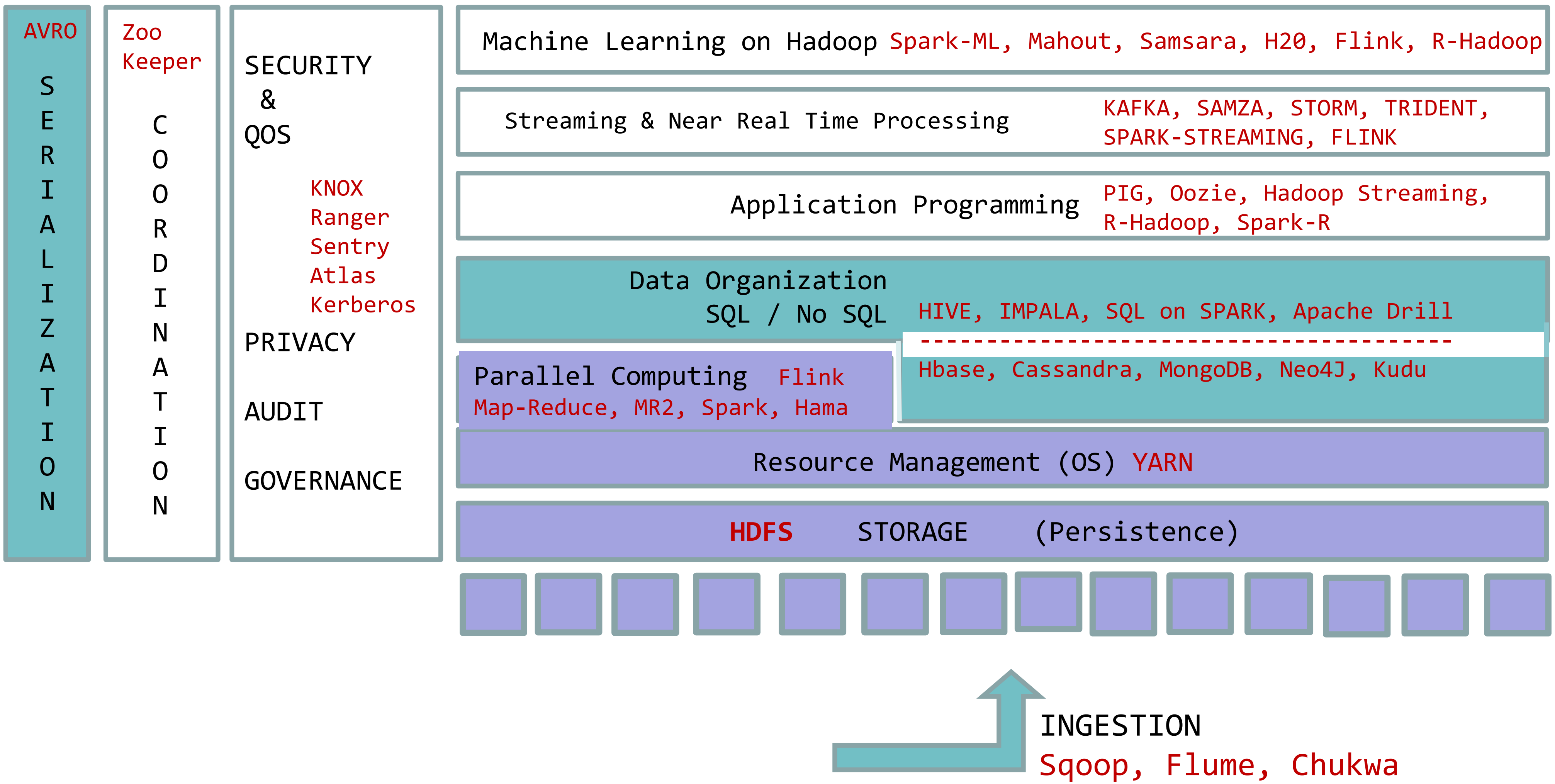
Inspire…Educate…Transform.

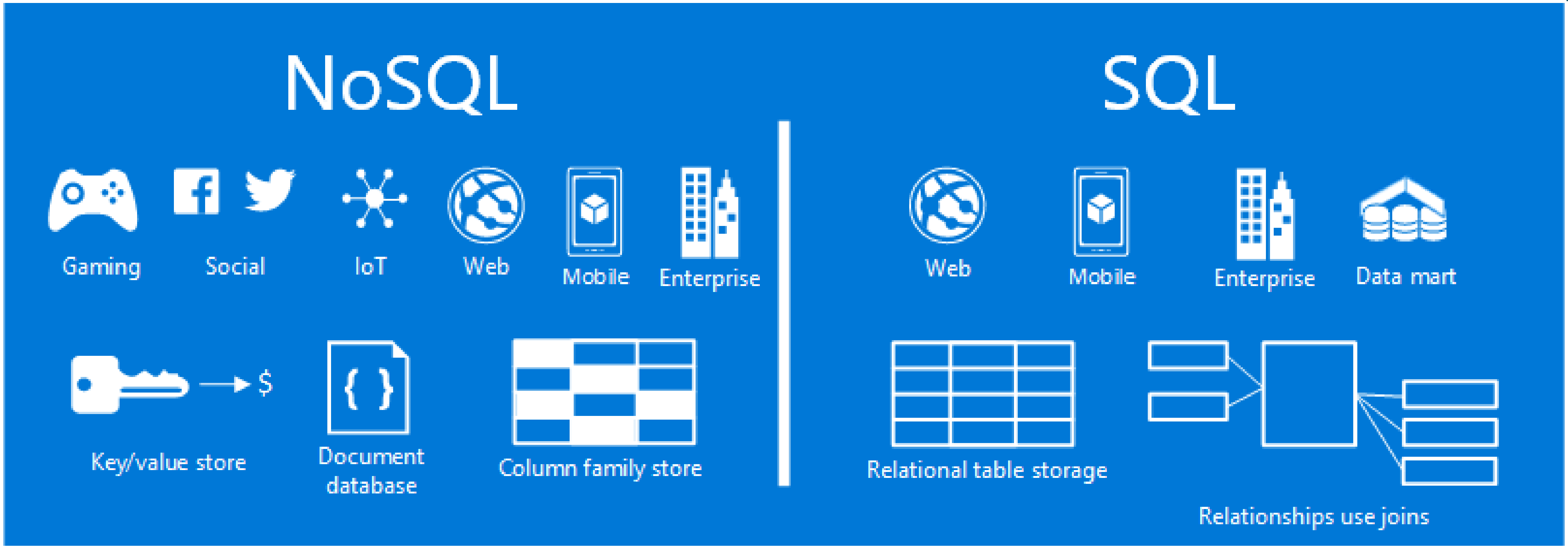
NOSql - Hbase

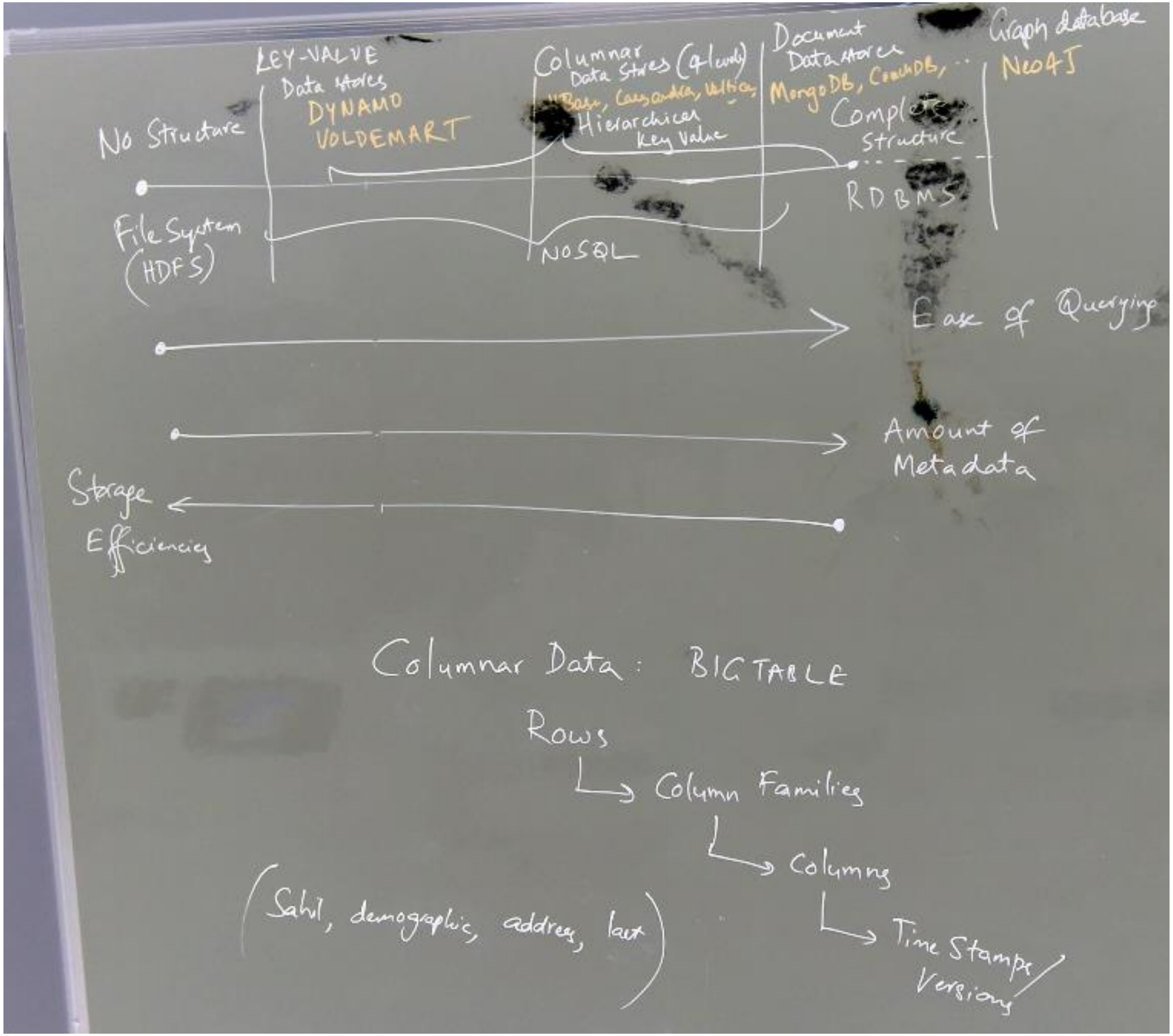
Our Focus: The open source big data ("Hadoop + Spark") ecosystem



N★SQL

*A DBA walks into a NOSQL bar, but turns and leaves because he couldn't find a **table!***




KEY-VALUE
Data stores
DYNAMO
VOLDEMART
Columnar Data Stores (4 level)
HBase, Cassandra, Vertica
Hierarchical Key Value
Document Data stores
MongoDB, CouchDB, ...
Complex Structure
Graph database
Neo4J
No Structure
File System
(HDFS)
NOSQL
RDBMS
Ease of Querying
Amount of Metadata
Storage Efficiency
Columnar Data : BIG TABLE
Rows
Column Families
Columns
Time Stamps
Versions
(Sahil, demographic, address, last)

http://nosql-database.org/ Lists >225 No-SQL databases today.

Next Generation Databases: **non-relational, distributed, open-source, horizontally scalable, schema-free, easy replication support, simple API, eventually consistent** / **BASE** (not ACID), **huge amounts of data.**

The original intention has been to build **modern web-scale databases**. The movement began early 2009.

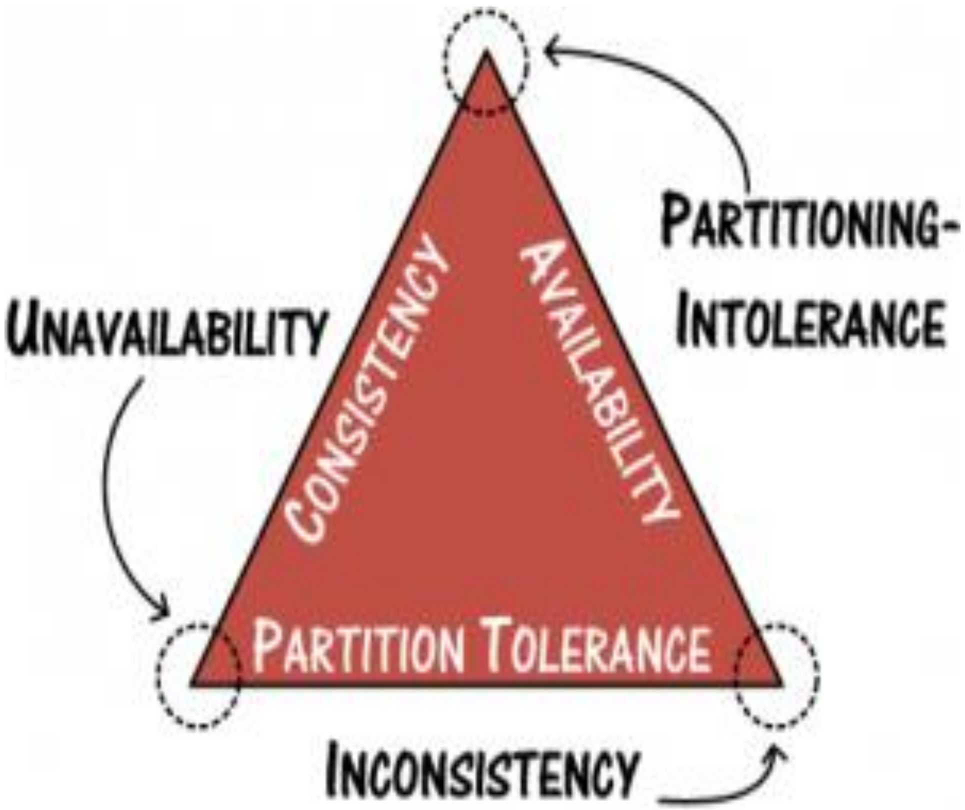
- **Document databases** pair each key with a complex data structure known as a document. Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.
- **Graph stores** are used to store information about networks, such as social connections. Graph stores include Neo4J and HyperGraphDB.
- **Key-value stores** are the simplest NoSQL databases. Every single item in the database is stored as an attribute name (or "key"), together with its value. Examples of key-value stores are Riak and Voldemort. Some key-value stores, such as Redis, allow each value to have a type, such as "integer", which adds functionality.
- **Wide-column stores** such as Cassandra and HBase are optimized for queries over large datasets, and store columns of data together, instead of rows

http://nosql-database.org/ Lists >225 No-SQL databases today.

Graph	Column	Document	Persistent Key/Value	Volatile Key/Value
neo4j	BigTable (Google)	MongoDB (~BigTable)	Dynamo (Amazon)	memcached
FlockDB (Twitter)	HBase (BigTable)	CouchDB	Voldemort (Dynamo)	Hazelcast
InfiniteGraph	Cassandra (Dynamo + BigTable)	Riak (Dynamo)	Redis	
	Hypertable (BigTable)		Membase (memcached)	
	SimpleDB (AmazonAWS)		Tokyo Cabinet	

CAP THEOREM

CAP Theorem



WE OFFER 3 KINDS OF SERVICES

GOOD-CHEAP-FAST

BUT YOU CAN PICK ONLY TWO

GOOD & CHEAP WON'T BE FAST

FAST & GOOD WON'T BE CHEAP

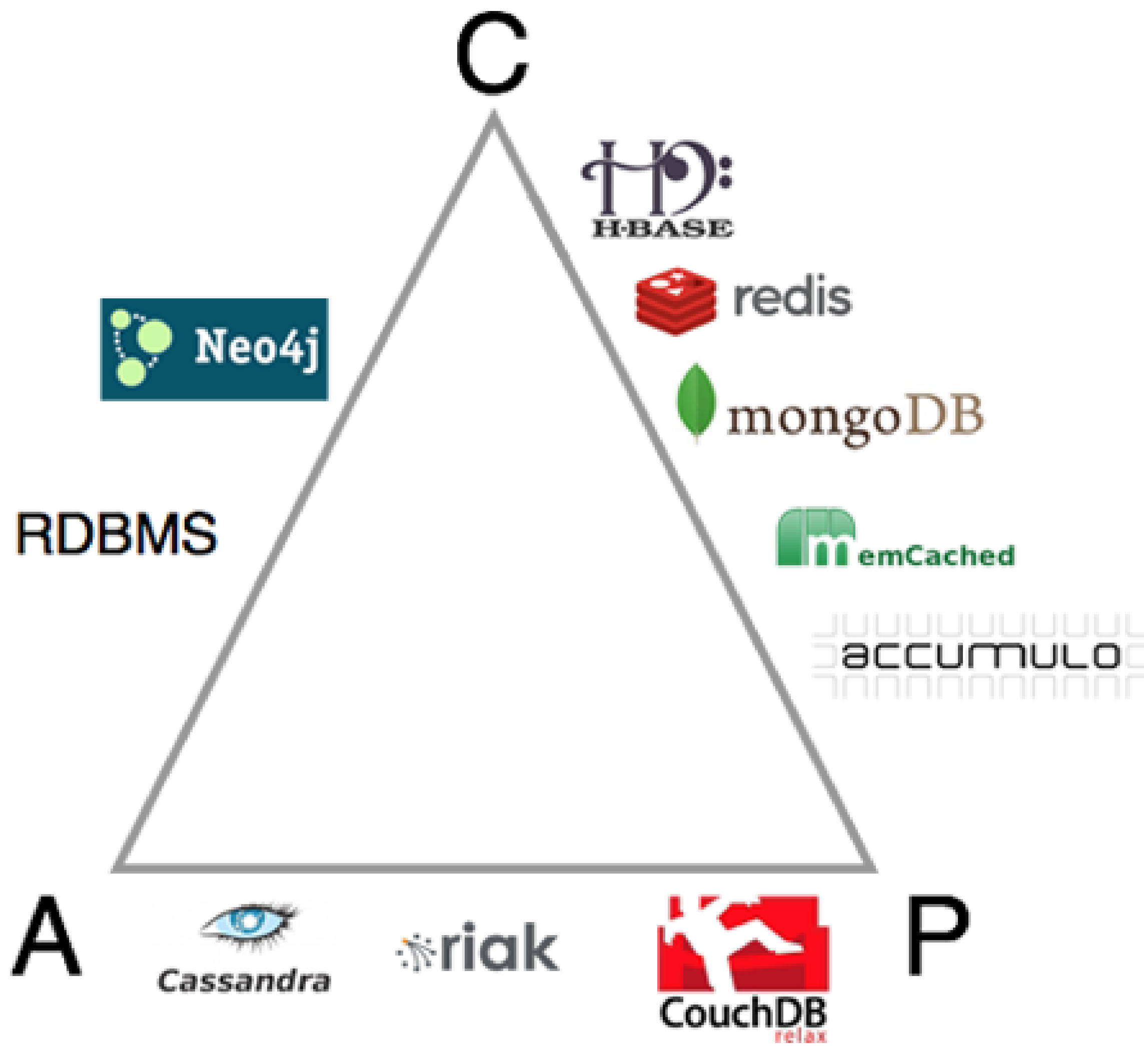
CHEAP & FAST WON'T BE GOOD

College Life

PICK2

C

A P


C
H-BASE
redis
mongoDB
memCached
accumulo
Neo4j
RDBMS
A
Cassandra
riak
CouchDB
relax
P

NoSQL Theory

- All NoSQL offerings relax one or more of the ACID properties: Atomicity, consistency, isolation, durability
  - BASE (**B**asically **A**vailable, **S**oft state, **E**ventual consistency)

CAP Theorem: It is impossible for a distributed computer system to simultaneously provide all three of the following guarantees:

- ***Consistency****:* all nodes see the same data at the same time
- ***Availability****:* a guarantee that every request receives a response on whether it was successful or failed
- ***Partition tolerance****:* the system continues to operate despite arbitrary message loss or failure of part of the system

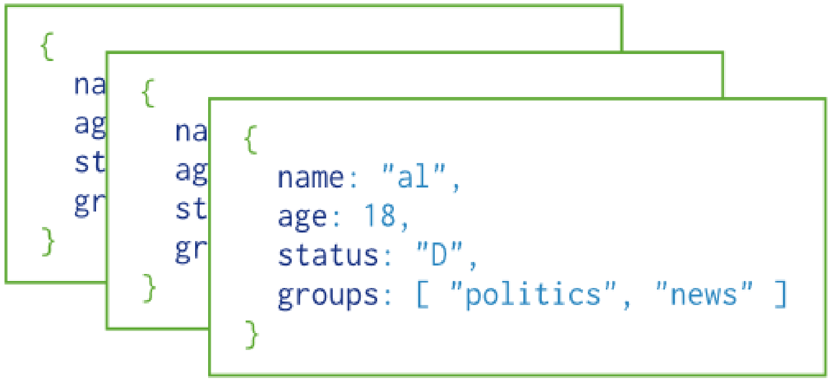
A distributed system can satisfy any two of these guarantees at the same time, but not all three.

Data is stored in JSON-like documents.
Dynamic schema. Open source.

SQL Terms/Concepts	MongoDB Terms/Concepts
database	database
table	collection
row	document
column	field
index	index
table joins (e.g. select queries)	embedded documents and linking
Primary keys	_**id** field is always the primary key
Aggregation (e.g. group by)	aggregation pipeline

MongoDB Data Model

A ***collection*** includes ***documents***.



Collection

The value of **field**:

- Native data types
- Arrays
- Other documents

Structure of a JSON-document:

Embedded documents:

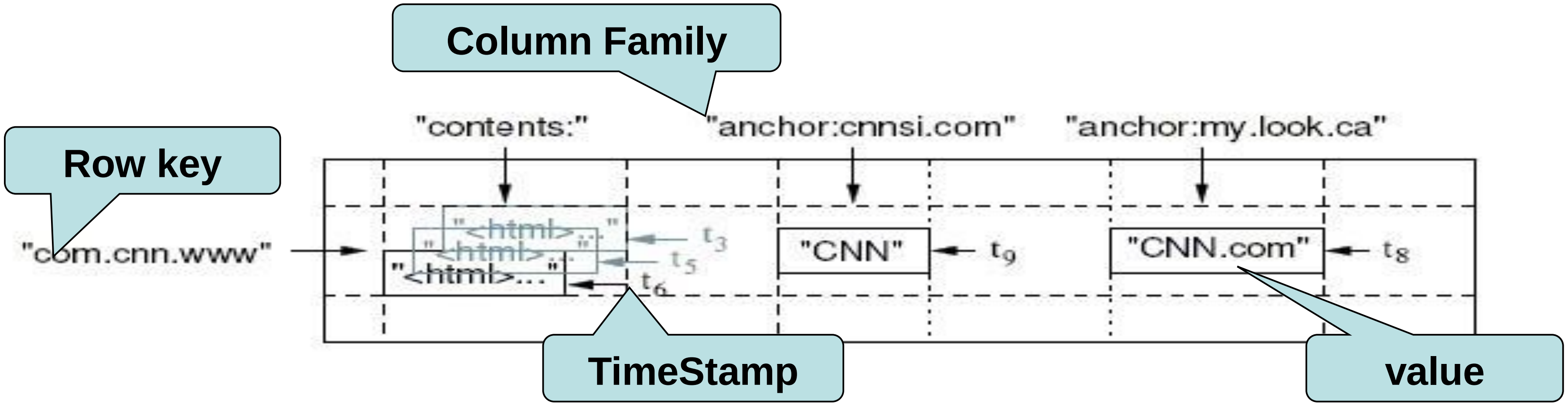
MongoDB Queries:

- CRUD (Create – Update – Delete)
  - Create a database: use database_name
  - Create a collection: db.createCollection(name, options)
    - options: specify the number of documents in a collection etc.
  - Insert a document:
    - db.<collection_name>.insert({“name”: “nguyen”, “age”: 24, “gender”: “male”})
  - Query [e.g. select all]
    - db.<collection_name>.find().pretty()
  - Query with conditions:
    - db.<collection_name>.find({ “gender”: “female”, “age”: {$lte:20} }).pretty()
  - db.<collection_name>.update(<select_criteria>,<updated_data>)
  - db.students.update({‘name':‘nguyen'}, { $set:{‘age': 20 } })
  - Replace the existing document with new one: save method:
    - db.students.save({_id:ObjectId(‘string_id’), “name”: “ben”, “age”: 23, “gender”: “male”}
  - Drop a database
    - Show database: show dbs
    - Use a database: use <db_name>
    - Drop it: db.dropDatabase()
  - Drop a collection: db.<collection_name>.drop()
  - Delete a document: db.<collection_name>.remove({“gender”: “male” })

An **open-source**, **distributed**, **column-oriented** database built on top of HDFS based on Google **BigTable**!

Big Table: Data Model

- Tables are sorted by Row
- Table schema only defines its *column families.*
  - Each family consists of any number of columns
  - Each column consists of any number of versions
  - Columns only exist when inserted, NULLs are free.
  - Columns within a family are sorted and stored together
- Everything except table names are byte[]
- (Row, Family: Column, Timestamp) → Value



HBase Logical View

Implicit PRIMARY KEY in RDBMS terms

Data is all `byte[]` in HBase

Different types of data separated into different "column families"

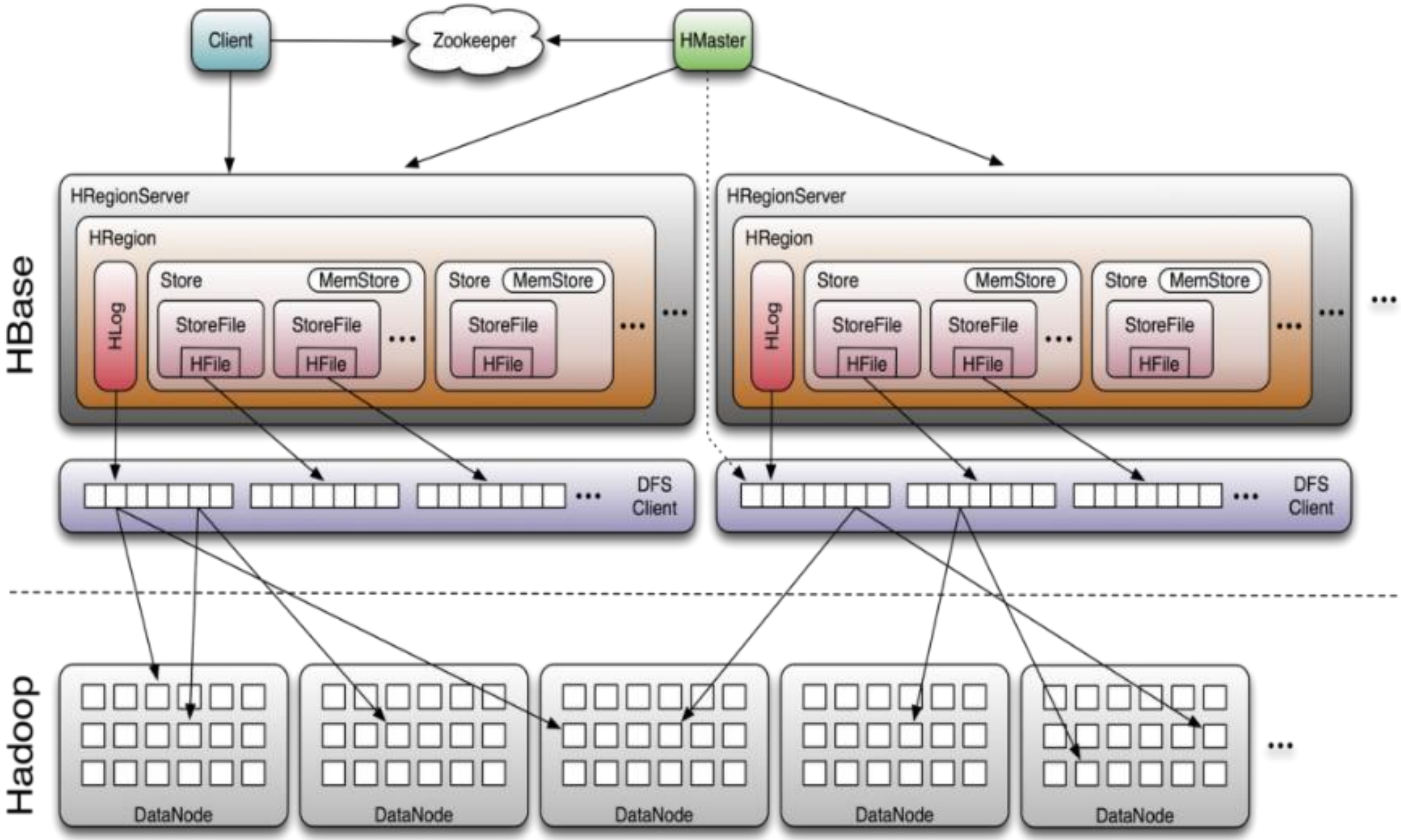
Row key	Data
cutting	info: { 'height': '9ft', 'state': 'CA' } roles: { 'ASF': 'Director', 'Hadoop': 'Founder' }
tlipcon	info: { 'height': '5ft7, 'state': 'CA' } roles: { 'Hadoop': 'Committer'@ts=2010, 'Hadoop': 'PMC'@ts=2011, 'Hive': 'Contributor' }

Different rows may have different sets of columns(table is *sparse*)

A single cell might have different values at different timestamps

Useful for *-To-Many mappings

HBase implements BigTable on Hadoop



HBase vs. HDFS

	Plain HDFS/MR	HBase
Write pattern	Append-only	Random write, bulk incremental
Read pattern	Full table scan, partition table scan	Random read, small range scan, or table scan
Hive (SQL) performance	Very good	4-5x slower
Structured storage	Do-it-yourself / TSV / SequenceFile / Avro / ?	Sparse column-family data model
Max data size	30+ PB	~1PB

HBase vs. RDBMS

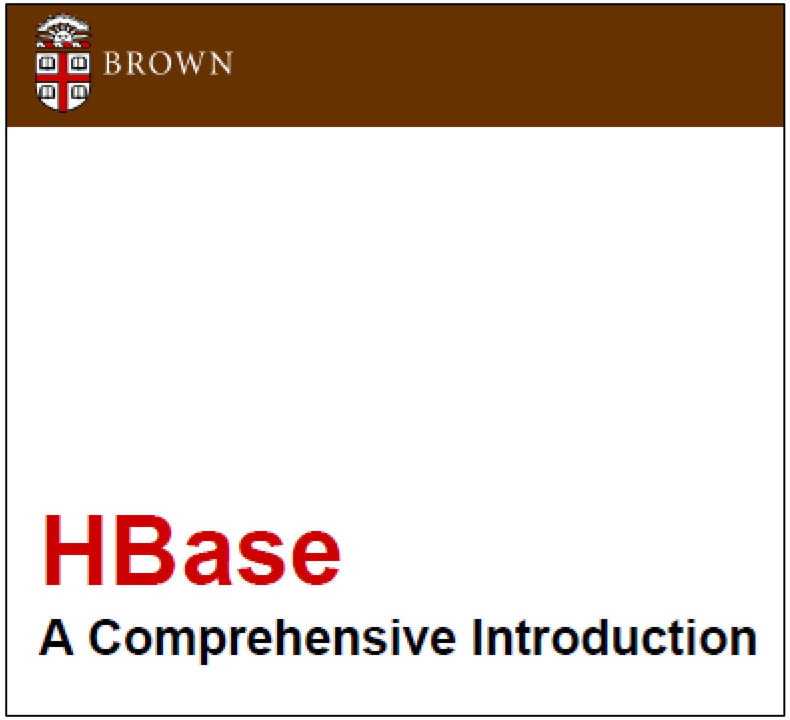
	RDBMS	HBase
Data layout	Row-oriented	Column-family-oriented
Transactions	Multi-row ACID	Single row only
Query language	SQL	get/put/scan/etc *
Security	Authentication/Authorization	Work in progress
Indexes	On arbitrary columns	Row-key only
Max data size	TBs	~1PB
Read/write throughput limits	1000s queries/second	Millions of queries/second

345 systems in ranking, September 2018

Rank			DBMS	Database Model	Score		
Sep 2018	Aug 2018	Sep 2017			Sep 2018	Aug 2018	Sep 2017
1.	1.	1.	Oracle	Relational DBMS	1309.12	-2.91	-49.97
2.	2.	2.	MySQL	Relational DBMS	1180.48	-26.33	-132.13
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1051.28	-21.37	-161.26
4.	4.	4.	PostgreSQL	Relational DBMS	406.43	-11.07	+34.07
5.	5.	5.	MongoDB	Document store	358.79	+7.81	+26.06
6.	6.	6.	DB2	Relational DBMS	181.06	-0.78	-17.28
7.	↑ 8.	↑ 10.	Elasticsearch	Search engine	142.61	+4.49	+22.61
8.	↓ 7.	↑ 9.	Redis	Key-value store	140.94	+2.37	+20.54
9.	9.	↓ 7.	Microsoft Access	Relational DBMS	133.39	+4.30	+4.58
10.	10.	↓ 8.	Cassandra	Wide column store	119.55	-0.02	-6.65
11.	11.	11.	SQLite	Relational DBMS	115.46	+1.73	+3.42
12.	12.	12.	Teradata	Relational DBMS	77.38	-0.02	-3.52
13.	13.	↑ 16.	Splunk	Search engine	74.03	+3.53	+11.45
14.	14.	↑ 18.	MariaDB	Relational DBMS	70.64	+2.34	+15.17
15.	15.	↓ 13.	Solr	Search engine	60.20	-1.69	-9.71
16.	↑ 18.	↑ 19.	Hive	Relational DBMS	59.63	+1.69	+11.02
17.	17.	↓ 15.	HBase	Wide column store	58.47	-0.33	-5.87

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

This document will be shared as additional reading today.


BROWN

HBase

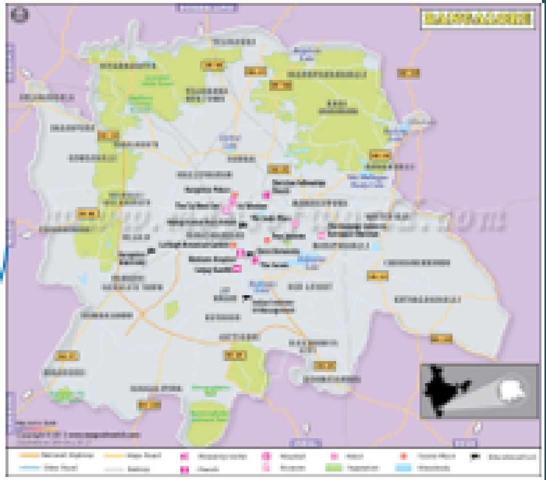
A Comprehensive Introduction

A good summary book on select NOSQL databases

WWW.CHRISTOF-STRAUCH.DE/NOSQLDBS.PDF

INSOFE

Inspire...Educate...Transform.



HYDERABAD

Office and Classrooms

Plot 63/A, Floors 1&2, Road # 13, Film Nagar, Jubilee Hills, Hyderabad - 500 033

+91-9701685511 (Individuals)

+91-9618483483 (Corporates)

BENGALURU

Office

Incubex, #728, Grace Platina, 4th Floor, CMH Road, Indira Nagar, 1st Stage, Bengaluru – 560038

+91-9502334561 (Individuals)

+91-9502799088 (Corporates)

Classroom

KnowledgeHut Solutions Pvt. Ltd., Reliable Plaza, Jakkasandra Main Road, Teacher's Colony, 14th Main Road, Sector – 5, HSR Layout, Bengaluru - 560102

Social Media

Web:	http://www.insofe.edu.in
Facebook:	https://www.facebook.com/insofe
Twitter:	https://twitter.com/Insofeedu
YouTube:	http://www.youtube.com/InsofeVideos
SlideShare:	http://www.slideshare.net/INSOFE
LinkedIn:	http://www.linkedin.com/company/international-school-of-engineering

This presentation may contain references to findings of various reports available in the public domain. INSOFE makes no representation as to their accuracy or that the organization subscribes to those findings.