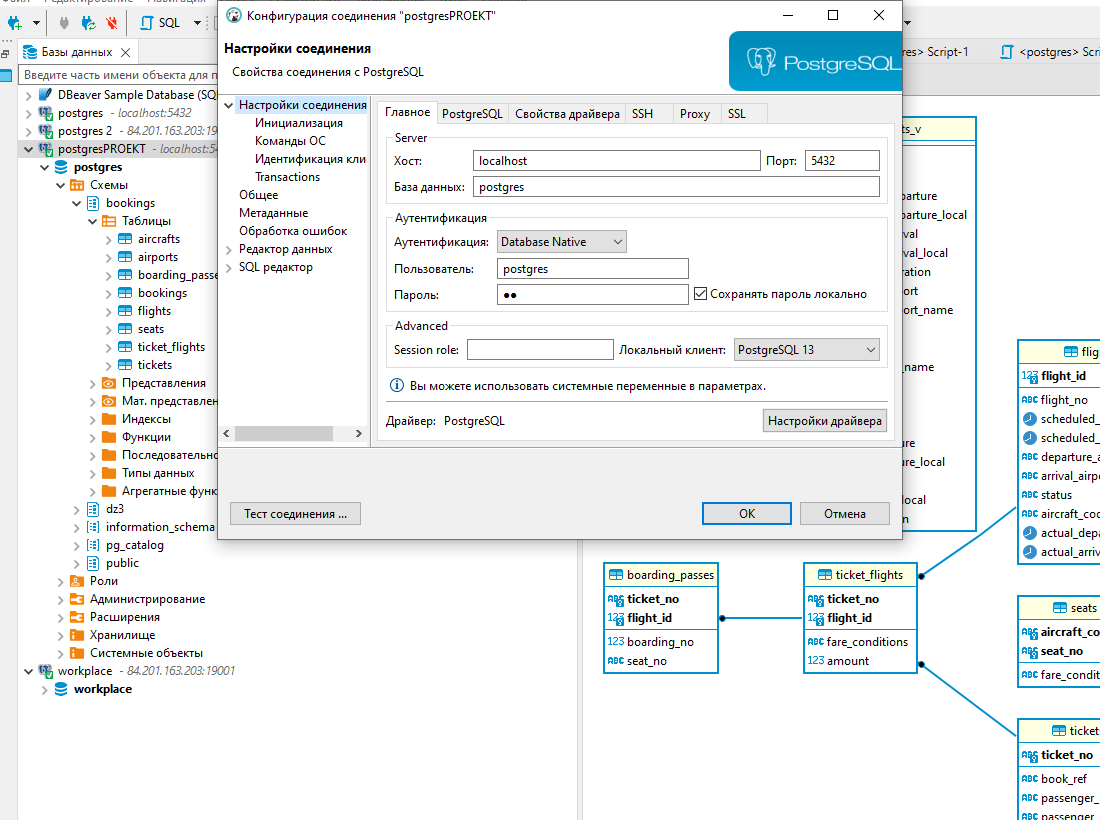
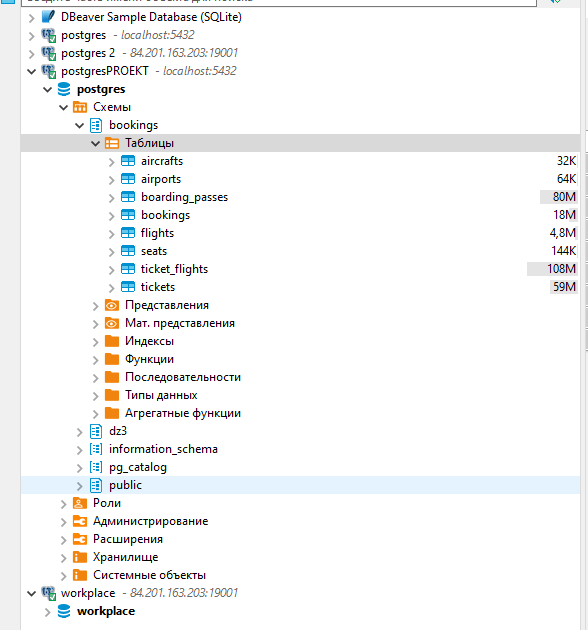
Итоговая работа

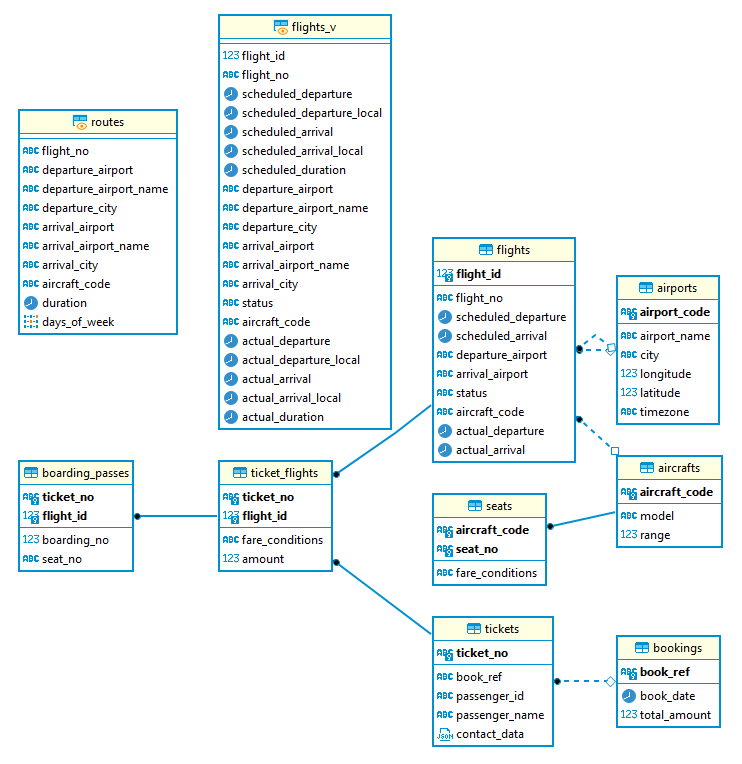
**1.Тип подключения.**

База была развернута из \*.backup файла, необходимо приложить скриншот успешного импорта или восстановления





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



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

|  |  |
| --- | --- |
| **Название таблицы** | **Содержание** |
| aircrafts | Самолеты |
| airports | Аэропорты |
| boarding\_passes | Посадочные талоны |
| bookings | Бронирования |
| flights | Рейсы |
| seats | Места |
| ticket\_flights | Перелеты |
| tickets | Билеты |
| **Название представления** | **Содержание** |
| flights\_v | Рейсы |
| **Название мат. представления** | **Содержание** |
| routes | Маршруты |

Развернутый анализ БД - описание таблиц, логики, связей и бизнес области (частично можно взять из описания базы данных, оформленной в виде анализа базы данных). Бизнес задачи, которые можно решить, используя БД.

Задание не вполне ясное, так как в файле достаточно полное описание БД (приведу ниже).

**4. Бизнес задачи и описание.**

Так как согласно описанию мы не можем вести точную базу клиентов (меняются имена и паспорта), то много из программ лояльности отпадают, но можно перераспределять самолётный парк согласно имеющимся маршрутам-клиентам.

Предлагать оптимальные по цене или расстоянию маршруты клиентам.

На основе предпочтений и загруженности создавать новые или закрывать старые маршруты.

Осуществлять акции с повышением уровня обслуживания на отдельных рейсах в случае перегруженности эконома.

**4.1 Описание БД.**

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, 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).

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

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).

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)/

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.

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

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.

Перелет соединяет билет с рейсом и идентифицируется их номерами.

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

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.

**5. Список SQL запросов из приложения №2**

-- 1)В каких городах больше одного аэропорта?

select city

from airports

group by city

having count(\*) > 1

--2)В каких аэропортах есть рейсы, выполняемые самолетом с максимальной дальностью перелета?

select distinct departure\_airport

from (select a.aircraft\_code

from aircrafts a

order by a."range" desc

limit 1

) a

join flights f on f.aircraft\_code = a.aircraft\_code

--3) Вывести 10 рейсов с максимальным временем задержки вылета

select actual\_departure - scheduled\_departure as departure\_delay, flight\_no, departure\_airport

, f.arrival\_airport

from flights f

where actual\_departure - scheduled\_departure is not null

order by departure\_delay desc

limit 10

--4) Были ли брони, по которым не были получены посадочные талоны?

select b.book\_ref, bp.boarding\_no

from bookings b

left join tickets t on t.book\_ref = b.book\_ref

left join boarding\_passes bp on t.ticket\_no = bp.ticket\_no

where bp.boarding\_no is null

--5) Найдите количество свободных мест для каждого рейса, их % отношение к общему количеству мест в самолете.

--Добавьте столбец с накопительным итогом - суммарное накопление количества вывезенных пассажиров из каждого аэропорта на каждый день. Т.е. в этом столбце должна отражаться --накопительная сумма - сколько человек уже вылетело из данного аэропорта на этом или более ранних рейсах в течении дня.

with cte as (

select s.aircraft\_code, count(s.seat\_no) as num, a.model

from seats s

join aircrafts a on a.aircraft\_code = s.aircraft\_code

group by s.aircraft\_code, a.model

)

select departure\_airport, actual\_departure , cte.num - count(bp.seat\_no) as "Свободные места",

(((cte.num - count(bp.seat\_no))::numeric / cte.num)::numeric(32,2)) \* 100 as "% свободных мест",

sum(count(bp.seat\_no)) over (partition by f.actual\_departure::date, f.departure\_airport order by f.actual\_departure)

from boarding\_passes bp

join flights f on f.flight\_id = bp.flight\_id

join cte on cte.aircraft\_code = f.aircraft\_code

group by f.flight\_id, cte.num

--6) Найдите процентное соотношение перелетов по типам самолетов от общего количества.

select model, (round(num::numeric / (sum(num) over ()), 2) \* 100) as "% от общего кол-ва перелётов"

from(

select count(flight\_id) as num, model

from flights f

join aircrafts a on a.aircraft\_code = f.aircraft\_code

group by model

) l

--7 Были ли города, в которые можно добраться бизнес - классом дешевле, чем эконом-классом в рамках перелета?

with cte as (

select distinct tf.flight\_id, max(amount) as price\_econom, fare\_conditions

from ticket\_flights tf

where tf.fare\_conditions = 'Economy'

group by flight\_id, fare\_conditions

order by flight\_id

) ,

cte1 as (

select distinct tf.flight\_id, min(amount) as price\_business , fare\_conditions

from ticket\_flights tf

where tf.fare\_conditions = 'Business'

group by flight\_id, fare\_conditions

order by flight\_id

)

select a.city as "Город прибытия"

from cte

join cte1 on cte1.flight\_id = cte.flight\_id

join flights f on f.flight\_id = cte.flight\_id

join airports a on a.airport\_code = f.arrival\_airport

where price\_business < price\_econom

--8 Между какими городами нет прямых рейсов?

create view cities\_v as

select v.departure\_city, v.arrival\_city

from flights\_v v

select distinct a.city, a1.city

from airports a

cross join airports a1

where a.city != a1.city

except

select c.departure\_city, c.arrival\_city

from cities\_v c

--9 Вычислите расстояние между аэропортами, связанными прямыми рейсами, сравните с допустимой максимальной дальностью перелетов в самолетах, обслуживающих эти рейсы

with cte as (

select f.departure\_airport,--, dep.longitude, dep.latitude, f.arrival\_airport, arr.longitude, arr.latitude,

f.arrival\_airport, f.aircraft\_code,

round(((acos((sind(dep.latitude)\*sind(arr.latitude) + cosd(dep.latitude) \* cosd(arr.latitude) \* cosd((dep.longitude - arr.longitude))))) \* 6371)::numeric, 2)

as distance\_airports ,

f.flight\_no,

dep.airport\_name as departure\_airport\_name,

arr.airport\_name as arrival\_airport\_name

from

flights f,

airports dep,

airports arr

where f.departure\_airport = dep.airport\_code and f.arrival\_airport = arr.airport\_code

)

select distinct cte.departure\_airport\_name, cte.arrival\_airport\_name, cte.distance\_airports,

a.range as aircraft\_flight\_distance,

case

when range > distance\_airports

then 'TRUE'

else 'FALSE'

end result

from aircrafts a

join cte on cte.aircraft\_code = a.aircraft\_code