POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Week 6-1

NoSQL

Big Data

Prof. Hwanjo Yu
POSTECH

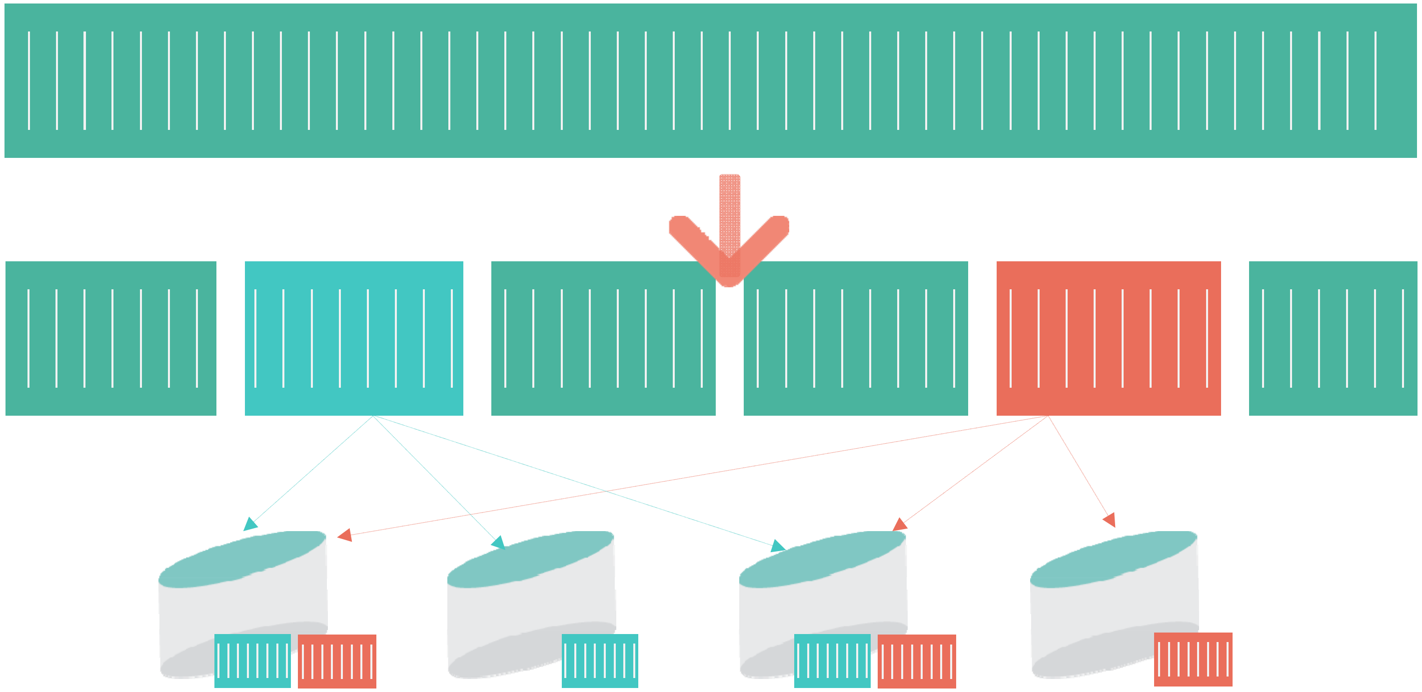
NoSQL and related systems, by feature

Year	System/ Paper	Scale to 1000s	Primary Index	Secondary Indexes	Transactions	Joins/ Analytics	Integrity Constraints	Views	Language/ Algebra	Data model	my label
1971	RDBMS	O	✔	✔	✔	✔	✔	✔	✔	tables	sql-like
2003	memcached	✓	✓	O	O	O	O	O	O	key-val	nosql
2004	MapReduce	✓	O	O	O	✓	O	O	O	key-val	batch
2005	CouchDB	✓	✓	✓	record	MR	O	✓	O	document	nosql
2006	BigTable/Hbase	✓	✓	✓	record	compat.w/MR	/	O	O	ext. record	nosql
2007	MongoDB	✓	✓	✓	EC, record	O	O	O	O	document	nosql
2007	Dynamo	✓	✓	O	O	O	O	O	O	ext. record	nosql
2008	Pig	✓	O	[illegible]	[illegible]	✓	✓	[illegible]	✓	tables	sql-like
2008	HIVE	✓	O	O	O	✓	✓	O	✓	tables	sql-like
2008	Cassandra	✓	✓	✓	EC, record	O	✓	✓	O	key-val	nosql
2009	Voldemort	✓	✓	O	EC, record	O	O	O	O	key-val	nosql
2009	Riak	✓	✓	✓	EC, record	MR	O			key-val	nosql
2009	Redis	✓	✓	✓	group	O	O	O	✓	key-val	nosql
2010	Dremel	✓	O	O	O	/	✓	O	✓	Tables	sql-like
2011	Megastore	✓	✓	✓	entity groups	O	/	O	/	Tables	nosql
2011	Tenzing	✓	O	O	O	O	✓	✓	✓	Tables	sql-like
2011	Spark/Shark	✓	O	O	O	✓	✓	O	✓	Tables	sql-like
2012	Spanner	✓	✓	✓	✓	?	✓	✓	✓	Tables	sql-like
2013	Impala	✓	O	O	O	✓	✓	O	✓	Tables	sql-like
2014	MS Cosmos	✓	✓	O	EC	O	O	O	✓	document	nosql

Scale was the primary motivation!

Extended from Bill Howe's Data Science course materials (2013)

NoSQL and related systems, by feature



1) We need to ensure high availability 2) We also want to support updates

Example

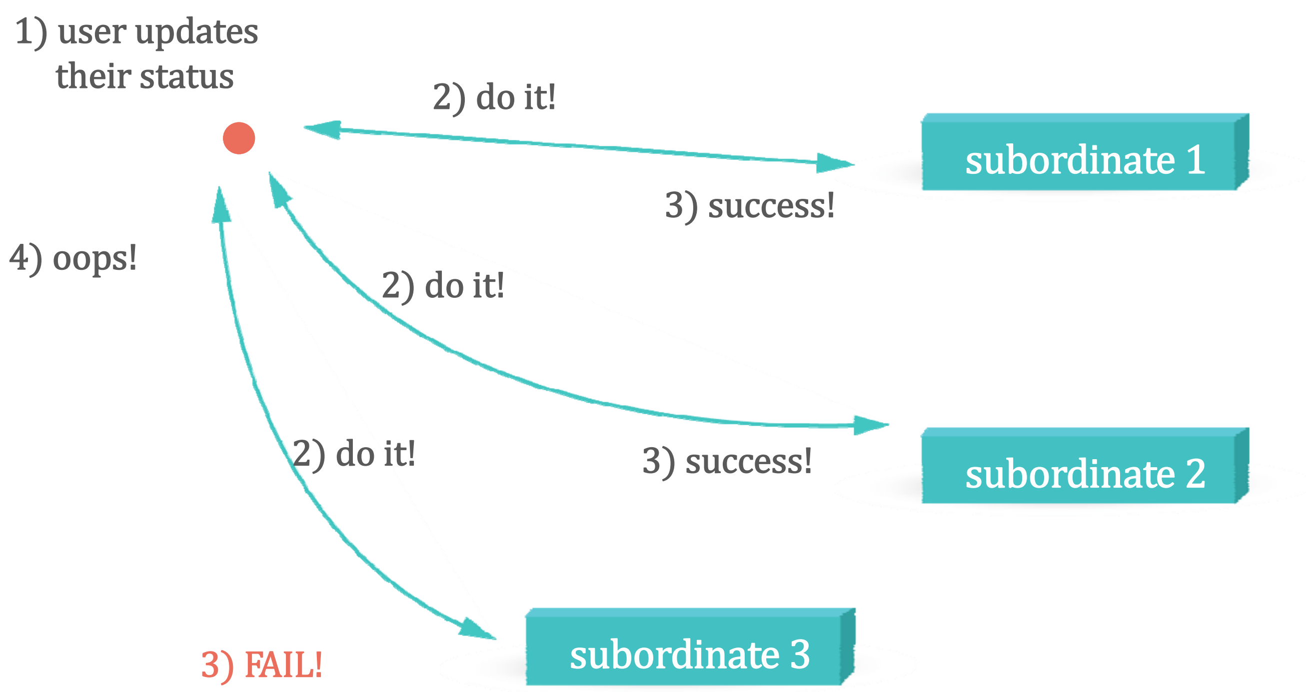
User: Sue
Friends: Joe, Kai, …
Status: "Headed to new Bond flick"
Wall: "…", "…"

User: Joe
Friends: Sue, …
Status: "I'm sleepy"
Wall: "…", "…"

User: Kai
Friends: Sue, …
Status: "Done for tonight"
Wall: "…", "…"

Write	Update Sue's status. Who sees the new status, and who sees the old one?
Databases	"Everyone MUST see the same thing, either old or new, no matter how long it takes."
NoSQL	"For large applications, we can't afford to wait that long, and maybe it doesn't matter anyway"

Two-phase commit motivation



Two-phase commit

Phase 1

- Coordinator sends “prepare to commit”
- Subordinates make sure they can do so no matter what
- Write the action to a log to tolerate failure
- Subordinates reply “ready to commit”

Phase 2

- If all subordinates ready, send “commit”
- If anyone failed, send “abort”

Two-phase commit

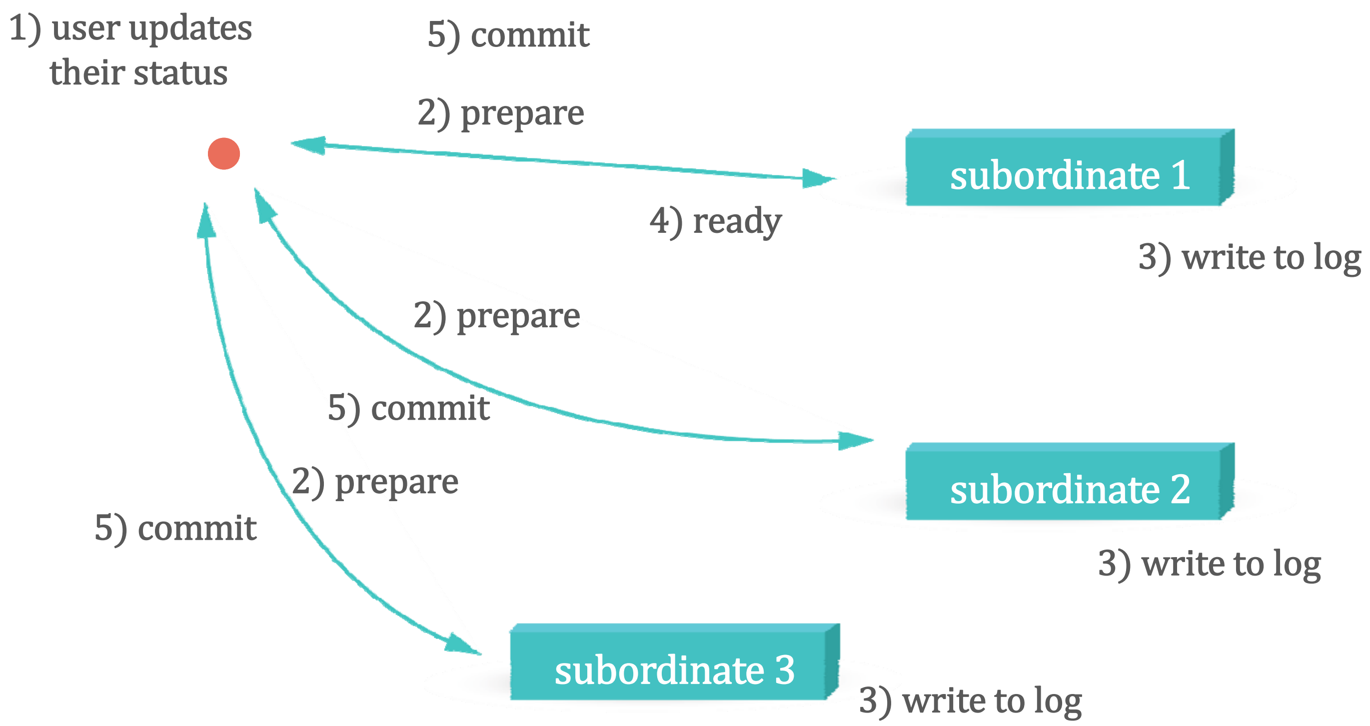
Phase 1

- Coordinator sends “prepare to commit”
- Subordinates make sure they can do so no matter what
- Write the action to a log to tolerate failure
- Subordinates reply “ready to commit”

Phase 2

- If all subordinates ready, send “commit”
- If anyone failed, send “abort”

Two-phase commit



Eventual consistency

- In absence of updates, all replicas converge towards identical copies
- What the application sees in the meantime is sensitive to replication mechanics and difficult to predict

Eventual consistency

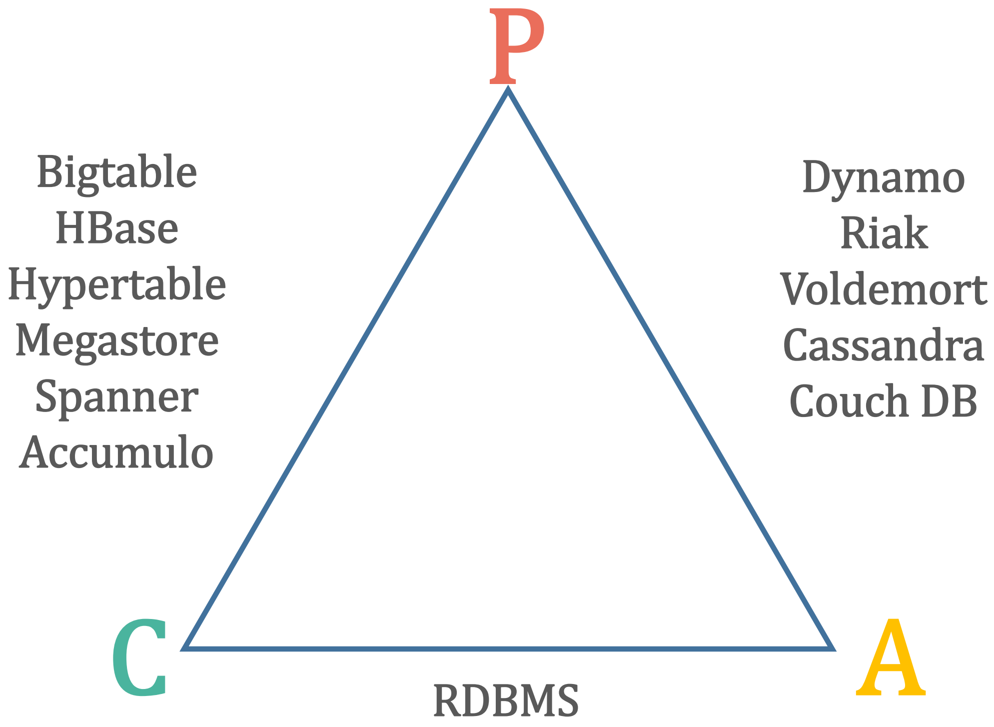
Year	System/ Paper	Scale to 1000s	Primary Index	Secondary Indexes	Transactions	Joins/ Analytics	Integrity Constraints	Views	Language/ Algebra	Data model	my label
2003	memcached	✔	✔	O	O	O	O	O	O	key-val	nosql
2005	CouchDB	✔	✔	✔	record	MR	O	✔	O	document	nosql
2006	BigTable (Hbase)	✔	✔	✔	record	compat. w/MR	/	O	O	ext. record	nosql
2007	MongoDB	✔	✔	✔	EC, record	O	O	O	O	document	nosql
2007	Dynamo	✔	✔	O	O	O	O	O	O	key-val	nosql
2008	Cassandra	✔	✔	✔	EC, record	O	✔	✔	O	key-val	nosql
2009	Voldemort	✔	✔	O	EC, record	O	O	O	O	key-val	nosql
2009	Riak	✔	✔	✔	EC, record	MR	O			key-val	nosql
2009	Redis	✔	✔	✔	group	O	O	O	✔	Key-val	nosql
2011	Megastore	✔	✔	✔	entity groups	O	/	O	/	tables	nosql
2012	Accumulo	✔	✔	✔	record	compat. w/MR	/	O	O	ext. record	nosql
2012	Spanner	✔	✔	✔	✔	?	✔	✔	✔	tables	sql-like
2014	MS Cosmos	✔	✔	O	EC	O	O	O	✔	document	nosql

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

CAP theorem [Brewer 2000, Lynch 2002]

- **Consistency**
  - Do all applications see all the same data?
- **Availability**
  - If some nodes fail, does everything still work?
- **Partitioning**
  - If two sections of your system cannot talk to each other, can they make forward progress on their own?
    - If not, you sacrifice Availability (slow response)
    - If so, you might have to sacrifice Consistency – can't have everything
- Conventional databases assume no partitioning, but to provide high availability, clusters are assumed to be small
- NoSQL systems may sacrifice consistency or availability

CAP theorem


P
Bigtable
HBase
Hypertable
Megastore
Spanner
Accumulo
Dynamo
Riak
Voldemort
Cassandra
Couch DB
C
A
RDBMS

Data stores

extensible record stores | document stores | key-value stores

Year	System/Paper	Scale to 1000s	Primary Index	Secondary Indexes	Transactions	Joins/Analytics	Integrity Constraints	Views	Language/ Algebra	Data model	my label
1971	RDBMS	O	✔	✔	✔	✔	✔	✔	✔	tables	sql-like
2003	memcached	✔	✔	O	O	O	O	O	O	key-val	nosql
2004	MapReduce	✔	O	O	O	✔	O	O	O	key-val	batch
2005	CouchDB	✔	✔	✔	record	MR	O	✔	O	document	nosql
2006	BigTable (Hbase)	✔	✔	✔	record	compat. w/MR	/	O	O	ext. record	nosql
2007	MongoDB	✔	✔	✔	EC, record	O	O	O	O	document	nosql
2007	Dynamo	✔	✔	O	O	O	O	O	O	key-val	nosql
2008	Pig	✔	O	O	O	✔	/	O	✔	tables	sql-like
2008	HIVE	✔	O	O	O	✔	✔	O	✔	tables	sql-like
2008	Cassandra	✔	✔	✔	EC, record	O	✔	✔	O	key-val	nosql
2009	Voldemort	✔	✔	O	EC, record	O	O	O	O	key-val	nosql
2009	Riak	✔	✔	✔	EC, record	MR	O			key-val	nosql
2009	Redis	✔	✔	✔	group	O	O	O	✔	key-val	nosql
2010	Dremel	✔	O	O	O	/	✔	O	✔	tables	sql-like
2011	Megastore	✔	✔	✔	entity grou ps	O	/	O	/	tables	nosql
2011	Tenzing	✔	O	O	O	O	✔	✔	✔	tables	sql-like
2011	Spark/Shark	✔	O	O	O	✔	✔	O	✔	tables	sql-like
2012	Spanner	✔	✔	✔	✔	?	✔	✔	✔	tables	sql-like
2012	Accumulo	✔	✔	✔	record	compat. w/MR	/	O	O	ext. record	nosql
2013	Impala	✔	O	O	O	✔	✔	O	✔	tables	sql-like
2014	MS Cosmos	✔	✔	O	EC	O	O	O	✔	document	nosql

Extended from Bill Howe's Data Science course materials (2013)

NoSQL features

- Ability to horizontally scale “simple operation” throughput over many servers

    Simple = key lookups, read/write of one or few records

- The ability to replicate and partition data over many servers
- Consider “sharding” and “horizontal partitioning” to be synonyms
- A simple API – no query language
- A weaker concurrency model than ACID transactions
- Efficient use of distributed indexes and RAM for data storage
- The ability to dynamically add new attributes to data records

ACID vs. BASE

- **ACID** = **A**tomicity, **C**onsistency, **I**solation, and **D**urability
- **BASE** = **B**asically **A**vailable, **S**oft state, **E**ventually consistent (not the Consistency of ACID)

- Don't use "BASE" – it didn't stick
- Aside: Consistency in ACID - Any data written to the database must be valid according to all defined rules

Major impact systems (Rick Cattel)

- "**Memcached** demonstrated that in-memory indexes can be highly scalable, distributing and replicating objects over multiple nodes."
- "**Dynamo** pioneered the idea of [using] eventual consistency as a way to achieve higher availability and scalability: data fetched are not guaranteed to be up-to-date, but updates are guaranteed to be propagated to all nodes eventually."
- "**BigTable (and HBASE)** demonstrated that persistent record storage could be scaled to thousands of nodes, a feat that most of the other systems aspire to."

Document store: CouchDB, MongoDB

Year	System/ Paper	Scale to 1000s	Primary Index	Secondary Indexes	Transactions	Joins/ Analytics	Integrity Constraints	Views	Language/ Algebra	Data model	my label
1971	RDBMS	O	✔	✔	✔	✔	✔	✔	✔	tables	sql-like
2003	memcached	✔	✔	O	O	O	O	O	O	key-val	nosql
2004	MapReduce	✔	O	O	O	✔	O	O	O	key-val	batch
2005	CouchDB	✔	✔	✔	record	MR	O	✔	O	document	nosql
2006	BigTable (Hbase)	✔	✔	✔	record	compat. w/MR	/	O	O	ext. record	nosql
2007	MongoDB	✔	✔	✔	EC, record	O	O	O	O	document	nosql
2007	Dynamo	✔	✔	O	O	O	O	O	O	key-val	nosql
2008	Pig	✔	O	O	O	✔	/	O	✔	tables	sql-like
2008	HIVE	✔	O	O	O	✔	✔	O	✔	tables	sql-like
2008	Cassandra	✔	✔	✔	EC, record	O	✔	✔	O	key-val	nosql
2009	Voldemort	✔	✔	O	EC, record	O	O	O	O	key-val	nosql
2009	Riak	✔	✔	✔	EC, record	MR	O			key-val	nosql
2009	Redis	✔	✔	✔	group	O	O	O	✔	key-val	nosql
2010	Dremel	✔	O	O	O	/	✔	O	✔	tables	sql-like
2011	Megastore	✔	✔	✔	entity groups	O	/	O	/	tables	nosql
2011	Tenzing	✔	O	O	O	O	✔	✔	✔	tables	sql-like
2011	Spark/Shark	✔	O	O	O	✔	✔	O	✔	tables	sql-like
2012	Spanner	✔	✔	✔	✔	?	✔	✔	✔	tables	sql-like
2012	Accumulo	✔	✔	✔	record	compat. w/MR	/	O	O	ext. record	nosql
2013	Impala	✔	O	O	O	✔	✔	O	✔	tables	sql-like
2014	MS Cosmos	✔	✔	O	EC	O	O	O	✔	document	nosql

Extended from Bill Howe's Data Science course materials (2013)

Data model

- Document-oriented
  - Document = set of key/value pairs
  - Example:

```
{
        "Subject": "I like Plankton"
        "Author": "Rusty"
        "PostedDate": "5/23/2006"
        "Tags": ["plankton", "baseball", "decisions"]
        "Body": "I decided today that I don't like baseball.
                 I like plankton."
}
```

Data model

```
"views":
{
	"all":
	{
		"map": "function(doc) {if (doc.Type=='customer’) emit(null, doc)}”
	},
	"by_lastname":
	{
		"map": "function(doc) {if (doc.Type == 'customer’) emit(doc.LastName, doc)}"
	},
	"total_purchases“:
	{
		"map": "function(doc) {if (doc.Type == 'purchase’) emit(doc.Customer, doc.Amount)}",
		"reduce": "function(keys, values) { return sum(values) }“
	}
}
```

NoSQL criticism

Two value propositions offered by NoSQL community

Performance	"I started with MySQL, but had a hard time scaling it out in a distributed environment"
Flexibility	"My data doesn't conform to a rigid schema"

NoSQL criticism

- **Who are the customers of NoSQL?**
  - Lots of startups
  - Very few enterprises

- **Why?** most applications are traditional OLTP on structured data; a few other applications around the “edges”, but considered less important

NoSQL criticism: Flexibility argument

- **No ACID Equals No Interest**
  - Screwing up mission-critical data is no-no-no
- **Low-level Query Language is Death**
- **NoSQL means NoStandards**
  - One (typical) large enterprise has 10,000 databases.
  - These need accepted standards