

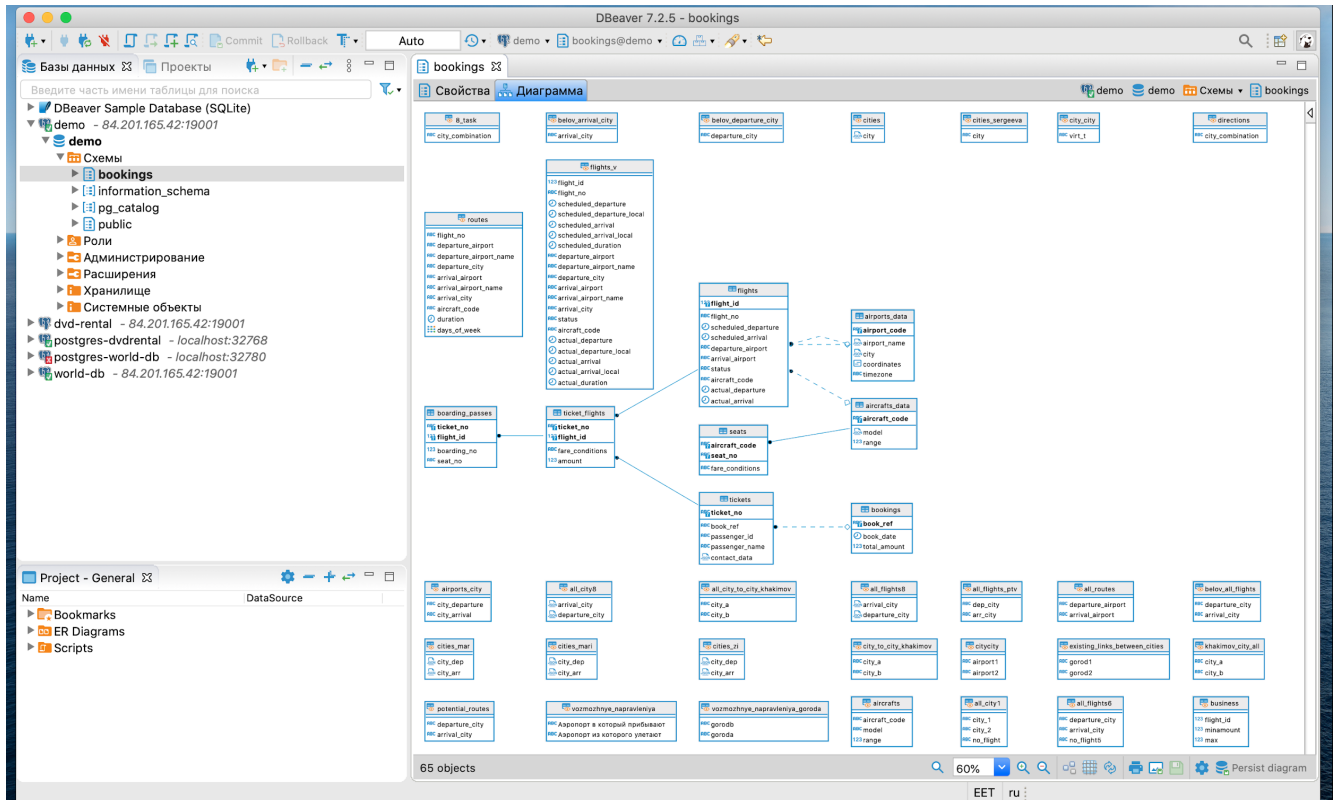
Итоговая работа по курсу “SQL и получение данных”

1	Подключение к базе данных	1
2	Описание и анализ базы данных	2
3	Решение задач на написание SQL-запросов к базе данных	11

1 Подключение к базе данных

В работе использовался облачный тип подключения.

Скриншот ER-диаграммы из DBeaver:



2 Описание и анализ базы данных

2.1 Краткое описание базы данных

База данных состоит из восьми таблиц и двух представлений.

Список отношений:

Имя	Тип	Описание
aircrafts_data	таблица	самолеты
airports_data	таблица	аэропорты
boarding_passes	таблица	посадочные талоны
bookings	таблица	бронирования
flights	таблица	рейсы
seats	таблица	места
ticket_flights	таблица	перелеты
tickets	таблица	билеты
flights_v	представление	рейсы
routes	материализованное представление	маршруты

2.2 Развернутый анализ базы данных: описание таблиц, представлений и связей между ними

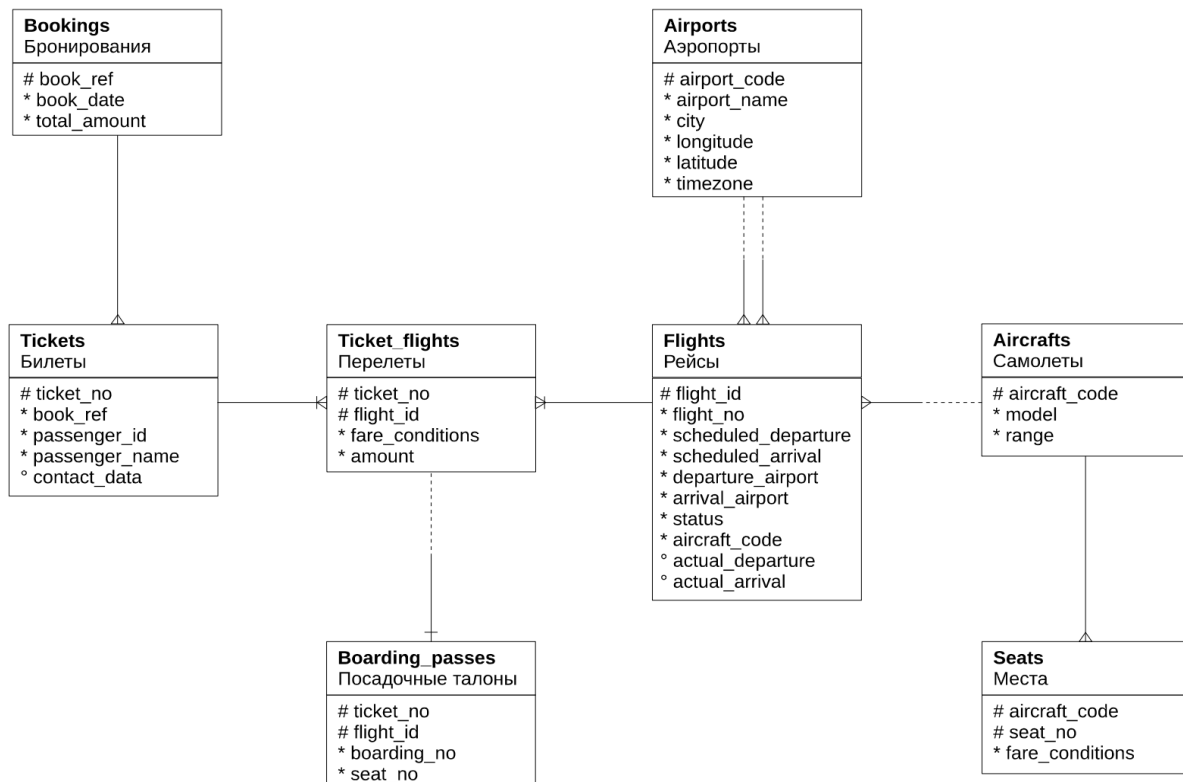


Таблица *aircrafts_data* (самолеты)

Каждая модель воздушного судна идентифицируется своим трехзначным кодом (*aircraft_code*). Указывается также название модели (*model*) и максимальная дальность полета в километрах (*range*).

Столбец	Тип	Модификаторы	Описание
<i>aircraft_code</i>	char(3)	NOT NULL	Код самолета, IATA
<i>model</i>	text	NOT NULL	Модель самолета
<i>range</i>	integer	NOT NULL	Максимальная дальность полета, км

Ограничение первичного ключа - *aircraft_code*.

Есть проверяющее ограничение для максимальной дальности полета (ее значение должно быть положительным): CHECK (*range* > 0).

На таблицу есть следующие ссылки извне:

В таблице *flights* ограничение внешнего ключа для атрибута *aircraft_code*;

В таблице *seats* ограничение внешнего ключа для атрибута *aircraft_code*.

Таблица *airports_data* (аэропорты)

Аэропорт идентифицируется трехбуквенным кодом (*airport_code*) и имеет свое имя (*airport_name*). Для города не предусмотрено отдельной сущности, но название (*city*) указывается и может служить для того, чтобы определить аэропорты одного города. Также указывается широта (*latitude*), долгота (*longitude*) и часовой пояс (*timezone*).

Столбец	Тип	Модификаторы	Описание
<i>airport_code</i>	char(3)	NOT NULL	Код аэропорта
<i>airport_name</i>	text	NOT NULL	Название аэропорта
<i>city</i>	text	NOT NULL	Город
<i>longitude</i>	float	NOT NULL	Координаты аэропорта: долгота
<i>latitude</i>	float	NOT NULL	Координаты аэропорта: широта
<i>timezone</i>	text	NOT NULL	Временная зона аэропорта

Ограничение первичного ключа - *airport_code*.

Ссылки извне:

В таблице *flights* ограничения внешнего ключа для атрибутов *arrival_airport* и *departure_airport* (ссылка на *airport_code*).

Таблица *boarding_passes* (посадочные талоны)

При регистрации на рейс, которая возможна за сутки до плановой даты отправления, пассажиру выдается посадочный талон. Он идентифицируется также, как и перелет - номером билета и номером рейса. Посадочным талонам присваиваются последовательные номера (*boarding_no*) в порядке регистрации пассажиров на рейс (этот номер будет уникальным только в пределах данного рейса). В посадочном талоне указывается номер места (*seat_no*).

Столбец	Тип	Модификаторы	Описание
<i>ticket_no</i>	<i>char(13)</i>	NOT NULL	Номер билета
<i>flight_id</i>	<i>integer</i>	NOT NULL	Идентификатор рейса
<i>boarding_no</i>	<i>integer</i>	NOT NULL	Номер посадочного талона
<i>seat_no</i>	<i>varchar(4)</i>	NOT NULL	Номер места

Первичный ключ является составным - *ticket_no*, *flight_id*.

Ограничение типа UNIQUE для атрибутов (*flight_id*, *boarding_no*).

Ограничение типа UNIQUE для атрибутов (*flight_id*, *seat_no*).

Ограничение внешнего ключа: атрибуты (*ticket_no*, *flight_id*) ссылаются на таблицу *ticket_flights* (*ticket_no*, *flight_id*).

Таблица *bookings* (бронирования)

Пассажир заранее (*book_date*, максимум за месяц до рейса) бронирует билет себе и, возможно, нескольким другим пассажирам.

Бронирование идентифицируется номером (*book_ref*, шестизначная комбинация букв и цифр). Поле *total_amount* хранит общую стоимость включенных в бронирование перелетов всех пассажиров.

Столбец	Тип	Модификаторы	Описание
<i>book_ref</i>	<i>char(6)</i>	NOT NULL	Номер бронирования
<i>book_date</i>	<i>timestampz</i>	NOT NULL	Дата бронирования
<i>total_amount</i>	<i>numeric(10,2)</i>	NOT NULL	Полная сумма бронирования

Ограничение первичного ключа - *book_ref*.

Ссылки извне:

В таблице *tickets* ограничение внешнего ключа для атрибута *book_ref*.

Таблица *flights* (рейсы)

Естественный ключ таблицы рейсов состоит из двух полей - номера рейса (*flight_no*) и даты отправления (*scheduled_departure*). Чтобы сделать внешние ключи на эту таблицу компактнее, в качестве первичного используется суррогатный ключ (*flight_id*).

Рейс всегда соединяет две точки - аэропорты вылета (*departure_airport*) и прибытия (*arrival_airport*). Такое понятие, как «рейс с пересадками» отсутствует: если из одного аэропорта до другого нет прямого рейса, в билет просто включаются несколько необходимых рейсов.

У каждого рейса есть плановые 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.

Статус рейса (*status*) может принимать одно из следующих значений:

- **Scheduled** - рейс доступен для бронирования. Это происходит за месяц до плановой даты вылета - до этого запись о рейсе не существует в базе данных.
- **On Time** - рейс доступен для регистрации (за сутки до плановой даты вылета) и не задержан.
- **Delayed** - рейс доступен для регистрации (за сутки до плановой даты вылета), но задержан.
- **Departed** - самолет уже вылетел и находится в воздухе.
- **Arrived** - самолет прибыл в пункт назначения.
- **Cancelled** - рейс отменен.

Столбец	Тип	Модификаторы	Описание
<i>flight_id</i>	<i>serial</i>	NOT NULL	Идентификатор рейса
<i>flight_no</i>	<i>char(6)</i>	NOT NULL	Номер рейса
<i>scheduled_departure</i>	<i>timestampz</i>	NOT NULL	Время вылета по расписанию
<i>scheduled_arrival</i>	<i>timestampz</i>	NOT NULL	Время прилёта по расписанию
<i>departure_airport</i>	<i>char(3)</i>	NOT NULL	Аэропорт отправления
<i>arrival_airport</i>	<i>char(3)</i>	NOT NULL	Аэропорт прибытия
<i>status</i>	<i>varchar(20)</i>	NOT NULL	Статус рейса
<i>aircraft_code</i>	<i>char(3)</i>	NOT NULL	Код самолета, IATA
<i>actual_departure</i>	<i>timestampz</i>		Фактическое время вылета
<i>actual_arrival</i>	<i>timestampz</i>		Фактическое время прилёта

Ограничение первичного ключа - flight_id.

Ограничение типа UNIQUE для атрибутов (flight_no, scheduled_departure).

Проверяющие ограничения:

CHECK (scheduled_arrival > scheduled_departure) - плановое время прибытия должно быть больше планового времени вылета.

CHECK ((actual_arrival is null) or ((actual_departure is not null and actual_arrival is not null) and (actual_arrival > actual_departure))) - реальное время прилета должно быть null либо реальное время вылета не null и реальное время прилета не null (реальное время прилета при этом больше реального времени вылета).

CHECK (status in ('On Time', 'Delayed', 'Departed', 'Arrived', 'Scheduled', 'Cancelled')) - статус рейса может иметь одно из перечисленных значений.

Ограничения внешнего ключа:

Атрибут aircraft_code ссылается на таблицу aircrafts_data (aircraft_code);

Атрибут arrival_airport ссылается на таблицу airports_data (airport_code);

Атрибут departure_airport ссылается на таблицу airports_data (airport_code).

Ссылки извне:

В таблице ticket_flights атрибут flight_id имеет ограничение внешнего ключа - ссылка на атрибут flight_id.

Таблица seats (места)

Места определяют схему салона каждой модели. Каждое место определяется своим номером (seat_no) и имеет закрепленный за ним класс обслуживания (fare_conditions) - Economy, Comfort или Business.

Столбец	Тип	Модификаторы	Описание
aircraft_code	char(3)	NOT NULL	Код самолета, IATA
seat_no	varchar(4)	NOT NULL	Номер места
fare_conditions	varchar(10)	NOT NULL	Класс обслуживания

Первичный ключ таблицы является составным - aircraft_code, seat_no.

Проверяющие ограничения:

CHECK (fare_conditions in ('Economy', 'Comfort', 'Business')) - класс обслуживания должен иметь одно из приведенных значений.

Ограничения внешнего ключа:

Атрибут aircraft_code ссылается на таблицу aircrafts_data (aircraft_code).

Таблица ticket_flights (перелеты)

Перелет соединяет билет с рейсом и идентифицируется их номерами.

Для каждого перелета указываются его стоимость (amount) и класс обслуживания (fare_conditions).

Столбец	Тип	Модификаторы	Описание
ticket_no	char(13)	NOT NULL	Номер билета
flight_id	integer	NOT NULL	Идентификатор рейса
fare_conditions	varchar(10)	NOT NULL	Класс обслуживания
amount	numeric(10,2)	NOT NULL	Стоимость перелета

Первичный ключ таблицы является составным - ticket_no, flight_id.

Ограничения-проверки:

Стоимость перелета должна быть неотрицательна: CHECK (amount >= 0);

Класс обслуживания должен иметь одно из приведенных значений: CHECK (fare_conditions in ('Economy', 'Comfort', 'Business')).

Ограничения внешнего ключа:

Атрибут flight_id ссылается на таблицу flights (flight_id);

Атрибут ticket_no ссылается на таблицу tickets (ticket_no).

Ссылки извне:

В таблице boarding_passes есть ограничение внешнего ключа для атрибутов (ticket_no, flight_id).

Таблица tickets (билеты)

Билет имеет уникальный номер (ticket_no), состоящий из 13 цифр.

Билет содержит идентификатор пассажира (`passenger_id`) - номер документа, удостоверяющего личность, его фамилию и имя (`passenger_name`) и контактную информацию (`contact_data`).

Ни идентификатор пассажира, ни имя не являются постоянными (можно поменять паспорт, можно сменить фамилию), поэтому однозначно найти все билеты одного и того же пассажира невозможно.

Столбец	Тип	Модификаторы	Описание
<code>ticket_no</code>	<code>char(13)</code>	<code>NOT NULL</code>	Номер билета
<code>book_ref</code>	<code>char(6)</code>	<code>NOT NULL</code>	Номер бронирования
<code>passenger_id</code>	<code>varchar(20)</code>	<code>NOT NULL</code>	Идентификатор пассажира
<code>passenger_name</code>	<code>text</code>	<code>NOT NULL</code>	Имя пассажира
<code>contact_data</code>	<code>jsonb</code>		Контактные данные пассажира

Ограничение первичного ключа - `ticket_no`.

Ограничение внешнего ключа:

Атрибут `book_ref` ссылается на таблицу `bookings` (`book_ref`).

Ссылки извне:

В таблице `ticket_flights` ограничение внешнего ключа для атрибута `ticket_no`.

Представление "flights_v" (рейсы)

Столбец	Тип	Описание
<code>flight_id</code>	<code>integer</code>	Идентификатор рейса
<code>flight_no</code>	<code>char(6)</code>	Номер рейса
<code>scheduled_departure</code>	<code>timestampz</code>	Время вылета по расписанию
<code>scheduled_departure_local</code>	<code>timestamp</code>	Время вылета по расписанию, местное время в пункте отправления
<code>scheduled_arrival</code>	<code>timestampz</code>	Время прилёта по расписанию
<code>scheduled_arrival_local</code>	<code>timestamp</code>	Время прилёта по расписанию, местное время в пункте прибытия
<code>scheduled_duration</code>	<code>interval</code>	Планируемая продолжительность полета
<code>departure_airport</code>	<code>char(3)</code>	Код аэропорта отправления
<code>departure_airport_name</code>	<code>text</code>	Название аэропорта отправления
<code>departure_city</code>	<code>text</code>	Город отправления
<code>arrival_airport</code>	<code>char(3)</code>	Код аэропорта прибытия
<code>arrival_airport_name</code>	<code>text</code>	Название аэропорта прибытия
<code>arrival_city</code>	<code>text</code>	Город прибытия
<code>status</code>	<code>varchar(20)</code>	Статус рейса
<code>aircraft_code</code>	<code>char(3)</code>	Код самолета, IATA
<code>actual_departure</code>	<code>timestampz</code>	Фактическое время вылета
<code>actual_departure_local</code>	<code>timestamp</code>	Фактическое время вылета, местное время в пункте отправления
<code>actual_arrival</code>	<code>timestampz</code>	Фактическое время прилёта
<code>actual_arrival_local</code>	<code>timestamp</code>	Фактическое время прилёта, местное время в пункте прибытия
<code>actual_duration</code>	<code>interval</code>	Фактическая продолжительность полета

Над таблицей flights создано представление flights_v.

Представление содержит дополнительную информацию:

- расшифровку данных об аэропорте вылета (departure_airport, departure_airport_name, departure_city),
- расшифровку данных об аэропорте прибытия (arrival_airport, arrival_airport_name, arrival_city),
- местное время вылета (scheduled_departure_local, actual_departure_local),
- местное время прибытия (scheduled_arrival_local, actual_arrival_local),
- продолжительность полета (scheduled_duration, actual_duration).

Материализованное представление routes (маршруты)

Таблица рейсов содержит избыточность: из нее можно было бы выделить информацию о маршруте (номер рейса, аэропорты отправления и назначения), которая не зависит от конкретных дат рейсов. Именно такая информация и составляет материализованное представление routes.

Столбец	Тип	Описание
flight_no	char(6)	Номер рейса
departure_airport	char(3)	Код аэропорта отправления
departure_airport_name	text	Название аэропорта отправления
departure_city	text	Город отправления
arrival_airport	char(3)	Код аэропорта прибытия
arrival_airport_name	text	Название аэропорта прибытия
arrival_city	text	Город прибытия
aircraft_code	char(3)	Код самолета, IATA
duration	interval	Продолжительность полета
days_of_week	integer[]	Дни недели, когда выполняются рейсы

2.3 Описание логики

Основной сущностью является бронирование (bookings). В одно бронирование можно включить несколько пассажиров, каждому из которых выписывается отдельный билет (tickets), соответственно, в одно бронирование может входить несколько билетов.

Как таковой пассажир не является отдельной сущностью. Как имя, так и номер документа пассажира могут меняться с течением времени, так что невозможно однозначно найти все билеты одного человека; для простоты можно считать, что все пассажиры уникальны.

Каждый билет включает один или несколько перелетов (ticket_flights). При этом, все билеты в одном бронировании имеют одинаковый набор перелетов.

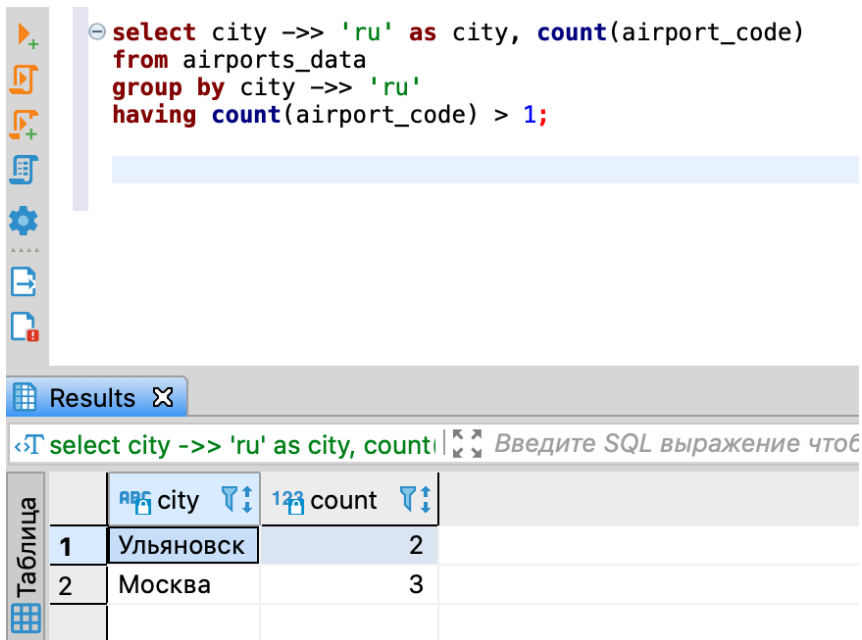
Каждый рейс (flights) - перелет следует из одного аэропорта (airports) в другой. Рейсы с одним номером имеют одинаковые пункты вылета и назначения, но будут отличаться датой отправления.

При регистрации на рейс пассажиру выдается посадочный талон (boarding_passes), в котором указано место в самолете. Комбинация рейса и места в самолете должна быть уникальной, чтобы не допустить выдачу двух посадочных талонов на одно место.

Количество мест (seats) в самолете и их распределение по классам обслуживания зависит от модели самолета (aircrafts), выполняющего рейс. Предполагается, что каждая модель самолета имеет только одну компоновку салона. Схема данных не контролирует, что места в посадочных талонах соответствуют имеющимся в самолете (такая проверка может быть сделана с использованием табличных триггеров или в приложении).

3 Решение задач на написание SQL-запросов к базе данных

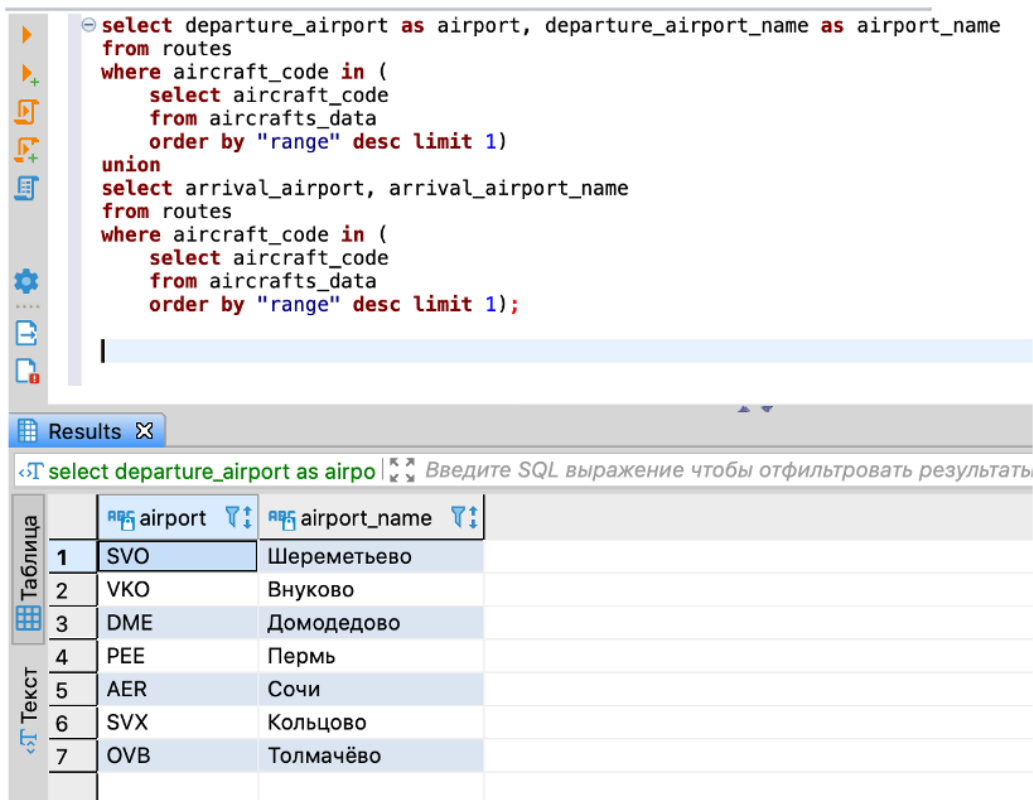
1. В каких городах больше одного аэропорта?



```
select city --> 'ru' as city, count(airport_code)
from airports_data
group by city --> 'ru'
having count(airport_code) > 1;
```

city	count
Ульяновск	2
Москва	3

2. В каких аэропортах есть рейсы, выполняемые самолетом с максимальной дальностью перелета?



```
select departure_airport as airport, departure_airport_name as airport_name
from routes
where aircraft_code in (
    select aircraft_code
    from aircrafts_data
    order by "range" desc limit 1)
union
select arrival_airport, arrival_airport_name
from routes
where aircraft_code in (
    select aircraft_code
    from aircrafts_data
    order by "range" desc limit 1);
```

airport	airport_name
SVO	Шереметьево
VKO	Внуково
DME	Домодедово
PEE	Пермь
AER	Сочи
SVX	Кольцово
OVB	Толмачёво

3. Вывести 10 рейсов с максимальным временем задержки вылета.

```

select
    flight_id,
    flight_no,
    scheduled_departure,
    actual_departure,
    (actual_departure - scheduled_departure) as difference
from flights
where actual_departure is not null
order by difference desc
limit 10;

```

	flight_id	flight_no	scheduled_departure	actual_departure	difference
1	17 368	PG0132	2017-05-22 12:25:00	2017-05-22 17:06:00	04:41:00
2	31 478	PG0531	2017-05-17 09:15:00	2017-05-17 13:56:00	04:41:00
3	29 148	PG0589	2017-07-29 15:30:00	2017-07-29 20:07:00	04:37:00
4	2 702	PG0164	2017-07-29 15:25:00	2017-07-29 19:53:00	04:28:00
5	44 711	PG0544	2017-07-06 09:55:00	2017-07-06 14:23:00	04:28:00
6	48 079	PG0364	2017-07-19 11:45:00	2017-07-19 16:12:00	04:27:00
7	53 560	PG0151	2017-07-03 14:00:00	2017-07-03 18:27:00	04:27:00
8	6 310	PG0337	2017-06-01 12:45:00	2017-06-01 17:10:00	04:25:00
9	55 271	PG0354	2017-07-03 08:45:00	2017-07-03 13:10:00	04:25:00
10	45 487	PG0444	2017-05-25 12:30:00	2017-05-25 16:53:00	04:23:00

4. Были ли брони, по которым не были получены посадочные талоны?

```

select distinct a.book_ref
from (
    select a.book_ref, b.ticket_no, c.flight_id, d.boarding_no
    from bookings as a
    left join tickets as b on b.book_ref = a.book_ref
    left join ticket_flights as c on c.ticket_no = b.ticket_no
    left join boarding_passes as d on d.ticket_no = c.ticket_no and d.flight_id = c.flight_id
    where d.boarding_no is null) as a;

```

	book_ref
1	000068
2	000181
3	0002D8
4	0002DB
5	00034E
6	000374
7	00044D
8	0004B0
9	0004E1
10	00053F
11	0005F4
12	0006C3
13	0006F5
14	000784
15	000836
16	000862
17	0008DF
18	0008F4

200 строк получено - 5.52s (+1ms)

5. Найдите свободные места для каждого рейса, их % отношение к общему количеству мест в самолете. Добавьте столбец с накопительным итогом - суммарное количество вывезенных пассажиров из аэропорта за день. Т.е. в этом столбце должна отражаться сумма - сколько человек уже вылетело из данного аэропорта на этом или более ранних рейсах за день.

```

select
  aa.flight_id, aa.departure_airport, aa.actual_departure,
  aa.aircraft_code,
  bb.all_seats, aa.busy_seats,
  (bb.all_seats-aa.busy_seats) as free_seats,
  round(((bb.all_seats-aa.busy_seats)/bb.all_seats)::float*100)::numeric,2) as percent_free,
  sum(aa.busy_seats) over
    (partition by aa.departure_airport, aa.actual_departure::date order by aa.actual_departure) as passengers
from
  (select a.flight_id, a.aircraft_code, count(b.seat_no) as busy_seats, a.departure_airport, a.actual_departure
   from flights as a
   left join boarding_passes as b on b.flight_id = a.flight_id
   group by a.flight_id, a.aircraft_code
   order by a.flight_id) as aa
left join (
  select aircraft_code, count(seat_no) as all_seats
  from seats
  group by aircraft_code) as bb on bb.aircraft_code = aa.aircraft_code
order by aa.departure_airport, aa.actual_departure;

```

	departure	aircraft_code	all_seats	busy_seats	free_seats	percent_free	passengers
1	7 12:26:00	SU9	97	4	93	95,88	4
2	7 13:09:00	733	130	49	81	62,31	53
3	8 12:25:00	SU9	97	7	90	92,78	7
4	8 13:08:00	733	130	47	83	63,85	54
5	9 12:27:00	SU9	97	2	95	97,94	2
6	9 13:07:00	733	130	50	80	61,54	52
7	0 12:26:00	SU9	97	5	92	94,85	5
8	0 13:09:00	733	130	49	81	62,31	54
9	1 12:28:00	SU9	97	1	96	98,97	1
10	1 13:06:00	733	130	51	79	60,77	52
11	2 12:30:00	SU9	97	5	92	94,85	5
12	2 13:07:00	733	130	60	70	53,85	65

200 строк получено - 1.432s (+5ms)

6. Найдите процентное соотношение перелетов по типам самолетов от общего количества.

```

select
  aircraft_code,
  count(flight_id) as count_flights,
  round((count(flight_id)/(select count(flight_id) from flights)::float*100)::numeric,2) as percentage
from flights
group by aircraft_code
order by percentage desc;

```

	aircraft_code	count_flights	percentage
1	CN1	18 394	28,01
2	CR2	17 920	27,29
3	SU9	16 870	25,69
4	321	3 872	5,9
5	733	2 522	3,84

7. Были ли города, в которые можно добраться бизнес - классом дешевле, чем эконом-классом в рамках перелета?

```

with
cte_1 as (
    select flight_id, max(amount) as max_economy
    from ticket_flights
    where fare_conditions = 'Economy'
    group by flight_id
    order by flight_id),
cte_2 as (
    select flight_id, min(amount) as min_business
    from ticket_flights
    where fare_conditions = 'Business'
    group by flight_id
    order by flight_id),
cte_3 as (
    select flight_id, arrival_city
    from flights_v)
select c1.flight_id, c1.max_economy, c2.min_business, c3.arrival_city
from cte_1 as c1
inner join cte_2 as c2 on c2.flight_id = c1.flight_id
inner join cte_3 as c3 on c3.flight_id = c2.flight_id and c3.flight_id = c1.flight_id
where c2.min_business < c1.max_economy

```

--- ОТВЕТ:
--- таких перелетов, в рамках которых можно было бы добраться в какой-либо город
--- бизнес-классом дешевле, чем эконом-классом НЕТ

ticket_flights(+)

with cte_1 as (select flight_id, m

flight_id	max_economy	min_business	arrival_city

8. Между какими городами нет прямых рейсов?

- Создали представление для всех возможных прямых рейсов между городами:

```

create view all_routess as (
    select distinct a.departure_city, b.arrival_city
    from routes as a, routes as b
    where a.departure_city != b.arrival_city
    order by a.departure_city, b.arrival_city);

select * from all_routess;

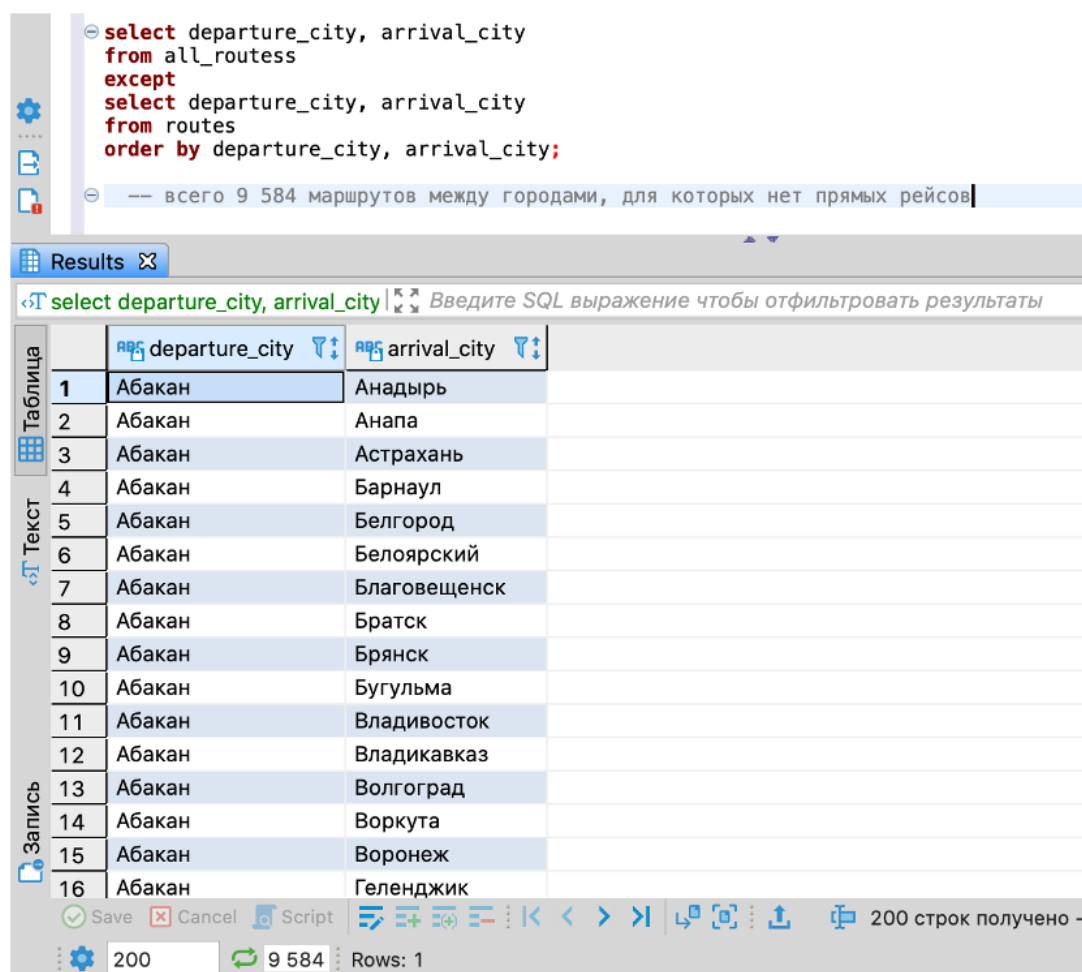
```

all_routess

select * from all_routess

	departure_city	arrival_city
1	Абакан	Анадырь
2	Абакан	Анапа
3	Абакан	Архангельск
4	Абакан	Астрахань
5	Абакан	Барнаул
6	Абакан	Белгород

- Нашли, между какими городами нет прямых рейсов:



```

select departure_city, arrival_city
from all_routes
except
select departure_city, arrival_city
from routes
order by departure_city, arrival_city;

```

-- всего 9 584 маршрутов между городами, для которых нет прямых рейсов

Results

select departure_city, arrival_city | Введите SQL выражение чтобы отфильтровать результаты

	departure_city	arrival_city
1	Абакан	Анадырь
2	Абакан	Анапа
3	Абакан	Астрахань
4	Абакан	Барнаул
5	Абакан	Белгород
6	Абакан	Белоярский
7	Абакан	Благовещенск
8	Абакан	Братск
9	Абакан	Брянск
10	Абакан	Бугульма
11	Абакан	Владивосток
12	Абакан	Владикавказ
13	Абакан	Волгоград
14	Абакан	Воркута
15	Абакан	Воронеж
16	Абакан	Геленджик

Save Cancel Script 200 строк получено - 200 9 584 Rows: 1

9. Вычислите расстояние между аэропортами, связанными прямыми рейсами, сравните с допустимой максимальной дальностью перелетов в самолетах, обслуживающих эти рейсы.

Кратчайшее расстояние между двумя точками А и В на земной поверхности (если принять ее за сферу) определяется зависимостью:

$$d = \arccos \{ \sin(\text{latitude_a}) \cdot \sin(\text{latitude_b}) + \cos(\text{latitude_a}) \cdot \cos(\text{latitude_b}) \cdot \cos(\text{longitude_a} - \text{longitude_b}) \}, \text{ где}$$

latitude_a и latitude_b - широты,

longitude_a, longitude_b - долготы данных пунктов,

d - расстояние между пунктами, измеряется в радианах длиной дуги большого круга земного шара.

Расстояние между пунктами, измеряемое в километрах, определяется по формуле: $L = d \cdot R$, где $R = 6371$ км - средний радиус земного шара.

-- вывели все прямые маршруты между аэропортами
 -- присоединили информацию по координатам аэропортов
 -- разделили координаты по столбцам на долготу / широту
 -- перевели координаты из градусов в радианы с помощью функции radians
 -- посчитали по формуле расстояние между аэропортами в радианах
 -- посчитали расстояние в километрах
 -- сравнили это расстояние с максимальной дальностью полета самолета

```

select
aa.departure_airport, aa.arrival_airport,
acos(sin(aa.latitude_a)*sin(aa.latitude_b)+cos(aa.longitude_a)*cos(aa.longitude_b)) as d,
acos(sin(aa.latitude_a)*sin(aa.latitude_b)+cos(aa.longitude_a)*cos(aa.longitude_b)) * 6371 as l,
bb."range",
(bb."range"-(acos(sin(aa.latitude_a)*sin(aa.latitude_b)+cos(aa.longitude_a)*cos(aa.longitude_b)) * 6371)) as differer
from (
select
a.departure_airport, a.arrival_airport,
radians(b.coordinates[0]) as longitude_a,
radians(b.coordinates[1]) as latitude_a,
radians(c.coordinates[0]) as longitude_b,
radians(c.coordinates[1]) as latitude_b,
a.aircraft_code
from routes as a
left join airports_data as b on b.airport_code = a.departure_airport
left join airports_data as c on c.airport_code = a.arrival_airport) as aa
left join aircrafts_data as bb on aa.aircraft_code = bb.aircraft_code;
  
```

routes(+)

select aa.departure_airport, aa.a

Введите SQL выражение чтобы отфильтровать результаты

	departure_airport	arrival_airport	d	l	range	difference
2	SGC	UIK	0,2601873543	1 657,6536342269	2 700	1 042,3463657731
3	IWA	AER	0,2356865013	1 501,5586997845	2 700	1 198,4413002155
4	AER	IWA	0,2356865013	1 501,5586997845	2 700	1 198,4413002155
5	DME	PKV	0,1002335237	638,5877792508	1 200	561,4122207492
6	PKV	DME	0,1002335237	638,5877792508	1 200	561,4122207492
7	VKO	JOK	0,1051639873	669,9997629922	1 200	530,0002370078
8	VKO	JOK	0,1051639873	669,9997629922	1 200	530,0002370078
9	JOK	VKO	0,1051639873	669,9997629922	1 200	530,0002370078
10	JOK	VKO	0,1051639873	669,9997629922	1 200	530,0002370078
11	IAR	TOF	0,4211656554	2 683,2463906527	2 700	16,7536093473
12	GDZ	VKO	0,1923313437	1 225,3429907051	2 700	1 474,6570092949
13	AER	SVO	0,2203983877	1 404,1581282246	11 100	9 695,8418717754

Save Cancel Script

200 строк получено - 415ms (+6ms)

200 710 Rows: 1