Лекция №7

Без модельный подход

Р-СУБД связаны жёсткой реляционной моделью, имеют ориентацию на SQL, поэтому применяются в случаях, когда требуется реализация строго типизированных запросов, требуется поддержка целостности и не критично количество пользователей, работающий в онлайне. Наиболее известные СУБД, поддерживающие модель, PostgreSQL, MySQL, SQLite, InterBase.

NoSQL

NoSQL (Mongo) не использует реляционную модель реализации данных, существует множество собственных реализаций для конкретных задач, без схемность решений допускает неограниченное формирование записей и хранение данных в виде связки (key → value).

Отличия от основных Р-СУБД:

1. Можно группировать коллекции данных с другими данными.
2. СУБД хранят данные как единое целое.
3. Не используется общий формат запроса, каждое решение использует собственную систему запросов.

Сравнение SQL и NoSQL:

1. По структуре и типу хранящихся данных
   SQL Р-СУБД требуют определённую структуру, в отличии от NoSQL
2. Запросы
   Р-СУБД реализуют SQL стандарты на основе специальной математики (РА и РИ)
   NoSQL реализует свой способ запроса данных
3. Масштабируемость
   Оба решения легко масштабируются вертикально (путём увеличения системных ресурсов)
   Горизонтальное масштабирование легче для NoSQL (Н-р, с помощью создания кластера из нескольких машин)
4. Надёжность
   «SQL решения однозначно выигрывают» © Доронина Ю. В.
5. Поддержка
   Поскольку Р-СУБД имеют долгую историю, хорошо исследованы и изучены – поддержка проста и понятна
   NoSQL – сложнее, в основном поддержка реализуется разработчиками
6. Поддержка и доступ к сложным структурам
   Р-СУБД предполагают работу со сложными структурами, чем превосходят NoSQL решения
7. В NoSQL не во всех случаях отсутствует структура, поэтому решения, связанные с созданием новой структуры, в отличии от SQL, реализуется значительно проще

Проблемы: Если мы не меняем логику приложения, значит мы ожидаем новое поле при чтении. Вариант решения:

1. Обновление поля во всех существующих документах
2. Проверка в коде приложения

Достоинства

+ «Эффективность работы с разреженными данными.» © Доронина Ю. В.
Столбцы не объявляются декларативно и могут меняться во время пользовательской сессии.

± Отсутствие ограничений со стороны БД (Н-р, not null, unique, check)

± Сложности в контроле структуры при параллельной работе с БД разных проектов (словари БД отсутствуют)

Представление данных в виде агрегатов

Во всех случаях реализуется «так называемое агрегатное моделирование, когда сущности объединяются в один логический объект, при этом хранятся конкретные

реализации позиций, что приводит к возможности минимума ДЖОИНОВ между объектами» © Доронина Ю. В.

Таким образом главное правило проектирования в NoSQL – это подчинение требованиям приложения и оптимизация под частые запросы. Из-за этого работа с большими денормализованными объектами и попытками произвольных доступов к данным приводит к тому, что «приходится извлекать целые агрегаты, большая часть информации из которых нам не нужна» © Доронина Ю. В.

	Нормализация	Данные в виде агрегатов
1	Целостность информации при обновлении (запись изменяется в одной таблице, а в связанных автоматически корректируется)	Оптимизация только под определённый вид запросов
2	Ориентированность на широкий спектр запросов	Сложности при обновлении денормализованных данных
3	Неэффективность в распределённой среде	Лучший способ добиться большой скорости на чтение в распределённой среде
4	Низкая скорость чтения при использовании JOIN	Поддержка атомарности на уровне записи
5	Несоответствие объектной модели приложения физической структуре данных	Возможность хранить физические объекты в том виде, в котором с ними работает приложение (легче кодировать и меньше ошибок при преобразовании)

Проблема NoSQL: для огромных массивов информации в распределенных системах обеспечить транзакционность операций, высокую доступность и быстрое время отклика невозможно.

«Крупнейшие интернет-компании заявляют, что максимальный отклик транзакции около 1 секунды» © Доронина Ю. В.

Распределённые системы без совместно используемых ресурсов – главная задача развития NoSQL. С интенсивным ростом информации и скоростью обработки встала проблема вертикальной масштабируемости. Единственный выбор – это горизонтальное масштабирование, когда несколько независимых серверов соединяются быстрой сетью и каждый обрабатывает только часть данных или часть запросов.

Одно из важных свойств NoSQL – репликация, которая позволяет добиться масштабируемости, повысить доступность и сохранность данных.

Виды репликации:

1. С мастер-узла
2. По кругу

Шардинг – распределение данных по узлам.

NoSQL развивается как простое средство манипулирования не критичными структурами данных. Современный «серьёзные» СУБД работают на взаимодействии SQL и NoSQL.