Big Data : technologies

Les Technologies du Big Data

> Ensembles des techniques et strategies permettant de répondre aux contraintes des grands **V** du Big Data.

Parradigmes du Big Data

Pour repondre aux besoins du Big Data, des sytemes permettant de depasser les limites des systemes traditionnels ont été developpés.

- Architectures scallables
- Solutions de stockages pour les données non structurées

- Architectures distribuables et massivement parallèles
- Algorithmes distibués et parallèles

Scalabilité

> Adapter la taille et/ou la puissance d'un système informatique pour repondre aux changements de la charge de travail.

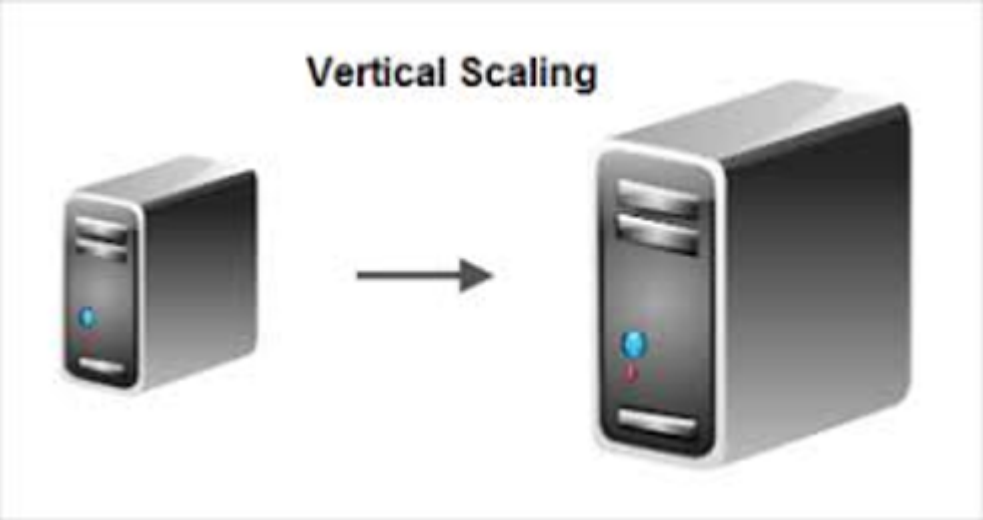
Il existe deux types de scallabilité :

- **Verticale** : Augmentation des ressources internes.
- **Horizontale** : Augmentation des ressources externes.

Scalabilité verticale

> Augmentation de la puissance (processeur, RAM, stockage) d'un système informatique.

- C'est la solution la plus simple et la plus rapide à mettre en oeuvre.

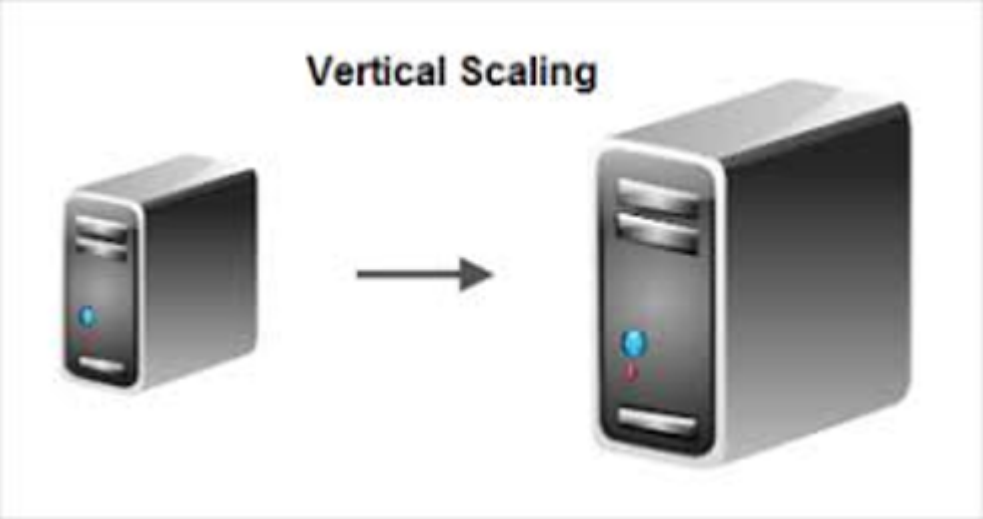


- Fréquemment utilisée dans les systemes traditionnels.

Limites de la scalabilité verticale

L'augementation de la puissance d'un système informatique est limitée par :

- Cout du materiels (augmentation exponentielle)

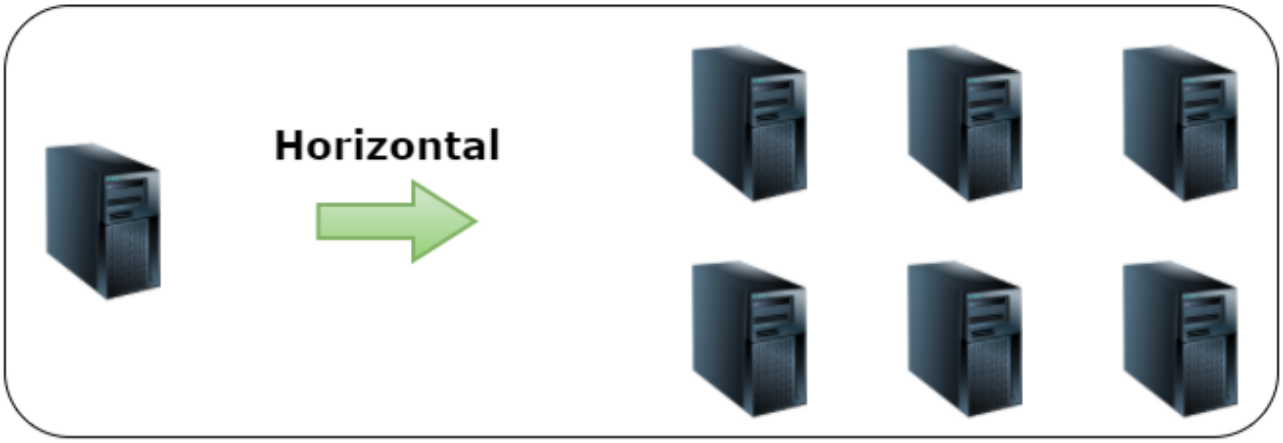


- Faible adaptabilité aux changements de la charge de travail
- Problemes en cas de pannes (Single point of failure)

Scalabilité horizontale

> Augmentation du nombre de machine de faible puissance pour augmenter la puissance globale.

- C'est la solution économique la plus adaptée.



- Pseudo-linéarité des performances.
- Decoupages et repliations des données.

Limites de la scalabilité horizontale

L'augementation du nombre de machine de faible puissance est limitée par :

- Augementation exponentielle des echanges

- Architectures reseaux complexes (cluster)
- Syncronisation des données

Modèles de stockage

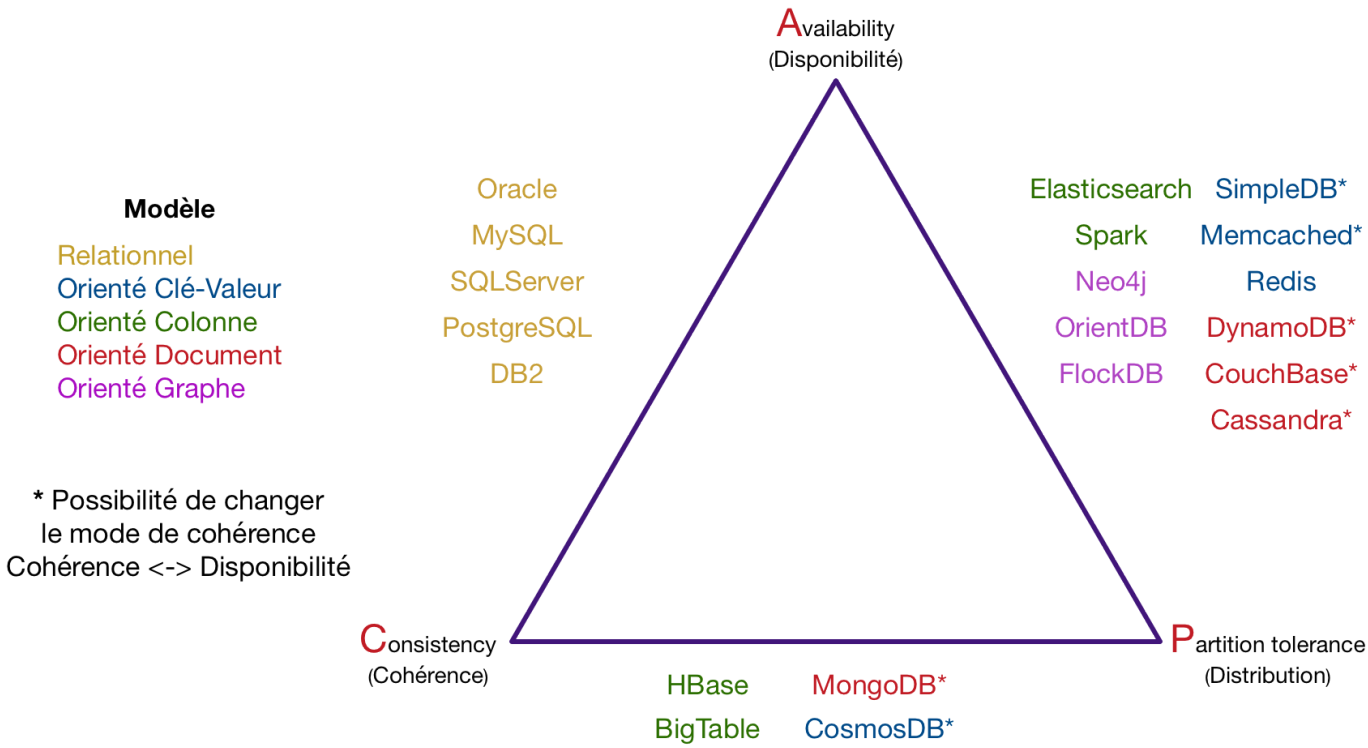
> L'avenement du big data a necessité l'adaptation des systemes de stockage pour s'adapter à la quantitées et aux types de données.

- Theoreme de CAP et big data
- Modeles NoSQL

Theoreme de CAP

> Tous systemes ne peut garantir que 2 des 3 propriétés suivantes :

- **C**onsistency (Cohérence) : accès aux mêmes données à tous moments.
- **A**vailability (Disponibilité) : accès à la lecture et l'ecriture à tous moments.
- **P**artition Tolerance (Tolérance aux pannes)



Stockage traditionnel

Les sytemes de stockage traditionnels utilisent des bases de données relationnelles.

- Données structurées et formatées

- Requetes et manipulations simples
- Transaction qui respectent les principes ACID
- Large pannel de solutions (longevité des technologies)

Limites des bases de données relationnelles

- Manque d'adaptabilité des schemas de données

- Limitées aux données strucuturées (<20%)
- Depenses massives en temps et ressources en cas de modifications de la structure des données.

Avenement du NoSQL

- Developper pour pallier aux limitations des bases de données relationnelles.
- Prise en charge des données non structurées.
- Adaptabilité des schemas de données.
- Abandon des principes ACID au profit des principes BASE.

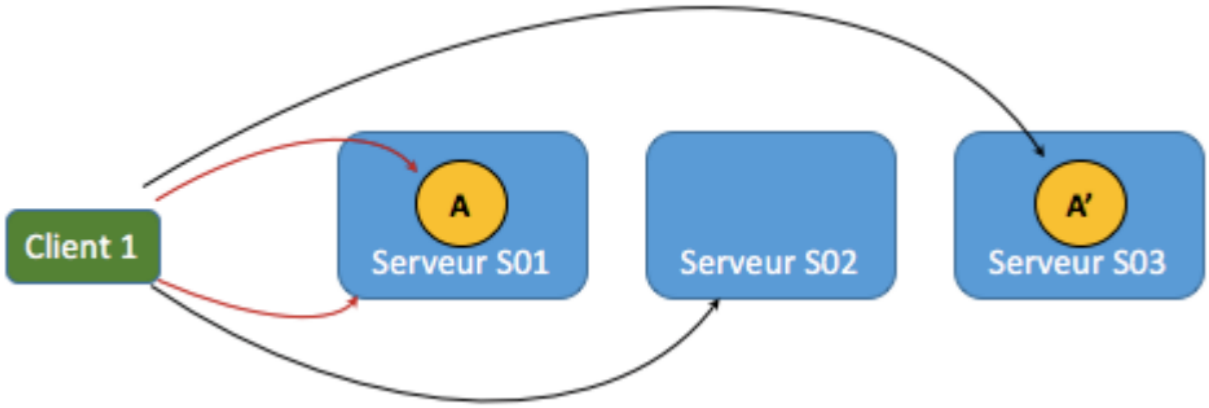
Les principes BASE

- **B**asically **A**vailable
- **S**oft-State
- **E**ventual Consistency

Basically Available

> Notion de disponibilité des données à tout moment.

- Le systeme doit pouvoir repondre aux requetes de tout utilisateur meme en cas de pannes.



- Les requetes peuvent etre obsolètes. (pert Isolment ACID).

Soft-State

> Notion que les données sont dans un flux d'utilisation constante.

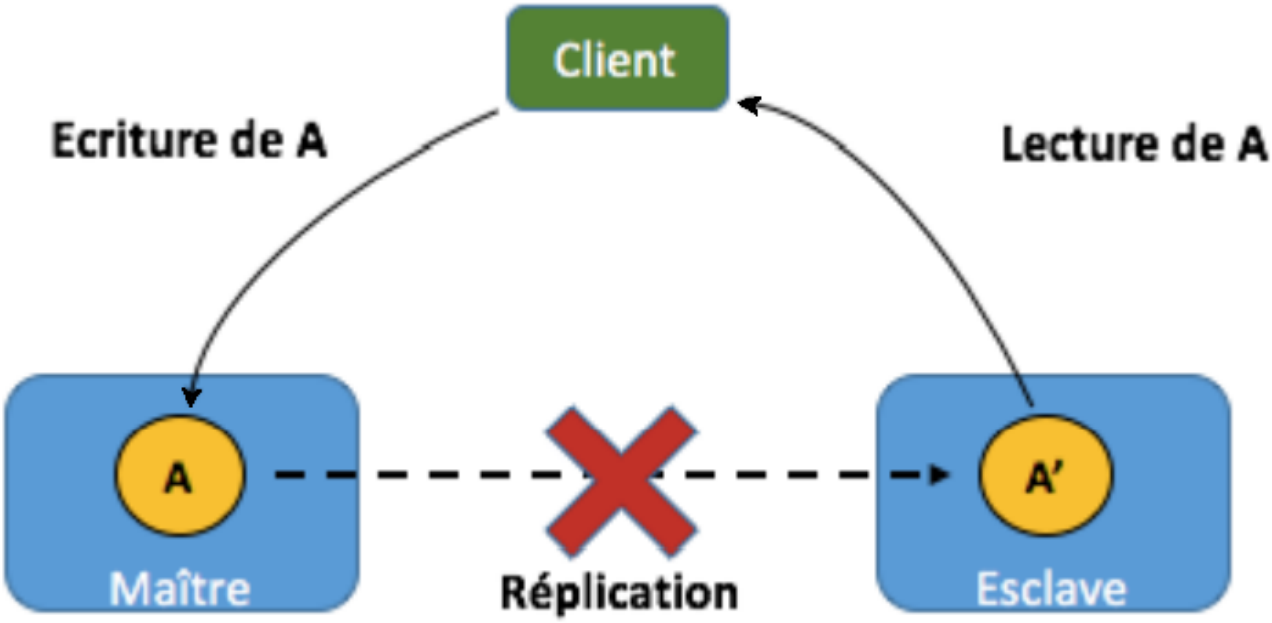
- Les données peuvent etre utilisées par plusieurs utilisateurs en meme temps.

- La coherence n'est pas garanti (pert Cohérence ACID).

Eventual Consistency

> Notion qu'au bout d'un certains temps le systeme sera cohérent.

- La synchronisation des données est faite en arriere plan.

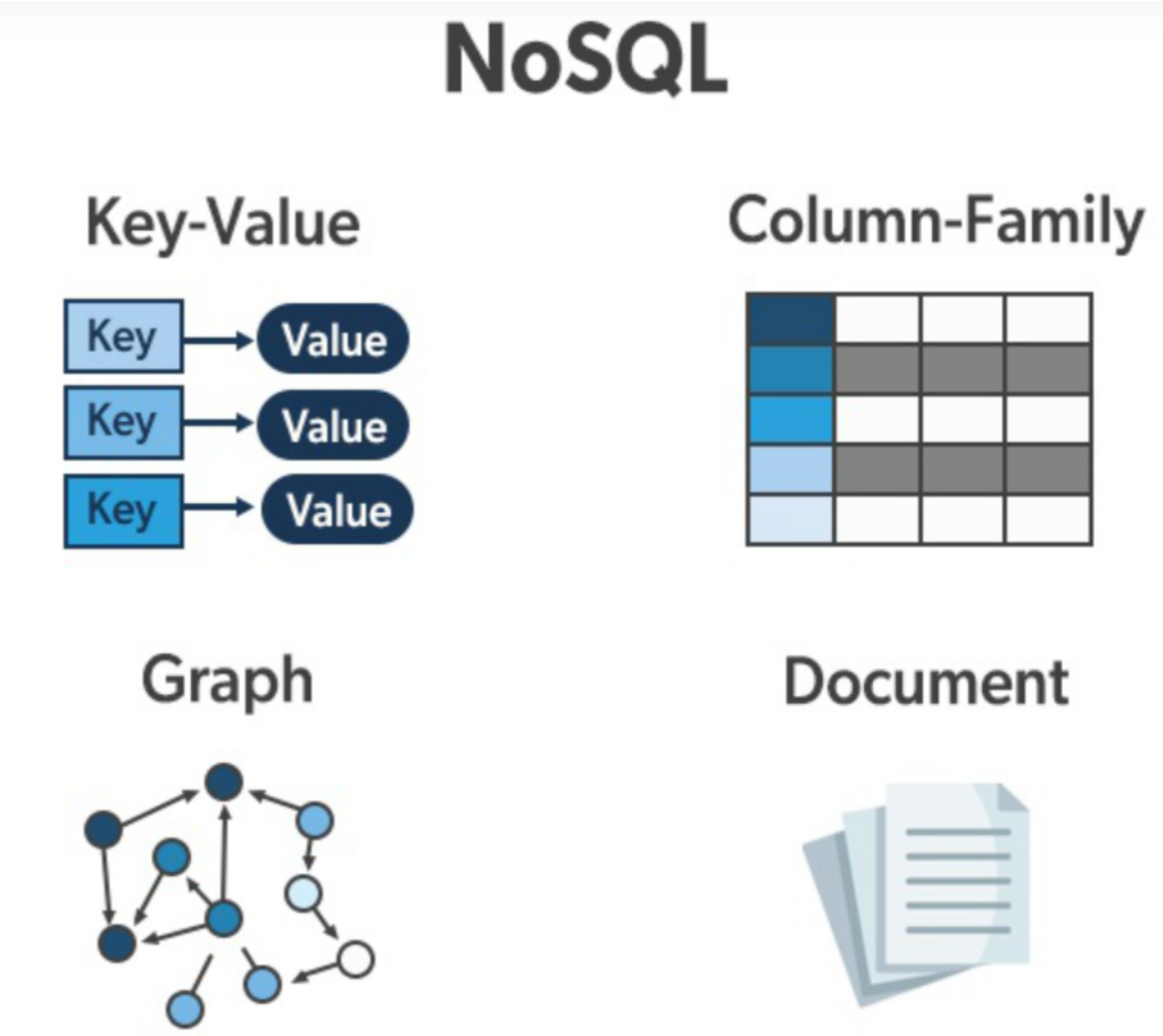


- Les données peuvent etre obsolètes (pert Cohérence ACID).

Les modèles NoSQL

A partir de ces principes, quatres modèles de stockage NoSql ont été developpés :

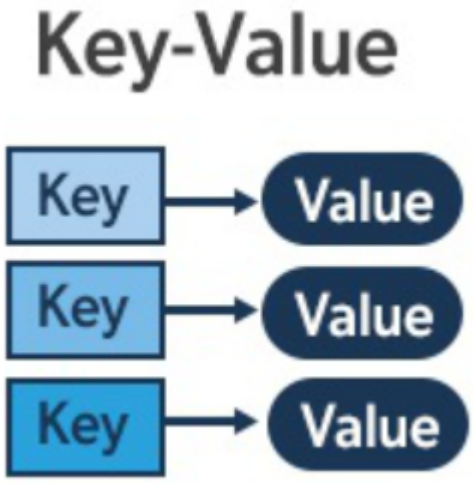
- Clé/valeur



- Colonne
- Document
- Graphe

Modèle clé/valeur

> Stockage sous forme d'un dictionnaire clé/valeur

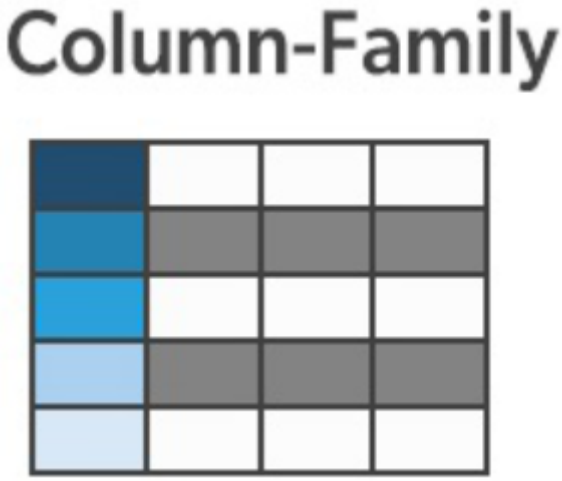


- Clé : chaine de caractères unique
- Valeur : typage au besoin

ex : redis (StackOverFlow),riak (GitHub), Memcached (wikipedia), voldemort (LinkedIn).

Modèle Colonne

> Stockage sous forme de table dénormalisée



- systeme proche des bases de données relationnelles.
- tableau de clé/valeur groupable en famille (+/- table).

ex: Cassandra (Nasa), HBase (Facebook, xiaomi), BigTable (GCP).

Modèle Document

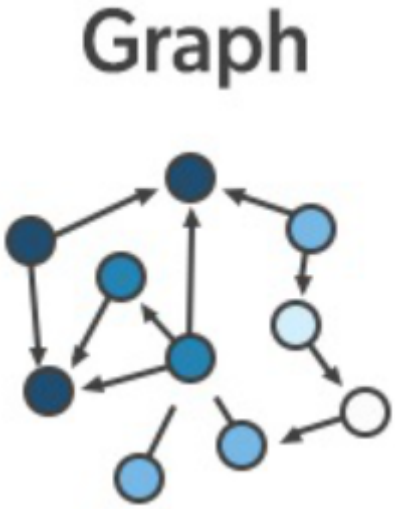
> Stockage sous forme de document

- Object JSON avec un id et des propriétés clé/valeur.
- valeur peuvent etre d'autres documents.

ex: MongoDB (SEGA, ThermoFisher Scientific), CouchDB (CERN)

Modèle Graphe

> Stockage sous forme de graphique relationnelle

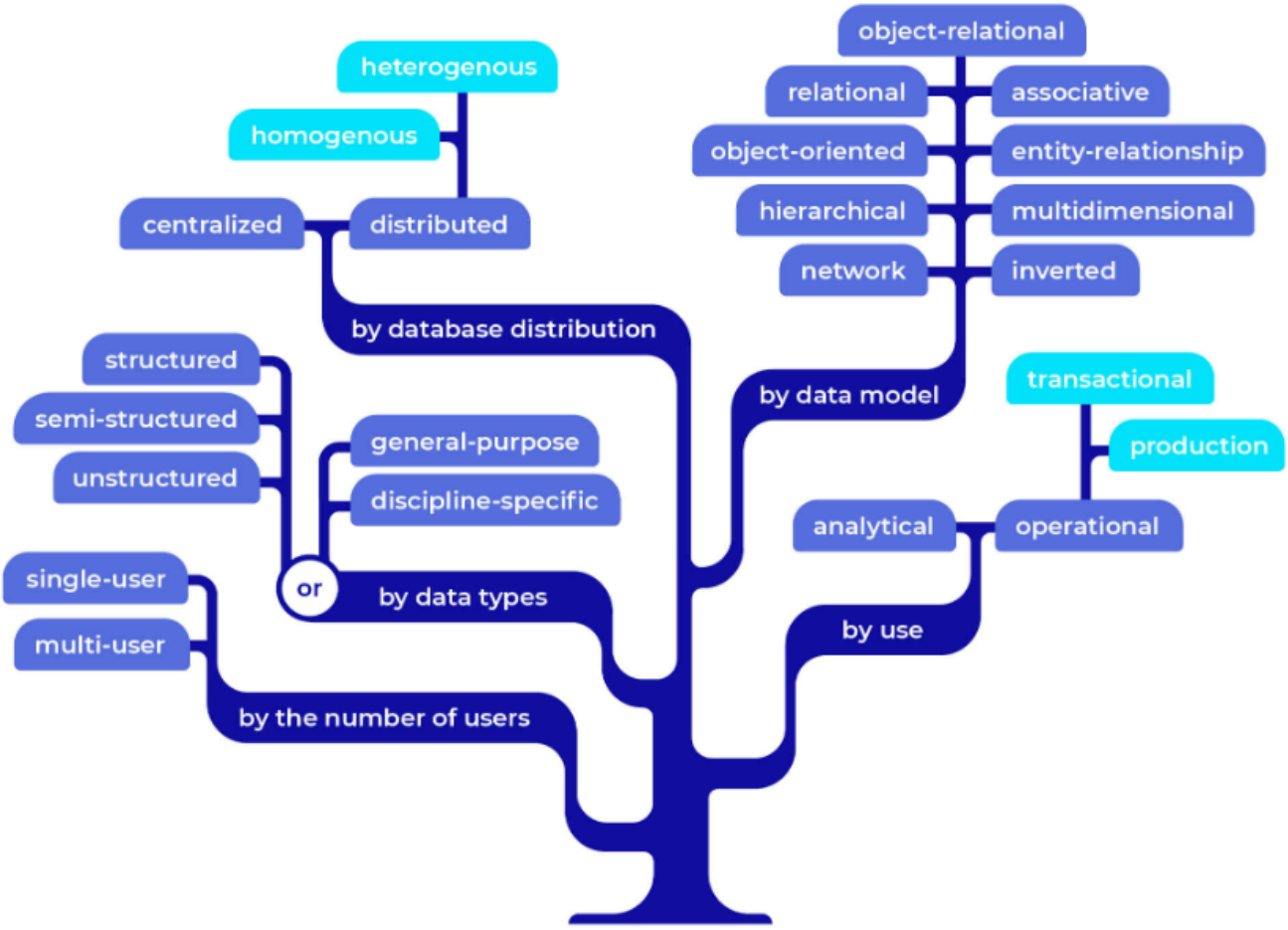


- Noeuds de données de type document clé/valeur
- ARC : relation orientée entre les noeuds proteur de propriétées.

ex: Neo4j (Orange, Airbus), OrientDB, Titan

Choix du modèle de stockage

Critere variable selon besoin



Exercice

Exercice 01

Architectures et algorithmes