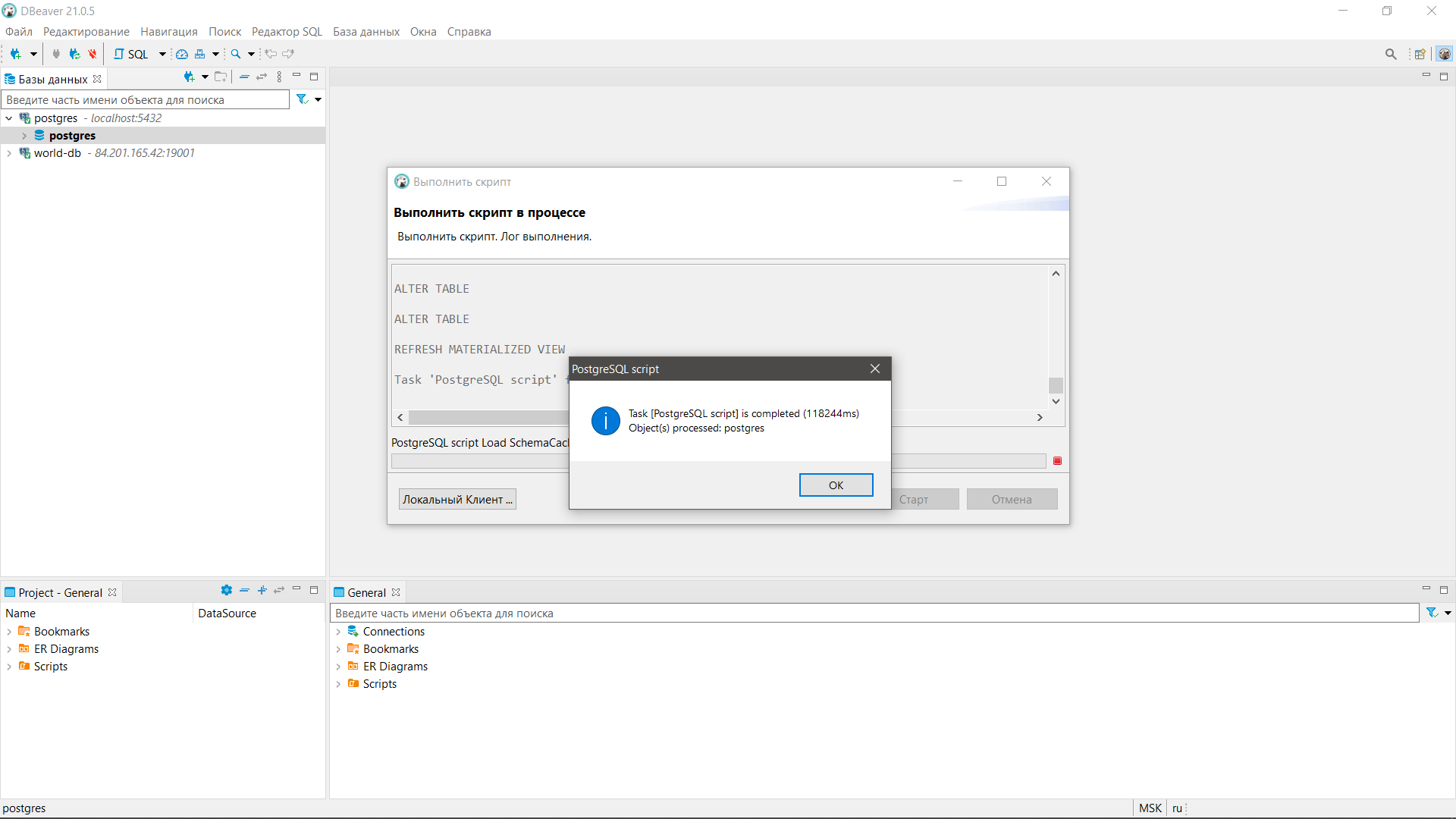
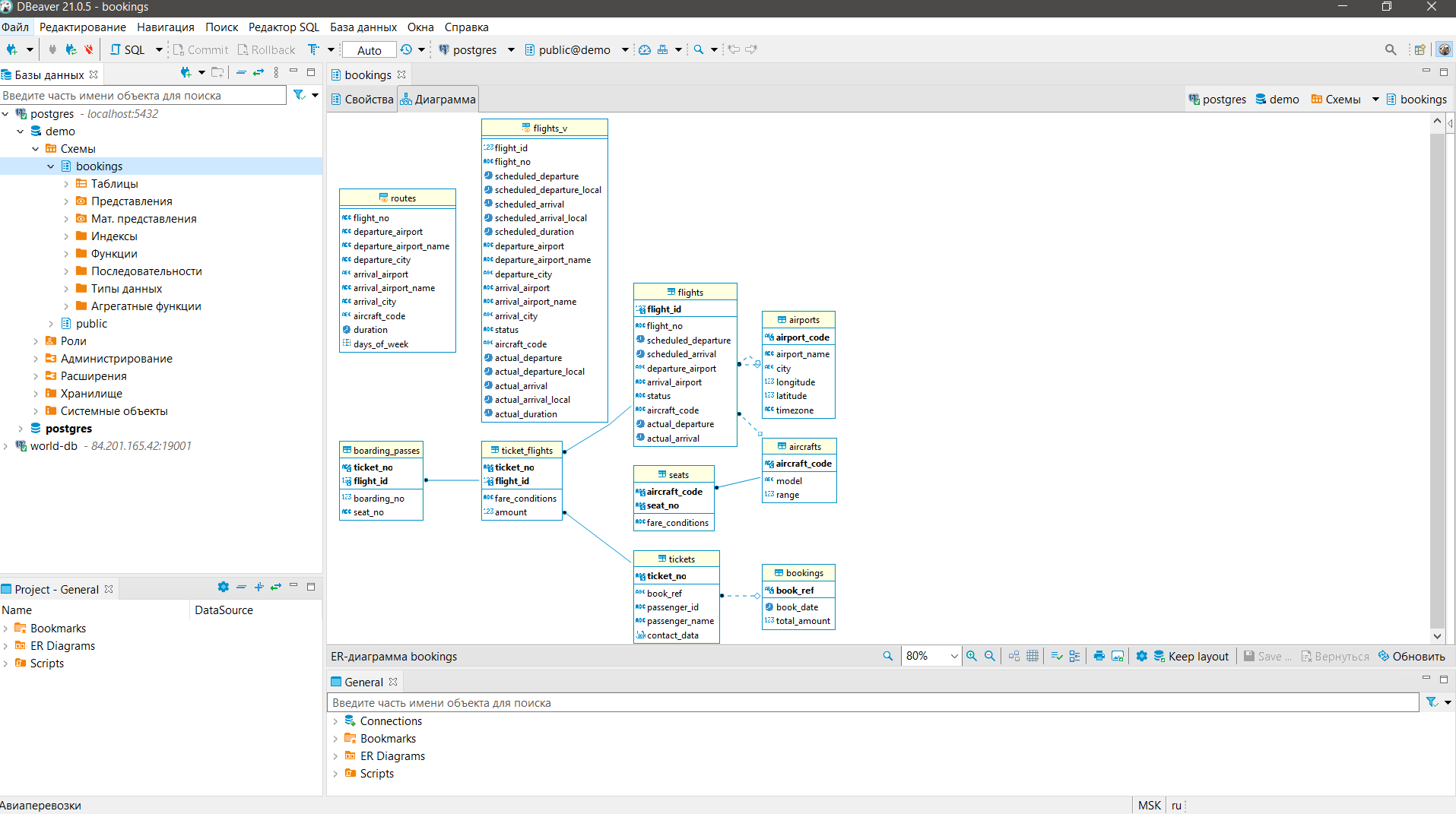
**Итоговая работа**

**1. В работе использовался локальный тип подключения.**



**2. Скриншот ER-диаграммы из DBeaver`a согласно подключения.**



**3. Краткое описание БД - из каких таблиц и представлений состоит.**

**Таблицы:**

|  |  |
| --- | --- |
| Названия | Первичные ключи |
| flights | flight\_id |
| airports | airport\_code |
| aircrafts | aircraft\_code |
| seats | aircraft\_code, seat\_no |
| bookings | book\_ref |
| tickets | ticket\_no |
| ticket\_flights | ticket\_no, flight\_id |
| boarding\_passes | ticket\_no, flight\_id |

**Представления:**

- routes

- flights\_v

**4 Развернутый анализ**

**Таблица bookings.aircrafts**

Каждая модель воздушного судна идентифицируется своим трехзначным кодом (aircraft\_code). Указывается также название модели (model) и максимальная дальность полетав километрах (range)

**Таблица 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).

**Таблица 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).

**Таблица 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

**Таблица 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.

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.

**Таблица 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.

**Таблица bookings.ticket\_flights**

Перелет соединяет билет с рейсом и идентифицируется их номерами.

Для каждого перелета указываются его стоимость (amount) и класс обслуживания (fare\_conditions).

**Таблица 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.

**Представление "bookings.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).

**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

**Бизнес задачи**

1. Определение загруженности в разные промежутки времени, для формирования цены, количества рейсов;
2. Контроль задержек рейсов, чтобы уменьшить оные;
3. Контроль забронированных мест, для отображения свободных мест потенциальным клиентам

**5 Список SQL запросов из приложения №2 с описанием логики их выполнения.**

**Задание №1**

1. Вывел 2 колонки:

- количество аэропортов, применив оператор count, чтобы посчитать города

- города

2. Выполнил группировку по городам

3. Прописал условие, что кол-во аэропортов должно быть > 1, для вывода только тех городов, в которых более одного аэропорта

**Задание №2**

1. Вывел колонки:

- код аэропорта (применив distinct, чтобы вывести уникальные значения)

- название аэропортов

- модель самолёта

- дальность полёта модели

2. Присоединил таблицы полётов и самолётов

3. Прописал условие – вывести модель с максимальной дальностью полёта, использовав подзапрос

**Задание № 3**

1. Вывел 2 колонки

- номер рейса

- задержка рейса (вычел «Время вылета по расписанию» из «Фактическое время вылета»

2. Прописал условие not nullдля задержки, чтобы исключить своевременные вылеты

3. Отсортировал значения задержек от большего к меньшему

4. Применил оператор limit для выдачи топ 10 по задержкам

**Задание №4**

1. Вывел 2 колонки

- номера брони

- номера посадочных талонов

2. Соединил таблицы bookings, tickets, boarding\_passes. Для boarding\_passes применил left join, чтобы вывести брони без посадочных талонов

3 Прописал условие is null*,* чтобы оставить только брони без посадочных талонов

**Задание №6**

1. Вывел 2 колонки

- коды моделей самолётов

- процентное соотношение

2. разделил количество полётов каждой модели (посчитал, применив group by) на общеее количество полётов всех моделей (посчитал, применив подзапрос) и умножил на 100. Применял оператор cast, чтобы установить для чисел тип данных с плавающей точкой

3. Группировка по кодам моделей самолётов

**Задание № 7**

1. Прописал 2 CTE

- эконом-класс

- бизнес-класс

2. Вывел колонку с городами, чтобы город, по условию задания, выводился единожды добавил оператор distinct

3. Соединил с flights другие таблицы airports, cte\_1, cte\_2

4. Прописал условие, по которому выведутся города, в которые можно добраться бизнес - классом дешевле, чем эконом-классом в рамках перелета

**Задание № 8**

Я исходил из того, чтобы в t\_1 вывести все варианты перелётов между аэропортами, какие могли бы быть, в t\_2 вывести варианты, какие есть на самом деле. Distinct, чтобы убрать повторения.

Использовал оператор except, чтобы из t\_1 убрать сущетвующие пары городов, между которыми осуществляются прямые рейсы

**Задание №9**

1. Сделал выборку из аэрорта отправления, прибытия их координат, а также дальности самолётов, осущетвлявших полёты на этих рейсах.

2. Присоединил дважды таблицу aiports под разными псевдонимами, чтобы у departure\_airport и arrival\_airport были соответствующие им координаты.

3. Присоединил таблицу aircrafts для вывода дальности самолётов

4. Вывел через запрос пары аэропортов с прямыми сообщениями, дистанцию между ними, использовав формулу и дальность самолётов для сравнения результата