# Лекция 7

# NoSQL

# NoSQL

NoSQL (not only SQL) — обозначение широкого класса разнородных систем управления базами данных, появившихся в конце 2000-х — начале 2010-х годов и существенно отличающихся от традиционных реляционных СУБД с доступом к данным средствами языка SQL. – *определение из Wiki*

Простыми словами, NoSQL — это общее название для нереляционных баз данных.

# Реляционные структуры данных (слайд из лекции 2)

- Навигационная природа ранних баз данных заставляла разработчиков писать слишком много деталей реализации
- Основной подход: создание такой структуры данных, которая была бы понятна большинству разработчиков БД (не математику и не специалисту), но в то же время несла чёткий математический смысл
- В качестве такой структуры была выбрана **неупорядоченная таблица** (без порядка строк и столбцов)

# Реляционные структуры данных

- Появились задачи и предметные области, которые плохо ложатся на реляционную модель с её «недостатками»:
  - скорость работы
  - избыточная сложность моделирования – не все данные можно засунуть в таблицу

- Появилась потребность в других подходах хранения информации.

# Типы систем

- Ключ-значение
- Колоночные СУБД
- Графовые СУБД
- Документо-ориентированные СУБД

# Ключ — значение

– наиболее простой вариант хранилища данных, использующий ключ для доступа к значению в рамках большой хэш-таблицы.

- Хранит хэш-таблицу ключей, где каждый ключ связан с *непрозрачным бинарным объектом*
- Быстрая запись, но доступ только к одной ячейке
- Стоимость поиска O(1)
- Легко горизонтально масштабируется, идеально для приложений с большими массивами простых данных

# Ключ — значение

Большинство БД «ключ-значение» работают по следующим правилам:

- Идентифицировать элемент можно только по ключу
- Нет агрегатных операций
- Вообще нет сложных операций
- Нет отношений между объектами
- Хранят данные в памяти и определяют позицию элемента в массиве с помощью хеш-функции

# Ключ — значение

- Применяют для хранения кеша, пользовательских сессий, создание очередей для брокера сообщений
- Примеры: Redis, Riak, Amazon DynamoDB, Oracle NoSQL Database, Berkeley DB, MemcacheDB, Tarantool

# Колоночные СУБД

- Иногда называют столбчатой
- БД, в которой данные группируются и хранятся (физически) не по строкам, а по столбцам
- Оптимизированы для быстрого извлечения столбцов данных и применяются, как правило, в аналитических приложениях

# Колоночные СУБД

- Высокая эффективность чтения за счет прямого доступа к нужным столбцам таблицы
- Низкая эффективность записи данных
- Высокая степень сжатия данных

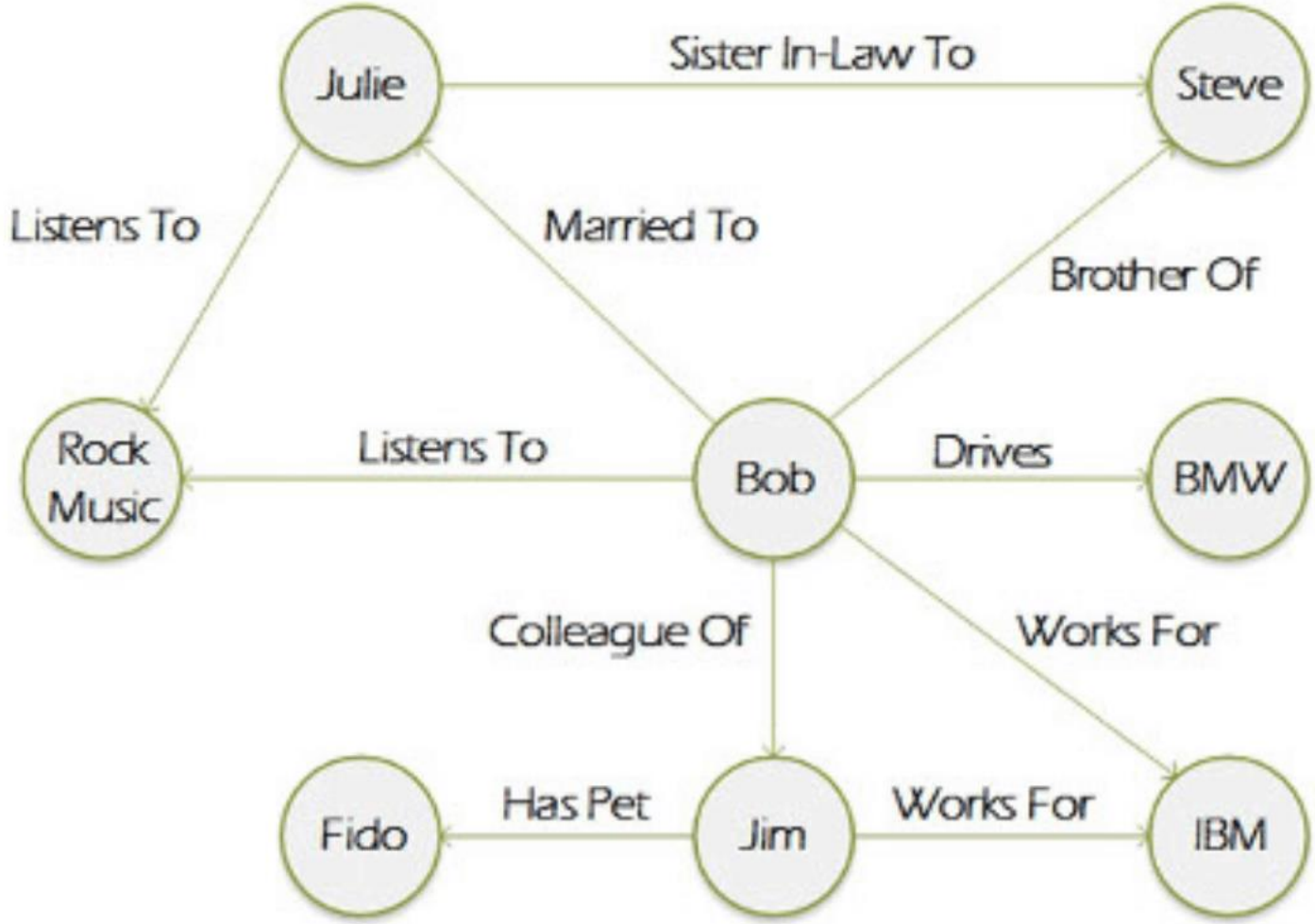
# Колоночные СУБД

- Применяется для задач, связанных с большими данными, с пониженными требованиями к согласованности.
- Примерами СУБД данного типа являются: ClickHouse, Cassandra, ScyllaDB, Hypertable, Greenplum

# Графовые СУБД

– разновидность баз данных с реализацией сетевой модели в виде направленного графа.

Являются обобщением сетевой модели данных.



# Графовые СУБД

- Данные в графовых СУБД представлены в виде узлов и ребер между этими узлами. Узлы — сущности или объекты, а ребра — отношения и связи между этими объектами.
- Так как рёбра графа материализованы, то есть, являются хранимыми, обход графа не требует дополнительных вычислений.

# Графовые СУБД

- Применяются для задач, в которых данные имеют большое количество связей, например, социальные сети, семантические паутины, сервисы рекомендаций.
- Примеры: Neo4j, OrientDB, Amazon Neptune, FlockDB, Titan, Tarantool Graph DB

# Документоориентированная СУБД

— СУБД, специально предназначенная для хранения иерархических структур данных (документов) и обычно реализуемая с помощью подхода NoSQL.

Модель данных – множество наборов ключ-значение.

Основной принцип использования – хранить в одном документе всю информацию об объекте.

# Документоориентированная СУБД

{ "id" : 1, "item" : "abc1", "quantity": 300, "price": 1000}

{ "id" : 2, "item" : "abc2", "quantity": 200, "price": 4000}

{ "id" : 3, "item" : "abc3", "quantity": 250, "price": 600}

# Документоориентированная СУБД

{ "id" : 1, "item" : "abc1", "quantity": 300, "price": 1000}

{ "id" : 2, "item" : "abc2", "quantity": 200, "price": 4000, "rating" : 9}

{ "id" : 3, "item" : "abc3", "quantity": 250, "price": 600, "discount": 10}

# Документоориентированная СУБД

- Данные хранятся как набор документов
- Документы имеют определённую структуру и кодировку данных (XML, JSON, и т.д.)
- Документы могут быть сгруппированы в коллекции
- Строгой структуры нет
- Нет join (бывают упрощенные аналоги)
- Нет ссылочной целостности

# Документоориентированная СУБД

- Документарная база рассчитана на хранение отдельных элементов, не имеющих дополнительных связей между собой.
- Хорошо работает в таких примерах использования, как каталоги, пользовательские профили и системы управления контентом, где каждый документ уникален и изменяется со временем
- Примеры СУБД данного типа — MongoDB, CouchDB, Couchbase

# Теорема Брюера

Теорема CAP (известная также как теорема Брюера) — база данных может обладать не более чем двумя свойствами одновременно:

- согласованность данных (**C**onsistency) — во всех вычислительных узлах в один момент времени данные не противоречат друг другу;
- доступность (**A**vailability) — любой запрос к распределённой системе завершается корректным откликом, однако без гарантии, что ответы всех узлов системы совпадают;
- устойчивость к разделению (**P**artition tolerance) — расщепление распределённой системы на несколько изолированных секций не приводит к некорректности отклика от каждой из секций.

# Набор свойств NoSQL

Основная черта NoSQL-подхода – решение проблемы масштабируемости и доступности за счет атомарности и согласованности данных.

Таким образом, проектировщики NoSQL-систем жертвуют согласованностью данных ради достижения двух других свойств из теоремы CAP.

# Еще про NoSQL

Другими характерными чертами NoSQL-решений являются:

- Применение различных типов хранилищ.
- Возможность разработки базы данных без задания схемы.
- Линейная масштабируемость (добавление процессоров увеличивает производительность).