Подключение к Базе Данных

Лекция 2
Реляционная модель

Николай Хащанов

Full–stack developer

О спикере:

- Разрабатываю и поддерживаю crm/erp системы
- Преподаю в Нетологии
- Окончил РГТЭУ по специальности Менеджмент
- Оптимизация и автоматизация бизнес-процессов

Я в Слаке:

@Николай Хащанов

Содержание

Подключение к облачной БД

0

Николай Хащанов

нетология

Подключение

Для подключения к облачной базе данных нужно следовать пунктам инструкции по ссылке https://github.com/netology-ds-team/sql-materials/tree/main/sqlfree

Реквизиты для подключения:

Хост	psql.ntlg.tech
Порт	19001
База данных	sqlfree
Пользователь	netology
Пароль	NetoSQL2019

Подключение

Еще раз напомним, что:

- версии программ могут отличаться от указанных в инструкции
- на курсе используется СУБД PostgreSQL
- при прохождении курса можете использовать любой IDE, DBeaver не обязателен
- если у вас корпоративная машина с жесткими требованиями по безопасности, то пробуйте использовать portable версию DBeaver или потребуется вмешательство IT специалистов вашей компании
- в схеме public находится исходная учебная база данных, доступная только для чтения
- вам нужно создать собственную схему и восстановить в нее дамп учебной базы данных

Реляционная модель

1

Николай Хащанов

нетология

Реляционная модель

Реляционная модель представляет собой фиксированную структуру математических понятий, которая описывает, как будут представлены данные;

Базовой единицей данных в пределах реляционной модели является таблица;

Таблица — это базовая единица данных. В реляционной алгебре она называется «отношение» (relation). Состоит из атрибутов (columns), которые определяют конкретные типы данных. Данные в таблице организованы в кортежи (rows), которые содержат множества значений столбцов.

Реляционная модель

Преимущества:

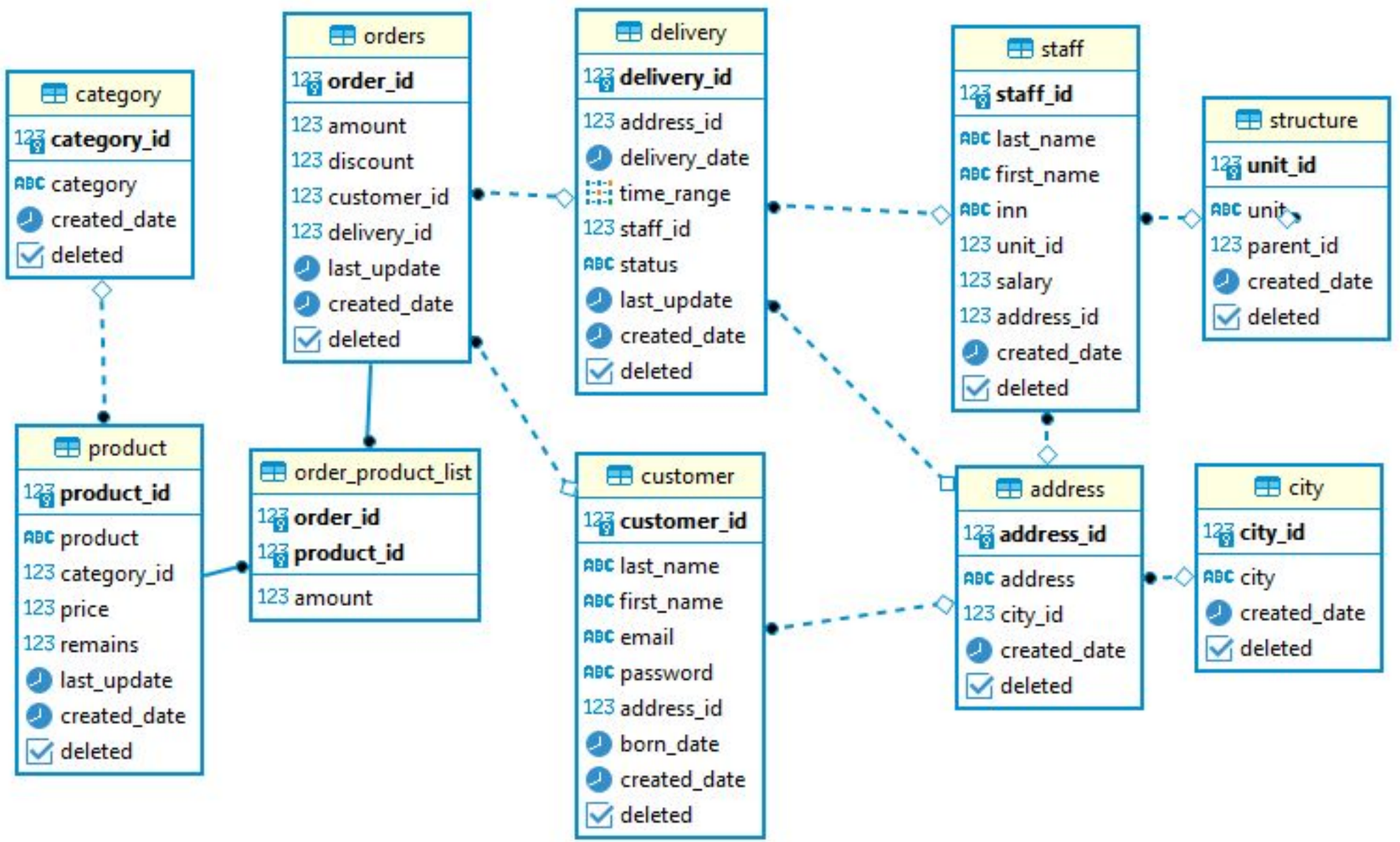
- Эффективное поддержание целостности данных;
- Блокировка и очередность доступа к данным;
- Атомарность данных (возможность использования сложных типов данных);
- Поддержка процедурных языков;
- Независимость физической и логической моделей.

Реляционная модель

Реляционная модель данных — созданная Эдгаром Коддом логическая модель данных, описывающая:

- структуры данных в виде наборов отношений;
- теоретико-множественные операции над данными: объединение (union), пересечение (intersect), разность(except) и декартово произведение (cross join);
- специальные реляционные операции: селекция (where), проекция (select att1, att2, att3...), соединение (join) и деление;
- специальные правила, обеспечивающие целостность данных.

Реляционная модель



Ограничения Первичных и Внешних ключей

2

Николай Хащанов

нетология

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), который гарантирует, что каждая строка таблицы содержит уникальный идентификатор.

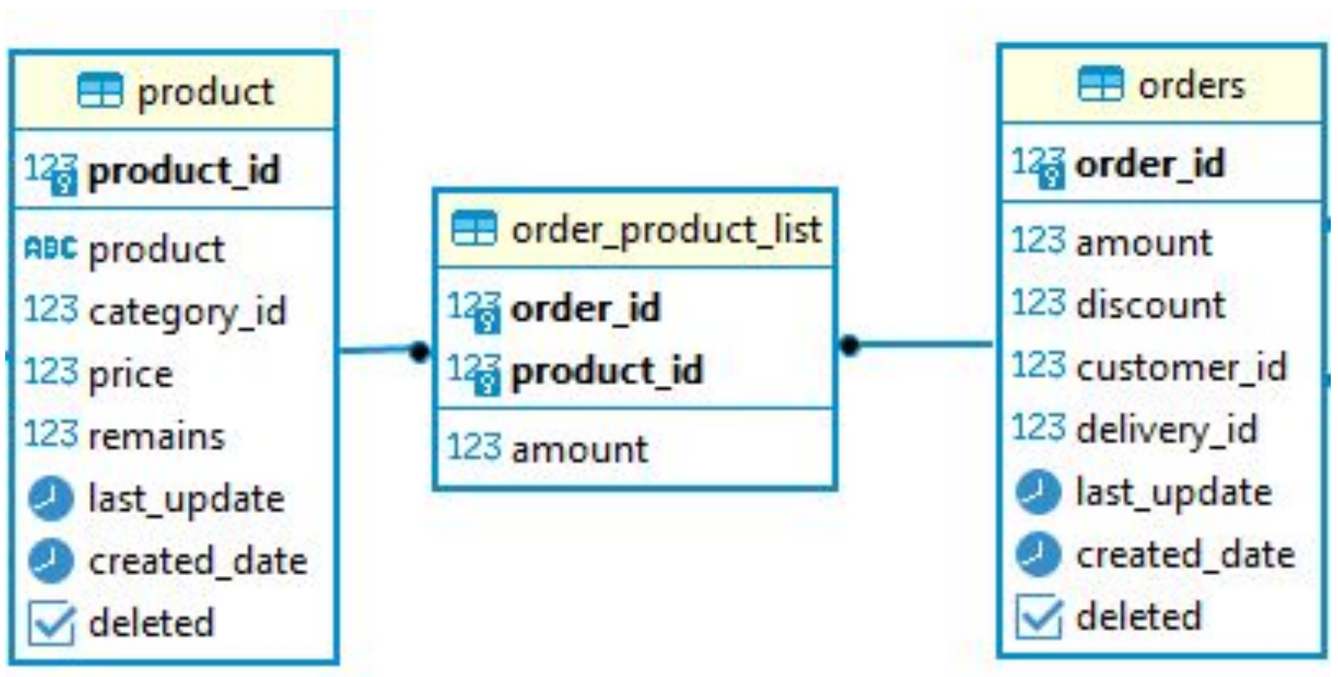
Правильным считается наличие первичного ключа во всех таблицах базы данных.

PRIMARY KEY = UNIQUE + NOT NULL + INDEX

Первичные ключи

Первичный ключ может состоять из одного или нескольких столбцов.

Первичные ключи, состоящие из нескольких столбцов называются «составными» (COMPOSITE).



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.

В справочнике стран натуральным первичным ключом может быть ISO код стран.

В справочнике граждан РФ натуральным составным первичным ключом может быть серия и номер паспорта.

Суррогатные первичные ключи

Как правило, 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 значения ссылаются на гарантированно существующие значения в первой таблице.

Это реализуется через создание в «дочерней» таблице столбца (может быть несколько столбцов), значениями которого являются значения первичного ключа (или иные уникальные значения) из «родительской» таблицы.

Внешние ключи

Вместе наборы этих столбцов составляют внешний ключ (FOREIGN KEY), который является механизмом базы данных, гарантирующим, что значения в «дочерних» столбцах присутствуют как первичные ключи (иил иные уникальные значения) в «родительских».

Это ограничение контролирует все операции на этих таблицах:

- добавление / изменение данных в «дочерней» таблице
- удаление / изменение данных в «родительской» таблице.

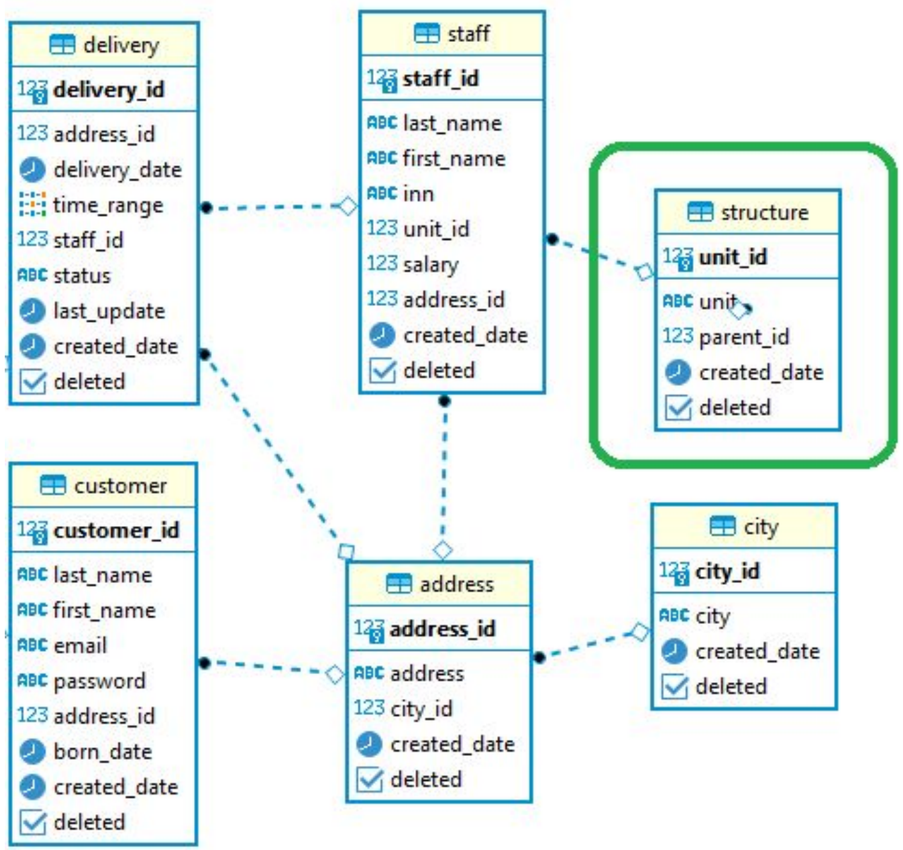
Внешний ключ проверяет, чтобы данные корректно присутствовали в обеих таблицах. Иначе операции будут отменены.

Внешние ключи

Внешние ключи могут быть составными.

Как правило «родителем» во внешнем ключе является первичный ключ, но при необходимости «родителем» может быть любой столбец, который имеет ограничение уникальности (UNIQUE).

В таблице structure внешний ключ назначен на столбец parent_id и ссылается на столбец unit_id собственной таблицы.



Типы данных

3

Николай Хащанов

нетология

Числовые типы

- integer: хранит числа от -2147483648 до +2147483647. Занимает 4 байта. Имеет псевдонимы int и int4
- bigint: хранит числа от -9223372036854775808 до +9223372036854775807. Занимает 8 байт. Имеет псевдоним int8
- real: хранит числа с плавающей точкой из диапазона от 1E-37 до 1E+37. Занимает 4 байта. Имеет псевдоним float4
- double precision: хранит числа с плавающей точкой из диапазона от 1E-307 до 1E+308. Занимает 8 байт. Имеет псевдоним float8
- numeric: хранит числа с плавающей точкой, имеет точность и масштаб, используется там, где требуется точность округления

Символьные типы

- character(n): представляет строку из фиксированного количества символов. С помощью параметра задаётся количество символов в строке. Имеет псевдоним char(n)
- character varying(n): представляет строку из нефиксированного количества символов. С помощью параметра задаётся максимальное количество символов в строке. Имеет псевдоним varchar(n)
- text: представляет текст произвольной длины

Типы для работы с датами и временем

- timestamp: хранит дату и время. Занимает 8 байт. Для дат самое нижнее значение - 4713 г. до н. э., самое верхнее значение - 294276 г. н. э.
- timestamp with time zone: то же самое, что и timestamp, только добавляет данные о часовом поясе
- date: представляет дату от 4713 г. до н. э. до 5874897 г. н. э. Занимает 4 байта
- time: хранит время с точностью до 1 микросекунды без указания часового пояса. Принимает значения от 00:00:00 до 24:00:00. Занимает 8 байт
- time with time zone: хранит время с точностью до 1 микросекунды с указанием часового пояса. Принимает значения от 00:00:00+1459 до 24:00:00-1459. Занимает 12 байт
- interval: представляет временной интервал. Занимает 16 байт

Логический тип

- Тип boolean может хранить одно из двух значений: true или false
- Вместо true можно указывать следующие значения:
  TRUE, 't', 'true', 'y', 'yes', 'on', '1'
- Вместо false можно указывать следующие значения:
  FALSE, 'f', 'false', 'n', 'no', 'off', '0'

Специальные типы данных

- json: хранит данные json в текстовом виде
- jsonb: хранит данные json в бинарном формате
- xml: хранит данные в формате XML
- массивы:
  text[] – массив, элементами которого являются строки
  int[] – массив, элементами которого являются числа

NULL

- NULL - это значение, которое означает, что значение отсутствует
- Для сравнения с NULL используется конструкция:
  WHERE column IS NULL
- Для сравнения, что значение не является NULL, используется конструкция:
  WHERE column IS NOT NULL

Логический порядок инструкции SELECT

4

Николай Хащанов

нетология

Последовательность при написании

1. SELECT DISTINCT
2. FROM
3. JOIN ON
4. WHERE
5. GROUP BY
6. WITH CUBE или WITH ROLLUP
7. HAVING
8. ORDER BY
9. LIMIT OFFSET

Последовательность при выполнении

1. FROM
2. ON
3. JOIN
4. WHERE
5. GROUP BY
6. WITH CUBE или WITH ROLLUP
7. HAVING
8. SELECT
9. DISTINCT
10. ORDER BY
11. OFFSET
12. LIMIT

Типовые операции с разными типами данных

5

Николай Хащанов

нетология

Типизация

При работе с разными типами данных и совершении над этими типами операций может происходить явное и не явное преобразование типа данных.

Для определения типа данных в PostgreSQL используется функция PG_TYPEOF().

Напишем запрос и посмотрим тип данных у значения 100

```
SELECT PG_TYPEOF(100); -- integer
```

Теперь давайте используем одинарные кавычки, что бы попробовать сделать из значения строку:

```
SELECT PG_TYPEOF('100');
```

Какой тип данных вернет данный запрос?

Преобразование типов

Для преобразования одного типа данных к другому используются операторы CAST или ::

```
SELECT CAST(100 AS TEXT) -- преобразует число к тексту
SELECT '01/01/2021'::DATE -- преобразует строку к дате
```

Работа со строками. LIKE / ILIKE

Выражение LIKE возвращает true, если строка соответствует заданному шаблону.

Выражение NOT LIKE возвращает false, когда LIKE возвращает true и наоборот.

Если шаблон не содержит знаков процента и подчёркиваний, тогда шаблон представляет в точности строку и LIKE работает как оператор сравнения.

Подчёркивание (_) в шаблоне подменяет (вместо него подходит) любой символ.

Работа со строками. LIKE / ILIKE

Знак процента (%) подменяет любую (в том числе и пустую) последовательность символов.
LIKE - регистрозависимый поиск ILIKE - регистронезависимый поиск

```
SELECT 'abc' LIKE 'abc'; -- true

SELECT 'abc' LIKE 'a%'; -- true

SELECT 'abc' LIKE '_b_'; -- true

SELECT 'abc' LIKE 'c'; -- false

SELECT 'abc’ ILIKE '%c%'; -- true
```

Функции для работы со строками

Возвращает положение указанной подстроки:

```
SELECT STRPOS('Hello world!', 'world') -- 7
```

Возвращает длину строки:

```
SELECT CHARACTER_LENGTH('Hello world!') -- 12
```

Заменяет подстроку:

```
SELECT OVERLAY('Hello world!' PLACING 'Max' FROM 7 FOR 5) -- Hello Max!
```

Извлекает подстроку:

```
SELECT SUBSTRING('Hello world!' FROM 7 FOR 5) -- world
```

Функции для работы с датами

Получить год:

```
SELECT EXTRACT(YEAR FROM '28.02.2020'::DATE) -- 2020
```

Получить месяц:

```
SELECT EXTRACT(MONTH FROM '28.02.2020'::DATE) -- 2
```

Получить день:

```
SELECT EXTRACT(DAY FROM '28.02.2020'::DATE) -- 28
```

Получить месяц с учётом года:

```
SELECT DATE_TRUNC('MONTH', '28.02.2020'::DATE) -- 2020-02-01 00:00:00
```

Получить день с учётом месяца и года:

```
SELECT DATE_TRUNC('DAY', '28.02.2020'::DATE) -- 2020-02-28 00:00:00
```

Получить текущие дата и время:

```
SELECT NOW() -- текущие дата и время
```

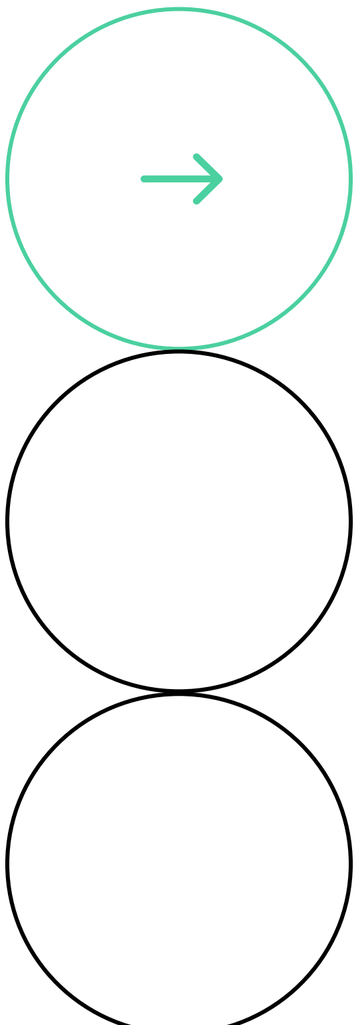
Итоги

6

Николай Хащанов

нетология

Подведем итог



Посмотрели на реляционную модель

Разобрали ограничения Первичных и Внешних ключей

Изучили различные типы данных и работу с ними

Полезные ссылки

Николай Хащанов

нетология

Операции над типами данных

- Математические https://www.postgresql.org/docs/12/functions-math.html
- Строковые https://www.postgresql.org/docs/12/functions-string.html
- Логические https://www.postgresql.org/docs/12/functions-logical.html

Подключение к облачной БД

0

Николай Хащанов

нетология

Подключение

Для подключения к облачной базе данных нужно следовать пунктам инструкции по ссылке https://github.com/netology-ds-team/sql-materials/tree/main/sqlfree

Реквизиты для подключения:

Хост	psql.ntlg.tech
Порт	19001
База данных	sqlfree
Пользователь	netology
Пароль	NetoSQL2019

Подключение

Еще раз напомним, что:

- версии программ могут отличаться от указанных в инструкции
- на курсе используется СУБД PostgreSQL
- при прохождении курса можете использовать любой IDE, DBeaver не обязателен
- если у вас корпоративная машина с жесткими требованиями по безопасности, то пробуйте использовать portable версию DBeaver или потребуется вмешательство IT специалистов вашей компании
- в схеме public находится исходная учебная база данных, доступная только для чтения
- вам нужно создать собственную схему и восстановить в нее дамп учебной базы данных

Спасибо за внимание!

Николай Хащанов

нетология