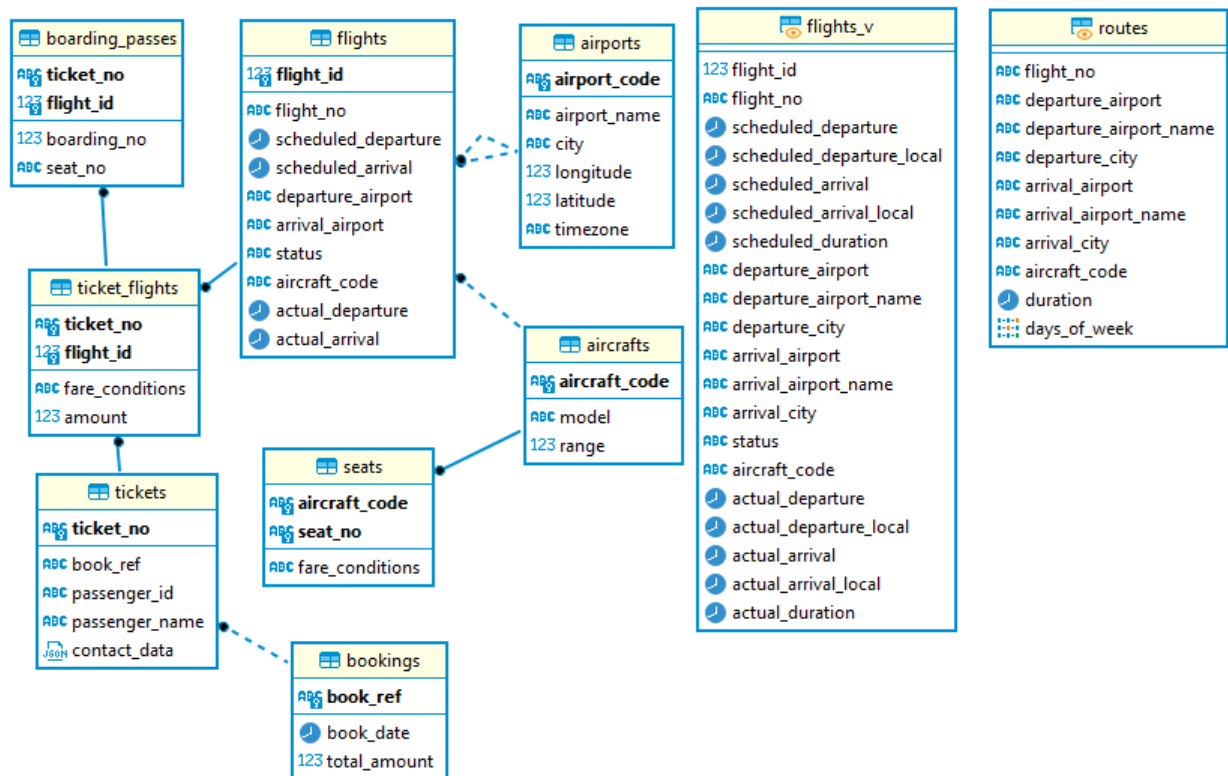


Запросы SQL к базе данных “Bookings”

Оглавление - задание

1. Скриншот ER-диаграммы из DBeaver.....	1
2. Краткое описание БД - из каких таблиц и представлений состоит.....	1
3. Развернутый анализ базы данных (БД)	2
4. Написать SQL запросы с описанием логики их выполнения и представить скрины их выполнения	3
4.1 В каких городах больше одного аэропорта?	3
4.2 В каких аэропортах есть рейсы, выполняемые самолетом с максимальной дальностью перелета?.....	4
4.3 Вывести 10 рейсов с максимальным временем задержки вылета	5
4.4 Были ли брони, по которым не были получены посадочные талоны?	6
4.5 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	7
4.6 Найдите процентное соотношение перелетов по типам самолетов от общего количества	8
4.7 Были ли города, в которые можно добраться бизнес - классом дешевле, чем эконом-классом в рамках перелета?	9
4.8 Между какими городами нет прямых рейсов?.....	9
4.9 Вычислите расстояние между аэропортами, связанными прямыми рейсами, сравните с допустимой максимальной дальностью перелетов в самолетах, обслуживающих эти рейсы	11

1. Скриншот ER-диаграммы из DBeaver



2. Краткое описание БД - из каких таблиц и представлений СОСТОИТ

База данных состоит из 8 таблиц, одного представления и одного материализованного представления.

Таблицы:

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

Представление: flights_v

Материализованное представление: routes

3. Развернутый анализ базы данных (БД)

БД относится к сфере пассажирских авиаперевозок. Основной сущностью является бронирование (**таблица bookings**).

В одно бронирование можно включить несколько пассажиров, каждому из которых выписывается отдельный билет (**таблица tickets**). Билет имеет уникальный номер и содержит информацию о пассажире.

Билет включает один или несколько перелетов (**таблица ticket_flights**).

Каждый рейс (**таблица flights**) следует из одного аэропорта (**таблица airports**) в другой. Рейсы с одним номером имеют одинаковые пункты вылета и назначения, но будут отличаться датой отправления.

При регистрации на рейс пассажиру выдаётся посадочный талон (**таблица 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 (**таблица seats**) в самолете и их распределение по классам обслуживания зависит от модели самолета (**таблица aircrafts**), выполняющего рейс. Предполагается, что каждая модель самолета имеет только одну компоновку салона.

Связи между таблицами базы данных представлены в следующей таблице.

Таблица	Первичный ключ	Внешний ключ	Связанная сущность
aircrafts	aircraft_code	--	--
airports	airport_code	--	--
boarding_passes	ticket_no flight_id	ticket_no	ticket_flights
		flight_id	ticket_flights
bookings	book_ref	--	--
flights	flight_id	aircraft_code	aircrafts
		arrival_airport	airports
		departure_airport	airports
seats	aircraft_code seat_no	aircraft_code	aircrafts
ticket_flights	ticket_no flight_id	flight_id	flights
		ticket_no	tickets
tickets	ticket_no	book_ref	bookings

У каждого рейса есть запланированные дата и время вылета (scheduled_departure) и прибытия (scheduled_arrival). 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 отменён.

Используя БД, можно получать разнообразную информацию, связанную с авиа рейсами: стоимость билетов и бронирований, заполняемость самолетов, время вылетов и прилетов, расстояние между аэропортами, количество перевезенных пассажиров и многое другое. В зависимости от поставленных целей информация может быть представлена в различных комбинациях и выборках, временных срезах, в привязке к конкретным самолетам, рейсам, аэропортам и т.д.

4. Написать SQL запросы с описанием логики их выполнения и представить скрины их выполнения

4.1 В каких городах больше одного аэропорта?

```
/* - группируем аэропорты по городам и вычисляем их количество в каждом городе
* - с помощью оператора HAVING оставляем в запросе те города, где аэропортов
* больше одного
*/
SELECT city "Город", COUNT(airport_code) "Количество аэропортов"
FROM airports a
GROUP BY city
HAVING COUNT(airport_code) > 1;
```

	Город	Количество аэропортов
1	Ульяновск	2
2	Москва	3

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











Можно полагать, что если самолет прилетел в аэропорт, то он из него улетит, поэтому для решения этой задачи было бы достаточно рассмотреть либо поле `flights.departure_airport`, либо `flights.arrival_airport`. Однако запрос должен быть универсальным. Предположим, в БД есть всего одна запись о перелете, тогда при таком подходе мы получим только один аэропорт, а должно быть два – прилета и вылета. Поэтому будем использовать оба поля.

```
/* - соединяем таблицы airports, flights и aircrafts
* - находим аэропорты вылета
* - с помощью оператора WHERE и подзапроса оставляем записи, удовлетворяющие
*    условию
* - аналогично с аэропортами прилета
* - с помощью оператора UNION объединяем два запроса. Такое объединение уберет
*    дубли
*/
SELECT a.airport_code , a.airport_name
FROM airports a JOIN flights f ON f.departure_airport = a.airport_code
      JOIN aircrafts ac ON ac.aircraft_code = f.aircraft_code
WHERE ac."range" = (select MAX ("range") FROM aircrafts)
UNION
SELECT a2.airport_code , a2.airport_name
FROM airports a2 JOIN flights f2 ON f2.arrival_airport = a2.airport_code
      JOIN aircrafts ac2 ON ac2.aircraft_code = f2.aircraft_code
WHERE ac2."range" = (select MAX ("range") FROM aircrafts);
```

	ABC airport_code	ABC airport_name
1	SVX	Кольцово
2	VKO	Внуково
3	OVB	Толмачёво
4	SVO	Шереметьево
5	DME	Домодедово
6	AER	Сочи
7	PEE	Пермь

4.3 Вывести 10 рейсов с максимальным временем задержки вылета

```
/* - формируем список перелетов с указанием времени задержки
* - используя оператор WHERE, оставляем только вылетевшие рейсы
* - сортируем список по убыванию
* - с помощью оператора LIMIT оставляем первые 10 строк
*/
SELECT f.flight_id, f.flight_no, f.actual_departure, f.scheduled_departure,
       f.actual_departure - f.scheduled_departure AS delay
FROM flights f
WHERE f.actual_departure IS NOT NULL
ORDER BY delay DESC
LIMIT 10;
```

	 flight_id 	 flight_no 	 actual_departure 	 scheduled_departure 	 delay 
1	14 750	PG0589	2016-09-26 19:07:00.000 +0300	2016-09-26 14:30:00.000 +0300	04:37:00
2	1 408	PG0164	2016-09-26 18:53:00.000 +0300	2016-09-26 14:25:00.000 +0300	04:28:00
3	24 253	PG0364	2016-09-16 15:12:00.000 +0300	2016-09-16 10:45:00.000 +0300	04:27:00
4	22 778	PG0568	2016-10-11 19:35:00.000 +0300	2016-10-11 15:15:00.000 +0300	04:20:00
5	2 852	PG0454	2016-09-30 13:23:00.000 +0300	2016-09-30 09:05:00.000 +0300	04:18:00
6	21 684	PG0096	2016-10-01 19:53:00.000 +0300	2016-10-01 15:35:00.000 +0300	04:18:00
7	11 426	PG0166	2016-10-10 17:51:00.000 +0300	2016-10-10 13:35:00.000 +0300	04:16:00
8	9 891	PG0278	2016-09-13 17:36:00.000 +0300	2016-09-13 13:20:00.000 +0300	04:16:00
9	13 645	PG0564	2016-10-08 12:44:00.000 +0300	2016-10-08 08:30:00.000 +0300	04:14:00
10	4 781	PG0669	2016-09-16 19:23:00.000 +0300	2016-09-16 15:15:00.000 +0300	04:08:00

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

Если подойти к заданию формально, то можно построить запрос, подсчитывающий количество перелетов во всех бронях, на которые не получены посадочные талоны, или количество таких броней. Но результат будет предсказуемым – положительным, поскольку на момент фиксации БД в ней есть рейсы, на которые продолжается регистрация или она вовсе еще не объявлена. Поэтому смотреть наличие таких броней имеет смысл только по состоявшимся рейсам.

```
/* - выберем все брони и сопоставим им полученные посадочные талоны с учетом того,
 *   что одно бронирование может включать несколько билетов, а билет может включать
 *   несколько перелетов. Чтобы получить все строки, а не только имеющие
 *   соответствующие им посадочные талоны используем оператор LEFT JOIN
 * - с помощью оператора WHERE получим записи, в которых посадочные талоны не
 *   получены и вылет рейса совершен
 * - посчитаем их количество и сделаем вывод
 */
SELECT CASE WHEN COUNT(t.book_ref) > 0
            THEN 'Есть брони, по которым не получены посадочные талоны'
            ELSE 'Нет броней, по которым не получены посадочные талоны' END
        "Результат"
FROM tickets t JOIN ticket_flights tf ON tf.ticket_no = t.ticket_no
              JOIN flights f ON f.flight_id = tf.flight_id
              LEFT JOIN boarding_passes bp
                    ON bp.ticket_no = tf.ticket_no
                    AND bp.flight_id = tf.flight_id
WHERE bp.boarding_no IS NULL AND f.actual_departure IS NOT NULL;
```

ABC Результат
Нет броней, по которым не получены посадочные талоны

4.5 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

```

/* - соединим таблицы flight и seats для получения всех мест рейса
* - добавим к соединению таблицу boarding_passes, используя оператор LEFT JOIN,
* чтобы в итоговой выборке остались все места, а не только занятые
* - с помощью оператора WHERE оставим только состоявшиеся рейсы
* - сгруппируем данные по flight_id для получения нужных данных и применим
* групповые операции
* - из полученного запроса сделаем CTE и применим к нему оконную функцию SUM с
* разделением по аэропорту и дате актуального отправления и сортировкой по
* актуальному времени вылета
*/

WITH departures AS (
    SELECT f.flight_id ,
           f.departure airport,
           f.actual_departure ,
           COUNT(bp.seat_no) passengers,
           ROUND((COUNT (s.seat_no) - COUNT(bp.seat_no))*1.0/COUNT (s.seat_no)*100, 2)
             "free_seats %",
           STRING_AGG(CASE WHEN bp.seat_no IS NULL THEN s.seat_no END, ', ') free_seats
    FROM flights f JOIN seats s USING (aircraft code)
      LEFT JOIN boarding_passes bp USING (flight_id, seat_no)
    WHERE f.actual_departure IS NOT NULL
    GROUP BY f.flight_id
)
SELECT departure_airport,
       flight_id, actual_departure,
       free_seats, "free_seats %",
       SUM(passengers) OVER (
           PARTITION BY departure_airport,
                        DATE_PART('year', actual_departure) * 10000 +
                        DATE_PART('month', actual_departure) * 100 +
                        DATE_PART('day', actual_departure)
           ORDER BY actual_departure) "Вылетело пассажиров с начала дня"
FROM departures d
ORDER BY departure_airport, actual_departure;

```


	asc departure_airport	flight_id	actual_departure	asc free_seats	free_seats %	Вылетело пассажиров с начала дня
1	AAQ	21 103	2016-09-13 11:35:00.000 +0300	14C, 2F, 6C, 6A, 12D, 18E, 1D, 19F, 17F, 19D, 14A, 15C, 4A, 15F, 18A, 15I	96,91	3
2	AAQ	20 993	2016-09-13 12:08:00.000 +0300	16E, 1A, 20D, 20E, 21A, 21B, 22E, 5D, 6F, 7D, 7E, 7A, 11D, 15E, 19C, 6B, 1	60,77	54
3	AAQ	21 096	2016-09-14 11:29:00.000 +0300	7C, 4A, 6C, 8C, 6A, 15F, 15D, 11F, 18C, 17C, 12F, 10D, 13A, 12A, 13D, 8C	96,91	3
4	AAQ	20 982	2016-09-14 12:07:00.000 +0300	13F, 16B, 14E, 13B, 20A, 22C, 9B, 2D, 19E, 16E, 11D, 11F, 9F, 17C, 13D, 1E	61,54	53
5	AAQ	21 049	2016-09-15 08:14:00.000 +0300	19A, 16D, 12C, 11B, 11E, 14D, 13C, 15B, 16A, 16E, 17A, 17B, 19B, 19E, 1A	100	0
6	AAQ	21 084	2016-09-15 11:26:00.000 +0300	4D, 19A, 16A, 3F, 4C, 16D, 14F, 1F, 10C, 13E, 18F, 16F, 12E, 10F, 15A, 18E	94,85	5
7	AAQ	21 041	2016-09-15 12:09:00.000 +0300	15F, 18A, 19C, 17D, 18B, 17C, 18C, 20F, 22B, 3A, 3C, 9C, 6E, 6D, 10D, 13,	61,54	55
8	AAQ	21 073	2016-09-16 11:26:00.000 +0300	4E, 8A, 6E, 9F, 3C, 20F, 18C, 17C, 9C, 6D, 3A, 2C, 14E, 2D, 20A, 16C, 12A	96,91	3
9	AAQ	21 014	2016-09-16 12:05:00.000 +0300	9D, 2A, 1F, 7F, 12E, 23D, 5C, 5A, 14B, 14F, 12B, 19C, 15F, 17D, 4C, 3F, 16	62,31	52
10	AAQ	21 065	2016-09-17 11:27:00.000 +0300	11C, 11A, 15C, 12D, 1D, 14A, 4A, 19F, 14C, 6A, 6C, 2F, 19D, 18E, 17F, 13I	95,88	4
11	AAQ	21 002	2016-09-17 12:07:00.000 +0300	23B, 20C, 19C, 21E, 23E, 12B, 18F, 16F, 14F, 10C, 15A, 5E, 8E, 17E, 21F, 8I	61,54	54
12	AAQ	21 058	2016-09-18 11:26:00.000 +0300	1D, 12D, 17F, 6C, 6A, 14C, 19D, 19F, 18E, 15C, 4A, 2F, 16C, 2D, 2C, 20A,	91,75	8
13	AAQ	20 981	2016-09-18 12:08:00.000 +0300	15D, 17C, 20F, 9C, 22B, 8A, 10D, 11F, 12F, 13D, 20C, 21E, 23E, 7B, 4A, 14	58,46	62
14	AAQ	21 061	2016-09-19 11:28:00.000 +0300	8C, 6C, 6A, 7C, 13D, 12F, 10D, 13A, 11F, 18C, 17C, 12A, 14A, 18E, 2F, 11I	94,85	5
15	AAQ	20 999	2016-09-19 12:10:00.000 +0300	16F, 18F, 23A, 2A, 5A, 7F, 12B, 12E, 9D, 13E, 14B, 14F, 14E, 16B, 9B, 22C,	58,46	59
16	AAQ	21 081	2016-09-20 11:28:00.000 +0300	1C, 15E, 11D, 20E, 17A, 20D, 19E, 6F, 7D, 5D, 7E, 5F, 6C, 18E, 14C, 8C, 2F	75,26	24
17	AAQ	21 039	2016-09-20 12:07:00.000 +0300	5E, 8F, 9A, 22F, 1F, 5A, 7F, 10C, 14F, 15A, 12E, 10E, 1A, 6F, 17A, 11D, 21E	41,54	100
18	AAQ	21 108	2016-09-21 11:27:00.000 +0300	12C, 13C, 11E, 4D, 14D, 4C, 3F, 19A, 4F, 9D, 7F, 8A, 3A, 6E, 6D, 3C, 4E, 1I	69,07	30
19	AAQ	21 006	2016-09-21 14:48:00.000 +0300	2A, 1F, 7F, 5C, 5A, 22F, 23D, 17D, 6B, 19C, 2F, 7C, 4A, 6C, 8C, 14A, 11F,	36,92	112
20	AAQ	21 046	2016-09-22 08:14:00.000 +0300	8D, 8E, 8F, 21F, 9A, 18D, 5B, 17E, 9E, 5E, 1F, 22F, 5C, 9D, 4B, 23A, 23D, 7I	100	0
21	AAQ	21 109	2016-09-22 11:26:00.000 +0300	11A, 11C, 8F, 8D, 8E, 13C, 12C, 11E, 14D, 11D, 10E, 15E, 3F, 4D, 4C, 19C,	52,58	46
22	AAQ	20 986	2016-09-22 15:21:00.000 +0300	21B, 22E, 5F, 20E, 7E, 14E, 22C, 13B, 16B, 20B, 16A, 15F, 20C, 18A, 15D, 2	26,92	141
23	AAQ	21 090	2016-09-23 11:26:00.000 +0300	17E, 9A, 8D, 18D, 8E, 10F, 12E, 2A, 15A, 5A, 14F, 13C, 16D, 12C, 15E, 20C	34,02	64
24	AAQ	21 029	2016-09-23 12:09:00.000 +0300	9B, 2F, 15C, 19D, 14C, 18E, 12D, 14A, 8E, 9A, 21A, 20D, 5D, 1C, 22E, 7D,	26,15	160
25	AAQ	21 078	2016-09-24 11:28:00.000 +0300	10E, 6E, 9F, 16C, 2D, 18C, 12F, 13D, 10D, 12C, 16A, 13C, 17E, 11C, 1F, 15	29,9	68
26	AAQ	20 995	2016-09-24 12:07:00.000 +0300	17D, 10C, 12B, 22A, 20E, 5F, 15E, 7C, 4A, 5C	7,69	188
27	AAQ	21 068	2016-09-25 11:30:00.000 +0300	17F, 6C, 8C, 11C, 13F, 9C, 8D, 9A, 4C, 17D, 16E, 11D	12,37	85
28	AAQ	21 040	2016-09-25 12:11:00.000 +0300	1D, 17F, 4A, 12D, 15C, 10E, 21A, 22D, 7A, 4F, 5A, 16F, 12E, 4E, 12F, 20F, i	20	189
29	AAQ	21 052	2016-09-26 11:27:00.000 +0300	17C, 6D, 15C, 11C, 11A, 1C, 15E, 7D, 10E, 5C, 12E	11,34	86
30	AAQ	21 027	2016-09-26 12:09:00.000 +0300	17D, 23F, 12B, 10D, 9F, 17C, 2C, 16C, 9B, 7D, 20E, 17F, 19F, 7C, 12D, 14A	14,62	197
31	AAQ	21 057	2016-09-27 11:30:00.000 +0300	13E, 1F, 7E, 8E, 15E, 10E, 14A	7,22	90
32	AAQ	21 028	2016-09-27 12:09:00.000 +0300	4A, 11A, 21E, 9E, 12E, 11D	4,62	214
33	AAQ	21 060	2016-09-28 11:27:00.000 +0300	[NULL]	0	97

4.6 Найдите процентное соотношение перелетов по типам самолетов от общего количества

/* - соединим таблицы flights и aircrafts
 * - составим подзапрос для подсчета общего количества рейсов
 * - сгруппируем записи и получим требуемые данные с использованием подзапроса
 */

```

SELECT a.aircraft_code , model,
       ROUND(COUNT(a.aircraft_code)*100.0 /
             (SELECT COUNT(f.flight_id)
              FROM flights f), 3) "Доля типа смаолета, %"
FROM flights f JOIN aircrafts a USING(aircraft_code)
GROUP BY a.aircraft_code, model;

```

	aircraft_code	model	Доля типа смаолета, %
1	CN1	Cessna 208 Caravan	27,997
2	CR2	Bombardier CRJ-200	27,318
3	763	Boeing 767-300	3,686
4	773	Boeing 777-300	1,842
5	319	Airbus A319-100	3,741
6	733	Boeing 737-300	3,847
7	SU9	Sukhoi SuperJet-100	25,676
8	321	Airbus A321-200	5,894

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

```
/* - сгруппируем таблицу ticket_flights по рейсам (перелетам)
* - с помощью оператора CASE получим максимальную стоимость эконом-класса и
* минимальную стоимость бизнес-класса в рамках перелета и включим их в
* условие отбора оператором HAVING
* - на основе полученного запроса сделаем СТЕ, который в случае пустого
* результата выдаст сообщение об отсутствии таких перелетов,
* иначе - сообщение о наличии
*/
WITH price AS (
    SELECT flight_id
    FROM ticket_flights
    GROUP BY flight_id
    HAVING MAX(CASE WHEN fare_conditions = 'Economy' THEN amount ELSE NULL END) >
           MIN(CASE WHEN fare_conditions = 'Business' THEN amount ELSE NULL END)
)
SELECT CASE WHEN (SELECT * FROM price) IS NULL
    THEN 'Нет городов, в которые можно добраться бизнес - классом дешевле, ' ||
         ' чем эконом-классом в рамках перелета'
    ELSE 'Есть города, в которые можно добраться бизнес - классом дешевле, ' ||
         ' чем эконом-классом в рамках перелета' END
    "Результат";
```

ABC Результат

Нет городов, в которые можно добраться бизнес - классом дешевле, чем эконом-классом в рамках перелета

4.8 Между какими городами нет прямых рейсов?

```
/* - на основании таблицы airports построим декартово множество городов, исключив из
* него пересечение города с самим собой
* - на основании таблиц flights и airports найдем города, между которыми есть прямые
* рейсы
* - с помощью оператора EXCEPT исключим из первой выборки строки, содержащиеся во
* второй выборке. Этот оператор уберет дубли.
* - отсортируем результат по алфавиту
*/
SELECT DISTINCT a.city city_from, a2.city city_to
FROM airports a CROSS JOIN airports a2
WHERE a.city != a2.city
    EXCEPT
SELECT DISTINCT a3.city , a4.city
FROM flights f JOIN airports a3 ON a3.airport_code = f.departure_airport
    JOIN airports a4 ON a4.airport_code = f.arrival_airport
ORDER BY city_from, city_to;
```

	ABC city_from	ABC city_to
1	Абакан	Анадырь
2	Абакан	Анапа
3	Абакан	Астрахань
4	Абакан	Барнаул
5	Абакан	Белгород
6	Абакан	Белоярский
7	Абакан	Благовещенск
8	Абакан	Братск
9	Абакан	Брянск
10	Абакан	Бугульма
11	Абакан	Владивосток
12	Абакан	Владикавказ
13	Абакан	Волгоград
14	Абакан	Воркута
15	Абакан	Воронеж
16	Абакан	Геленджик
17	Абакан	Горно-Алтайск
18	Абакан	Екатеринбург
19	Абакан	Иваново
20	Абакан	Ижевск
21	Абакан	Иркутск
22	Абакан	Йошкар-Ола
23	Абакан	Казань
24	Абакан	Калининград
25	Абакан	Калуга
26	Абакан	Кемерово
27	Абакан	Киров
28	Абакан	Когалым
29	Абакан	Комсомольск-т

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

Для вычисления расстояния будем использовать формулу:

$$d = 2r \arcsin \left(\sqrt{\sin^2 \left(\frac{\varphi_2 - \varphi_1}{2} \right) + \cos(\varphi_1) \cos(\varphi_2) \sin^2 \left(\frac{\lambda_2 - \lambda_1}{2} \right)} \right),$$

где

d - расстояние

r - средний радиус Земли (6371 км)

φ_1 и φ_2 - широта первой и второй точки

λ_1 и λ_2 - долгота первой и второй точки

```
/*
 * - Соединяем таблицы flights, airports и aircrafts
 * - Получаем данные и рассчитываем нужные параметры. Параметр "соответствие дальности"
 *   принимает значение True, если максимальная дальность самолета больше расстояния между
 *   аэропортами, и False - в противном случае
 * - Сортируем результат по городам в алфавитном порядке
 */
SELECT DISTINCT a.city "город вылета",
a.airport_name "аэропорт вылета",
a2.city "город прилета",
a2.airport_name "аэропорт прилета",
range,
2 * 6371 * ASIN(SQRT((SIND((a2.latitude - a.latitude) / 2)) ^ 2 +
COSD(a.latitude) * COSD(a2.latitude) * (SIND((a2.longitude - a.longitude) / 2)) ^ 2))
"расстояние",
range > 2 * 6371 * ASIN(SQRT((SIND((a2.latitude - a.latitude) / 2)) ^ 2 + COSD(a.latitude)
* COSD(a2.latitude) * (SIND((a2.longitude - a.longitude) / 2)) ^ 2))
"соответствие дальности"
FROM flights f JOIN airports a ON a.airport_code = f.departure_airport
JOIN airports a2 ON a2.airport_code = f.arrival_airport
JOIN aircrafts ac ON ac.aircraft_code = f.aircraft_code
ORDER BY "город вылета", "город прилета";
```

	авс город вылета	авс аэропорт вылета	авс город прилета	авс аэропорт прилета	123 range	123 расстояние	соответствие дальности
1	Абакан	Абакан	Архангельск	Талаги	6 700	3 041,97	[v]
2	Абакан	Абакан	Грозный	Грозный	4 200	3 484,15	[v]
3	Абакан	Абакан	Кызыл	Кызыл	1 200	307,01	[v]
4	Абакан	Абакан	Москва	Домодедово	6 700	3 366,3	[v]
5	Абакан	Абакан	Новосибирск	Толмачёво	1 200	582,7	[v]
6	Абакан	Абакан	Томск	Богашёво	1 200	490,53	[v]
7	Анадырь	Анадырь	Москва	Внуково	6 700	6 220,25	[v]
8	Анадырь	Анадырь	Москва	Домодедово	6 700	6 226,05	[v]
9	Анадырь	Анадырь	Москва	Шереметьево	6 700	6 177,08	[v]
10	Анадырь	Анадырь	Хабаровск	Хабаровск-Новый	6 700	3 074,2	[v]
11	Анапа	Витязево	Белгород	Белгород	3 000	629,86	[v]
12	Анапа	Витязево	Москва	Шереметьево	4 200	1 219,88	[v]
13	Анапа	Витязево	Новокузнецк	Спиченково	4 200	3 634,02	[v]
14	Архангельск	Талаги	Абакан	Абакан	6 700	3 041,97	[v]
15	Архангельск	Талаги	Иркутск	Иркутск	6 700	3 778,09	[v]
16	Архангельск	Талаги	Москва	Домодедово	2 700	1 005,09	[v]
17	Архангельск	Талаги	Нарьян-Мар	Нарьян-Мар	2 700	663,02	[v]
18	Архангельск	Талаги	Пермь	Пермь	1 200	1 096,68	[v]
19	Архангельск	Талаги	Томск	Богашёво	2 700	2 552,68	[v]
20	Архангельск	Талаги	Тюмень	Рощино	2 700	1 554,15	[v]