Итоговая работа по курсу

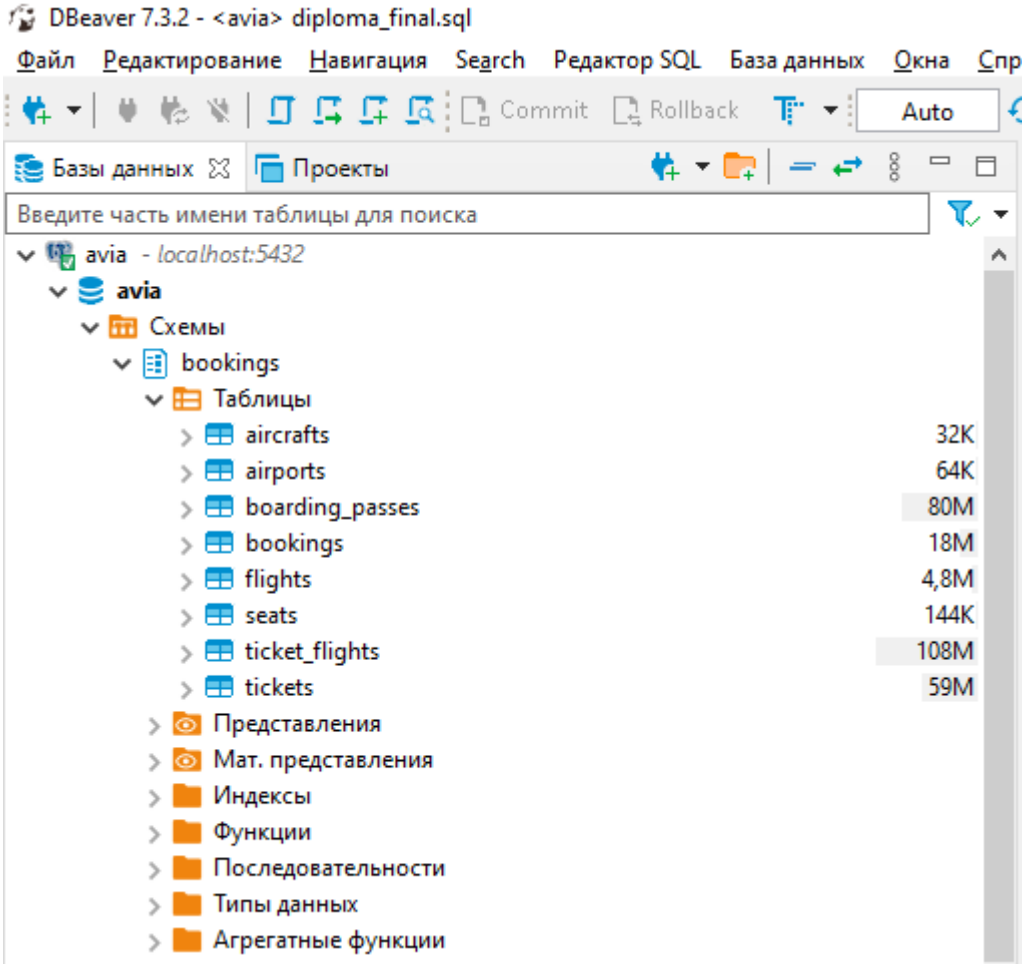
«SQL и получение данных»

Выполнила Прилукова Полина,

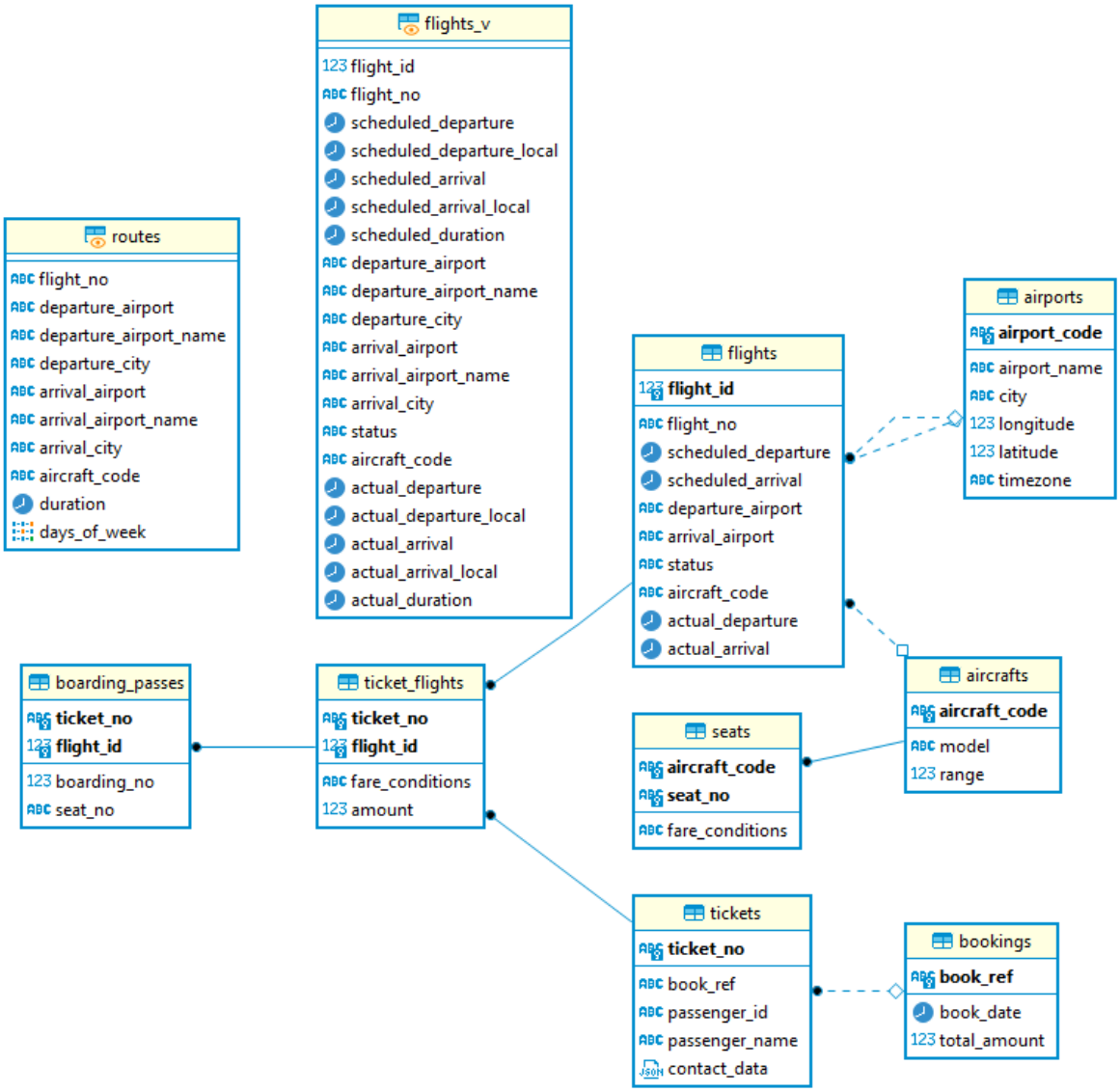
группа SQL-25

февраль 2021

1. В работе использовался локальный тип подключения



2. Скриншот ER-диаграммы из DBeaver`а согласно моего подключения



3. Краткое описание БД

База данных включает в себя следующие таблицы:

Название таблицы	Описание	Атрибуты
aircrafts	Самолеты	- Код самолета, IATA - Модель самолета - Максимальная дальность полета, км
airports	Аэропорты	- Код аэропорта - Название аэропорта - Город - Координаты аэропорта: долгота - Координаты аэропорта: широта - Временная зона аэропорта
boarding_passes	Посадочные талоны	- Номер билета - Идентификатор рейса - Номер посадочного талона - Номер места
bookings	Бронирования	- Номер бронирования - Дата бронирования - Полная сумма бронирования
flights	Рейсы	- Идентификатор рейса - Номер рейса - Время вылета по расписанию - Время прилёта по расписанию - Аэропорт отправления - Аэропорт прибытия - Статус рейса - Код самолета, IATA - Фактическое время вылета - Фактическое время прилёта
seats	Места	- Код самолета, IATA - Номер места - Класс обслуживания
ticket_flights	Перелеты	- Номер билета - Идентификатор рейса - Класс обслуживания - Стоимость перелета
tickets	Билеты	- Номер билета - Номер бронирования - Идентификатор пассажира - Имя пассажира - Контактные данные пассажира

База данных включает в себя представления:

Название	Тип	Описание	Атрибуты
flights_v	Представление	Рейсы	- Идентификатор рейса - Номер рейса - Время вылета по расписанию - Местное время вылета по расписанию в пункте отправления - Время прилёта по расписанию - Местное время прилёта по расписанию в пункте прибытия - Планируемая продолжительность полета - Код аэропорта отправления - Название аэропорта отправления - Город отправления - Код аэропорта прибытия - Название аэропорта прибытия - Город прибытия - Статус рейса - Код самолета, IATA - Фактическое время вылета - Фактическое время вылета, местное время в пункте отправления - Фактическое время прилёта - Фактическое время прилёта, местное время в пункте прибытия - Фактическая продолжительность полета
routes	Мат. представление	Маршруты	- Номер рейса - Код аэропорта отправления - Название аэропорта - Город отправления - Код аэропорта прибытия - Название аэропорта прибытия - Город прибытия - Код самолета, IATA - Продолжительность полета - Дни недели, когда выполняются рейсы

4. Развернутый анализ БД - описание таблиц, логики, связей и бизнес области.

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.

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».

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.

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

Описание таблиц:

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.
- 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).
- 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)

Основными задачами фактически любого бизнеса являются увеличение клиентской базы и грамотное распределение имеющихся ресурсов. Это же вполне применимо к аэропортам и авиакомпаниям. Исходя из общих представлений об их работе, могу предположить следующее:

- С помощью данных о бронированиях и перелетах из базы можно понять загруженность различных направлений, что позволит оперативно манипулировать расписанием рейсов, реагируя на притоки и оттоки пассажиров. Это поможет минимизировать количество малозаполненных, неэффективных рейсов на менее востребованных маршрутах и быть готовыми выделить дополнительные мощности для востребованных. Записи в базе охватывают не очень широкий временной период (если смотреть по таблице бронирований, то самая ранняя дата бронирования – это 2016-08-19, а самая поздняя - 2016-10-13, т.е. около 2х месяцев), но, если бы мы располагали аналогичными данными хотя бы в течении года, можно было бы говорить об определенной сезонности поведения пассажиров и строить прогнозные модели на будущее.

- Оценка загруженности рейсов позволит рассчитать прямые и косвенные расходы, связанные с осуществлением перелетов. Зная количество рейсов, можно рассчитывать затраты на топливо, на тех. обслуживание самолетов, на оплату труда экипажей. Зная количество пассажиров на рейсе, можно определить необходимый объем закупок продуктов питания для приготовления обедов, если это рейсы дальнего следования, и прочие аналогичные затраты.

- На основе имеющихся записей так же можно отслеживать активность отдельных пассажиров и предлагать им различные скидки и бонусы в зависимости от их активности. Можно выделить определенные группы пассажиров, которые будут особенно заинтересованы в программах лояльности от аэропорта. Например, по записям базы можно найти людей, которые совершают перелеты достаточно часто в течение длительного промежутка времени. Скорее всего это будут рабочие, командировочные или вахтовые поездки, т.е. эти пассажиры так или иначе вынуждены пользоваться авиаперелетами сейчас и в дальнейшем, поэтому им можно предложить скидку на следующий рейс, бонусные мили или другой способ снизить цену авиационных услуг. Так же в базе мы можем найти пассажиров, которые совершают цепочки перелетов, т.е. вынуждены сделать несколько пересадок, прежде чем доберутся до пункта назначения. Такие варианты перелетов всегда связаны с не самыми удобными моментами при переходе с одного рейса на следующий, зачастую с необходимостью долго ждать своего рейса в пункте пересадки. Поэтому такие люди могут быть заинтересованы в бонусах от неавиационной составляющей бизнеса – это может быть проход в VIP-зал ожидания, скидки на магазины и пункты питания в аэропорту, скидки на отели и гостиницы.

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

   В этом пункте привожу текст заданий и краткое описание логики построения запроса для их решения. Более развернутый комментарий оставляю в прилагаемом .sql файле, так же в файле указываю время, за которое запросы выполнялись на моем компьютере.

   1) В каких городах больше одного аэропорта?

      Решение: группировка таблицы airports по полю city, использование условия having, в котором указано, что количество сгруппированных записей должно быть больше 1.

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

      Условие: использовать подзапрос.

      Решение: Первый шаг – найти самолет с максимальной дальностью перелета. Это будет подзапрос. Необходимо использовать этот подзапрос в условии where во внешнем запросе, чтобы найти полеты flights, где был задействован найденный самолет, и, соответственно, аэропорты отправления этих рейсов.

   3) Вывести 10 рейсов с максимальным временем задержки вылета

      Условие: использовать оператор LIMIT.

      Решение: Выбрать из рейсов flights те, для кого возможно рассчитать время задержки вылета, найти для этих рейсов разницу между фактическим временем вылета и временем по расписанию, отсортировать записи по этому показателю (по убыванию), использовать limit, чтобы оставить 10 записей.

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

      Условие: использовать верный тип join.

      Решение: Если по брони не получен посадочный талон, то скорее всего этот рейс еще не состоялся, и его вылет запланирован на время более позднее, чем момент создания БД. Но неочевидно, могут ли быть другие причины, например, из-за технической ошибки на бронь не оформлен ни один билет и соответственно связать с талоном эту бронь нельзя. Чтобы охватить все возможные причины, достаточно из множества всех броней вычесть те, по которым посадочный талон точно получен. Идентификаторы броней с посадочными талонами можно получить подзапросом, а следующим шагом связать таблицу bookings с этим подзапросом, используя left join и условие, что id брони из подзапроса есть null. Таким образом будут просмотрены все записи таблицы bookings, и в результат попадут те, по которым посадочного талона нет.

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.

   Условие: использовать подзапросы и оконные функции.

   Решение: Одним подзапросом можно рассчитать общее число мест на каждую модель самолета. Другим подзапросом можно рассчитать общее количество мест из посадочных талонов на каждый рейс. В результате соединяем эти подзапросы с таблицей рейсов и рассчитываем количество пустых мест как разницу общего количества мест из 1го подзапроса и количества занятых мест на рейсе из 2го. Для получения накопительного итога нужно использовать оконную функцию. Построение окна будет выполнено по полям «аэропорт» и «актуальное время вылета», причем последнее должно быть приведено из формата «дата-время» к формату «дата». Сортировка в окне будет выполнена по актуальному времени вылета. По окну рассчитывается сумма занятых мест из 2го подзапроса.

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

   Условие: использование подзапросов и оператора round.

   Решение: Внешний запрос будет обращаться к таблице aircrafts для получения всех моделей самолетов. Подзапрос будет использован при расчете процента. В подзапросе будет рассчитываться количество рейсов flights, совершенных конкретной моделью самолета. Поэтому в подзапросе будет использовано условие where на модель самолета, где для сравнения берется код самолета из внешнего запроса.

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

   Условие: использование CTE.

   Решение: Можно создать 2 отдельных CTE: в одном идентификаторы рейсов и стоимости для всех перелетов бизнес-классом, в другом то же самое для перелетов экономом. Соединив эти две таблицы по идентификатору рейса при условии, что на этом рейсе стоимость бизнеса не превышала стоимость эконома, получим список рейсов, удовлетворяющих условию. И чтобы вывести ответ в требуемом формате, необходимо соединить рейсы с таблицей аэропортов и достать оттуда город аэропорта прибытия.

8) Между какими городами нет прямых рейсов?

   Условие: использование декартова произведения во FROM, своих представлений, оператора EXCEPT.

Решение: Для решения необходимо из всех возможных комбинаций двух городов исключить те, между которыми прямые рейсы есть. Все комбинации можно получить, использовав декартово произведение таблицы airports самой на себя (только там есть информация про города). Из этого произведения нужно убрать пары, в которых города одинаковы, останутся все возможные комбинации различных маршрутов. Для получения пар городов, между которыми есть прямые рейсы, надо собрать все уникальные комбинации городов отправления и прибытия из таблицы рейсов. С этой целью можно создать представление. Последним шагом будет исключение с помощью EXCEPT результата выполнения запроса представления из декартова произведения.

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

   Условие: использование оператора RADIANS или sind/cosd

   Решение: Вначале необходимо найти все уникальные комбинации прямых рейсов и самолетов, их осуществляющих. Потом для этих записей получить координаты аэропортов отправления и прибытия, рассчитать расстояние по формуле и сравнить полученное расстояние со значением атрибута “range” того самолета, который привязан к этому рейсу.