Introduction NoSQL

Licence Pro MDS parcours IDSBD

2020-2021

Raquel Urena

raquel.urena@univ-amu.fr

Planning du Cour

- Introduction NoSQL
- NoSQL modèle Document
- NoSQL modèle Graphe

Objective 1er séance

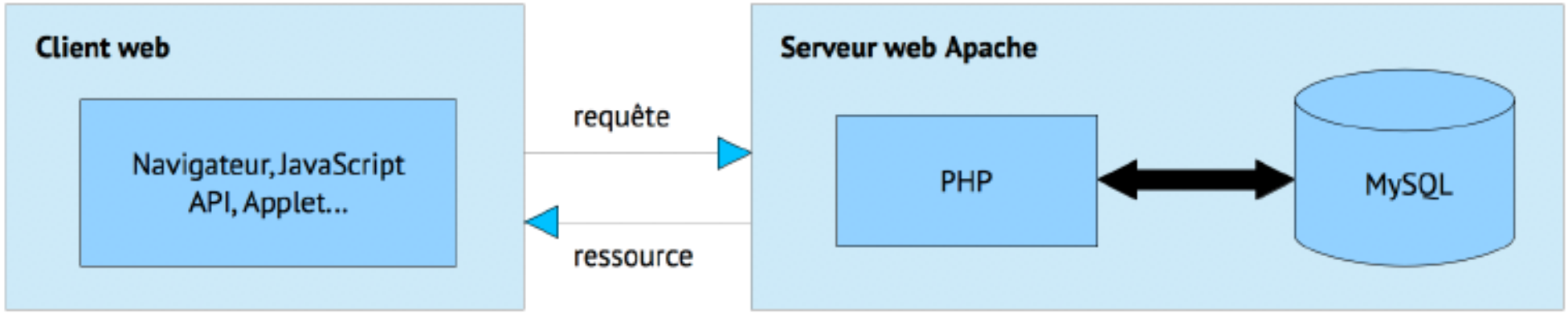
- Reconnaitre la différence entre BBDD relationnelle et BBDD NoSQL
- Utilité général de BBDD NoSQL
- Types de BD NoSQL
- Applications réels de BD NoSQl

Introduction

- Une base de données est un ensemble de données appartenant au même contexte et systématiquement stockées pour une utilisation ultérieure (chiffres, dates, mots…) .

- Le système de gestion de base de données est un program qui manipule la structure de la base de données et dirige l'accès aux données qui y sont stockées.

Architecture classique (Stack LAMP)

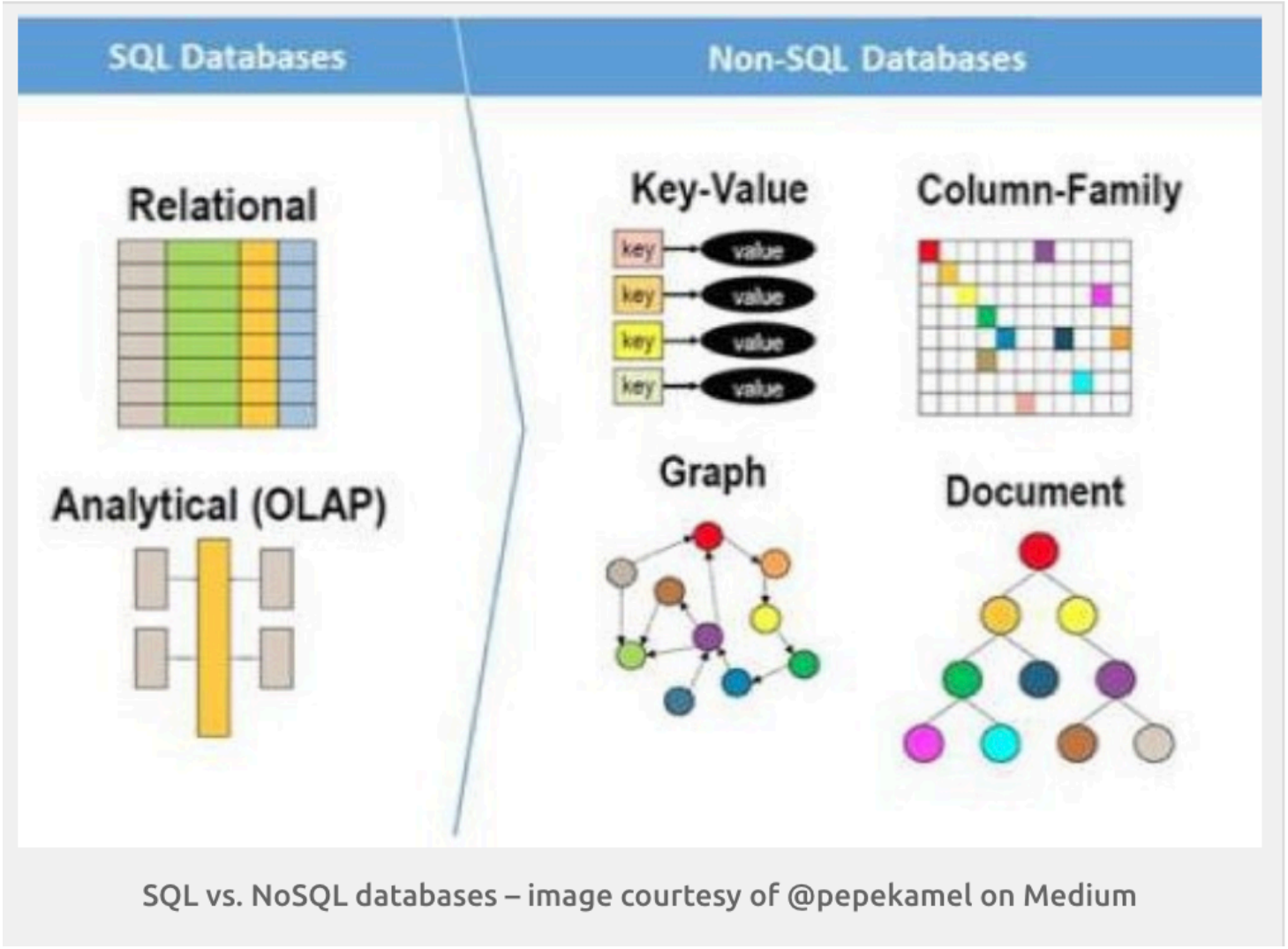



```
SELECT * FROM nom_table
WHERE condition
GROUP BY champ1, champ2
HAVING groupe condition
ORDER BY champ
LIMIT limite, taille;
```

SQL vs NoSQL

- Base de données relationnelle :
  - donnes structures
  - stockes dans des tables
  - langage SQL (Structured Query Language)

- Base de données NoSQL:
  - donnes avec plusieurs structures diferentes.
  - Optimisés pour les applications qui nécessitent un volume de données important, une faible latence et des modèles de données flexibles.



SQL vs. NoSQL databases – image courtesy of @pepekamel on Medium

SQL vs NoSQL

- Les bases de données NoSQL ont été inventées afin de résoudre des problèmes insolubles par les bases de données relationnelle et non pas pour les remplacer.

BBDD NoSQL	BBDD Relationnelle
stockage de masse	stockage fiable
données diverses	données formatées
scalabilité horizontale	scalabilité verticale

Quand utiliser NoSQL?

- **Les bases de données relationnelles** nous imposent de recourir à des schémas de données.
- il faut segmenter ces données en unités atomiques, puis les organiser dans des tableaux sous forme de **colonnes et de lignes.** (Nom, prènom, date de naissance, adresse …)
- si les données sont difficilement stockables sous forme tabulaire ou interrogeables en langage SQL, regardez donc du côté du NoSQL. (images, différents types des Structures des donnes)

Quand utiliser NoSQL?

- Service réparti sur plusieurs continents ?
- Accès concurrent de plusieurs centaines de milliers de personnes ?
- BBDD n'échelonne pas votre trafic à un coût acceptable
- La taille de votre schéma de données a augmenté de manière disproportionnée.
- Beaucoup de données temporaires qui ne correspondent pas au magasin de données principal (paniers d'achat, personnalisation des portails) ?

Quand utiliser NoSQL?

- Le jeu de données contient de grandes quantités de texte, d'images et d'objets BLOB
- Consultations contre des données qui n'impliquent pas de simples relations hiérarchiques;
  - recommandations ou demandes de renseignements commerciaux.
  - Exemple: "toutes les personnes d'un réseau social qui n'ont pas acheté de livre cette année et qui ont été retirées des personnes qui l'ont" –

Propriétés base de données traditionnelles

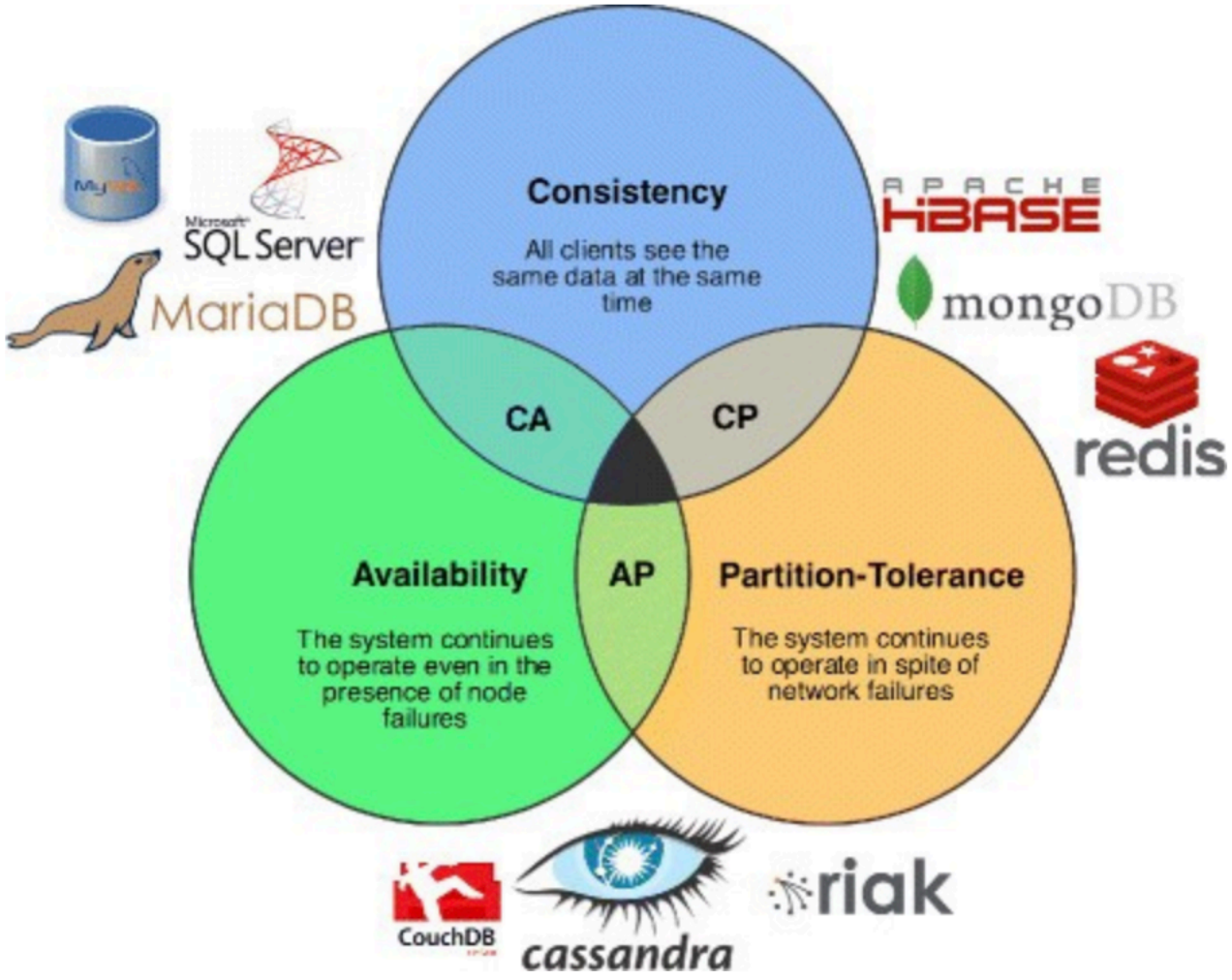
- Afin de garantir l'intégrité des données, la plupart des systèmes de base de données classiques reposent **sur les transactions** :
- Ces caractéristiques transactionnelles (aussi connues sous l'acronyme ACID):
  - **Atomicité:** Tout ou rien. Soit l'opération se fait en entier, soit elle ne se fait pas du tout.
  - **Cohérence: E**tat valide avant et après l'opération
  - **Isolation:** Les modifications d'une transaction ne sont visibles/modifiables que quand celle-ci a été validée
  - **Durabilité:** Une fois la transaction validée, l'état de la base est permanent (non affecté par les pannes ou autre)
- Les propriétés ACID sont garanties par les SGBD relationnels
-

BASE pour NoSQL

- **B***asically* ***A****vailable* : quelle que soit la charge de la base de données (données/requêtes), le système garantie un taux de disponibilité de la donnée
- **S***oft-state* : La base peut changer lors des mises à jour ou lors d'ajout/suppression de serveurs. La base NoSQL n'a pas à être cohérente à tout instant
- **E***ventually consistent* : À terme, la base atteindra un état cohérent

Théorème de CAP

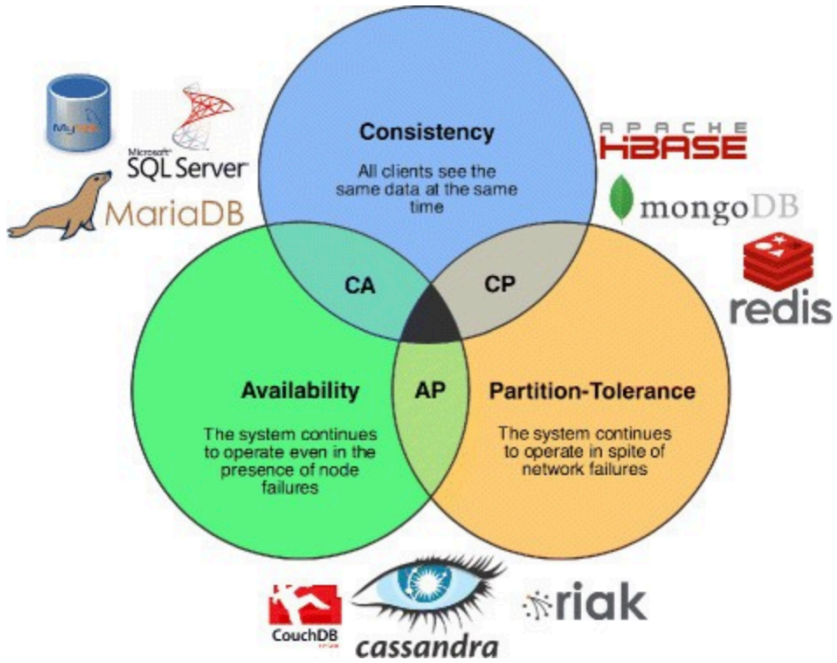
- Dans toute base de données, vous ne pouvez respecter au plus que 2 propriétés parmi
  - la *cohérence*, (consistency) Une donnée n'a qu'un seul état visible quel que soit le nombre de réplicas.
  - la *disponibilité (availability)* Tant que le système tourne (distribué ou non), la donnée doit être disponible.
  - la *distribution (partition tolerance):*Quel que soit le nombre de serveurs, toute requête doit fournir un résultat correct.



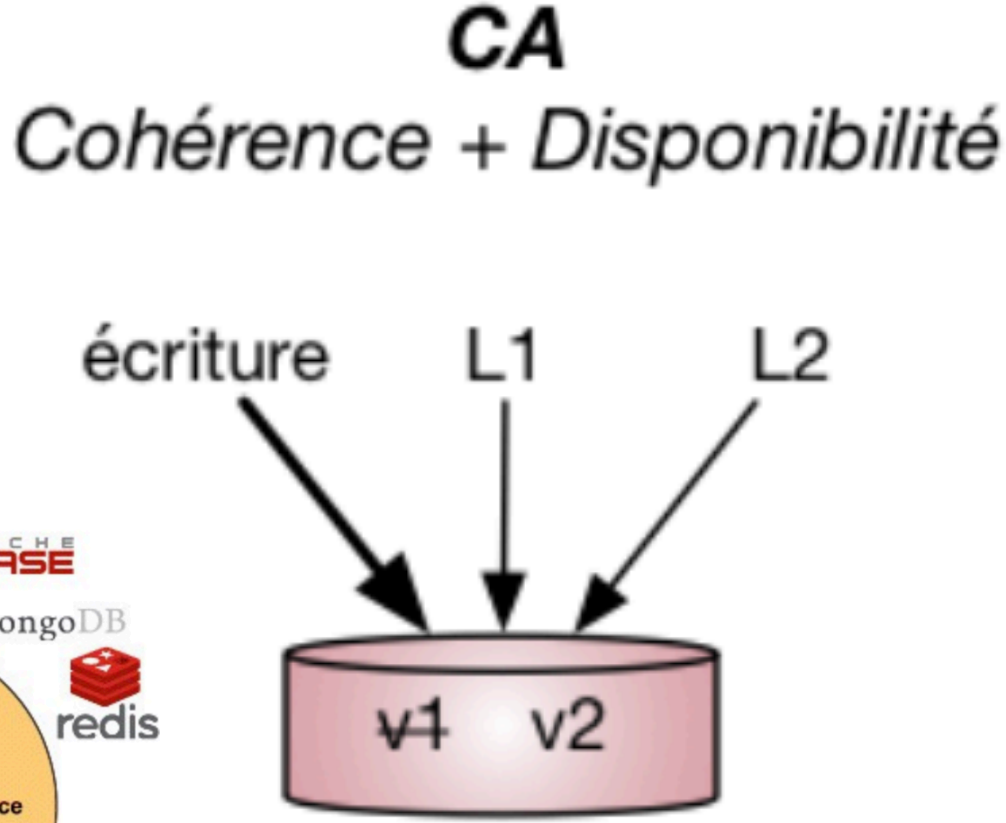
CAP Theorem

Théorème de CAP

- **CA (Consistency-Availability),** lors d'opérations concurrentes sur une même donnée, les requêtes L1 et L2 retournent la nouvelle version (v2) et sans délai d'attente.

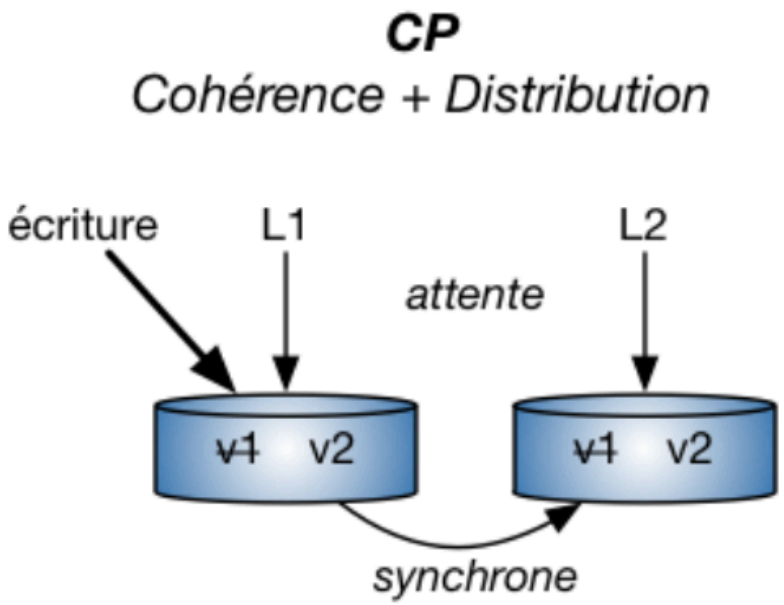


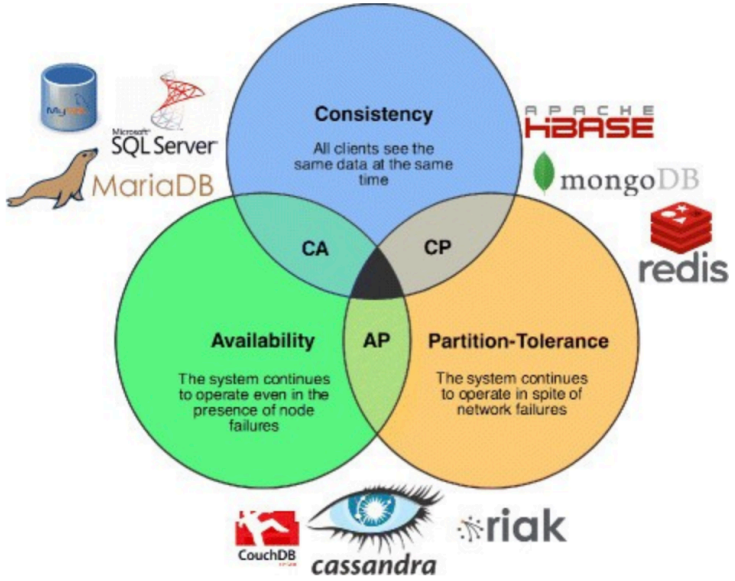
CAP Theorem



Théorème de CAP

- **CP (Consistency-Partition Tolerance)** les données sont distribuées sur plusieurs serveurs en **garantissant la tolérance aux pannes (réplication).**
- Nécessaire de vérifier **la cohérence des données** en garantissant la valeur retournée malgré des mises à jour concurrentielles.
- La gestion de cette cohérence nécessite un **protocole de synchronisation** des réplicas, introduisant des **délais de latence** dans les temps de réponse (L1 et L2 attendent la synchronisation pour voir v2.

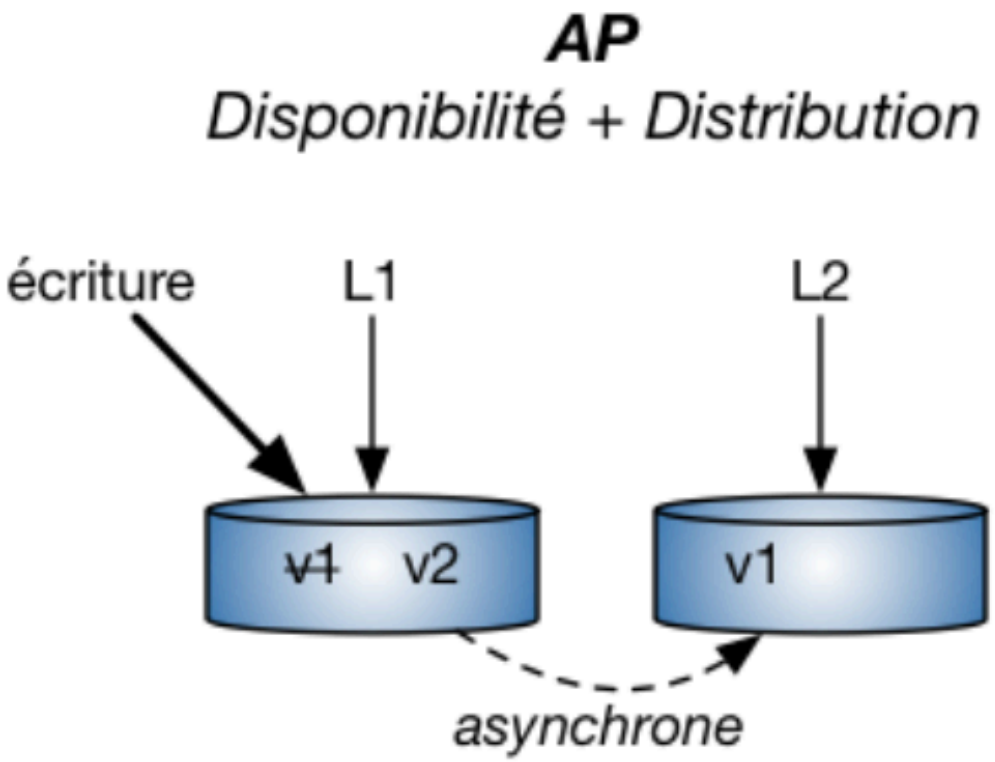


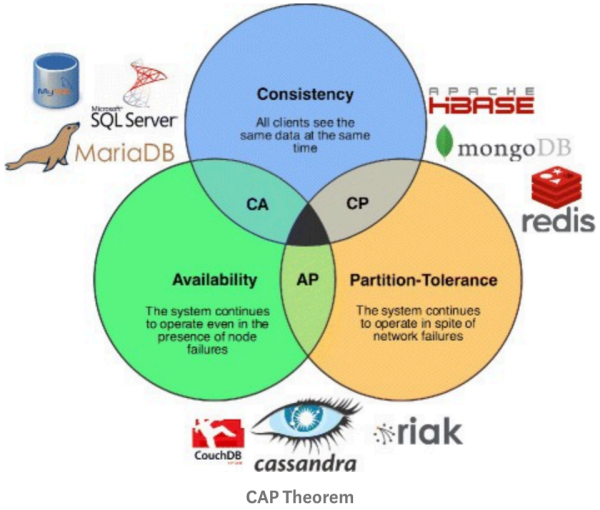


CAP Theorem

Théorème de CAP

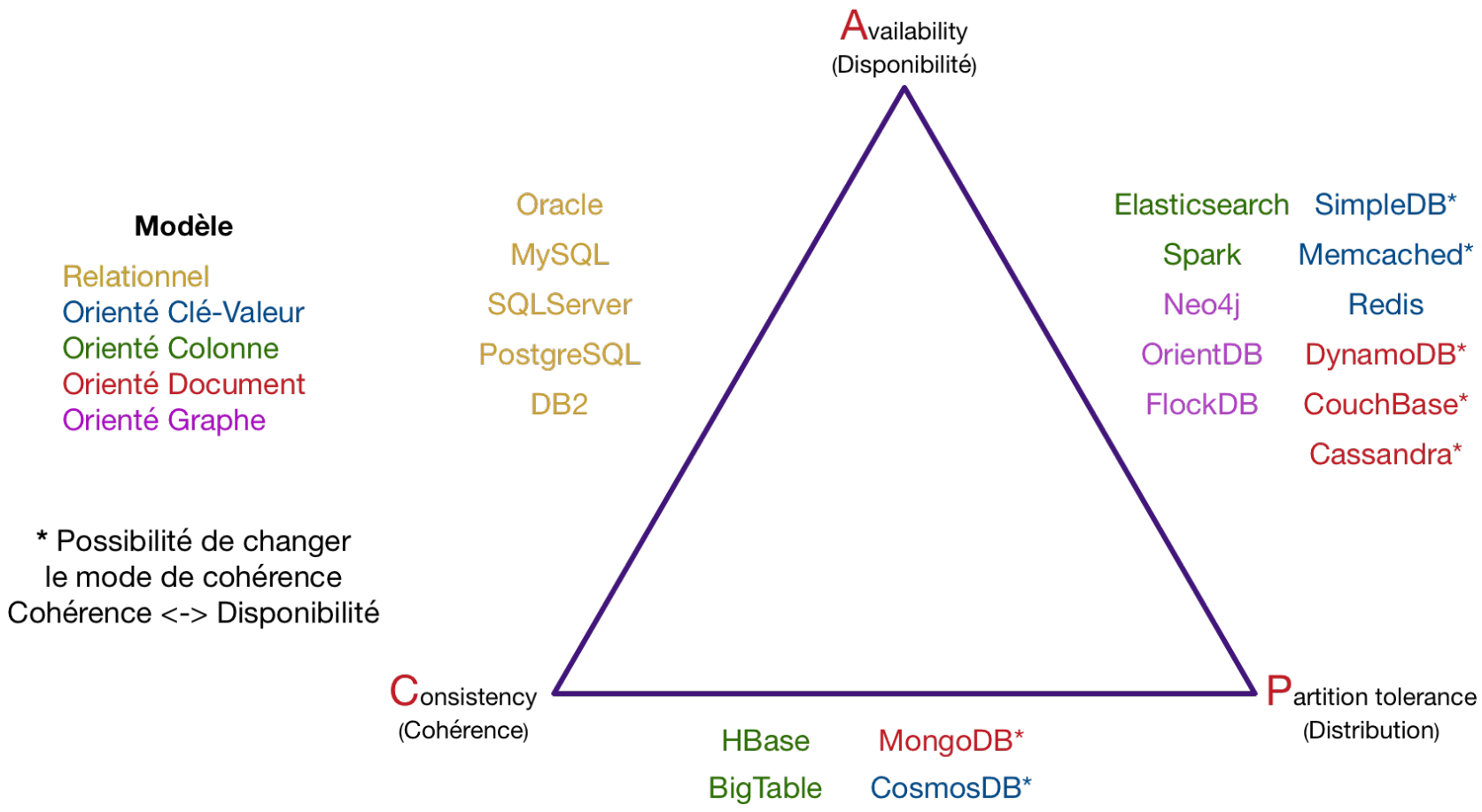
- **AP (Availability-Partition Tolerance)** s'intéresse à fournir un **temps de réponse rapide tout en distribuant les données et les réplicas**. De fait, les mises à jour sont asynchrones sur le réseau, et la donnée est "*Eventually Consistent*" (L1 voit la version v2, tandis que L2 voit la version v1).





CAP Theorem

Le triangle CAP

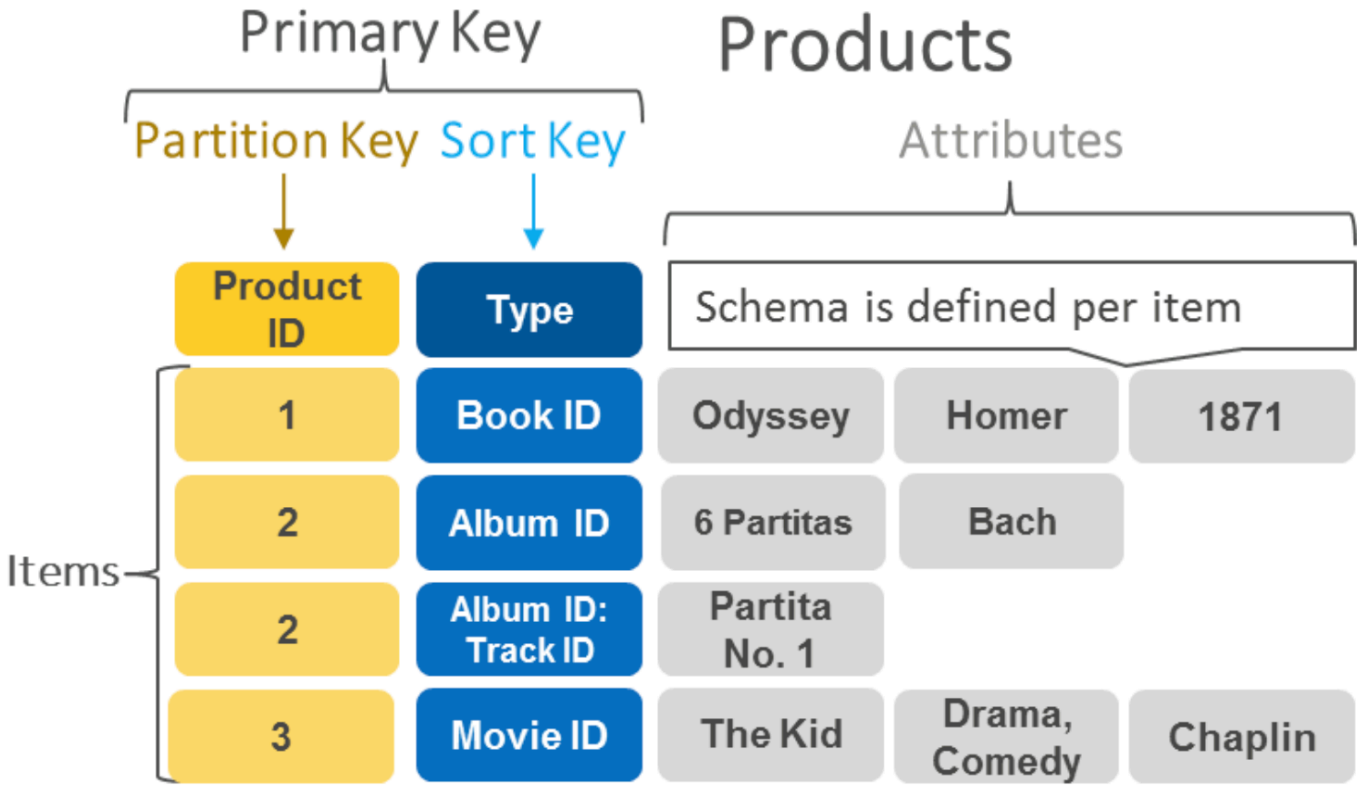


Base de donnes NoSQL

- Les bases de données NoSQL sont largement reconnues pour leur facilité de développement, leur fonctionnalité et leur performance à l'échelle.
- Elles utilisent une variété de modèles de données:
  - le document,
  - le graphe,
  - la valeur clé,
  - Colonnes.

Types de Bases de donnes NoSQL

- **Valeur-clé** : stocke les données sous forme de paires clé-valeur dans lesquelles une clé sert d'identifiant unique.
- Les clés et les valeurs peuvent se présenter sous toutes les formes, des objets simples aux objets composés complexes.
- Examples:
  - Amazon DynamoDB
  - Apache Cassandra



amazon DynamoDB

cassandra

Cas d'utilisation Valeur-clé

- **Magasin de sessions:**

  - une application **Web ouvre une session lorsqu'un utilisateur se connecte**, puis ferme la session lorsque l'utilisateur se déconnecte ou lorsque la session expire.

  - l'application stocke toutes les données liées à la session dans la mémoire principale ou dans une base de données.

  - Les données de session peuvent inclure des informations sur le profil d'utilisateur, des messages, des données et des thèmes personnalisés, des recommandations, des promotions ciblées et des remises.

  - Chaque session d'utilisateur possède un identifiant unique. Les données de session sont uniquement interrogées par une clé primaire

Cas d'utilisation **Valeur-clé**

- **Panier d'achat:** un site Web d'e-commerce peut recevoir des milliards de commandes en quelques secondes.
  - Les bases de données clé-valeur peuvent gérer la mise à l'échelle de grandes quantités de données et de grands volumes de changements d'état tout en répondant aux besoins de millions d'utilisateurs simultanés grâce à un traitement et à un stockage distribués.

  - Les bases de données clé-valeur intègrent également une capacité de redondance, ce qui leur permet de gérer la perte de nœuds de stockage.

Base de données de documents

- Données sous forme de documents de type JSON.
- Les bases de données de documents permettent de stocker et d'interroger une base de données en utilisant le même format de modèle de document que celui qu'ils utilisent dans leur code d'application.
- La nature souple, semi-structurée et hiérarchique des documents et des bases de données de documents leur permet d'évoluer avec les besoins des applications.

```
[
    {
        "year" : 2013,
        "title" : "Turn It Down, Or Else!",
        "info" : {
            "directors" : [ "Alice Smith", "Bob Jones"],
            "release_date" : "2013-01-18T00:00:00Z",
            "rating" : 6.2,
            "genres" : ["Comedy", "Drama"],
            "image_url" : "http://ia.media-imdb.com/images/N/O9ERWAU7FS797AJ7LU8
            "plot" : "A rock band plays their music at high volumes, annoying th
            "actors" : ["David Matthewman", "Jonathan G. Neff"]
        }
    },
    {
        "year": 2015,
        "title": "The Big New Movie",
        "info": {
            "plot": "Nothing happens at all.",
            "rating": 0
        }
    }
]
```

mongoDB

CouchDB

Cas d’utilisation: Base de données de documents

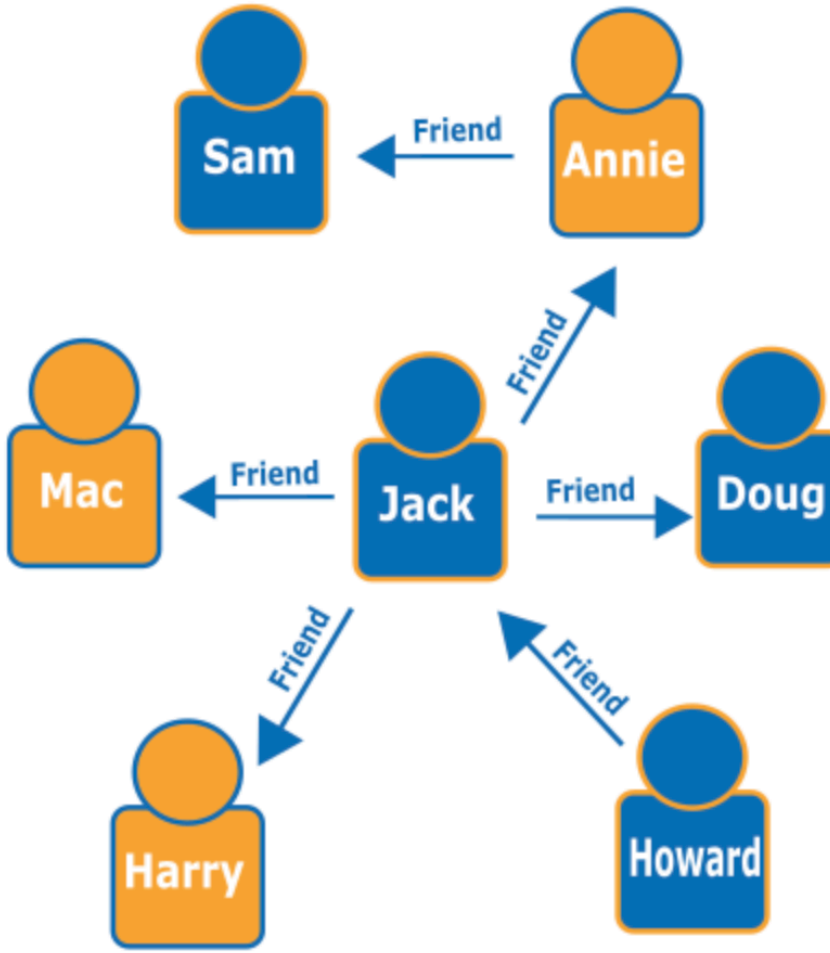
- Gestion de contenu blogs: Avec une base de données de documents, **chaque entité** que suit l'application peut être stockée comme un **document unique.**
- La base de données de documents est plus intuitive pour qu'un développeur puisse mettre à jour une application à mesure que les **exigences évoluent**.
- De plus, si le **modèle de données doit être modifié**, seuls les **documents concernés doivent être mis à jour**. Aucun schéma de mise à jour ou interruption de la base de données n'est nécessaire pour effectuer les modifications.

Cas d’utilisation: Base de données de documents

- **Catalogues:** dans une application d'e-commerce, des produits différents ont souvent des attributs différents.
- Gérer des milliers d'attributs dans des bases de données relationnelles n'est pas efficace, et cela nuit aux performances de lecture.
- En utilisant une base de données de documents, **les attributs de chaque produit peuvent être décrits dans un seul document** pour une gestion aisée et une vitesse de lecture supérieure. Si vous **modifiez les attributs d'un produit, ceux d'un autre produit ne seront pas modifiés.**

Base de données orientée graphe

- Les bases de données orientées graphe sont conçues pour stocker et rechercher des relations.
- Nœuds pour stocker les entités de données, ainsi qu'à des périphéries pour stocker les relations entre les entités
- Une périphérie possède toujours un nœud de départ, un nœud de fin, un type et une direction.
- Un graphe contenu dans une base de données orientée graphe peut être traversé le long de types de périphéries spécifiques ou à travers le graphe tout entier.
- Dans les bases de données orientées graphe, traverser des liaisons ou des relations se fait très rapidement, car les relations entre les nœuds ne sont pas calculés aux temps de demande, mais sont maintenues dans la base de données.



Base de données orientée graphe: cas d'utilisation

- les réseaux sociaux,
- les moteurs de recommandation:
  - stocker dans un graphe des relations entre des catégories d'informations telles que les intérêts d'un client, ses amis et son historique d'achat.
  - recommander des produits à un utilisateur en fonction des produits achetés par les autres utilisateurs qui sont abonnés à la même page sportive et dont l'historique d'achat est similaire.
  - trouver les personnes qui ont un ami en commun, mais qui ne se connaissent pas encore et émettre une recommandation de mise en relation.

Base de données orientée graphe: cas d'utilisation

- Détection des fraudes:

Les bases de données orientées graphe peuvent détecter les fraudes de manière sophistiquée. Avec des bases de données orientées graphe, vous pouvez utiliser les relations pour traiter les transactions financières et les transactions d'achat en temps presque réel.

En formulant des requêtes de graphe rapides, vous pouvez par exemple détecter qu'un acheteur potentiel utilise la même adresse e-mail et la même carte de crédit enregistrées lors d'un précédent cas de fraude.

Les bases de données orientées graphe peuvent également vous aider dans la conception de requêtes de graphe afin de facilement détecter les modèles de relations, tels que les cas d'utilisation d'une adresse e-mail personnelle par plusieurs personnes ou de partage d'adresse IP entre plusieurs personnes se trouvant à des adresses physiques différentes.

- Exemples:
  - Neo4J
  - Amazon Neptune

Bases de données orientées colonnes

- Sont organisées en familles de colonnes *(column family)*. (se rapproche du concept de table base de données relationnelle.)
- Les colonnes d'une base de données relationnelle sont statiques et sont présentes pour chaque ligne,
- Les colonnes d'une base de données orientée colonnes sont dynamiques et sont présentes uniquement pour les lignes concernées. Il est possible d'ajouter des colonnes à une ligne à tout moment.

	A	B	C	D	E
1	foo	bar	hello		
2		Tom			
3			java	scala	cobol

Organisation d'une table dans

1	A foo	B bar	C hello
2	B Tom		
3	C java	D scala	E cobol

Organisation d'une table dans

Bases de données orientées colonnes, Cas d'utilisation

- La multiplication massive du nombre de colonnes rend ce modèle capable de stocker les relations *one-to-many.*
- Monde du Web, les bases de données orientées colonnes permettront de supporter la montée en charge progressive.
  - listes d'articles pour chaque utilisateur
  - liste des actions effectuées par un utilisateur
  - chronologie d'évènement maintenue et accédée en temps réel
  - commandes en attentes (pour lesquelles le stockage de la liste des articles sera simple)

Choix du type de BD NoSQL

	Modélisation	Cas d'utilisation
Clé / Valeur	Modélisation simple, permettant d'indexer des informations diverses via une clé	Mise en cache Documents
Documents	Modélisation souple permettant de stocker des documents au format JSON dans des collections	Stockage de masse
Graphes	Modélisation optimisée pour les relations entre données.	Réseaux sociaux Systèmes de recommandation
Colonnes	colonnes dynamiques et présentes uniquement pour les lignes concernées	listes d'articles

TD-1 Applications réels de BD NoSQl

- Trouver un exemple réel d'utilisation de chaque type de base de données NoSQL. (Facebook, netflix…)
- Expliquer comment les donnes sont stockes en chaque cas et ajouter une schéma.
- Indiquer la BD utilisé, (Cassandra, Big Table, MongoDB …)
- Justifier pourquoi cet type de BD a était choisi.
- Expliquer comment fonctionne chaque exemple par rapport au Théorème de CAP
- Expliquer comment fonctionne chaque exemple par rapport au BASE
- En groupes de 4 faire une présentation Power Point de 10 min pour le prochain Mardi