Введение в SQL. Установка и знакомство с ПО

Занятие 1.1

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик

Алексей Кузьмин

Директор разработки,
Data Scientist ДомКлик

В конце занятия вы

1. Будете знать,
зачем нужны БД

2. Познакомитесь
с инструментарием курса

3. Потренируетесь запускать контейнеры
и просматривать атрибуты БД

4. Напишете свой первый
запрос в SQL

1. Зачем нужны БД
2. Функции СУБД
3. Наш инструментарий
4. Проведём установку и первичную настройку ПО
5. Устройство реляционной СУБД

Введение в SQL

1

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик

Зачем нужны БД

Основная функция базы данных — предоставление единого хранилища для всей информации, относящейся к определённой теме.

Вместо того чтобы выискивать нужные сведения в документах Word, таблицах Excel, текстовых файлах, сообщениях электронной почты и самоклеющихся заметках, их можно взять из единой базы.

База данных может содержать всё что угодно, будь то список приглашённых на свадьбу гостей или информация о каждом клиенте, посетившем web-сайт электронного магазина и разместившего там свои заказы.

Основные понятия

Базы данных (БД) — это структурная совокупность взаимосвязанных данных определённой предметной области: реальных объектов, процессов, явлений и т. д.

Пример. БД о наличии медикаментов, БД документов учеников школы, картотека отдела кадров.

Основные понятия

Появление компьютерной техники повысило эффективность работы с базами данных. Доступ к данным и управление ими происходит в среде специального программного пакета — системы управления базами данных (СУБД).

СУБД — это программа, с помощью которой осуществляется хранение, обработка и поиск информации в базе данных.

Функции СУБД

СУБД используются для выполнения различных операций с данными:

- ввод
- хранение
- манипулирование
- обработка запросов к БД
- поиск
- выборка
- сортировка
- обновление
- защита данных от несанкционированного доступа или потери

Инструментарий курса

2

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик

С чем мы работаем?

PostgreSQL Database Server — СУБД

DBeaver – клиент для подключения к СУБД

PostgreSQL Database Server

- Скачать дистрибутив версии 12.x
  по ссылке согласно вашей ОС
  https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

- Если у вас операционная система 32bit,
  то скачиваете версию 10.x

- Производите установку согласно инструкции
  https://docs.google.com/document/d/1p2CufXVD0xB_hUcsen1nLNCjdNc2DpP3KawVmqoIFvA/

DBeaver

- Скачать и установить отсюда: https://dbeaver.io
- Необходима Community Edition версия, отличие от платной Enterprise версии в отсутствии поддержки NoSQL баз данных
- Среда для подключения и работы с базами данных

Время практики

Настройка подключения
к локальному серверу
PostgreSQL

3

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик

PostgreSQL Database Server

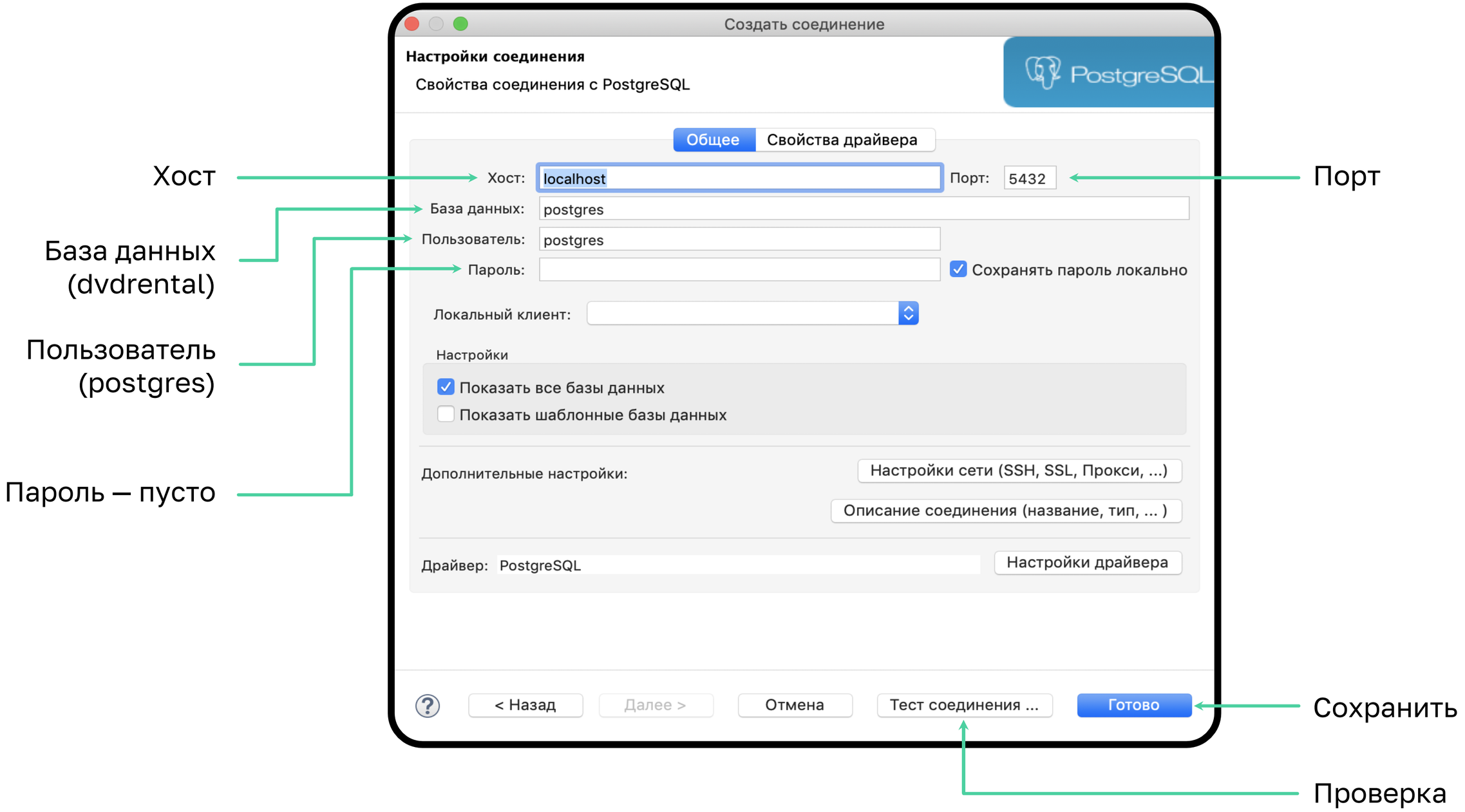
1. После установки локального сервера он автоматически запускается в фоновых процессах. Если этого не произошло — необходимо перезагрузить компьютер
2. Если при установке данные для подключения оставляли по умолчанию, то они будут:

   host: localhost

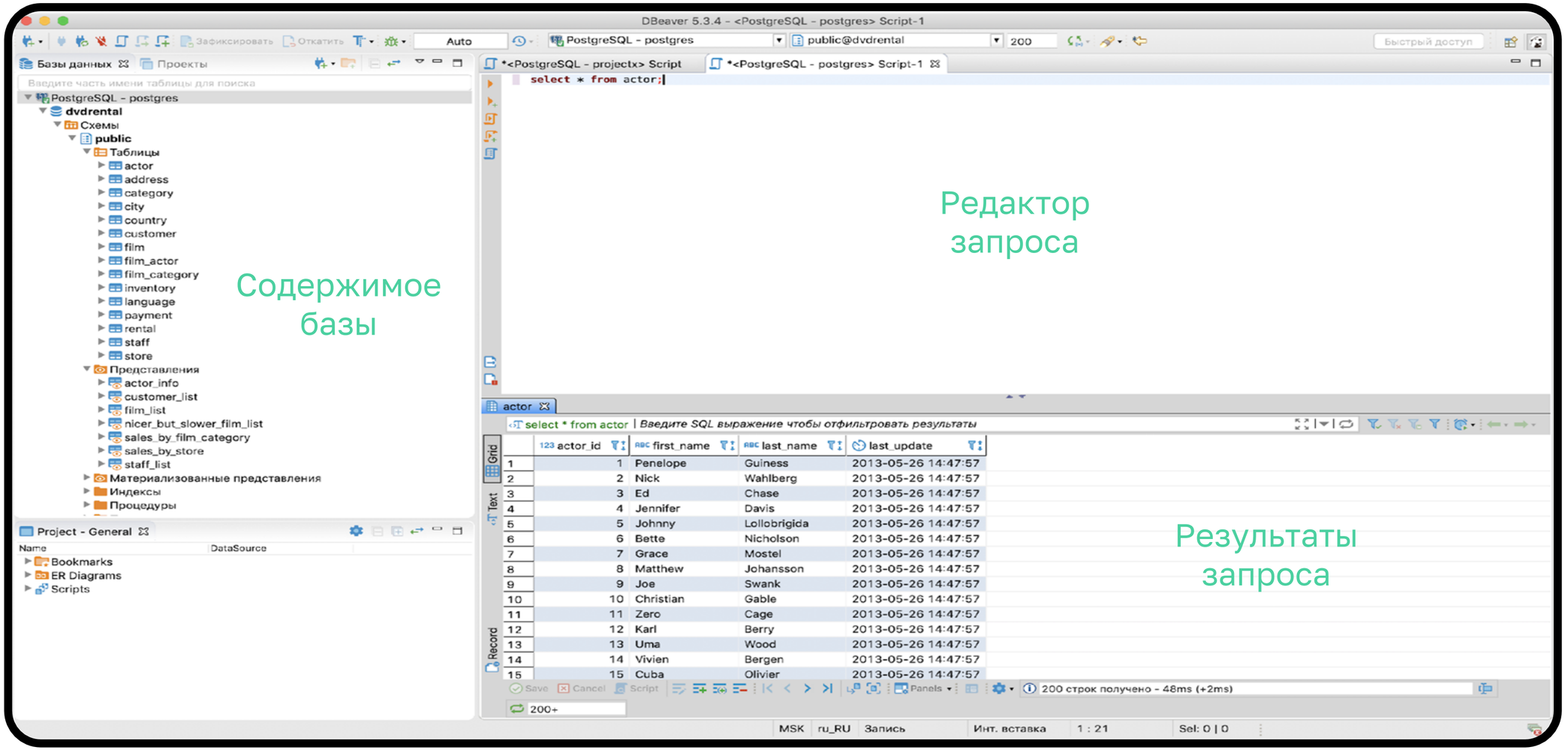
   port: 5432

   user: postgres

   password: тот, который указали при установке PostgreSQL


Создать соединение
Настройки соединения
Свойства соединения с PostgreSQL
PostgreSQL
Общее
Свойства драйвера
Хост:
localhost
Порт:
5432
База данных:
postgres
Пользователь:
postgres
Пароль:
Сохранять пароль локально
Локальный клиент:
Настройки
Показать все базы данных
Показать шаблонные базы данных
Дополнительные настройки:
Настройки сети (SSH, SSL, Прокси, ...)
Описание соединения (название, тип, ...)
Драйвер:
PostgreSQL
Настройки драйвера
< Назад
Далее >
Отмена
Тест соединения ...
Готово
Хост
База данных (dvdrental)
Пользователь (postgres)
Пароль — пусто
Порт
Сохранить
Проверка

SELECT * FROM actor;



Реляционная модель

4

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик

Реляционная модель

Модель представляет собой фиксированную структуру математических понятий, которая описывает, как будут представлены данные.

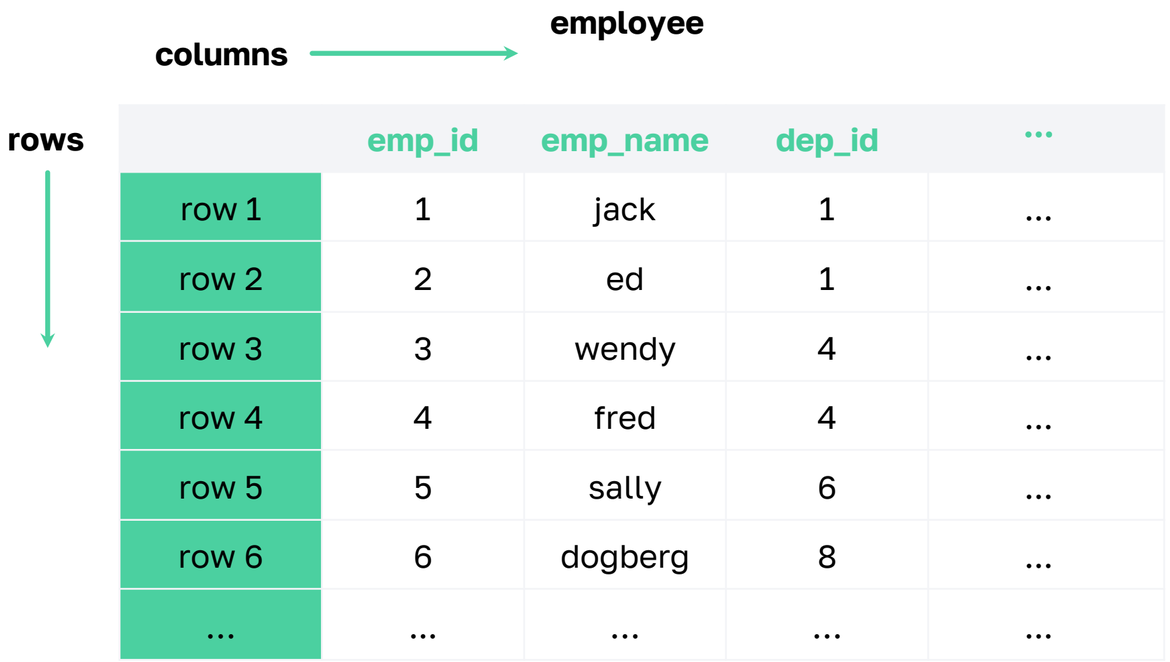
Базовой единицей данных в пределах реляционной модели является таблица.

Таблица

Таблица — это базовая единица данных. В реляционной алгебре она называется «отношение» (relation).

Состоит из столбцов (columns), которые определяют конкретные типы данных.

Данные в таблице организованы в строки (rows), которые содержат множества значений столбцов.



Первичные ключи

При создании таблицы могут быть использованы различные «ограничения» (constraints), которые содержат правила, указывающие, какие данные представлены в ней.

Одним из самых используемых ограничений является первичный ключ (primary key constraint), который гарантирует, что каждая строка таблицы содержит уникальный идентификатор.

Первичный ключ может состоять из одного или нескольких столбцов. Первичные ключи, состоящие из нескольких столбцов, также называются «составными» (composite)

Первичные ключи

Правильным считается наличие первичного ключа во всех таблицах базы данных. При этом существует два варианта первичных ключей:

1. естественный (натуральный, natural primary key)
2. искусственный (суррогатный, surrogate primary key)

Натуральные первичные ключи

Представляют собой данные, которые уже присутствуют в описываемой предметной области. Например, почтовые индексы могут быть использованы как естественные первичные ключи без дополнительной обработки.

Их использование, если оно возможно, считается более правильным, чем искусственных.

Суррогатные первичные ключи

Представляют собой целочисленный идентификатор.

Применяется там, где нет возможности использовать
натуральный первичный ключ. Позволяют решать
те же практические задачи, что и естественные:
улучшение производительности памяти
и индексов при операциях обновления.

Внешние ключи

В то время как одна таблица имеет первичный ключ, другая таблица может иметь ограничение, описывающее, что её строки ссылаются на гарантированно существующие строки в первой таблице.

Это реализуется через создание во «внешней» таблице («потомке») столбца (может быть и нескольких), значениями которого являются значения первичного ключа из «локальной» таблицы («родителя»).

Вместе наборы этих столбцов составляют внешний ключ (foreign key constraint), который является механизмом базы данных, гарантирующим, что значения в «удалённых» столбцах присутствуют как первичные ключи в «локальных».

Это ограничение контролирует все операции на этих таблицах:

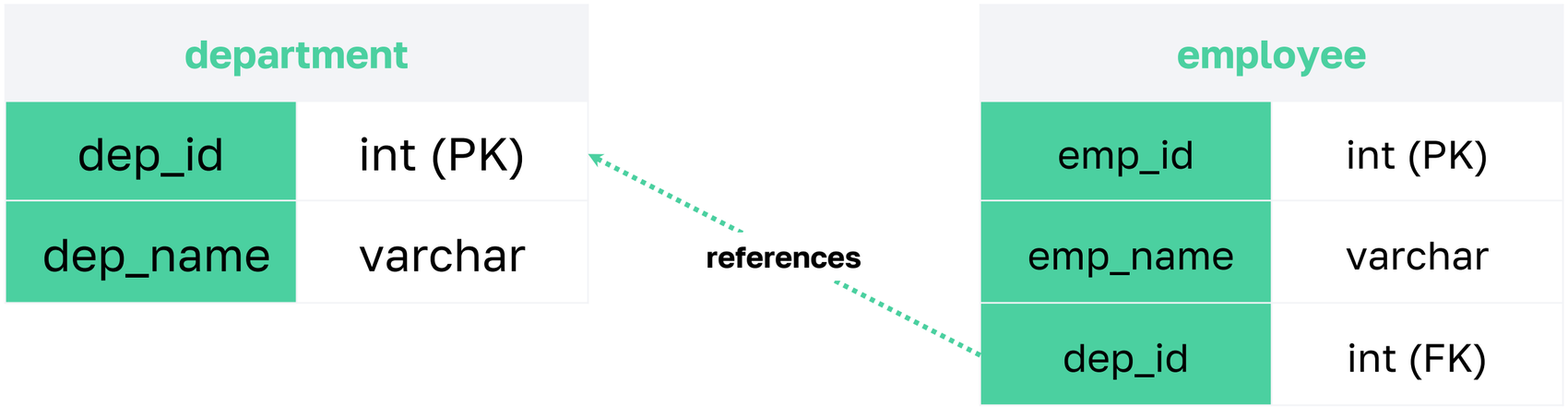
- добавление / изменение данных во «внешней» таблице
- удаление / изменение данных в «родительской» таблице

Внешний ключ проверяет, чтобы данные корректно присутствовали в обеих таблицах. Иначе операции будут отменены.

Внешние ключи

Внешние ключи могут быть составными, если входящие в них первичные ключи являются таковыми.

В примере представлена таблица department, которая связана с таблицей employee через отношение столбцов employee.dep_id и department.dep_id:



Нормализация

Реляционная модель базируется на реляционной алгебре, одним из ключевых понятий которой является нормализация.

Основная идея нормализации в исключении повторяющихся данных так, чтобы конкретная часть данных была представлена только в одном месте.

Этот подход позволяет упростить данные до максимально атомарного вида, с которым намного проще работать: искать, производить какие-либо операции.

Классический пример денормализованных данных

name	language	department
Dilbert	C++	Systems
Dilbert	Java	Systems
Wally	Python	Engineering
Wendy	Scala	Engineering
Wendy	Java	Engineering

1. Строки в этой таблице могут быть уникально идентифицированы по столбцам name и language, которые являются потенциальным ключом

2. По теории нормализации таблица из примера нарушает вторую нормальную форму. Потому как не основной атрибут department логически связан только со столбцом name

3. Правильная нормализация в данном случае выглядит следующим образом

name	language
Dilbert	C++
Dilbert	Java
Wally	Python
Wendy	Scala
Wendy	Java

name	department
Dilbert	Systems
Wally	Engineering
Wendy	Engineering

Теперь наглядно видно, как вторая форма улучшила структуру данных. Изначально пример содержал повторы связок полей name и department так часто, как часто встречался уникальный для данного имени «язык». Улучшенный же вариант сделал связки name/department и name/language независимыми друг от друга.

Ограничения данных, такие как первичные и внешние ключи, предназначены как раз для достижения состояния нормализации.

Для примера выше это будет выглядеть так:

- "Employee Department -> name" — первичный ключ
- "Employee Language -> name, language" — составной первичный ключ
- "Employee Language -> name", в свою очередь, — внешний ключ, на поле "Employee Department -> name"

Если таблицу удаётся сходу свернуть в отношения ключей, то это зачастую значит, что она не нормализована.

Что мы сегодня узнали

Что мы сегодня узнали

1. Базы данных — единое хранилище информации по определённой теме

2. Сущности в базе данных реализованы в виде таблиц со связями, которые помогают поддерживать её целостность

3. Синтаксис SQL очень похож на простые общеупотребимые слова английского языка: выбрать (select), из (from) и т. д.

Полезные материалы

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик

Полезные материалы

- Документация Postgres
  https://postgrespro.ru/docs/postgresql
- Документация DBeaver
  https://github.com/dbeaver/dbeaver/wiki

Спасибо за внимание!

Алексей Кузьмин
Директор разработки,
Data Scientist ДомКлик