Introduction to NoSQL

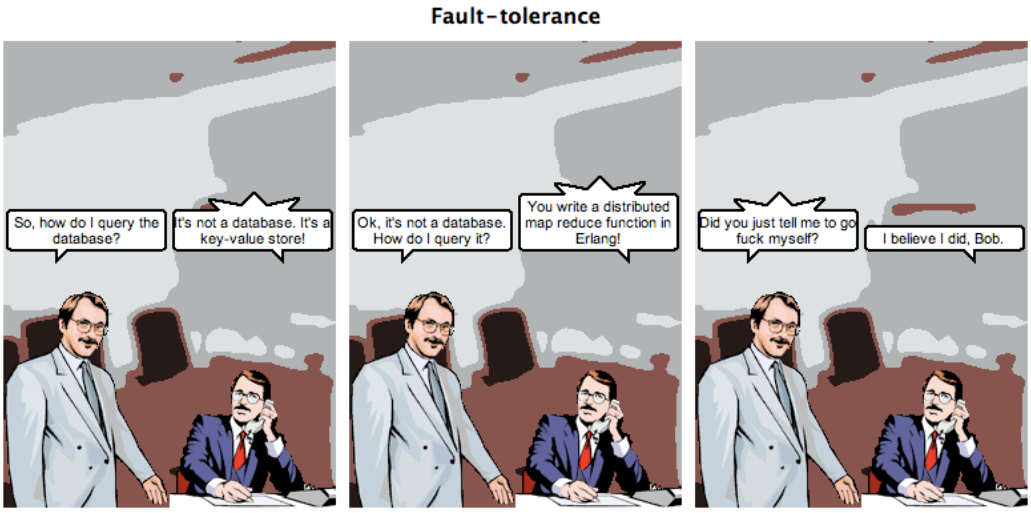
Nicolas Travers
CNAM – France

Schedule & Organization

- Introduction to **NoSQL** databases
  - 3V, ACID vs BASE, families, CAP theorem, JSon
- Presentation of **MongoDB**
  - Language, distribution, replication, application
- **Practice Works** on MongoDB
  - Queries : find + aggregate

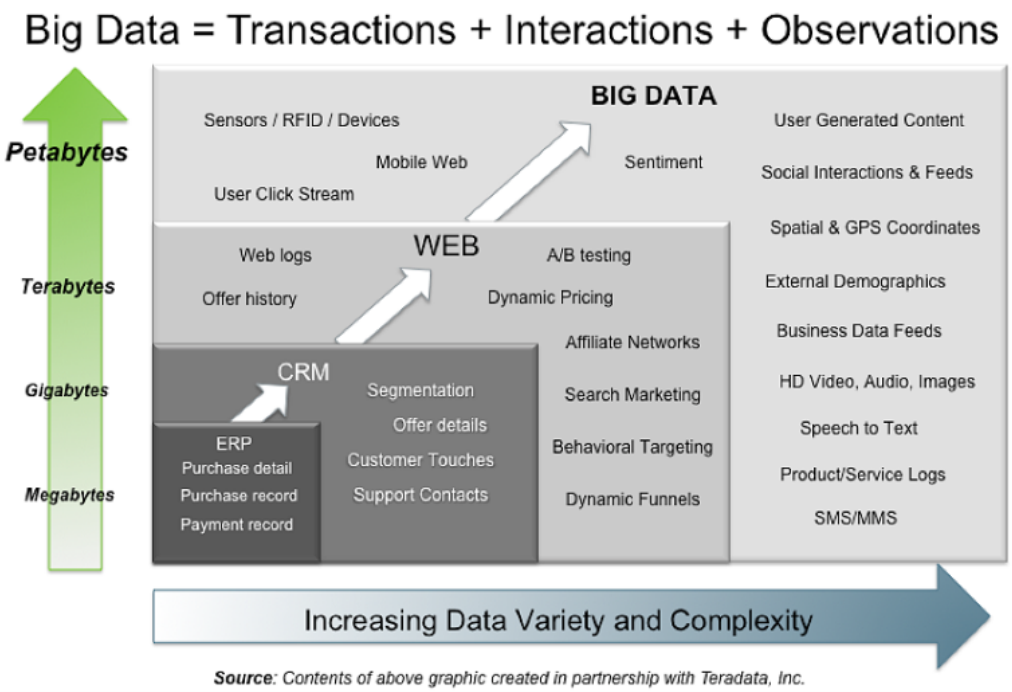
- Material available at: http://www.chewbii.com/ESSEC

DBMS vs NoSQL



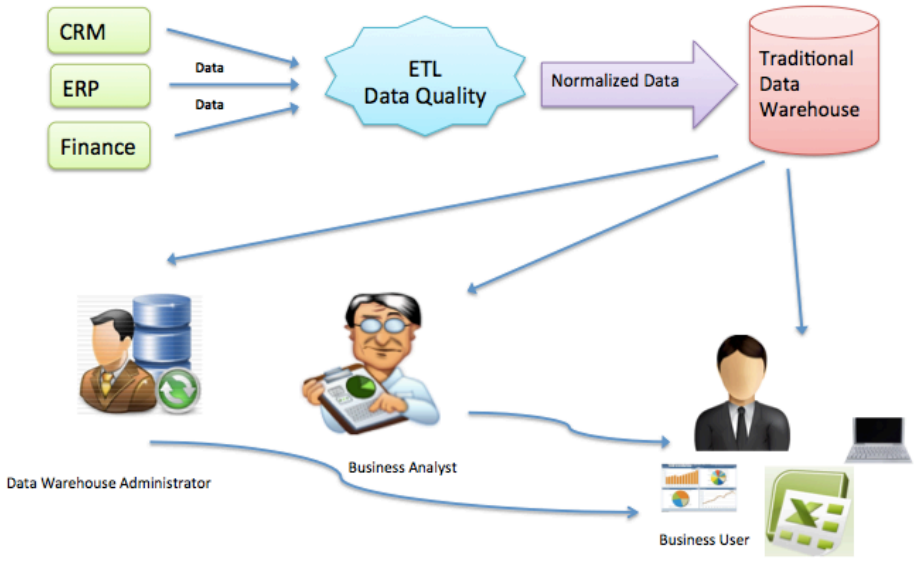
Context

- Applications and Web platforms
  - Exponential growth of the amount of Data (x2 / 2 years)
  - Unprecedent management of this volume
    - Need to distribute both computation and data
    - Huge number of servers
    - Heterogeneous data, maybe complex and often linked
- Ex :
  - Google, Amazon, Facebook
  - Google DataCenter :
    - 5000 servers/data center, ~1M de servers
  - Facebook :
    - 1 PetaBytes of data



***Source**: Contents of above graphic created in partnership with Teradata, Inc.*

BI : Traditional methods



Decisional vs 3V

Incompatible classical approach with the **3V** of *BigData* :

- **Volume**: Designed to store GB/TB of data, but needs PB (maybe EB).
- **Variety**: Heterogeneous and variable types of data, text, semi-structured
- **Velocity**: Data are produced more and more quickly

DBMS: Limitations

- Standard databases
  - Functionalities
    - Joins between tables
    - Complex queries
    - Strong coherency of data

➢ Requirements in a distributed context

- Links between entities => same server
- ++ links => difficulties for data organization

RDBMS vs Distribution

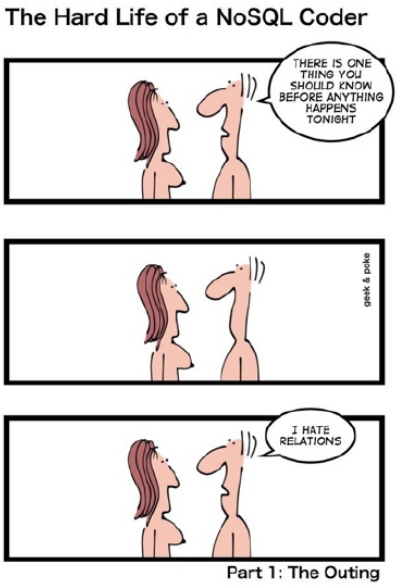
- ACID vs BASE
  - **ACID** properties for transactions
    - ***A**tomicity*: integral completion or none
    - ***C**onsistency*: consistent at start and end
    - ***I**solation*: no communication between them
      ***D**urability*: an operation cannot be reversed
  - Systèmes distribués : modèle **BASE**
    - ***B**asically **A**vailable* :
      - Any request => An answer
      - Even in a changing state
    - ***S**oft-state* :
      - Opposite to Durability.
      - System's state (servers or data) could change over time (without any update)
    - ***E**ventually consistent* :
      - With time, data can be consistent
      - Updates have to be propagated

Solution: NoSQL

- NoSQL : **Not <u>Only</u> SQL**
  - New data storage/management approach
  - Scales up the system (through distribution)
  - Complex metadata management
    - No schema

➢ Do not substitute DBMS, dedicated to:
  - Very huge volume of data (PetaBytes)
  - Very short response time
  - Consistency is not mandatory

Databases and NoSQL



NoSQL DB: Characteristics

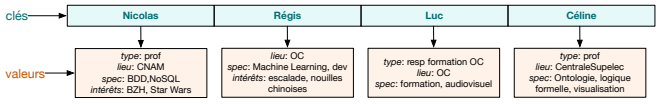
- **No relations** => *Collections*
  - No fix structures (nay none)
- **Complex data** (e.g. *documents*)
  - Objects, nesting, arrays
- **Data distribution**
  - High parallelization (Map/Reduce)
- **Data replication**
  - Disponibility vs Consistency (no transactions)
  - Few writes, many reads

Sharding : Scalability

- Datablocks are distributed in a cluster of servers
- Horizontal partitioning
- 3 types of technics:
  1. Resource allocation based: *HDFS*
     - *++ Fault tolerance and massive computations*
  2. Tree-based structure: *Clustered index* (sort)
     - *++ Physical data clustering and dynamicity*
  3. Hash-based structure: *Consistent Hashing*
     - *++ elasticity and self-management*

Several NoSQL systems

Key-Value stores



Column oriented

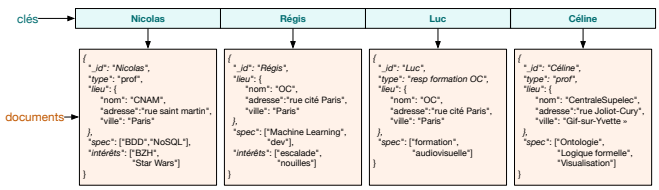
id	type
Nicolas	prof
Céline	prof
Luc	resp formation OC

id	lieu
Céline	Centrale Supelec
Nicolas	CNAM
Régis	OC
Luc	OC

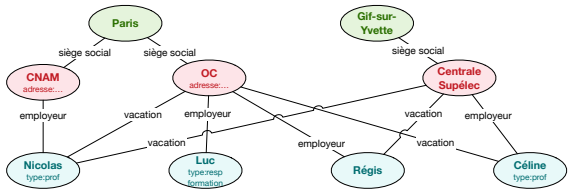
id	spec
Nicolas	BDD
Nicolas	NoSQL
Régis	Machine Learning
Régis	Dev
Luc	formation
Luc	audiovisuel
Céline	Ontologie
Céline	logique formelle
Céline	visualisation

id	intérêts
Nicolas	BZH
Nicolas	Star Wars
Régis	escalade
Régis	nouilles chinoises

Document Oriented



Graph oriented



I - NoSQL & Key-Value store

- Similar to a distributed "*HashMap*"
- Key + Value
  - No fixed schema on values (strings, object, integer, binaries…)
- Drawbacks:
  - No structures nor typing
  - No structured-based queries

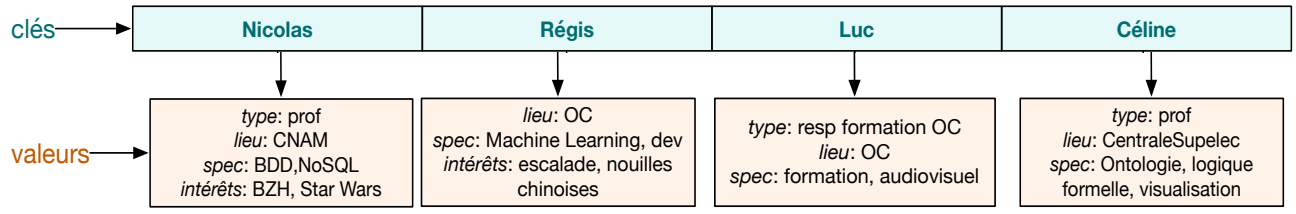
➢ DynamoDB (Amazon), Redis (VMWare), Voldemort (LinkedIn)

I - NoSQL & Key-Value store

Example

Relationnel

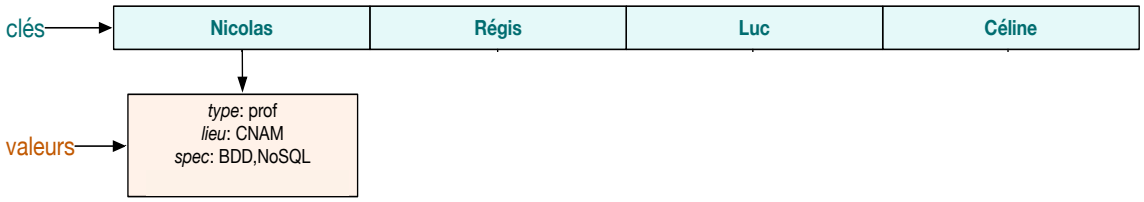
id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	



I - NoSQL & Key-Value store

Queries

- CRUD
  - CREATE (**clé**, **valeur**)
    - CREATE (**"Nicolas"**, **"type:'prof',lieu:'CNAM',spec:'BDD,NoSQL',interets:'BZH,Star Wars' "**) → **OK**
  - READ(**clé**)
    - READ(**"Nicolas"**) → **"type:'prof',lieu:'CNAM',spec:'BDD,NoSQL',interets:'BZH,Star Wars' "**
  - UPDATE(**clé**, **valeur**)
    - UPDATE(**"Nicolas"**, **"type:'prof',lieu:'CNAM,CS',spec:'BDD,NoSQL' "**) → **OK**
  - DELETE(**clé**)
    - DELETE(**"Nicolas"**) → **OK**



I - NoSQL & Key-Value store

solutions

Amazon SimpleDB

Microsoft Azure

Efficiency

Easy to set up

II – NoSQL & Columns

- Column-based storage
  - DBMS: tuples (lines)
- Easy to insert a new column
  - Dynamic schema

➢ BigTable/Hbase (Google), Cassandra (Facebook&Apache), SimpleDB (Amazon)

II - NoSQL & Columns

example

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	

id	type
Nicolas	prof
Céline	prof
Luc	resp formation OC

id	lieu
Céline	Centrale Supelec
Nicolas	CNAM
Régis	OC
Luc	OC

id	spec
Nicolas	BDD
Nicolas	NoSQL
Régis	Machine Learning
Régis	Dev
Luc	formation
Luc	audiovisuel
Céline	Ontologie
Céline	logique formelle
Céline	visualisation

id	intérêts
Nicolas	BZH
Nicolas	Star Wars
Régis	escalade
Régis	nouilles chinoises

II - NoSQL & Columns

Queries

- Column-oriented queries
  - How many **professors (type)** are at **CentraleSupelec (lieu)**

id	type
Nicolas	prof
Céline	prof

id	lieu
Céline	Centrale Supelec

id	spec
Nicolas	BDD
Nicolas	NoSQL
Régis	Machine Learning
Régis	Dev
Luc	formation
Luc	audiovisuel
Céline	Ontologie
Céline	logique formelle
Céline	visualisation

id	intérêts
Nicolas	BZH
Nicolas	Star Wars
Régis	escalade
Régis	nouilles chinoises

II - NoSQL & Columns

solutions

APACHE HBASE

elasticsearch

Spark SQL

Aggregates

Correlations

III – NoSQL & Documents

- Based on the key-value store
  - Add semi-structured data (JSon/XML)
- HTTP API
  - More complex than CRUD

➢ MongoDB, CouchDB (Apache), RavenDB, Terrastore

III - NoSQL & Documents

example

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	

clés → Nicolas | Régis | Luc | Céline

documents →

```
{
  "_id": "Nicolas",
  "type": "prof",
  "lieu": {
      "nom": "CNAM",
      "adresse":"rue saint martin",
      "ville": "Paris"
  },
  "spec": ["BDD","NoSQL"],
  "intérêts": ["BZH",
               "Star Wars"]
}
```

```
{
  "_id": "Régis",
  "lieu": {
      "nom": "OC",
      "adresse":"rue cité Paris",
      "ville": "Paris"
  },
  "spec": ["Machine Learning",
           "dev"],
  "intérêts": ["escalade",
               "nouilles"]
}
```

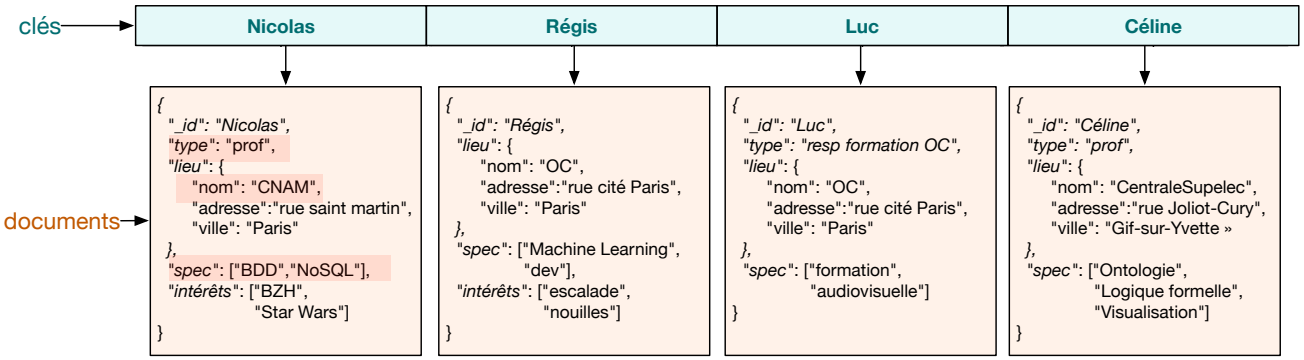
```
{
  "_id": "Luc",
  "type": "resp formation OC",
  "lieu": {
      "nom": "OC",
      "adresse":"rue cité Paris",
      "ville": "Paris"
  },
  "spec": ["formation",
           "audiovisuelle"]
}
```

```
{
  "_id": "Céline",
  "type": "prof",
  "lieu": {
      "nom": "CentraleSupelec",
      "adresse":"rue Joliot-Cury",
      "ville": "Gif-sur-Yvette »
  },
  "spec": ["Ontologie",
           "Logique formelle",
           "Visualisation"]
}
```

III - NoSQL & Documents

Queries

- Manipulations on documents content
  - **Establishment (lieu.nom)** of **professors (type)** specialized in **DB (*in* spec)**



III - NoSQL & Documents

solutions

mongoDB.

Couchbase

cassandra

DynamoDB

Richness of queries

Manage objects

IV – NoSQL & Graph

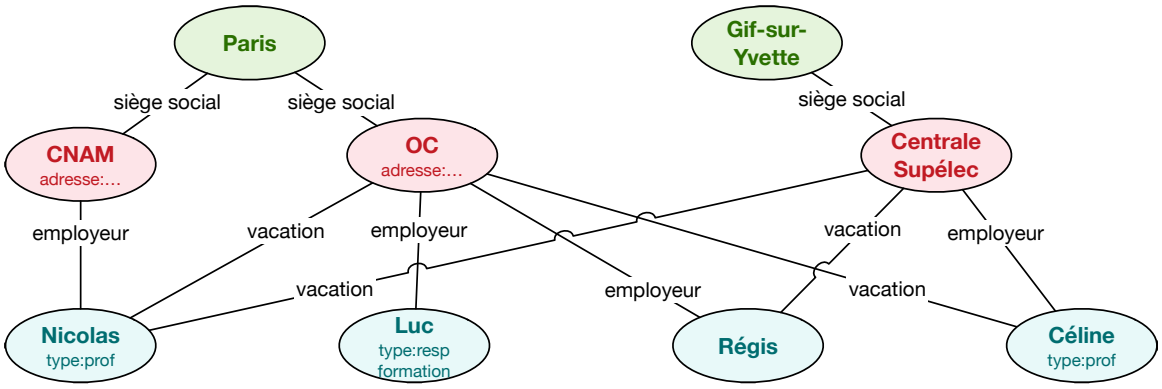
- Storage: nodes, relations and properties
  - Graph Theory
  - Path querying on the graph
  - Data are loaded on demand
  - Difficulties for modeling

➢ Neo4j, OrientDB (Apache), FlockDB (Twitter)

IV - NoSQL & Graph

example

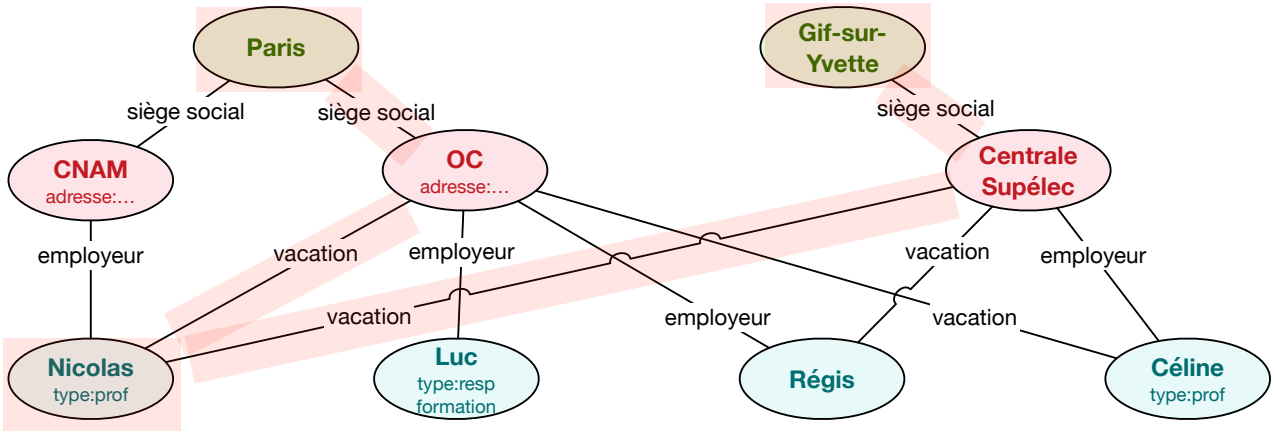
id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	



IV - NoSQL & Graph

queries

- Pattern queries
  - **Persons** giving **vacations** at **Paris** *and* **Gif-sur-Yvette**



IV - NoSQL & Graph

solutions

neo4j

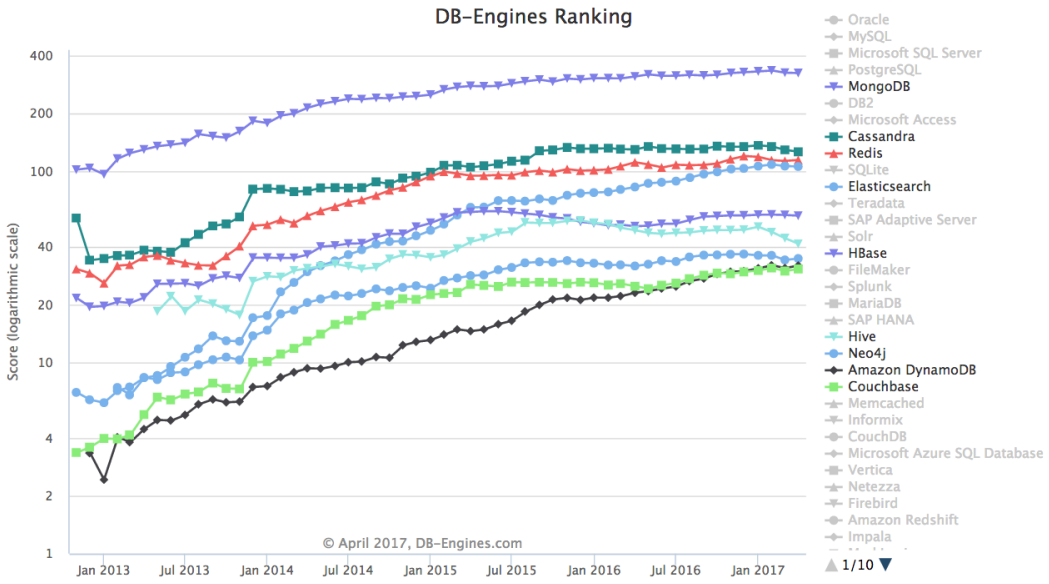
Azure Cosmos DB:

OrientDB®

FlockDB

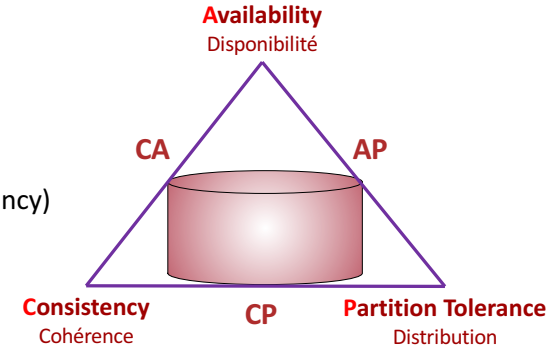
Network

Recommandation



Brewer's CAP Theorem (2000)

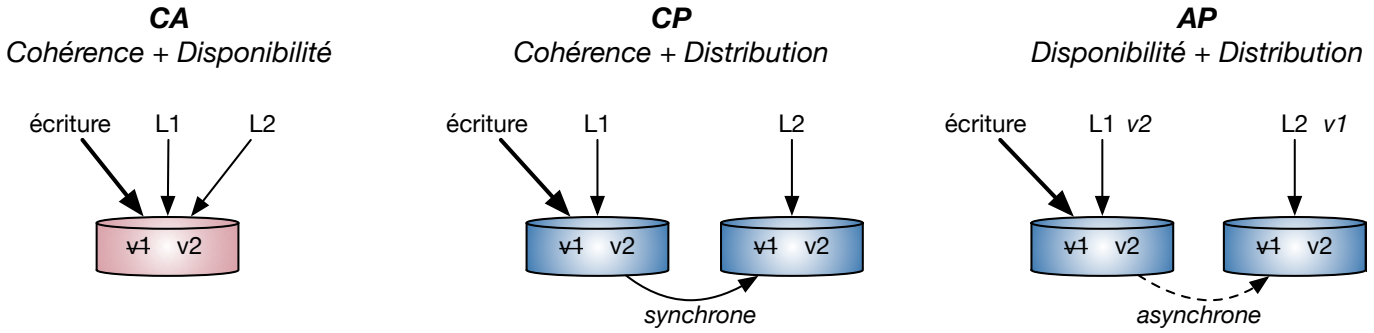
- 3 main properties for distributed management
  1. ***Consistency***:
     - A data have the same value at the same time (coherency)
  2. ***Availability***:
     - Even if a server is down, data is available
  3. ***Partition Tolerance***:
     - Even if the system is partitioned, a query must have an answer (unless for global failures)



➢ Theorem: *A distributed, networked system can have only <u>two</u> of these three properties.*

"CAP *Theorem*"

illustration



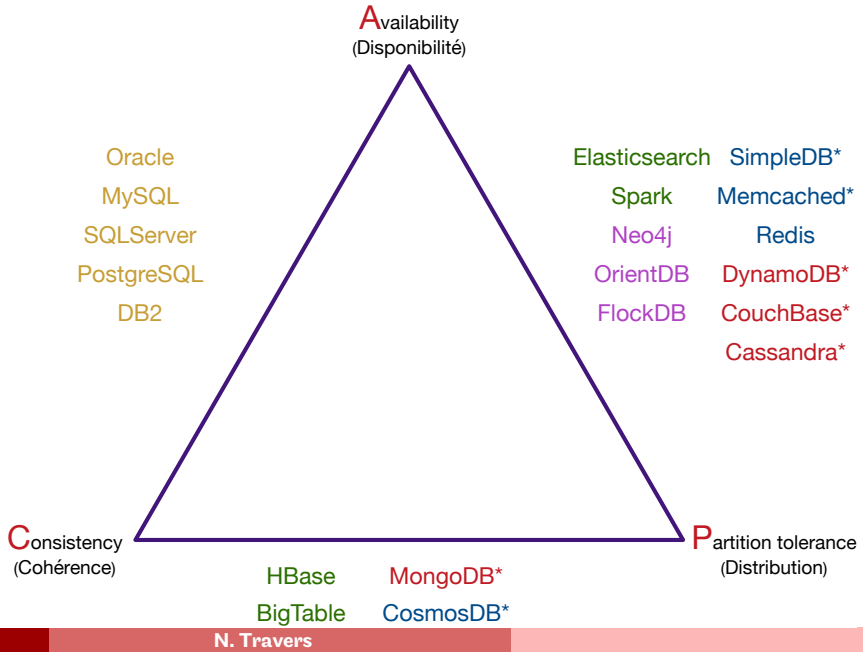
"CAP *Theorem*"

Triangle of CAP

Modèle

Relationnel
Orienté Clé-Valeur
Orienté Colonne
Orienté Document
Orienté Graphe

* Possibilité de changer le mode de cohérence
Cohérence <-> Disponibilité



JSON
JavaScript Object Notation

- Initially XML used for complex internet communications (Web Services)
  - Too verbose
- **JSON (JavaScript Object Notation)**
  - Lightweight, text-oriented, language independent
  - Used for several Web services (Google API, Twitter API)

JSon : Structures

- Key + Value
  - “lastname” : “Travers”
  - Keys with quotations
- Objects/documents
  - Collection of key/values
  - { “lastname” : “Travers”,
    “firstname” : “Nicolas”,
    “kind” : 1}

Data types

- Scalar : String, Integer, float, boolean, null…
- List : arrays […]
- Documents : objetcs {…}

Arrays

- No typing inside arrays
  - “lessons” : [“SQL”, 1, 4.2, null, “NoSQL”]
- Can nest documents
  - “doc” : [{“test” : 1},
    {“test” : {“nesting” : 1.0}},
    {“key” : “text”, “value” : null}]

JSon : Identifiers

- Key « _id » commonly used to identify documents
  - Overwrite already stored ids
  - Can be automaticaly generated
    - Ex MongoDB : "_id" : ObjectId(1234567890)

JSon : complete example

```
{
    “_id” : 1234,
    “lastname” : “Travers”, “firstname” : “Nicolas”,
    “work” : {
        “company” : “Cnam”,
        “location” : {
            “street” : “2 rue conté”,
            “city” : “Paris”,
            “zip” : 75141
        },
    },
    “fields” : [ “DB” , “DB tuning”, “XML”, “NoSQL”, “IR” ]
}
```