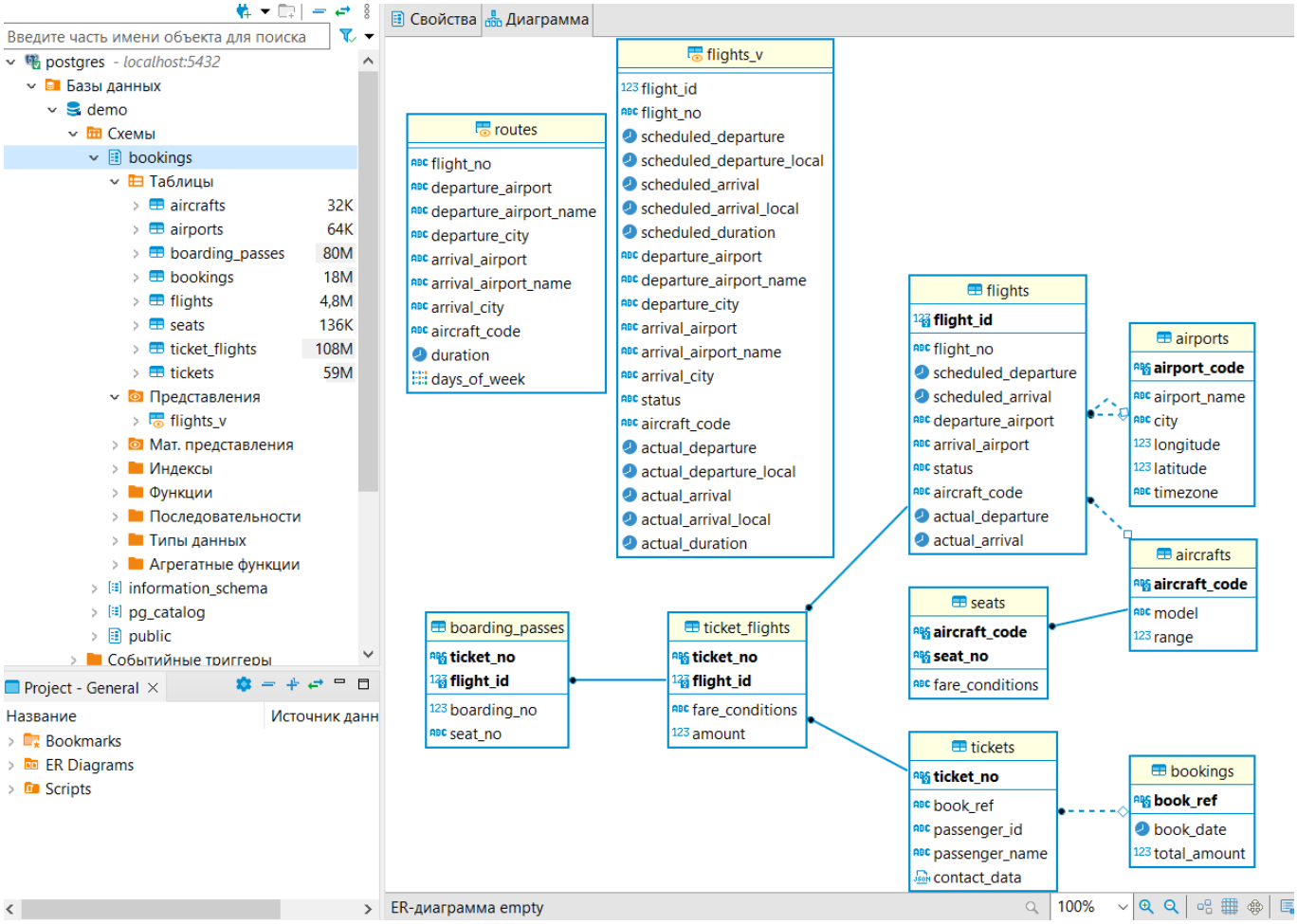
Итоговая работа по модулю “SQL и получение данных”

1. Тип подключения

В работе использовался локальный тип подключения.

2. Скриншот ER-диаграммы из DBeaver`a



3. Таблицы и представления БД

База данных demo. Схема bookings

Имя	Тип	Описание
aircrafts	таблица	Самолеты
airports	таблица	Аэропорты
boarding_passes	таблица	Посадочные талоны
bookings	таблица	Бронирования
flights	таблица	Рейсы
seats	таблица	Места
ticket_flights	таблица	Рейсы
tickets	таблица	Билеты
flights_v	представление	Рейсы
routes	мат. представление	Маршруты

4. Развернутый анализ БД

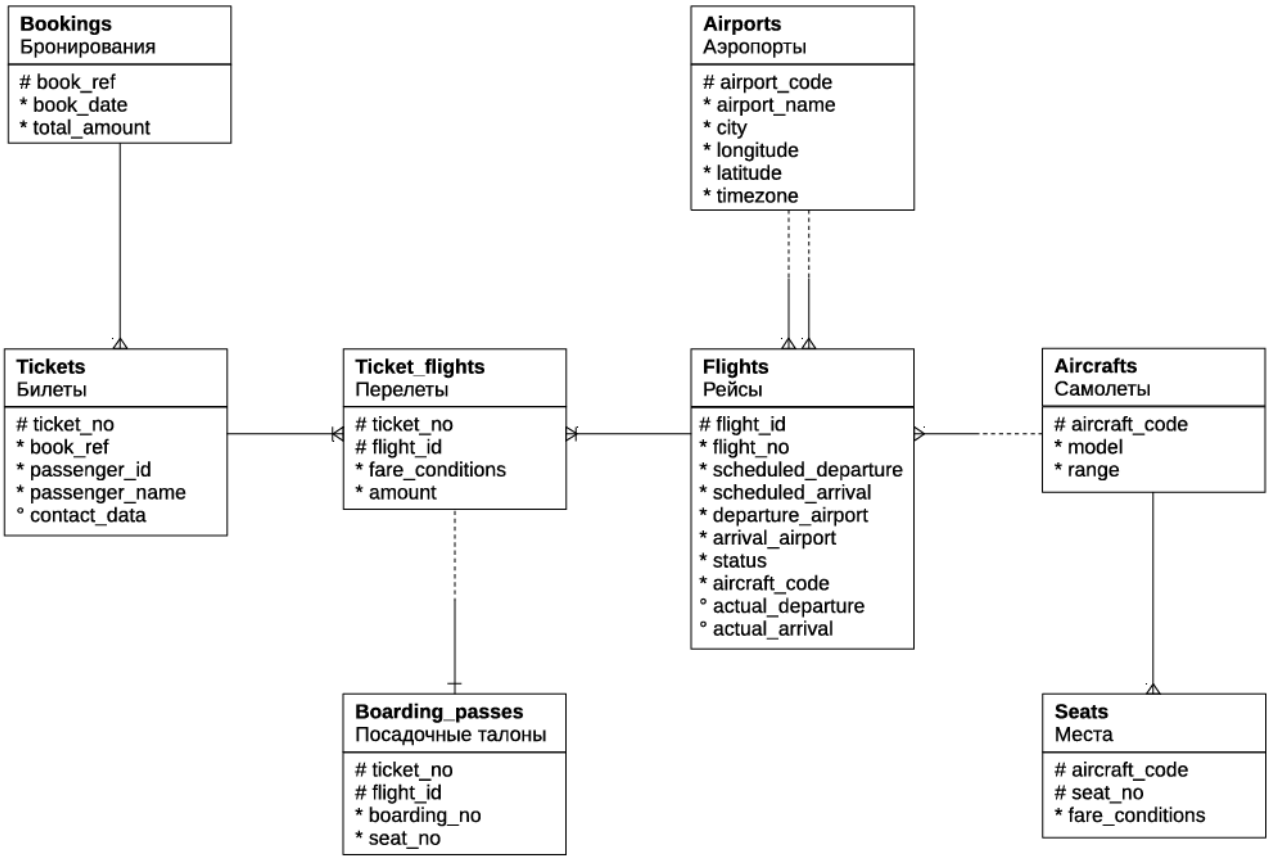


Рисунок 1 - Диаграмма схемы данных bookings

- **Таблица bookings.aircrafts**

  Таблица aircrafts состоит из трех столбцов:
  - aircraft_code
  - model
  - range

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
---------------+---------+--------------+-----------------------------------
 aircraft_code | char(3) | NOT NULL     | Код самолета, IATA
 model         | text    | NOT NULL     | Модель самолета
 range         | integer | NOT NULL     | Максимальная дальность полета, км
Индексы:
    PRIMARY KEY, btree (aircraft_code)
Ограничения-проверки:
    CHECK (range > 0)
Ссылки извне:
    TABLE "flights" FOREIGN KEY (aircraft_code)
        REFERENCES aircrafts(aircraft_code)
    TABLE "seats" FOREIGN KEY (aircraft_code)
        REFERENCES aircrafts(aircraft_code) ON DELETE CASCADE
```

Рисунок 2 - Свойства таблицы bookings.aircrafts

- **Таблица bookings.airports**

Таблица airports состоит из шести столбцов:
- airport_code
- airport_name
- city
- longitude
- latitude
- timezone

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 (longitude), долгота (latitude) и часовой пояс (timezone).

```
   Столбец    |   Тип   | Модификаторы |           Описание
--------------+---------+--------------+-------------------------------
 airport_code | char(3) | NOT NULL     | Код аэропорта
 airport_name | text    | NOT NULL     | Название аэропорта
 city         | text    | NOT NULL     | Город
 longitude    | float   | NOT NULL     | Координаты аэропорта: долгота
 latitude     | float   | NOT NULL     | Координаты аэропорта: широта
 timezone     | text    | NOT NULL     | Временная зона аэропорта
Индексы:
    PRIMARY KEY, btree (airport_code)
Ссылки извне:
    TABLE "flights" FOREIGN KEY (arrival_airport)
        REFERENCES airports(airport_code)
    TABLE "flights" FOREIGN KEY (departure_airport)
         REFERENCES airports(airport_code)
```

Рисунок 3 - Свойства таблицы bookings.airports

- **Таблица bookings.boarding_passes**

Таблица boarding_passes состоит из четырех столбцов:
- ticket_no
- flight_id
- boarding_no
- seat_no

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
-------------+------------+--------------+--------------------------
 ticket_no   | char(13)   | NOT NULL     | Номер билета
 flight_id   | integer    | NOT NULL     | Идентификатор рейса
 boarding_no | integer    | NOT NULL     | Номер посадочного талона
 seat_no     | varchar(4) | NOT NULL     | Номер места
Индексы:
    PRIMARY KEY, btree (ticket_no, flight_id)
    UNIQUE CONSTRAINT, btree (flight_id, boarding_no)
    UNIQUE CONSTRAINT, btree (flight_id, seat_no)
Ограничения внешнего ключа:
    FOREIGN KEY (ticket_no, flight_id)
        REFERENCES ticket_flights(ticket_no, flight_id)
```

Рисунок 4 - Свойства таблицы bookings.boarding_passes

- **Таблица bookings.bookings**

Таблица bookings состоит из трех столбцов:
- book_ref
- book_date
- total_amount

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
--------------+---------------+--------------+---------------------------
 book_ref     | char(6)       | NOT NULL     | Номер бронирования
 book_date    | timestamptz   | NOT NULL     | Дата бронирования
 total_amount | numeric(10,2) | NOT NULL     | Полная сумма бронирования
Индексы:
    PRIMARY KEY, btree (book_ref)
Ссылки извне:
    TABLE "tickets" FOREIGN KEY (book_ref) REFERENCES bookings(book_ref)
```

Рисунок 5 - Свойства таблицы bookings.bookings

- **Таблица bookings.flights**

Таблица flights состоит из десяти столбцов:
- flight_id
- flight_no
- scheduled_departure
- scheduled_arrival
- departure_airport
- arrival_airport
- status
- aircraft_code
- actual_departure
- actual_arrival

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 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.

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
---------------------+-------------+--------------+----------------------------
 flight_id           | serial      | NOT NULL     | Идентификатор рейса
 flight_no           | char(6)     | NOT NULL     | Номер рейса
 scheduled_departure | timestamptz | NOT NULL     | Время вылета по расписанию
 scheduled_arrival   | timestamptz | NOT NULL     | Время прилёта по расписанию
 departure_airport   | char(3)     | NOT NULL     | Аэропорт отправления
 arrival_airport     | char(3)     | NOT NULL     | Аэропорт прибытия
 status              | varchar(20) | NOT NULL     | Статус рейса
 aircraft_code       | char(3)     | NOT NULL     | Код самолета, IATA
 actual_departure    | timestamptz |              | Фактическое время вылета
 actual_arrival      | timestamptz |              | Фактическое время прилёта
Индексы:
    PRIMARY KEY, btree (flight_id)
    UNIQUE CONSTRAINT, btree (flight_no, scheduled_departure)
Ограничения-проверки:
    CHECK (scheduled_arrival > scheduled_departure)
    CHECK ((actual_arrival IS NULL)
       OR  ((actual_departure IS NOT NULL AND actual_arrival IS NOT NULL)
            AND (actual_arrival > actual_departure)))
    CHECK (status IN ('On Time', 'Delayed', 'Departed',
                      'Arrived', 'Scheduled', 'Cancelled'))
Ограничения внешнего ключа:
    FOREIGN KEY (aircraft_code)
        REFERENCES aircrafts(aircraft_code)
    FOREIGN KEY (arrival_airport)
        REFERENCES airports(airport_code)
    FOREIGN KEY (departure_airport)
        REFERENCES airports(airport_code)
Ссылки извне:
    TABLE "ticket_flights" FOREIGN KEY (flight_id)
        REFERENCES flights(flight_id)
```

Рисунок 6 - Свойства таблицы bookings.flights

- **Таблица bookings.seats**

Таблица seats состоит из трех столбцов:
- aircraft_code
- seat_no
- fare_conditions

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
Индексы:
    PRIMARY KEY, btree (aircraft_code, seat_no)
Ограничения-проверки:
    CHECK (fare_conditions IN ('Economy', 'Comfort', 'Business'))
Ограничения внешнего ключа:
    FOREIGN KEY (aircraft_code)
        REFERENCES aircrafts(aircraft_code) ON DELETE CASCADE
```

Рисунок 7 - Свойства таблицы bookings.seats

- **Таблица bookings.ticket_flights**

Таблица ticket_flights состоит из четырех столбцов:
- ticket_no
- flight_id
- fare_conditions
- amount

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
Индексы:
    PRIMARY KEY, btree (ticket_no, flight_id)
Ограничения-проверки:
    CHECK (amount >= 0)
    CHECK (fare_conditions IN ('Economy', 'Comfort', 'Business'))
Ограничения внешнего ключа:
    FOREIGN KEY (flight_id) REFERENCES flights(flight_id)
    FOREIGN KEY (ticket_no) REFERENCES tickets(ticket_no)
Ссылки извне:
    TABLE "boarding_passes" FOREIGN KEY (ticket_no, flight_id)
        REFERENCES ticket_flights(ticket_no, flight_id)
```

Рисунок 8 - Свойства таблицы bookings.ticket_flights

- **Таблица bookings.tickets**

Таблица tickets состоит из пяти столбцов:
- ticket_no
- book_ref
- passenger_id
- passenger_name
- contact_data

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_date).

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
----------------+-------------+--------------+----------------------------
 ticket_no      | char(13)    | NOT NULL     | Номер билета
 book_ref       | char(6)     | NOT NULL     | Номер бронирования
 passenger_id   | varchar(20) | NOT NULL     | Идентификатор пассажира
 passenger_name | text        | NOT NULL     | Имя пассажира
 contact_data   | jsonb       |              | Контактные данные пассажира
Индексы:
    PRIMARY KEY, btree (ticket_no)
Ограничения внешнего ключа:
    FOREIGN KEY (book_ref) REFERENCES bookings(book_ref)
Ссылки извне:
    TABLE "ticket_flights" FOREIGN KEY (ticket_no) REFERENCES tickets(ticket_no)
```

Рисунок 9 - Свойства таблицы bookings.tickets

- **Представление bookings.flights_v**

Над таблицей flights создано представление flights_v, содержащее 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).

Столбец	Тип	Описание
flight_id	integer	Идентификатор рейса
flight_no	char(6)	Номер рейса
scheduled_departure	timestamptz	Время вылета по расписанию
scheduled_departure_local	timestamp	Время вылета по расписанию, местное время в пункте отправления
scheduled_arrival	timestamptz	Время прилёта по расписанию
scheduled_arrival_local	timestamp	Время прилёта по расписанию, местное время в пункте прибытия
scheduled_duration	interval	Планируемая продолжительность полета
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
status	varchar(20)	Статус рейса
aircraft_code	char(3)	Код самолета, IATA
actual_departure	timestamptz	Фактическое время вылета
actual_departure_local	timestamp	Фактическое время вылета, местное время в пункте отправления
actual_arrival	timestamptz	Фактическое время прилёта
actual_arrival_local	timestamp	Фактическое время прилёта, местное время в пункте прибытия
actual_duration	interval	Фактическая продолжительность полета

Рисунок 10 - Свойства представления "bookings.flights_v"

- **Материализованное представление bookings.routes**

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

Рисунок 11 - Свойства материализованного представления bookings.routes

- **Особенности функции bookings.now()**

Демонстрационная база содержит временной «срез» данных — так, как будто в некоторый момент была сделана резервная копия реальной системы. Например, если некоторый рейс имеет статус Departed, это означает, что в момент резервного копирования самолет вылетел и находился в воздухе.

Позиция «среза» сохранена в функции bookings.now(). Ей можно пользоваться в запросах там, где в обычной жизни использовалась бы функция now().

Кроме того, значение этой функции определяет версию демонстрационной базы данных. Актуальная версия на текущий момент — от 13.10.2016.

- **Бизнес задачи, которые можно решить, используя БД.**

1. Определить загруженность аэропортов. Найти аэропорты, откуда вылетало и прилетало наибольшее количество людей за день/месяц/год.

2. Определить аэропорты, где чаще всего происходят задержки и отмены рейсов.

3. Определить самые непопулярные рейсы, чтобы сделать по этим направлениям скидки.

4. Бонусы для клиентов. Найти клиентов, потративших на перелеты больше определенной суммы, и предложить им бонусы/скидки на следующие перелеты. Но это только для тех клиентов, которые не меняли свои данные (фамилию, паспорт и т.д.)

5. Список SQL запросов из приложения №2 с описанием логики их выполнения

```
-- 1. В каких городах больше одного аэропорта?

select city "Города", count(airport_code) "Количество аэропортов" -- выводим города и
количество аэропортов в них
from airports  -- получили данные по аэропортам
group by city  -- группируем по городам
having count(airport_code) > 1 -- после группировки находим города, где количество
аэропортов больше 1


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

select a.airport_name "Аэропорты"  -- вывели названия аэропортов
from airports a  -- объединили таблицы с аэропортами, полетами и самолетами с максимальной
дальностью перелета
      join flights f on a.airport_code = f.departure_airport
      join aircrafts a2 on f.aircraft_code = a2.aircraft_code
      join (select max(range) max_r  -- нашли максимальную дальность полета
            from aircrafts) t on a2."range" = t.max_r
group by a.airport_code  -- сгруппировали по кодам аэропортов
order by a.airport_name  -- отсортировали по названиям аэропортов


-- 3. Вывести 10 рейсов с максимальным временем задержки вылета

select flight_id "ID рейса", (f.actual_departure - f.scheduled_departure) as "Задержка
вылета" -- вывели рейсы и их время задержки вылета
from flights f  -- получили данные по полетам
where (f.actual_departure - f.scheduled_departure) is not null -- оставили рейсы, где была
задержка вылета
order by "Задержка вылета" desc  -- отсортировали рейсы по задержке вылета от большего к
меньшему
limit 10  -- оставили первые 10 рейсов
```

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

select distinct t.book_ref "Брони без посадочных талонов" -- вывели уникальные значения номеров бронирования
	from tickets t
	left join boarding_passes bp on t.ticket_no = bp.ticket_no
where bp.boarding_no is null -- оставили данные, где номер посадочного талона имеет значение null


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

with cte_1 as ( -- создали cte с объединенными и сгруппированными данными из таблиц рейсы, самолеты, места в самолетах и аэропорты
	select f.flight_id, f.departure_airport, ap.airport_name, f.actual_departure, a.aircraft_code, a.model,
		count(s.seat_no) seats -- посчитали общее количество мест в самолете
	from flights f
	join aircrafts a on f.aircraft_code = a.aircraft_code
	join seats s on a.aircraft_code = s.aircraft_code
	join airports ap on f.departure_airport = ap.airport_code
	group by 1, 2, 3, 4, 5 ),
cte_2 as ( -- создали cte с объединенными и сгруппированными данными из таблиц ticket_flights и посадочные талоны
	select tf.flight_id, count(bp.seat_no) b_seats -- выбрали id рейсов и посчитали количество занятых мест по каждому рейсу
	from ticket_flights tf
	join boarding_passes bp on tf.ticket_no = bp.ticket_no and tf.flight_id = bp.flight_id
	group by 1 )
select cte_1.airport_name "Аэропорт", cte_1.flight_id "Рейс", cte_1.model "Модель самолета", cte_1.actual_departure "Дата и время вылета",
	cte_2.b_seats "Количество пассажиров",
	sum(cte_2.b_seats) -- посчитали суммарное накопление пассажиров с помощью оконной функции
	over (partition by cte_1.departure_airport, cte_1.actual_departure::date order by cte_1.actual_departure) "Суммарное накопление пассажиров",
	(cte_1.seats - cte_2.b_seats) "Свободные места", -- посчитали количество свободных мест в рейсах
	((cte_1.seats - cte_2.b_seats)*100/cte_1.seats) "% св.мест от общего кол-ва мест" -- нашли % свободных мест от общего количества мест в самолете
from cte_1 -- объединили данные из двух cte
	join cte_2 on cte_2.flight_id = cte_1.flight_id


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

select a.model "Модель самолета", t.count_a "Количество перелетов",
	round((count_a * 100)/all_flights::numeric, 2) "% от общего кол-ва перелетов" -- рассчитали % перелетов от общего числа рейсов, округлив до 2 знаков после запятой
from (
	select f.flight_id, f.aircraft_code,
		count(f.flight_id) over (partition by f.aircraft_code) count_a, -- посчитали количество рейсов для каждого самолета
		count(f.flight_id) over () all_flights -- посчитали общее количество рейсов (все рейсы)
		from flights f ) t
```

```
join aircrafts a on t.aircraft_code = a.aircraft_code -- полученные в подзапросе данные из таблицы рейсы объединили с таблицей самолеты
group by 1, 2, 3 -- сгруппировали данные по модели самолетов, числу перелетов и % перелетов


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

with cte_1 as ( -- создали cte с данными по всем рейсам с местами эконом-класса и их стоимостью
	select f.flight_id, tf.fare_conditions, tf.amount, a.city
		from flights f
		join ticket_flights tf on f.flight_id = tf.flight_id
		join airports a on f.arrival_airport  = a.airport_code
	where tf.fare_conditions = 'Economy'
	group by 1, 2, 3, 4
	order by 1, 2 ),
cte_2 as ( -- создали cte с данными по всем рейсам с местами бизнес-класса и их стоимостью
	select f.flight_id, tf.fare_conditions, tf.amount, a.city
		from flights f
		join ticket_flights tf on f.flight_id = tf.flight_id
		join airports a on f.arrival_airport  = a.airport_code
	where tf.fare_conditions = 'Business'
	group by 1, 2, 3, 4
	order by 1, 2 )
select cte_2.city "Город прибытия", cte_1.flight_id "Рейс", cte_1.amount "Цена эконом-класса", cte_2.amount "Цена бизнес-класса"
from cte_1
	join cte_2 on cte_1.flight_id = cte_2.flight_id -- объединили cte_1 и cte_2
where cte_2.amount < cte_1.amount -- с условием, что стоимость бизнес-класса меньше стоимости эконом-класса


-- 8. Между какими городами нет прямых рейсов?

create view flights_cities as  -- создали представление, которое находит по всем рейсам город отправления самолета и город прибытия самолета
select a.city "Город отправления", a1.city "Город прибытия"
	from flights f
	join airports a on f.departure_airport = a.airport_code
	join airports a1 on f.arrival_airport = a1.airport_code
	group by 1, 2 -- сгруппировали данные по городам, чтобы убрать повторы

select a.city "Город отправления", a1.city "Город прибытия"
	from airports a, airports a1 -- из городов таблицы аэропорты сдалали декартово произведение
	where a.city != a1.city -- условие, что город вылета не равен городу прилета
except -- из всех вариантов перелетов между городами исключили (вычли) существующие перелеты
select "Город отправления", "Город прибытия"
	from flights_cities
order by 1  -- отсортировали по алфавиту города отправления для удобства восприятия


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

select model "Модель самолета", range "Max дальность полета", name1 "Отправление", name2 "Прибытие", L "Дальность полета (км)",
	case
		when "range" > L then 'Долетит' -- если максимальная дальность полета самолета больше дальности его перелета в рейсе, то самолет долетит
```

```
            else 'Не долетит'  -- в остальных случаях - не долетит
      end "Долетит/Не долетит"
from (
      select a.model, a.range, name1, name2,
                  round(((acos(sind(lat1) * sind(lat2) + cosd(lat1) * cosd(lat2) *
cosd(lon1 - lon2))) * 6371)::numeric, 3) as L -- рассчитали расстояние между аэропортом
вылета и аэропортом прилета по каждому рейсу и округлили до тысячных (до метров)
      from (
            select f.aircraft_code, f.departure_airport, a1.airport_name name1,
            a1.longitude::double precision lon1, a1.latitude::double precision lat1,
            f.arrival_airport, a2.airport_name name2, a2.longitude::double precision lon2,
            a2.latitude::double precision lat2 -- все данные по широте и долготе
преобразовали в тип double precision
            from flights f -- обогатили таблицу рейсы данными из таблицы аэропорты, чтобы
получить названия аэропортов
            join airports a1 on f.departure_airport = a1.airport_code
            join airports a2 on f.arrival_airport = a2.airport_code
            group by 1, 2, 3, 4, 5, 6, 7, 8, 9 ) t  -- сгруппировали данные
            join aircrafts a on t.aircraft_code = a.aircraft_code -- объединили обогащенную
таблицу рейсы с таблицей самолеты
      order by 1) t1
```