h e g
Haute école de gestion
Genève

NoSQL & BigData

Ch. Stettler – HEG-Genève

Hes·SO Genève
Haute École Spécialisée
de Suisse occidentale

Plan

- Définition, Motivation, Conséquence
- Distribution de l'information
- Transaction ou non
  - SGBDR : ACID
  - NoSQL : BASE
- Théorème CAP
- OLTP / OLAP
- Types de bdd NoSQL : Clé-valeur, Documents, Colonnes, Graphes

Définition

- NoSQL signifie Not Only SQL
  - il devrait plutôt s'appeler NoRelational

- BigData : de plus en plus de données stockées, qui sont de plus en plus variées, provenant de très nombreuses sources.

Motivation

- À la création des SGBDR (et du langage SQL) :
  - la place disque coutait cher (pas de redondance inutile).
  - la BDD est stockée sur un seul serveur.
- 3 aspects ont motivé la création du NoSQL :
  - **Volume** : les données créées doublent tous les 2 ans.
  - **Variété** : les données sont de plus en plus hétérogènes, non structurées.
  - **Vélocité** : les données sont créées de plus en plus vite, et nécessitent des traitements immédiats (Internet of Things).

Il y a des autre V maintenant aussi : Véracité des données (pour essayer de garantir une cohérence), Valeur, Visualisation
Car dans ces bases de données, on a plus la cohérence qu'on retrouve dans le relationnel SGBR

Conséquence

- Répartition/distribution des données sur plusieurs serveurs.

- On souhaite privilégier la performance (à la cohérence).

Distribution de l'information

- SGBDR : bdd stockée sur un seul serveur
  - la cohérence des données est fondamentale.
  - notion de relation très importante (jointures entre les tables).

- NoSQL : distribution sur plusieurs serveurs
  - vu l'accroissement de la quantité de données, produites par des systèmes applicatifs différents, du nombre croissant d'utilisateurs, de la nécessité de garantir l'accès continu, avec des temps d'accès réduits, …

Concepts SGBDR / Distribution

- SGBDR :
  - notion de transaction : propriétés **ACID**

Une transaction est un ensemble d'instruction.
Par exemple, on inserte un commande, il faut mettre à jours le client, le stock, etc..

En NoSQL, il n'y a pas cette notion de transaction. On a que des instructions isoles.

- Systèmes distribués :
  - modèle **BASE**
  - élasticité : adaptation automatique en fonction du nombre de serveurs et de la quantité de données à répartir. Permet de gérer des pics d'activité.

Transaction SQL

- Une transaction est un ensemble d'ordres SQL, qui met à jour une base de données en garantissant la cohérence de ses états successifs.
- Elle comporte un début, une suite d'ordres SQL, puis une fin.
- Elle doit respecter les principes résumés par l'acronyme **ACID**.

Transaction SQL

- **A**tomique
- **C**ohérente
- **I**solée
- **D**urable

Transaction SQL - Atomique

- Une transaction doit être complètement exécutée.
- Sinon, elle ne doit laisser aucune trace de son exécution dans la BDD.
- Tout ou Rien
- Commit / Rollback

Transaction SQL - Cohérente

- Une transaction fait passer une BDD d'un état cohérent à un autre état cohérent (lorsque la transaction est terminée, le résultat final doit respecter de nouveau toutes les contraintes).
- *Durant l'exécution de la transaction, certaines contraintes de cohérence peuvent être mises à mal.*

Transaction SQL - Isolée

- Les modifications d'une transaction ne sont visibles que quand toute la transaction a été validée.
- Si plusieurs modifications doivent se faire sur les mêmes données, elles seront faites l'une après l'autre (en série).

Tant qu'on fait pas de Commit, il n'y a rien dans la base de donnée.

Transaction SQL - Durable

- Lorsque la transaction est validée, le nouvel état est durablement inscrit dans le système.
- En cas de panne système, on repartira du dernier état validé (la dernière mise-à-jour n’est pas perdue !)

Tout ce qui a été "commité", tout doit pouvoir se retrouver dans un bas SQL.

ACID / BASE

BASIC c'est le contraire de ACID, d'ou le jeu de mot

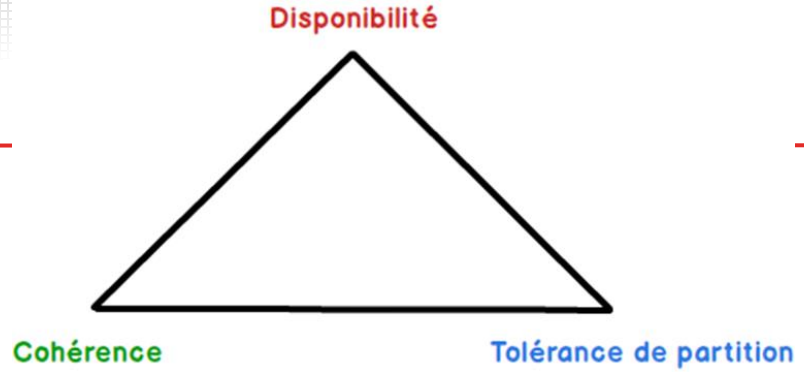
Doit être dispnible de suite.
La BDD peut changer même sans mise-à-jour
Aucune cohérence des données.

BDD NoSQL : Pas de transaction : BASE

NoSQL : PAS de transaction

- Caractérisée avec les propriétés **BASE** :
  - **B**asically **A**vailable : quelle que soit la charge de la BDD, le système garantit un taux de disponibilité.
  - **S**oft-state : la base peut changer même sans mise-à-jour (avec le temps), pas de cohérence à tout instant.
  - **E**ventually consistent : à terme, la(les) base(s) sera(seront) cohérente(s).

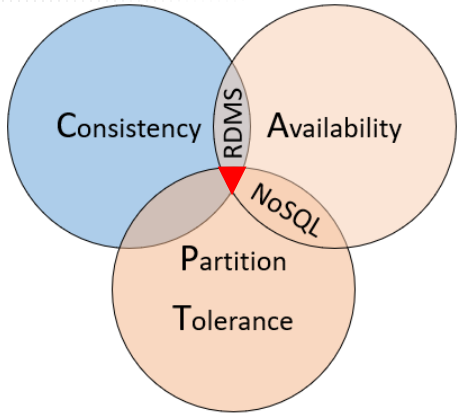
- Objectif :
  - Il est plus important d'obtenir une réponse immédiate que d'avoir un état continuellement cohérent.



Théorème CAP

- **C**onsistency (cohérence)
  - les données sont cohérentes et identiques partout (sur chaque nœud).
- **A**vailability (disponibilité)
  - la perte d'un nœud n'empêche pas le système de fonctionner et de servir l'intégralité des données.
- **P**artition tolerance (partitionnement, résistance au morcellement)
  - chaque nœud doit pouvoir fonctionner de manière autonome.
- **Ces 3 propriétés ne peuvent pas être satisfaites ensemble !**

On doit péjoré au moins 1.
En SQL, on a la dispo et la cohérence, mais pas de distribution sur plusieurs serveurs

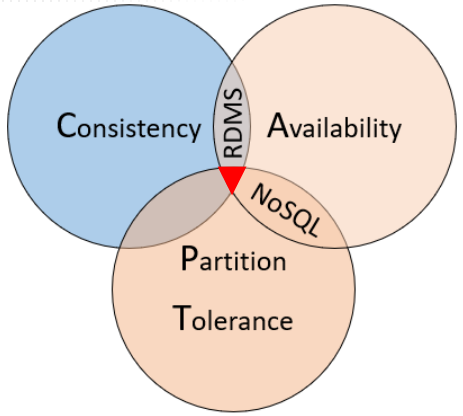


Théorème CAP

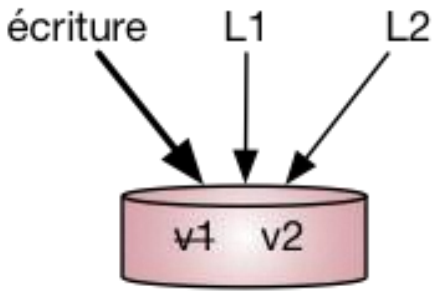
- SGBDR
  - Cohérence et disponibilité, mais pas de partitionnement/distribution.

- NoSQL
  - On privilégie la disponibilité et le partitionnement.
  - Si on souhaitait garantir la cohérence des données, comment s'assurer que l'information retrouvée soit bien la dernière version ?

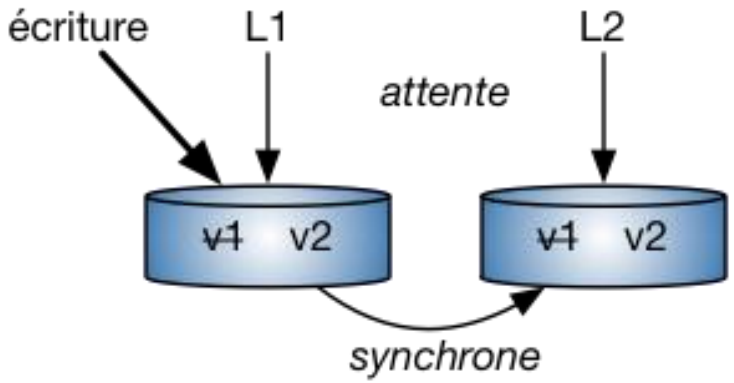
CAP



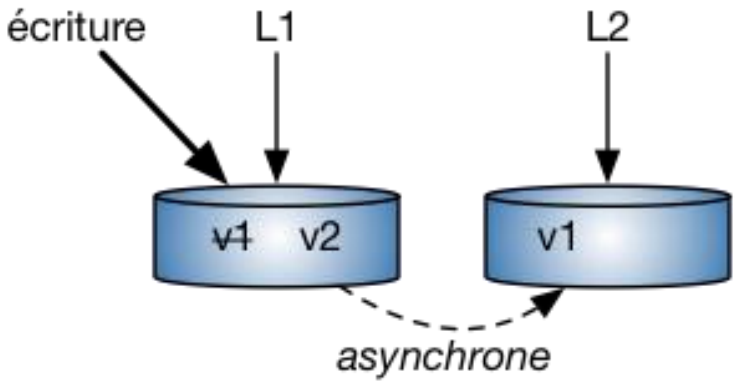
CA
Cohérence + Disponibilité



CP
Cohérence + Distribution



AP
Disponibilité + Distribution

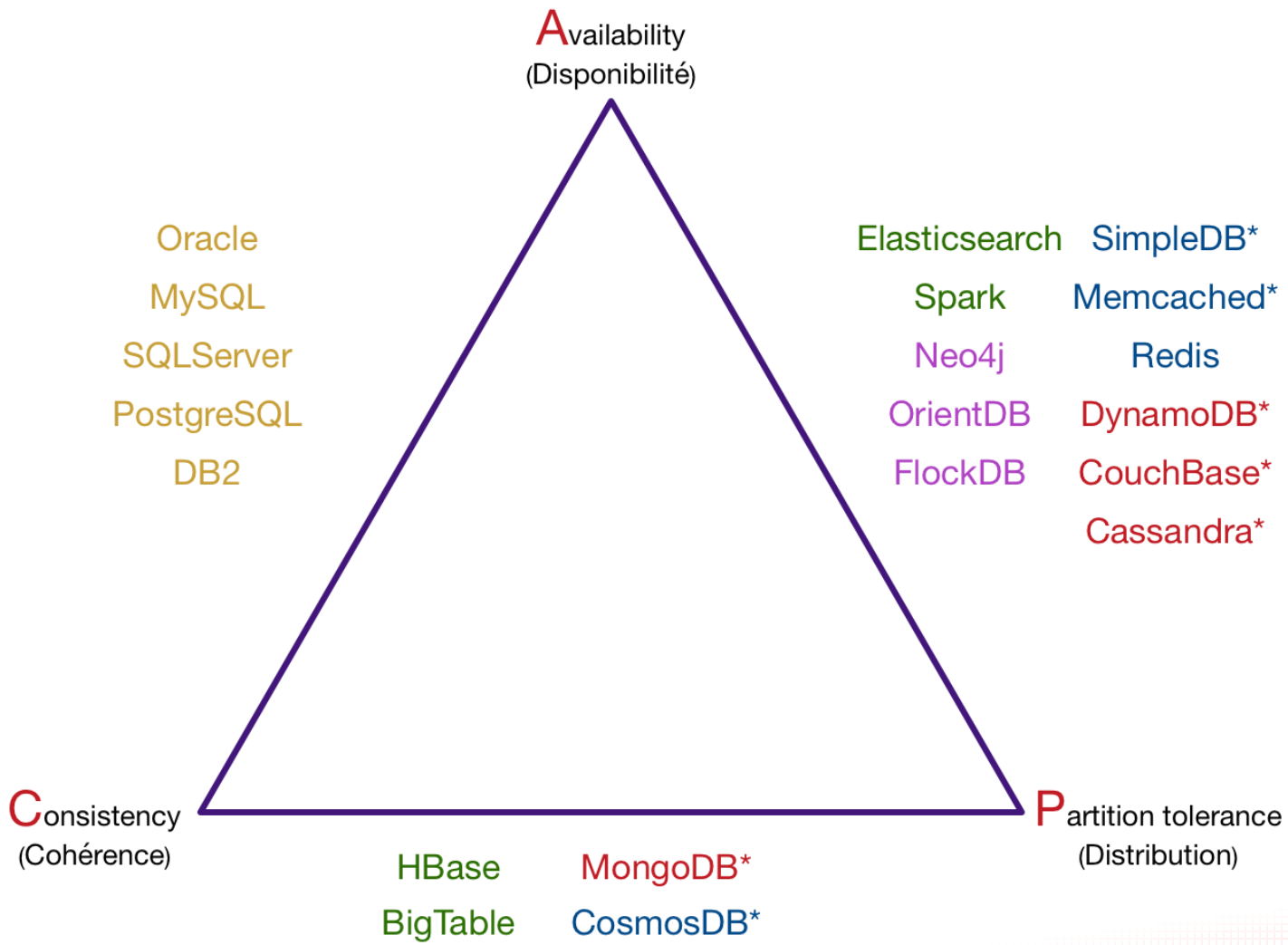


CAP

Modèle

Relationnel
Orienté Clé-Valeur
Orienté Colonne
Orienté Document
Orienté Graphe

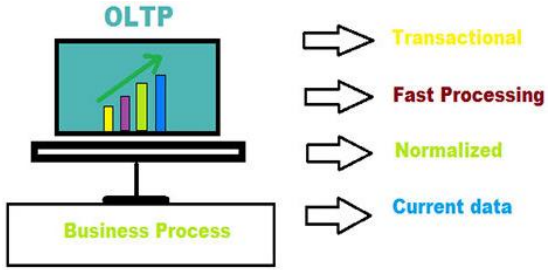
* Possibilité de changer
le mode de cohérence
Cohérence <-> Disponibilité

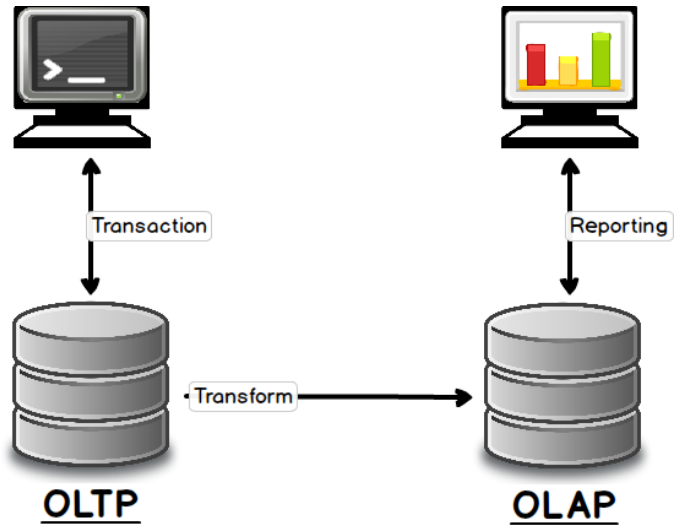


SQL / NoSQL : OLTP / OLAP

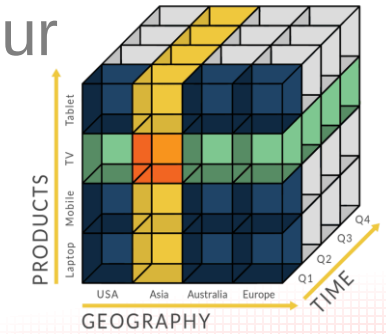
OLTP / OLAP

- OLTP : Online Transactional Processing
  - Beaucoup de mises-à-jour de résultats réduits
    - Ex : Factures clients
  - SQL



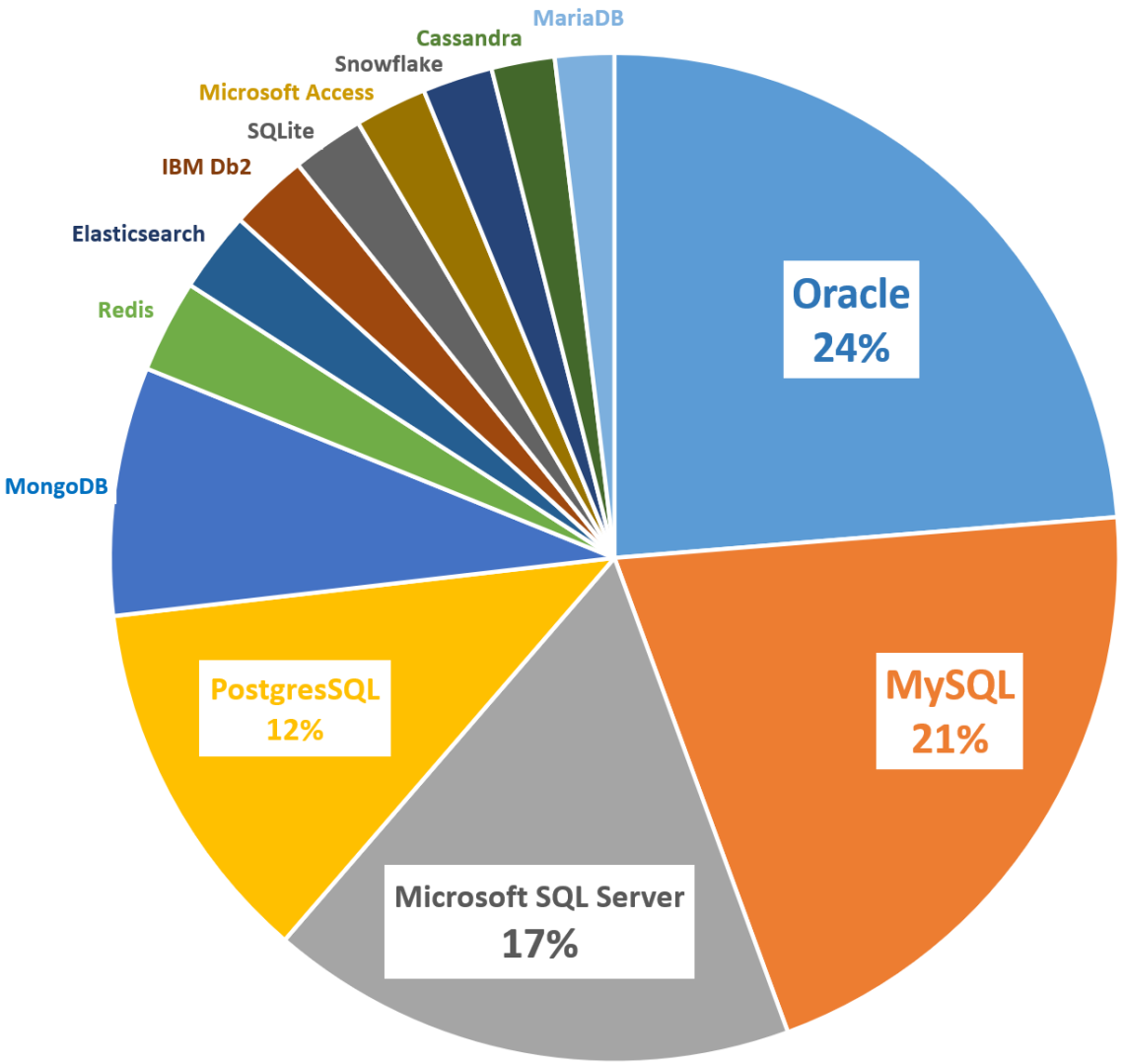


- OLAP : Online Analytical Processing
  - Beaucoup de lectures, peu de mises-à-jour
    - Ex : Business Intelligence, Big Data
  - NoSQL



On doit u moins 1.
En SQL on a la dispo et la cohérence, mais pas de distribution sur plusieurs serveurs

Popularité des SGBD



1	**Oracle**	**Relational**
2	MySQL	Relational
3	Microsoft SQL Server	Relational
4	PostgreSQL	Relational
5	**MongoDB**	**Document**
6	**Redis**	**Key-value**
7	Elasticsearch	Search engine
8	IBM Db2	Relational
9	SQLite	Relational
10	Microsoft Access	Relational
11	Snowflake	Relational
12	Cassandra	Wide column
13	MariaDB	Relational
14	Splunk	Search engine
15	Microsoft Azure SQL	Relational
16	Amazon DynamoDB	Multi-model info
17	Databricks	Multi-model info
18	Hive	Relational
19	Google BigQuery	Relational
20	Teradata	Relational
21	FileMaker	Relational
22	**Neo4j**	**Graph**
23	SAP HANA	Relational
24	Solr	Search engine
25	SAP Adaptive Server	Relational

Source: db-engines.com - 2023

Types de bdd NoSQL