10. NoSQL

- Introducció
- Bases de dades SQL
- Bases de dades NoSQL
- Bases de dades NewSQL

NoSQL: El nom

- "SQL" = SGBD relacionals tradicionals
- S'acaba la tendència "One size fits all": No tots els problemes de gestió/anàlisi de dades es resolen millor usant ***exclusivament*** SGBD relacionals tradicionals
- "No SQL" ≠ No usar SQL
- "NoSQL" = "Not only SQL" = No usar ***només*** SGBD tradicionals

No tots els problemes de gestió/anàlisi de dades es resolen millor usant SGBD relacionals tradicionals

- Persistent
- Eficient
- Massiu
- Multi-usuari
- Segur
- Fiable
- Convenient

Per algunes aplicacions és més del que necessitem

Avui en dia molt, molt, més massives i més eficients

SQL Characteristics

- Data stored in columns and tables
- Relationships represented by data
- Data Manipulation Language
- Data Definition Language
- Transactions
- Abstraction from physical layer

SQL Physical Layer Abstraction

- Applications specify what, not how
- Query optimization engine
- Physical layer can change without modifying applications
  - Create indexes to support queries
  - In Memory databases

Data Definition Language

- Schema defined at the start
- Create Table (Column1 Datatype1, Column2 Datatype 2, …)
- Constraints to define and enforce relationships
  - Primary Key
  - Foreign Key
  - Etc.
- Triggers to respond to Insert, Update , & Delete
- Stored Modules
- Alter …
- Drop …
- Security and Access Control

Data Manipulation Language (DML)

- Data manipulated with Select, Insert, Update, & Delete statements
  - Select T1.Column1, T2.Column2 …
    From Table1, Table2 …
    Where T1.Column1 = T2.Column1 …
- Data Aggregation
- Compound statements
- Functions and Procedures
- Explicit transaction control

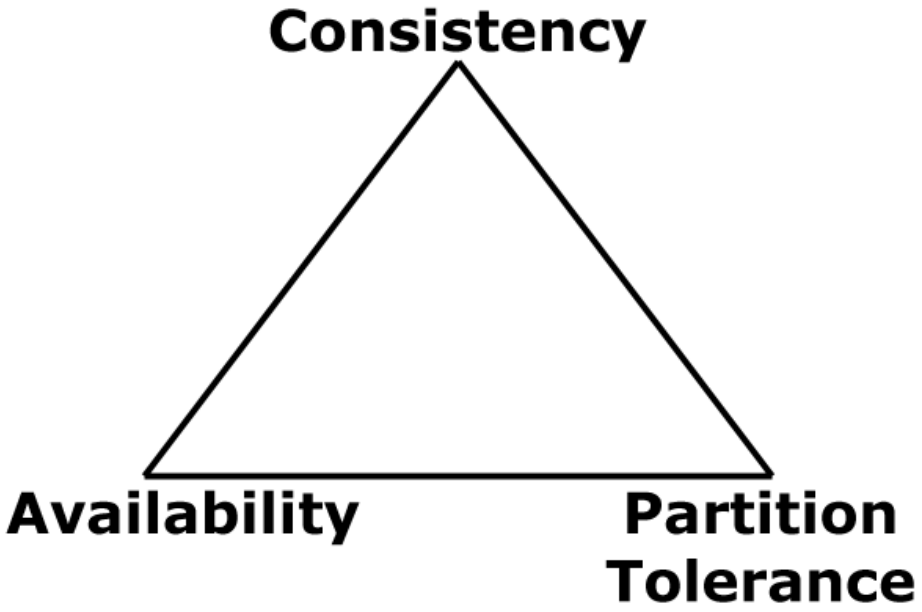
Transactions – ACID Properties

- **A**tomic – All of the work in a transaction completes (commit) or none of it completes
- **C**onsistent – A transaction transforms the database from one consistent state to another consistent state. Consistency is defined in terms of constraints.
- **I**solated – The results of any changes made during a transaction are not visible until the transaction has committed.
- **D**urable – The results of a committed transaction survive failures

NoSQL Distinguishing Characteristics

- Large data volumes
  - Google's "big data"
- Scalable replication and distribution
  - Potentially thousands of machines
  - Potentially distributed around the world
- Queries need to return answers quickly
- Mostly query, few updates
- Asynchronous Inserts & Updates
- Schema-less
- ACID transaction properties are not needed – BASE
- CAP Theorem
- Open source development

CAP THEOREM



The CAP Theorem by **Eric Brewer**.

CAP stands for **C**onsistency, **A**vailability, (network) **P**artition tolerance and the theorem claims that ***in a distributed system, when there is an inevitable network partition (and the cluster breaks into two or more "islands"), you can't guarantee both availability (for updates) and consistency.***

BASE Transactions

- Acronym contrived to be the opposite of ACID
  - **B**asically **A**vailable,
  - **S**oft state,
  - **E**ventually Consistent
- Characteristics
  - Weak consistency – stale data OK
  - Availability first
  - Best effort
  - Approximate answers OK
  - Aggressive (optimistic)
  - Simpler and faster

A (Tentative) Classification of NOSQL Databases

- Key-Value Stores
  - Origin: Massive web-based distributed systems (e.g., Amazon, Google, Facebook, etc.)
  - Goal: Scalability, efficiency, fault-tolerance, support for unstructured data
  - Data Model: Collection of <Key, Value> pairs
  - Example: BigTable (HBase), Cassandra, Voldemort, SimpleDB / Dynamo, etc.

- Document-stores
  - Origin: An evolution of key-value stores where the value is semi-structured
  - Goal: Scalability, efficiency, fault-tolerance, support for semi-structured data
  - Data Model: Collection of <Key, Doc> pairs, where Doc is typically a XML or JSON document
  - Example: MongoDB (JSON), CouchDB (JSON), eXisT (XML), etc.

A (Tentative) Classification of NOSQL Databases

- Graph Databases
  - Origin: Ocurrence-based systems, where most queries refer to individuals (e.g., *traverse* queries)
  - Goal: Deal with relationships as first-class citizens (e.g., N-M relationships)
  - Data Model: Graph structures (i.e., nodes and edges)
  - Architecture: Diverse
  - Example: Neo4J, DEX, Virtuoso, ArangoDB, OrientDB, Pregel, Jiraffe, etc.

Summary

- SQL Databases
  - Predefined Schema
  - Standard definition and interface language
  - Tight consistency
  - Well defined semantics
- NoSQL Database
  - No predefined Schema
  - Per-product definition and interface language
  - Getting an answer quickly is more important than getting a correct answer

NewSQL

Give Up SQL?

- Compiler translates SQL at compile time into a sequence of low level operations
- Similar to what the NoSQL products make you program in your application
- 30 years of RDBMS experience
  - + Hard to beat the compiler
  - + High level languages are good (data independence, less code, …)
  - + Stored procedures are good!
    - — One round trip from app to DBMS rather than one one round trip per record
    - — Move the code to the data, not the other way around

NewSQL

Give Up ACID

- If you need data consistency, giving up ACID is a decision to tear your hair out by doing database "heavy lifting" in user code
- Can you guarantee you won't need ACID tomorrow?

ACID = goodness, in spite of what these guys say

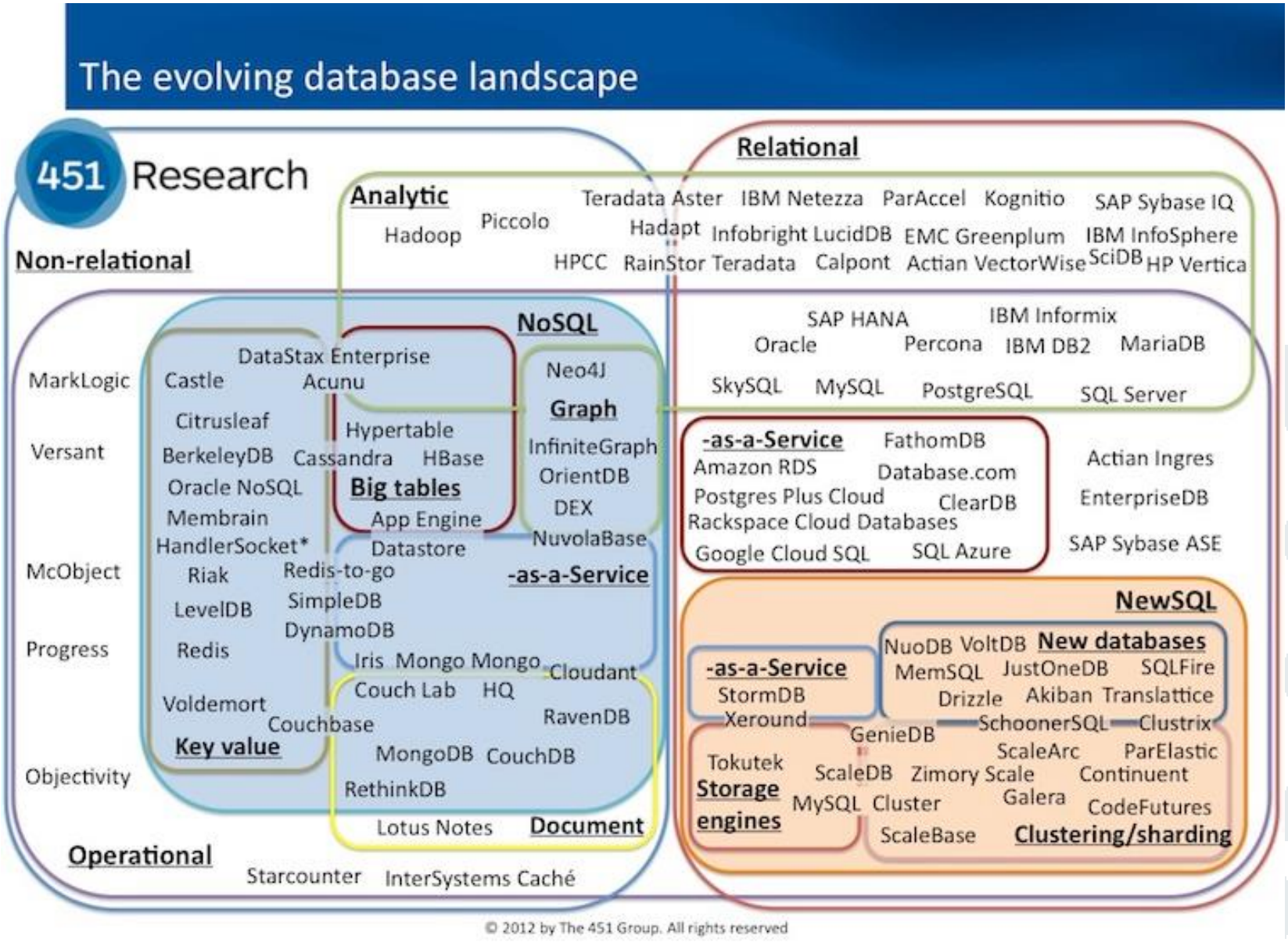
- Column-Oriented Databases
  - Origin: Query-oriented systems (e.g., data warehouses)
  - Goal: Improve the valid read ratio
  - Data Model: Relational
  - Example: HP Vertica, MonetDB, C-Store, etc.

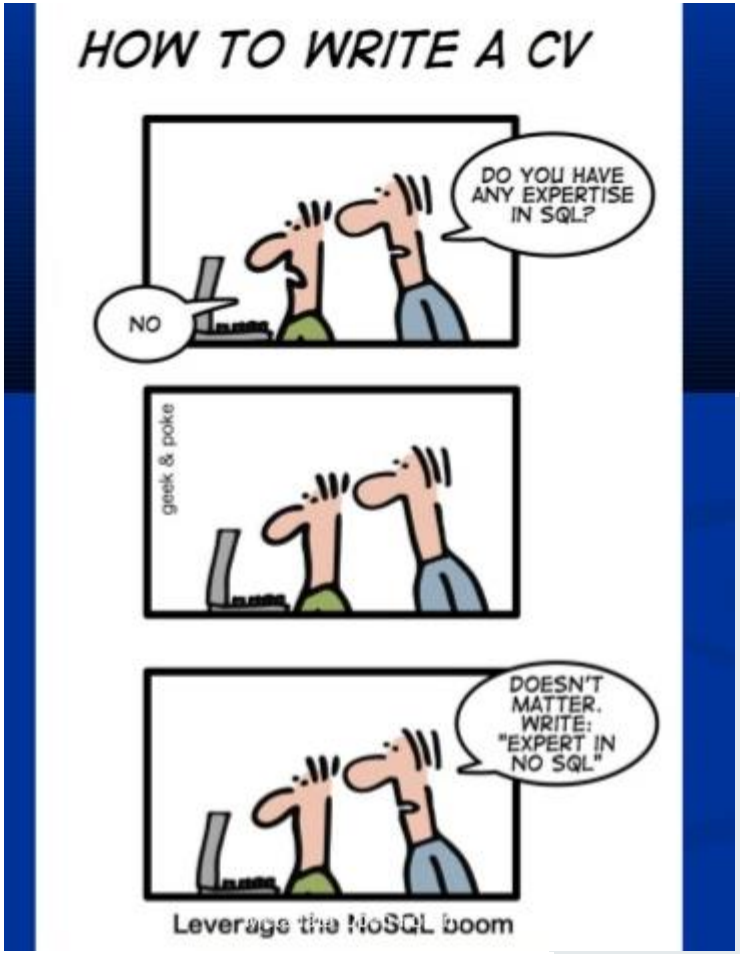
  * Note: Some key-value stores introduce the concept of column families (e.g., HBase, Cassandra), which implies local column-oriented storage at the node level

- In-memory Databases / Embedded Databases
  - Origin: SQL is not the problem
  - Goal: Improve performance
  - Data Model: Relational
  - Example:

    VoltDB (in-memory),

    SQLite (embedded, optionally also in-memory) the most widely deployed database engine (firefox, chrome, android, apple *IOS, nokia …)*


The evolving database landscape
451 Research
Non-relational
Relational
Analytic
Hadoop
Piccolo
Teradata Aster
IBM Netezza
ParAccel
Kognitio
SAP Sybase IQ
Hadapt
Infobright
LucidDB
EMC Greenplum
IBM InfoSphere
HPCC
RainStor
Teradata
Calpont
Actian VectorWise
SciDB
HP Vertica
NoSQL
MarkLogic
Versant
McObject
Progress
Objectivity
Castle
DataStax Enterprise
Acunu
Citrusleaf
Hypertable
BerkeleyDB
Cassandra
HBase
Oracle NoSQL
Big tables
Membrain
App Engine
HandlerSocket*
Datastore
Riak
Redis-to-go
LevelDB
SimpleDB
DynamoDB
Redis
Iris
Mongo
Mongo
Cloudant
Couch Lab
HQ
Voldemort
Couchbase
RavenDB
Key value
MongoDB
CouchDB
RethinkDB
Lotus Notes
Document
Neo4J
Graph
InfiniteGraph
OrientDB
DEX
NuvolaBase
-as-a-Service
SAP HANA
IBM Informix
Oracle
Percona
IBM DB2
MariaDB
SkySQL
MySQL
PostgreSQL
SQL Server
-as-a-Service
FathomDB
Amazon RDS
Database.com
Actian Ingres
Postgres Plus Cloud
ClearDB
EnterpriseDB
Rackspace Cloud Databases
Google Cloud SQL
SQL Azure
SAP Sybase ASE
NewSQL
-as-a-Service
StormDB
Xeround
NuoDB
VoltDB
New databases
MemSQL
JustOneDB
SQLFire
Drizzle
Akiban
Translattice
GenieDB
SchoonerSQL
Clustrix
ScaleArc
ParElastic
Tokutek
ScaleDB
Zimory Scale
Continuent
Storage engines
MySQL Cluster
Galera
CodeFutures
ScaleBase
Clustering/sharding
Operational
Starcounter
InterSystems Caché
© 2012 by The 451 Group. All rights reserved


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

• Keith Hare, A comparison of SQL adn NOSQl database http://www.slideshare.net/Muratakal/rdbms-vs-nosql-15797058

•The 451 Group. The database landscape. http://blogs.the451group.com/information_management/2012/11/02/updated-database-landscape-graphic/

Michael Stonebraker, Samuel Madden, Daniel J. Abadi, Stavros Harizopoulos, Nabil Hachem, Pat Helland: **The End of an Architectural Era (It's Time for a Complete Rewrite).** VLDB 2007: 1150-1160

•Michael Stonebraker: **Stonebraker on NoSQL and enterprises.** Commun. ACM 54(8): 10-11 (2011)

•Michael Stonebraker: **New opportunities for New SQL.** Commun. ACM 55(11): 10-11 (2012)