1) Какими самолетами, кроме машин компаний Airbus и Boeing, располагает наша авиакомпания (aircrafts)?

select \* from aircrafts where model not like 'Air%' and model not like 'Boe%';

2) В каких различных часовых поясах(timezone) располагаются аэропорты(airports)?

select distinct timezone from airports;

3) Найти три самых восточных аэропорта(longitude).

select airport\_name, longitude from airports order by longitude desc limit 3;

4) Выбрать все места (seat\_no) и их класс (fare\_conditions), предусмотренные компоновкой салона самолета (model) Cessna 208 Caravan (таблицы seats и aircrafts). Результат отсортировать по номеру места.

select aircrafts.model, seats.seat\_no, seats.fare\_conditions from aircrafts join seats on aircrafts.aircraft\_code = seats.aircraft\_code where aircrafts.model like 'Cessn%' order by seat\_no;

5) Написать запрос тремя способами:  Сколько всего маршрутов нужно было бы сформировать, если бы требовалось соединить каждый город со всеми остальными городами(city из airports).

select (count(distinct city)\*(count(distinct city)-1)) as ways from airports; ещё 2 не придумал

6) Вывести сколько маршрутов(aircraft\_code) обслуживают самолеты каждого типа (model)? Таблицы routes и aircrafts.

select aircrafts.model, count(flights.flight\_id) from aircrafts join flights on aircrafts.aircraft\_code = flights.aircraft\_code group by aircrafts.model;

7) Какая модель самолета не участвует в выполнении рейсов?

select aircrafts.model from aircrafts except all select aircrafts.model from aircrafts join flights on aircrafts.aircraft\_code = flights.aircraft\_code group by aircrafts.model;

8) Определить, сколько существует маршрутов из каждого города(departure\_city) в другие города, и вывести названия городов, из которых в другие города существует не менее 15 маршрутов. Таблица routes.

select count(flight\_no), departure\_city from routes group by departure\_city having count(flight\_no) >= 15;

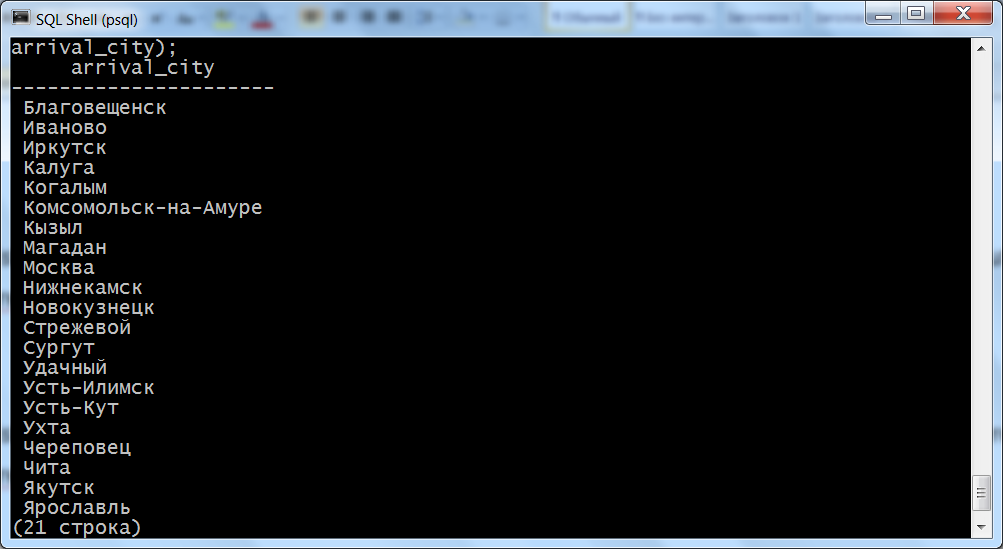
9) Вывести города, в которых более 1 аэропорта, не используя подзапросы.

select count(airport\_code), city from airports group by city having count (airport\_code) > 1;

10) Написать запрос с использованием **EXISTS** и **IN**. Выведите в какие города нет рейсов из Москвы. (2 отдельных запроса)

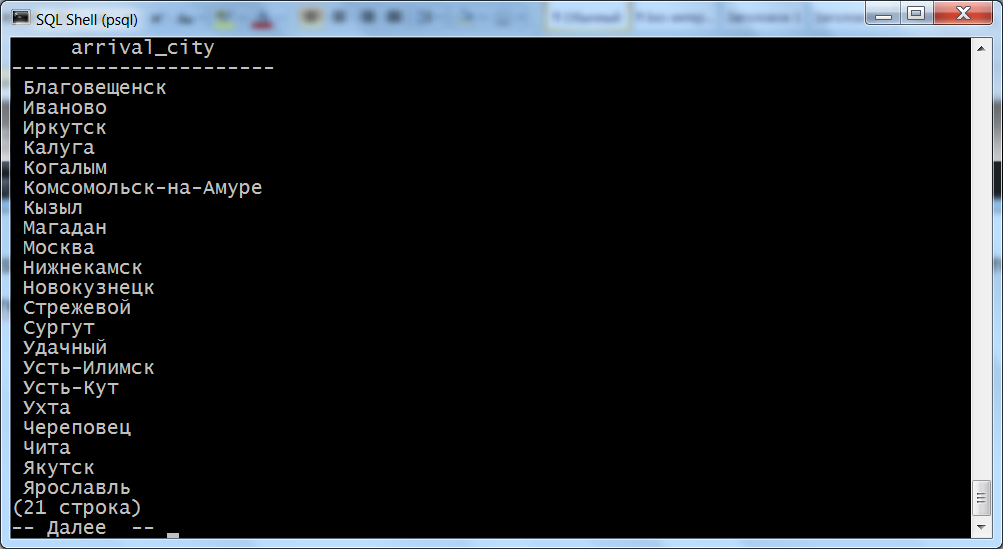
**EXISTS**

select distinct arrival\_city from routes as r1 where not exists (select arrival\_city from routes where departure\_city = 'Москва' and arrival\_city = r1.arrival\_city);



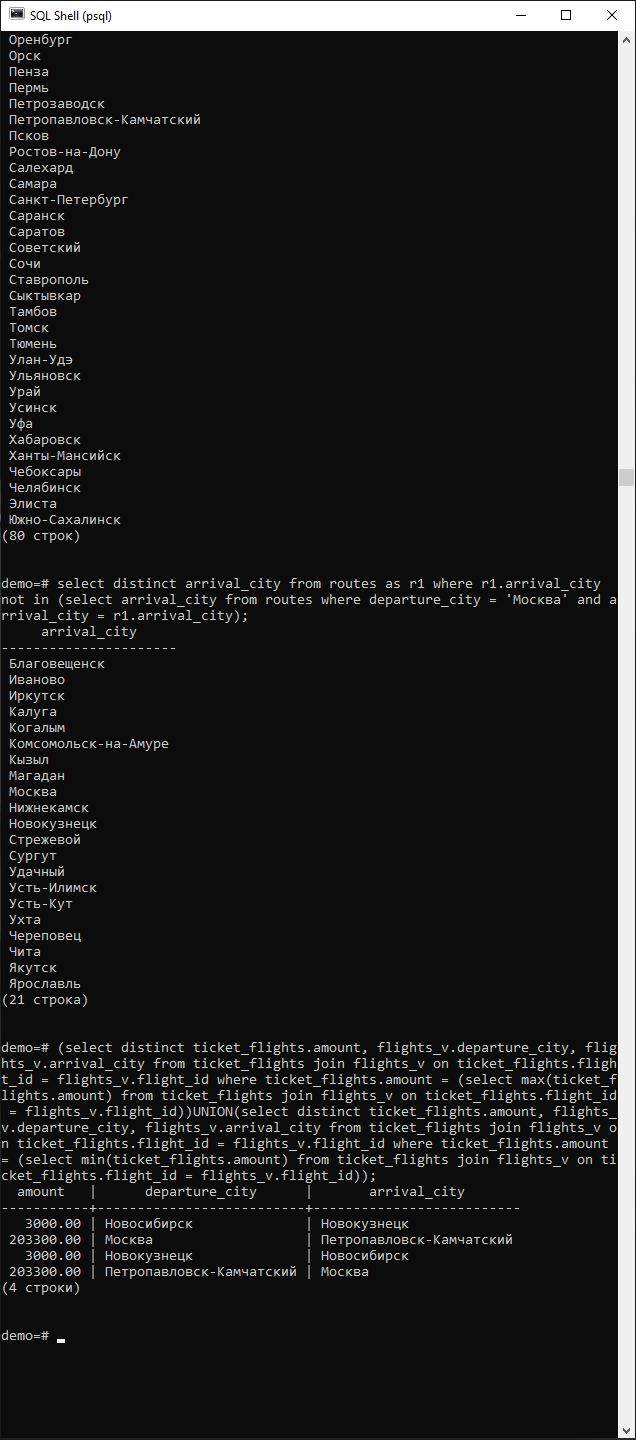
**IN**

select distinct arrival\_city from routes as r1 where r1.arrival\_city not in (select arrival\_city from routes where departure\_city = 'Москва' and arrival\_city = r1.arrival\_city);



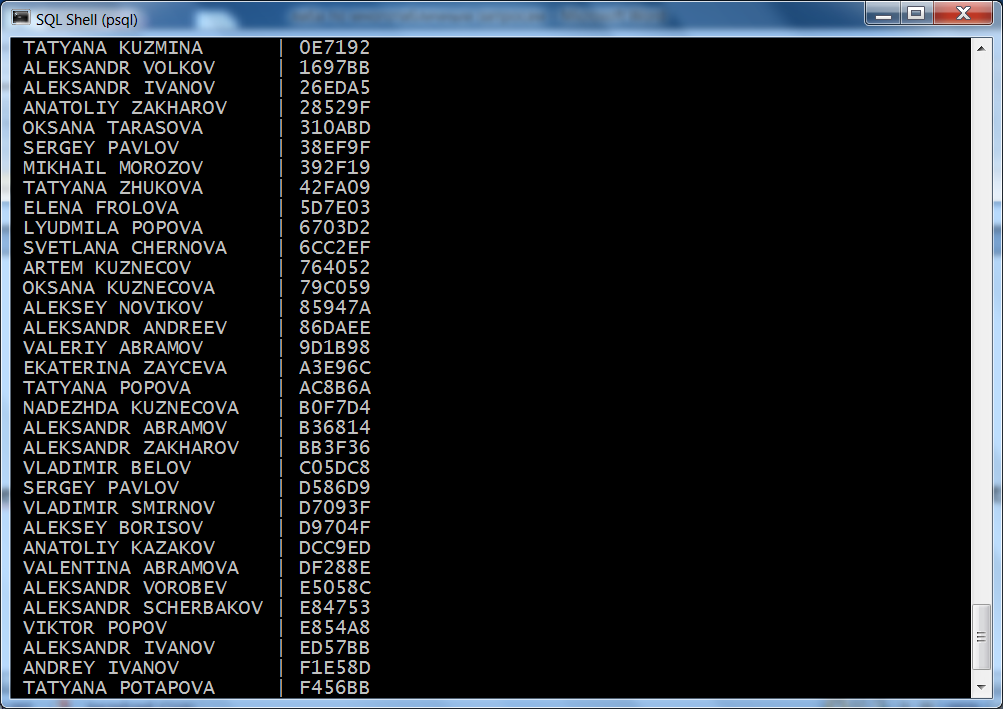
11) Вывести самый дорогой(возможно, не один) **перелет** и самый дешевый (возможно, не один).(Объединил через UNION)

(select distinct ticket\_flights.amount, flights\_v.departure\_city, flights\_v.arrival\_city from ticket\_flights join flights\_v on ticket\_flights.flight\_id = flights\_v.flight\_id where ticket\_flights.amount = (select max(ticket\_flights.amount) from ticket\_flights join flights\_v on ticket\_flights.flight\_id = flights\_v.flight\_id)) UNION (select distinct ticket\_flights.amount, flights\_v.departure\_city, flights\_v.arrival\_city from ticket\_flights join flights\_v on ticket\_flights.flight\_id = flights\_v.flight\_id where ticket\_flights.amount = (select min(ticket\_flights.amount) from ticket\_flights join flights\_v on ticket\_flights.flight\_id = flights\_v.flight\_id));



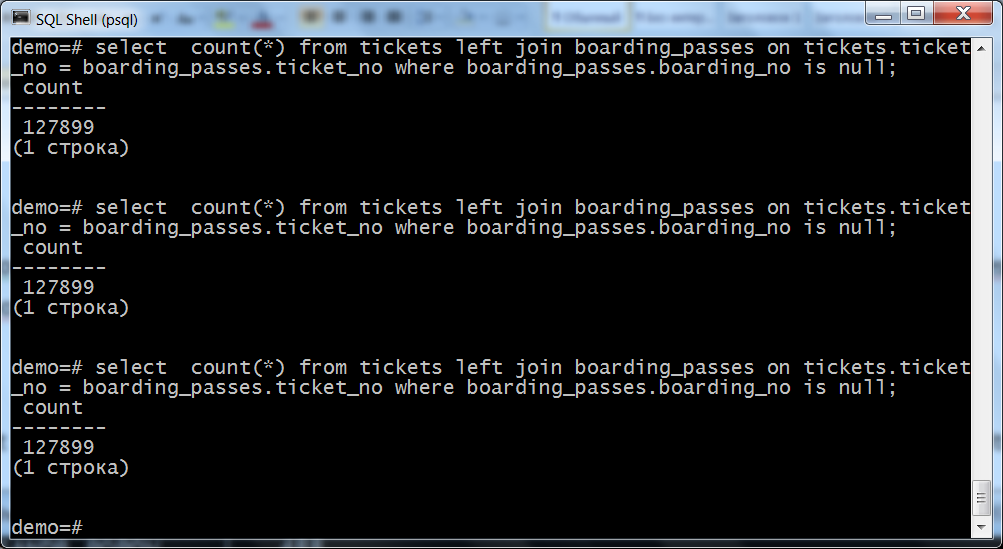
12) Вывести пассажиров, которые бронировали более одного билета за раз.

select passenger\_name, book\_ref from tickets group by (book\_ref, passenger\_name) having count(\*)>1;



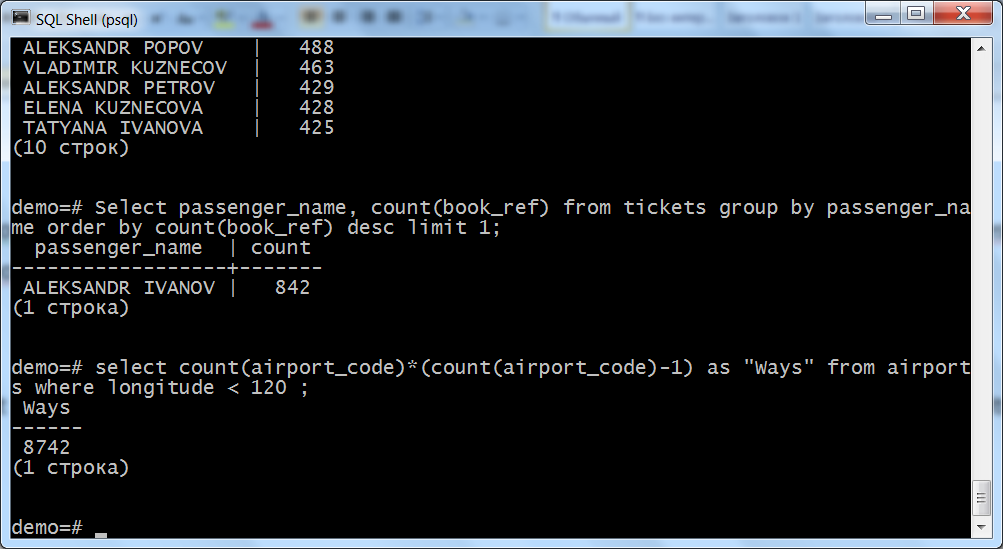
13) Вывести число билетов, которые были забронированы, но на них не был получен посадочный талон (пассажир не пришел на регистрацию)

select count(\*) from tickets left join boarding\_passes on tickets.ticket\_no = boarding\_passes.ticket\_no where boarding\_passes.boarding\_no is null;

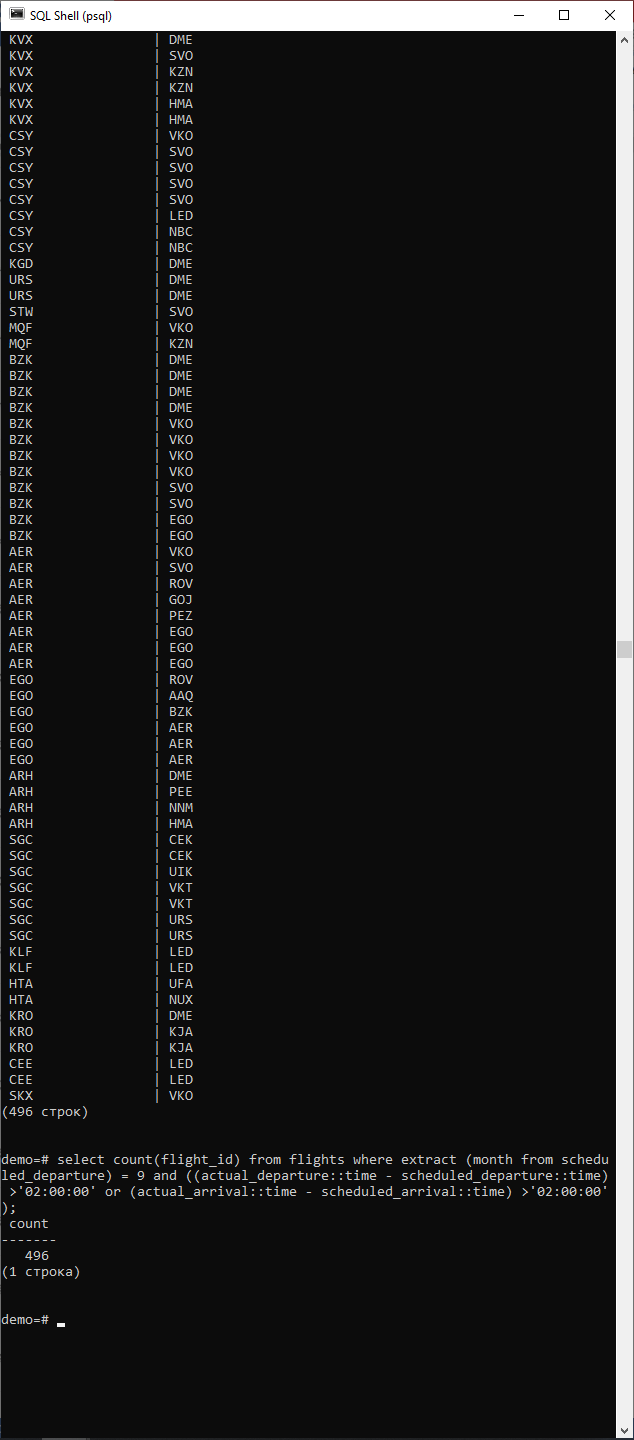


14) Каким будет общее число различных маршрутов, которые теоретически можно проложить между всеми городами, расположенными западнее 120 долготы. БЕЗ ПОДЗАПРОСА.

select count(airport\_code)\*(count(airport\_code)-1) as "Ways" from airports where longitude < 120 ; /%при условии что можно летать между аэропортами одного города%/

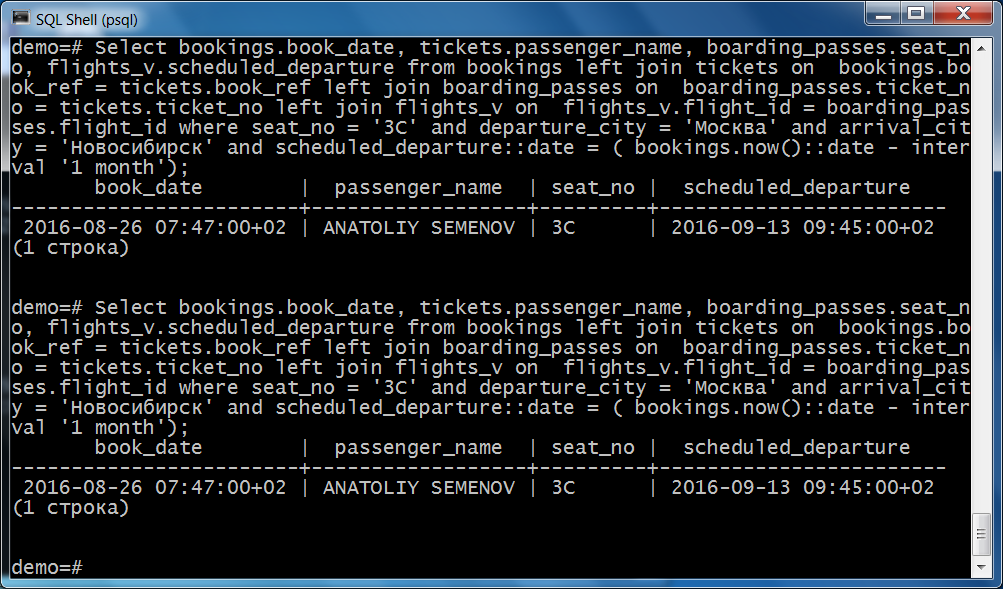


15) У каждого рейса есть запланированные дата и время вылета scheduled\_departure и прибытия scheduled\_arrival. Реальные время вылета actual\_departure и прибытия actual\_arrival могут отличаться: обычно не сильно, но иногда и на несколько часов, если рейс задержан. Вывести рейсы выполняемые в январе, которые задержали более чем на 2 часа.(Взял сентябрь)



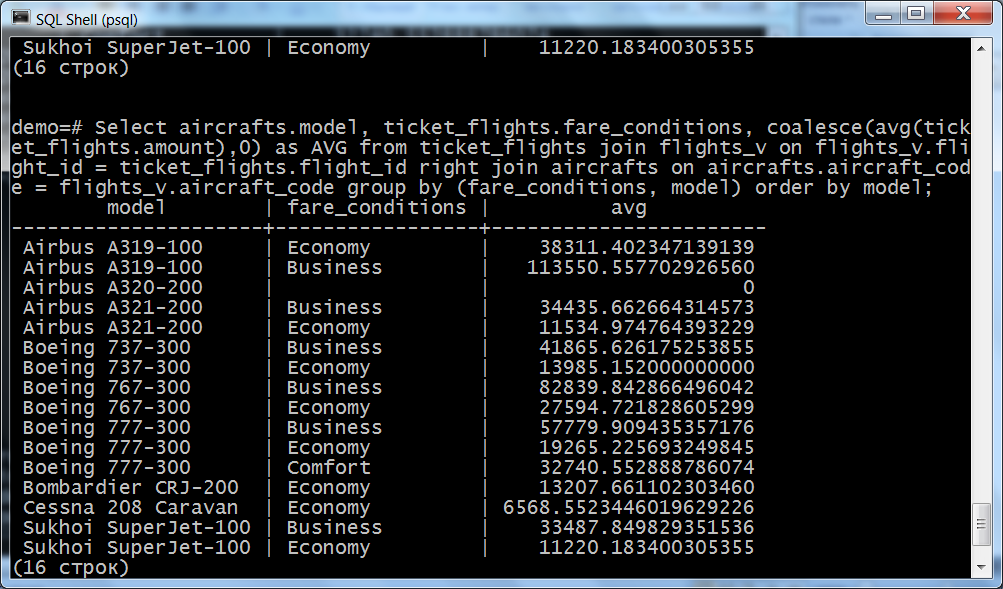
16) Кто летел месяц назад рейсом Москва — Новосибирск  на месте 3С? Когда было выполнено бронирование билета? «Месяц назад» отсчитывается от booking.now()

Select bookings.book\_date, tickets.passenger\_name, boarding\_passes.seat\_no, flights\_v.scheduled\_departure from bookings left join tickets on bookings.book\_ref = tickets.book\_ref left join boarding\_passes on boarding\_passes.ticket\_no = tickets.ticket\_no left join flights\_v on flights\_v.flight\_id = boarding\_passes.flight\_id where seat\_no = '3C' and departure\_city = 'Москва' and arrival\_city = 'Новосибирск' and scheduled\_departure::date = ( bookings.now()::date - interval '1 month');



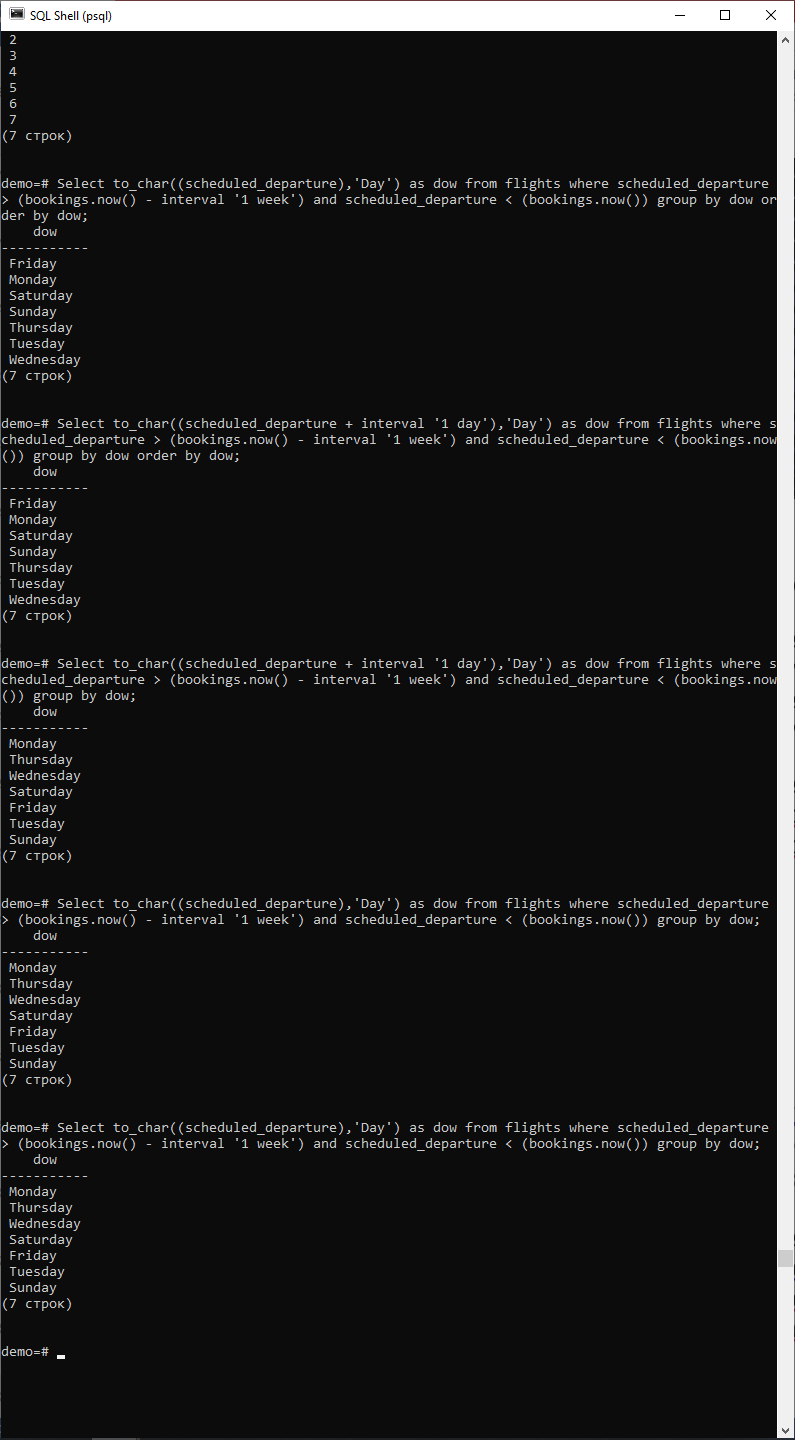
18) Посчитать среднюю цену билетов по каждому классу обслуживани в различных самолетах. Если для самолета нет проданных билетов, то он должен быть выведент в результате с ценой 0.

demo=# Select aircrafts.model, ticket\_flights.fare\_conditions, coalesce(avg(ticket\_flights.amount),0) as AVG from ticket\_flights join flights\_v on flights\_v.flight\_id = ticket\_flights.flight\_id right join aircrafts on aircrafts.aircraft\_code = flights\_v.aircraft\_code group by (fare\_conditions, model) order by model;

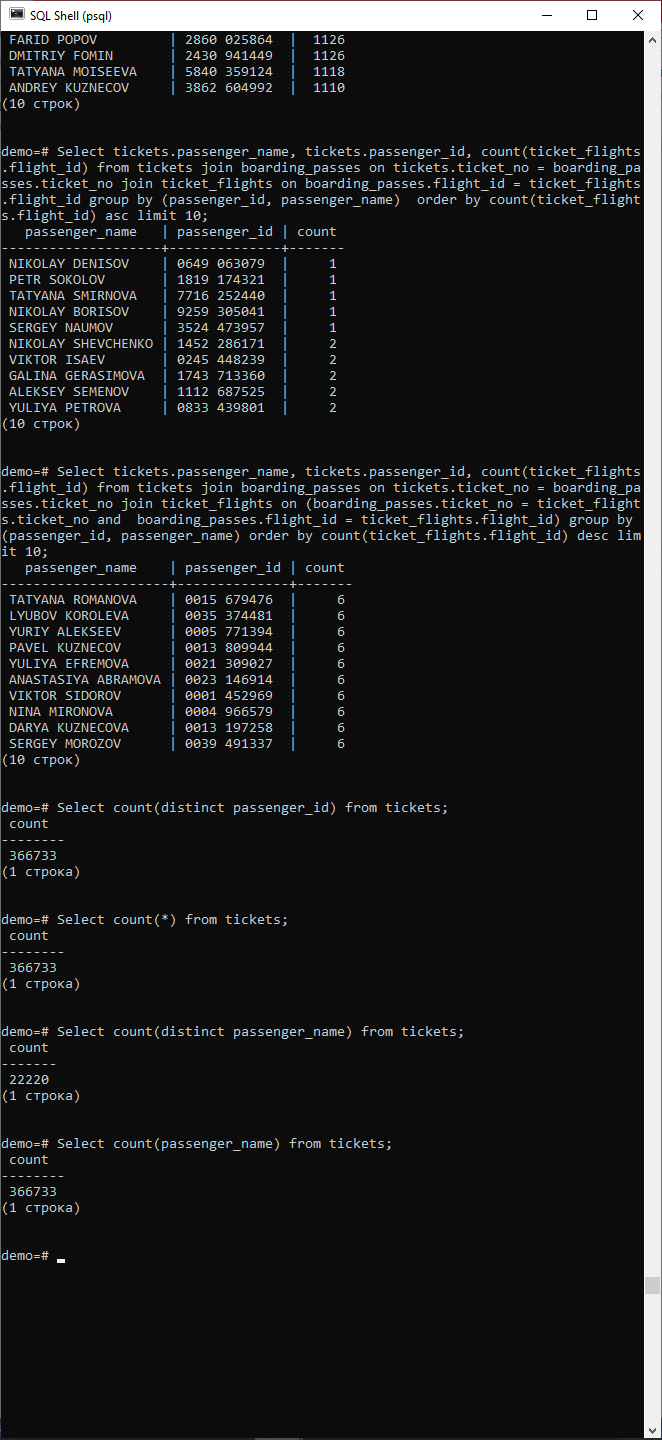


19) В какие дни недели состоялись вылеты самолетов за последнюю неделю. «Неделю» отсчитывать от booking.now() (Вместо EXTRACT() использовал TO\_CHAR())

Select to\_char((scheduled\_departure),'Day') as dow from flights where scheduled\_departure > (bookings.now() - interval '1 week') and scheduled\_departure < (bookings.now()) group by dow;



20)

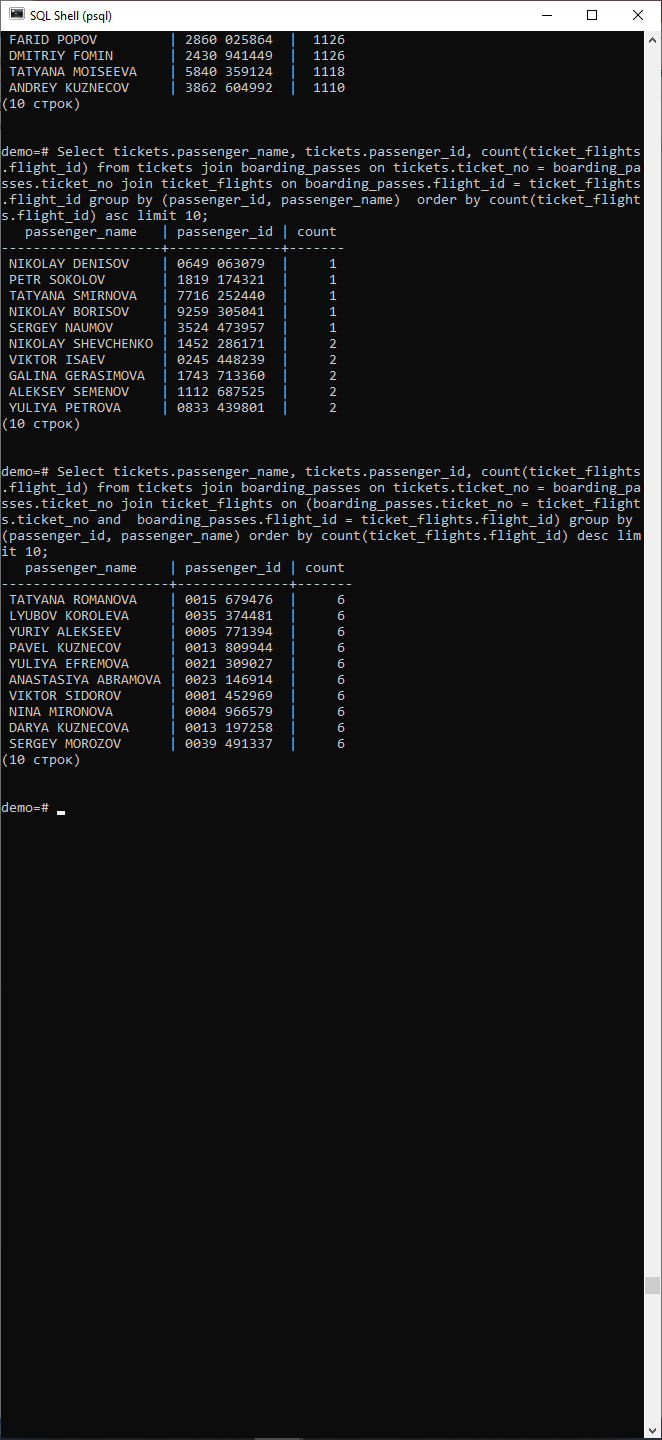


Исходя из данной информации можно утверждать, что каждому пассажиру принадлежит только один билет. Следовательно, пассажир оплативший самое большое количество перелетов тот, у которого этих самых перелетов в билете наибольшее количество, а пассажир совершивший наибольшее количество перелетов это тот, кто еще получил по каждому перелету соответствующий посадочный талон (прошел регистрацию на рейс)

Вывести данные о пассажире, который **совершил** самое большое количество перелетов.

(объединение таблиц с билетами, перелётами и посадочными талонами)

Select tickets.passenger\_name, tickets.passenger\_id, count(ticket\_flights.flight\_id) from tickets join boarding\_passes on tickets.ticket\_no = boarding\_passes.ticket\_no join ticket\_flights on (boarding\_passes.ticket\_no = ticket\_flights.ticket\_no and boarding\_passes.flight\_id = ticket\_flights.flight\_id) group by (passenger\_id, passenger\_name) order by count(ticket\_flights.flight\_id) desc limit 10;



Вывести данные о пассажире, который **оплатил** самое большое количество перелетов.

(объединение таблиц с билетами и перелётами)

Select tickets.passenger\_name, tickets.passenger\_id, count(ticket\_flights.flight\_id) from tickets join ticket\_flights on ticket\_flights.ticket\_no = tickets.ticket\_no group by (passenger\_id, passenger\_name) order by count(ticket\_flights.flight\_id) desc limit 10;

