Федеральное государственное автономное

образовательное учреждение

Высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| институт |
| Информатика |
| кафедра |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ**

|  |
| --- |
| Практическое занятие 4. Запросы |
| тема |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | Е. П. Моргунов |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-17/2Б, 032156940 |  |  |  | Н. А. Самарин |
|  | номер группы, зачётной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2023

**СОДЕРЖАНИЕ**

1 Задание............................................................................................................... 3

2 Ход работы........................................................................................................ 3

**1 Задание**

- Изучить материал главы 6 учебного пособия: Моргунов, Е. П.  
PostgreSQL. Основы языка SQL [Текст] : учеб. пособие / Е. П. Моргунов ; под  
ред. Е. В. Рогова, П. В. Лузанова. – СПб. : БХВ-Петербург, 2018. – 336 с. : ил. –  
ISBN 978-5-9775-4022-3. Можно дополнительно использовать презентации и  
видеоуроки: Моргунов, Е. П. Язык SQL [Электронный ресурс] : практический  
курс. – М. : Постгрес Профессиональный, 2019.;

- Выполнить задания, приведенные в разделе "Контрольные вопросы и  
задания".;

- Подготовить отчет. Включить в него тексты заданий из пособия. Для  
каждого задания сделать снимки экрана с введенными командами языка SQL и  
результатами их выполнения в среде утилиты psql..

**2 Ход работы**

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 '% %%';

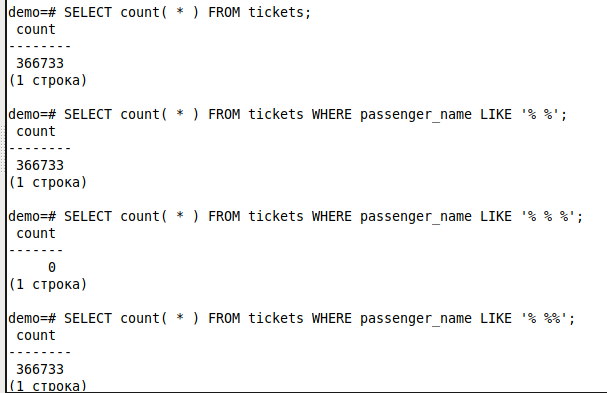


Рисунок 1 – Задание 1

3. В разделе документации 9.7.2 «Регулярные выражения SIMILAR TO»  
рассматривается оператор SIMILAR TO. Он работает аналогично оператору

LIKE, но использует шаблоны, соответствующие определению регулярных  
выражений, приведенному в стандарте SQL. Регулярные выражения SQL  
представляют собой комбинацию синтаксиса LIKE с синтаксисом обычных  
регулярных выражений. Самостоятельно ознакомьтесь с оператором SIMILAR  
TO.

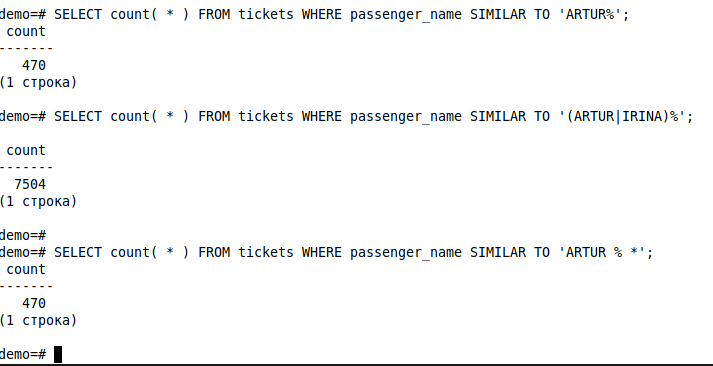


Рисунок 2 – Задание 3

5. В разделе документации 9.17 «Условные выражения» представлены  
условные выражения, которые поддерживаются в PostgreSQL. В тексте главы  
была рассмотрена конструкция CASE. Самостоятельно ознакомьтесь с  
функциями COALESCE, NULLIF, GREATEST и LEAST.

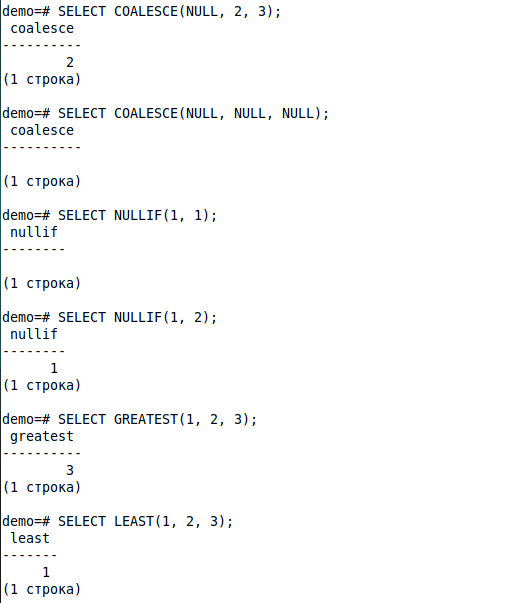


Рисунок 3 – Задание 5

7. Самые крупные самолеты в нашей авиакомпании — это Boeing  
777-300. Выяснить, между какими парами городов они летают, поможет запрос:  
SELECT DISTINCT departure\_city, arrival\_city FROM routes r JOIN aircrafts a ON  
r.aircraft\_code = a.aircraft\_code WHERE a.model = 'Boeing 777-300' ORDER BY 1;  
К сожалению, в этой выборке информация дублируется. Пары городов  
приведены по два раза: для рейса «туда» и для рейса «обратно».  
Модифицируйте запрос таким образом, чтобы каждая пара городов была  
выведена только один раз.

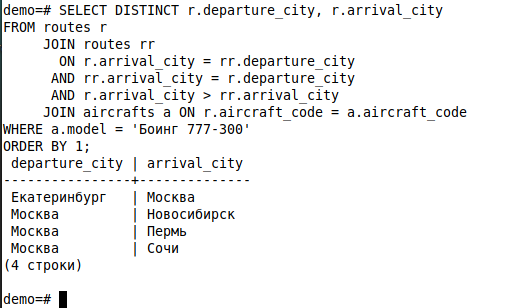


Рисунок 4 – Задание 7

9. Для ответа на вопрос, сколько рейсов выполняется из Москвы в  
Санкт-Петербург, можно написать совсем простой запрос: SELECT count( \* )  
FROM routes WHERE departure\_city = 'Москва' AND arrival\_city =  
'Санкт-Петербург'; А с помощью какого запроса можно получить результат в  
таком виде?

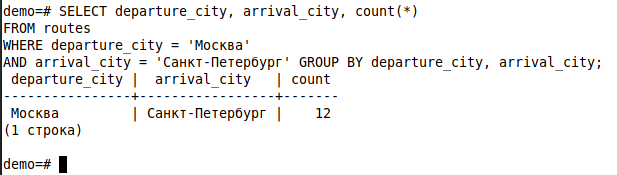


Рисунок 5 – Задание 9

11. 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. Указание:  
воспользуйтесь функцией array\_length.

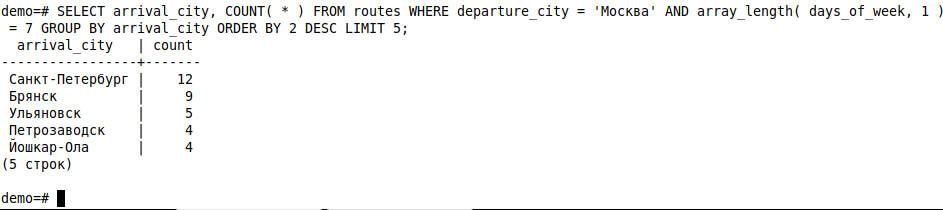


Рисунок 6 – Задание 11

13. Ответить на вопрос о том, каковы максимальные и минимальные

цены билетов на все направления, может такой запрос: SELECT f.departure\_city,  
f.arrival\_city, max( tf.amount ), min( tf.amount ) FROM flights\_v f JOIN  
ticket\_flights tf ON f.flight\_id = tf.flight\_id GROUP BY 1, 2 ORDER BY 1, 2; А  
как выявить те направления, на которые не было продано ни одного билета?  
Один из вариантов решения такой: если на рейсы, отправляющиеся по  
какомуто направлению, не было продано ни одного билета, то максимальная и  
минимальная цены будут равны NULL. Модифицируйте запрос, приведенный  
выше.

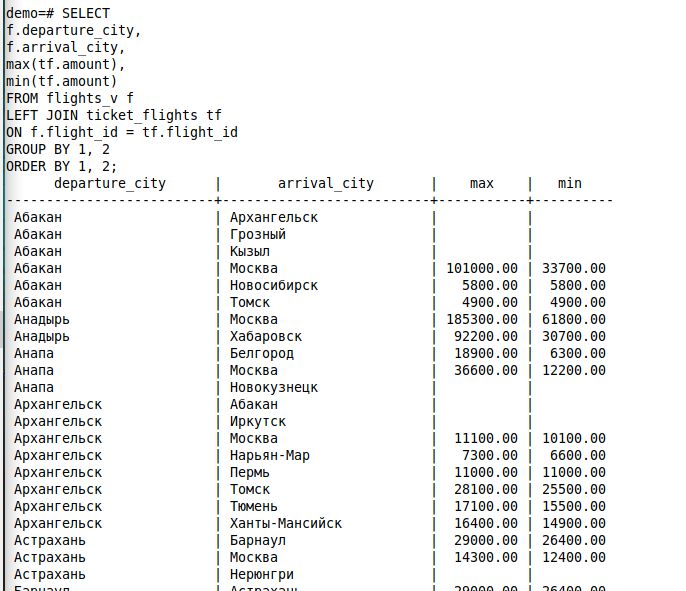


Рисунок 7 – Задание 13

15.\* В тексте главы были кратко рассмотрены оконные функции.  
Самостоятельно прочитайте разделы документации, которые рекомендуется  
изучить для более детального ознакомления с этим классом функций.  
Подумайте, в какой ситуации, связанной с базой данных «Авиаперевозки»,  
было бы полезно применить оконные функции, и напишите запрос.

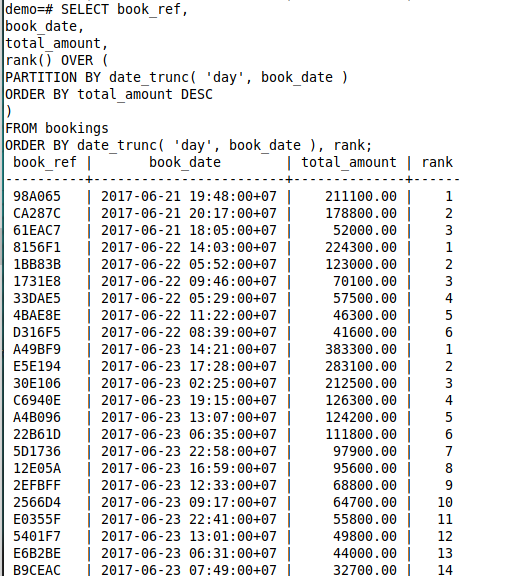


Рисунок 8 – Задание 15

17. В тексте главы в разделе 6.4 мы рассмотрели два способа получения  
ответа на вопрос: как распределяются места с разными классами обслуживания  
в самолетах всех типов? А с помощью какого запроса можно получить  
результат в таком виде? aircraft\_code | model | fare\_conditions | count

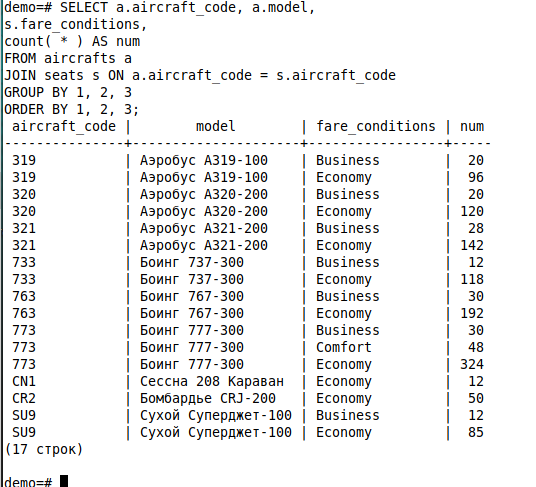


Рисунок 9 – Задание 17

19.\* В разделе 6.4 мы использовали рекурсивный алгоритм в общем  
табличном выражении. Изучите этот пример, чтобы лучше понять работу  
рекурсивного алгоритма: WITH RECURSIVE ranges ( min\_sum, max\_sum ) AS (  
VALUES( 0, 100000 ), ( 100000, 200000 ), ( 200000, 300000 ) UNION ALL  
SELECT min\_sum + 100000, max\_sum + 100000 FROM ranges WHERE max\_sum  
< ( SELECT max( total\_amount ) FROM bookings ) ) SELECT \* FROM ranges;  
Задание 1. Модифицируйте запрос, добавив в него столбец level (можно назвать  
его и iteration). Этот столбец должен содержать номер текущей итерации,  
поэтому нужно увеличивать его значение на единицу на каждом шаге. Не  
забудьте задать начальное значение для добавленного столбца в предложении  
VALUES. Задание 2. Для завершения экспериментов замените UNION ALL на  
UNION и выполните запрос. Сравните этот результат с предыдущим, когда мы  
использовали UNION ALL.

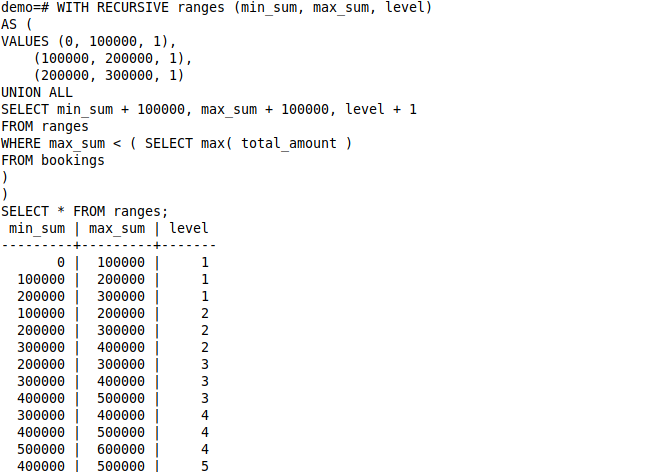


Рисунок 10 – Задание 19 часть 2

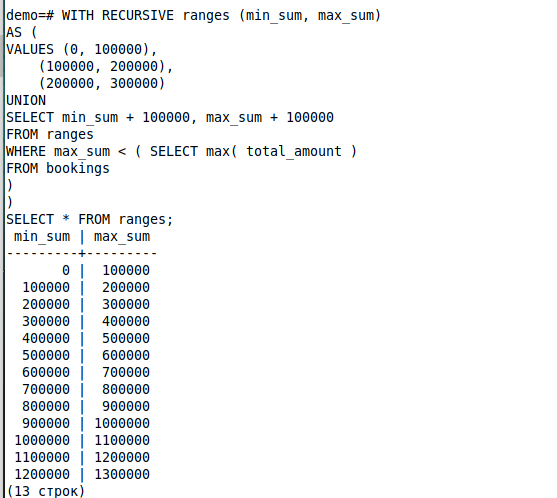


Рисунок 11 – Задание 19 часть 2

21. В тексте главы был приведен запрос, выводящий список городов, в  
которые нет рейсов из Москвы. 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;  
Можно предложить другой вариант, в котором используется одна из операций  
над множествами строк: объединение, пересечение или разность. Вместо знака  
«?» поставьте в приведенном ниже запросе нужное ключевое слово — UNION,  
INTERSECT или EXCEPT — и обоснуйте ваше решение. SELECT city FROM

airports WHERE city <> 'Москва' ? SELECT arrival\_city FROM routes WHERE  
departure\_city = 'Москва' ORDER BY city;

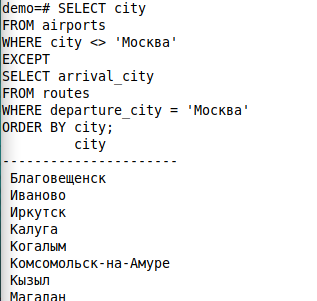


Рисунок 12 – Задание 21

23. Предположим, что департамент развития нашей авиакомпании  
задался вопросом: каким будет общее число различных маршрутов, которые  
теоретически можно проложить между всеми городами? Если в каком-то  
городе имеется более одного аэропорта, то это учитывать не будем, т. е.  
маршрутом будем считать путь между городами, а не между аэропортами.  
Здесь мы используем соединение таблицы с самой собой на основе неравенства  
значений атрибутов. SELECT count( \* ) FROM ( SELECT DISTINCT city FROM  
airports ) AS a1 JOIN ( SELECT DISTINCT city FROM airports ) AS a2 ON a1.city  
<> a2.city; Задание. Перепишите этот запрос с общим табличным выражением.

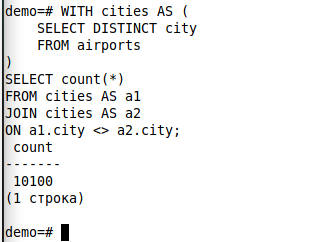


Рисунок 13 – Задание 23

25.\* При планировании новых маршрутов и оценке экономической  
эффективности уже существующих может потребоваться информация о том,  
какова усредненная степень заполнения самолетов на всех направлениях. Будем  
учитывать только уже прибывшие рейсы. WITH tickets\_seats AS ( SELECT  
f.flight\_id, f.flight\_no, 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 ) SELECT ts.departure\_city, ts.arrival\_city, sum( ts.fact\_passengers ) AS  
sum\_pass, sum( ts.total\_seats ) AS sum\_seats, round( sum( ts.fact\_passengers  
)::numeric / sum( ts.total\_seats )::numeric, 2 ) AS frac FROM tickets\_seats ts  
GROUP BY ts.departure\_city, ts.arrival\_city ORDER BY ts.departure\_city; Для  
того чтобы лучше уяснить, как работает запрос в целом, вычлените из него  
отдельные подзапросы и выполните их, посмотрите, что они выводят. Задание.  
Модифицируйте этот запрос, чтобы он выводил те же отчетные данные, но с  
учетом классов обслуживания, т. е. Business, Comfort и Economy.

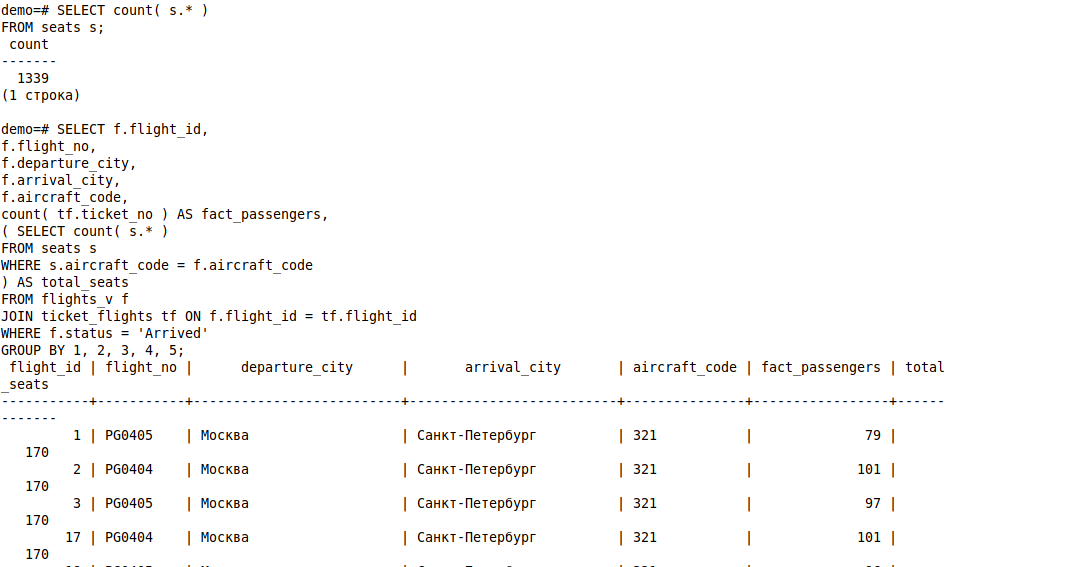


Рисунок 14 – Задание 25 часть 1

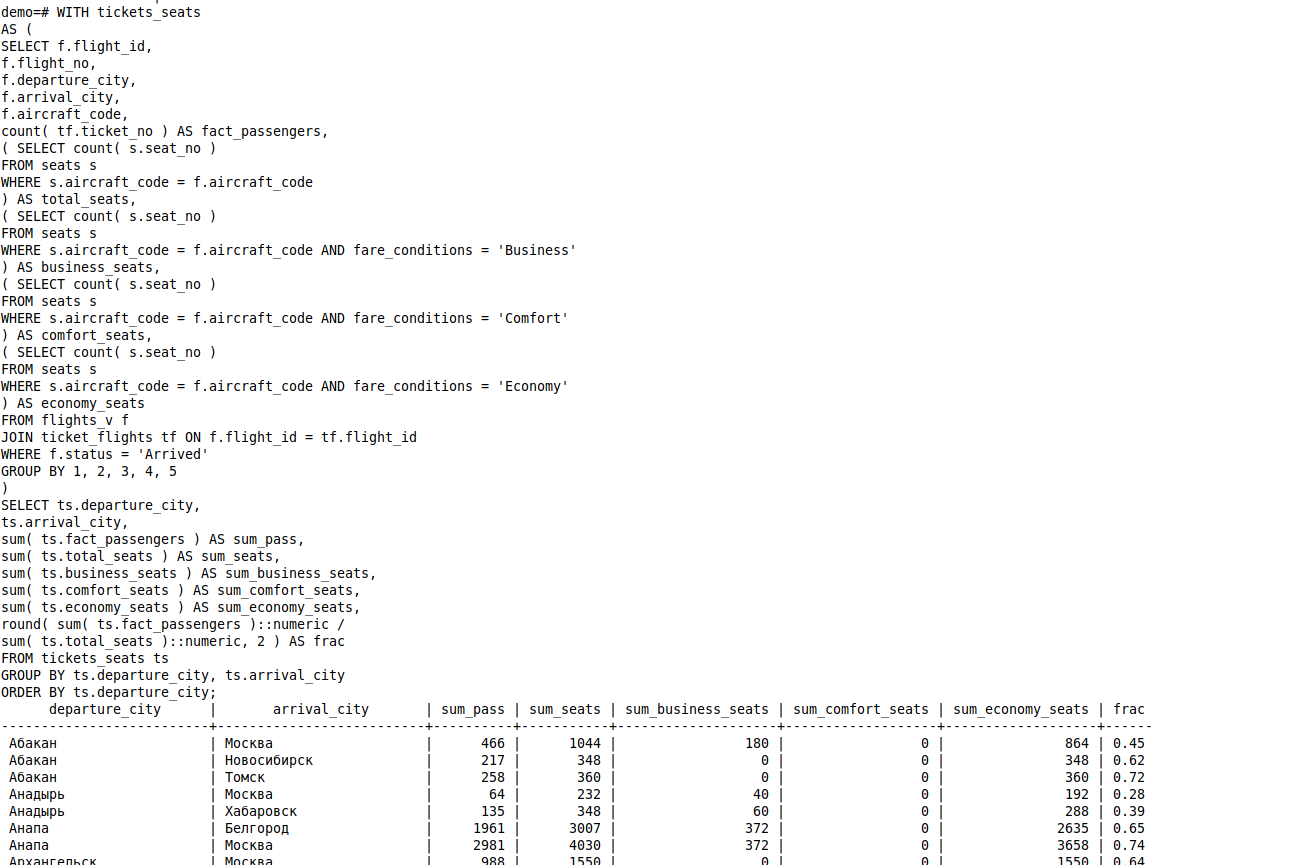


Рисунок 15 – Задание 25 часть 2