

NoSQL



School of Computer Science,
UCD

Scoil na Ríomheolaíochta,
UCD

Outline

- Motivation
- CAP Theorem
- Characteristics
- Database Landscape
- Example: MongoDB

Take home message:

RDBMS do not scale well -> NoSQL which play around with the Consistency/Availability dilemma



MOTIVATIONS



A Long Time ago...

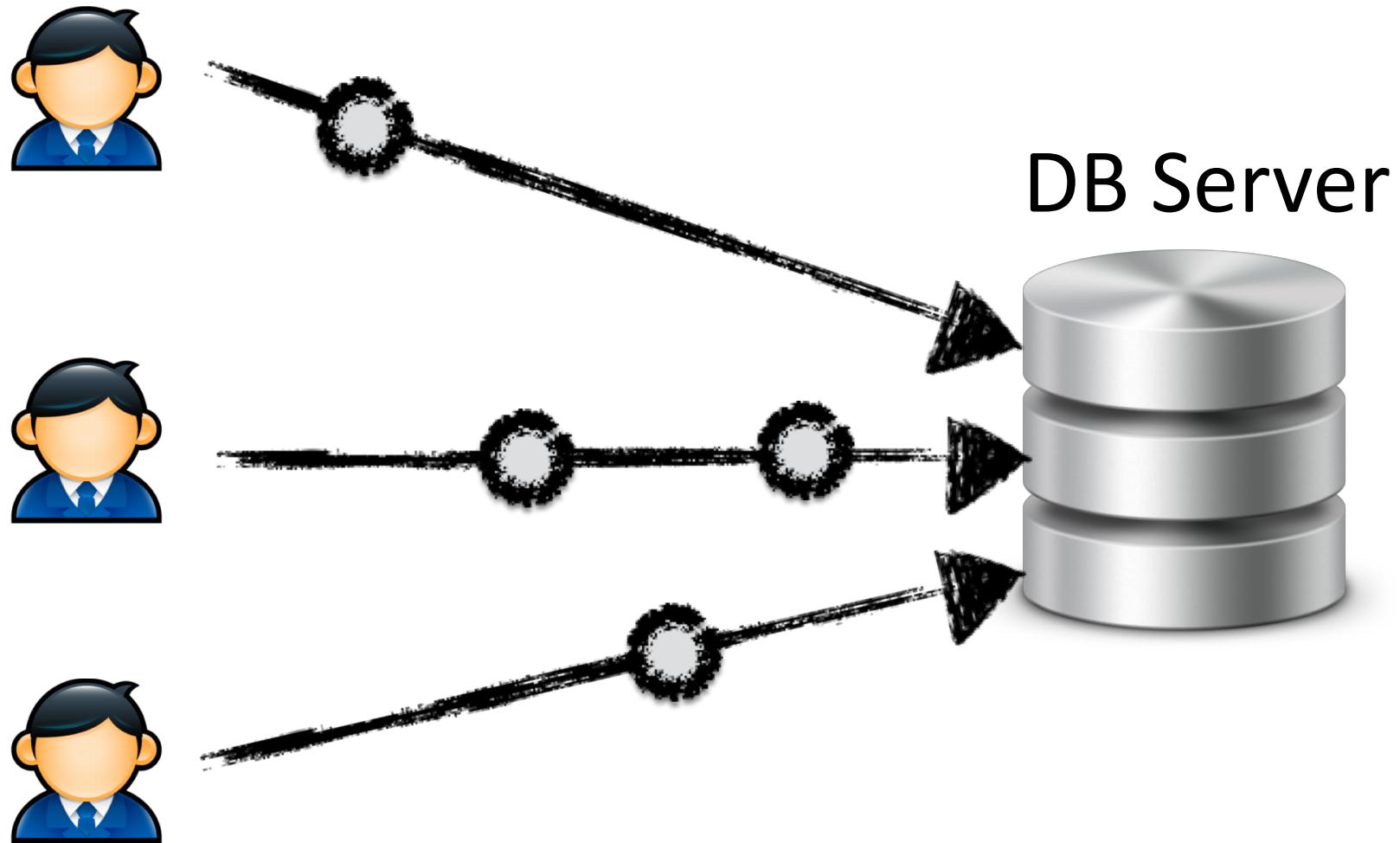
- Everybody was happy with RDBMS
 - Centralised
 - Easy maintenance
 - Available anytime
 - Stable performance
 - Reliable (ACID)



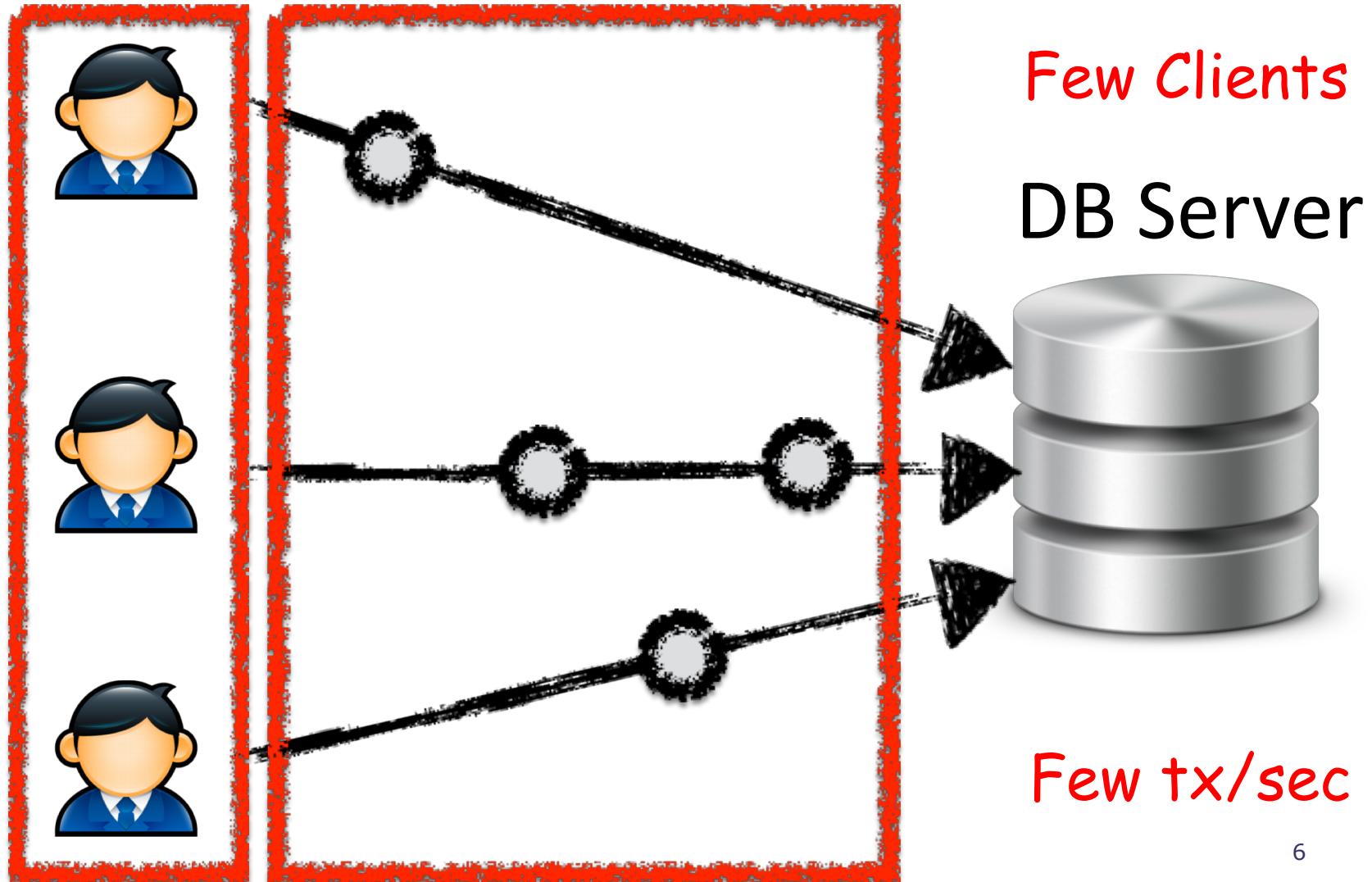
ORACLE



Traditional RDBMS



Traditional RDBMS



RDBMS meet Big Data

- ***Large number of sources***: users, devices, sensors
- ***Swarm of transactions***: e.g. 1,000tx/sec/source
- ***Huge data size***: e.g. 10MB/sec stream



RDBMS are not Agile

- Facebook added a new feature: replying with a photo
(June 2013)

A screenshot of a Facebook post interface. The main post is from Aaditya Ailawadhi, asking if there are any plans for Gurgaon-like ambience or something of the sort in Delhi. Below the post are several comments from users Abhishek Bhatnagar and Aaditya Ailawadhi. A comment from Abhishek Bhatnagar includes a reply with a photo, which shows a green and yellow object, possibly a toy or a piece of equipment.

Aaditya Ailawadhi: sweet, and do they have any plans for gurgaon like ambience or something of the sort or any delhi?

Like · Comment · Unfollow Post · Share · Promote

Abhishek Bhatnagar: so will they have like actual google products, like the nexus lineup, chromebook pixel etc. on display and for sale?
12 minutes ago

Abhishek Bhatnagar: Aaditya Ailawadhi - this year end in few months you might see them.
12 minutes ago · Like

Aaditya Ailawadhi: sweet, and do they have any plans for gurgaon like ambience or something of the sort or any delhi?
6 minutes ago · Like

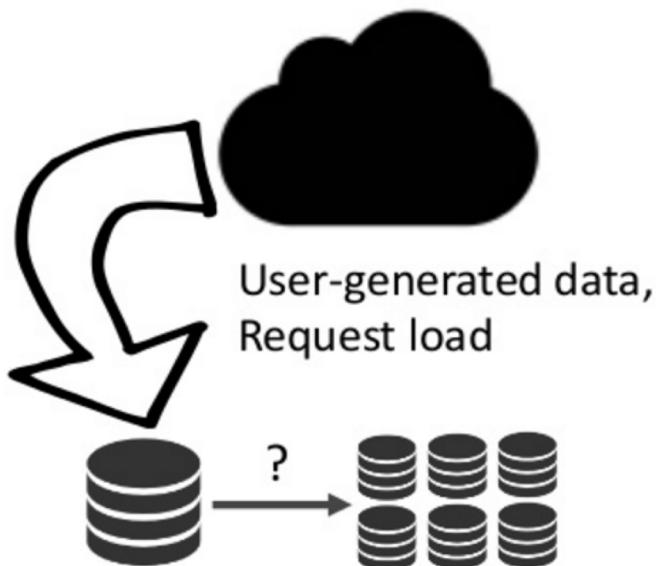
Abhishek Bhatnagar: Aaditya Ailawadhi - i have no idea as if now about their plans 😊
5 minutes ago · Like

Abhishek Bhatnagar: Test photo upload using new facebook photo comment reply feature

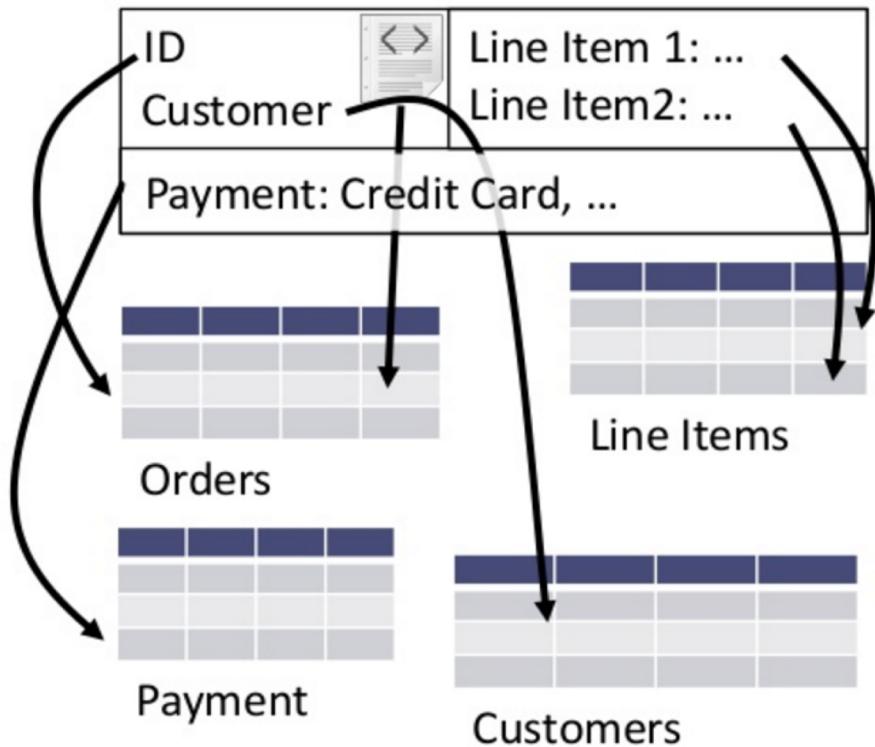


Motivation for NoSQL

Scalability



Impedance Mismatch



CHARACTERISTICS



Key Features of NoSQL

- Term coined in 2009 (Not Only SQL)



- Scalable (e.g. horizontal scaling)
- Easy replication through the network
- “Simple” API language (NO query language)
- Tunable consistency model (NO ACID properties)
- Ability to add new attributes to record (NO schema)



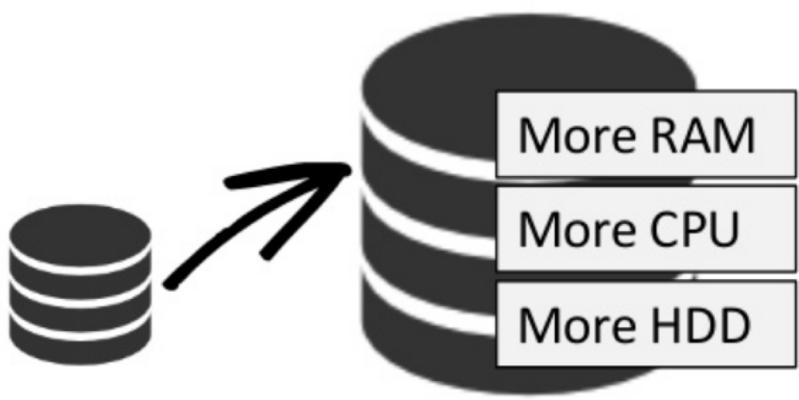
(Old) SQL vs NewSQL vs NoSQL

	OldSQL	NoSQL
ACID/Transaction	✓	✗
Language/Algebra	✓	✗
Schema/Constraints	✓	✗
Scalability	✗	✓ ✓ ✓
In-memory Tx	✗	✓
Easy replication	✗	✓

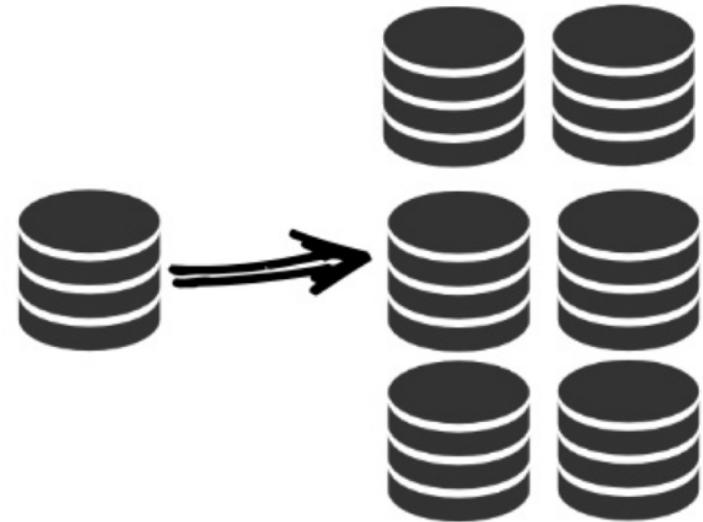


Vertical vs Horizontal Scaling

Scale-Up (*vertical* scaling):



Scale-Out (*horizontal* scaling):

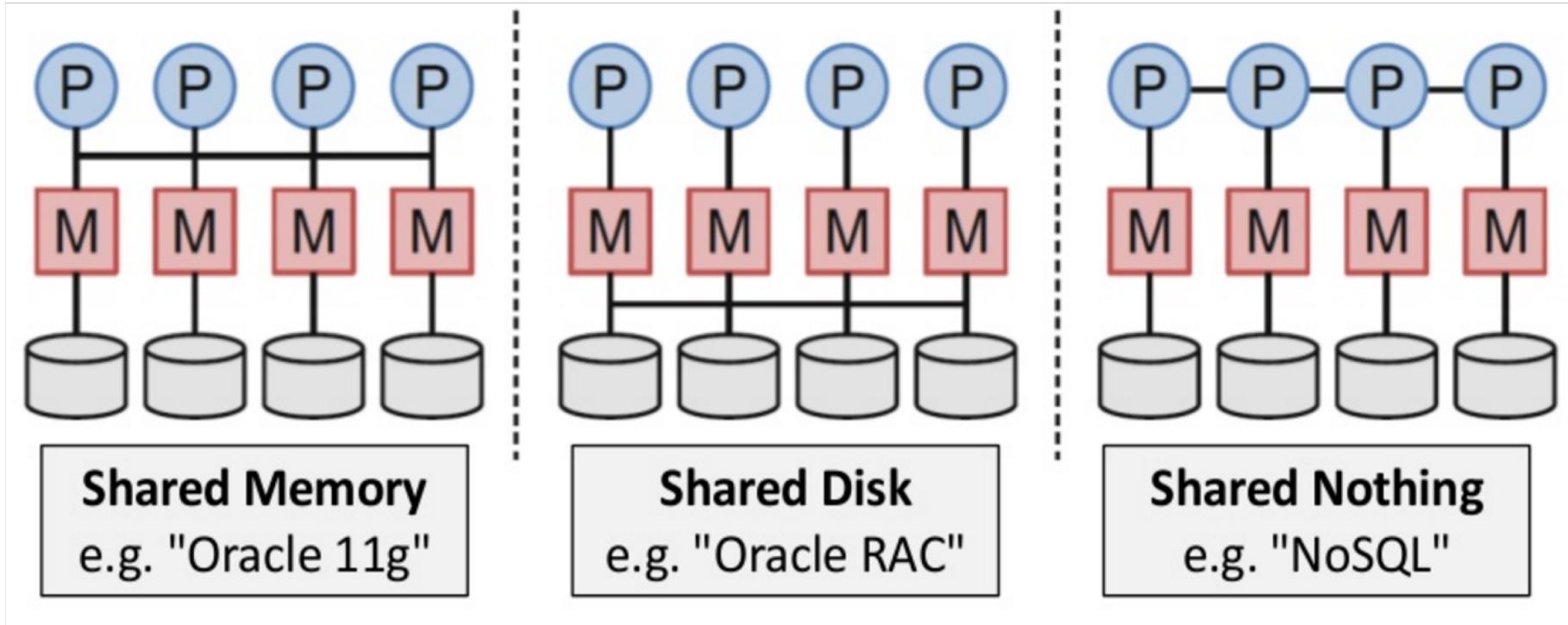


Commodity
Hardware

Shared-Nothing
Architecture



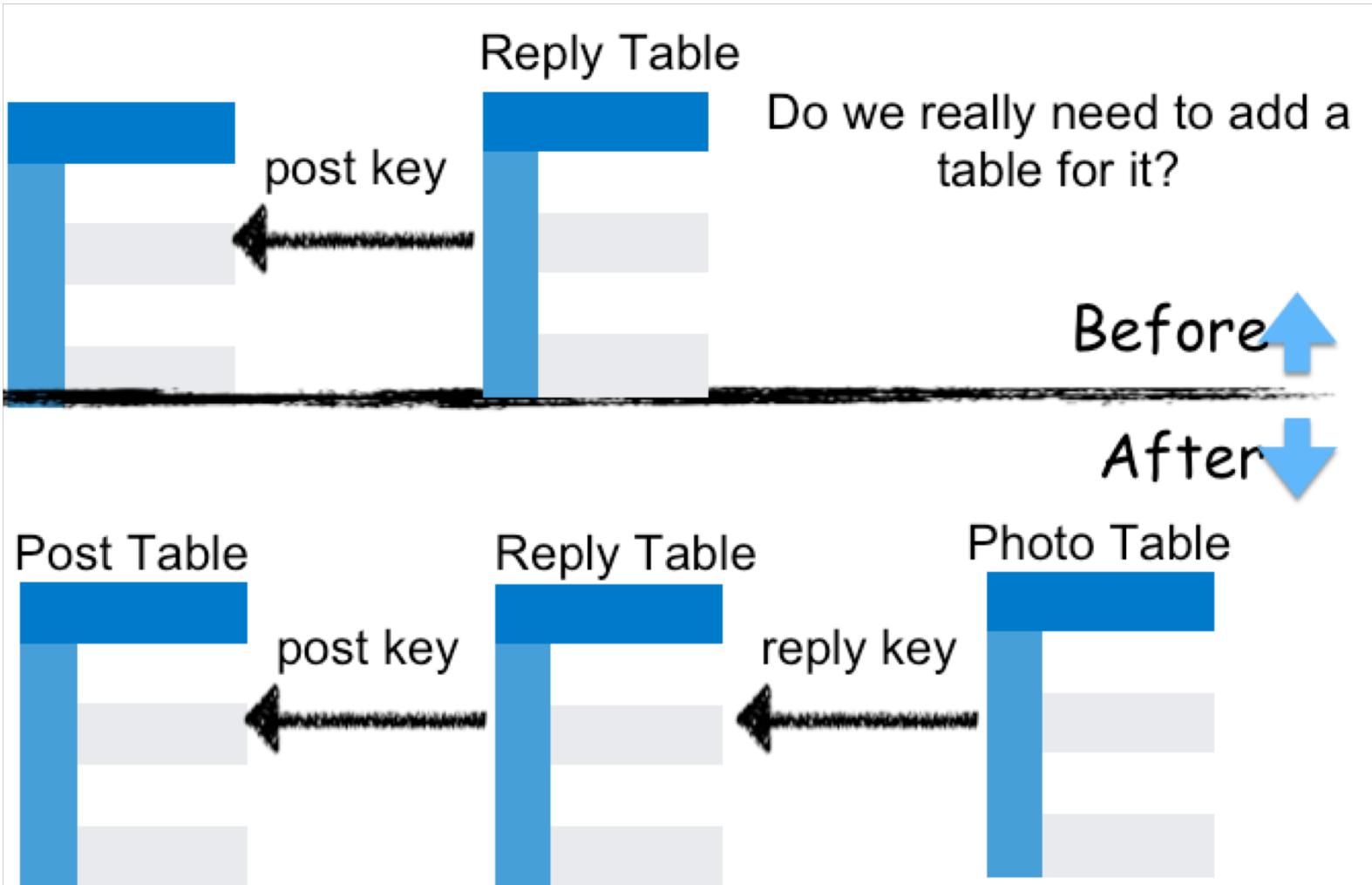
Shift Towards “Shared Nothing”



- When each node is independent and self-sufficient, there is no single point of contention across the system
- the potential for scaling is huge !

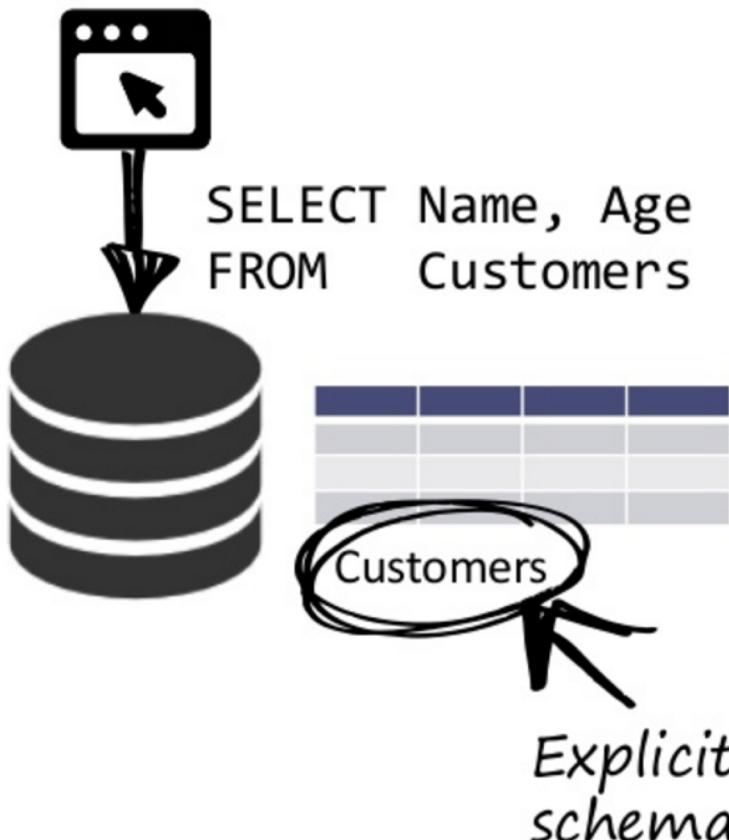


Schema-dependent

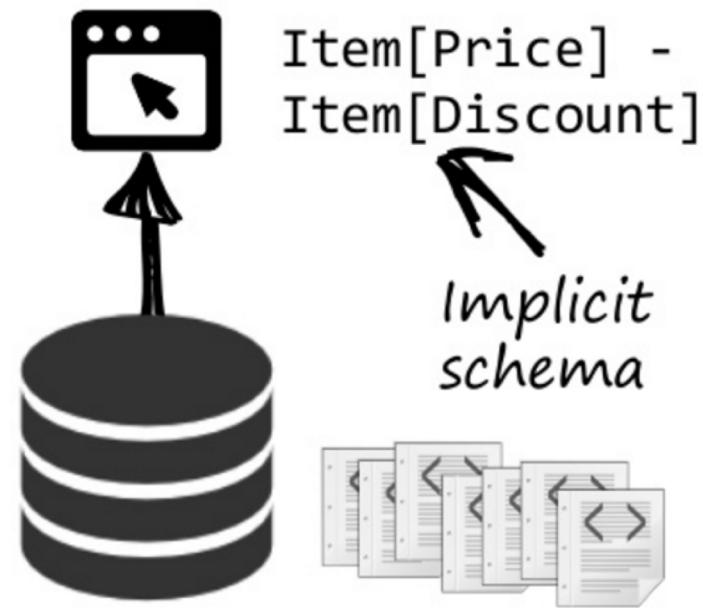


Schema-free

RDBMS:



NoSQL DB:



Benefits of Schema-free

- Flexibility
- Easy to replicate
- Easy to partition
- Join-free



CAP THEOREM



Visual Guide to NoSQL Systems

Availability:
Each client can
always read
and write.

Data Models

Relational (comparison)
Key-Value
Column-Oriented/Tabular
Document-Oriented

A

CA

RDBMSs
(MySQL,
Postgres,
etc)

Aster Data
Greenplum
Vertica

AP

Dynamo
Voldemort
Tokyo Cabinet
KAI

Cassandra
SimpleDB
CouchDB
Riak

Pick Two

C

Consistency:
All clients always
have the same view
of the data.

CP

BigTable
Hypertable
Hbase

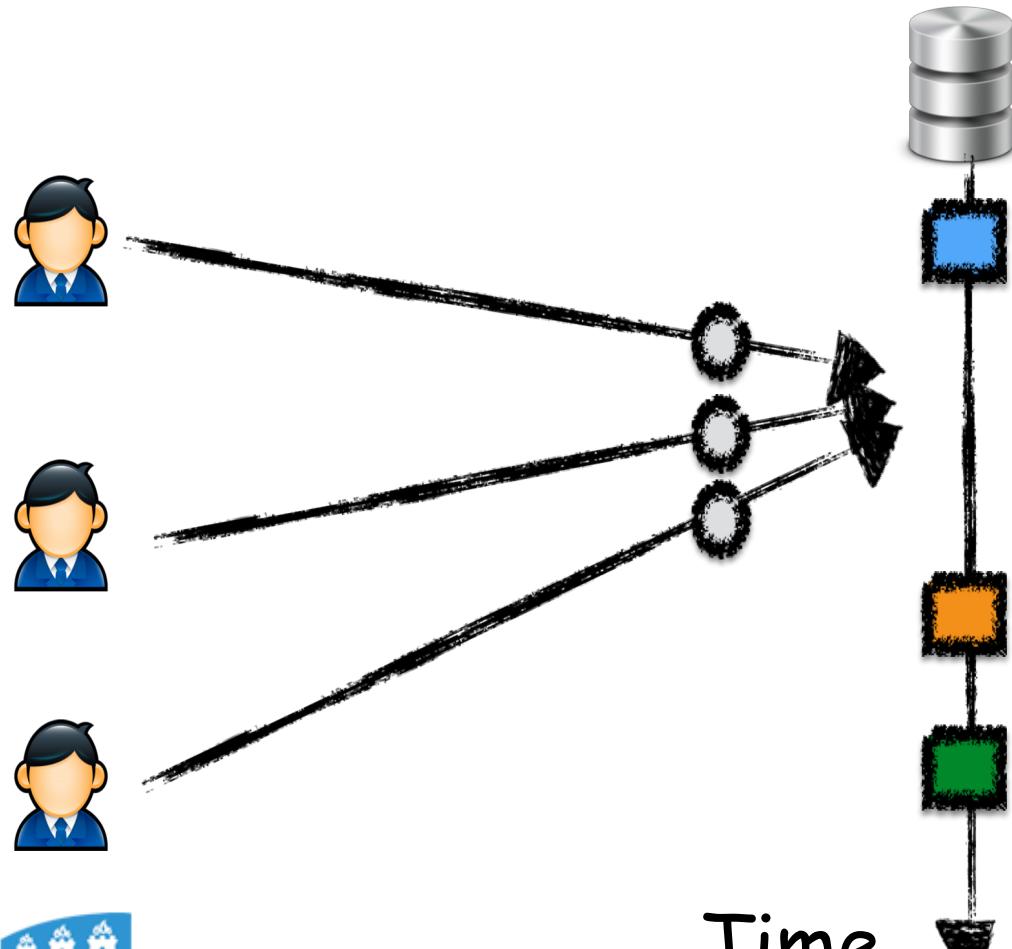
MongoDB
Terrastore
Scalarmis

Berkeley DB
MemcacheDB
Redis

P

Partition Tolerance:
The system works
well despite physical
network partitions.

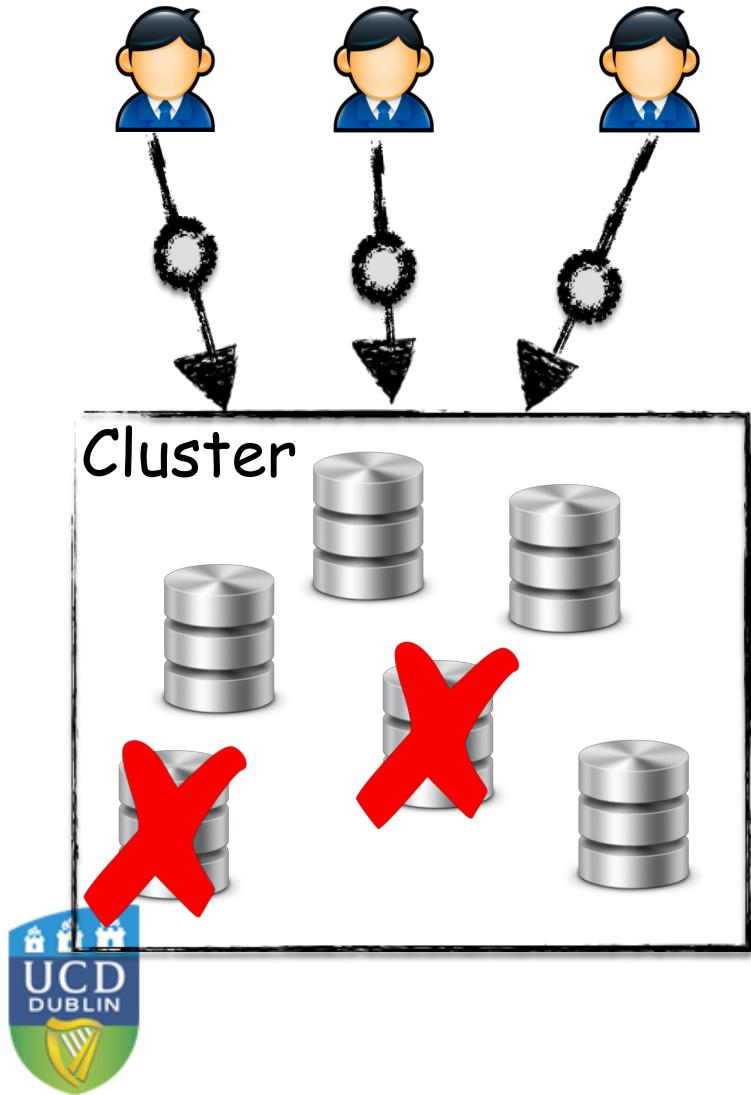
Consistency



All clients have the same view on the data

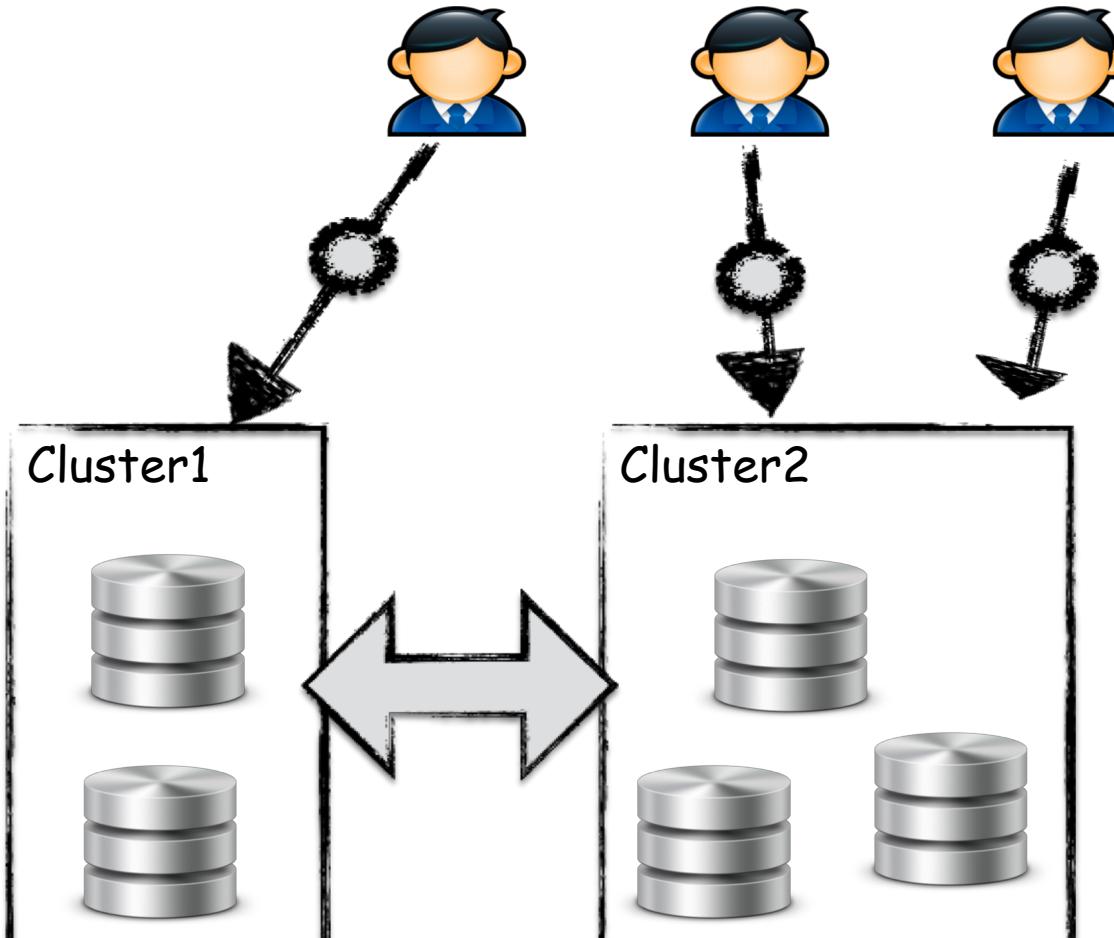


Availability



Every request to a non-failed node must result in a correct response

Partition Tolerance

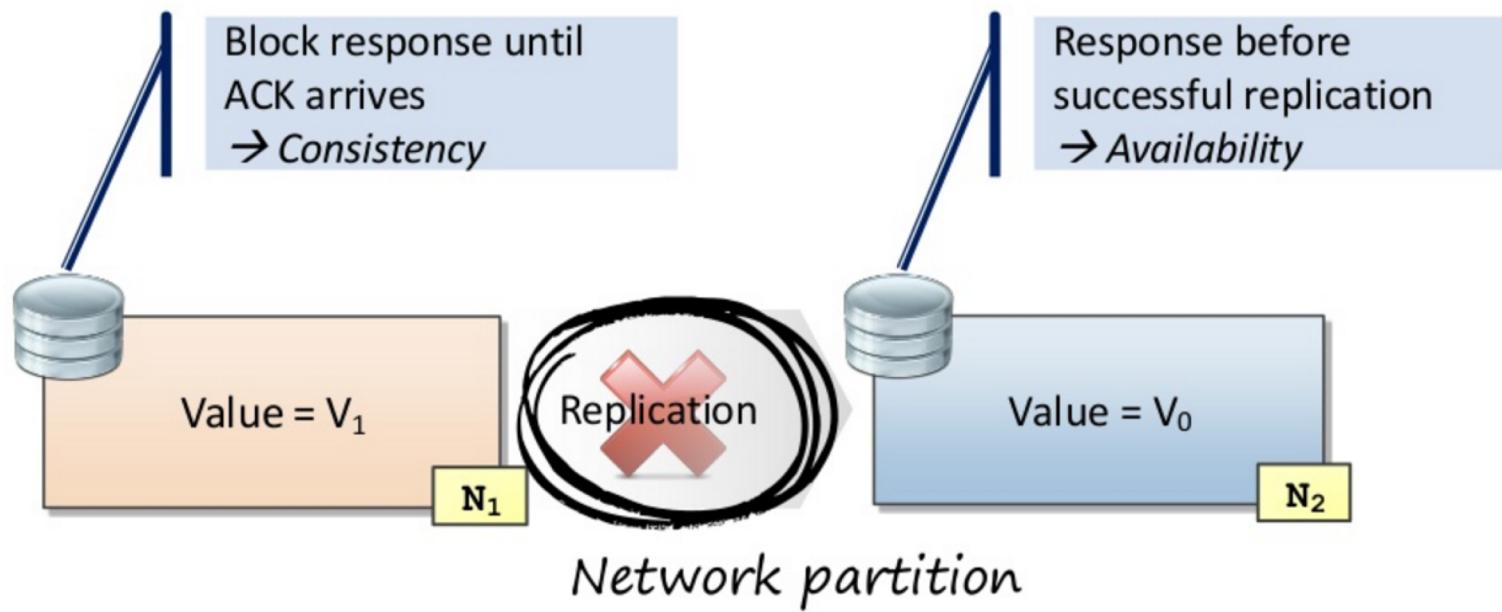


The system has to continue working, even under arbitrary network partitions



Ideas

- When a network partition occurs, either consistency or availability is not ensured



DATABASE LANDSCAPE

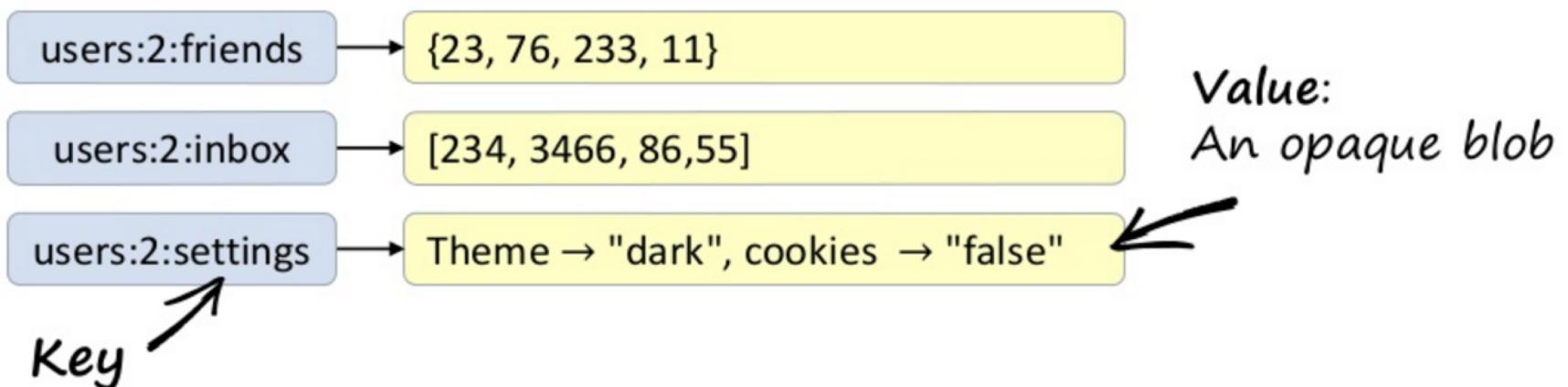


NoSQL Landscape



Key-Value

- **single (key) -> value, no foreign key/value indices** appropriate for instant access to data (shop basket)

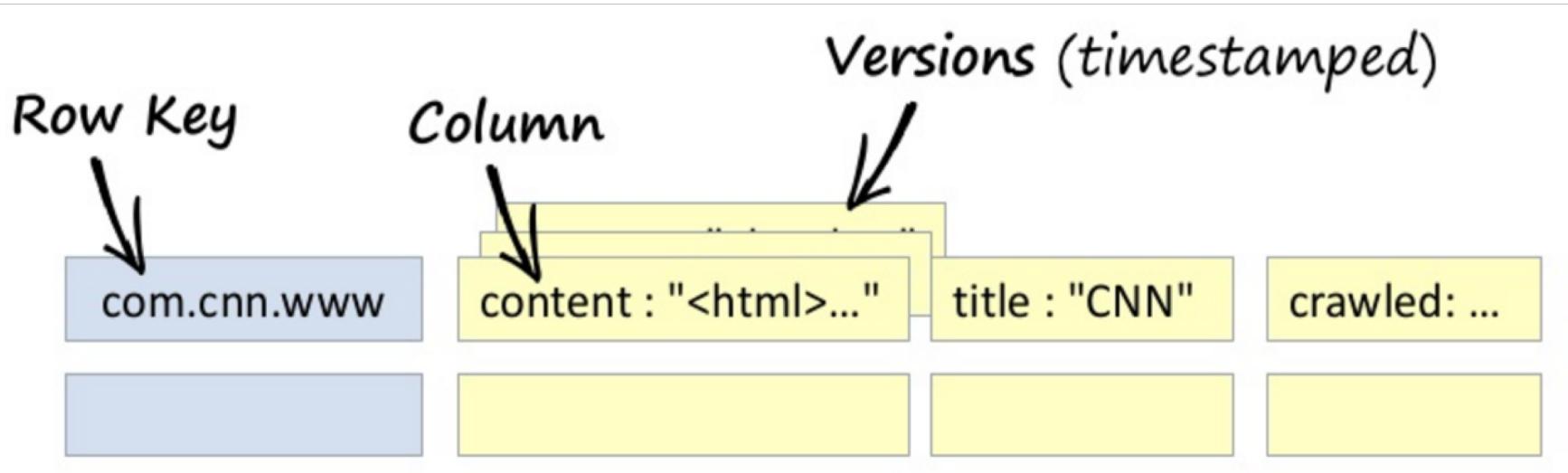


- Example: Dynamo (AP), Riak (AP), Redis (CP)



Wide-column

- ***data indexed by a 3D map: (row, col, time) -> val***
appropriate for aggregation and analytics (reports)



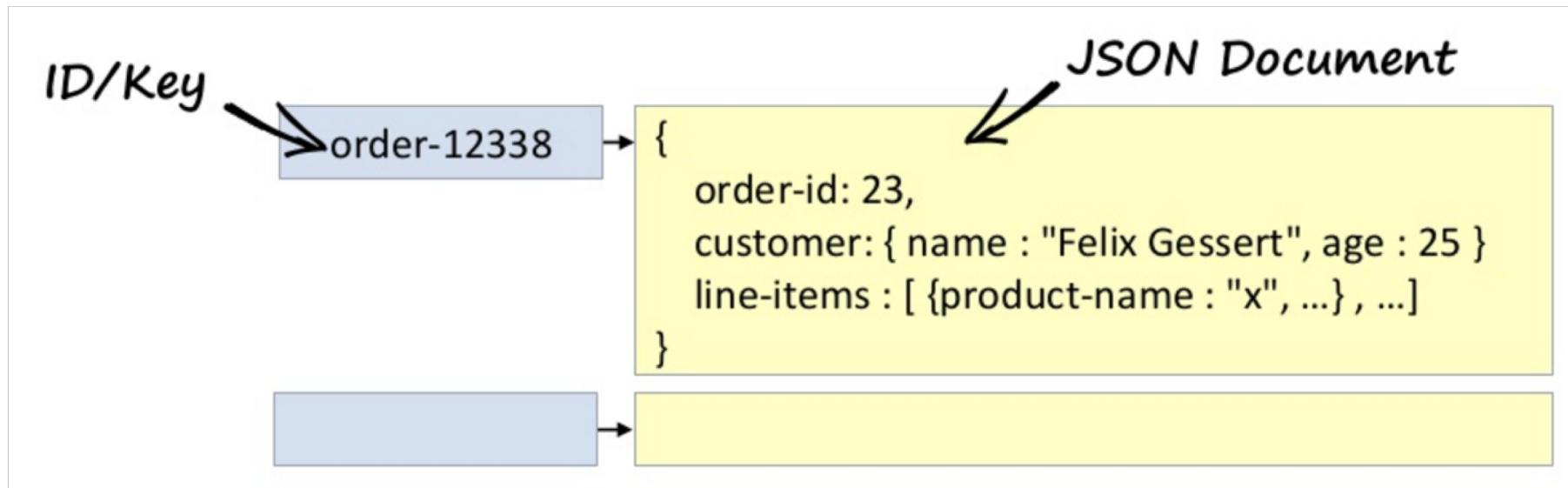
- Example: Cassandra (AP), BigTable (CP), HBase (CP)



Document

- ***similar to JSON/XML: (coll, key) -> doc***

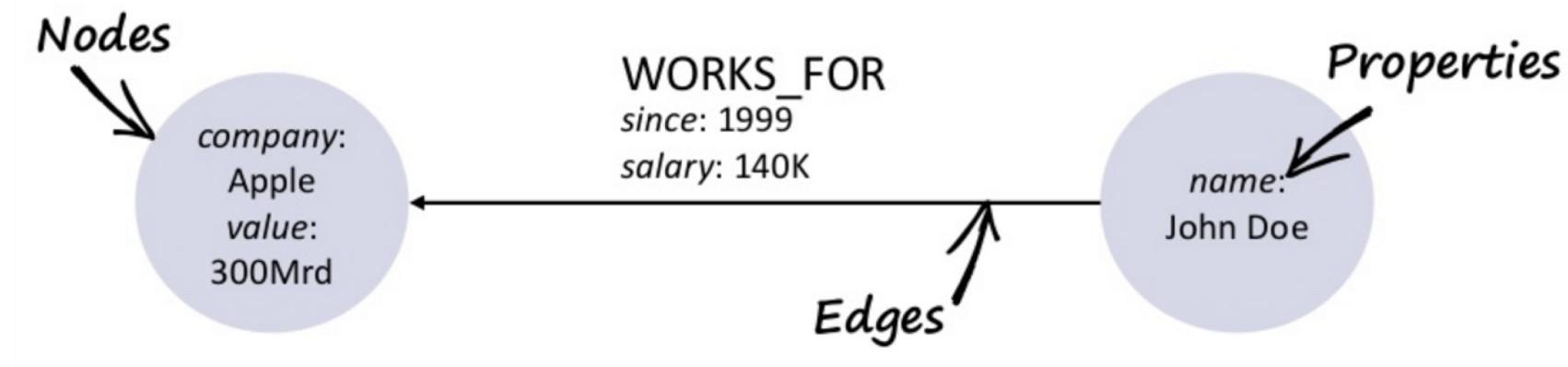
appropriate for data not frequently changing (logs)



- Example: CouchDB (AP), SimpleDB (AP), MongoDB (CP)

Graph

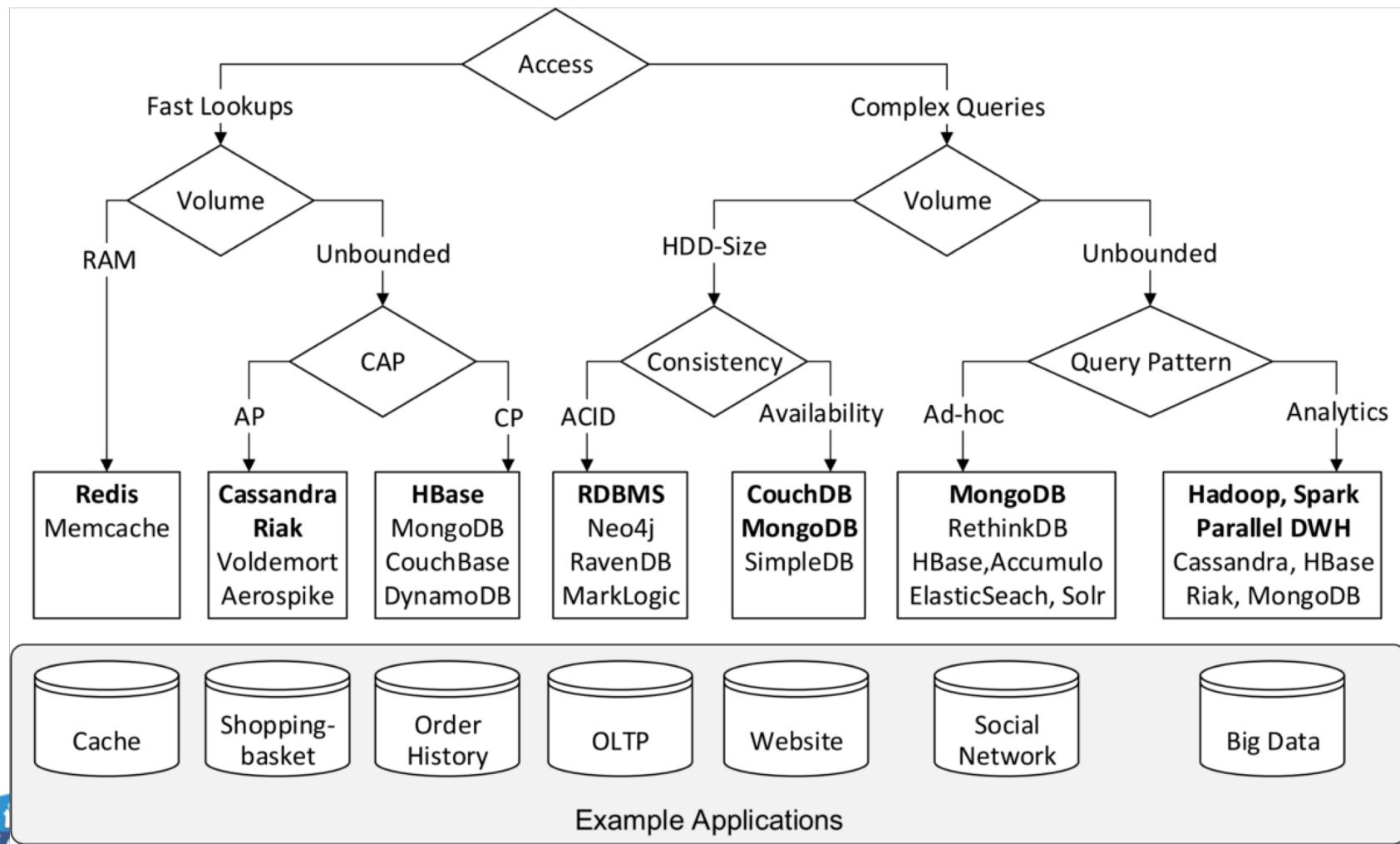
- ***data represented by vertices/edges: $G = (V, E)$*** appropriate for complex structures (social network)



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



Decision Tree



Source: <https://medium.baqend.com/nosql-databases-a-survey-and-decision-guidance-ea7823a822d>



Database Ranking

341 systems in ranking, January 2018

Rank			DBMS	Database Model	Score		
Jan 2018	Dec 2017	Jan 2017			Jan 2018	Dec 2017	Jan 2017
1.	1.	1.	Oracle 	Relational DBMS	1341.94	+0.40	-74.78
2.	2.	2.	MySQL 	Relational DBMS	1299.71	-18.36	-66.58
3.	3.	3.	Microsoft SQL Server 	Relational DBMS	1148.07	-24.42	-72.89
4.	4.	↑ 5.	PostgreSQL 	Relational DBMS	386.18	+0.75	+55.81
5.	5.	↓ 4.	MongoDB 	Document store	330.95	+0.18	-0.96
6.	6.	6.	DB2 	Relational DBMS	190.28	+0.70	+7.78
7.	7.	↑ 8.	Microsoft Access	Relational DBMS	126.70	+0.82	-0.75
8.	↑ 9.	↓ 7.	Cassandra 	Wide column store	123.88	+0.67	-12.57
9.	↓ 8.	9.	Redis 	Key-value store	123.14	-0.10	+4.44
10.	10.	↑ 11.	Elasticsearch 	Search engine	122.55	+2.77	+16.38
11.	11.	↓ 10.	SQLite 	Relational DBMS	114.25	-0.94	+1.88
12.	12.	12.	Teradata	Relational DBMS	72.63	-2.11	-1.54
13.	↑ 14.	13.	SAP Adaptive Server 	Relational DBMS	65.46	-0.22	-3.64
14.	↓ 13.	14.	Solr	Search engine	64.37	-1.93	-3.71
15.	15.	↑ 16.	Splunk	Search engine	64.00	+0.21	+8.51
16.	16.	↓ 15.	HBase	Wide column store	61.64	-1.78	+2.50
17.	17.	↑ 20.	MariaDB 	Relational DBMS	58.30	+1.56	+13.26
18.	↑ 19.	↑ 19.	Hive 	Relational DBMS	55.49	+0.81	+4.35
19.	↓ 18.	↓ 17.	FileMaker	Relational DBMS	55.20	+0.00	+1.72
20.	20.	↓ 18.	SAP HANA 	Relational DBMS	46.16	-0.33	-5.77
21.	21.	21.	Neo4j 	Graph DBMS	39.96	+1.24	+3.70



<https://db-engines.com/en/ranking>

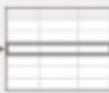
Summary



Data Model



Key-Value



Wide-Column



Document



Graph



Consistency/Availability Trade-Off



AP: Available & Partition Tolerant

CP: Consistent & Partition Tolerant

CA: Not Partition Tolerant



The era of one-size-fits-all DBMS is over

EXAMPLE: MONGODB



MongoDB in a Nutshell

- Created by MongoDB Inc. in Oct. 2007
 - MongoDB <= huMONGous DB
- Document-oriented data model
 - BSON (similar to JSON)
- Supports:
 - indexing
 - replication
 - load balancing
 - server-side JavaScript execution.



Insert/Update/Delete

- Insert: db.<collection>.insert(<document>)
 - db.users.insert({name:"Aoife",city:"Dublin"})
 - no need to create a table upfront!
- Update: db.<collection>.update(<query>, <values>)
 - db.users.update({name:"Niamh"}, {\$set: {city:"Cork"})
- Delete: db.<collection>.remove(<query>)
 - db.users.remove({name:"Deirdre"})



Select/Queries

- Simple Select: db.<collection>.find(<query>)
 - db.users.find({name:"Fionna"})
- Nested Select: use dot-notation
 - db.users.find({"score.math":30})
- Show specific columns only
 - db.users.find({}, {"score.math":1, "score.lang":1})
- Query if
 - db.users.find({"score.math":{\$exists:true}}, {name:1})



Other Useful Operations

- `count()`: counting the number of documents
 - `db.users.find().count()`
 - `db.users.count({name:"john"})`
- `sort()`: sorting documents
 - `db.users.find().sort({name:-1})`
- List collections (tables) and databases
 - `show collections/tables ; show databases`
- `limit()`: Show a specified number of documents
 - `db.users.find().limit(5)`

