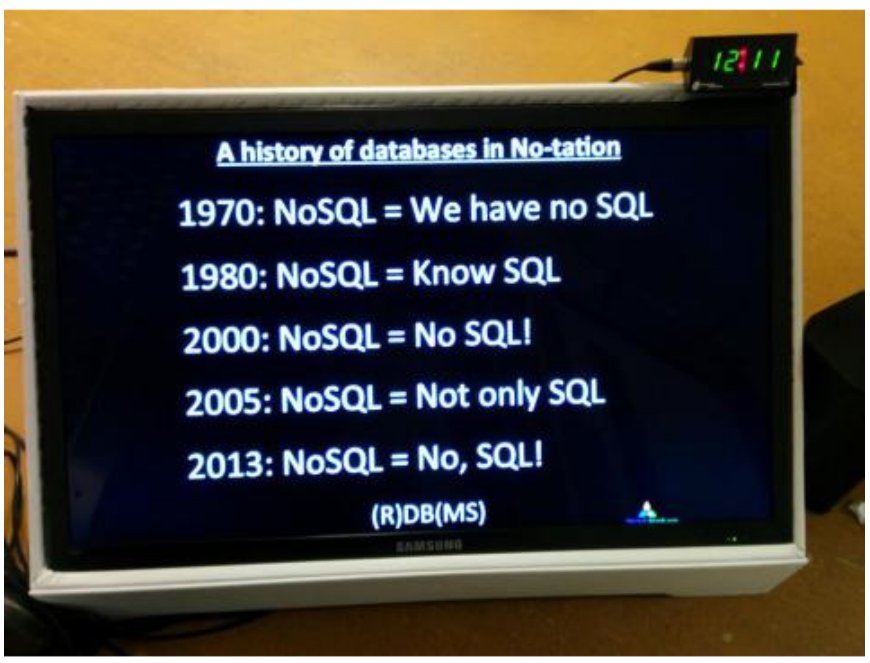
Знакомство с NoSQL БД



History of NoSQL by Mark Madsen. Picture published by Edd Dumbill

Базы данных NoSQL появились потому, что **традиционные базы данных, существовавшие на момент их изобретения, не могли справляться с требуемыми масштабами задач.**

Причины появления NoSQL

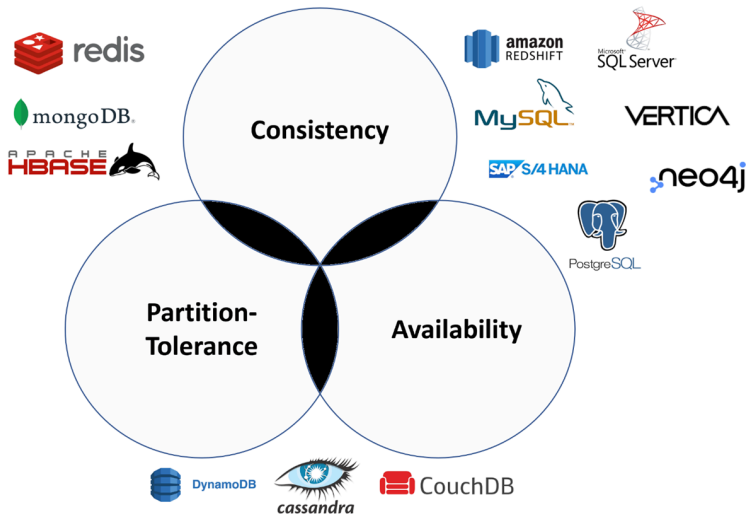
- В**озникла потребность в распределённых СУБД.** Масштабированию реляционные БД поддаются плохо: чем больше в системе серверов, тем больше усилий требуется для поддержания согласованности данных в узлах;
- **Работа с данными ускорилась.** В отличие от нереляционных БД, SQL запрашивает данные из нескольких таблиц. А когда количество информации растёт, таблиц и связей становится слишком много — скорость получения ответа снижается;
- **Разработчики стремились избавиться от ограниченности реляционных схем.** Жёсткая реляционная модель подходит не для всех предметных областей. Иногда получается либо нагромождение избыточного количества таблиц, либо плохо структурированная предметная область.

Выбор в пользу NoSQL

Выбрать NoSQL стоит в тех случаях, когда:

- Нужно хранить большие объёмы неструктурированных или быстро меняющихся данных;
- Нет возможности описать схему СУБД до начала проектирования базы данных или предполагается, что в процессе работы архитектура хранилища будет меняться;
- Необходимо быстро, без большой предварительной подготовки, запустить прототип продукта;
- Требуется высокая масштабируемость системы без лишних ресурсозатрат;
- Строгой согласованностью можно пожертвовать ради производительности и доступности.

Теорема CAP



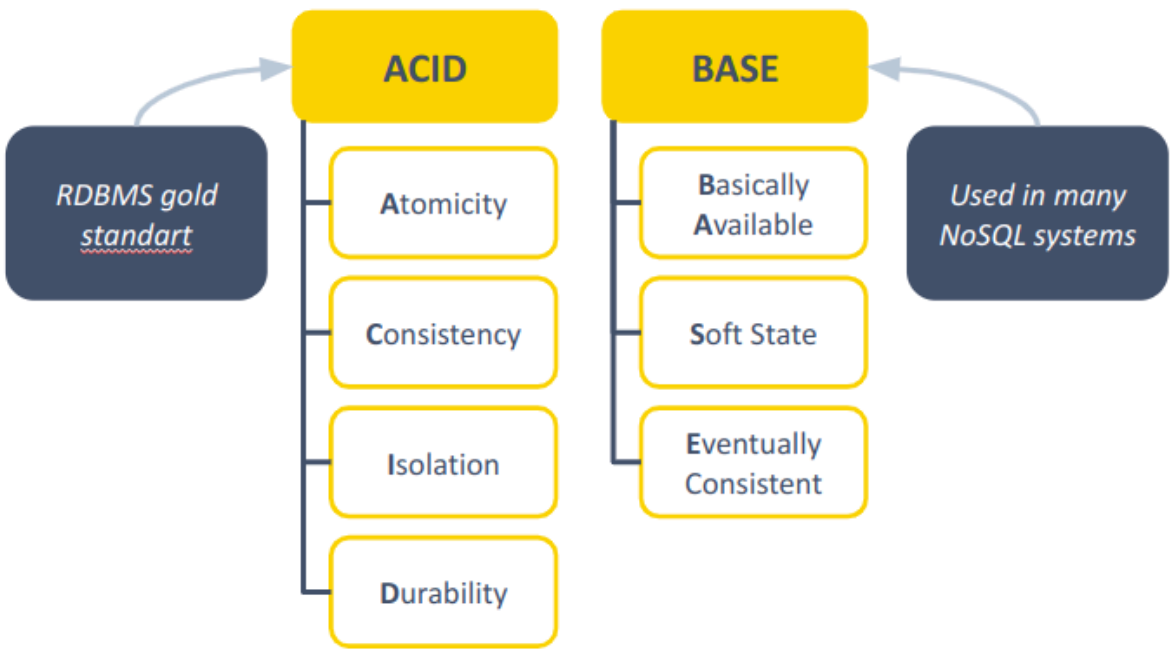
Теорема CAP (известная также как теорема Брюера) — в любой реализации распределённых вычислений возможно обеспечить не более 2 из 3 следующих свойств:

- **Согласованность данных (consistency)** — во всех вычислительных узлах в один момент времени данные не противоречат друг другу;
- **Доступность (availability)** — любой запрос к распределённой системе завершается корректным откликом, однако без гарантии, что ответы всех узлов системы совпадают;
- **Устойчивость к разделению (partition tolerance)** — расщепление распределённой системы на несколько изолированных секций не приводит к некорректности отклика от каждой из секций.

Например, свойствами **Consistency + Partition tolerance** обладают большинство NoSQL БД: Apache HBase, MongoDB, Redis, MemcasheDB, Berkley DB, HyperTable и Google Big Table.

BASE архитектура

BASE (Basically Available, Soft-state, Eventually consistent) — базовая доступность, неустойчивое состояние, согласованность в конечном счёте



Сравнение терминологии

SQL	MongoDB	DynamoDB	Cassandra	Couchbase
Таблица	Коллекция	Таблица	Таблица	Корзина данных
Ряд	Документ	Элемент	Ряд	Документ
Столбец	Поле	Атрибут	Столбец	Поле
Первичный ключ	ObjectId	Первичный ключ	Первичный ключ	ИД документа
Индекс	Индекс	Вторичный индекс	Индекс	Индекс
Представление	Представление	Глобальный вторичный индекс	Материализованное представление	Представление
Вложенная таблица или объект	Встроенный документ	Карта	Карта	Карта
Массив	Массив	Список	Список	Список