscow
 SKX          | Саранск         | Саранск     |    45.2123 |   54.1251 | Europe/Moscow
(10 rows)

demo=# SELECT * FROM aircrafts limit 10;
 aircraft_code |        model        | range
---------------+---------------------+-------
 773           | Boeing 777-300      | 11100
 763           | Boeing 767-300      |  7900
 SU9           | Sukhoi SuperJet-100 |  3000
 320           | Airbus A320-200     |  5700
 321           | Airbus A321-200     |  5600
 319           | Airbus A319-100     |  6700
 733           | Boeing 737-300      |  4200
 CN1           | Cessna 208 Caravan  |  1200
 CR2           | Bombardier CRJ-200  |  2700
(9 rows)
```

Рисунок 3 — содержимое таблиц аэропортов и самолетов

```
demo=# SELECT * FROM aircrafts WHERE model LIKE 'Airbus%';
 aircraft_code |      model      | range
---------------+-----------------+-------
 320           | Airbus A320-200 |  5700
 321           | Airbus A321-200 |  5600
 319           | Airbus A319-100 |  6700
(3 rows)

demo=# SELECT * FROM airports WHERE airport_name LIKE 'Уфа';
 airport_code | airport_name | city | longitude | latitude  |      timezone
--------------+--------------+------+-----------+-----------+--------------------
 UFA          | Уфа          | Уфа  | 55.874417 | 54.557511 | Asia/Yekaterinburg
(1 row)

demo=# SELECT * FROM aircrafts WHERE model ~ '^(A|Boe)';
 aircraft_code |      model      | range
---------------+-----------------+-------
 773           | Boeing 777-300  | 11100
 763           | Boeing 767-300  |  7900
 320           | Airbus A320-200 |  5700
 321           | Airbus A321-200 |  5600
 319           | Airbus A319-100 |  6700
 733           | Boeing 737-300  |  4200
(6 rows)

demo=# SELECT * FROM aircrafts WHERE model !~ '300$';
 aircraft_code |        model        | range
---------------+---------------------+-------
 SU9           | Sukhoi SuperJet-100 |  3000
 320           | Airbus A320-200     |  5700
 321           | Airbus A321-200     |  5600
 319           | Airbus A319-100     |  6700
 CN1           | Cessna 208 Caravan  |  1200
 CR2           | Bombardier CRJ-200  |  2700
(6 rows)
```

Рисунок 4 — запросы с использованием строковых шаблонов

Ознакомимся с оператором “BETWEEN” и получением вычисляемых колонок в результате Select-запроса (рисунок 5).

```
demo=# SELECT * FROM aircrafts WHERE range BETWEEN 3000 AND 6000;
 aircraft_code |        model        | range
---------------+---------------------+-------
 SU9           | Sukhoi SuperJet-100 |  3000
 320           | Airbus A320-200     |  5700
 321           | Airbus A321-200     |  5600
 733           | Boeing 737-300      |  4200
(4 rows)

demo=# SELECT model, range, range / 1.609 AS miles FROM aircrafts;
        model        | range |         miles
---------------------+-------+-----------------------
 Boeing 777-300      | 11100 | 6898.6948415164698571
 Boeing 767-300      |  7900 | 4909.8819142324425109
 Sukhoi SuperJet-100 |  3000 | 1864.5121193287756370
 Airbus A320-200     |  5700 | 3542.5730267246737104
 Airbus A321-200     |  5600 | 3480.4226227470478558
 Airbus A319-100     |  6700 | 4164.0770665009322561
 Boeing 737-300      |  4200 | 2610.3169670602858919
 Cessna 208 Caravan  |  1200 |  745.8048477315102548
 Bombardier CRJ-200  |  2700 | 1678.0609073958980733
(9 rows)

demo=# SELECT model, range, round( range / 1.609, 2 ) AS miles FROM aircrafts;
        model        | range |  miles
---------------------+-------+---------
 Boeing 777-300      | 11100 | 6898.69
 Boeing 767-300      |  7900 | 4909.88
 Sukhoi SuperJet-100 |  3000 | 1864.51
 Airbus A320-200     |  5700 | 3542.57
 Airbus A321-200     |  5600 | 3480.42
 Airbus A319-100     |  6700 | 4164.08
 Boeing 737-300      |  4200 | 2610.32
 Cessna 208 Caravan  |  1200 |  745.80
 Bombardier CRJ-200  |  2700 | 1678.06
(9 rows)
```

Рисунок 5 — получение вычисляемых колонок в select-запросе

Ознакомимся с оператором упорядочивания результатов запроса “ORDER BY” (рисунок 6).

```
demo=# SELECT * FROM aircrafts ORDER BY range DESC;
 aircraft_code |        model        | range
---------------+---------------------+-------
 773           | Boeing 777-300      | 11100
 763           | Boeing 767-300      |  7900
 319           | Airbus A319-100     |  6700
 320           | Airbus A320-200     |  5700
 321           | Airbus A321-200     |  5600
 733           | Boeing 737-300      |  4200
 SU9           | Sukhoi SuperJet-100 |  3000
 CR2           | Bombardier CRJ-200  |  2700
 CN1           | Cessna 208 Caravan  |  1200
(9 rows)
```

Рисунок 6 — упорядочивание результатов запроса

Ознакомимся с оператором “DISTINCT”, позволяющим выбрать уникальные значения (рисунки 6-7).

```
demo=# SELECT timezone FROM airports limit 10;
      timezone
--------------------
 Asia/Yakutsk
 Europe/Moscow
 Asia/Novokuznetsk
 Europe/Moscow
 Europe/Moscow
 Europe/Moscow
 Asia/Yekaterinburg
 Asia/Yekaterinburg
 Europe/Moscow
 Europe/Moscow
(10 rows)
```

Рисунок 6 — запрос с сохранением дублирующихся строк

```
demo=# SELECT DISTINCT timezone FROM airports ORDER BY 1;
      timezone
--------------------
 Asia/Anadyr
 Asia/Chita
 Asia/Irkutsk
 Asia/Kamchatka
 Asia/Krasnoyarsk
 Asia/Magadan
 Asia/Novokuznetsk
 Asia/Novosibirsk
 Asia/Omsk
 Asia/Sakhalin
 Asia/Vladivostok
 Asia/Yakutsk
 Asia/Yekaterinburg
 Europe/Kaliningrad
 Europe/Moscow
 Europe/Samara
 Europe/Volgograd
(17 rows)
```

Рисунок 7 — запрос с выборкой уникальных строк

Ознакомимся с операторами “LIMIT” и “OFFSET”, выбирая 3 самых восточных и 3 следующих аэропорта (рисунок 8).

```
demo=# SELECT airport[redacted]ame, city, longitude
FROM airports
ORDER BY longitude DESC
LIMIT 3;
 airport[redacted]ame |           city           | longitude
--------------+--------------------------+-----------
 Анадырь      | Анадырь                  | 177.741483
 Елизово      | Петропавловск-Камчатский | 158.453669
 Магадан      | Магадан                  | 150.720439
(3 rows)

demo=# SELECT airport[redacted]ame, city, longitude
FROM airports
ORDER BY longitude DESC
LIMIT 3
OFFSET 3;
  airport[redacted]ame   |         city         | longitude
-----------------+----------------------+-----------
 Хомутово        | Южно-Сахалинск       | 142.717531
 Хурба           | Комсомольск-на-Амуре |    136.934
 Хабаровск-Новый | Хабаровск            | 135.188361
(3 rows)
```

Рисунок 8 — использование limit и offset

Ознакомимся в оператором “CASE”, классифицурая самолеты по дальности полета (рисунок 9).

```
demo=# SELECT model, range,
CASE WHEN range < 2000 THEN 'Ближнемагистральный'
WHEN range < 5000 THEN 'Среднемагистральный'
ELSE 'Дальнемагистральный'
END AS type
FROM aircrafts
ORDER BY model;
        model         | range |        type
----------------------+-------+---------------------
 Airbus A319-100      |  6700 | Дальнемагистральный
 Airbus A320-200      |  5700 | Дальнемагистральный
 Airbus A321-200      |  5600 | Дальнемагистральный
 Boeing 737-300       |  4200 | Среднемагистральный
 Boeing 767-300       |  7900 | Дальнемагистральный
 Boeing 777-300       | 11100 | Дальнемагистральный
 Bombardier CRJ-200   |  2700 | Среднемагистральный
 Cessna 208 Caravan   |  1200 | Ближнемагистральный
 Sukhoi SuperJet-100  |  3000 | Среднемагистральный
(9 rows)
```

Рисунок 9 — классификация самолетов по дальности полета

Ознакомимся с оператором соединения “JOIN”, запросив список сидений для заданной модели самолета (рисунок 10).

```
demo=# SELECT a.aircraft_code, a.model, s.seat_no, s.fare_conditions
FROM seats AS s
JOIN aircrafts AS a
ON s.aircraft_code = a.aircraft_code
WHERE a.model ~ '^Cessna'
ORDER BY s.seat_no;
 aircraft_code |       model        | seat_no | fare_conditions
---------------+--------------------+---------+-----------------
 CN1           | Cessna 208 Caravan | 1A      | Economy
 CN1           | Cessna 208 Caravan | 1B      | Economy
 CN1           | Cessna 208 Caravan | 2A      | Economy
 CN1           | Cessna 208 Caravan | 2B      | Economy
 CN1           | Cessna 208 Caravan | 3A      | Economy
 CN1           | Cessna 208 Caravan | 3B      | Economy
 CN1           | Cessna 208 Caravan | 4A      | Economy
 CN1           | Cessna 208 Caravan | 4B      | Economy
 CN1           | Cessna 208 Caravan | 5A      | Economy
 CN1           | Cessna 208 Caravan | 5B      | Economy
 CN1           | Cessna 208 Caravan | 6A      | Economy
 CN1           | Cessna 208 Caravan | 6B      | Economy
(12 rows)
```

Рисунок 10 — использование соединений

Рассмотрим внешние соединения, просмотрев какие самолеты на скольки маршрутах используются (рисунок 11). Внешнее соединение позволит увидеть неиспользуемый самолет.

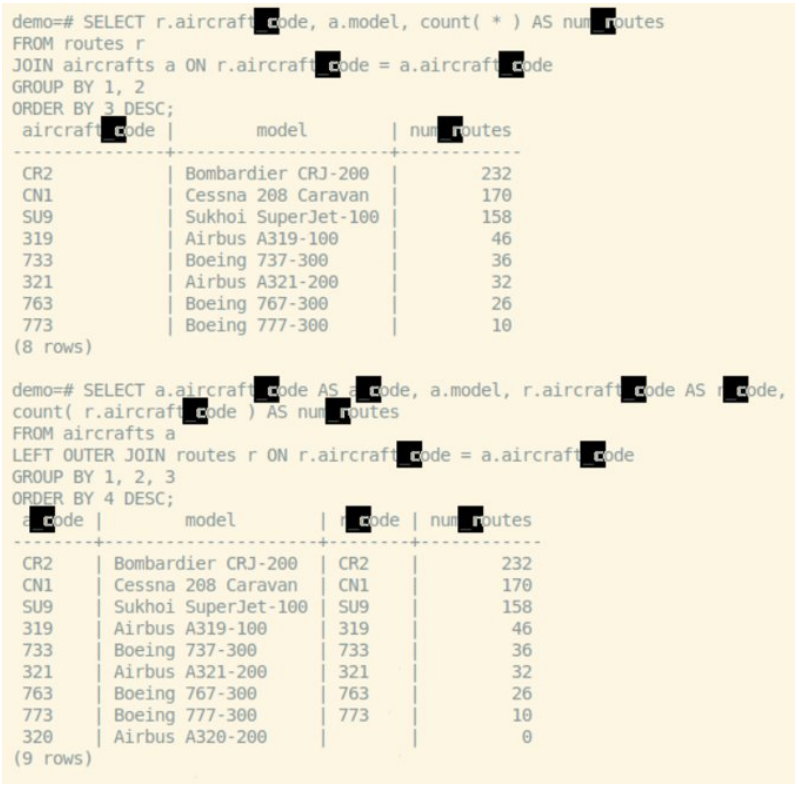


Рисунок 11 — запрос без и с использованием внешнего соединения

Рассмотрим многотабличный запрос на примере запроса, выводящего список пассажиров, не прошедших регистрацию на рейс (рисунок 12).

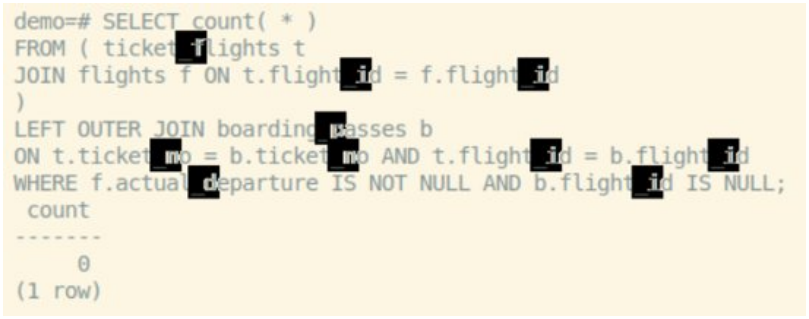


Рисунок 12 — список пассажиров, не прошедших регистрацию на рейс

Ознакомимся с оператором “UNION”, запросив в какие города можно полететь из Москвы или Санкт-Петербурга (рисунки 13-14).

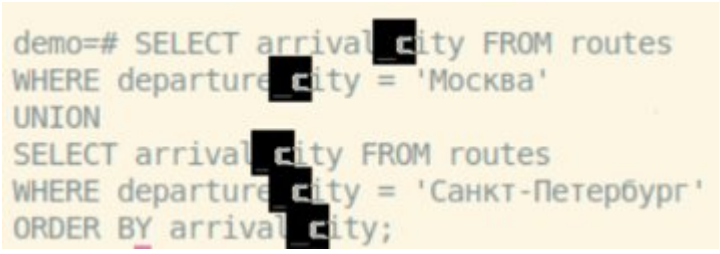
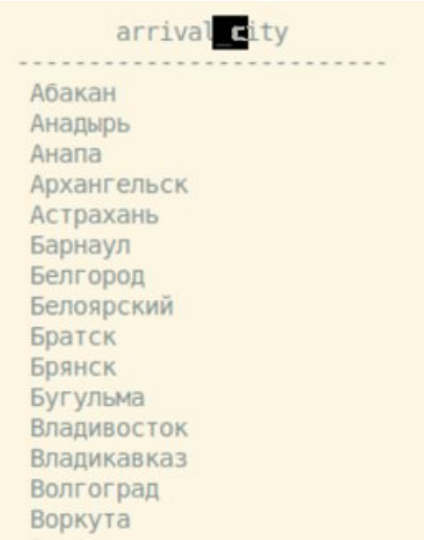


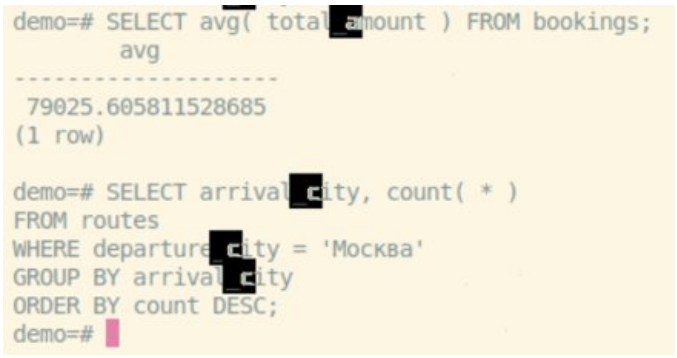
Рисунок 14 — запрос с оператором UNION



```
      arrival_city
--------------------------
 Абакан
 Анадырь
 Анапа
 Архангельск
 Астрахань
 Барнаул
 Белгород
 Белоярский
 Братск
 Брянск
 Бугульма
 Владивосток
 Владикавказ
 Волгоград
 Воркута
```

Рисунок 15 — результат выполнения запроса

Ознакомимся с агрегатными функциями, запросив информацию о средней стоимости билеов, а так же подсчитаем количество рейсов из Москвы в другие города (рисунки 16-17).



```
demo=# SELECT avg( total_amount ) FROM bookings;
        avg
--------------------
 79025.605811528685
(1 row)

demo=# SELECT arrival_city, count( * )
FROM routes
WHERE departure_city = 'Москва'
GROUP BY arrival_city
ORDER BY count DESC;
demo=#
```

Рисунок 16 — средняя стоимость билетов и запрос количества рейсов из Москвы

arrival_city	count
Санкт-Петербург	12
Брянск	9
Ульяновск	5
Йошкар-Ола	4
Петрозаводск	4
Ростов-на-Дону	3
Курган	3
Барнаул	3
Кемерово	3
Геленджик	3
Анадырь	3
Советский	3
Пенза	3
Сочи	2

Рисунок 17 — количество рейсов из Москвы

Так же, запросим обобщенную информацию о частоте рейсов; названия городов, из которых в другие города существует не менее 15 маршрутов (рисунок 18).

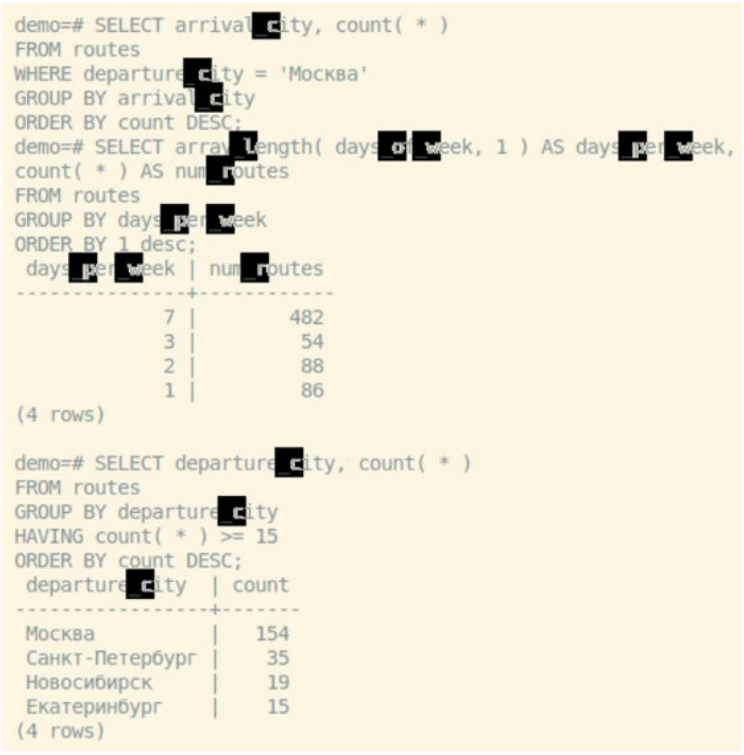


Рисунок 18 — информация о частоте рейсов и количестве маршрутов

Ознакомимся с оконными функциями, запросив накопительное число проданных билетов по месяцам (рисунки 19-20).

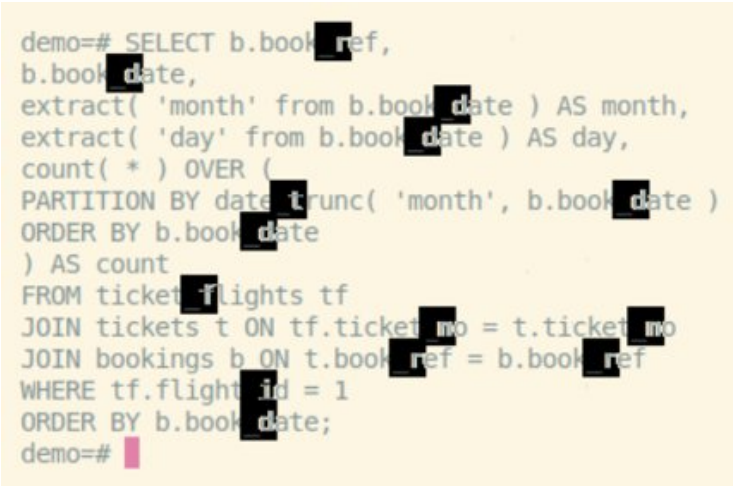


Рисунок 19 — запрос числа проданных билетов по месяцам

book_ref	book_date	month	day	count
A60039	2016-08-22 07:02:00+03	8	22	1
554340	2016-08-23 18:04:00+03	8	23	2
854C4C	2016-08-24 05:52:00+03	8	24	5
854C4C	2016-08-24 05:52:00+03	8	24	5
854C4C	2016-08-24 05:52:00+03	8	24	5
81D8AF	2016-08-25 05:22:00+03	8	25	6
D9BB20	2016-08-25 14:47:00+03	8	25	7
11B509	2016-08-25 17:58:00+03	8	25	8
1F26A0	2016-08-25 22:51:00+03	8	25	9
3F351E	2016-08-26 00:57:00+03	8	26	10
429B95	2016-08-26 07:19:00+03	8	26	11
AAAF88	2016-08-26 10:07:00+03	8	26	12
496D37	2016-08-26 19:15:00+03	8	26	14
496D37	2016-08-26 19:15:00+03	8	26	14
205824	2016-08-27 01:38:00+03	8	27	15
6BECE5	2016-08-27 06:39:00+03	8	27	16
606590	2016-08-27 07:19:00+03	8	27	18
606590	2016-08-27 07:19:00+03	8	27	18
BE3F54	2016-08-27 07:34:00+03	8	27	19
F1B913	2016-08-27 12:47:00+03	8	27	21
F1B913	2016-08-27 12:47:00+03	8	27	21
C25AAA	2016-08-27 18:50:00+03	8	27	23
C25AAA	2016-08-27 18:50:00+03	8	27	23

Рисунок 20 — число проданных билетов по месяцам

Рассмотрим простейший подзапрос, подсчитав суммы бронирования, которые больше средней (рисунок 21).

```
demo=# SELECT count( * ) FROM bookings
WHERE total_amount >
( SELECT avg( total_amount ) FROM bookings );
 count
-------
 87224
(1 row)
```

Рисунок 21 — простейший подзапрос

Рассмотрим так же подзапрос, для обнаружения соединенных маршрутом городов в часовом поясе Красноярска (рисунок 22) и обнаружения городов, куда нет маршрутов из Москвы (рисунок 23).

```
demo=# SELECT flight_no, departure_city, arrival_city
FROM routes
WHERE departure_city IN (
SELECT city
FROM airports
WHERE timezone ~ 'Krasnoyarsk'
)
AND arrival_city IN (
SELECT city
FROM airports
WHERE timezone ~ 'Krasnoyarsk'
);
 flight_no | departure_city | arrival_city
-----------+----------------+--------------
 PG0070    | Абакан         | Томск
 PG0071    | Томск          | Абакан
 PG0313    | Абакан         | Кызыл
 PG0314    | Кызыл          | Абакан
 PG0653    | Красноярск     | Барнаул
 PG0654    | Барнаул        | Красноярск
(6 rows)
```

Рисунок 22 — города маршрутов в часовом поясе Красноярска

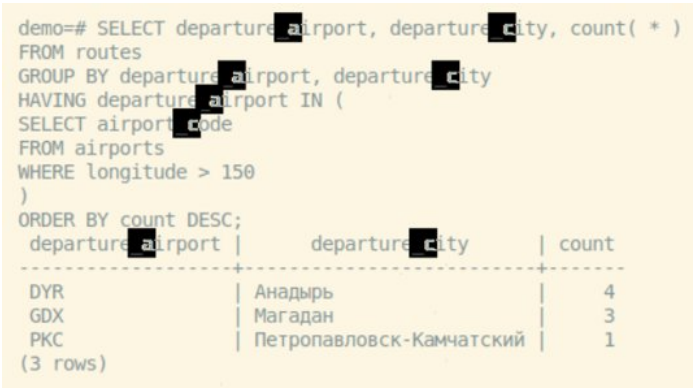
```
demo=# SELECT DISTINCT a.city
FROM airports a
WHERE NOT EXISTS (
SELECT * FROM routes r
WHERE r.departure_city = 'Москва'
AND r.arrival_city = a.city
)
AND a.city <> 'Москва'
ORDER BY city;
         city
----------------------
 Благовещенск
 Иваново
 Иркутск
 Калуга
 Когалым
 Комсомольск-на-Амуре
 Кызыл
 Магадан
 Нижнекамск
 Новокузнецк
 Стрежевой
 Сургут
 Удачный
 Усть-Илимск
 Усть-Кут
 Ухта
 Череповец
 Чита
 Якутск
 Ярославль
(20 rows)
```

Рисунок 23 — города, не соединенные маршрутом с Москвой

Кроме того, выведем список аэропортов из городов с несколькими аэропортами (рисунок 24) и определим число маршрутов, исходящих из тех аэропортов, которые расположены восточнее географической долготы 150◦ (рисунок 25).

```
demo=# SELECT aa.city, aa.airport_code, aa.airport_name
FROM (
SELECT city, count( * )
FROM airports
GROUP BY city
HAVING count( * ) > 1
) AS a
JOIN airports AS aa ON a.city = aa.city
ORDER BY aa.city, aa.airport_name;
   city    | airport_code |    airport_name
-----------+--------------+---------------------
 Москва    | VKO          | Внуково
 Москва    | DME          | Домодедово
 Москва    | SVO          | Шереметьево
 Ульяновск | ULV          | Баратаевка
 Ульяновск | ULY          | Ульяновск-Восточный
(5 rows)
```

Рисунок 24 — список аэропортов из городов с несколькими аэропортами



```
demo=# SELECT departure_airport, departure_city, count( * )
FROM routes
GROUP BY departure_airport, departure_city
HAVING departure_airport IN (
SELECT airport_code
FROM airports
WHERE longitude > 150
)
ORDER BY count DESC;
 departure_airport |      departure_city      | count
-------------------+--------------------------+-------
 DYR               | Анадырь                  |     4
 GDX               | Магадан                  |     3
 PKC               | Петропавловск-Камчатский |     1
(3 rows)
```

Рисунок 25 — число маршрутов из аэропортов, которые расположены восточнее географической долготы 150◦

Используем вложенные подзапросы, чтобы определить степень заполнения самолетов (листинг 1, рисунок 26).

Листинг 1 — запрос для вычисления степени заполнения самолетов

```
SELECT ts.flight_id,
ts.flight_no,
ts.scheduled_departure_local,
ts.departure_city,
ts.arrival_city,
a.model,
ts.fact_passengers,
ts.total_seats,
round( ts.fact_passengers::numeric /
ts.total_seats::numeric, 2 ) AS fraction
FROM (
SELECT f.flight_id,
f.flight_no,
f.scheduled_departure_local,
f.departure_city,
f.arrival_city,
f.aircraft_code,
count( tf.ticket_no ) AS fact_passengers,
( SELECT count( s.seat_no )
FROM seats s
WHERE s.aircraft_code = f.aircraft_code
) AS total_seats
FROM flights_v f
JOIN ticket_flights tf ON f.flight_id = tf.flight_id
WHERE f.status = 'Arrived'
GROUP BY 1, 2, 3, 4, 5, 6
) AS ts
JOIN aircrafts AS a ON ts.aircraft_code = a.aircraft_code
ORDER BY ts.scheduled_departure_local;
```

flight_id	flight_no	scheduled_departure_local	departure_city	arrival_city	model	fact_passengers	total_seats	fraction
28205	PG0032	2016-09-13 08:00:00	Пенза	Москва	Cessna 208 Caravan	2	12	0.17
9467	PG0360	2016-09-13 08:00:00	Санкт-Петербург	Оренбург	Bombardier CRJ-200	6	50	0.12
7130	PG0591	2016-09-13 08:00:00	Москва	Томск	Sukhoi SuperJet-100	25	97	0.26
6223	PG0120	2016-09-13 08:00:00	Москва	Мирный	Boeing 737-300	34	130	0.26
1764	PG0239	2016-09-13 08:05:00	Москва	Ханты-Мансийск	Sukhoi SuperJet-100	55	97	0.57
32948	PG0550	2016-09-13 08:05:00	Владикавказ	Москва	Bombardier CRJ-200	11	50	0.22
3625	PG0414	2016-09-13 08:05:00	Москва	Мурманск	Bombardier CRJ-200	13	50	0.26
9307	PG0198	2016-09-13 08:10:00	Санкт-Петербург	Иркутск	Airbus A321-200	19	170	0.11
15025	PG0683	2016-09-13 08:15:00	Пермь	Ульяновск	Sukhoi SuperJet-100	32	97	0.33
21969	PG0240	2016-09-13 08:15:00	Ханты-Мансийск	Москва	Sukhoi SuperJet-100	5	97	0.05
1452	PG0509	2016-09-13 08:15:00	Москва	Элиста	Sukhoi SuperJet-100	20	97	0.21
2485	PG0482	2016-09-13 08:20:00	Москва	Кемерово	Sukhoi SuperJet-100	15	97	0.15
25029	PG0044	2016-09-13 08:20:00	Ижевск	Москва	Bombardier CRJ-200	4	50	0.08
15750	PG0244	2016-09-13 08:20:00	Якутск	Санкт-Петербург	Airbus A319-100	5	116	0.04
17534	PG0123	2016-09-13 08:25:00	Нижнекамск	Ростов-на-Дону	Bombardier CRJ-200	2	50	0.04
28883	PG0177	2016-09-13 08:25:00	Чебоксары	Москва	Bombardier CRJ-200	8	50	0.16
24836	PG0014	2016-09-13 08:30:00	Тюмень	Урай	Sukhoi SuperJet-100	41	97	0.42
16590	PG0369	2016-09-13 08:30:00	Нижневартовск	Москва	Bombardier CRJ-200	14	50	0.28

Рисунок 26 — степень заполнения самолетов

Рассмотренный сложный запрос можно сделать более наглядным за счет выделения подзапроса в отдельную конструкцию, которая называется общее табличное выражение (Common Table Expression — CTE).

Листинг 2 — запрос для вычисления степени заполнения самолетов с использованием Common Table Expression

```
WITH ts AS
( SELECT f.flight_id,
f.flight_no,
f.scheduled_departure_local,
f.departure_city,
f.arrival_city,
f.aircraft_code,
count( tf.ticket_no ) AS fact_passengers,
( SELECT count( s.seat_no )
FROM seats s
WHERE s.aircraft_code = f.aircraft_code
) AS total_seats
FROM flights_v f
JOIN ticket_flights tf ON f.flight_id = tf.flight_id
WHERE f.status = 'Arrived'
GROUP BY 1, 2, 3, 4, 5, 6
)
SELECT ts.flight_id,
ts.flight_no,
ts.scheduled_departure_local,
ts.departure_city,
ts.arrival_city,
a.model,
ts.fact_passengers,
ts.total_seats,
round( ts.fact_passengers::numeric /
ts.total_seats::numeric, 2 ) AS fraction
FROM ts
JOIN aircrafts AS a ON ts.aircraft_code = a.aircraft_code
ORDER BY ts.scheduled_departure_local;
```

4. Контрольные вопросы и задания

1. В документации сказано, что служебный символ «%» в шаблоне оператора LIKE соответствует любой последовательности символов, в том числе и пустой последовательности, однако ничего не сказано насчет правил обработки пробелов. В таблице «Билеты» (tickets) столбец passenger_name содержит имя и фамилию пассажира, записанные заглавными латинскими буквами и разделенные одним пробелом. Выясните правила обработки пробелов самостоятельно, выполнив следующие команды и сравнив полученные результаты:

*SELECT count(*) FROM tickets;*

*SELECT count(*) FROM tickets WHERE passenger_name LIKE '% %';*

*SELECT count(*) FROM tickets WHERE passenger_name LIKE '% % %';*

*SELECT count(*) FROM tickets WHERE passenger_name LIKE '% %%';*

Результаты выполнения запросов представлены на рисунке 27.

Рисунок 27 — исследование правил обработки пробелов

Из полученных результатов можно сделать вывод, что пробел в операторе LIKE обрабатывается аналогично обыкновенному буквенному или цифровому символу.

2. Этот запрос выбирает из таблицы «Билеты» (tickets) всех пассажиров с именами, состоящими из трех букв (в шаблоне присутствуют три символа «_»):

SELECT passenger_name

FROM tickets

WHERE passenger_name LIKE '___ %';

Предложите шаблон поиска в операторе LIKE для выбора из этой таблицы всех пассажиров с фамилиями, состоящими из пяти букв.

Аналогично представленному шаблону, для выполнения задания необходимо заменить имя символом “%” и разделить его пробелом с пятью “_”, обозначающими символы фамилии (рисунок 28).

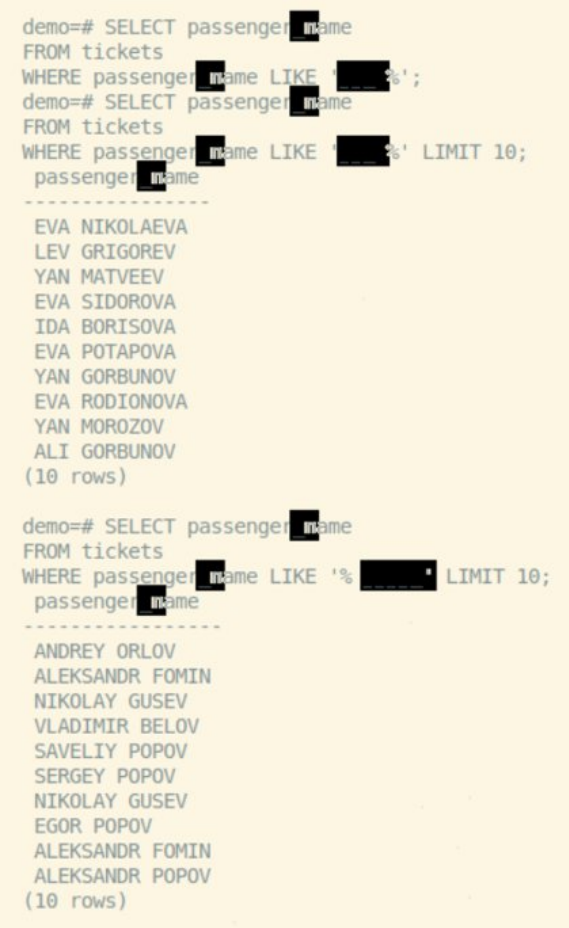


Рисунок 28 — фильтрация пассажиров по длине имени и по длине фамилии

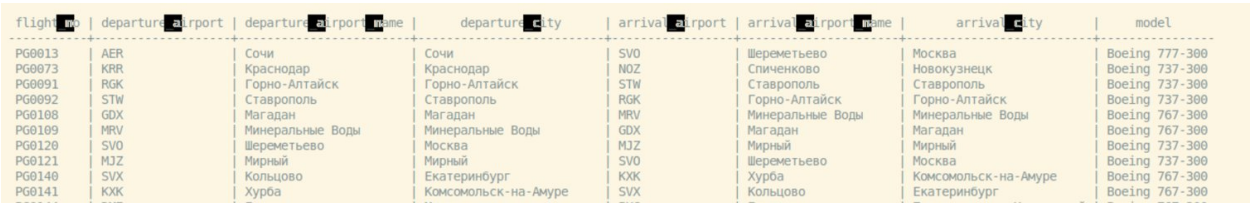
4. Выясните, на каких маршрутах используются самолеты компании Boeing. В выборке вместо кода модели должно выводиться ее наименование, например, вместо кода 733 должно быть Boeing 737-300.

Указание: можно воспользоваться соединением представления «Маршруты» (routes) и таблицы «Самолеты» (aircrafts).

Запрос, возвращающий требуемый результат представлен в листинге 3, результат — на рисунке 29.

Листинг 3 — запрос маршрутов самолетов Boeing

```
SELECT flight_no,
departure_airport,
departure_airport_name,
departure_city,
arrival_airport,
arrival_airport_name,
arrival_city,
model
FROM routes as r
JOIN aircrafts as a
ON a.aircraft_code = r.aircraft_code
WHERE model LIKE '%Boeing%';
```



flight_no	departure_airport	departure_airport_name	departure_city	arrival_airport	arrival_airport_name	arrival_city	model
PG0013	AER	Сочи	Сочи	SVO	Шереметьево	Москва	Boeing 777-300
PG0073	KRR	Краснодар	Краснодар	NOZ	Спиченково	Новокузнецк	Boeing 737-300
PG0091	RGK	Горно-Алтайск	Горно-Алтайск	STW	Ставрополь	Ставрополь	Boeing 737-300
PG0092	STW	Ставрополь	Ставрополь	RGK	Горно-Алтайск	Горно-Алтайск	Boeing 737-300
PG0108	GDX	Магадан	Магадан	MRV	Минеральные Воды	Минеральные Воды	Boeing 767-300
PG0109	MRV	Минеральные Воды	Минеральные Воды	GDX	Магадан	Магадан	Boeing 767-300
PG0120	SVO	Шереметьево	Москва	MJZ	Мирный	Мирный	Boeing 737-300
PG0121	MJZ	Мирный	Мирный	SVO	Шереметьево	Москва	Boeing 737-300
PG0140	SVX	Кольцово	Екатеринбург	KXK	Хурба	Комсомольск-на-Амуре	Boeing 767-300
PG0141	KXK	Хурба	Комсомольск-на-Амуре	SVX	Кольцово	Екатеринбург	Boeing 767-300

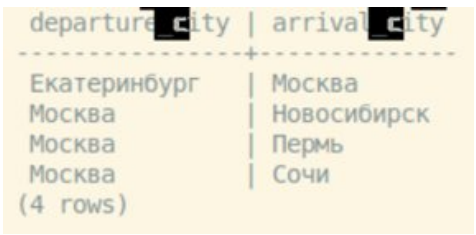
Рисунок 29 — результат выполнения запроса

4. Самые крупные самолеты в нашей авиакомпании — это Boeing 777-300. Выяснить, между какими парами городов они летают. Каждая пара городов была выведена только один раз.

Запрос, возвращающий требуемый результат представлен в листинге 4, результат — на рисунке 30.

Листинг 4 — города на маршрутах Boeing 777

```
SELECT DISTINCT r.departure_city, r.arrival_city
FROM routes AS r, aircrafts AS a
WHERE a.aircraft_code = r.aircraft_code AND a.model = 'Boeing 777-300' AND
r.arrival_city > r.departure_city;
```



departure_city	arrival_city
Екатеринбург	Москва
Москва	Новосибирск
Москва	Пермь
Москва	Сочи

(4 rows)

Рисунок 30 — результат выполнения запроса

5. Выяснить, сколько различных рейсов выполняется из каждого города, без учета частоты рейсов в неделю, можно с помощью обращения к представлению routes (маршруты).

Запрос, возвращающий требуемый результат представлен в листинге 5, результат — на рисунке 31.

Листинг 5 — число рейсов из различных городов

```
SELECT departure_city, count( * )
FROM routes
GROUP BY departure_city
ORDER BY count DESC;
```

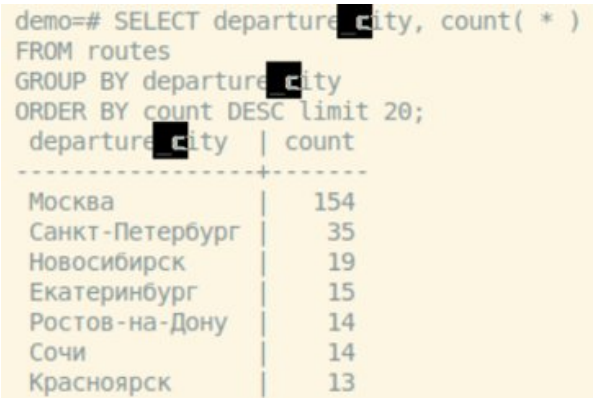


Рисунок 31 — результат выполнения запроса

6. Модифицируйте этот запрос так, чтобы он выводил число направлений, по которым летают самолеты из каждого города. Например, из Москвы в СанктПетербург летает несколько различных рейсов, но все эти рейсы относятся к одному направлению.

Запрос, возвращающий требуемый результат представлен в листинге 6, результат — на рисунке 32.

Листинг 6 — число направлений рейсов из различных городов

```
SELECT departure_city, count(distinct arrival_city)
FROM routes
GROUP BY departure_city
ORDER BY count DESC;
```

```
demo=# SELECT departure_city, count(distinct arrival_city
FROM routes
GROUP BY departure_city
ORDER BY count DESC limit 20;
 departure_city  | count
-----------------+-------
 Москва          |    80
 Санкт-Петербург |    22
 Новосибирск     |    14
 Екатеринбург    |    13
 Сочи            |    11
 Ростов-на-Дону  |    10
 Красноярск      |    10
```

Рисунок 32 — результат выполнения запроса

7. В материализованном представлении «Маршруты» (routes) имеется столбец days_of_week, который содержит списки (массивы) номеров дней недели, когда выполняется каждый рейс. Для оптимизации расписания вылетов из Москвы нужно выявить пять городов, в которые из столицы отправляется наибольшее число ежедневных рейсов (маршрутов). Строки в выборке следует расположить в убывающем порядке числа выполняемых рейсов.

Запрос, возвращающий требуемый результат представлен в листинге 7, результат — на рисунке 33.

Листинг 7 — пять наиболее связанных с Москвой городов

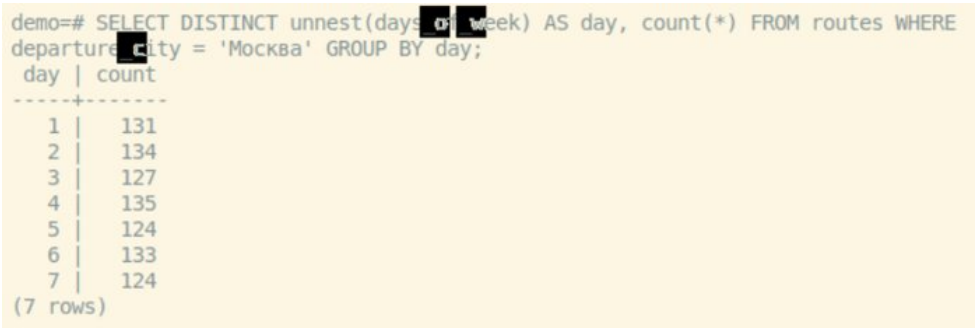
```
SELECT arrival_city, count(*) AS count FROM routes WHERE
array_length(days_of_week, 1) = 7 AND departure_city = 'Москва' GROUP BY
arrival_city ORDER BY count DESC;
```

```
-----------------+-------
 Санкт-Петербург |    12
 Брянск          |     9
 Ульяновск       |     5
 Йошкар-Ола      |     4
 Петрозаводск    |     4
```

Рисунок 33 — результат выполнения запроса

8. Предположим, что служба материального снабжения нашей авиакомпании запросила информацию о числе рейсов, выполняющихся из Москвы в каждый день недели.

Соответствующий запрос и результат его выполнения представлены на рисунке 34.



```
demo=# SELECT DISTINCT unnest(days_of_week) AS day, count(*) FROM routes WHERE
departure_city = 'Москва' GROUP BY day;
 day | count
-----+-------
   1 |   131
   2 |   134
   3 |   127
   4 |   135
   5 |   124
   6 |   133
   7 |   124
(7 rows)
```

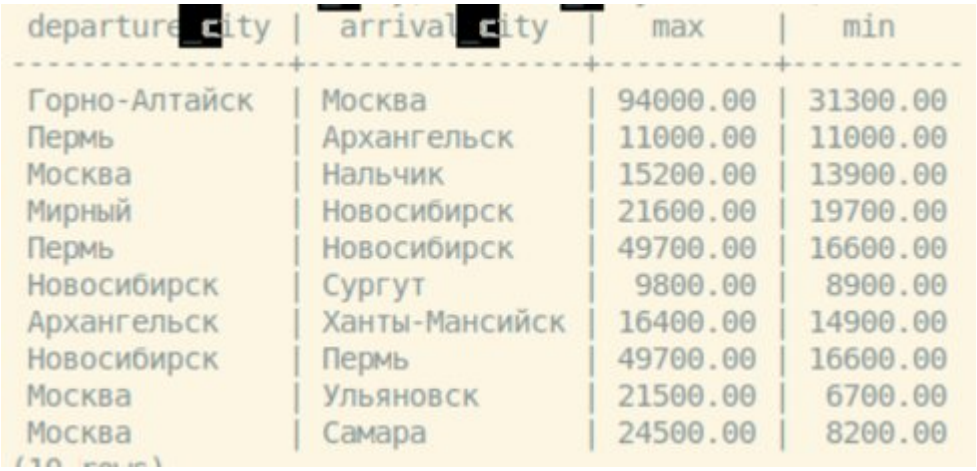
Рисунок 34 — число рейсов из Москвы по дням

13. Каковы максимальные и минимальные цены билетов на все направления. Оператор SELECT должен возвращать departure_city, arrival_city, max(amount), min(amount).

Запрос, возвращающий требуемый результат представлен в листинге 8, результат — на рисунке 35.

Листинг 8 — максимальные и минимальные цены билетов на все направления

```
SELECT departure_city, arrival_city
FROM flights_v as f
JOIN ticket_flights as t
ON f.flight_id = t.flight_id;
SELECT departure_city, arrival_city, max(amount), min(amount)
FROM flights_v as f
JOIN ticket_flights as t
ON f.flight_id = t.flight_id
GROUP BY departure_city, arrival_city;
```



```
 departure_city  |  arrival_city   |   max    |   min
-----------------+-----------------+----------+----------
 Горно-Алтайск   | Москва          | 94000.00 | 31300.00
 Пермь           | Архангельск     | 11000.00 | 11000.00
 Москва          | Нальчик         | 15200.00 | 13900.00
 Мирный          | Новосибирск     | 21600.00 | 19700.00
 Пермь           | Новосибирск     | 49700.00 | 16600.00
 Новосибирск     | Сургут          |  9800.00 |  8900.00
 Архангельск     | Ханты-Мансийск  | 16400.00 | 14900.00
 Новосибирск     | Пермь           | 49700.00 | 16600.00
 Москва          | Ульяновск       | 21500.00 |  6700.00
 Москва          | Самара          | 24500.00 |  8200.00
(10 rows)
```

Рисунок 35 — результат выполнения запроса

Вывод: По итогам выполнения данной лабораторной работы были изучены различные подходы к разработке сложных запросов SELECT. Были рассмотрены разнообразные примеры запросов с широким спектром операторов, предикатов и функций. После этого были проработаны практические задачи, связанные с разработкой запросов.