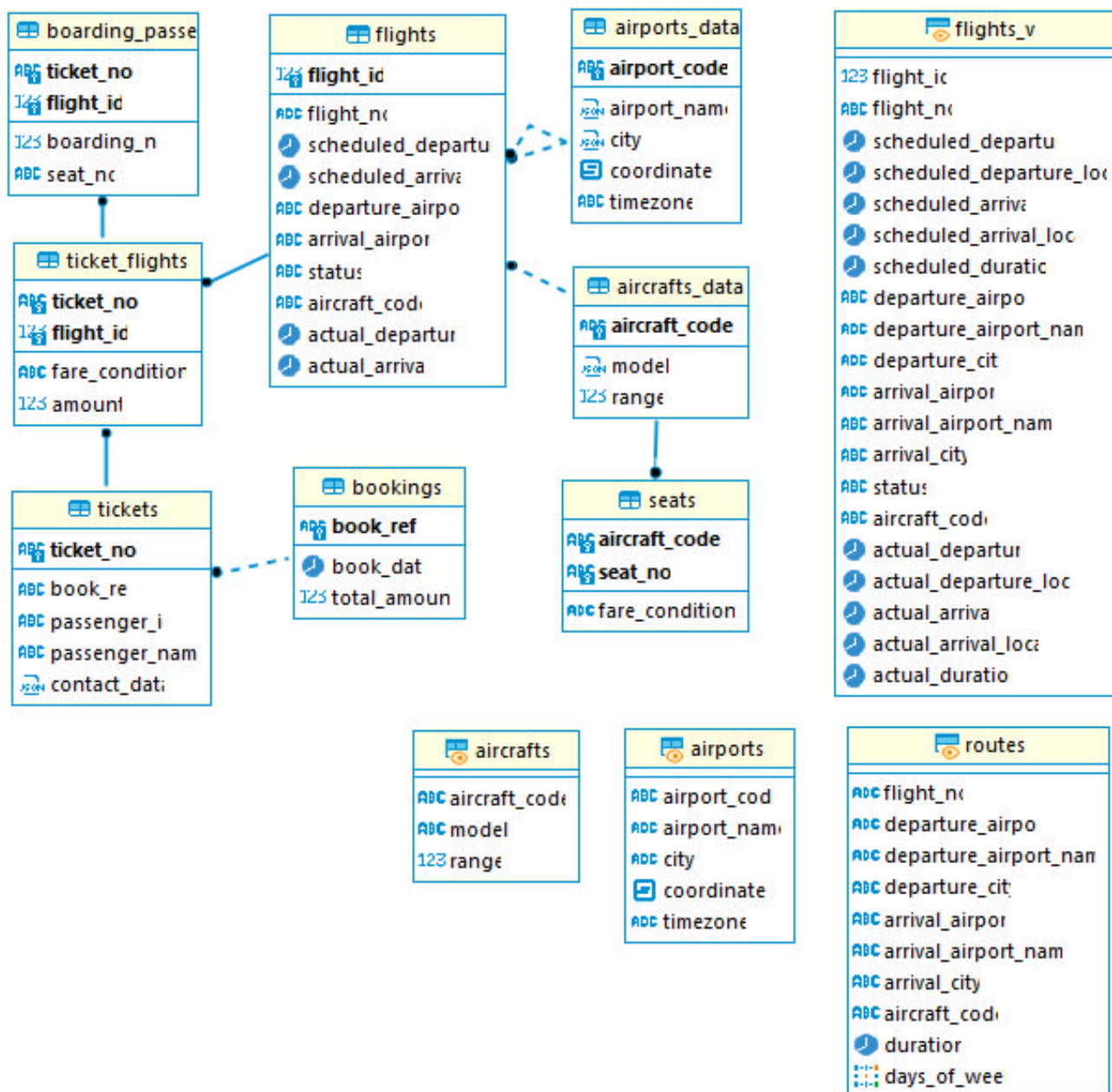




Итоговая работа: “SQL и получение данных”.

В работе использовался облачный тип подключения.

ER-диаграмма из DBeaver в соответствии с типом подключения:



Краткое описание БД.

База данных состоит из 8 таблиц:

1. `aircrafts_data` (самолеты) – код самолета, модель самолета, максимальная дальность полёта (км).
2. `airports_data` (аэропорты) – код аэропорта, название аэропорта, город, координаты (долгота/широта в градусах), временная зона аэропорта.
3. `boarding_passes` (посадочные талоны) – номер билета, идентификатор рейса, номер посадочного, номер места.
4. `bookings` (бронирование) – номер бронирования, дата бронирования, полная сумма бронирования.
5. `flights` (рейсы) – идентификатор рейса, номер рейса, время вылета и прилета по расписанию, аэропорты отправления и прибытия, статус рейса, код самолета, фактическое время вылета и прилета.
6. `seats` (места) – код самолета, номер места, класс обслуживания.
7. `ticket_flights` (связь билеты_рейсы) – номер билета, идентификатор рейса, класс обслуживания, стоимость перелета.
8. `tickets` (билеты) – номер билета, номер бронирования, идентификатор пассажира, ФИ пассажира, контактные данные пассажира.

Также в базе данных 4 представления:

1. `flights_v` – идентификатор рейса, номер рейса, время вылета по расписанию + местное, время прилета по расписанию + местное, планируемая продолжительность полета, код аэропорта отправления, название аэропорта отправления, город отправления, код аэропорта прибытия, название аэропорта прибытия, город прибытия, статус рейса, код самолета, фактическое время вылета + местное, фактическое время прилета + местное, фактическая продолжительность полета.
2. `routes` – номер рейса, код аэропорта отправления, название аэропорта отправления, город отправления, код аэропорта прибытия, название аэропорта прибытия, город прибытия, код самолёта, продолжительность полета, дни недели, когда выполняется рейс.
3. `aircrafts` – код самолета, модель самолета, максимальная дальность полёта (км).
4. `airports` – код аэропорта, название аэропорта, город, координаты (долгота/широта в градусах), временная зона аэропорта.

Развернутый анализ БД.

Описание БД.

В качестве предметной области в демонстрационной базе данных для СУБД PostgreSQL выбраны авиаперевозки по России.

База содержит временной «срез» данных — так, как будто в некоторый момент была сделана резервная копия реальной системы.

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; для простоты можно считать, что все пассажиры уникальны.

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

В схеме данных нет жесткого ограничения, но предполагается, что все билеты в одном бронировании имеют одинаковый набор перелетов. 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. 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).

Описание таблиц.

Таблица aircrafts data.

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

Первичный ключ - aircraft_code, связана с таблицами flights и seats по ключу aircraft_code.

```
aircraft_code bpchar(3) NOT NULL,  
model jsonb NOT NULL,  
"range" int4 NOT NULL,  
CONSTRAINT aircrafts_pkey PRIMARY KEY (aircraft_code),  
CONSTRAINT aircrafts_range_check CHECK ((range > 0))
```

Таблица airports data.

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).

Первичный ключ - airport_code, связана с таблицей flights по ключам arrival_airport и departure_airport.

```
airport_code bpchar(3) NOT NULL,  
airport_name jsonb NOT NULL,  
city jsonb NOT NULL,  
coordinates point NOT NULL,  
timezone text NOT NULL,  
CONSTRAINT airports_data_pkey PRIMARY KEY (airport_code)
```

Таблица 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.

Первичный ключ - book_ref, связана с таблицей tickets по ключу book_ref.

```
book_ref bpchar(6) NOT NULL,  
book_date timestampz NOT NULL,  
total_amount numeric(10, 2) NOT NULL,  
CONSTRAINT bookings_pkey PRIMARY KEY (book_ref)
```

Таблица 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).

Первичный ключ составной - ticket_no + flight_id, связана с таблицей ticket_flights по ключу ticket_no + flight_id.

```
ticket_no bpchar(13) NOT NULL,  
flight_id int4 NOT NULL,  
boarding_no int4 NOT NULL,  
seat_no varchar(4) NOT NULL,  
CONSTRAINT boarding_passes_flight_id_boarding_no_key  
UNIQUE (flight_id, boarding_no),  
CONSTRAINT boarding_passes_flight_id_seat_no_key  
UNIQUE (flight_id, seat_no),  
CONSTRAINT boarding_passes_pkey  
PRIMARY KEY (ticket_no, flight_id),  
CONSTRAINT boarding_passes_ticket_no_fkey  
FOREIGN KEY (ticket_no, flight_id)  
REFERENCES bookings.ticket_flights(ticket_no, flight_id)
```

Таблица 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.

Первичный ключ составной - aircraft_code, seat_no, связана с таблицей aircrafts по ключу aircraft_code.

```
aircraft_code bpchar(3) NOT NULL,  
seat_no varchar(4) NOT NULL,  
fare_conditions varchar(10) NOT NULL,  
CONSTRAINT seats_fare_conditions_check  
CHECK (((fare_conditions)::text = ANY(ARRAY[  
('Economy'::character_varying)::text,  
('Comfort'::character_varying)::text,  
('Business'::character_varying)::text]))),  
CONSTRAINT seats_pkey PRIMARY KEY (aircraft_code, seat_no)
```


Таблица 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.

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. 6
- Arrived Самолет прибыл в пункт назначения.
- Cancelled Рейс отменен.

Первичный ключ - flight_id, связана с таблицами: - aircrafts по ключу aircraft_code; - airports по ключам arrival_airport и departure_airport; - ticket_flights по ключу flight_id.

```
CREATE TABLE bookings.flights (  
    flight_id serial4 NOT NULL,  
    flight_no bpchar(6) NOT NULL,  
    scheduled_departure timestampz NOT NULL,  
    scheduled_arrival timestampz NOT NULL,  
    departure_airport bpchar(3) NOT NULL,  
    arrival_airport bpchar(3) NOT NULL,  
    status varchar(20) NOT NULL,  
    aircraft_code bpchar(3) NOT NULL,  
    actual_departure timestampz NULL,  
    actual_arrival timestampz NULL,  
    CONSTRAINT flights_check  
        CHECK ((scheduled_arrival > scheduled_departure)),  
    CONSTRAINT flights_check1 CHECK (((actual_arrival IS NULL)  
        OR ((actual_departure IS NOT NULL)  
        AND (actual_arrival IS NOT NULL)  
        AND (actual_arrival > actual_departure)))),  
    CONSTRAINT flights_flight_no_scheduled_departure_key  
        UNIQUE (flight_no, scheduled_departure),  
    CONSTRAINT flights_pkey PRIMARY KEY (flight_id),  
    CONSTRAINT flights_status_check CHECK (((status)::text =  
        ANY (ARRAY[('On Time'::character varying)::text,  
        ('Delayed'::character varying)::text,  
        ('Departed'::character varying)::text,  
        ('Arrived'::character varying)::text,  
        ('Scheduled'::character varying)::text,  
        ('Cancelled'::character varying)::text]))),  
    CONSTRAINT flights_aircraft_code_fkey FOREIGN KEY (aircraft_code)  
        REFERENCES bookings.aircrafts_data(aircraft_code),  
    CONSTRAINT flights_arrival_airport_fkey FOREIGN KEY (arrival_airport)  
        REFERENCES bookings.airports_data(airport_code),  
    CONSTRAINT flights_departure_airport_fkey FOREIGN KEY (departure_airport)  
        REFERENCES bookings.airports_data(airport_code)
```

Таблица 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_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.

Первичный ключ - ticket_no, связана с таблицами ticket_flights по ключу ticket_no и bookings по ключу book_ref.

```
ticket_no bpchar(13) NOT NULL,  
book_ref bpchar(6) NOT NULL,  
passenger_id varchar(20) NOT NULL,  
passenger_name text NOT NULL,  
contact_data jsonb NULL,  
CONSTRAINT tickets_pkey PRIMARY KEY (ticket_no),  
CONSTRAINT tickets_book_ref_fkey FOREIGN KEY (book_ref)  
REFERENCES bookings.bookings(book_ref)
```

Таблица ticket_flights.

Перелет соединяет билет с рейсом и идентифицируется их номерами. Для каждого перелета указываются его стоимость (amount) и класс обслуживания (fare_conditions).

Первичный ключ составной - ticket_no + flight_id, связана с таблицами: - boarding_passes по ключу ticket_no + flight_id; - flights по ключу flight_id; - tickets по ключу ticket_no

```
ticket_no bpchar(13) NOT NULL,  
flight_id int4 NOT NULL,  
fare_conditions varchar(10) NOT NULL,  
amount numeric(10, 2) NOT NULL,  
CONSTRAINT ticket_flights_amount_check  
CHECK ((amount >= (0)::numeric)),  
CONSTRAINT ticket_flights_fare_conditions_check  
CHECK (((fare_conditions)::text = ANY (ARRAY[  
( 'Economy'::character varying)::text,  
( 'Comfort'::character varying)::text,  
( 'Business'::character varying)::text ]))),  
CONSTRAINT ticket_flights_pkey  
PRIMARY KEY (ticket_no, flight_id),  
CONSTRAINT ticket_flights_flight_id_fkey  
FOREIGN KEY (flight_id)  
REFERENCES bookings.flights(flight_id),  
CONSTRAINT ticket_flights_ticket_no_fkey  
FOREIGN KEY (ticket_no)  
REFERENCES bookings.tickets(ticket_no)
```


Описание представлений.

Представление 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).

| Название | # | Тип данных | Not Null | Комментарий |
|---------------------------|----|-------------|----------|--|
| 123 flight_id | 1 | int4 | [] | Flight ID |
| ABC flight_no | 2 | bpchar(6) | [] | Flight number |
| 🕒 scheduled_departure | 3 | timestamptz | [] | Scheduled departure time |
| 🕒 scheduled_departu... | 4 | timestamp | [] | Scheduled departure time, local time at the point of departure |
| 🕒 scheduled_arrival | 5 | timestamptz | [] | Scheduled arrival time |
| 🕒 scheduled_arrival_l... | 6 | timestamp | [] | Scheduled arrival time, local time at the point of destination |
| 🕒 scheduled_duration | 7 | interval | [] | Scheduled flight duration |
| ABC departure_airport | 8 | bpchar(3) | [] | Deprature airport code |
| ABC departure_airport_... | 9 | text | [] | Departure airport name |
| ABC departure_city | 10 | text | [] | City of departure |
| ABC arrival_airport | 11 | bpchar(3) | [] | Arrival airport code |
| ABC arrival_airport_name | 12 | text | [] | Arrival airport name |
| ABC arrival_city | 13 | text | [] | City of arrival |
| ABC status | 14 | varchar(20) | [] | Flight status |
| ABC aircraft_code | 15 | bpchar(3) | [] | Aircraft code, IATA |
| 🕒 actual_departure | 16 | timestamptz | [] | Actual departure time |
| 🕒 actual_departure_l... | 17 | timestamp | [] | Actual departure time, local time at the point of departure |
| 🕒 actual_arrival | 18 | timestamptz | [] | Actual arrival time |
| 🕒 actual_arrival_local | 19 | timestamp | [] | Actual arrival time, local time at the point of destination |
| 🕒 actual_duration | 20 | interval | [] | Actual flight duration |




Представление 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 представление routes.

| Название | # | Тип данных | Not Null | Комментарий |
|--|----|------------|----------|---|
|  flight_no | 1 | bpchar(6) | [] | Flight number |
|  departure_airport | 2 | bpchar(3) | [] | Code of airport of departure |
|  departure_airport_name | 3 | text | [] | Name of airport of departure |
|  departure_city | 4 | text | [] | City of departure |
|  arrival_airport | 5 | bpchar(3) | [] | Code of airport of arrival |
|  arrival_airport_name | 6 | text | [] | Name of airport of arrival |
|  arrival_city | 7 | text | [] | City of arrival |
|  aircraft_code | 8 | bpchar(3) | [] | Aircraft code, IATA |
|  duration | 9 | interval | [] | Scheduled duration of flight |
|  days_of_week | 10 | _int4 | [] | Days of week on which flights are scheduled |

Представление aircrafts.

В таблице с самолетами столбец (model) содержит в себе данные в формате json. В представлении aircrafts данные столбца (model) приведены к текстовому формату на русском языке.

| Название | # | Тип данных | Not Null | Комментарий |
|---|---|------------|----------|-----------------------------|
|  aircraft_code | 1 | bpchar(3) | [] | Aircraft code, IATA |
|  model | 2 | text | [] | Aircraft model |
|  range | 3 | int4 | [] | Maximal flying distance, km |

Представление airports.

В таблице с аэропортами столбцы (airport_name) и (city) также содержат в себе данные в формате json. В представлении **airports** данные приведены к текстовому формату на русском языке.

| Название | # | Тип данных | Not Null | Комментарий |
|--|---|------------|----------|--|
|  airport_code | 1 | bpchar(3) | [] | Airport code |
|  airport_name | 2 | text | [] | Airport name |
|  city | 3 | text | [] | City |
|  coordinates | 4 | point | [] | Airport coordinates (longitude and latitude) |
|  timezone | 5 | text | [] | Airport time zone |

Бизнес-задачи, которые можно решить, используя БД.

При помощи БД можно решать различные бизнес-задачи по улучшению качества обслуживания и лояльности клиентов, оптимизации расходов и устранению возможных проблем и трудностей.

К примеру, можно создавать и отслеживать бронирования перелетов, проверять заполняемость самолетов, выявлять наиболее популярные маршруты. Также проверять соответствие самолетов требованиям к рейсу (дальность, наличие класса обслуживания и т.д.). Анализировать время задержки рейсов с учетом аэропорта, рейса, времени суток и т.д., сравнить стоимости перелетов в разные города и т.д.