Итоговая работа по модулю

"SQL и получение данных"

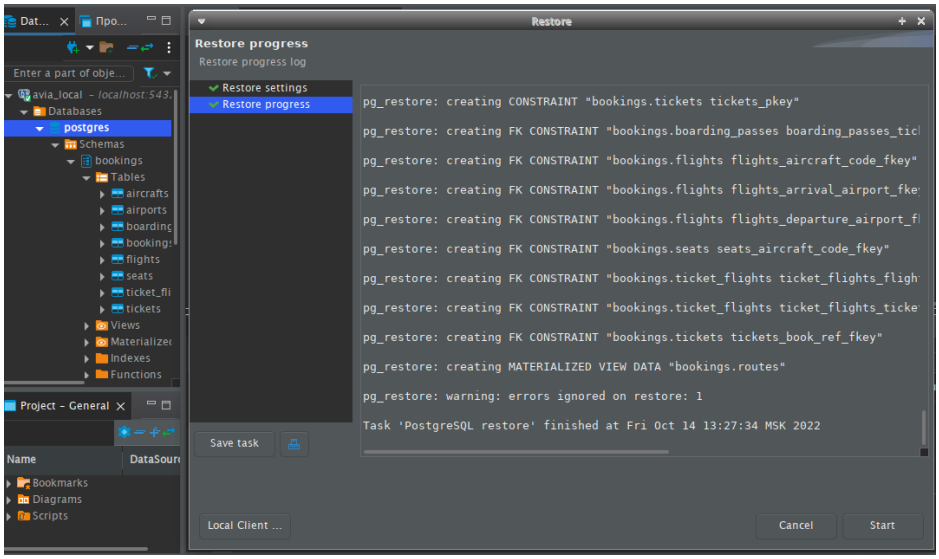
(Нетология)

2022

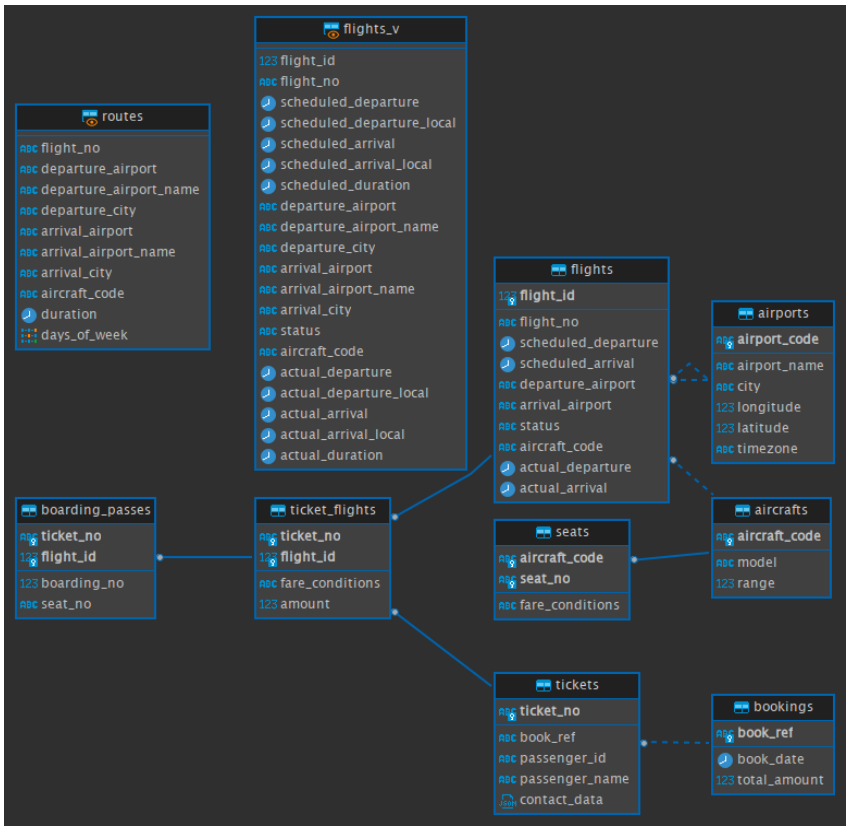
Кравченков Антон

1. Введение

В работе использовался локальный тип подключения.



ER-диаграмма:



2. Краткое описание БД

Таблицы

1. aircrafts
2. airports
3. boarding_passes
4. bookings
5. flights
6. seats
7. ticket_flights
8. tickets

Представления

1. flights_v
2. routes

3. Развернутый анализ БД

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; 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 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).

Объекты схемы

Список отношений

1. Таблица **aircrafts**

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

2. 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*).

3. 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).

4. 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.

5. 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.

6. 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.

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

8. 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.

1. 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*).

2. Материализованное 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 материализованное представление **routes**.

4. Бизнес задачи, которые можно решить, используя БД

a. На основании данных о бронированиях пассажиров, предполагающих перелеты с пересадками, может быть оценен **спрос на открытие новых направлений** между городами, ранее не имевшими прямых перелетов.

b. На основании данных о среднем количестве пассажиров может быть оценена **рентабельность** конкретных рейсов, рейсов в определенные дни недели/месяцы или направлений в целом.

c. На основании данных о среднем количестве пассажиров и данных о дальности перелетов может быть проведена **оптимизация используемых воздушных судов**.

d. На основании данных о количестве бронирований пассажира, направлениях перелетов и предпочитаемом классе обслуживания могут быть разработаны **персональные предложения**.

e. На основании данных о ежедневной загруженности аэропортов может быть скорректирован **график полетов, минимизирующий риски возникновения коллапсов** в случае отмены или задержки рейсов, напр. из-за неблагоприятного прогноза погоды.

5. Список SQL запросов с описанием логики их выполнения

См. приложение - ***SQL_final_project_kravchenkov.sql***
Описание логики выполнено в виде комментариев перед каждым sql запросом.