# Introduction to NoSQL Data Stores

MSBA 6330 Prof Liu

1



## Learning Objectives

- Understand strengths and limitations of traditional SQL databases
- Understand common characteristics of NoSQL databases
- Understand the trade-offs facing distributed databases in light of the CAP theorem
- Distinguish between ACID and BASE consistency models
- Be familiar with different types of NoSQL / NewSQL databases and their use cases
- Understand the complexity of choosing different database technologies for big data applications

2

Carlson School of Management

## Topics

- Limitations of RDBMS for the Web
- What is NoSQL?
- NoSQL and the CAP theorem
- Different kinds of NoSQL data stores
- NewSQL
- How to choose?

3

Introduction to NoSQL

# LIMITATIONS OF RDBMS FOR THE WEB

4



## RDBMS has been a dominant data storage model for a few decades

- Relational databases have been a mainstay of how data is stored since 1970s (and still are)
  - Shines with complicated queries
  - Declarative queries (SQL) rather than programing
  - Fast query & analysis, but not necessarily on large datasets
- Strong support for transactions – ACID Properties
  - **Atomic** – All of the work in a transaction completes (commit) or none of it
  - **Consistent** – A transaction transforms the database from one consistent state to another consistent state. Consistency is defined in terms of constraints.
  - **Isolated** – The results of any changes made during a transaction are not visible until the transaction has committed.
  - **Durable** – The results of a committed transaction survive failures

5



## But there are limits in scaling RDBMSs

- Best way to provide ACID and a rich query model is to have the dataset on a single machine.
- However, there are limits to scaling up (vertical scaling) as the dataset gets big
- Past a certain point, an organization will find it is cheaper and more feasible to scale out (horizontal scaling) than scaling up
- DBAs began to look at ways of scaling out
  - *master-slave*: all writes go to the master. all reads can perform against a fleet of replicated slave databases (inconsistency; do not scale well)
  - *sharding*: partition database based on value range, hash-key, or dictionaries (unable to join across partitions; no referential integrity)

http://adam.herokuapp.com/past/2009/7/6/sql_databases_dont_scale/

6

## Inadequacy of RDMS for the web

- Hooking your RDBMS to a web-based application was a recipe for headaches
  - "Slashdot effect": when a popular website links to a smaller website, causing a massive spike in traffic
- Sharding is like putting a bandage on a broken leg
  - It does not scale to handle hundreds of thousands of visitors in a short-time span
  - People with the largest datasets (terabyte/petabyte), e.g. Google, Facebook, Twitter, began to explore alternative ways to store data, especially around 2008/2009.

7



Introduction to NoSQL

## WHAT IS NOSQL?

8



## Drivers of the NoSQL

- Two papers were the seeds of the NoSQL movement
  - BigTable (Google, 2006):
    - Wide-column data model
    - Consistent, fault-tolerant, scale to large clusters (1000+ nodes)
  - Dynamo (Amazon, 2007):
    - Distributed key-value data store
    - Eventual consistency, always writable, fault-tolerant
- In addition,
  - The advent of Amazon S3 signals to others it was okay to look at alternative storage solutions
  - Many NoSQL solutions are open source software

9

## What is NoSQL?

- Stands for **N**ot **O**nly **SQL** (coined by Eric Evans)
  - *"to solve a problem that relational databases are a bad fit for."*
- (No standard definition) nosql-database.org:
  - Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontal scalable.

*NoSQL-Databases.org:*

*Current list has over 150 NoSQL systems*

10

Carlson School of Management

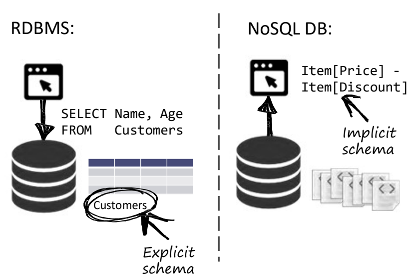
## Common NoSQL Characteristics

- don't use the relational data model
- optimized for running on a cluster
- open source
- schema-less (schema-free)
- tunable consistency

11

Carlson School of Management

## Schema-free Data Modeling




12

## NoSQL Paradigm Shift: Open Source & Commodity Hardware

| | |
|---|---|
| Commercial DBMS | Open-Source DBMS |
| Specialized DB hardware (Oracle Exadata, etc.) | Commodity hardware |
| Highly available network (Infiniband, Fabric Path, etc.) | Commodity network (Ethernet, etc.) |
| Highly Available Storage (SAN, RAID, etc.) | Commodity drives (standard HDDs, JBOD) |

13



## NoSQL Pros and Cons

- Pros:
  - Massive scalability
  - Flexible schema
  - Quicker/cheaper to set up
  - Relaxed consistency (NO ACID) → higher performance & availability
- Cons:
  - No declarative query language (SQL) → more programming
    - May not integrate well with other applications that support SQL
    - May lose powerful SQL queries e.g., joins, group by, order by
  - Relaxed consistency → fewer guarantees (NO ACID)

14



## Challenge: Coordination

- The solution to availability and scalability is to decentralize and replicate functions and data…but how do we coordinate the nodes?
  - data consistency
  - update propagation
  - mutual exclusion
  - consistent global states
  - group membership
  - group communication
  - event ordering
  - distributed consensus
  - quorum consensus

15

Introduction to NoSQL

# NOSQL AND THE CAP THEOREM

16

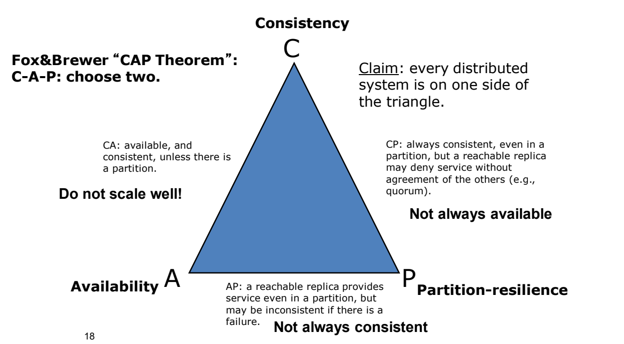


## CAP Theorem

- Proposed as a conjecture by Eric Brewer (Berkeley ,1999)
  - Subsequently proved by Gilbert and Lynch (MIT) in 2002 as a theorem.
- In a distributed data store you can satisfy **at most 2 out of the 3 guarantees**:
  - **Consistency**: Every read receives the most recent write or an error
  - **Availability**: Every request receives a (non-error) response, without the guarantee that it contains the most recent write
  - **Partition-tolerance**: the system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes
- Proof: When a network partition failure happens, should we
  - Cancel the operation and thus decrease the availability but ensure consistency?
  - Proceed with the operation and thus provide availability but risk inconsistency?

17






18

## Availability

- Traditionally, thought of as the server/process available five 9's (99.999 %).
- However, for large node system, at almost any point in time there's a good chance that a node is either down or there is a network disruption among the nodes.
  - Want a system that is resilient in the face of network disruption

19

Carlson School of Management

## Consistency Model

- A consistency model determines rules for visibility and apparent order of updates.
- For example:
  - Row X is replicated on nodes M and N
  - Client A writes row X to node N
  - Some period of time t elapses.
  - Client B reads row X from node M
  - Does client B see the write from client A?
- Consistency is a continuum with tradeoffs
  - CAP Theorem states: Strict Consistency can't be achieved at the same time as availability and partition-tolerance.

20

Carlson School of Management

## Eventual Consistency

- When no updates occur for a long period of time, eventually all updates will propagate through the system and all the nodes will be consistent
- For a given accepted update and a given node, eventually either the update reaches the node or the node is removed from service
- Known as **BASE (Basically Available, Soft state, Eventual consistency)**, as opposed to ACID
  - *Basically Available*: possibilities of faults but not a fault of the whole system
  - *Soft state*: copies of a data item may be inconsistent
  - *Eventually Consistent*: copies becomes consistent at some later time if there are no more updates to that data item

21

## Implications of BASE consistency model

- Characteristics
  - Weak consistency – stale data OK
  - Availability first
  - Best effort
  - Approximate answers OK
  - Aggressive (optimistic)
  - Simpler and faster

22

Carlson School of Management

Introduction to NoSQL

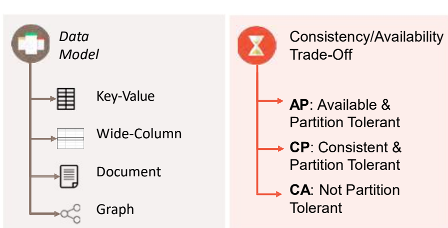
# DIFFERENT KINDS OF NOSQL DATA STORES

23

Carlson School of Management

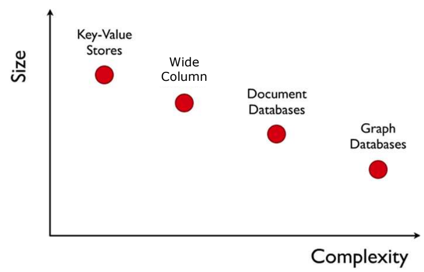
## NoSQL System Classification

- two common criteria




24

## NoSQL Data Models: Size and Complexity




25

Carlson School of Management

Introduction to NoSQL

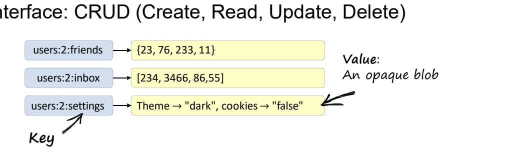
## KEY-VALUE STORES

26

Carlson School of Management

## Key-Value Stores

- Data model: (key) -> value
- Interface: CRUD (Create, Read, Update, Delete)




- Examples: Amazon Dynamo (AP), Riak (AP), Redis (CP)

High availability but not always consistent

Strict consistency
but not always available
during partitions

27

## Key-Value Store Pros and Cons

- Pros:
  - very fast
  - very scalable
  - simple model
  - able to distribute horizontally
- Cons:
  - many data structures (objects) can't be easily modeled as key value pairs
  - Only primary index: lookup by key

28



## Key-value Store Use Cases

- Key-value data stores are ideal for storing user profiles, blog comments, product recommendations (user preferences), session information, shopping cart data.
  - Twitter uses **Redis** to deliver your Twitter timeline
  - Pinterest uses **Redis** to store lists of users, followers, unfollowers, boards, and more
  - Quora uses **Memcached** to cache results from slower, persistent databases
- avoid if we need to
  - query the database by specific data value
  - need relationships between data values
  - need to operate on multiple unique keys

29



Introduction to NoSQL

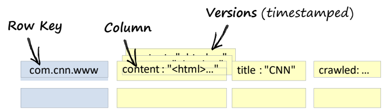
## WIDE-COLUMN STORES

30

Carlson School of Management

## Wide-Column Stores

- Data model: (rowkey, column, timestamp) -> value
  - can be interpreted as two/three dimensional key-value store
  - You can group related columns into **column families**.
- Interface: CRUD, Scan




- Examples: Cassandra (AP), Google BigTable (CP), HBase (CP)

High availability but not always consistent

Strict consistency but not always available during partitions

31

Carlson School of Management

## Wide-column Store Advantages & Use Cases

- Column stores offer very high performance and a highly scalable architecture. They are fast to load and query
  - Ideal for real-time data logging, analysis, and random read/write access of huge amounts of data.
  - But does not support transactions
- Popular among companies and organizations dealing with big data, IoT, and user recommendation and personalization engines.
  - Spotify uses **Cassandra** to store user profile attributes and metadata about artists, songs, etc. for better personalization
  - Facebook builts its revamped Messages on top of **HBase**, but is now also used for Nearby Friends feature and search
  - Outbrain uses **Cassandra** to serve over 190 billion personalized content recommendations each month

32

Carlson School of Management

Introduction to NoSQL

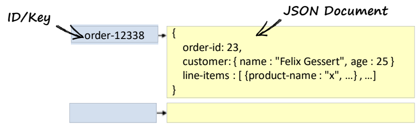
## DOCUMENT STORES

33

Carlson School of Management

## Document Stores

- Data model: (collection, key) -> document (JSON, XML, BSON)
  - can index common columns for faster performance
- Interface: CRUD, Queries, Map-Reduce




- Examples: CouchDB (AP), RethinkDB (CP), MongoDB (CP)

34

Carlson School of Management

### Document Databases Advantages & Use Cases

- Can store data with flexible and evolving schema, scale out well, and perform fast ad hoc queries using secondary indexes.
- Good for content management systems, blogging platforms, analytics platforms, e-commerce platforms
  - SEGA uses cloud hosted MongoDB for handling 11 million in-game accounts
  - Aer Lingus uses MongoDB with Studio 3T to handle ticketing and internal apps
  - Built on MongoDB, The Weather Channel's iOS and Android apps deliver weather alerts to 40 million users in real-time
- avoid if one must run complex search queries or require complex transactions

35

Carlson School of Management

Introduction to NoSQL

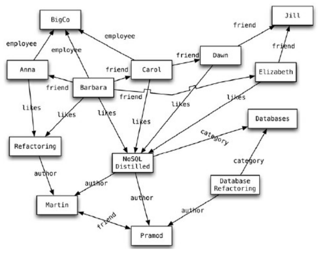
## GRAPH DATABASES

36

Carlson School of Management

## Graph Databases

- Some data are more naturally represented as a graph

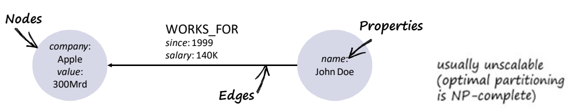



37

Carlson School of Management

## Graph Databases

- Data model: G = (V, E): Graph-Property Model
- Interface: Traversal algorithms, queries, transactions




- Examples: Neo4j (CA), InfiniteGraph (CA), OrientDB (CA)

38

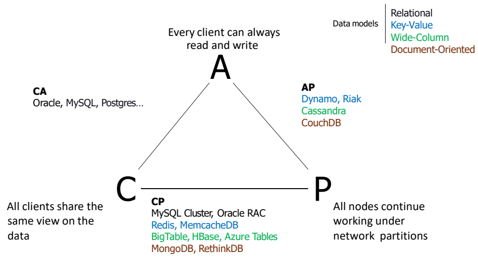
Carlson School of Management

## Graph Database Advantages and Use Cases

- Graph databases are great for establishing data relationships especially when dealing with large data sets.
  - Fault detection, real-time recommendation engines, master data management, network and IT operations, identity and access management.
- They offer blazing fast query performance and flexibility that relational databases cannot compete with, especially as data grows much deeper.
  - Walmart uses Neo4j to provide customers personalized, relevant product recommendations and promotions
  - Medium uses Neo4j to build their social graph to enhance content personalization
  - Cisco uses Neo4j to mine customer support cases to anticipate bugs

39

## NoSQL Triangle




Nathan Hurst: Visual Guide to NoSQL Systems
http://blog.nahurst.com/visual-guide-to-nosql-systems

40



Introduction to NoSQL

## NEWSQL

41



## NewSQL

- NewSQL refers to a new class of databases providing transactional consistency and massive scalability
  - Transactional support of SQL + NoSQL massive scalability
  - Some guarantee stronger consistency, others (MemSQL) provide tunable consistency
  - NewSQL favors consistency over availability (CP)
  - They are cloud-first databases
    - lock-free concurrency control + share nothing architecture
- Examples
  - SAP HANA, NuoDB, MemSQL, VoltDB, Google Spanner, CockRoachDB

42

Carlson School of Management

Introduction to NoSQL

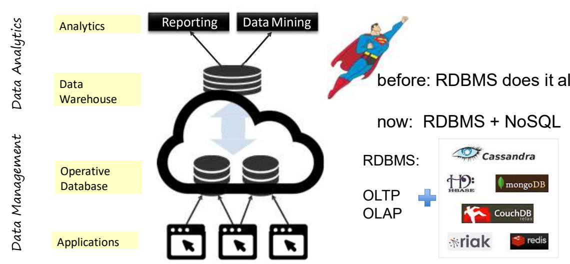
## HOW TO CHOOSE?

43

Carlson School of Management

## The era of one-size-fits-all database systems is over

- Typical Data Architecture:




44

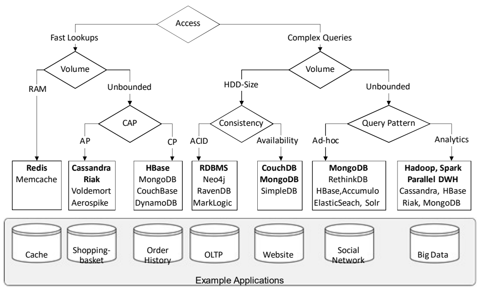
Carlson School of Management

## Which database to choose? Why?

- Large part of the choice depends on the business use-case and tradeoffs between scale, consistency, and availability
  - The CAP Theorem
- It also depends on the type of queries you plan to perform on the data?
  - Are they simple look ups or more complex queries?
  - Different SQL/NoSQL has different
- One need also take into account what is the cost of not having ACID?

45

## NoSQL Decision Tree



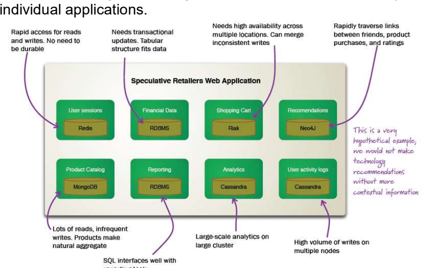

source: [An In-Depth Tutorial on NoSQL Data Stores](), by Gesset, Wingerath, and Ritter, 2017

46



## Using multiple data storage technologies

- Using multiple data storage technologies, chosen based upon the way data is being used by individual applications.




47



## NoSQL - Summary

- NoSQL was initially motivated by inability of traditional RDBMS in dealing with web scale data request
- NoSQL databases reject:
  - Overhead of ACID transactions
  - “Complexity” of SQL
  - Burden of up-front schema design
  - Declarative query expression
- But CAP theorem dictates that there is no one size-fit all solution
  - NewSQL embraces ACID and scalability at the cost of availability
- Choices of form of databases depend on business use-case and nature of data

48