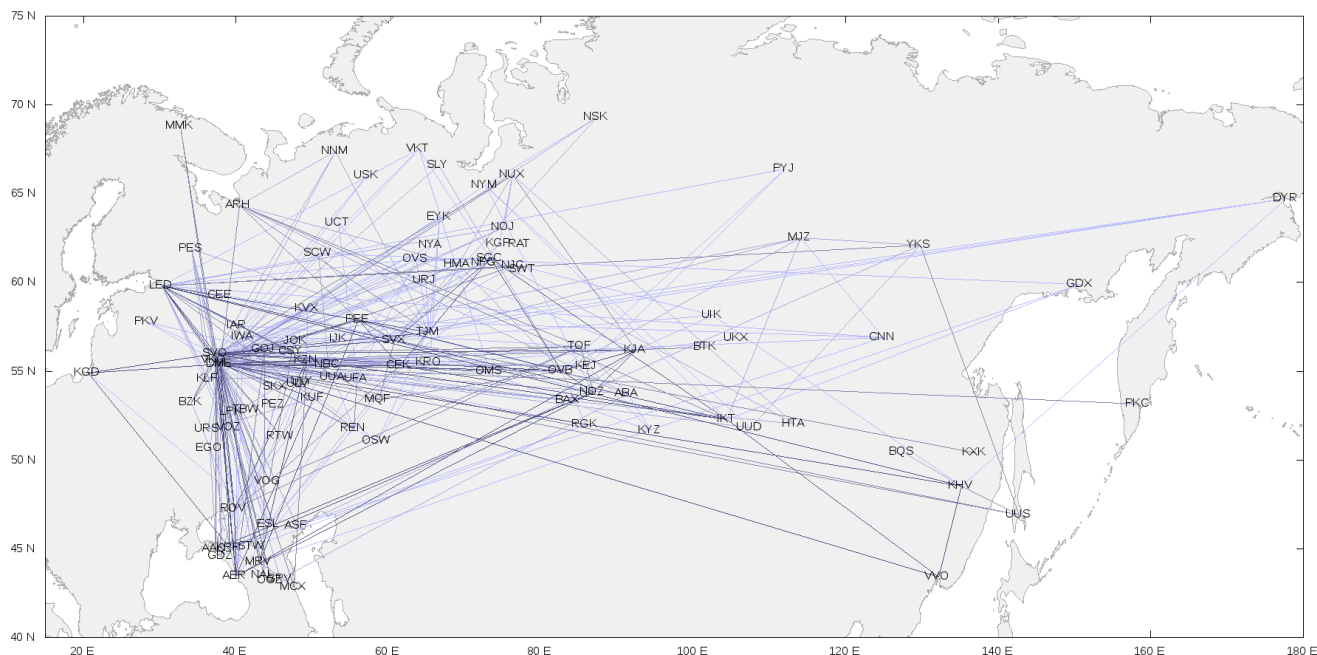


# Приложение L. Демонстрационная база данных «Авиаперевозки»

Представляем вам демонстрационную базу данных для PostgreSQL. В этом приложении к документации описана схема данных, состоящая из восьми таблиц и нескольких представлений. В качестве предметной области выбраны авиаперевозки по России. Базу данных можно загрузить с нашего сайта, см. Раздел L.1.

**Рисунок L.1. Воздушное сообщение в России**



База данных может использоваться, например:

- для самостоятельного изучения языка запросов SQL;
- для подготовки книг, пособий и учебных курсов по языку SQL;
- для демонстрации возможностей Postgres Pro 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](mailto:edu@postgrespro.ru).

## L.1. Установка

Демонстрационная база данных доступна на [edu.postgrespro.ru](http://edu.postgrespro.ru) в трёх версиях, которые отличаются только объёмом данных:

- *demo\_small.zip* (21 МБ) — данные по полётам за один месяц (размер БД 265 МБ);

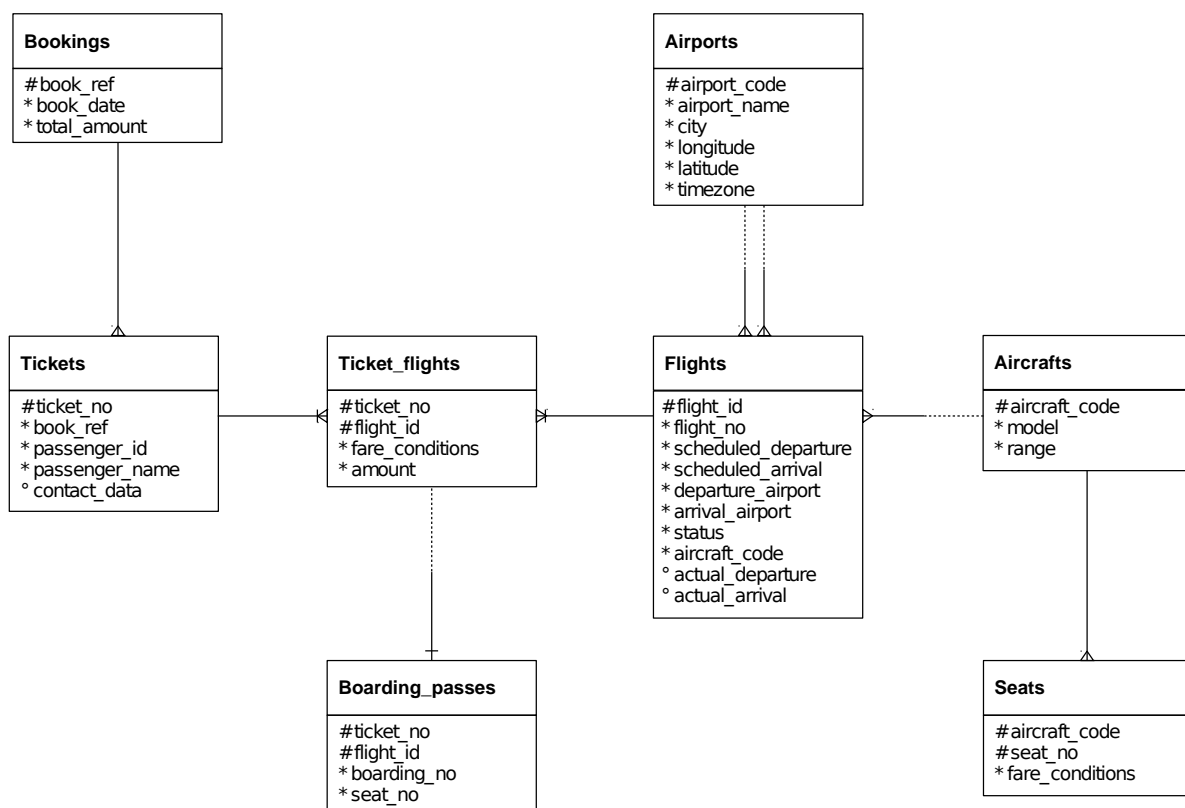
- *demo\_medium.zip* (62 МБ) — данные по полётам за три месяца (размер БД 666 МБ);
- *demo\_big.zip* (232 МБ) — данные по полётам за год (размер БД 2502 МБ).

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

Файлы содержат SQL-скрипт, создающий базу данных *demo* и наполняющий её 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.

## L.2. Диаграмма схемы данных

Рисунок L.2. Диаграмма схемы Bookings



## L.3. 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 жёсткого 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 выдаётся 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).

## L.4. Объекты схемы

### L.4.1. Список отношений

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

### L.4.2. Таблица `bookings.aircrafts`

Каждая модель воздушного судна идентифицируется своим трёхзначным кодом (`aircraft_code`). Указывается также название модели (`model`) и максимальная дальность полета в километрах (`range`).

Столбец	Тип	Модификаторы	Описание
<code>aircraft_code</code>	<code>char(3)</code>	NOT NULL	Код самолета, IATA
<code>model</code>	<code>text</code>	NOT NULL	Модель самолета
<code>range</code>	<code>integer</code>	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`

### L.4.3. Таблица `bookings.airports`

Аэропорт идентифицируется трехбуквенным кодом (`airport_code`) и имеет своё 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 (latitude), долгота (longitude) 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	Часовой пояс 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)

#### L.4.4. Таблица bookings.boarding\_passes

При регистрации на рейс, которая возможна за сутки до плановой даты отправления, пассажиру выдаётся посадочный талон. Он идентифицируется также, как и перелёт — 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)

#### L.4.5. 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 включённых 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)

## L.4.6. 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 отменён.

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)
```

#### L.4.7. Таблица `bookings.seats`

Места определяют схему салона каждой модели. Каждое место определяется своим номером (`seat_no`) и имеет закреплённый 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
```

#### L.4.8. Таблица `bookings.ticket_flights`

Перелёт 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)
```

### L.4.9. 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
<code>ticket_no</code>	<code>char(13)</code>	<code>NOT NULL</code>	Номер билета
<code>book_ref</code>	<code>char(6)</code>	<code>NOT NULL</code>	Номер бронирования
<code>passenger_id</code>	<code>varchar(20)</code>	<code>NOT NULL</code>	Идентификатор пассажира
<code>passenger_name</code>	<code>text</code>	<code>NOT NULL</code>	Имя пассажира
<code>contact_data</code>	<code>jsonb</code>		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)`

### L.4.10. Представление `bookings.flights_v`

Над таблицей `flights_v` создано представление `flights`, 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
<code>flight_id</code>	<code>integer</code>	Идентификатор рейса
<code>flight_no</code>	<code>char(6)</code>	Номер рейса
<code>scheduled_departure</code>	<code>timestamp</code>	Время вылета по расписанию
<code>scheduled_departure_local</code>	<code>timestamp</code>	Время вылета по расписанию, местное время в пункте отправления
<code>scheduled_arrival</code>	<code>timestamp</code>	Время прилёта по расписанию
<code>scheduled_arrival_local</code>	<code>timestamp</code>	Время прилёта по расписанию, местное время в пункте прибытия
<code>scheduled_duration</code>	<code>interval</code>	Планируемая продолжительность полета
<code>departure_airport</code>	<code>char(3)</code>	Код аэропорта отправления
<code>departure_airport_name</code>	<code>text</code>	Название аэропорта отправления
<code>departure_city</code>	<code>text</code>	Город отправления
<code>arrival_airport</code>	<code>char(3)</code>	Код аэропорта прибытия
<code>arrival_airport_name</code>	<code>text</code>	Название аэропорта прибытия
<code>arrival_city</code>	<code>text</code>	Город прибытия
<code>status</code>	<code>varchar(20)</code>	Статус рейса
<code>aircraft_code</code>	<code>char(3)</code>	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

#### L.4.11. Материализованное представление bookings.routes

Таблица рейсов (bookings.flights) содержит избыточность: из неё 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 полёта
days_of_week	integer[]	Дни, когда выполняются рейсы

#### L.4.12. 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() function. 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.

### L.5. Использование

#### L.5.1. 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 её от стандартной функции now.