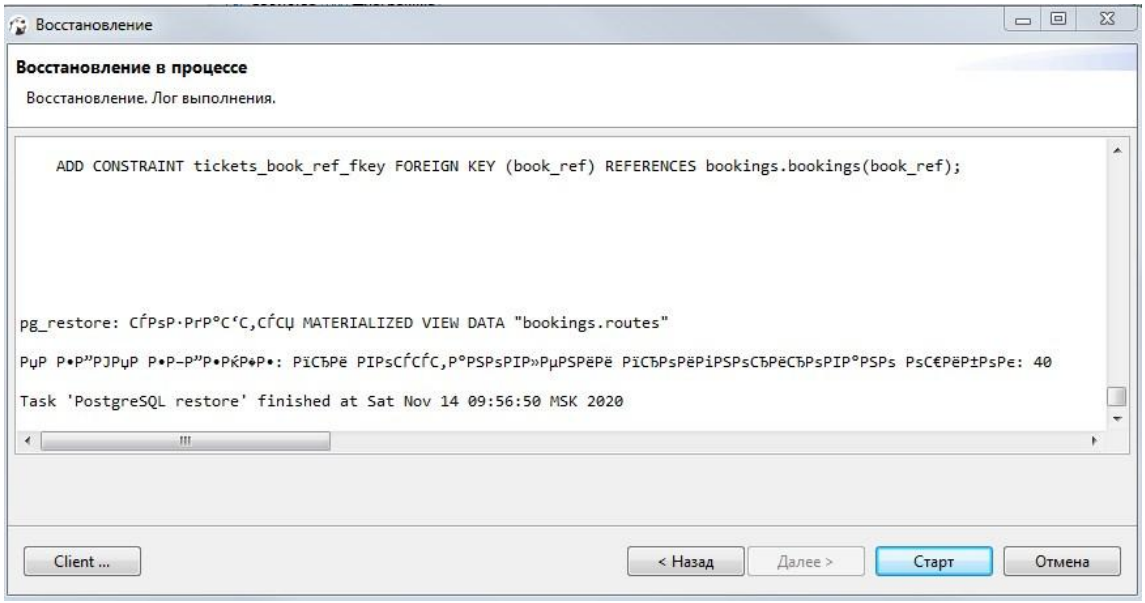
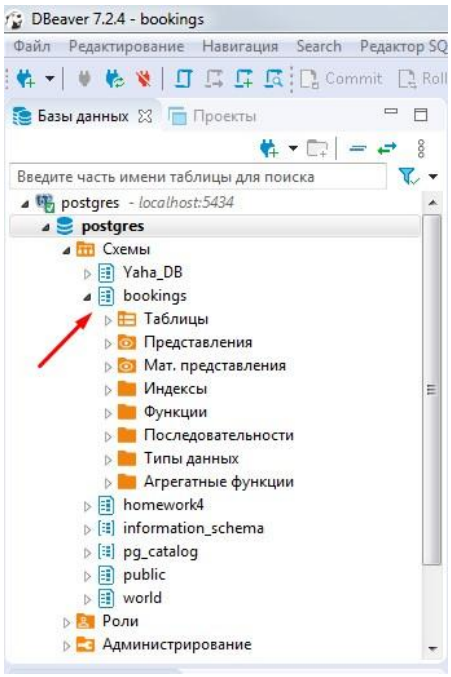
Проектная работа по модулю “SQL и получение данных”

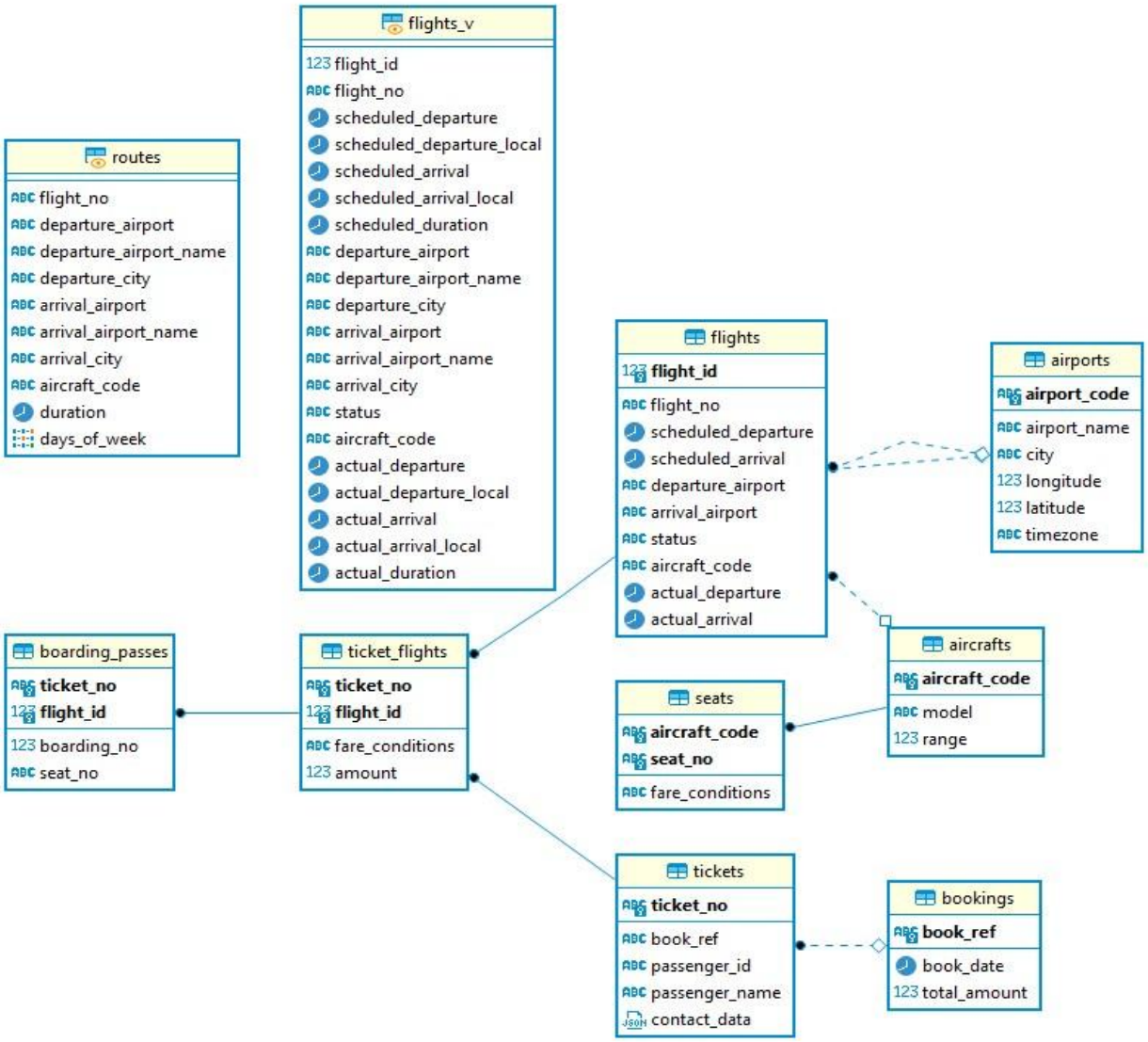
1. В работе использовался локальный тип подключения. Для восстановления из .backup файла в DBeaver’е на БД postgres нажимаем ПКМ - Инструменты - Восстановить. В Файл резерва указываем путь к .backup файлу и жмем Старт.

   Скриншоты восстановления:





2. Скриншот ER-Диаграммы:



3. База данных состоит из следующих таблиц:
    - Бронирования (bookings)
    - Билеты (tickets)
    - Рейсы (flights)
    - ticket_flights - связь между билетами и рейсами
    - Посадочные талоги (boarding_passes)
    - Аэропорты (airports)
    - Самолеты (aircrafts)
    - Места и класс (seats)
    - Представление "bookings.flights_v"
    - Материализованное представление bookings.routes - маршруты

4. 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).

Таблица aircrafts

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

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

Таблица 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).

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)

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).

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)

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.

Индексы:

PRIMARY KEY, btree (book_ref)

Ссылки извне:

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

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.

- *Arrived*: Самолет прибыл в пункт назначения.

- *Cancelled*: Рейс отменен.

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)

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.

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 ticket_flights

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

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)

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_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.

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"

Представление "bookings.flights_v" создано над таблицей flights, 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).

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.

4.1 Какие бизнес задачи можно решать основываясь на данных этой базы:

- Можно посмотреть как часто в одном билете присутствуют несколько рейсов, т.е. Как часто и из какого города в какой люди летают с пересадками, может есть смысл добавить прямой рейс.
- По каким маршрутам и с какой периодичностью летают полупустые самолеты, подумать над изменением графика полетов, например если рейсы идут каждый день, сделать их 2 или 3 раза в неделю. Или изменить самолеты на менее вместительные.

5. Список SQL запросов с описанием логики их выполнения в приложенном файле Final_work1_1.sql.

Выполнил:
Селихов Д.Ю.

2020