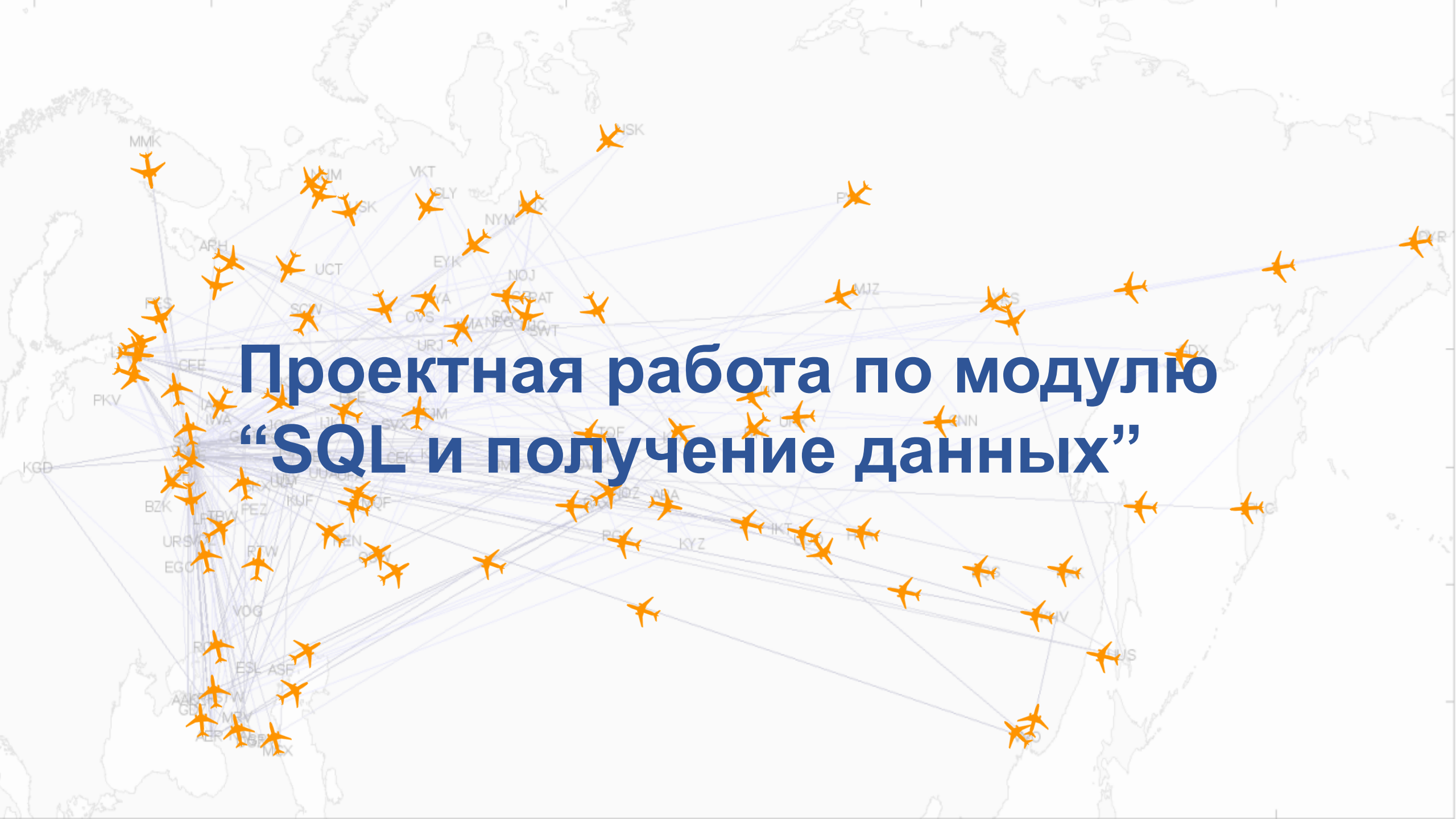
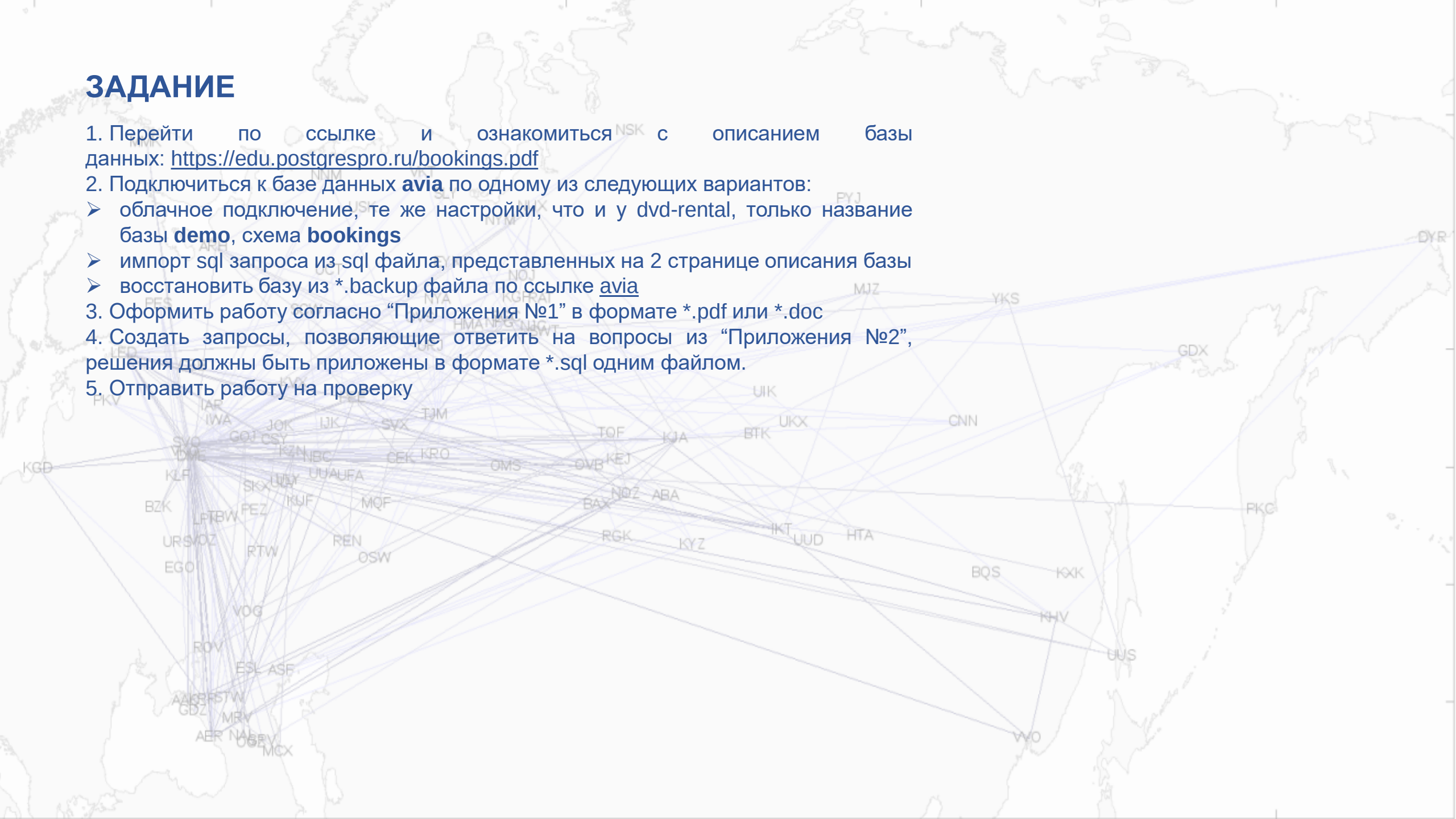


Проектная работа по модулю “SQL и получение данных”

ЗАДАНИЕ

1. Перейти по ссылке и ознакомиться с описанием базы данных: https://edu.postgrespro.ru/bookings.pdf
2. Подключиться к базе данных **avia** по одному из следующих вариантов:
   - облачное подключение, те же настройки, что и у dvd-rental, только название базы **demo**, схема **bookings**
   - импорт sql запроса из sql файла, представленных на 2 странице описания базы
   - восстановить базу из *.backup файла по ссылке avia
3. Оформить работу согласно “Приложения №1” в формате *.pdf или *.doc
4. Создать запросы, позволяющие ответить на вопросы из “Приложения №2”, решения должны быть приложены в формате *.sql одним файлом.
5. Отправить работу на проверку



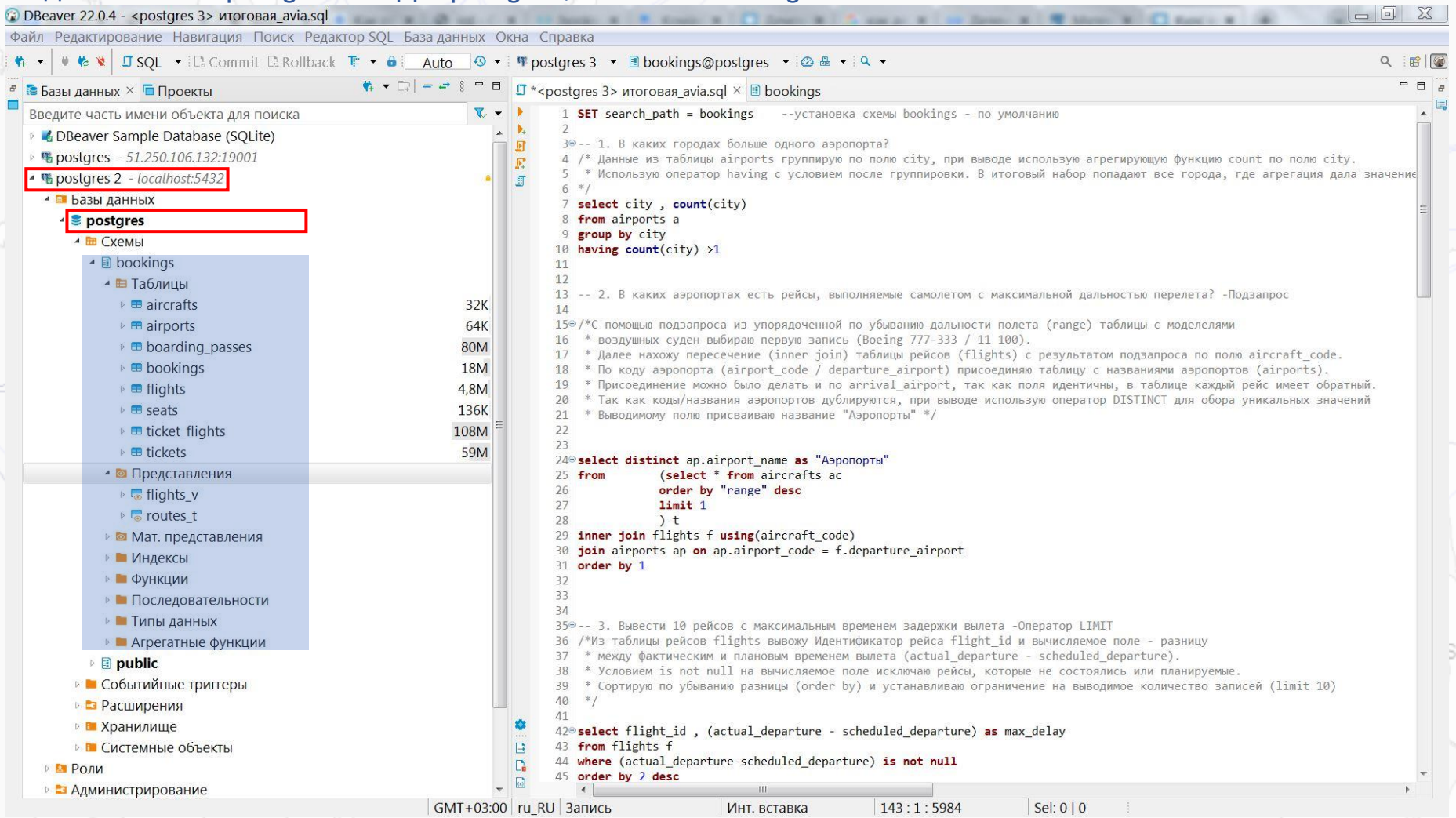
Приложение №1

Итоговая работа

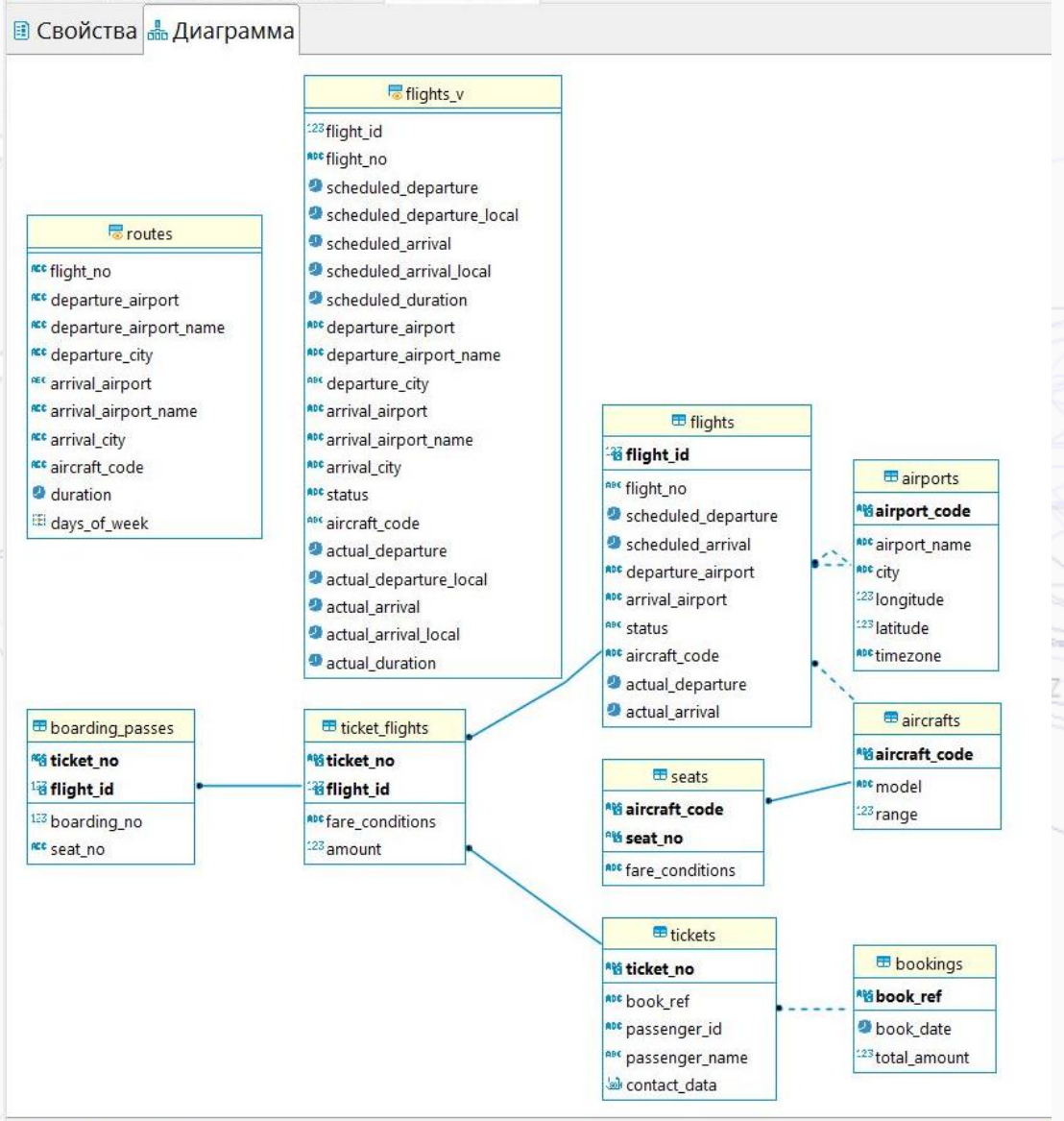
1. В работе использовался локальный тип подключения.

Ниже приводится скриншот подключения.

Подключение – postgres 2. БД – postgres, Схема – bookings.



2. ER-диаграмма.



3. БД состоит из восьми таблиц, одного представления и одного материализованного представления.

Элементы БД:

	Название	Вид
1	flights	таблица
2	ticket_flights	таблица
3	bording_passes	таблица
4	tickets	таблица
5	bookings	таблица
6	aircrafts	таблица
7	seats	таблица
8	airports	таблица
9	flights_v	представление
10	routes	мат. представление

В процессе работы создается представление routes_v, назначение и описание которого приводится в задании 8 Приложения 2.

4. В таблице приводится детальная информация о существующих в БД отношениях, полях, первичных и внешних ключах, связывающих отношения.

	Название	Вид	Описание отношений	Название полей отношений	Первичный ключ	Внешний ключ
1	flights	таблица	Рейсы	➢ Идентификатор рейса ➢ Номер рейса ➢ Время вылета, план ➢ Время прилёта , план ➢ Аэропорт вылета ➢ Аэропорт прилёта ➢ Статус рейса ➢ Код самолета ➢ Время вылета, факт ➢ Время прилёта, факт	Суррогатный первичный ключ PRIMARY KEY, btree (flight_id) UNIQUE CONSTRAINT, btree (flight_no, scheduled_departure)	FOREIGN KEY (aircraft_code) FOREIGN KEY (arrival_airport) FOREIGN KEY (departure_airport)
2	ticket_flights	таблица	Перелёты	➢ Номер билета ➢ Идентификатор рейса ➢ Класс обслуживания ➢ Стоимость перелета	Составной первичный ключ PRIMARY KEY, btree (ticket_no, flight_id)	FOREIGN KEY (flight_id) FOREIGN KEY (ticket_no)

Приложение №1

	Название	Вид	Описание отношений	Название полей отношений	Первичный ключ	Внешний ключ
3	bording_passes	таблица	Посадочные талоны	➢ Номер билета ➢ Идентификатор рейса ➢ Номер посадочного талона ➢ Номер места	Составной ключ PRIMARY KEY, btree (ticket_no, flight_id) UNIQUE CONSTRAINT, btree (flight_id, boarding_no) UNIQUE CONSTRAINT, btree (flight_id, seat_no)	FOREIGN KEY (ticket_no, flight_id)
4	tickets	таблица	Билеты	➢ Номер билета ➢ Номер бронирования ➢ Идентификатор пассажира ➢ Имя пассажира ➢ Контактные данные пассажира	Простой ключ PRIMARY KEY, btree (ticket_no)	FOREIGN KEY (book_ref)
5	bookings	таблица	Бронирования	➢ Номер бронирования ➢ Дата бронирования ➢ Полная сумма бронирования	Простой ключ PRIMARY KEY, btree (book_ref)	–
6	aircrafts	таблица	Самолёты	➢ Код самолёта, ➢ модель самолёта, ➢ максимальная дальность полёта, км	Простой ключ PRIMARY KEY, btree (aircraft_code)	FOREIGN KEY (aircraft_code)

Приложение №1

	Название	Вид	Описание отношений	Название полей отношений	Первичный ключ	Внешний ключ
7	seats	таблица	Места	➢ Код самолета ➢ Номер места ➢ Класс обслуживания	Составной ключ PRIMARY KEY, btree (aircraft_code, seat_no)	FOREIGN KEY (aircraft_code)
8	airports	таблица	Аэропорты	➢ Код аэропорта ➢ Название аэропорта ➢ Город ➢ Долгота ➢ Широта ➢ Временная зона	PRIMARY KEY, btree (airport_code)	FOREIGN KEY (arrival_airport) FOREIGN KEY (departure_airport)
9	flights_v	представление	Рейсы	Расширение таблицы flights: ➢ расшифровка данных об аэропортах, ➢ местное время вылета и прилёта (план/факт), ➢ продолжительность полета (план/факт)	–	–
10	routes	мат. представление	Маршруты	Удаление избыточности таблицы flights. Информация о маршрутах, без учета конкретных дат рейсов.	–	–

Демонстрационная база содержит временной «срез» данных. Позиция «среза» сохранена в функции bookings.now(). Значение этой функции определяет версию демонстрационной базы данных.
Актуальная версия — от 13.10.2016.
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, так что невозможно однозначно найти все билеты одного человека; для простоты 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).

Примеры задач, решаемых с помощью рассматриваемой БД по авиаперевозкам:

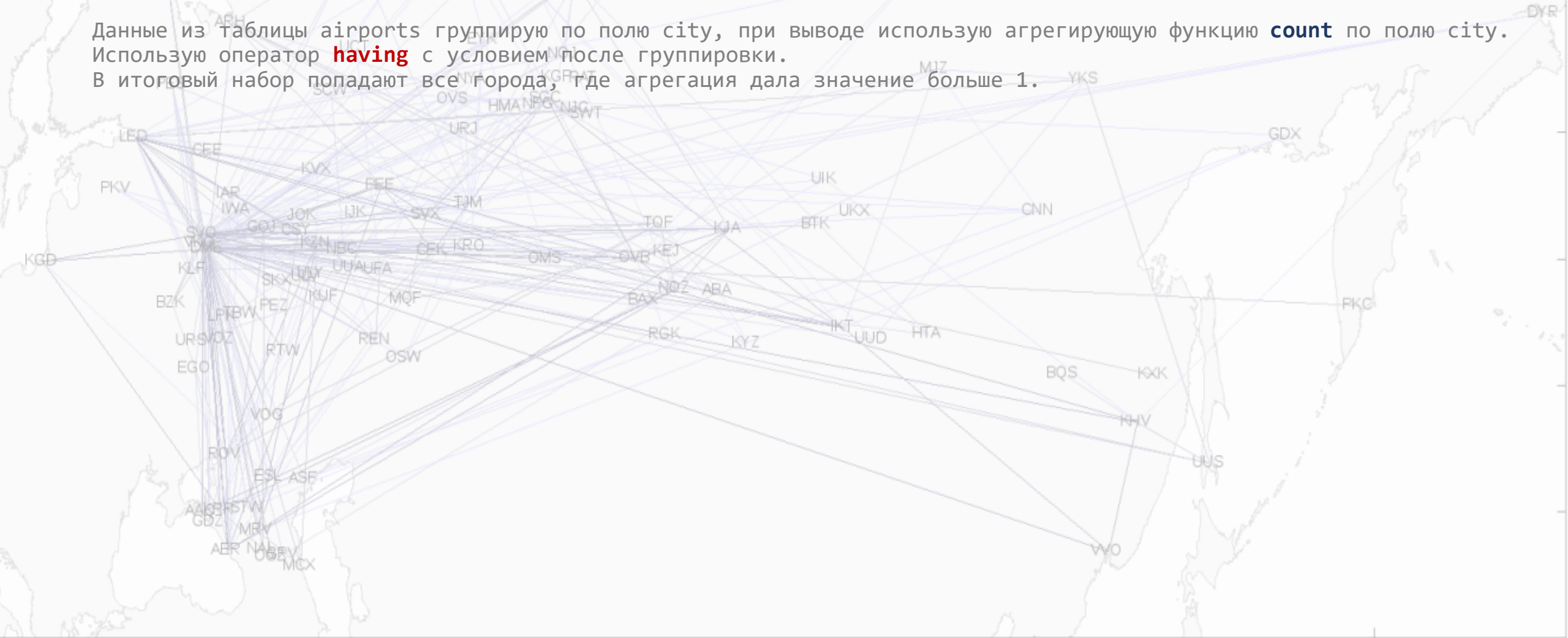
- оптимизация маршрутов авиаперевозок
- планирование штатного техобслуживания воздушных судов после определенного расстояния/времени в пути
- прогнозирование задержек рейсов с помощью Big Data
- прогнозирование спроса на авиаперевозки.

Далее в Приложении 2 приводится список SQL запросов с описанием логики их выполнения.
Перелет, рейс = flight_id

№	Вопрос	Использовать	Запрос

1	В каких городах больше одного аэропорта?	-	`select city , count(city)`
			`from airports a`
			`group by city`
			`having count(city) >1`

Данные из таблицы airports группирую по полю city, при выводе использую агрегирующую функцию **count** по полю city.
Использую оператор **having** с условием после группировки.
В итоговый набор попадают все города, где агрегация дала значение больше 1.



№	Вопрос	Использовать	Запрос

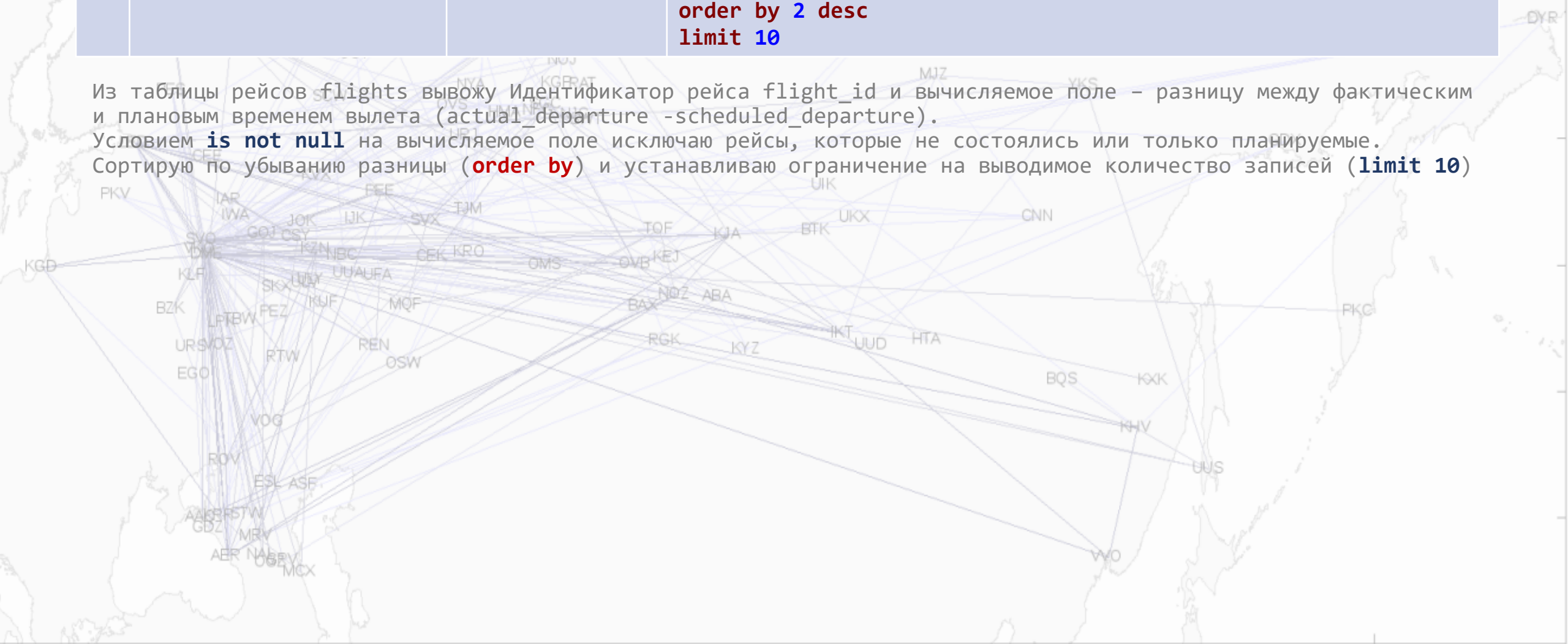
2	В каких аэропортах есть рейсы, выполняемые самолетом с максимальной дальностью перелета?	Подзапрос	`select distinct ap.airport_name as "Аэропорты"`
			`from (select * from aircrafts ac`
			`order by "range" desc`
			`limit 1`
			`) t`
			`inner join flights f using(aircraft_code)`
			`join airports ap on ap.airport_code = f.departure_airport`
			`order by 1`

С помощью подзапроса из упорядоченной по убыванию дальности полета (range) таблицы с моделями воздушных судов выбираю первую запись (Boeing 777-333 / 11 100).
Далее нахожу пересечение (**inner join**) таблицы рейсов (flights) с результатом подзапроса по полю aircraft_code.
По коду аэропорта (airport_code / departure_airport) присоединяю таблицу с названиями аэропортов (airports).
Присоединение можно было делать и по arrival_airport, так как наборы значений полей идентичны, в таблице каждый рейс имеет обратный.
Так как коды/названия аэропортов дублируются, при выводе использую оператор **distinct** для отбора уникальных значений. Выводимому полю присваиваю название «Аэропорты».

№	Вопрос	Использовать	Запрос

3	Вывести 10 рейсов с максимальным временем задержки вылета	Оператор LIMIT	`select flight_id , (actual_departure - scheduled_departure) as max_delay`
			`from flights f`
			`where (actual_departure-scheduled_departure) is not null`
			`order by 2 desc`
			`limit 10`

Из таблицы рейсов flights вывожу Идентификатор рейса flight_id и вычисляемое поле – разницу между фактическим и плановым временем вылета (actual_departure -scheduled_departure).
Условием **is not null** на вычисляемое поле исключаю рейсы, которые не состоялись или только планируемые.
Сортирую по убыванию разницы (**order by**) и устанавливаю ограничение на выводимое количество записей (**limit 10**)



№	Вопрос	Использовать	Запрос

4	Были ли брони, по которым не были получены посадочные талоны?	Верный тип JOIN	`select count(*) "Количество броней без посадочных"`
			`from bookings b`
			`join tickets t using(book_ref)`
			`left join boarding_passes bp using(ticket_no)`
			`where bp.boarding_no is null`

Таблицу с бронированиями (bookings) джойню с таблицей билетов (tickets) по номеру бронирования для того, чтобы далее присоединить слева таблицу с посадочными талонами (boarding_passes).
В случае наличия броней без посадочных талонов номер посадочного талона будет **null**.
Задавая условие **where** bp.boarding_no **is null**, оставляю в выборке только брони без посадочных талонов.
В **select** агрегирую оставшиеся записи с помощью count.
Если счетчик будет нулевым, то количество броней и количество посадочных талонов совпадают.

№	Вопрос	Использовать	Запрос

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.	- Оконная функция
		- Подзапросы или/и cte
			`with cte as (`
			`select f.flight_id ,`
			`f.aircraft_code ,`
			`f.departure_airport ,`
			`f.actual_departure ,`
			`count(boarding_no) as actual_fill`
			`from flights f`
			`join boarding_passes bp using(flight_id)`
			`where status = 'Arrived' or status = 'Departed'`
			`group by flight_id`
			`)`
			`select flight_id,`
			`cte.departure_airport, cte.actual_departure::date,`
			`actual_fill,`
			`sum(actual_fill) over (partition by cte.departure_airport,`
			`cte.actual_departure::date order by cte.actual_departure) as`
			`cumulative,`
			`capacity,`
			`(capacity-actual_fill) as "vacant, pc",`
			`round((capacity-actual_fill)*100.0/capacity, 2) /*|| '%'*/ as`
			`"vacant,%"`
			`from (select s.aircraft_code , count (s.seat_no) as capacity`
			`from seats s`
			`group by s.aircraft_code`
			`) uq`
			`join cte using (aircraft_code)`

Использую CTE, в котором из таблицы flights, соединенной (**join**) с таблицей boarding_passes, выбираются рейс (flight_id), код самолета (aircraft_code), аэропорт отправления (departure_airport), фактические дата и время вылета (actual_departure), а также и количество (агрегация **count**) выданных посадочных талонов, т.е. фактическое заполнение борта.

С помощью **where** выбираются рейсы со статусом Arrived или Departed, так как в остальных случаях фактическая информация о загрузке на момент вылета может быть неверной. Группировка по flight_id.

В основном запросе использую подзапрос в операторе **from** для подсчета вместимости судна в разрезе модели самолета.

Джойню CTE к результату подзапроса по коду самолета (aircraft_code).

В **select** указываю flight_id, departure_airport, actual_departure в типе данных **date**, фактическое заполнение борта.

С помощью оконной функции определяю суммарное накопление количества вывезенных пассажиров из каждого аэропорта на каждый день.

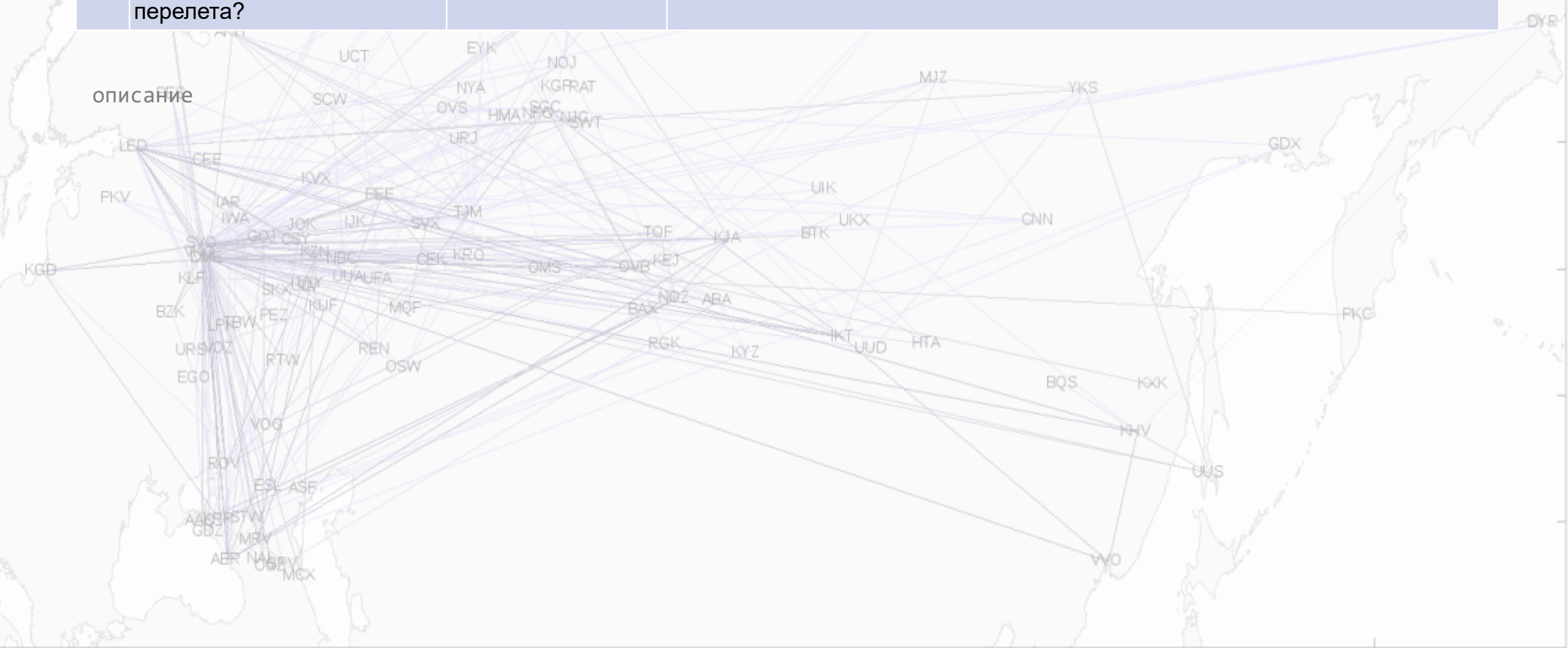
Далее: вместимость, свободные места на каждом рейсе и процентное отношение к общему количеству мест в самолете.

№	Вопрос	Использовать	Запрос
6	Найдите процентное соотношение перелетов по типам самолетов от общего количества.	- Подзапрос или окно - Оператор ROUND	select a.model, count(f.flight_id) as "by_type", (select count(flight_id) from flights f where status = 'Arrived') as total, round(count(f.flight_id)*100.0/(select count(flight_id) from flights f where status = 'Arrived'),2) as "by_type/total" from aircrafts a join flights f using(aircraft_code) group by a.model , f.aircraft_code

В основном запросе таблицу aircrafts джойню с таблицей рейсов flights по полю кода самолета. Результат группирую по модели и коду самолета.
В операторе **select** для вывода результата с помощью подзапроса агрегирую количество рейсов с условием завершенности: status = 'Arrived'.
Для подсчета процентного соотношения применяю оператор **round** - округление до двух десятичных знаков.

№	Вопрос	Использовать	Запрос
7	Были ли города, в которые можно добраться бизнес - классом дешевле, чем эконом-классом в рамках перелета?	- CTE	

описание



№	Вопрос	Использовать	Запрос

8	Между какими городами нет прямых рейсов?	- Декартово произведение в предложении FROM.
		- Самостоятельно созданные представления.
		Оператор EXCEPT
			`create view routes_t as`
			`select distinct`
			`a.city as departure,`
			`a2.city as arrival`
			`from flights f`
			`join airports a on a.airport_code = f.departure_airport`
			`join airports a2 on a2.airport_code = f.arrival_airport`
			
			`select a1.city as point_A,`
			`a2.city as point_B`
			`from airports a1`
			`cross join airports a2`
			`where a1.city <> a2.city`
			`except`
			`select *`
			`from routes_t`

Над таблицей рейсов (flights) создаю представление с городами, между которыми существуют прямые рейсы:
- Так как в flights нет названий городов, в которых находятся аэропорты; приджойнил их.
- Оператором distinct исключаю дубликаты.

Далее, кросс-джойн таблицы airports, получаю всевозможные сочетания городов.
Исключаю из выборки записи где города совпадают.
На вывод в select только два поля с городами, так как оператор except работает с одинаковыми таблицами (по полям).
Из полученной выборки исключаю результат созданного ранее представления routes_t, т.е. города, между которыми существуют рейсы.

№	Вопрос	Использовать	Запрос

9	Вычислите расстояние между аэропортами, связанными прямыми рейсами, сравните с допустимой максимальной дальностью перелетов в самолетах, обслуживающих эти рейсы *
	
	описание
		- Оператор RADIANS или использование sind/cosd
		- CASE
			`select distinct`
			`a1.airport_name as departure,`
			`a2.airport_name as arrival,`
			`a.range as max_distance,`
			`round((acos(sind(a1.coordinates[0]) * sind(a2.coordinates[0]) +`
			`cosd(a1.coordinates[0]) * cosd(a2.coordinates[0]) *`
			`cosd(a1.coordinates[1]-a2.coordinates[1])) * 6371)::decimal, 2)`
			`as distance,`
			`case when`
			`(a.range - round((acos(sind(a1.coordinates[0]) *`
			`sind(a2.coordinates[0]) + cosd(a1.coordinates[0]) *`
			`cosd(a2.coordinates[0]) * cosd(a1.coordinates[1]-`
			`a2.coordinates[1])) * 6371)::decimal, 2))<0`
			`then 'Nope'`
			`else 'Yep'`
			`end "Check"`
			`from flights f`
			`join airports a1 on f.departure_airport = a1.airport_code`
			`join airports a2 on f.arrival_airport = a2.airport_code`
			`join aircrafts a on a.aircraft_code = f.aircraft_code`

* - В облачной базе координаты находятся в столбце airports_data.coordinates - работаете, как с массивом. В локальной базе координаты находятся в столбцах airports.longitude и airports.latitude.
Кратчайшее расстояние между двумя точками A и B на земной поверхности (если принять ее за сферу) определяется зависимостью:
d = arccos {sin(latitude_a)·sin(latitude_b) + cos(latitude_a)·cos(latitude_b)·cos(longitude_a - longitude_b)},
где latitude_a и latitude_b — широты, longitude_a, longitude_b — долготы данных пунктов,
d — расстояние между пунктами измеряется в радианах длиной дуги большого круга земного шара.
Расстояние между пунктами, измеряемое в километрах, определяется по формуле:
L = d·R, где R = 6371 км — средний радиус земного шара.

К таблице рейсов джойню таблицу с аэропортами для связи названий и кодов аэропортов прилета и вылета.
Так как данное задание выполняю на другом компьютере и нет доступа к локальной БД, то координаты для расчета расстояния между аэропортами вытягиваю из массивов таблицы airports.
В оператор **case** как условие ввожу разность между дальностью полета модели самолета на рейсе (**range**) и рассчитанным расстоянием между аэропортами.
В зависимости от результата (больше или меньше нуля) вывожу отметку о проверке.