Модуль SQL

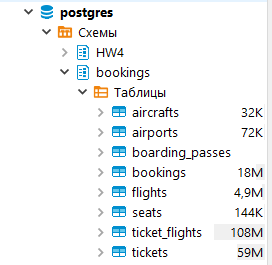
Итоговая работа

bookings Database

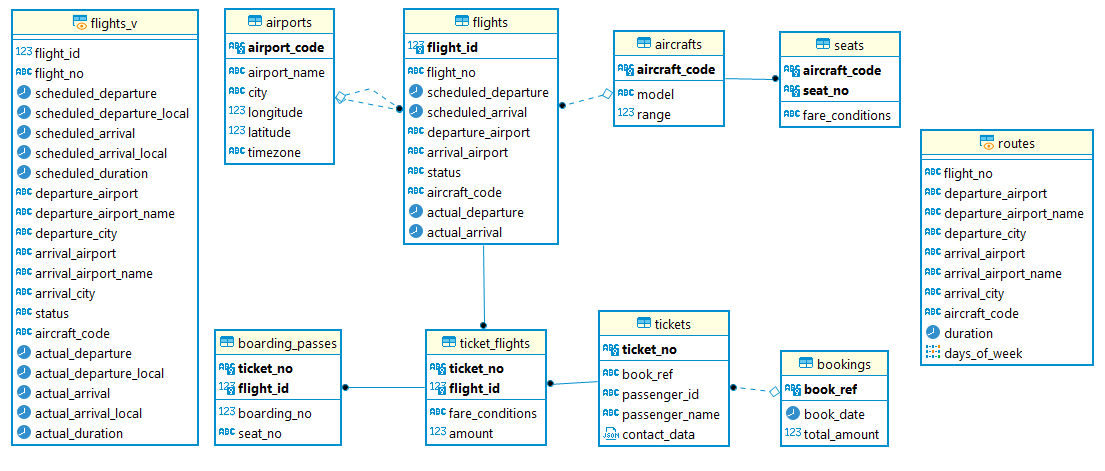


**Группа SQL-29**

1. В работе использовался локальный тип подключения



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

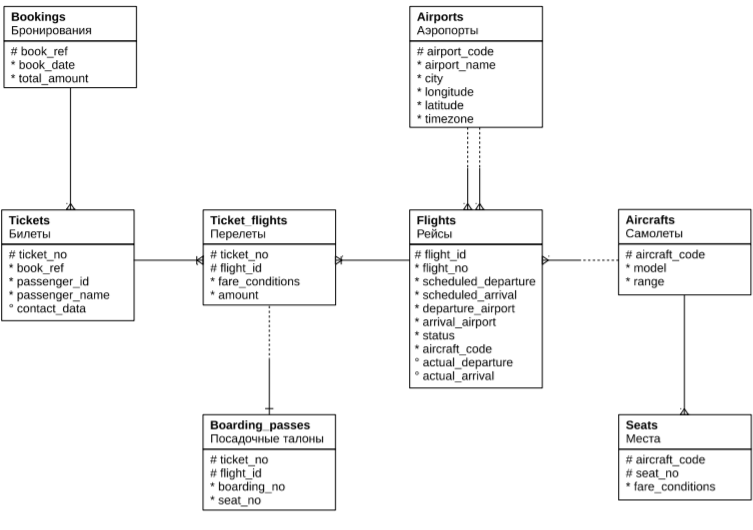


1. Краткое описание БД - из каких таблиц и представлений состоит.

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Таблица** | **Описание** | **На ER Диаграмме** |
| 1 | bookings.aircrafts | 1. Код самолета 2. Модель самолета 3. Максимальная дальность 4.полета |  |
| 2 | bookings.airports | 1. Код аэропорта 2. Название аэропорта 3. Город 4. Координаты аэропорта: долгота 5. Координаты аэропорта: широта 6. Временная зона аэропорта |  |
| 3 | bookings.boarding\_passes | 1. Номер билета 2. Идентификатор рейса  3. Номер посадочного талона 4. Номер места |  |
| 4 | bookings.bookings | 1. Номер бронирования 2. Дата бронирования 3. Полная сумма бронирования |  |
| 5 | bookings.flights | 1. Идентификатор рейса 2. Номер рейса 3. Время вылета по расписанию 4. Время прилёта по расписанию 5. Аэропорт отправления 6. Аэропорт прибытия 7. Статус рейса 8. Код самолета 9. Фактическое время вылета 10. Фактическое время прилёта |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 6 | bookings.seats | 1. Код самолета 2. Номер места 3. Класс обслуживания |  |
| 7 | bookings.ticket\_flights | 1. Номер билета 2. Идентификатор рейса 3. Класс обслуживания 4. Стоимость перелета |  |
| 8 | bookings.tickets | 1. Номер билета 2. Номер бронирования 3. Идентификатор пассажира 4. Имя пассажира 5. Контактные данные пассажира |  |
| 9 | Представление  bookings.flights\_v | 1. Идентификатор рейса 2. Номер рейса 3. Время вылета по расписанию 4. Время вылета по расписанию, местное время в пункте отправления 5. Время прилёта по расписанию 6. Время прилёта по расписанию, местное время в пункте прибытия 7. Планируемая продолжительность полета 8. Код аэропорта отправления 9. Название аэропорта отправления 10. Город отправления 11. Код аэропорта прибытия 12. Название аэропорта прибытия 13. Город прибытия 14. Статус рейса 15. Код самолета 16. Фактическое время вылета 17. Фактическое время вылета, местное время в пункте отправления 18. Фактическое время прилёта 19. Фактическое время прилёта, местное время в пункте прибытия 20. Фактическая продолжительность полета |  |
| 10 | Материализованное представление bookings.routes | 1. Номер рейса 2. Код аэропорта отправления 3. Название аэропорта отправления 4. Город отправления 5. Код аэропорта прибытия 6. Название аэропорта прибытия 7. Город прибытия 8. Код самолета 9. Продолжительность полета 10. Дни недели, когда выполняются рейсы |  |

4.1 Развернутый анализ БД



Основной сущностью является бронирование (bookings). В одно бронирование можно включить несколько пассажиров, каждому из которых выписывается отдельный билет (tickets). Как таковой пассажир не является отдельной сущностью. 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».

Каждый рейс (flights) следует из одного аэропорта (airports) в другой. При регистрации на рейс пассажиру выдается посадочный талон (boarding\_passes), в котором указано место в самолете.

Количество мест (seats) в самолете и их распределение по классам обслуживания зависит от модели самолета (aircrafts), выполняющего рейс.

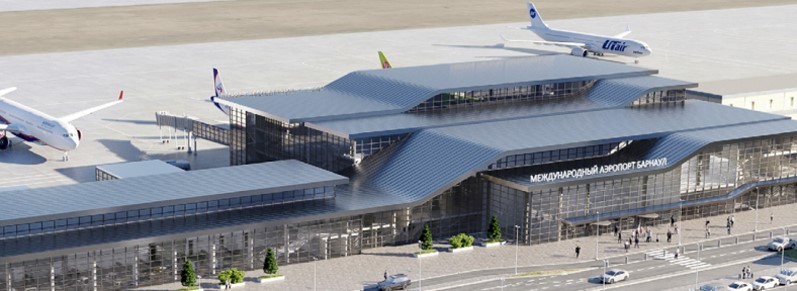
* 1. Описание таблиц

Таблица bookings.aircrafts



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

**•** **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. 6

**• 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).

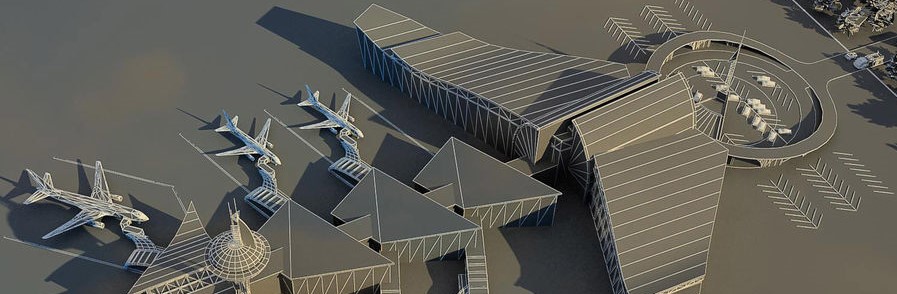
Таблица bookings.tickets



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).

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.

* 1. Бизнес задачи, которые можно решить



1. Анализ цены перелета в зависимости от сезонности
2. Получение списка наиболее часто летающих пассажиров
3. На каких направлениях чаще всего происходят задержки рейса
4. Конверсия по % проданных мест в самолете
5. Получение Доли того или иного Класса перелета от общего кол-ва мест в салоне

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



**5.1 В каких городах больше одного аэропорта?**

**select** city, # Выбираем города

**count**(airport\_name) **as** "Airports\_num" # и считаем аэропорты

**from** airports # из этой таблицы

**group** **by** city # группируем по городам

**having** **count**(airport\_name) > 1; # аэропорты, которых > 1

**5.2 В каких аэропортах есть рейсы, выполняемые самолетом с максимальной дальностью перелета? (Подзапрос)**

**select** # Выбираем уникальные

**distinct** airport\_name # имена аэропортов

**from** airports ai # из этой таблицы,

**join** flights f

**on** ai.airport\_code = f.departure\_airport # cоединенной с этими

**join** aircrafts a # двумя таблицами

**on** f.aircraft\_code = a.aircraft\_code

**where** a."range" = # для условия возвращать

(**select** **max**(**range**) **from** aircrafts a2) # результат по макс. радиусу

**order** **by** airport\_name;

**5.3 Вывести 10 рейсов с максимальным временем задержки вылета (Оператор LIMIT)**

**select** \* # Выбираем все из

**from** # подзапроса, в котором

(**select** flight\_id, flight\_no, # получаем эти столбцы

actual\_departure - scheduled\_departure # и время задержки

**as** delay\_time

**from** flights) **as** delay

**where** delay\_time **is** **not** **null** # c фильтром без пустыхстрок

**order** **by** delay\_time **desc** # по убыванию задержки

**limit** 10; # и лимитом в 10 строк

**5.4 Были ли брони, по которым не были получены посадочные талоны? (Верный тип JOIN)**

**select** "Номер Брони", # Выбираем столбец номер брони

**count**("Номер Брони") # и считаем пустые талоны

**as** "Не были Получены"

**from** \_

(**select** bo.book\_ref **as** "Номер Брони", |

b.boarding\_no **as** "Посадочный Талон" |

**from** bookings bo |# В подзапросе джойним

**left** **join** tickets |# необходимые, чтобы

**using** (book\_ref) |# получить номера броней

**left** **join** boarding\_passes b |# с пустыми посадочными

**using** (ticket\_no) |# талонами

**where** b.boarding\_no **is** **null**) **as** boo \_|

**group** **by** "Номер Брони" # С группировкой по брони

**order by** "Номер Брони"

5.5 **Найдите свободные места 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 за день. (Окно, подзапрос или CTE)**

**select** flight\_no, departure\_airport, actual\_departure, # Выбираем итоговые

available\_seats, available\_seats\_share, \_# данные по условию

**sum**(seat\_count - available\_seats) |

**over** (**partition** **by** actual\_departure::**date**, |# Добавляем столбец,

departure\_airport |# где будет жить

**order** **by** actual\_departure) |# кол-во вылетевших

**as** departed\_passengers \_|# Пассажиров

**from** \_

(**select** f.flight\_no, f.departure\_airport, |

f.actual\_departure, sc.seat\_count, |# Получаем процент

af.available\_seats, |# Свободных мест

**round**(af.available\_seats::**numeric** / |

sc.seat\_count::**numeric** \* 100, 2) |

**as** available\_seats\_share \_|

**from** flights f

**left** **join** \_

(**select** f.flight\_id, f.flight\_no, |# Получаем кол-во

**count**(f.flight\_id) **as** available\_seats\_|# свободных мест

**from** flights f

**left** **join** seats s # Соединение с номерами мест

**on** f.aircraft\_code = s.aircraft\_code

**left** **join** boarding\_passes bp # Соединение с Бронированием

**on** s.seat\_no = bp.seat\_no

**and** f.flight\_id = bp.flight\_id

**where** bp.seat\_no **is** **null** # Только по свободным местам

**group** **by** f.flight\_id) **as** af # в рамках одного самолета

**on** f.flight\_id = af.flight\_id

**left** **join** # Соединяем с общим кол-вом мест

\_

(**select** a.aircraft\_code, |

**count**(s.seat\_no) **as** seat\_count |

**from** aircrafts a |

**left** **join** seats s |# Получение таблицы с

**on** a.aircraft\_code = s.aircraft\_code |# общим кол-вом мест

**group** **by** a.aircraft\_code) **as** sc |# во всех самолетах

**on** f.aircraft\_code = sc.aircraft\_code |

**group** **by** f.flight\_id, sc.seat\_count, |

af.available\_seats) **as** sub \_|

**where** actual\_departure **is** **not** **null** # Убираем лишние данные

**order** **by** departure\_airport, actual\_departure

**5.6 Найдите процентное соотношение перелетов по типам самолетов от общего количества. (Подзапрос, оператор Round)**

\_

**select** model, |# Выводим модель и

**round**(**count**(flight\_id) / |# отношение с округлением

**sum**(**count**(flight\_id)) **over**(), 4) \* 100 |# суммируем по окну для

**as** flights\_share \_|# работы двух ф-ий

**from** \_

(**select** \* |# Соединяем таблицы

**from** flights f |# для получения

**join** aircrafts a |# модели самолетов

**using** (aircraft\_code)) **as** sub \_|

**group** **by** model;

**5.7 Были ли города, в которые можно добраться бизнес-классом дешевле, чем эконом-классом в рамках перелета? (CTE)**

**WITH** cte\_price **AS**( \_ # Создаем представление

**select** f.flight\_id, tf.fare\_conditions, |# где оставляем только

tf.amount, a.city |# id, город, цену и

**from** ticket\_flights tf |# класс перелета. Для

**join** flights f |# этого соединяем эти

**using** (flight\_id) |# таблицы

**join** airports a |

**on** f.departure\_airport = a.airport\_code |

**group** **by** a.city, tf.amount, |# Группируем по этим

tf.fare\_conditions, f.flight\_id |# столбцам, исключая

**having** tf.fare\_conditions **in** ('Economy', 'Business')|# комфорт-класс

**order** **by** a.city) \_|

**select** **distinct** cte1.city |# Выбираем уникальные

**as** "Города, c дешевым Бизнес-Классом" |# из представления и

**from** cte\_price cte1 |# и соединяем с самим

**join** cte\_price cte2 |# собой для написания

**on** cte1.city = cte2.city \_|# необходимых условий

**where** cte1.flight\_id = cte2.flight\_id # Задаем четыре условия

**and** cte1.fare\_conditions = 'Economy' # для вывода дешевых

**and** cte2.fare\_conditions = 'Business' # перелетов

**and** cte1.amount > cte2.amount; # бизнес-классом



**Автор работы: Щербаков Н.В.**

**Июнь 2021**