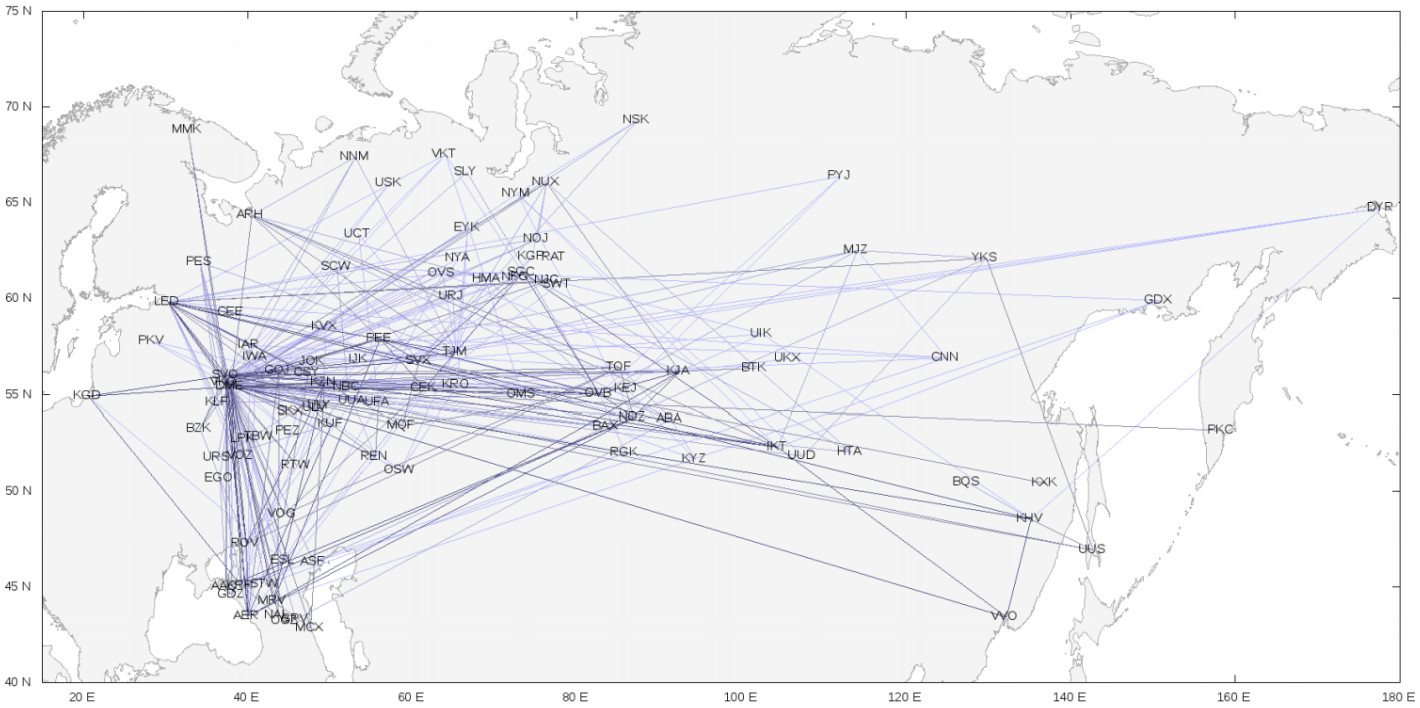
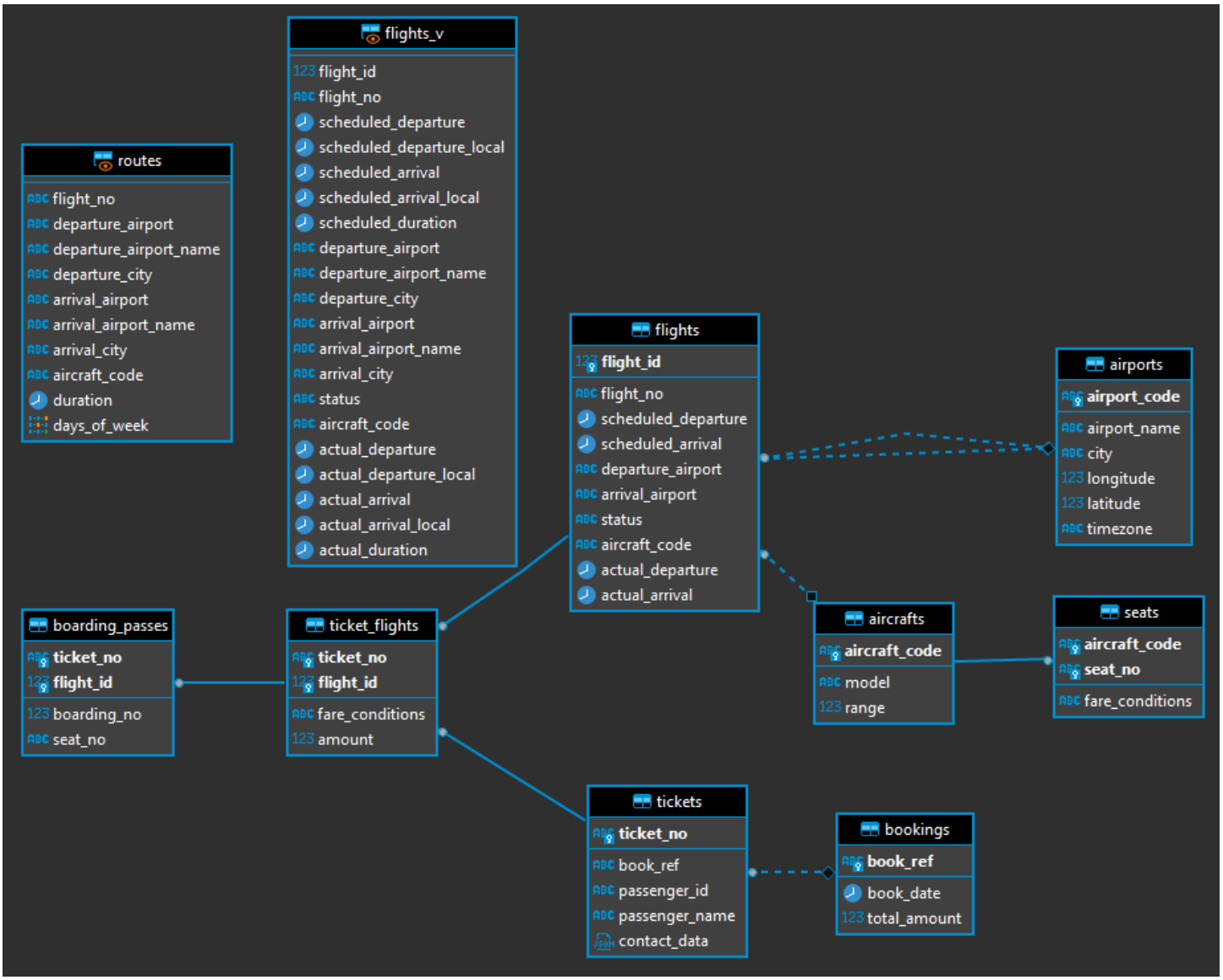
SQL и получение данных

В качестве предметной области выбраны авиаперевозки по России



1)База была развернута из *.backup файла

2)ER-диаграмма



Описание схемы

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

Билет имеет уникальный номер и содержит информацию о пассажире. 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.

Билет включает один или несколько перелетов (ticket_flights). Несколько перелетов могут включаться в билет в случаях, когда нет нет прямого рейса, соединяющего пункты отправления и назначения (полет с пересадками), либо когда билет взят «туда и обратно».

Каждый рейс (flights) следует из одного аэропорта (airports) в другой. Рейсы с одним

номером имеют одинаковые пункты вылета и назначения, но будут отличаться датой отправления.

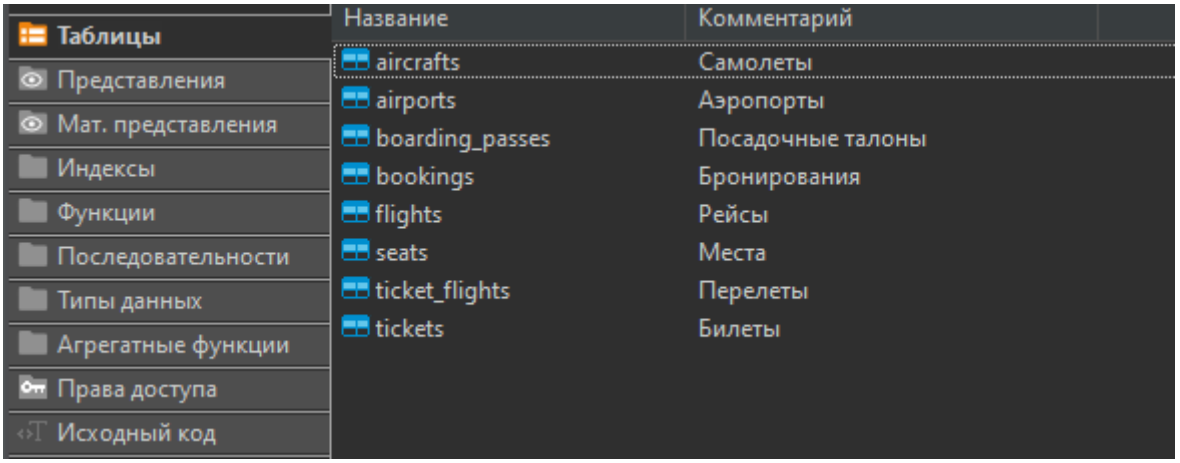
При регистрации на рейс пассажиру выдается посадочный талон (boarding_passes), в котором указано место в самолете. Пассажир может зарегистрироваться только на тот рейс, который есть у него в билете. Комбинация рейса и места в самолете должна быть уникальной, чтобы не допустить выдачу двух посадочных 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.

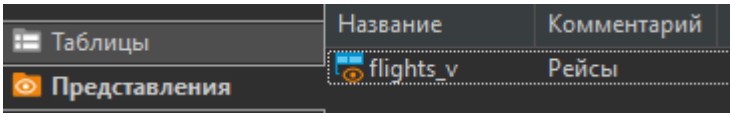
3)Краткое описание БД

Схема данных состоит из восьми таблиц и нескольких представлений.

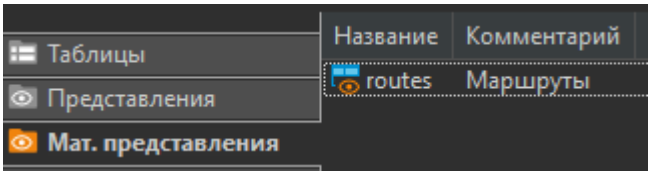
Отношения



Представления



Мат. представления



4)Развернутый анализ БД:

Описание таблиц

Самолеты bookings.aircrafts

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
```

Аэропорты bookings.airports

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)
```

Посадочные талоны bookings.boarding_passes

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)
```

Бронирования bookings.bookings

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)
```

Рейсы bookings.flights

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.

```
       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)
```

Места bookings.seats

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
```

Таблица связей bookings.ticket_flights

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)
```

Билеты bookings.tickets

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)
```

Представление "bookings.flights_v"

Над таблицей flights создано представление flights_v, содержащее дополнительную информацию:

```
         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
```

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.

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

Примеры бизнес задач, которые можно решить используя БД

1)Определять % заполняемости самолетов по рейсам.
Можно использовать, чтобы подобрать оптимальный самолет для рейса, увеличить % заполняемости, управлять ценой на билеты и предлагать акции на свободные места.

2)Определять выручку за рейс
Помогает управлять экономикой перевозок

3)Искать ближайшие рейсы, вылетающие по нужному направлению.
Информацию по конкретным рейсам и свободных местах можно использовать для продажи билетов через рекламу.

4)Определять какие самолеты можно ставить на какие рейсы
Самолеты не всегда летают по одному рейсу, иногда есть необходимость послать самолет из конкретного города в другой, и надо быть уверенным что максимальная дальность полета позволяет это сделать.

5)Определение самого выгодного предложения по перелету в рамках рейса по классам.
Иногда продать бизнес класс дешевле эконом класса выгоднее. Можно использовать в рекламе или как способ повышения лояльности для постоянных клиентов.

6)Узнавать информацию о текущих бронях на будущие рейсы
Помогает заранее заполнить самолеты

5)Список SQL запросов

Вопросы на которые надо ответить используя SQL запросы

1. В каких городах больше одного аэропорта?

```sql
--1)В каких городах больше одного аэропорта?
 SELECT city, count_airport
 FROM (SELECT
     a.city
     ,count(a.airport_code) AS count_airport
     FROM airports a
     GROUP BY a.city ) AS t
 WHERE count_airport > 1
```

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

```sql
-- 2)В каких аэропортах есть рейсы, выполняемые самолетом с максимальной дальностью перелета?
SELECT
    f.departure_airport AS "Аэропорт вылета"
FROM flights f
WHERE f.aircraft_code IN (SELECT
        aircraft_code
    FROM (SELECT
    aircraft_code
    ,model
    ,"range"
    ,ROW_NUMBER () OVER ()
    FROM aircrafts
    ORDER BY "range" DESC) AS t
    WHERE ROW_NUMBER = 1)
GROUP BY
    f.departure_airport
```

3. Вывести 10 рейсов с максимальным временем задержки вылета

```sql
-- 3) Вывести 10 рейсов с максимальным временем задержки вылета
SELECT
--  f.flight_id AS "ID Полета"
    f.flight_no AS "Номер рейса"
--  ,f.scheduled_departure AS "Запланированный отъезд"
--  ,f.actual_departure AS "Фактический отъезд"
    ,age(f.actual_departure, f.scheduled_departure) AS "Время задержки вылета"
FROM flights f
WHERE f.actual_departure IS NOT NULL
ORDER BY age(f.actual_departure, f.scheduled_departure) DESC
LIMIT 10
```

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

```sql
-- 4)Были ли брони, по которым не были получены посадочные талоны?
SELECT
    t.book_ref
    ,count(bp.boarding_no) AS count_boarding_no
FROM tickets t
LEFT JOIN boarding_passes bp ON bp.ticket_no = t.ticket_no
GROUP BY book_ref
HAVING count(bp.boarding_no) = 0
```

5. Найдите свободные места 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 за день.

```sql
-- 5) Найдите свободные места 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 за день.
SELECT
    f.flight_no
    ,f.aircraft_code
    ,f.actual_departure
    ,f.departure_airport
    ,t.all_seat_aircraft - t2.booked_seat AS free_seat
    ,round((t.all_seat_aircraft - t2.booked_seat)::float / t.all_seat_aircraft * 100) AS "%_free_seat"
    ,sum(t2.booked_seat) OVER (PARTITION BY f.departure_airport, f.actual_departure::date
    ORDER BY f.actual_departure, f.aircraft_code) AS sum_pas
FROM flights f
LEFT JOIN (SELECT
    aircraft_code
    ,count(seat_no) AS all_seat_aircraft
    FROM seats s
    GROUP BY aircraft_code) AS t ON t.aircraft_code = f.aircraft_code
LEFT JOIN (SELECT
    flight_id
    ,count(seat_no) AS booked_seat
    FROM boarding_passes bp
    GROUP BY flight_id ) AS t2 ON t2.flight_id = f.flight_id
WHERE t2.booked_seat IS NOT NULL AND f.actual_departure IS NOT NULL
```

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

```sql
--6) Найдите процентное соотношение перелетов по типам самолетов от общего количества.
SELECT
    aircraft_code
    ,count_flight
--  ,sum(count_flight) OVER () AS sum_fligth
    ,round(count_flight / sum(count_flight) OVER () * 100, 2) AS "% от всех полетов"
FROM (SELECT
        f.aircraft_code
        ,count(f.flight_id) AS count_flight
    FROM flights f
    GROUP BY aircraft_code ) AS t
```

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

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

-- Логика решения:
-- 7.1) Создаем 2 CTE с группировкой по flight_id, fare_conditions
-- и агрегацией max(amount) для Economy и min(amount) для Business
-- 7.2) Выводим в SELECT данные из CTE economy
-- 7.3) Добавляем CTE business по flight_id
-- 7.4) Обогащаем данными из flights, airports
-- 7.5) Фильтруем строки по условию что бизнес - классом дешевле, чем эконом-классом
WITH economy AS (
SELECT
    flight_id
    ,fare_conditions
    ,max(amount) AS max_economy_flight
FROM ticket_flights tf
WHERE fare_conditions = 'Economy'
GROUP BY
    flight_id
    ,fare_conditions
ORDER BY flight_id
), business AS (
SELECT
    flight_id
    ,fare_conditions
    ,min(amount) AS min_business_flight
FROM ticket_flights tf
WHERE fare_conditions = 'Business'
GROUP BY
    flight_id
    ,fare_conditions
ORDER BY flight_id
)
SELECT
    economy.flight_id
    ,economy.max_economy_flight
    ,bu.min_business_flight
    ,bu.min_business_flight < economy.max_economy_flight AS business_cheaper
--  ,fl.departure_airport
    ,ai.city AS city_from
--  ,fl.arrival_airport
    ,ai2.city AS city_to
FROM economy
INNER JOIN business bu ON bu.flight_id = economy.flight_id
LEFT JOIN flights fl ON fl.flight_id = economy.flight_id
LEFT JOIN airports ai ON ai.airport_code = fl.departure_airport
LEFT JOIN airports ai2 ON ai2.airport_code = fl.arrival_airport
WHERE bu.min_business_flight < economy.max_economy_flight
```

8. Между какими городами нет прямых рейсов?

```sql
-- 8) Между какими городами нет прямых рейсов?

--Логика решения:
-- Чтобы понять между какими городами нет прямых рейсов, надо:
-- 1)Получить список перелетов между городами
-- 2)Надо получить все возможные сценарии полетов для каждого города
-- 3)Вычесть из списка № 2  список № 1 текущие полеты для этого города
-- и останутся только те куда не было полетов (EXCEPT)

-- 8.1)Получить текущих между какими городами есть перелеты.
SELECT
    a.city AS departure_city
    ,a2.city AS arrival_city
FROM flights f
LEFT JOIN airports a ON a.airport_code = f.departure_airport
LEFT JOIN airports a2 ON a2.airport_code = f.arrival_airport
GROUP BY a.city, a2.city
ORDER BY departure_city, arrival_city

-- 8.2) Уникальный список все возможных сценариев городов для перелетво из таблицы airports
WITH unique_city AS (
SELECT DISTINCT
    a.city
FROM airports a
)
SELECT
    uc.city
    ,uc2.city
FROM unique_city uc, unique_city uc2
WHERE uc.city != uc2.city
ORDER BY uc.city, uc2.city

-- 8.3) Вычесть из списка № 2  список № 1 текущие полеты для этого города
-- и останутся только те куда не было полетов
WITH unique_city AS (
SELECT DISTINCT a.city
FROM airports a)
SELECT
    uc.city
    ,uc2.city
FROM unique_city uc, unique_city uc2
WHERE uc.city != uc2.city
EXCEPT
SELECT
    a.city AS departure_city
    ,a2.city AS arrival_city
FROM flights f
LEFT JOIN airports a ON a.airport_code = f.departure_airport
LEFT JOIN airports a2 ON a2.airport_code = f.arrival_airport
GROUP BY a.city, a2.city
ORDER BY 1, 2
```

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

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

-- Логика решения:
-- 9.1) Берем таблицу flights и обогащаем городами, также в таблице airports находятся координаты аэропортов
-- 9.2) Расчитываем d — расстояние между пунктами, измеряется в радианах длиной дуги большого круга земного шара.

-- Кратчайшее расстояние между двумя точками A и B на земной поверхности (если принять ее за сферу) определяется зависимостью:
-- d = arccos {sin(latitude_a)·sin(latitude_b) + cos(latitude_a)·cos(latitude_b)·cos(longitude_a - longitude_b)}
-- 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.

-- 9.3) Определяем расстояние в км
-- Расстояние между пунктами, измеряемое в километрах, определяется по формуле:
-- L = d·R, где R = 6371 км — средний радиус земного шара.
-- 9.4) Сравниваем с допустимой максимальной дальностью перелетов в самолетах через CASE
```

```
SELECT DISTINCT
    f.departure_airport
    ,f.arrival_airport
    ,a.city
    ,a2.city
    ,f.aircraft_code
    ,a3."range"
    ,round(acos(sind(a.latitude) * sind(a2.latitude) + cosd(a.latitude) *
    cosd(a2.latitude) * cosd(a.longitude - a2.longitude)) * 6371) AS distance_km
    ,CASE
        WHEN a3."range" < round(acos(sind(a.latitude) * sind(a2.latitude) +
        cosd(a.latitude) * cosd(a2.latitude) * cosd(a.longitude - a2.longitude)) * 6371) THEN 'We have a problem'
        WHEN a3."range" > round(acos(sind(a.latitude) * sind(a2.latitude)
        + cosd(a.latitude) * cosd(a2.latitude) * cosd(a.longitude - a2.longitude)) * 6371) THEN 'Good'
    END AS will_fly_or_not
FROM flights f
LEFT JOIN airports a ON a.airport_code = f.departure_airport
LEFT JOIN airports a2 ON a2.airport_code = f.arrival_airport
LEFT JOIN aircrafts a3 ON a3.aircraft_code = f.aircraft_code
ORDER BY
    f.departure_airport
    ,f.arrival_airport
```