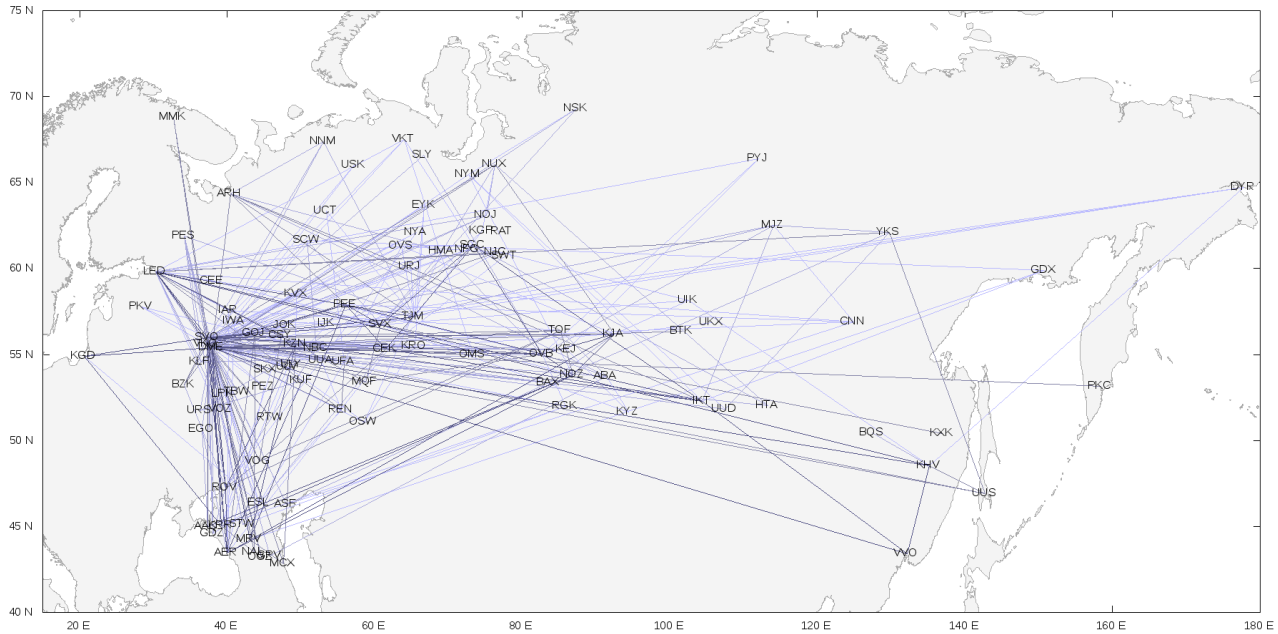
Авиаперевозки

Описание демонстрационной базы данных



Представляем вам демонстрационную базу данных для СУБД PostgreSQL. В этом документе описана схема данных, состоящая из восьми таблиц и нескольких представлений. В качестве предметной области выбраны авиаперевозки по России. Базу данных можно скачать с нашего сайта, см. раздел «Установка».

База данных может использоваться, например,

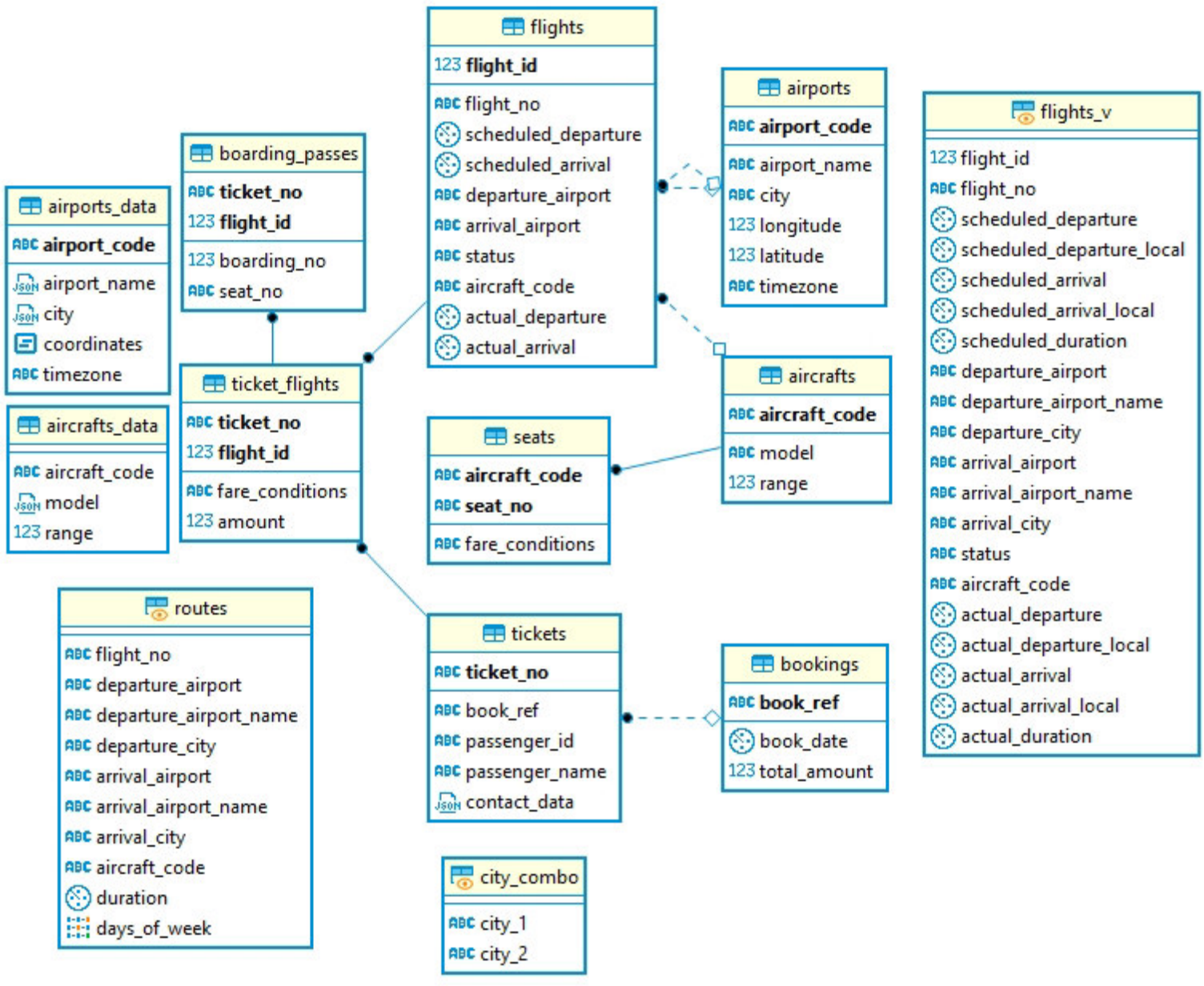
- для самостоятельного изучения языка запросов SQL,
- для подготовки книг, пособий и учебных курсов по языку SQL,
- для демонстрации возможностей PostgreSQL в статьях и заметках.

При разработке демонстрационной базы данных мы преследовали несколько целей:

- схема данных должна быть достаточно простой, чтобы быть понятной без особых пояснений,
- в то же время схема данных должна быть достаточно сложной, чтобы позволять строить осмысленные запросы,
- база данных должна быть наполнена данными, напоминающими реальные, с которыми будет интересно работать.

Демонстрационная база данных распространяется под лицензией PostgreSQL.

Свои замечания и пожелания направляйте нам по адресу edu@postgrespro.ru.


flights
flight_id
flight_no
scheduled_departure
scheduled_arrival
departure_airport
arrival_airport
status
aircraft_code
actual_departure
actual_arrival
airports
airport_code
airport_name
city
longitude
latitude
timezone
flights_v
flight_id
flight_no
scheduled_departure
scheduled_departure_local
scheduled_arrival
scheduled_arrival_local
scheduled_duration
departure_airport
departure_airport_name
departure_city
arrival_airport
arrival_airport_name
arrival_city
status
aircraft_code
actual_departure
actual_departure_local
actual_arrival
actual_arrival_local
actual_duration
boarding_passes
ticket_no
flight_id
boarding_no
seat_no
airports_data
airport_code
airport_name
city
coordinates
timezone
ticket_flights
ticket_no
flight_id
fare_conditions
amount
aircrafts_data
aircraft_code
model
range
aircrafts
aircraft_code
model
range
seats
aircraft_code
seat_no
fare_conditions
routes
flight_no
departure_airport
departure_airport_name
departure_city
arrival_airport
arrival_airport_name
arrival_city
aircraft_code
duration
days_of_week
tickets
ticket_no
book_ref
passenger_id
passenger_name
contact_data
bookings
book_ref
book_date
total_amount
city_combo
city_1
city_2

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. 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).

Объекты схемы

Список отношений

```
       Имя       |      Тип      |  Small | Medium |   Big  |     Описание
-----------------+---------------+--------+--------+--------+-------------------
 aircrafts       | таблица       |  16 kB |  16 kB |  16 kB | Самолеты
 airports        | таблица       |  48 kB |  48 kB |  48 kB | Аэропорты
 boarding_passes | таблица       |  31 MB | 102 MB | 427 MB | Посадочные талоны
 bookings        | таблица       |  13 MB |  30 MB | 105 MB | Бронирования
 flights         | таблица       |   3 MB |   6 MB |  19 MB | Рейсы
 flights_v       | представление |   0 kb |   0 kB |   0 kB | Рейсы
 routes          | мат. предст.  | 136 kB | 136 kB | 136 kB | Маршруты
 seats           | таблица       |  88 kB |  88 kB |  88 kB | Места
 ticket_flights  | таблица       |  64 MB | 145 MB | 516 MB | Перелеты
 tickets         | таблица       |  47 MB | 107 MB | 381 MB | Билеты
```

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).

```
    Столбец    |   Тип   | Модификаторы |             Описание
---------------+---------+--------------+---------------------------------
 aircraft_code | char(3) | NOT NULL     | Код самолета, IATA
 model         | text    | NOT NULL     | Модель самолета
 range         | integer | NOT NULL     | Максимальная дальность полета, км
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 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).

```
   Столбец    |   Тип   | Модификаторы |           Описание
--------------+---------+--------------+------------------------------
 airport_code | char(3) | NOT NULL     | Код аэропорта
 airport_name | text    | NOT NULL     | Название аэропорта
 city         | text    | NOT NULL     | Город
 longitude    | float   | NOT NULL     | Координаты аэропорта: долгота
 latitude     | float   | NOT NULL     | Координаты аэропорта: широта
 timezone     | text    | NOT NULL     | Временная зона аэропорта
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 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).

```
   Столбец   |    Тип     | Модификаторы |         Описание
-------------+------------+--------------+-------------------------
 ticket_no   | char(13)   | NOT NULL     | Номер билета
 flight_id   | integer    | NOT NULL     | Идентификатор рейса
 boarding_no | integer    | NOT NULL     | Номер посадочного талона
 seat_no     | varchar(4) | NOT NULL     | Номер места
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.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.

```
   Столбец    |      Тип      | Модификаторы |         Описание
--------------+---------------+--------------+--------------------------
 book_ref     | char(6)       | NOT NULL     | Номер бронирования
 book_date    | timestamptz   | NOT NULL     | Дата бронирования
 total_amount | numeric(10,2) | NOT NULL     | Полная сумма бронирования
Индексы:
    PRIMARY KEY, btree (book_ref)
Ссылки извне:
    TABLE "tickets" FOREIGN KEY (book_ref) REFERENCES bookings(book_ref)
```

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). 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.

```
       Столбец       |     Тип     | Модификаторы |          Описание
---------------------+-------------+--------------+----------------------------
 flight_id           | serial      | NOT NULL     | Идентификатор рейса
 flight_no           | char(6)     | NOT NULL     | Номер рейса
 scheduled_departure | timestamptz | NOT NULL     | Время вылета по расписанию
 scheduled_arrival   | timestamptz | NOT NULL     | Время прилёта по расписанию
 departure_airport   | char(3)     | NOT NULL     | Аэропорт отправления
 arrival_airport     | char(3)     | NOT NULL     | Аэропорт прибытия
 status              | varchar(20) | NOT NULL     | Статус рейса
 aircraft_code       | char(3)     | NOT NULL     | Код самолета, IATA
 actual_departure    | timestamptz |              | Фактическое время вылета
 actual_arrival      | timestamptz |              | Фактическое время прилёта
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 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.

```
     Столбец     |     Тип     | Модификаторы |      Описание
-----------------+-------------+--------------+--------------------
 aircraft_code   | char(3)     | NOT NULL     | Код самолета, IATA
 seat_no         | varchar(4)  | NOT NULL     | Номер места
 fare_conditions | varchar(10) | NOT NULL     | Класс обслуживания
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 bookings.ticket_flights

Перелет соединяет билет с рейсом и идентифицируется их номерами.

Для каждого перелета указываются его стоимость (amount) и класс обслуживания (fare_conditions).

```
     Столбец     |      Тип      | Модификаторы |      Описание
-----------------+---------------+--------------+---------------------
 ticket_no       | char(13)      | NOT NULL     | Номер билета
 flight_id       | integer       | NOT NULL     | Идентификатор рейса
 fare_conditions | varchar(10)   | NOT NULL     | Класс обслуживания
 amount          | numeric(10,2) | NOT NULL     | Стоимость перелета
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 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).

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.

```
    Столбец     |     Тип     | Модификаторы |          Описание
----------------+-------------+--------------+----------------------------
 ticket_no      | char(13)    | NOT NULL     | Номер билета
 book_ref       | char(6)     | NOT NULL     | Номер бронирования
 passenger_id   | varchar(20) | NOT NULL     | Идентификатор пассажира
 passenger_name | text        | NOT NULL     | Имя пассажира
 contact_data   | jsonb       |              | Контактные данные пассажира
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)
```

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).

Столбец	Тип	Описание
flight_id	integer	Идентификатор рейса
flight_no	char(6)	Номер рейса
scheduled_departure	timestamptz	Время вылета по расписанию
scheduled_departure_local	timestamp	Время вылета по расписанию, местное время в пункте отправления
scheduled_arrival	timestamptz	Время прилёта по расписанию
scheduled_arrival_local	timestamp	Время прилёта по расписанию, местное время в пункте прибытия
scheduled_duration	interval	Планируемая продолжительность полета
departure_airport	char(3)	Код аэропорта отправления
departure_airport_name	text	Название аэропорта отправления
departure_city	text	Город отправления
arrival_airport	char(3)	Код аэропорта прибытия
arrival_airport_name	text	Название аэропорта прибытия
arrival_city	text	Город прибытия
status	varchar(20)	Статус рейса
aircraft_code	char(3)	Код самолета, IATA
actual_departure	timestamptz	Фактическое время вылета
actual_departure_local	timestamp	Фактическое время вылета, местное время в пункте отправления
actual_arrival	timestamptz	Фактическое время прилёта
actual_arrival_local	timestamp	Фактическое время прилёта, местное время в пункте прибытия
actual_duration	interval	Фактическая продолжительность полета

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.

Именно такая информация и составляет материализованное представление routes.

```
        Столбец         |    Тип    |              Описание
------------------------+-----------+-------------------------------------
 flight_no              | char(6)   | Номер рейса
 departure_airport      | char(3)   | Код аэропорта отправления
 departure_airport_name | text      | Название аэропорта отправления
 departure_city         | text      | Город отправления
 arrival_airport        | char(3)   | Код аэропорта прибытия
 arrival_airport_name   | text      | Название аэропорта прибытия
 arrival_city           | text      | Город прибытия
 aircraft_code          | char(3)   | Код самолета, IATA
 duration               | interval  | Продолжительность полета
 days_of_week           | integer[] | Дни недели, когда выполняются рейсы
```

Функция now

Демонстрационная база содержит временной «срез» данных — так, как будто в некоторый момент была сделана резервная копия реальной системы. Например, если некоторый рейс имеет статус Departed, это означает, что в момент резервного копирования самолет вылетел и находился в воздухе.

Позиция «среза» сохранена в функции bookings.now(). Ей можно пользоваться в запросах там, где в обычной жизни использовалась бы функция now().

Кроме того, значение этой функции определяет версию демонстрационной базы данных. Актуальная версия на текущий момент — от 13.10.2016.

Использование

Схема bookings

Все объекты демонстрационной базы данных находятся в схеме bookings. Это означает, что при обращении к объектам вам необходимо либо явно указывать имя схемы (например: `bookings.flights`), либо предварительно изменить конфигурационный параметр search_path (например: `SET search_path = bookings, public;`).

Однако для функции bookings.now в любом случае необходимо явно указывать схему, чтобы отличать ее от стандартной функции now.

Поставленные вопросы и решения в виде sql запросов:

1. В каких городах больше одного аэропорта?

```
select city
from airports
group by city
having count(1)>1;
```

Для отбора городов была использована конструкция HAVING, которая при расчете сгруппированного по городу значения выражения count(1) (подсчет количества строк, для таблицы airport это количество аэропортов) сравнивает его “>1”. В результате получаем строки (города), где больше одного аэропорта.

2. В каких аэропортах есть рейсы, которые обслуживаются самолетами с максимальной дальностью перелетов?

```
select f.departure_airport
from flights f
join (select aircraft_code
        from aircrafts
        order by "range" desc
        limit 1) t on t.aircraft_code = f.aircraft_code
group by f.departure_airport;
```

Для того, чтобы отобрать нужные нам аэропорты, для начала нужно отобрать самолеты с максимальной дальностью перелета, что и происходит в подзапросе в выражении FROM основного запроса. Для этого были отсортирована таблица aircraft по признаку “range”, которая выражает дальность перелета судна. После этого с помощью выражения limit 1 возвращается самая верхняя строка, с максимальным значением.

А в основном запросе отбираются аэропорты с помощью таблицы flights, которая в свою очередь отфильтрована подзапросом с помощью команды join(или inner join), которая возвращает только те строки, которые присутствуют в обеих таблицах.

Далее код аэропорта группируется, во избежание дублей.

3. Узнать максимальное время задержки вылетов самолетов

```
select max(actual_departure - scheduled_departure) delay
from flights f;
```

Вычисляем задержку рейса вычитая из фактического плановое время вылета.
Это выражение помещаем в функцию max и получаем максимальное значение

4. Были ли брони, по которым не совершались перелеты?

```
select b.book_ref
from bookings b
left join (select b.book_ref
        from bookings b
        join tickets t on t.book_ref = b.book_ref
        join ticket_flights tf on tf.ticket_no = t.ticket_no
        join flights f on f.flight_id = tf.flight_id
        where f.status = 'Arrived'
        group by b.book_ref) b1 on b.book_ref = b1.book_ref
where b1.book_ref is null;
```

Чтобы получить брони, по которым не совершались ни одного перелета, для начала нужно получить брони, по которым совершился хоть один перелет. Это происходит в подзапросе, где последовательно произведено соединение (join) от bookings к tickets, далее к ticket_flights и наконец к flights по их ключам. Далее были отобраны перелеты, которые совершились, это отражается в поле status, значением “Arrived”(прибыл в аэропорт прибытия).

Вторым этапом были отобраны брони, которые не соединились командой join со списком броней, по которым были совершены перелеты. Для этого таблицу bookings соединили с таблицей в подзапросе с помощью left join, тем самым мы сохранили строки в таблице «слева», которые не нашли совпадения с таблицей справа.

И в предикате WHERE отбирая по значения null присоединенной таблицы получает нужный нам список броней, по которым не совершались перелеты.

5. Самолеты каких моделей совершают наибольший % перелетов?

```
select a.model,round(cnt_flights/SUM(cnt_flights) over() *100,2) as share_flights
from (select aircraft_code,count(1) cnt_flights
        from flights f
        group by aircraft_code) c
join aircrafts a on a.aircraft_code = c.aircraft_code
order by cnt_flights desc
limit 1;
```

В подзапросе подсчитываем кол-во перелетов по каждой модели самолета. В основном запросе подсчитываем число всех перелетов, как сумма перелетов по всем моделям и сравниваем с кол-вом перелетов каждого самолета. Отсортированные по убыванию кол-ва перелетов и усеченный по 1 строке получаем модель с наибольшей долей перелетов

6. Между какими городами нет прямых рейсов*?

```
CREATE OR REPLACE view city_combo as
with cities as
        (select city, row_number() OVER() city_id
        from airports
        group by city
        order by city)
select c1.city city_1,c2.city city_2
from cities c1,cities c2
where c1.city_id < c2.city_id;

--Теперь ищем пары городов без прямого сообщения
select c.*
from city_combo c
left join (select departure_city,arrival_city
        from routes r
        group by departure_city,arrival_city) r on r.departure_city = c.city_1 and r.arrival_city = c.city_2
where r.arrival_city is null;
```

Для выявления пар городов, у которых нет прямых рейсов между собой было создано представление, которое формирует все возможные пары городов, для этого: 1. Получаем уникальные названия городов и присваиваем каждому уникальный номер с помощью оконной функции row_number(). 2. Делаем кросс-джоин 2 экземпляров полученных таблиц с предикатом, что код города из первой таблицы меньше кода города из второй таблицы, таким образом получаем уникальные пары городов.

К представлению описанному выше присоединяем left-join'ом все уникальные комбинации город вылета – город прилета, после отбора null – строк получаем список городов, у которых нет прямого сообщения.