Viktor Lokhanko

October 15, 2021

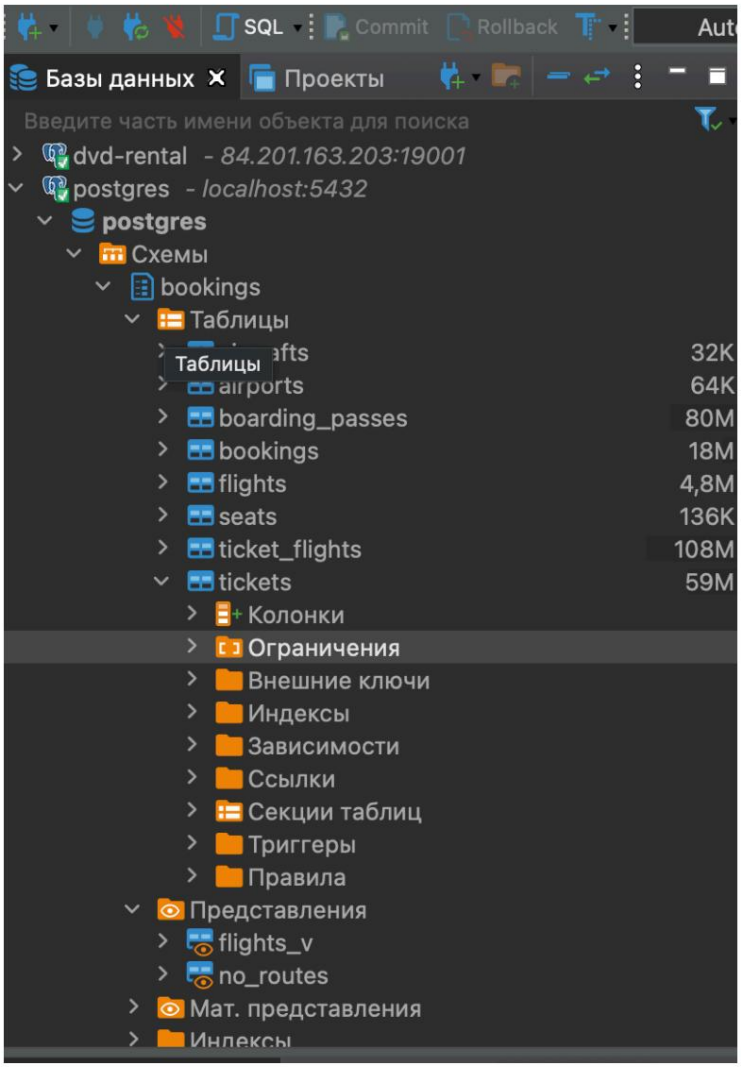
netology

Project work on the module “SQL and Data Retrieval”

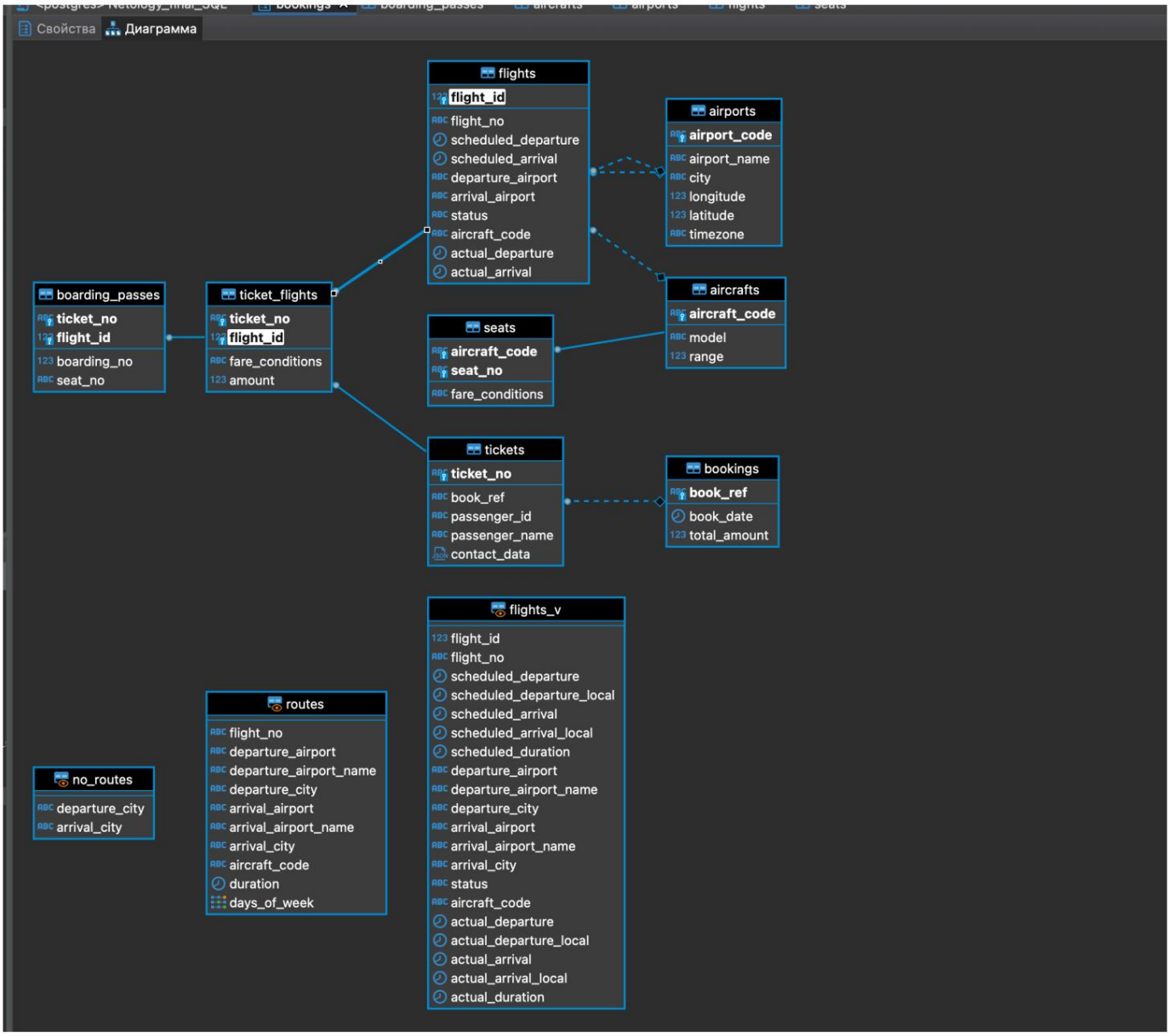
air transportation

Final work

1. The local connection type was used in the work.



2. Screenshot of ER diagram from DBeaver according to local connection



3. Brief description of the database. Tables and Views

tables

aircrafts: IATA aircraft code, aircraft model, maximum flight range, km

airports: airport code, airport name, city, airport coordinates: longitude,
airport coordinates: latitude, airport coordinates: latitude

boarding_passes: ticket number, flight id, boarding pass number,
place number

bookings: booking number, booking date, total booking amount

flights: flight id, flight number, scheduled departure time, scheduled arrival time, departure airport, arrival airport,
flight status, IATA aircraft code, actual departure time, actual arrival time

seats: IATA aircraft code, seat number, class of service

ticket_flights: ticket number, flight id, class of service, cost
flight

tickets: ticket number, booking number, passenger ID, name
passenger, passenger contact details

Representation

flights_v view: flight id, flight number, departure time
scheduled departure time, scheduled departure time, local time at departure point, scheduled arrival time,
scheduled arrival time, local time at destination point, planned flight duration, airport code

departure, departure airport name, departure city, arrival airport code, arrival airport name, arrival city, flight status,
IATA aircraft code, actual departure time, actual departure time, local time at origin, actual arrival time, actual arrival
time, local time at point of arrival, actual flight duration

Books.routes materialized view: flight number, airport code
departure, departure airport name, departure city, arrival airport code, arrival airport name, arrival city, IATA aircraft
code, flight duration, days of the week when flights are operated

4. Detailed analysis of the database - description of tables, logic, relationships and business areas

tables

aircrafts

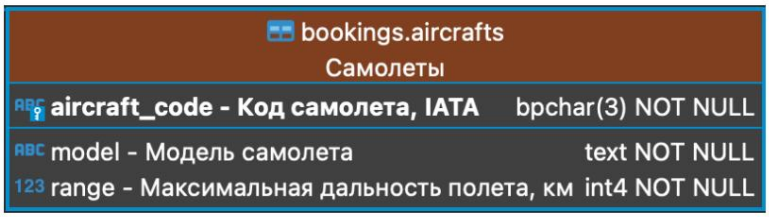
Each aircraft model is identified by its three-digit code *(aircraft_code).* The name of the model *(model)* and the maximum flight range in kilometers (range).

Indices:

PRIMARY KEY, btree (aircraft_code)

check-limits:

CHECK (range > 0)



Links from outside:

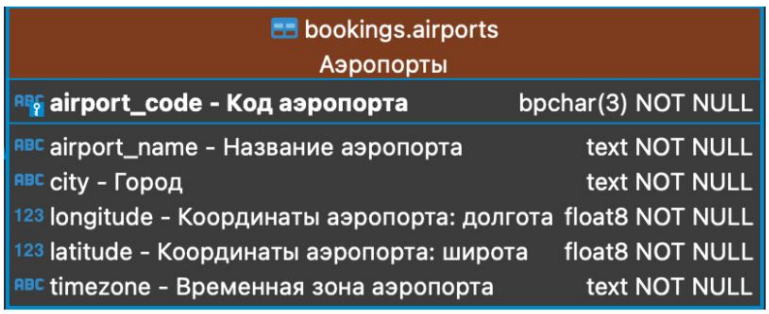
TABLE "flights" FOREIGN KEY (aircraft_code) REFERENCES aircrafts(aircraft_code)
TABLE "seats" FOREIGN KEY (aircraft_code) REFERENCES aircrafts(aircraft_code) ON DELETE CASCADE

airports

The airport is identified by a three-letter code *(airport_code)* and has its own name *(airport_name).* There is no separate entity for the city, but the name *(city)* is specified and can be used to identify the airports of one city.
The latitude *(longitude),* longitude *(latitude)* and time zone *(timezone)* are also indicated .

Indices:

PRIMARY KEY, btree (airport_code)



Links from outside:

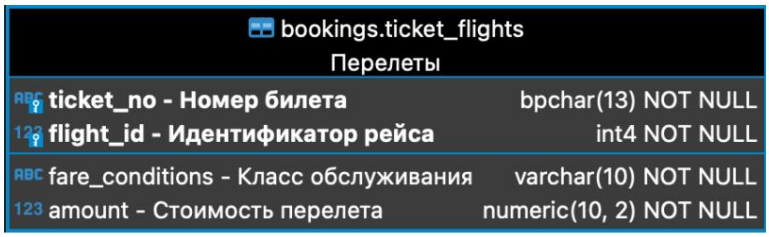
TABLE "flights" FOREIGN KEY (arrival_airport) REFERENCES airports(airport_code) TABLE "flights" FOREIGN KEY (departure_airport) REFERENCES airports(airport_code)

boarding_passes

When checking in for a flight, which is possible one day before the planned departure date, the passenger is issued a boarding pass. It is identified in the same way as the flight - ticket number and flight number. Boarding passes are assigned consecutive numbers *(boarding_no)* in the order in which passengers check in for the flight (this number will be unique only within this flight). at the boarding the ticket indicates the seat number *(seat_no).*

Indices:

PRIMARY KEY, btree (ticket_no, flight_id)
UNIQUE CONSTRAINT, btree (flight_id, boarding_no)
UNIQUE CONSTRAINT, btree (flight_id, seat_no)



Foreign key constraints:

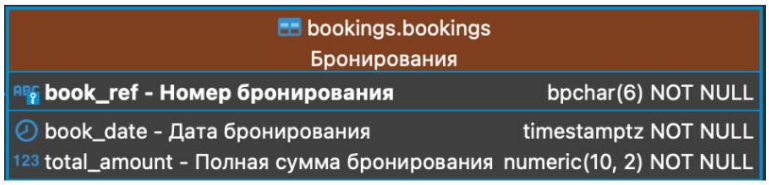
FOREIGN KEY (ticket_no, flight_id) REFERENCES ticket_flights(ticket_no, flight_id)

bookings

The passenger in advance *(book_date,* maximum one month before the flight) books a ticket for himself and, possibly, for several other passengers. Booking is identified number *(book_ref,* a six-digit combination of letters and numbers). The *total_amount* field stores the total cost of flights included in the booking for all passengers.

Indices:

PRIMARY KEY, btree (book_ref)



Links from outside:

TABLE "tickets" FOREIGN KEY (book_ref) REFERENCES bookings(book_ref)

flights

The natural key of the flight table consists of two fields - flight number *(flight_no)* and departure date *(scheduled_departure).* To make foreign keys to this
the table is more compact, a surrogate key is used as the primary
(flight_id). A flight always connects two points - departure airports *(departure_airport)* and
arrival *(arrival_airport).* There is no such thing as a "flight with transfers": if
there is no direct flight from one airport to another, the ticket simply includes
several required flights. Each flight has a scheduled date and
departure time *(scheduled_departure)* and arrival time *(scheduled_arrival).* real time
departure *(actual_departure)* and arrival *(actual_arrival)* may differ: usually not
strongly, but sometimes for several hours if the flight is delayed. Flight *status (status)* can take one
of the following values:

• *Scheduled* Flight is available for booking. This happens a month before
planned date of departure; prior to this, the flight record does not exist in the database.

• *On Time* The flight is available for check-in (one day before the scheduled departure date) and
not detained.

• *Delayed* The flight is available for check-in (one day before the scheduled departure date), but
detained.

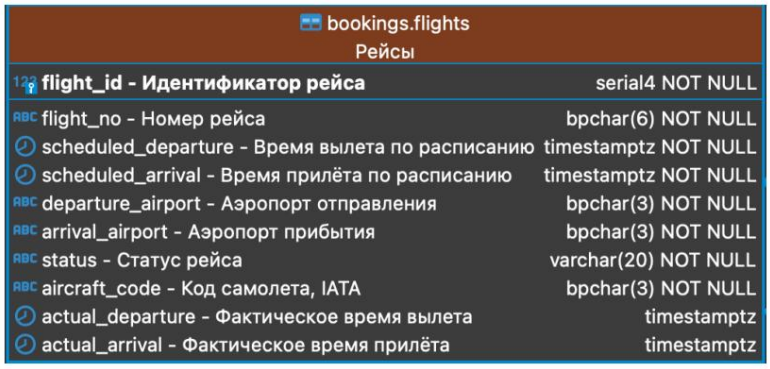
• *Departed* The aircraft has already departed and is in the air.

• *Arrived* The plane has arrived at its destination.

• *Cancelled* Flight canceled

Indices:

PRIMARY KEY, btree (flight_id)
UNIQUE CONSTRAINT,
btree (flight_no,
scheduled_departure)



check-limits:

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'))

Foreign key constraints:

FOREIGN KEY (aircraft_code) REFERENCES aircrafts(aircraft_code)
FOREIGN KEY (arrival_airport) REFERENCES airports(airport_code)
FOREIGN KEY (departure_airport) REFERENCES airports(airport_code)

Links from outside:

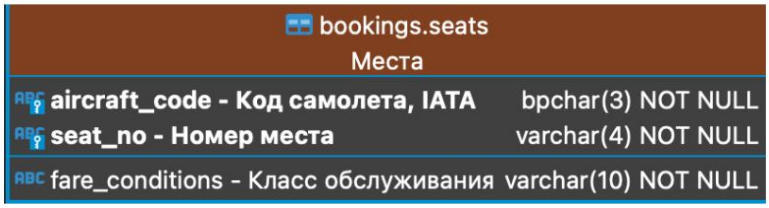
TABLE "ticket_flights" FOREIGN KEY

seats

The seats define the interior layout of each model. Each place is defined by its number *(seat_no)* and has a class of service assigned to it *(fare_conditions)* - Economy, Comfort or Business.

Indices:

PRIMARY KEY, btree (aircraft_code, seat_no)



check-limits:

CHECK(fare_conditions IN('Economy', 'Comfort', 'Business'))

Foreign key constraints:

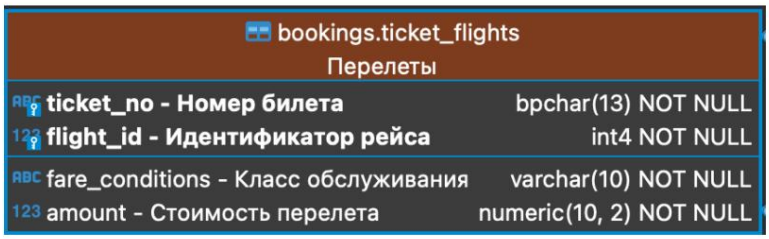
FOREIGN KEY (aircraft_code) REFERENCES aircrafts(aircraft_code) ON DELETE CASCADE

ticket_flights

A flight connects a ticket to a flight and is identified by their flight numbers. For everybody flight, its cost (amount) and class of service *(fare_conditions) are indicated.*

Indices:

PRIMARY KEY, btree (ticket_no, flight_id)



check-limits:

CHECK (amount >= 0) CHECK (fare_conditions IN ('Economy', 'Comfort', 'Business'))

Foreign key constraints:

FOREIGN KEY (flight_id) REFERENCES flights(flight_id)

FOREIGN KEY (ticket_no) REFERENCES tickets(ticket_no)

Links from outside:

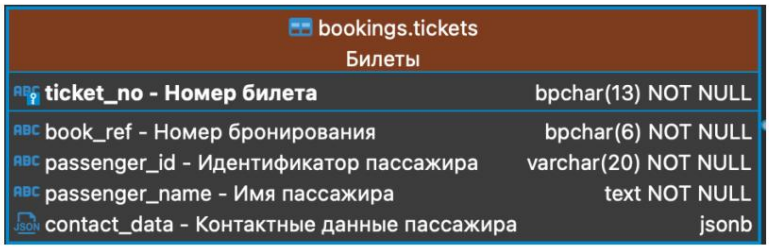
TABLE "boarding_passes" FOREIGN KEY (ticket_no, flight_id)
REFERENCES ticket_flights(ticket_no, flight_id)

tickets

The ticket has a unique number *(ticket_no)* consisting of 13 digits. Ticket contains passenger ID *(passenger_id)* — number of the document certifying identity, - his last name and first name (passenger_name) and contact information *(contact_date).* Neither the passenger ID nor the name is permanent (you can change your passport, you can change your last name), so you can definitely find everything tickets of the same passenger is not possible.

Indices:

PRIMARY KEY, btree (ticket_no)



Foreign key constraints:

FOREIGN KEY (book_ref) REFERENCES bookings(book_ref)

Links from outside:

TABLE "ticket_flights" FOREIGN KEY (ticket_no) REFERENCES tickets(ticket_no)

Representation

flights_v **view**

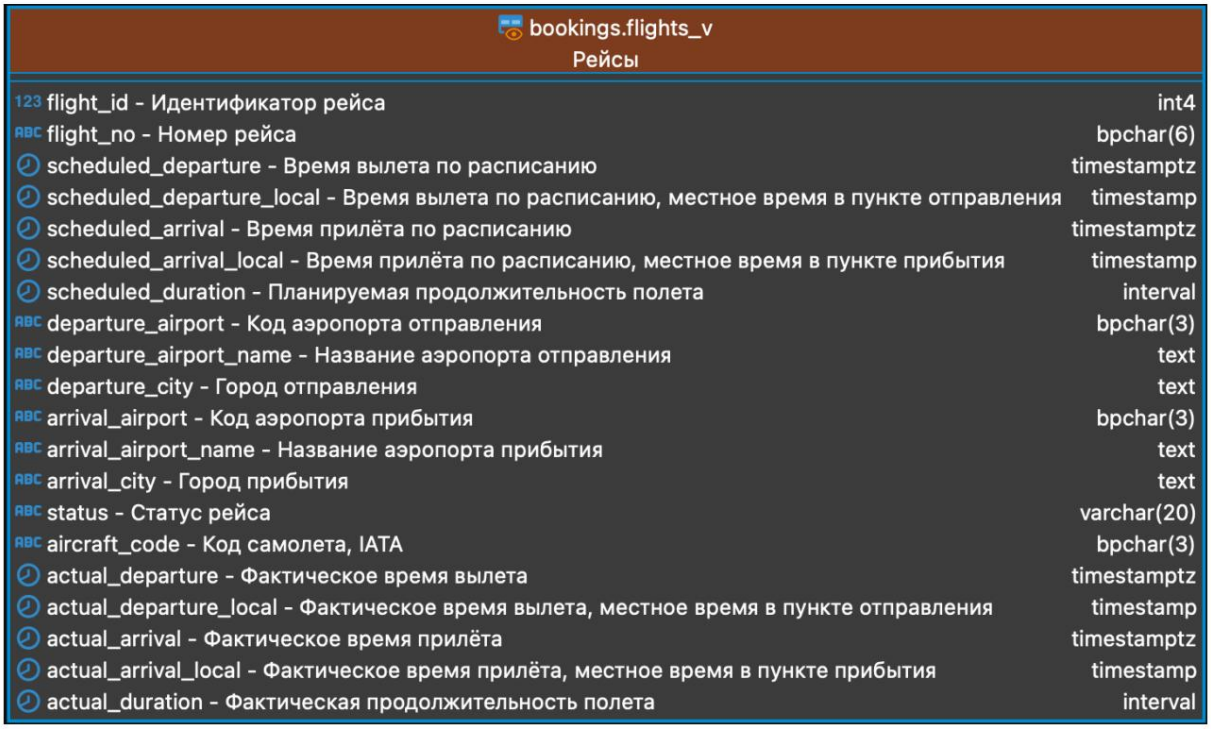
A *flights_v* view is created above the *flights* table, containing an additional information:

• decryption of data about the airport of departure *(departure_airport, departure_airport_name, departure_city),*

• decoding data about the airport of arrival *(arrival_airport, arrival_airport_name, arrival_city),*

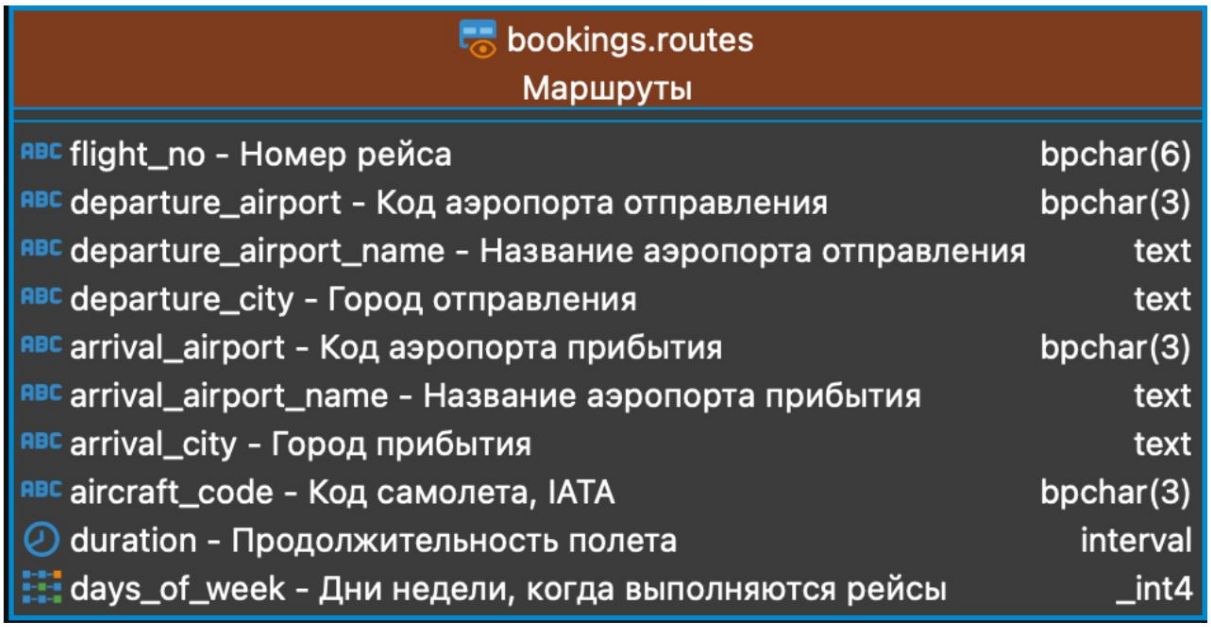
• local departure time *(scheduled_departure_local, actual_departure_local),* • local arrival time *(scheduled_arrival_local, actual_arrival_local),*

• flight duration *(scheduled_duration, actual_duration).*



Materialized view **bookings.routes**

The flight table contains redundancy: from it it would be possible to extract information about the route (flight number, airports of departure and destination), which does not depend on specific flight dates. It is this information that makes up the materialized view of routes.



What business problems can be solved using this data

1 Key business metrics for airlines:

Capacity - the number of seats available (ASM). The number of seats is calculated by multiplying the number of miles a given aircraft will fly by the number of seats available for that flight.

Traffic - Revenue Passenger Miles (RPM). Income passenger miles
are calculated by multiplying the number of paying passengers by the distance travelled.

The load factor is a percentage that reflects how efficiently an airline generates revenue. To calculate an airline's load factor, divide revenue passenger miles (RPM) by the number of seats available (ASM).

Revenue - Revenue per available seat (empty or full) per flight mile (RASM). The calculation of revenue per available seat mile (RASM) is a general
operating revenue divided by the number of seats available.

Cost - Cost Per Available Seat Mile (CASM)

Passenger revenue per mile of available seat and counts all seats, including empty seats (PRASM)

Based on the methodology for calculating the above metrics and the data available to us, we can only calculate capacity, traffic, and load factor. Having made additional calculations, we obtain the following data (taking into account that we have flight data for 1 month):

Capacity - Seats Available (ASM) - 1,145,118,307 seat miles

Traffic - Revenue passenger miles (RPM) - 579,254,762 passenger miles

Load Factor (RPM/ASM) - 0.51

Let's see what was the statistics on the coefficient of statistics on the market during this period. Publicly available data from the US Bureau of Transportation Statistics. In the period under consideration (September-November 2016), in the sector of domestic passenger air transportation, the load factor was *0.85. (Bureau of Transportation Statistics).* This is 1.66 times more than in the

airlines.

This suggests that, most likely, this airline is unprofitable in domestic passenger air transportation.

What can we advise to increase the load factor:

1. Reducing the frequency of departures on low-load routes

2. Increasing the frequency of departures on highly loaded destinations

3. Analysis of the correctness of the time of departure / arrival (morning, afternoon, evening)

4. Replacement with aircraft of smaller capacity (on highly loaded routes - with aircraft of larger capacity).

5. Increase in prices, tariffs

6. On potentially promising areas that can "shoot"
   - carrying out marketing activities

Of course, all these changes need to be carried out by carrying out a huge analytical work with the analysis of a lot of additional data.

5. List of SQL queries from application No. 2 with a description of the logic of their execution.

The query execution logic is written in the form of comments in the SQL query.