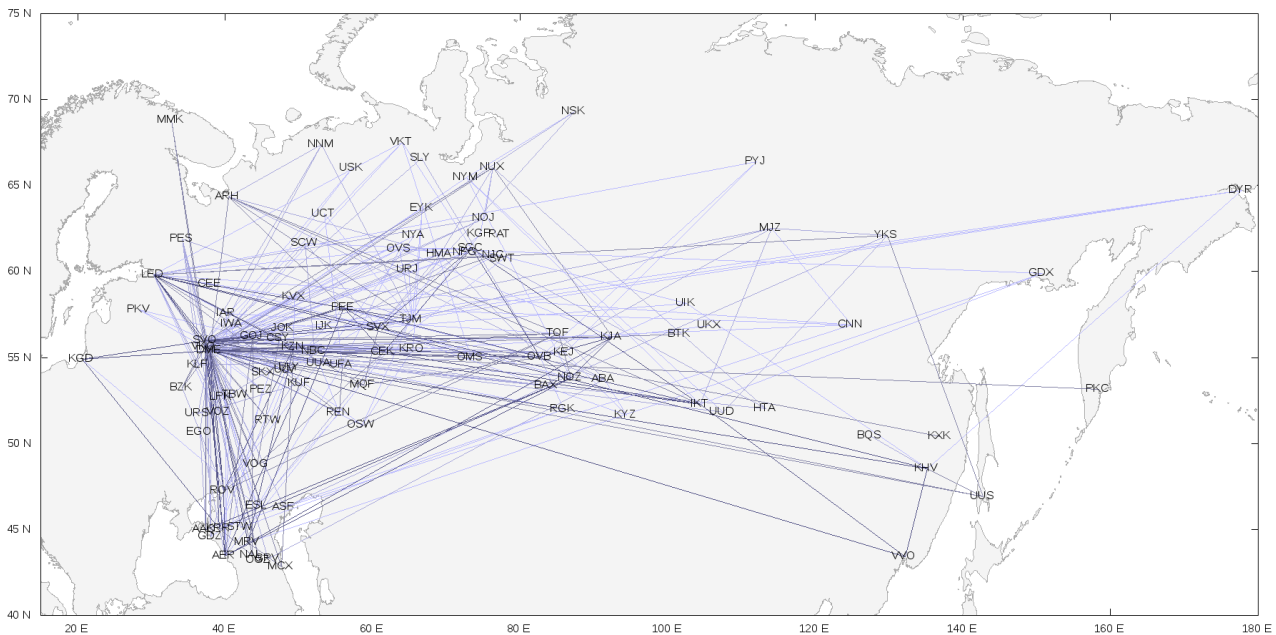


Авиаперевозки

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.

Установка

Установка с сайта

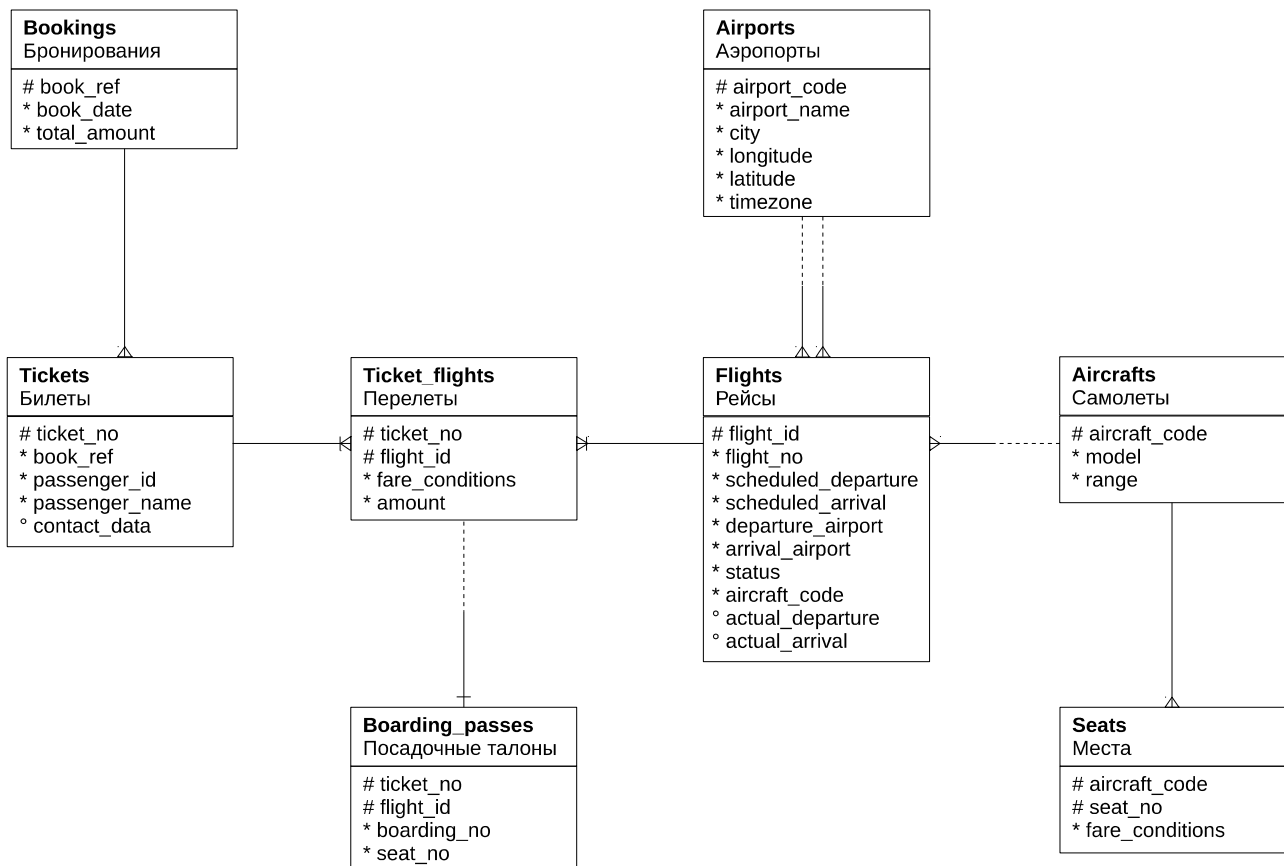
Демонстрационная база данных доступна на edu.postgrespro.ru в трех версиях, которые отличаются только объемом данных:

- [demo_small.zip](#) (21 МБ) — данные по полетам за один месяц (размер БД 265 МБ),
- [demo_medium.zip](#) (62 МБ) — данные по полетам за три месяца (размер БД 666 МБ),
- [demo_big.zip](#) (232 МБ) — данные по полетам за год (размер БД 2502 МБ).

Небольшая база годится для того, чтобы писать запросы, и при этом не займет много места на диске. База большого размера позволит почувствовать, как ведут себя запросы на больших объемах данных, и задуматься об оптимизации.

Файлы содержат SQL-скрипт, создающий базу данных «demo» и наполняющий ее данными (фактически, это резервная копия, созданная утилитой `pg_dump`). Обратите внимание, что при установке существующая база данных «demo» будет удалена и создана заново! Владелец базы данных «demo» станет пользователь СУБД, выполнявший скрипт.

Диаграмма схемы данных



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 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	timestampz	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). Реальное 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	timestampz	NOT NULL	Время вылета по расписанию
scheduled_arrival	timestampz	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	timestampz		Фактическое время вылета
actual_arrival	timestampz		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_data).

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	timestampz	Время вылета по расписанию
scheduled_departure_local	timestamp	Время вылета по расписанию, местное время в пункте отправления
scheduled_arrival	timestampz	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	timestampz	Фактическое время вылета
actual_departure_local	timestamp	Фактическое время вылета, местное время в пункте отправления
actual_arrival	timestampz	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.

Примеры запросов

Чтобы лучше познакомиться с содержимым демонстрационной базы данных, посмотрим на результаты нескольких простых запросов.

Результаты, представленные ниже, были получены для версии с небольшой базой данных (demo_small) от 13 октября 2016. Если в вашей системе запросы выдают другие данные, проверьте версию демонстрационной базы (функция bookings.now). Незначительные отклонения могут быть связаны с местным временем, отличным от московского, и настройками локализации.

Все рейсы выполняются несколькими типами самолетов:

```
SELECT * FROM aircrafts;
```

aircraft_code	model	range
773	Boeing 777-300	11100
763	Boeing 767-300	7900
SU9	Sukhoi SuperJet-100	3000
320	Airbus A320-200	5700
321	Airbus A321-200	5600
319	Airbus A319-100	6700
733	Boeing 737-300	4200
CN1	Cessna 208 Caravan	1200
CR2	Bombardier CRJ-200	2700

(9 строк)

Для каждого типа самолета поддерживается список мест в салоне. Например, вот где можно разместиться в небольшом самолете Cessna 208 Caravan:

```
SELECT  a.aircraft_code,
        a.model,
        s.seat_no,
        s.fare_conditions
FROM    aircrafts a
        JOIN seats s ON a.aircraft_code = s.aircraft_code
WHERE   a.model = 'Cessna 208 Caravan'
ORDER BY s.seat_no;
```

aircraft_code	model	seat_no	fare_conditions
CN1	Cessna 208 Caravan	1A	Economy
CN1	Cessna 208 Caravan	1B	Economy
CN1	Cessna 208 Caravan	2A	Economy
CN1	Cessna 208 Caravan	2B	Economy
CN1	Cessna 208 Caravan	3A	Economy
CN1	Cessna 208 Caravan	3B	Economy
CN1	Cessna 208 Caravan	4A	Economy
CN1	Cessna 208 Caravan	4B	Economy
CN1	Cessna 208 Caravan	5A	Economy
CN1	Cessna 208 Caravan	5B	Economy
CN1	Cessna 208 Caravan	6A	Economy
CN1	Cessna 208 Caravan	6B	Economy

(12 строк)

Самолеты большего размера имеют больше посадочных мест с разными классами обслуживания:

```

SELECT  s2.aircraft_code,
        string_agg (s2.fare_conditions || '(' || s2.num::text || ')',
                    ', ' ) as fare_conditions
FROM    (
        SELECT  s.aircraft_code, s.fare_conditions, count(*) as num
        FROM    seats s
        GROUP BY s.aircraft_code, s.fare_conditions
        ORDER BY s.aircraft_code, s.fare_conditions
        ) s2
GROUP BY s2.aircraft_code
ORDER BY s2.aircraft_code;

```

aircraft_code	fare_conditions
319	Business(20), Economy(96)
320	Business(20), Economy(120)
321	Business(28), Economy(142)
733	Business(12), Economy(118)
763	Business(30), Economy(192)
773	Business(30), Comfort(48), Economy(324)
CN1	Economy(12)
CR2	Economy(50)
SU9	Business(12), Economy(85)

(9 строк)

База данных содержит список аэропортов практически всех крупных городов России. В большинстве городов есть только один аэропорт. Исключение составляют:

```

SELECT  a.airport_code as code,
        a.airport_name,
        a.city,
        a.longitude,
        a.latitude,
        a.timezone
FROM    airports a
WHERE   a.city IN (
        SELECT  aa.city
        FROM    airports aa
        GROUP BY aa.city
        HAVING  COUNT(*) > 1
        )
ORDER BY a.city, a.airport_code;

```

code	airport_name	city	longitude	latitude	timezone
DME	Домодедово	Москва	37.906111	55.408611	Europe/Moscow
SVO	Шереметьево	Москва	37.414589	55.972642	Europe/Moscow
VKO	Внуково	Москва	37.261486	55.591531	Europe/Moscow
ULV	Баратаевка	Ульяновск	48.2267	54.268299	Europe/Samara
ULY	Ульяновск-Восточный	Ульяновск	48.8027	54.401	Europe/Samara

(5 строк)

Чтобы понять, откуда и куда можно улететь, удобно использовать материализованное представление routes, в котором агрегируется информация о всех рейсах. Вот, например, куда, в какие дни недели и за какое время можно долететь из Волгограда:

```

SELECT  r.arrival_city as city,
        r.arrival_airport as airport_code,
        r.arrival_airport_name as airport_name,
        r.days_of_week,
        r.duration
FROM    routes r
WHERE   r.departure_city = 'Волгоград';

```

city	airport_code	airport_name	days_of_week	duration
Москва	SVO	Шереметьево	{1, 2, 3, 4, 5, 6, 7}	01:15:00
Челябинск	CEK	Челябинск	{1, 2, 3, 4, 5, 6, 7}	01:50:00
Ростов-на-Дону	ROV	Ростов-на-Дону	{1, 2, 3, 4, 5, 6, 7}	00:30:00
Москва	VKO	Внуково	{1, 2, 3, 4, 5, 6, 7}	01:10:00
Чебоксары	CSY	Чебоксары	{1, 2, 3, 4, 5, 6, 7}	02:45:00
Томск	TOF	Богашёво	{3}	03:50:00

(6 строк)

База данных была сформирована на момент времени, возвращаемый функцией `bookings.now()`:

```
SELECT bookings.now() as now;
           now
```

```
-----
2016-10-13 17:00:00+03
```

Относительно именно этой временной точки все рейсы делятся на прошедшие и будущие:

```
SELECT  status,
         count(*) as count,
         min(scheduled_departure) as min_scheduled_departure,
         max(scheduled_departure) as max_scheduled_departure
FROM    flights
GROUP BY status
ORDER BY min_scheduled_departure;
```

status	count	min_scheduled_departure	max_scheduled_departure
Arrived	16707	2016-09-13 00:50:00+03	2016-10-13 16:25:00+03
Cancelled	414	2016-09-16 10:35:00+03	2016-11-12 19:55:00+03
Departed	58	2016-10-13 08:55:00+03	2016-10-13 16:50:00+03
Delayed	41	2016-10-13 14:15:00+03	2016-10-14 16:25:00+03
On Time	518	2016-10-13 16:55:00+03	2016-10-14 17:00:00+03
Scheduled	15383	2016-10-14 17:05:00+03	2016-11-12 19:40:00+03

(6 строк)

Найдем ближайший рейс, вылетающий из Екатеринбурга в Москву. Использовать для такого запроса таблицу `flight` не очень удобно, так как в ней нет информации о городах отправления и прибытия. Поэтому воспользуемся представлением `flights_v`:

```
\x
SELECT  f.*
FROM    flights_v f
WHERE   f.departure_city = 'Екатеринбург'
AND     f.arrival_city = 'Москва'
AND     f.scheduled_departure > bookings.now()
ORDER BY f.scheduled_departure
LIMIT  1;
```

```
-[ RECORD 1 ]-----+-----
flight_id      | 10927
flight_no     | PG0226
scheduled_departure | 2016-10-14 07:10:00+03
scheduled_departure_local | 2016-10-14 09:10:00
scheduled_arrival | 2016-10-14 08:55:00+03
scheduled_arrival_local | 2016-10-14 08:55:00
scheduled_duration | 01:45:00
departure_airport | SVX
departure_airport_name | Кольцово
departure_city | Екатеринбург
arrival_airport | SVO
arrival_airport_name | Шереметьево
```

```

arrival_city      | Москва
status           | On Time
aircraft_code    | 773
actual_departure |
actual_departure_local |
actual_arrival   |
actual_arrival_local |
actual_duration  |

```

Обратите внимание, что в представлении `flights_v` указано не только московское время, но и местное время в аэропортах вылета и прилета.

Бронирования

Каждое бронирование может включать несколько билетов, по одному на каждого пассажира. Билет, в свою очередь, может включать несколько перелетов. Полная информация о бронировании находится в трех таблицах: `bookings`, `tickets` и `ticket_flights`.

Найдем несколько бронирований с самой высокой стоимостью:

```

SELECT *
FROM bookings
ORDER BY total_amount desc
LIMIT 10;

```

book_ref	book_date	total_amount
3B54BB	2016-09-02 16:08:00+03	1204500.00
3AC131	2016-09-28 00:06:00+03	1087100.00
65A6EA	2016-08-31 05:28:00+03	1065600.00
D7E9AA	2016-10-06 04:29:00+03	1062800.00
EF479E	2016-09-30 14:58:00+03	1035100.00
521C53	2016-09-05 08:25:00+03	985500.00
514CA6	2016-09-24 04:07:00+03	955000.00
D70BD9	2016-09-02 11:47:00+03	947500.00
EC7EDA	2016-08-30 15:13:00+03	946800.00
8E4370	2016-09-25 01:04:00+03	945700.00

(10 строк)

Посмотрим, из каких билетов состоит бронирование с кодом 521C53:

```

SELECT ticket_no,
       passenger_id,
       passenger_name
FROM tickets
WHERE book_ref = '521C53';

```

ticket_no	passenger_id	passenger_name
0005432661914	8234 547529	IVAN IVANOV
0005432661915	2034 201228	ANTONINA KUZNECOVA

(2 строки)

Если нас интересует, какие перелеты включены в билет Антонины Кузнецовой, то это можно узнать запросом:

```

SELECT  to_char(f.scheduled_departure, 'DD.MM.YYYY') as when,
        f.departure_city || '(' || f.departure_airport || ')' as departure,
        f.arrival_city || '(' || f.arrival_airport || ')' as arrival,
        tf.fare_conditions as class,
        tf.amount
FROM    ticket_flights tf
        JOIN flights_v f ON tf.flight_id = f.flight_id
WHERE   tf.ticket_no = '0005432661915'
ORDER BY f.scheduled_departure;

```

when	departure	arrival	class	amount
26.09.2016	Москва(SVO)	Анадырь(DYR)	Business	185300.00
30.09.2016	Анадырь(DYR)	Хабаровск(KHV)	Business	92200.00
01.10.2016	Хабаровск(KHV)	Благовещенск(BQS)	Business	18000.00
06.10.2016	Благовещенск(BQS)	Хабаровск(KHV)	Business	18000.00
10.10.2016	Хабаровск(KHV)	Анадырь(DYR)	Economy	30700.00
15.10.2016	Анадырь(DYR)	Москва(SVO)	Business	185300.00

(6 строк)

Как видим, высокая стоимость бронирования объясняется большим количеством перелетов на дальние расстояния бизнес-классом.

Часть перелетов в этом билете имеет более ранние даты, чем значение `bookings.now()`: это значит, что они уже выполнены. А последний полет еще предстоит. После регистрации на рейс выписывается посадочный талон с указанием места в самолете. Мы можем посмотреть какие именно места занимала Антонина (обратите внимание на внешнее левое соединение с таблицей `boarding_passes`):

```

SELECT  to_char(f.scheduled_departure, 'DD.MM.YYYY') as when,
        f.departure_city || '(' || f.departure_airport || ')' as departure,
        f.arrival_city || '(' || f.arrival_airport || ')' as arrival,
        f.status,
        bp.seat_no
FROM    ticket_flights tf
        JOIN flights_v f ON tf.flight_id = f.flight_id
        LEFT JOIN boarding_passes bp ON tf.flight_id = bp.flight_id
        AND tf.ticket_no = bp.ticket_no
WHERE   tf.ticket_no = '0005432661915'
ORDER BY f.scheduled_departure;

```

when	departure	arrival	status	seat_no
26.09.2016	Москва(SVO)	Анадырь(DYR)	Arrived	5C
30.09.2016	Анадырь(DYR)	Хабаровск(KHV)	Arrived	1D
01.10.2016	Хабаровск(KHV)	Благовещенск(BQS)	Arrived	2C
06.10.2016	Благовещенск(BQS)	Хабаровск(KHV)	Arrived	2D
10.10.2016	Хабаровск(KHV)	Анадырь(DYR)	Arrived	20B
15.10.2016	Анадырь(DYR)	Москва(SVO)	Scheduled	

(6 строк)

Новое бронирование

Попробуем отправить Александра Николаевича Радищева по маршруту, который принес ему известность. Разумеется бесплатно и бизнес-классом. Предварительно найдем «завтрашний» рейс, а также позаботимся о возвращении через неделю.

```

BEGIN;

INSERT INTO bookings (book_ref, book_date, total_amount)
VALUES      ('_QWE12', bookings.now(), 0);

INSERT INTO tickets (ticket_no, book_ref, passenger_id, passenger_name)
VALUES      ('_000000000001', '_QWE12', '1749 051790', 'ALEKSANDR RADISHCHEV');

INSERT INTO ticket_flights (ticket_no, flight_id, fare_conditions, amount)
VALUES      ('_000000000001', 9720, 'Business', 0),
            ('_000000000001', 6662, 'Business', 0);

COMMIT;

```

Мы начинаем идентификаторы с подчеркивания, чтобы не пересекаться с диапазоном значений, присутствующих в базе.

Сразу регистрируемся на завтрашний рейс:

```

INSERT INTO boarding_passes (ticket_no, flight_id, boarding_no, seat_no)
VALUES      ('_000000000001', 9720, 1, '1A');

```

Проверим информацию о созданном бронировании:

```

SELECT      b.book_ref,
            t.ticket_no,
            t.passenger_id,
            t.passenger_name,
            tf.fare_conditions,
            tf.amount,
            f.scheduled_departure_local,
            f.scheduled_arrival_local,
            f.departure_city || '(' || f.departure_airport || ')' as departure,
            f.arrival_city || '(' || f.arrival_airport || ')' as arrival,
            f.status,
            bp.seat_no
FROM        bookings b
            JOIN tickets t ON b.book_ref = t.book_ref
            JOIN ticket_flights tf ON tf.ticket_no = t.ticket_no
            JOIN flights_v f ON tf.flight_id = f.flight_id
            LEFT JOIN boarding_passes bp ON tf.flight_id = bp.flight_id
            AND tf.ticket_no = bp.ticket_no
WHERE      b.book_ref = '_QWE12'
ORDER BY   t.ticket_no, f.scheduled_departure;

```

```

-[ RECORD 1 ]-----+-----
book_ref      | _QWE12
ticket_no     | _000000000001
passenger_id  | 1749 051790
passenger_name | ALEKSANDR RADISHCHEV
fare_conditions | Business
amount        | 0.00
scheduled_departure_local | 2016-10-14 08:45:00
scheduled_arrival_local   | 2016-10-14 09:35:00
departure      | Санкт-Петербург(LED)
arrival        | Москва(SVO)
status         | On Time
seat_no       | 1A

```



```
-[ RECORD 2 ]-----+-----
book_ref           | _QWE12
ticket_no          | _0000000000001
passenger_id       | 1749 051790
passenger_name     | ALEKSANDR RADISHCHEV
fare_conditions    | Business
amount             | 0.00
scheduled_departure_local | 2016-10-21 09:20:00
scheduled_arrival_local  | 2016-10-21 10:10:00
departure          | Москва(SVO)
arrival            | Санкт-Петербург(LED)
status             | Scheduled
seat_no           |
```

Надеемся, что эти несколько простых примеров помогли составить представление о содержимом демонстрационной базы данных.