
# Welcome to NoSQL session

# Scale (Out / Up)

Horizontal Scaling / scale out

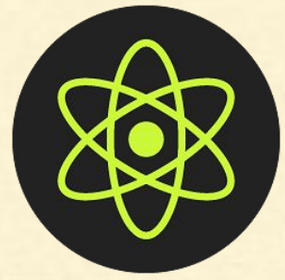
Vertical Scaling / Scale Up

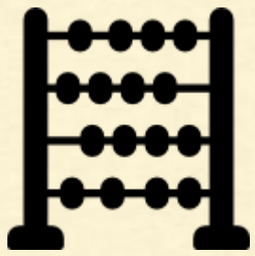
When a service need to be scaled?

# **ACID** - Properties of DB Transactions

**A**tomicity

Either all or nothing





**C**onsistency

DB to be in Valid State

**I**solation

No two transactions mingle

**D**urablity

Transaction is Saved

# RDBMS - Story

1.Initial public launch

*Move from local workstation to shared, remotely hosted MySQL instance with a well-defined schema.*

2. Service becomes more popular; too many reads hitting the database

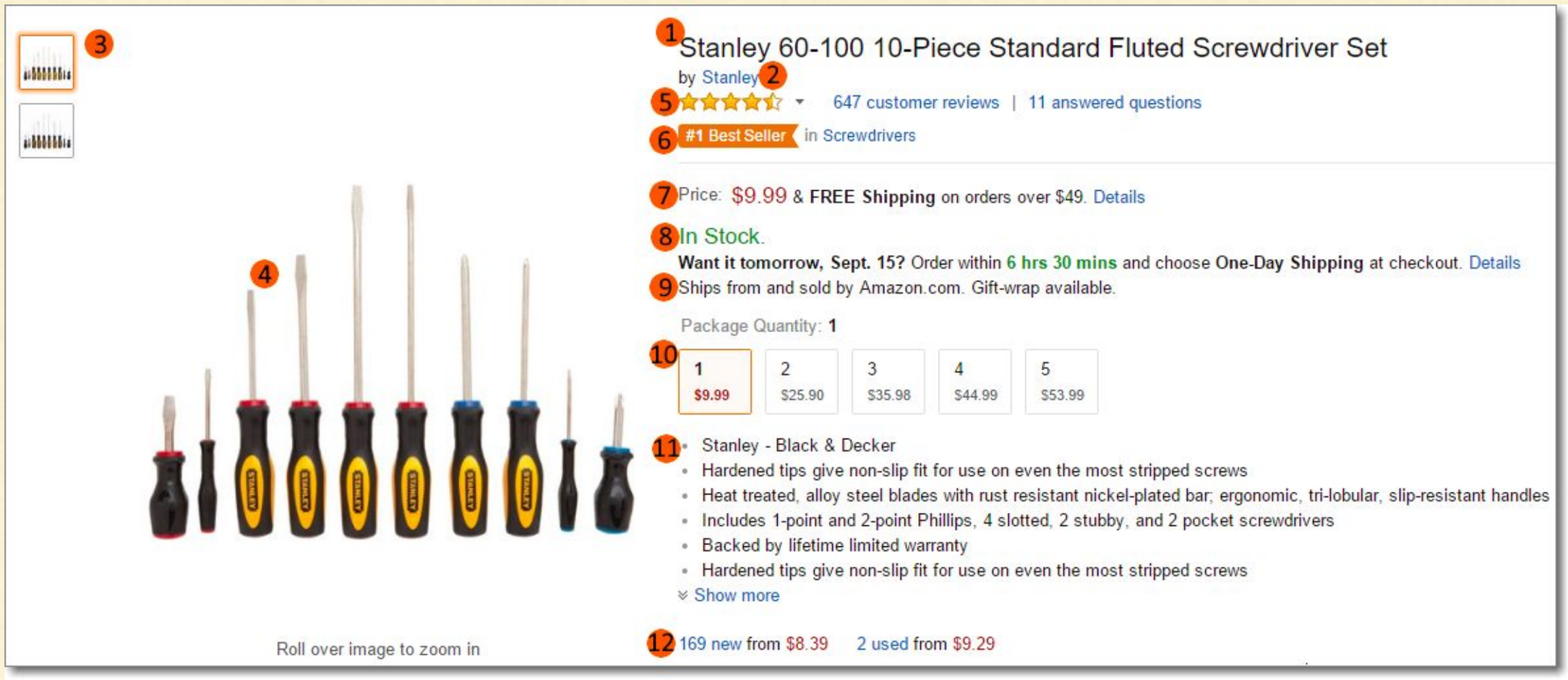
*Add memcached to cache common queries. Reads are now no longer strictly ACID; cached data must expire.*

3. Service continues to grow in popularity; too many writes hitting the database

*Scale MySQL vertically by buying a beefed-up server with 16 cores, 128 GB of RAM, and banks of 15 k RPM hard drives. Costly.*

# RDBMS - Story

## 4.New features increase query complexity; now we have too many joins




*De-normalize your data to reduce joins.*
*(That's not what they taught me in DBA school!)*

# RDBMS - Story

5. Rising popularity swamps the server;

*Things are too slow. Stop doing any server-side computations.*

6. Some queries are still too slow

*Periodically pre-materialize the most complex queries, and try to stop joining in most cases.*

7. Reads are OK, but writes are getting slower and slower

Drop secondary indexes and triggers (no indexes?).

# RDBMS Story - Then why RDBMS?

So, we are left with:

- No ACID properties due to caching
- No Normalized schema
- No stored procedures, triggers and secondary indexes

# What is NoSQL?

1. Big Data

2. High Availability
   Cater to many users

3. Scale-out architecture

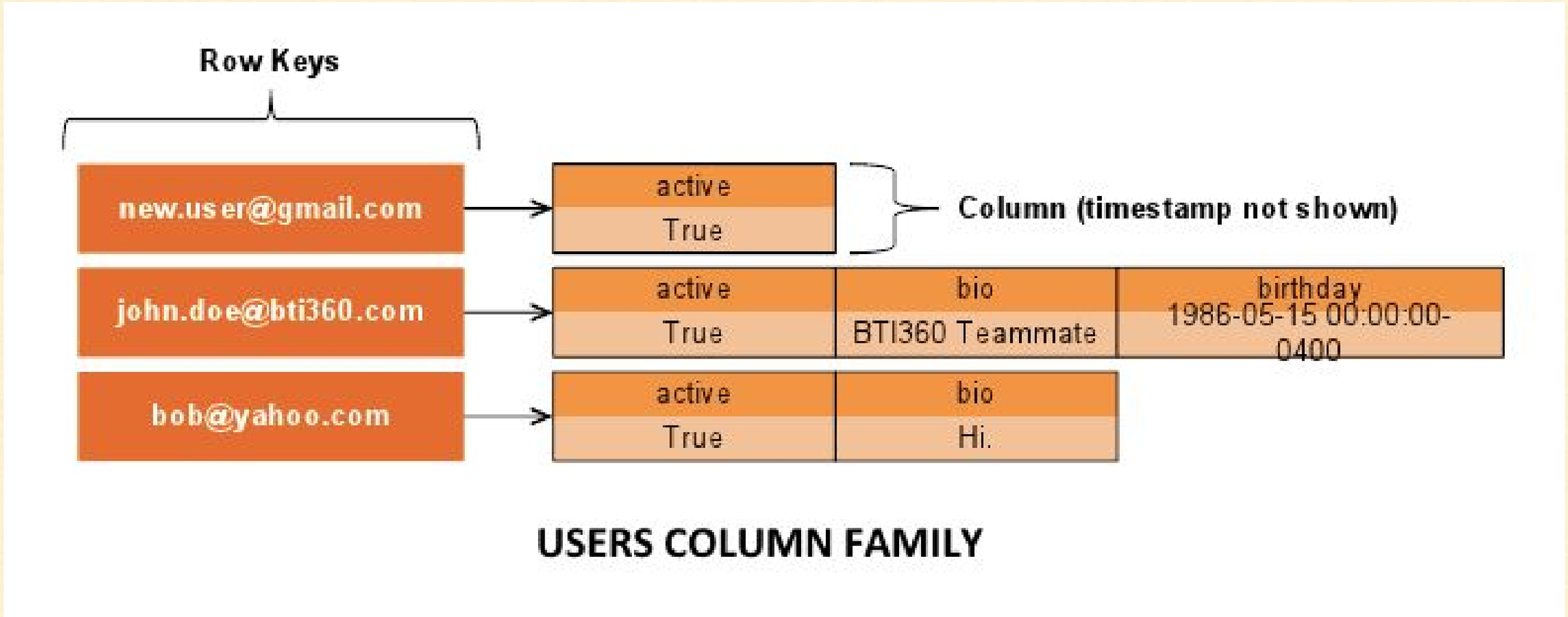
   Commodity hardware

# What is NoSQL?

- Non-Relational - Schema-Free
- Open-Source

# Types Of NoSQL Stores

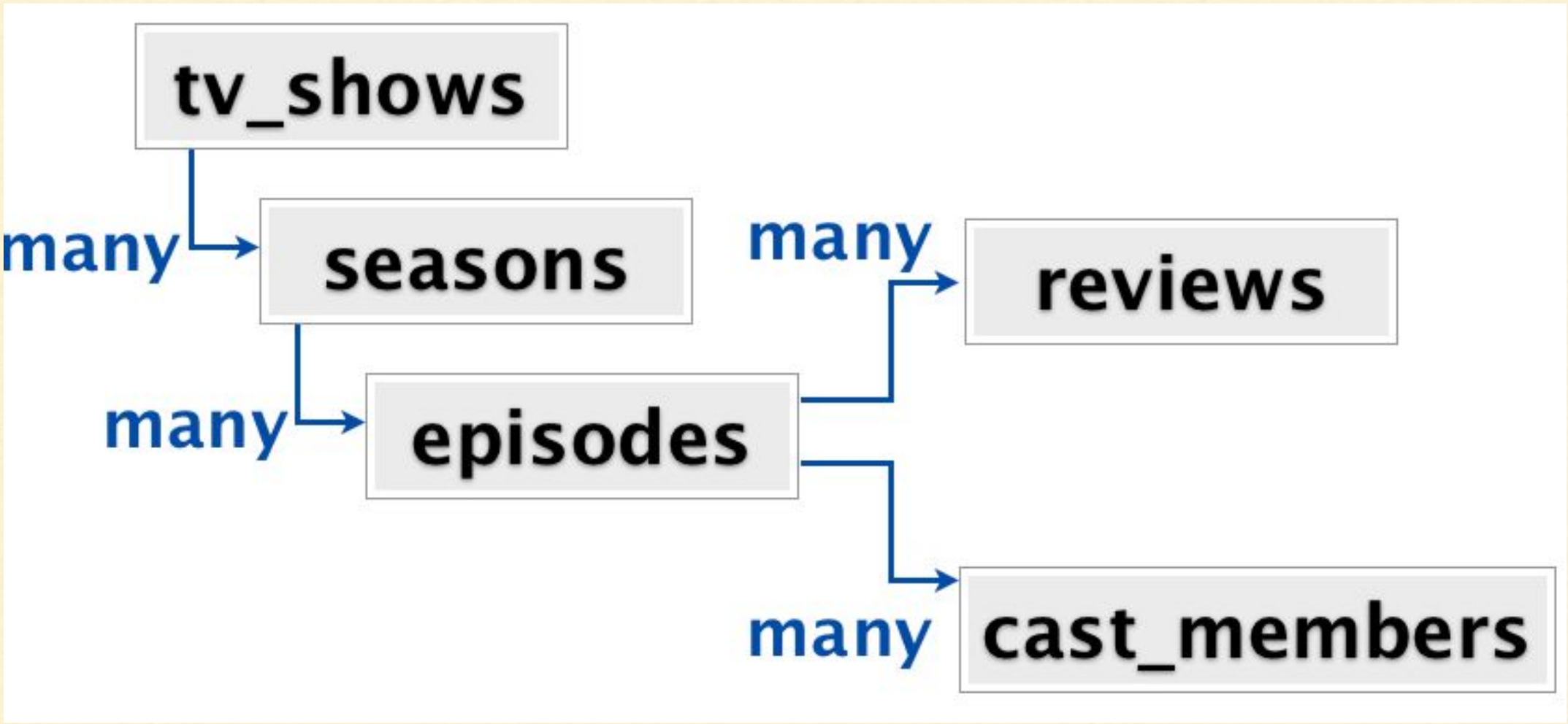
## 1. Column Oriented / wide-column




HBase, Cassandra, Accumulo

# Types Of NoSQL Stores

## 2. Document Oriented




MongoDB, Couchbase, Clusterpoint, MarkLogic

# Types Of NoSQL Stores

### 3. Key-value data store

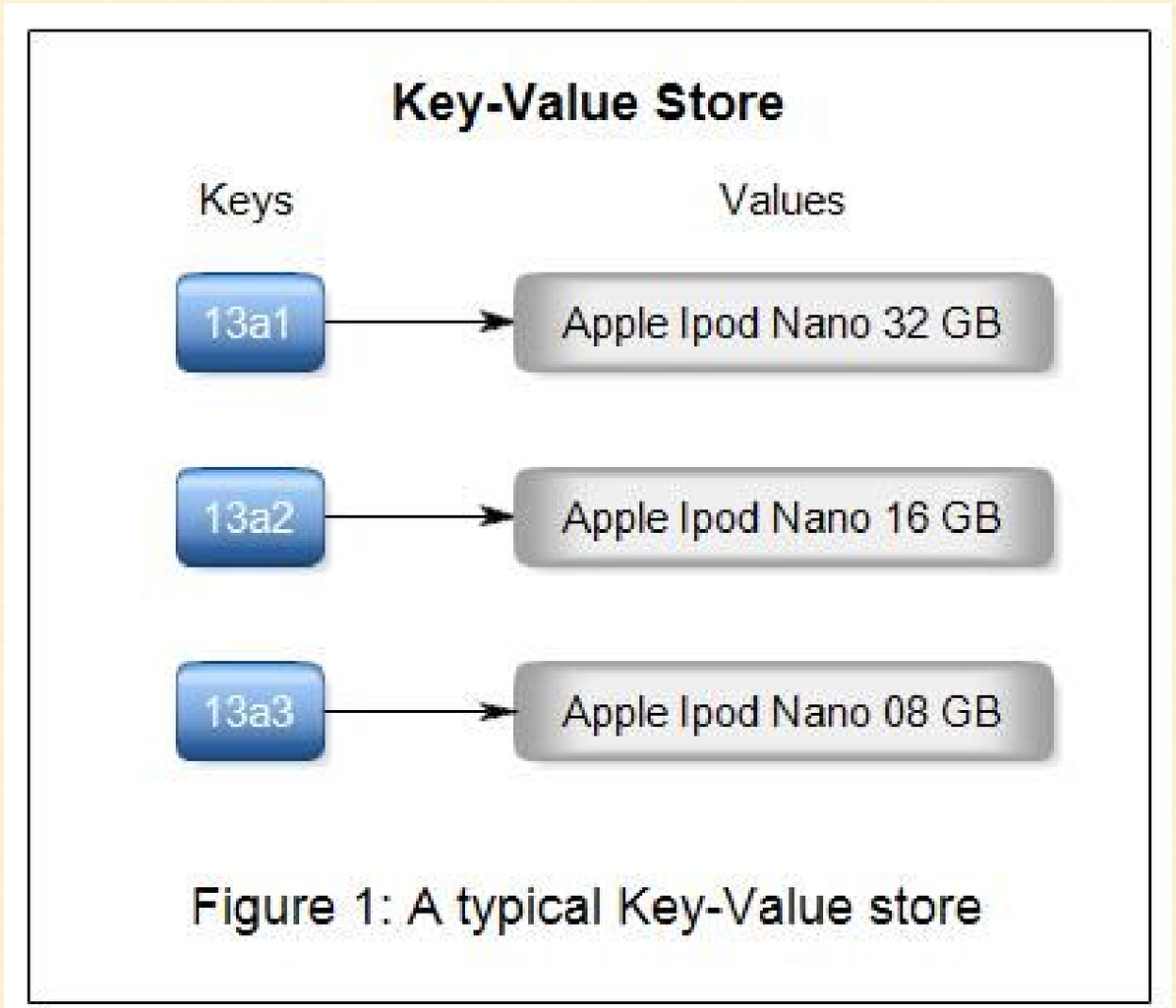
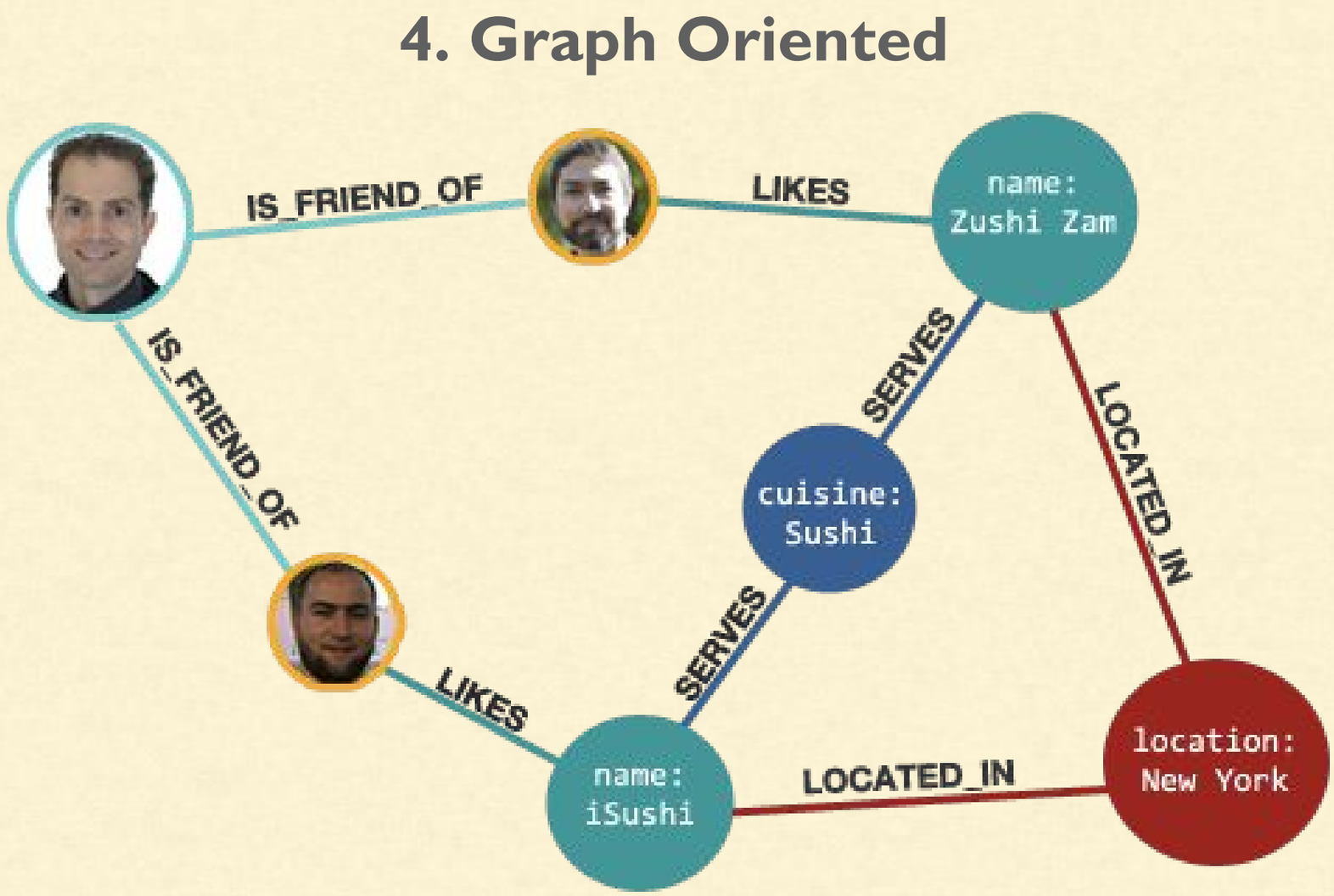



Figure 1: A typical Key-Value store

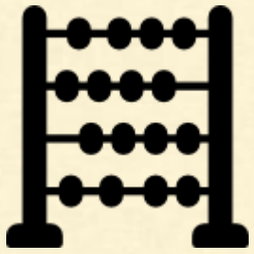
Dynamo, MemcacheDB, Project Voldemort, Redis, Riak

# Types Of NoSQL Stores




Allegro, Neo4J, OrientDB, Virtuoso, Giraph

# CAP Theorem - At most 2 out of 3



1. **C**onsistency

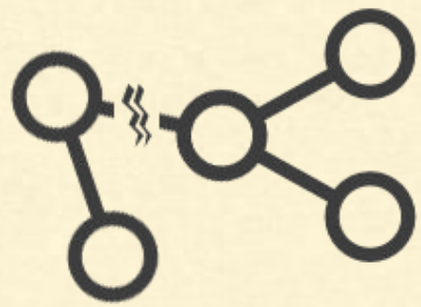
All nodes see the same data at the same time

**2. Availability**

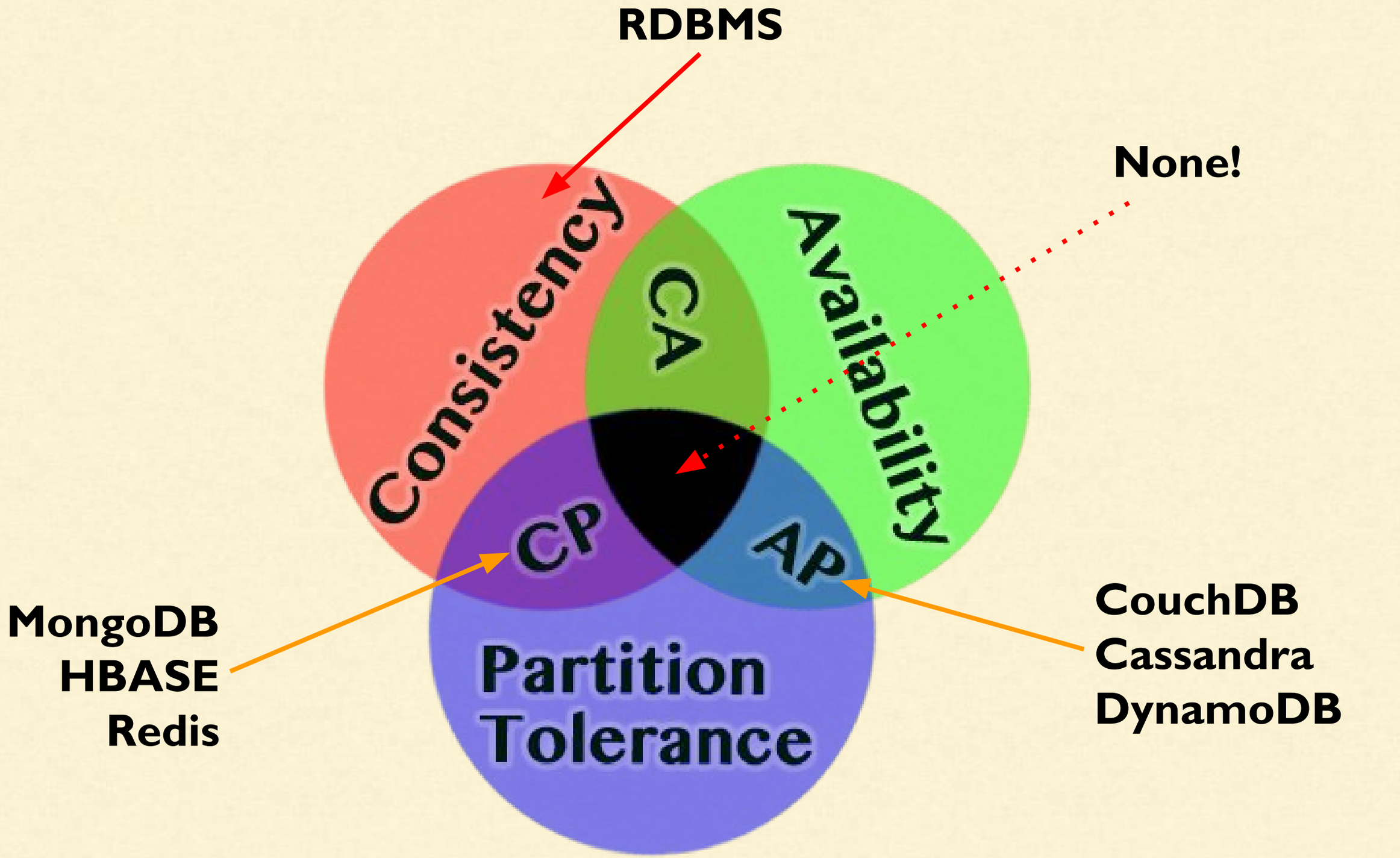
A guarantee that every request receives a response about whether it was successful or failed

**3. Partition tolerance**

The system continues to operate despite arbitrary message loss or failure of part of the system



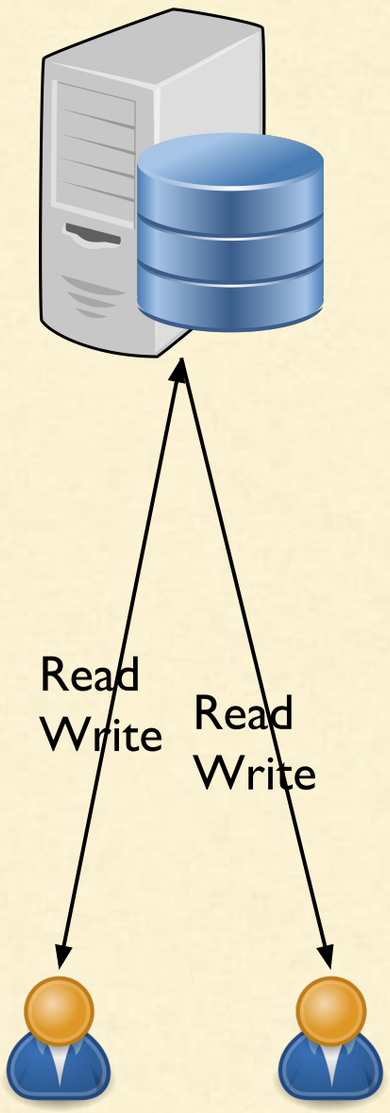
# CAP Theorem


RDBMS
None!
Consistency
CA
Availability
CP
AP
Partition Tolerance
MongoDB
HBASE
Redis
CouchDB
Cassandra
DynamoDB

## *Cap Theorem*

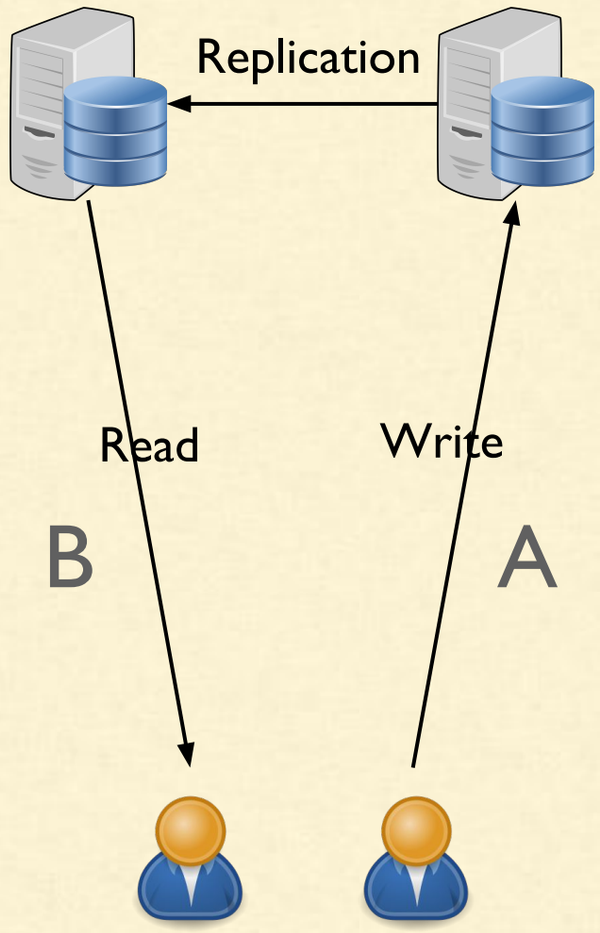
### RDBMS

**C**onsistent
**A**vailable
~~**P**artition Tolerant~~



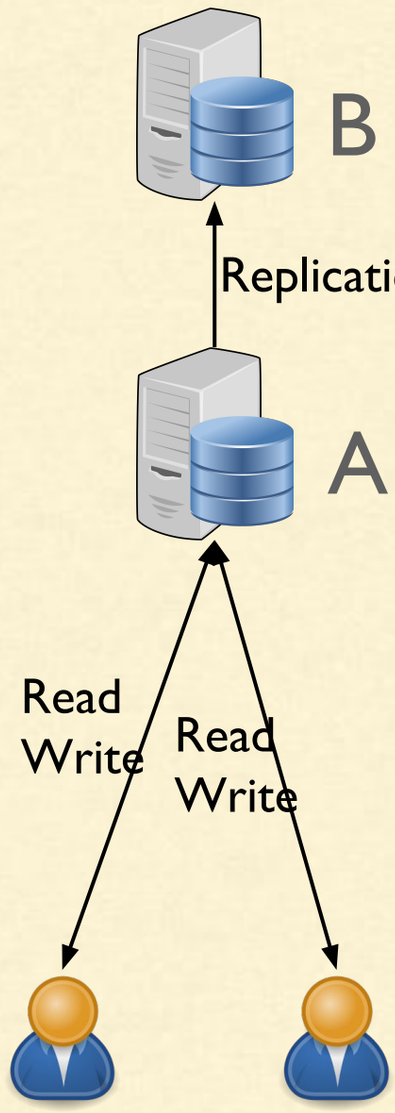

### MongoDB, ZooKeeper

~~**C**onsistent~~ Eventually Consistent
**A**vailable
**P**artition Tolerant




### HBase Master, Namenode with backup, RDBMS with failover

**C**onsistent
~~**A**vailable~~
**P**artition Tolerant



# Serialization

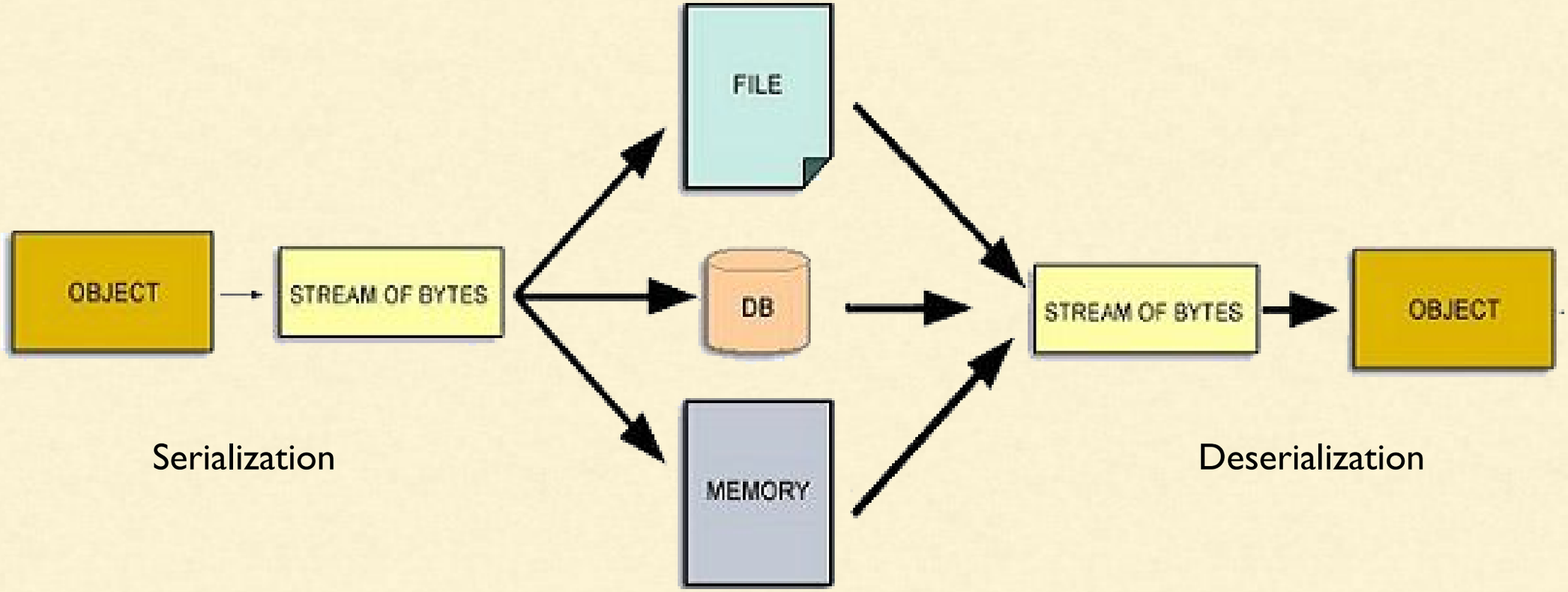
**Process of converting objects into array of bytes**

# Serialization

**Process of converting objects into array of bytes**

**Name:** Sandeep
**Company:** CloudxLab
**Gender:** Male

| { | N | a | m | e | : | S | a | n | d | e | e | p | , | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|



# Column Oriented Database

**Data in columns stored nearby**
**as opposed to the rows being nearby**

| EMPID | NAME |
|---|---|
| 10 | Joe |
| 12 | Mary |
| 11 | Cathy |

10,Joe;12,Mary;11,Cathy;

Row Oriented

10,12,11; Joe,Mary,Cathy;

Column Oriented

# Column Family Oriented DataStore

## Data in columnfamily stored together

| CF1 | | CF2 |
|---|---|---|
| EMPID | NAME | AGE |
| 10 | Joe | 23 |
| 12 | Mary | 33 |
| 11 | Cathy | 45 |

001:10,joe;002:12,Mary;003:11,Cathy;
001:23,002:33,003:45

Column Family cf1:empid, name, cf2:age

# Column Family Oriented DataStore

## Data in columnfamily stored together

| *CF1* | CF2 | CF3 |
|---|---|---|
| *EMPID* | *NAME* | AGE |
| *10* | *Joe* | 23 |
| *12* | *Mary* | 33 |
| *11* | *Cathy* | 45 |

?

| *CF1* | | |
|---|---|---|
| *EMPID* | *NAME* | AGE |
| *10* | *Joe* | 23 |
| *12* | *Mary* | 33 |
| *11* | *Cathy* | 45 |

?

# Column Family Oriented DataStore

## Data in columnfamily stored together

| *CF1* | CF2 | CF3 |
|---|---|---|
| *EMPID* | *NAME* | AGE |
| *10* | *Joe* | 23 |
| *12* | *Mary* | 33 |
| *11* | *Cathy* | 45 |

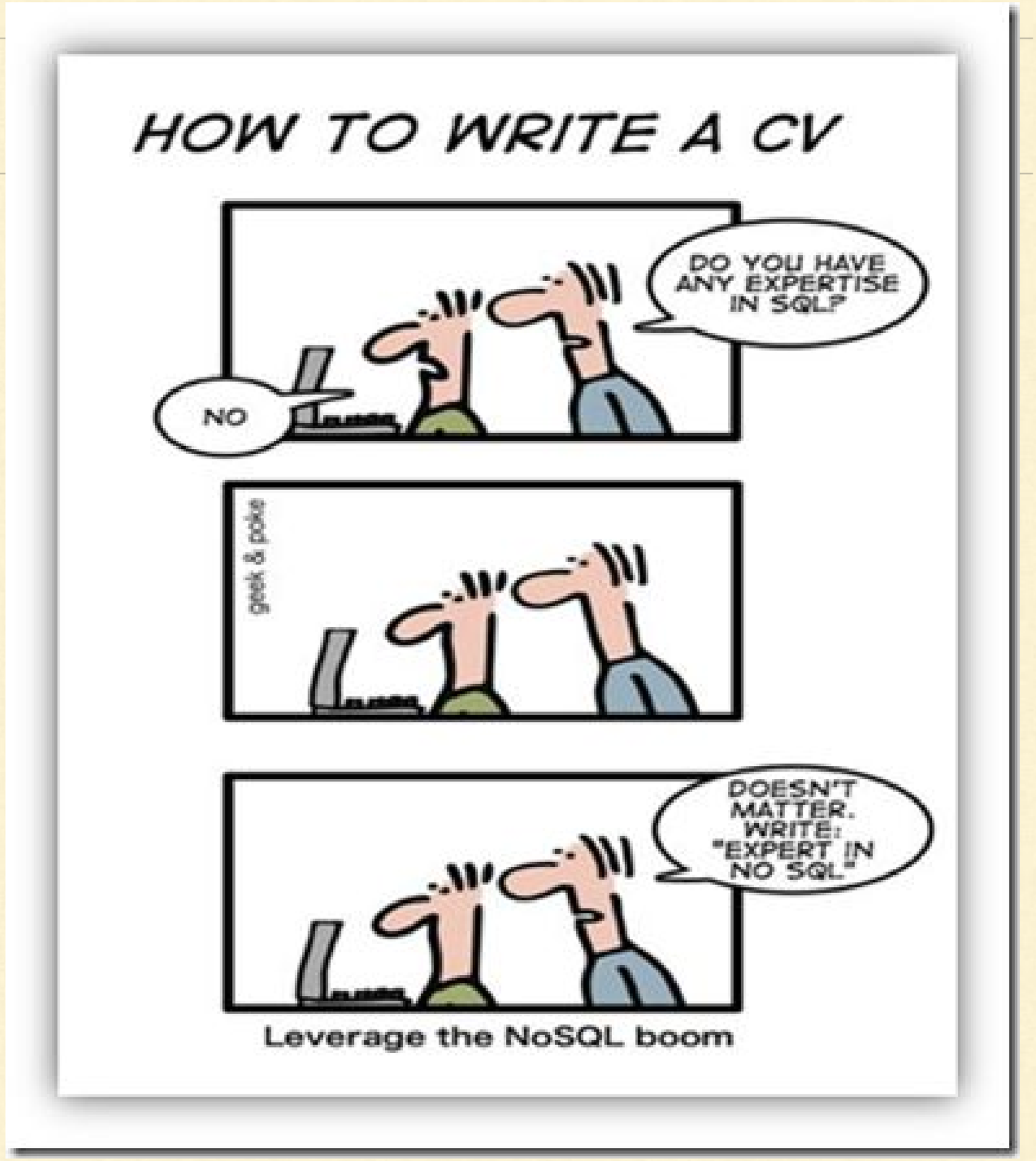
Columnwise

| *CF1* | | |
|---|---|---|
| *EMPID* | *NAME* | AGE |
| *10* | *Joe* | 23 |
| *12* | *Mary* | 33 |
| *11* | *Cathy* | 45 |

Row wise

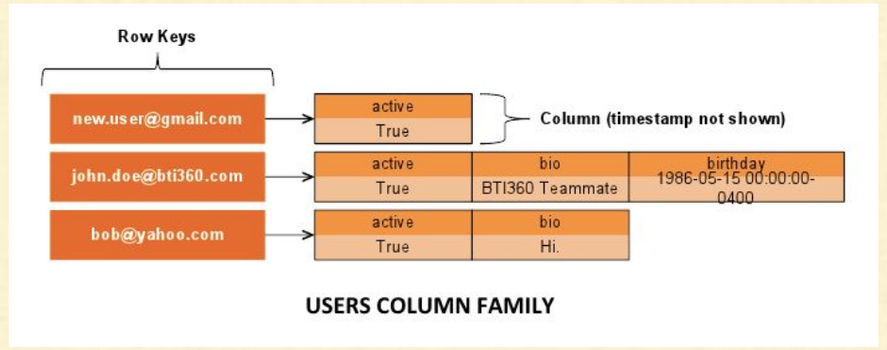

Thank you!

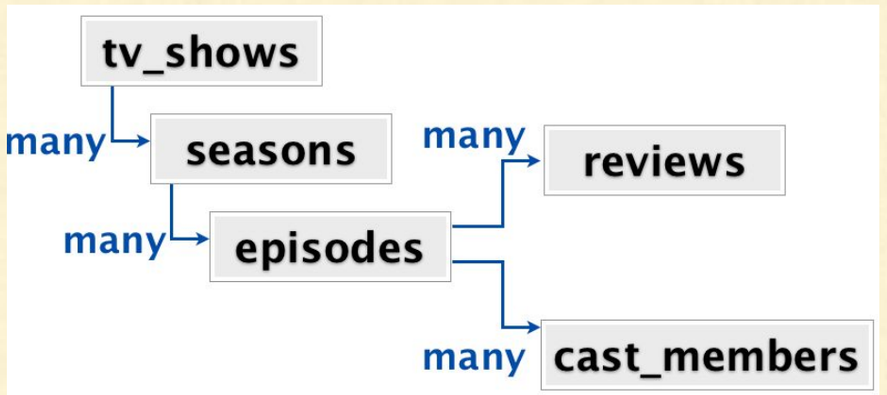
# NoSQL?

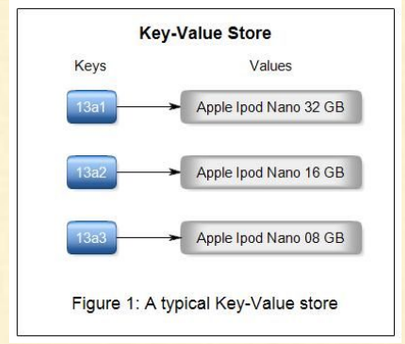

HOW TO WRITE A CV
DO YOU HAVE ANY EXPERTISE IN SQL?
NO
geek & poke
DOESN'T MATTER. WRITE: "EXPERT IN NO SQL"
Leverage the NoSQL boom

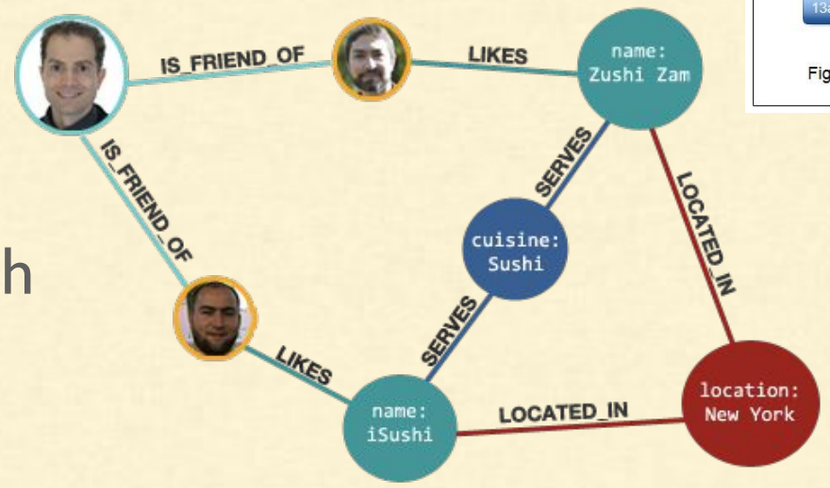
# Types of NoSQL Stores

- **Column Oriented / wide-column**
  - Accumulo, Cassandra, HBase
- **Document**
  - Clusterpoint, Couchbase, MarkLogic, MongoDB
- **Key-value**
  - Dynamo, MemcacheDB, Project Voldemort, Redis, Riak
- **Graph**
  - Allegro, Neo4J, OrientDB, Virtuoso, giraph



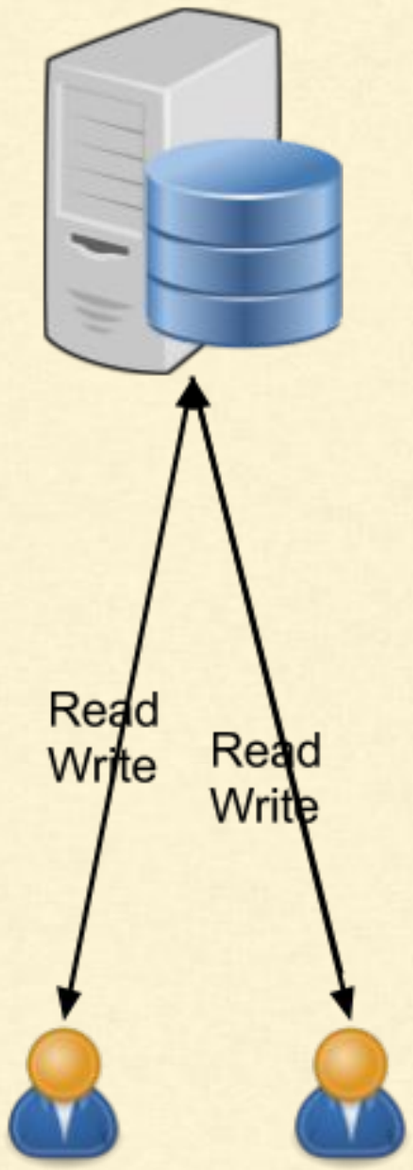
# NoSQL Vs SQL Summary

| | SQL DATABASES | NOSQL DATABASES |
|---|---|---|
| Types | One type - minor variations | Many different types |
| Development History | 1970s | 2000s |
| Examples | MySQL, Postgres, Oracle Database | MongoDB, Cassandra, HBase, Neo4j |
| Data Storage Model | Relational | Varies based on database type. |
| Schemas | Structure and data types are fixed in advance. | Typically dynamic. |
| Scaling | Vertically, | Horizontally, |
| Development Model | Mix | Open-source |
| Supports Transactions | ACID | In certain circumstances and at certain levels (e.g., document level vs. database level) |
| Data Manipulation | SQL | Through object-oriented APIs |
| Consistency | Can be configured for strong consistency | Depends on product. |

# CAP Theorem

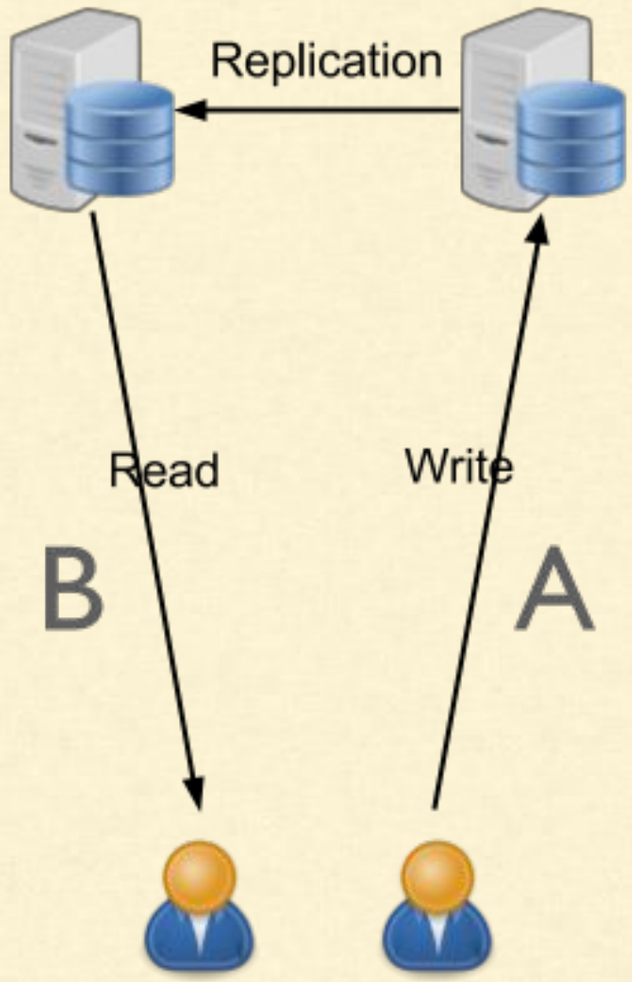
RDBMS

**C**onsistent
**A**vailable
~~**P**artition Tolerant~~



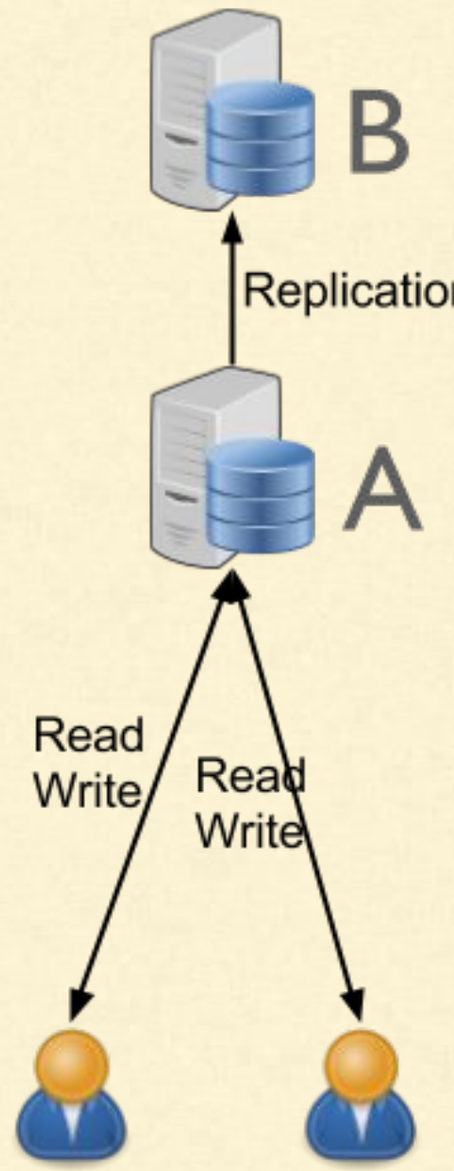

Mongodb, zookeeper

~~**C**onsistent~~ Eventually Consistent
**A**vailable
**P**artition Tolerant




HBase Master, Namenode with backup, RDBMS with failover

**C**onsistent
~~**A**vailable~~
**P**artition Tolerant




*Sandeep Giri*