Introduction aux bases de données **NoSQL**

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

elmouelhi.achref@gmail.com

Plan

NoSQL

Deux types de base de données

- **Base de données relationnelle** : données structurées et stockées dans des tables et gérées par le langage **SQL** (**S**tructured **Q**uery **L**anguage)
- **Base de données non relationnelle** : données avec plusieurs structures différentes, non stockées dans des tables et gérées par un langage de la mouvance (ou du mouvement) **NoSQL** (**N**ot **O**nly **S**QL)

NoSQL

La mouvance **NoSQL**

- ne désigne pas un groupe de personnes
- { idées + approches + technologies } qui ont émergé en réaction aux limitations des bases de données relationnelles traditionnelles
- terme popularisé en 2009

NoSQL

Base de données **NoSQL**

- permettent de stocker des données semi-structurées ou non structurées
- conçues pour s'adapter à une croissance importante du volume de données en répartissant les données sur plusieurs serveurs
- conçues pour garantir une haute disponibilité en répliquant les données sur plusieurs nœuds et en permettant la récupération en cas de panne

NoSQL

Base de données **NoSQL**

- permettent de stocker des données semi-structurées ou non structurées
- conçues pour s'adapter à une croissance importante du volume de données en répartissant les données sur plusieurs serveurs
- conçues pour garantir une haute disponibilité en répliquant les données sur plusieurs nœuds et en permettant la récupération en cas de panne

Remarque

Les bases de données **NoSQL** ne remplacent pas les bases de données relationnelles mais elles les complètent.

Principalement cinq types de base de données **NoSQL**

- **NoSQL orienté clé/valeur** :
  - Stockent les données sous forme de paires clé-valeur simples
  - Exemples : **Redis**, **Amazon DynamoDB**, **LevelDB** (**Google**), **Oracle NoSQL**
- **NoSQL orienté document** :
  - Stockent les données sous forme de documents semi-structurés (**JSON**, **BSON**, **XML**)
  - Exemples : **MongoDB**, **CouchDB**, **RethinkDB**
- **NoSQL orienté colonnes** :
  - Stockent les données de manière tabulaire,
  - Exemples : **BigTable** (**Google**), **Apache Cassandra**, **Hbase** (utilisé par **Hadoop**), **ClickHouse**
- **NoSQL orienté graphe** :
  - Stockent les données sous forme de graphes : avec des nœuds représentant des entités et des arêtes représentant les relations entre ces entités
  - Exemples : **Neo4j**, **Amazon Neptune**, **Blazegraph**
- **NoSQL multi-modèles** :
  - Prennent en charge plusieurs modèles de données au sein d'un même système
  - Exemples : **OrientDB** (Supporte document et graphe), **ArangoDB** (Supporte document, graphe, et clé-valeur)

NoSQL

Les propriétés **A.C.I.D** : (**A**tomicité, **C**ohérence, **I**solation, **D**urabilité)

forment un ensemble de propriétés dont l'objectif est de garantir qu'une transaction informatique sera exécutée en toute fiabilité

- **Atomicité** : une transaction se fait au complet ou pas du tout.
- **Cohérence** : chaque transaction amènera la base d'un état valide à un autre état valide.
- **Isolation** : Toute transaction doit s'exécuter comme si elle était la seule sur le système. Ses modifications ne sont accessibles que lorsque la transaction a été validée.
- **Durabilité** : Une fois la transaction validée, elle demeure enregistrée même à la suite d'une panne ou autre.

NoSQL

Remarques

- Les propriétés **A.C.I.D** sont garanties par les SGBD relationnels.
- Les bases de données **NoSQL** sont incompatibles avec les propriétés **A.C.I.D**.

NoSQL

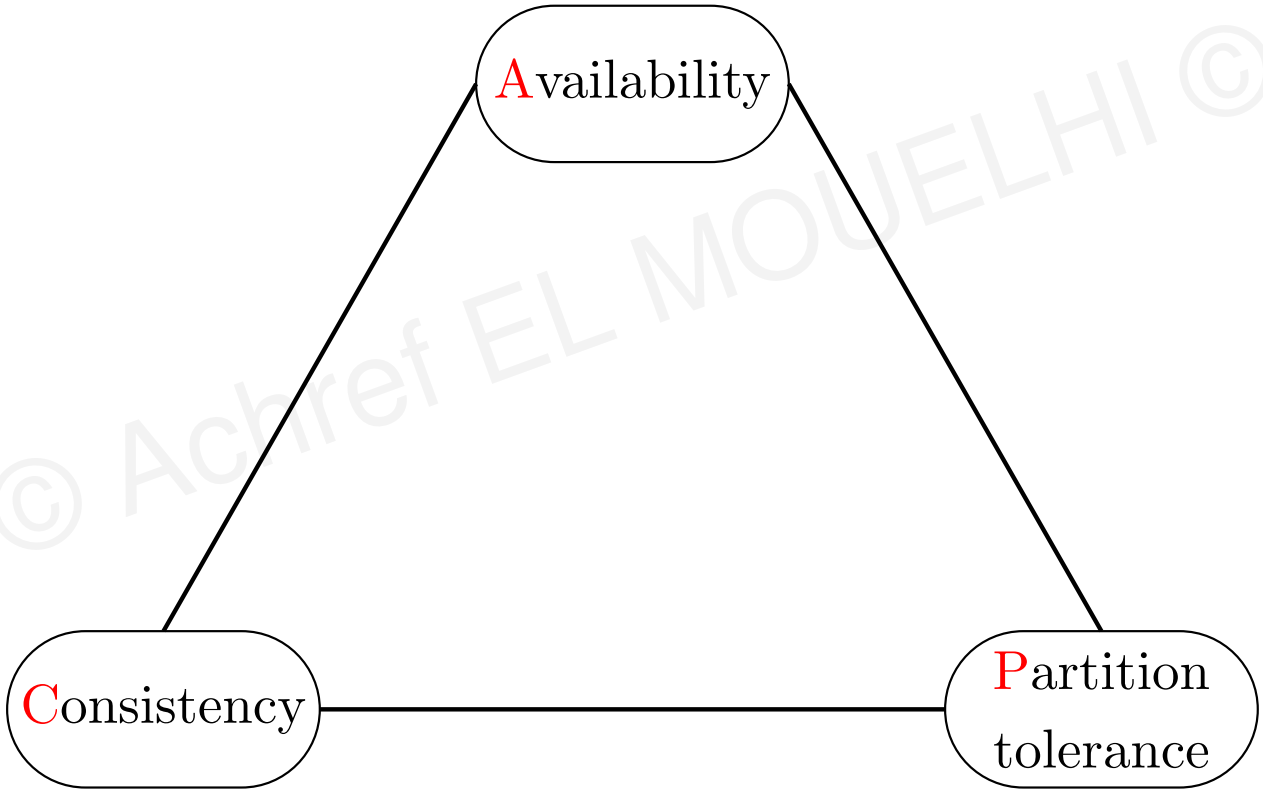
Caractérisation des BD : théorème de **CAP** [Bewer 2000]

Quelle que soit la base de données (relationnelle ou **NoSQL**), on ne peut respecter qu'au plus que 2 propriétés parmi les trois : la cohérence, la disponibilité et la distribution.

- La cohérence (**C**onsistency) : considérant une donnée d, à un instant t, tous les utilisateurs récupère une seule et même valeur de la donnée d
- La disponibilité (**A**vailability) : on doit retourner une réponse pour toute requête reçue
- La distribution (**P**artition Tolerance) : Quel que soit le nombre de serveurs, toute requête doit fournir un résultat correct

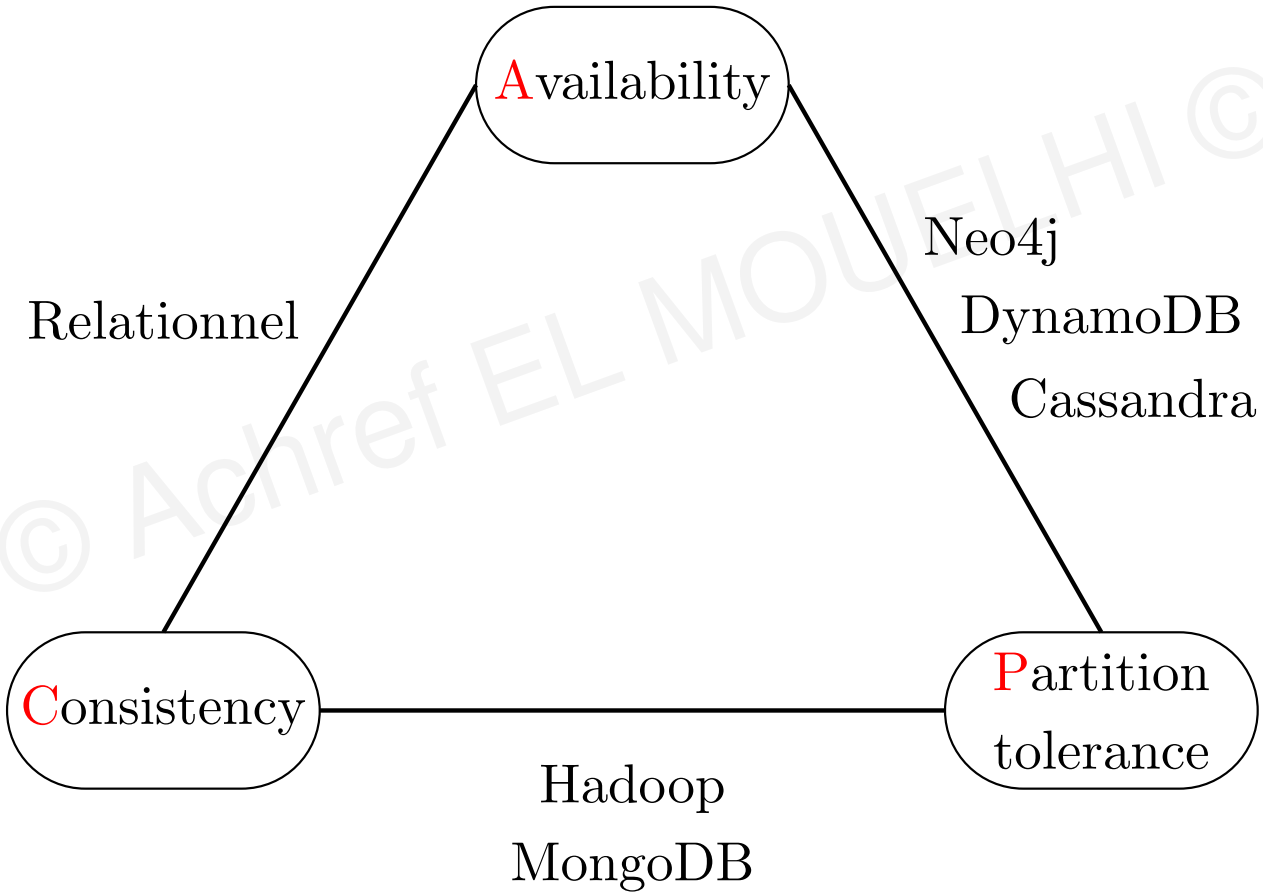
NoSQL

Schématisation du théorème de CAP

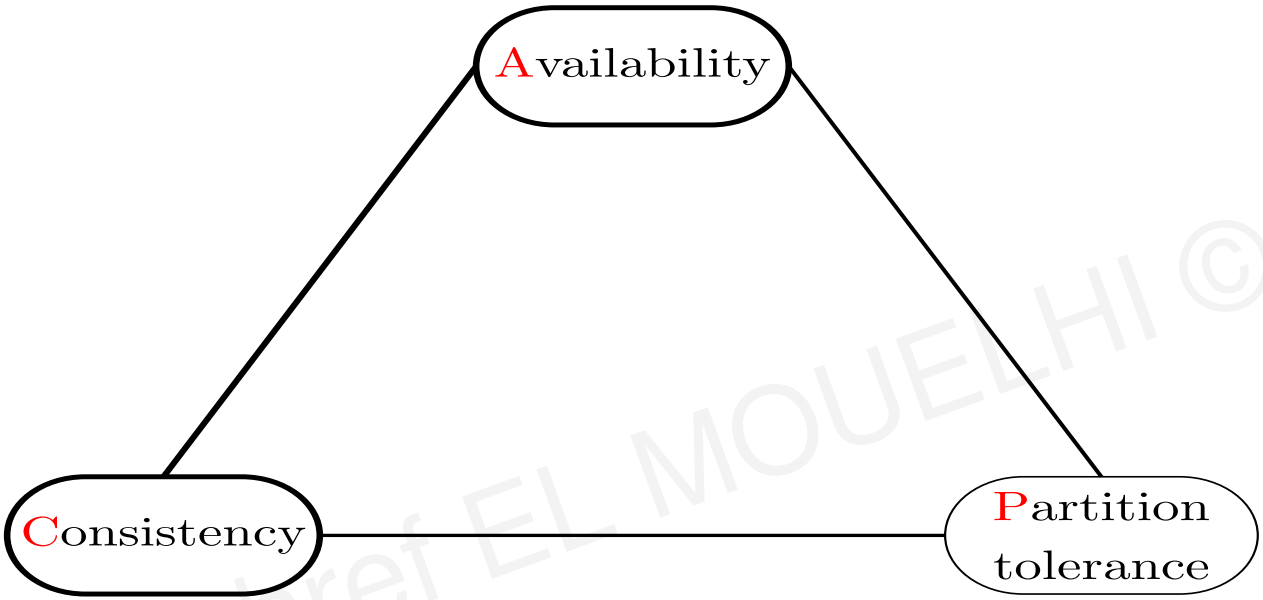


NoSQL

Répartition des BD



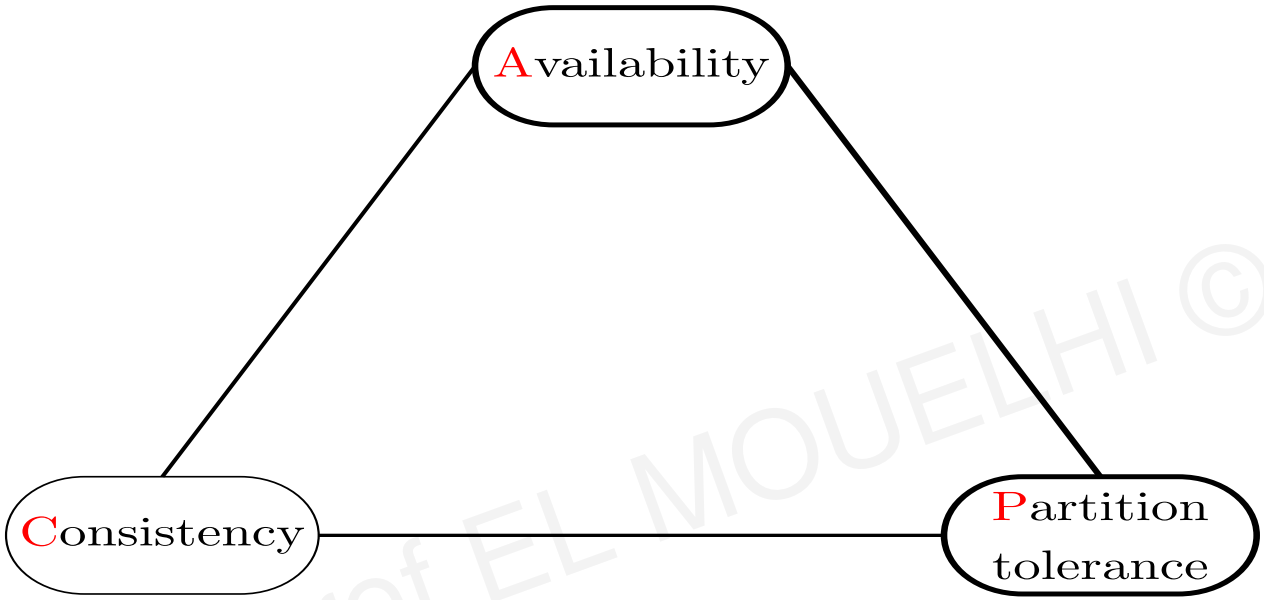
NoSQL



Le couple CA (Consistency-Availability)

On garantit, après chaque modification, la cohérence et la disponibilité des données. Mais, il ne faut pas que la base de données soit distribuée.

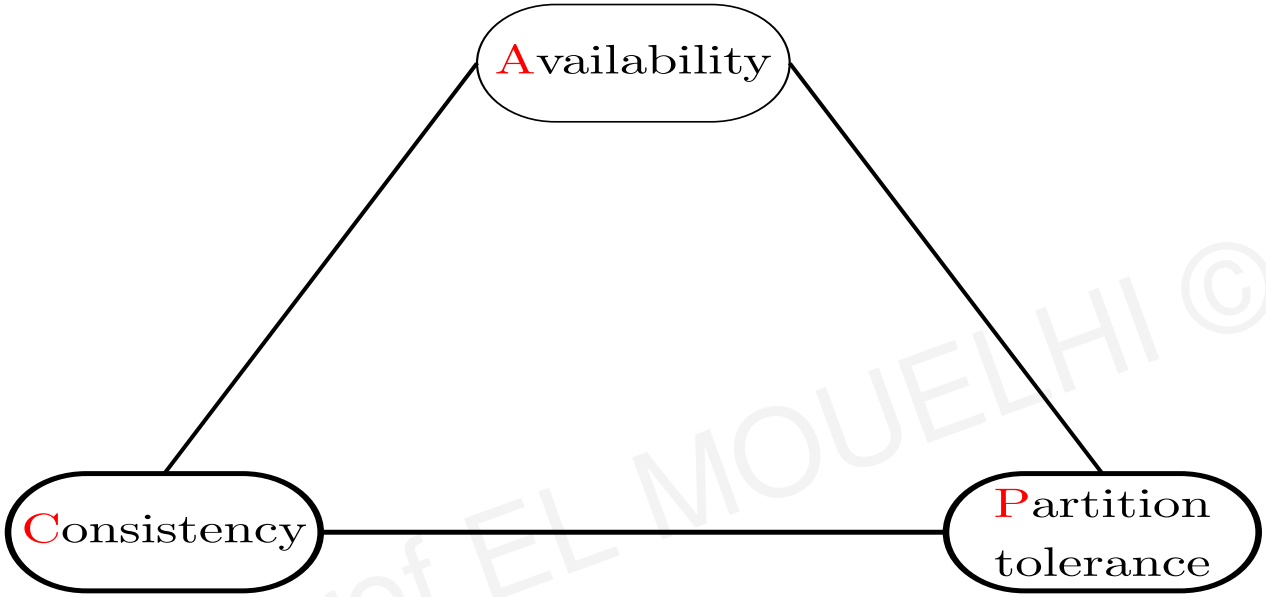
NoSQL



Le couple AP (Availability-Partition Tolerance)

On garantit, après chaque modification, la disponibilité des données sur les différentes partitions. Mais, on ne garantit jamais la cohérence à 100 % (il faut du temps pour mettre à jour toutes les partitions).

NoSQL



Le couple CP (Consistency-Partition Tolerance)

On garantit, après chaque modification, la cohérence des données sur les partitions. Mais, on ne garantit jamais la disponibilité (il faut du temps pour que la donnée soit la même sur toutes les partitions).