esprit
Ecole Supérieure Privée
d'Ingénierie et de Technologies

Chp6 – SGBDNR

2016-2017

Mme Ines Slimene
ines.slimen@esprit.tn

Plan module

- Introduction
- Écosystème Hadoop
- HDFS
- MapReduce
- Langages de requête Hadoop :Pig, Hive
- **SGBDNR**
  - Différences entre une BDNR et une BD relationnelle
  - Typologies des BD non relationnelles
- Etude d'un SGBDNR : HBase

Plan

- Introduction
- NOSQL vs. BDR
- Bases de données NOSql
- Types de Bases de données NOSql
  - Clef/valeur
  - Orientées colonnes
  - Orientées documents
  - Orientées graphes

Introduction

- NoSQL : Not Only SQL.
- Que signifie vraiment « Not Only » ???
- Historique:
  - Le terme « NoSQL » dans sa signification actuelle vient du nom d'un séminaire (pendant une conférence sur Hadoop) qui a eu lieu en 2009 sur les nouveaux types de bases de données qui n'utilisent pas SQL.
  - Il fallait un nom qui fasse un bon hashtag sur Twitter : court, facile à mémoriser, et non populaire sur Google pour espérer pouvoir remonter dans les premiers résultats.

NOSQL vs. BDR

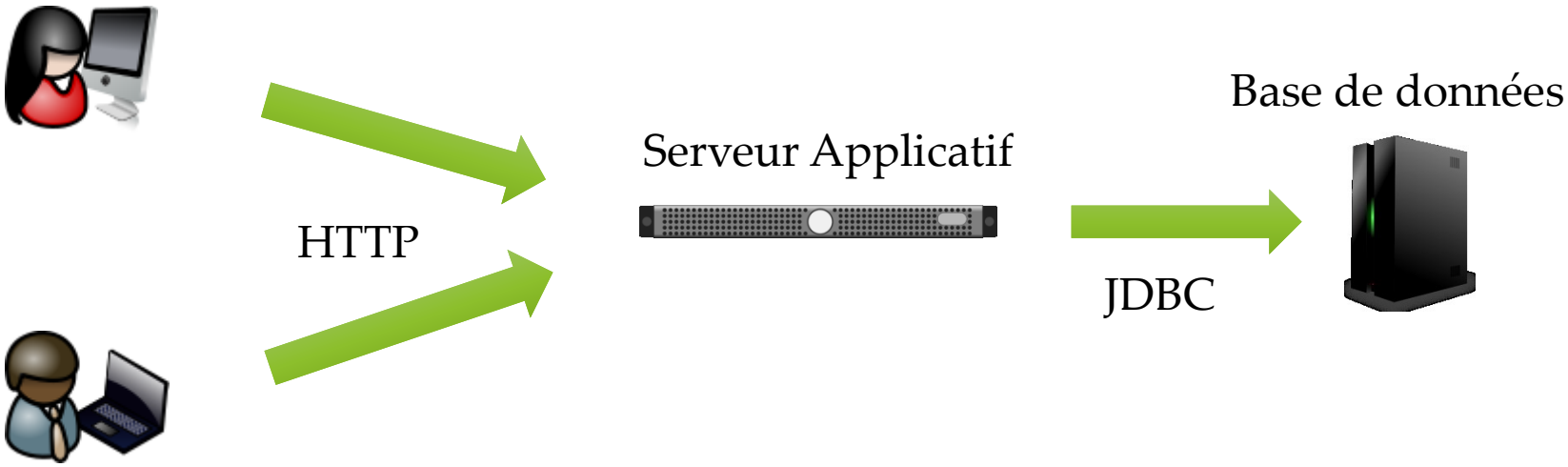
Bases de données relationnelles

Modèle	• Entités - relation • Simple • Universel
Requête	• SQL • Puissant • Ad-hoc
Transaction	• ACID
Maturité	• Utilisé massivement • Nombreux moteurs sur le marché • Nombreux outils

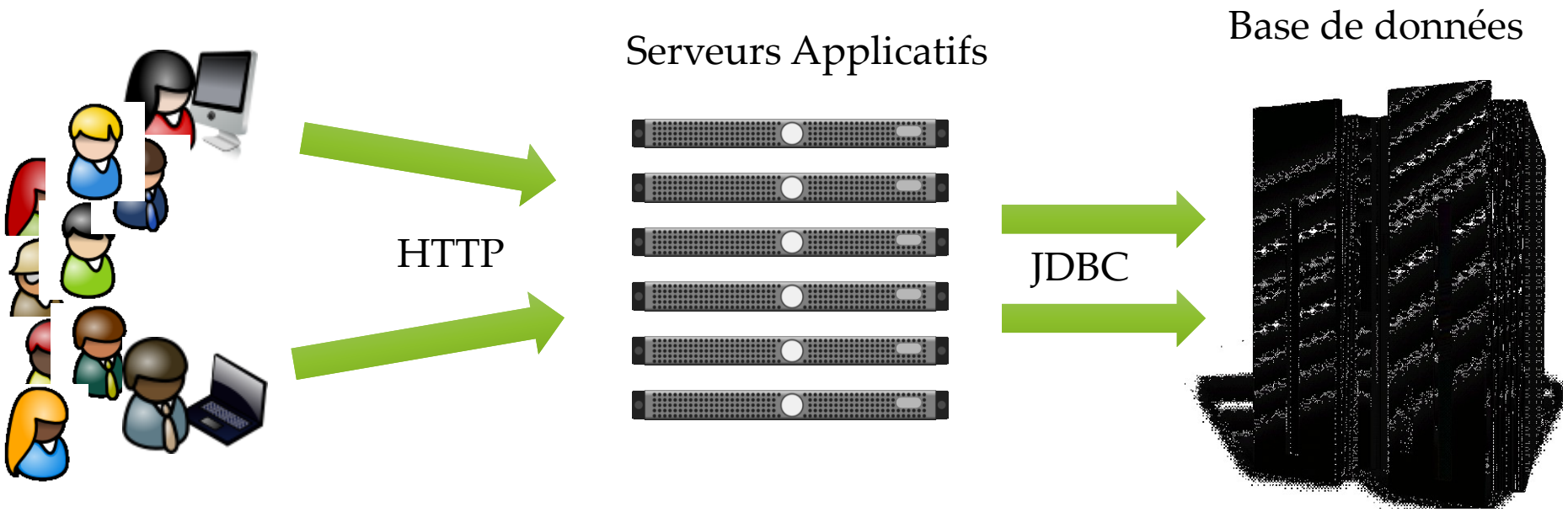
SQL

- Un langage de requête plus ou moins normé
- Tout information est décrite par des listes de n-uplets
- Opérations puissante
  - Sélection (where)
  - Projection (select)
  - Produit cartésien (join)
  - Union
  - Intersection
  - Différence

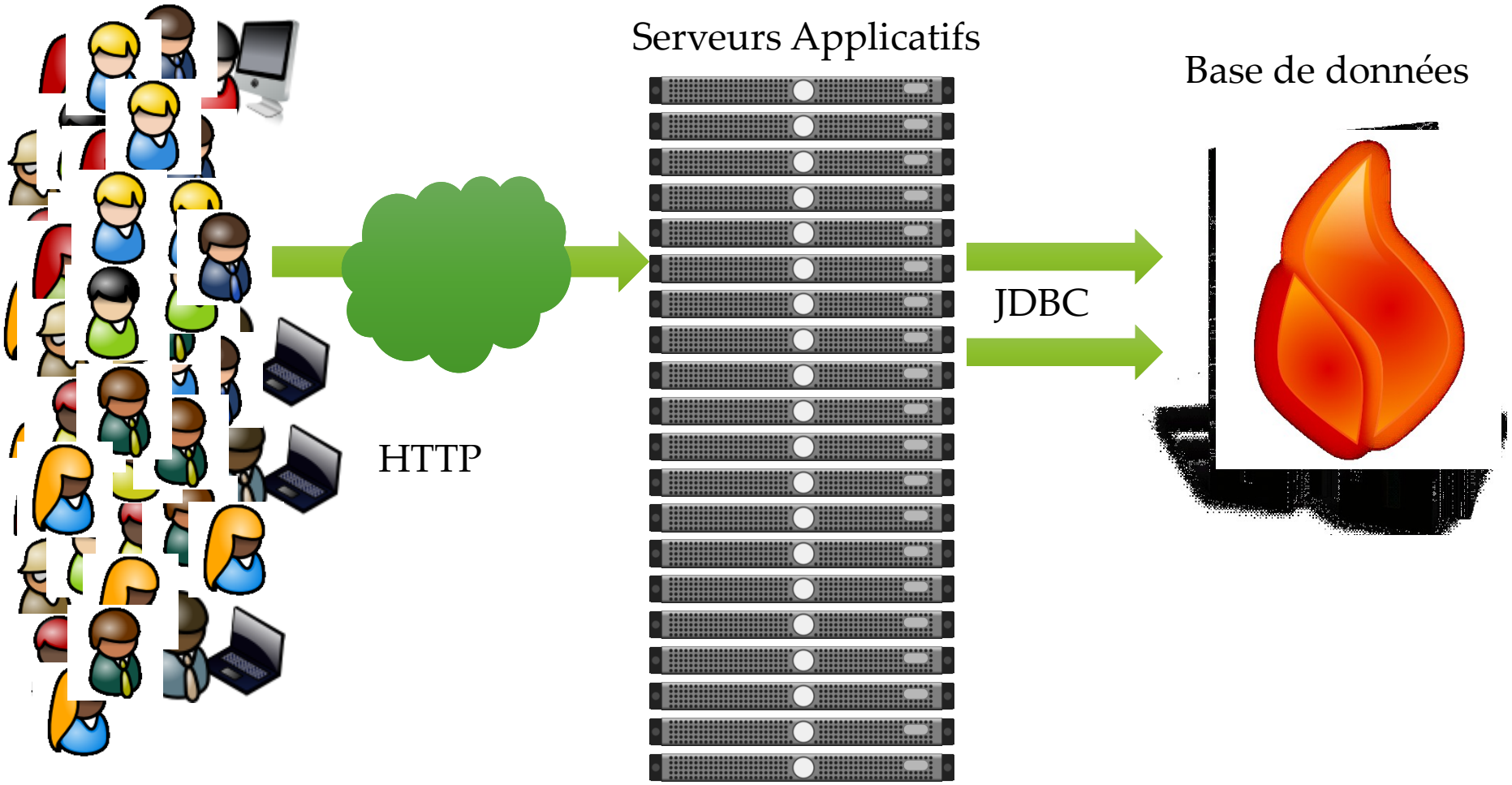
Mise en œuvre d'un SGBD-R



Mise en œuvre d'un SGBD-R



Mise en œuvre d'un SGBD-R

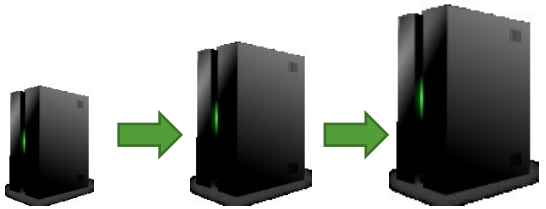


Limites des BD relationnelles

- Montée en charge difficile
  - Les règles d'intégrité compliquent la montée horizontale
  - Montée en charge verticale
    - Coût non linéaire
    - Atteint une limite
    - Point unique de défaillance

Les limites des BD relationnelles

- Scalabilité
  - Le modèle de consistance des RDBMS empêche l'utilisation de plusieurs machines pour répartir la charge (au moins en écriture)
  - Pour augmenter la performance d'accès à la base, pas d'autres moyens que d'acheter un plus gros serveur, puis un autre, puis un autre, …



Limites des BD relationnelles

- Relation : Contient un ensemble de n-uplets (les lignes) qui contiennent des attributs (les colonnes). Le domaine de chaque attribut est fixé et doit être composé de valeurs atomiques simples, i.e. non décomposables (1FN).
- Une requête SQL de type select retourne toujours une relation.
- Contraintes ACID impossibles à respecter sur un cluster (et a fortiori sur Internet) pour conserver un fonctionnement correct (haute disponibilité, rapidité d'exécution et cohérence permanente de la base, partition accidentelle du cluster).

BDR : Propriétés ACID

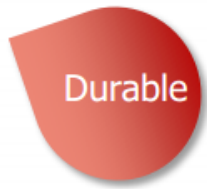
Atomicité : Toute transaction doit s'exécuter entièrement sans erreur ou pas du tout

Consistance : La base de données doit toujours être dans un état cohérent entre deux transactions (assurer par les contraintes d'intégrité).

Propriétés ACID

Les modifications d'une transaction ne sont pas visibles par les autres tant qu'elle n'a pas été validée.

Une fois validées, les données sont permanentes jusqu'à leur prochaine modification (persistance).



Besoin: Performances prédictibles

- La performance des opérations doit être prédictible
- Amazon:
  - Perte de 1 % de chiffre d'affaire si le temps d'affichage des pages augmente de 0,1 s
  - Plan qualité interne : Temps de réponse doit être < 300 ms pour 99,9 % des requêtes pour un pic de 500 requêtes par secondes
- Google pénalise les sites dont les pages s'affichent en plus de 1,5 s

Besoin: Prise en compte des pannes

- Amazon:
  - Un datacenter de 100 000 disques
  - entre 6 000 et 10 000 disques en panne par an (25 disques par jour)
- Les sources de panne sont nombreuses
  - Matériel : serveur (disque)
  - Matériel : réseau
  - Alimentation électrique
  - Anomalies logicielles
  - Mises à jour des logiciels et des OS.

Bases de données NOSql

- Bases de données NOSQL (Not OnlySQL) :
  - Bases de données non-relationnelles et largement distribuées
  - Permet une analyse et une organisation rapides des données de très grands volumes et de types de données divers
  - Appelées également
    - Cloud Databases
    - Non-Relational Databases
    - BigData Databases

Bases de données NOSql

- Le NoSQL vise :
  - Gestion d'énormes quantités de données
  - Structuration faible du modèle
  - Montée en charge
- Caractéristiques
  - Conçu pour être exécutée dans un cluster,
  - Tendance à être Open Source,
  - Modèle de données sans schéma
  - Architecture distribuée
  - Utilisation de langages et interfaces qui ne sont pas uniquement du SQL

Bases de données NOSql

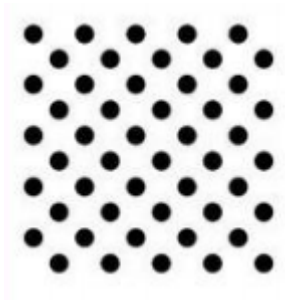
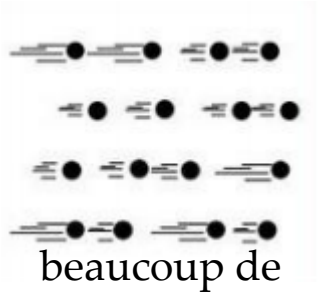
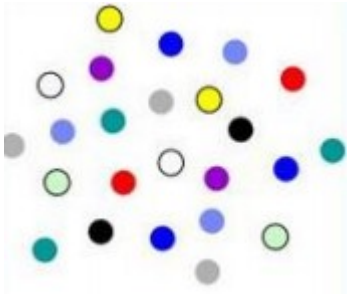
Avantages	Inconvénients
Fiabilité (tolérance aux pannes)	Logiciel
Extensibilité	Réparation
Partage des ressources	Réseau
Vitesse/Performance	Sécurité

Bases de données NOSql

- Business Intelligence et Analyse
  - Capacité des bases NOSQL à exploiter les données collectées pour dériver des idées
  - Extraction d'informations décisionnelles à partir d'un grand volume de données, difficile à obtenir avec des bases relationnelles
- Capacités transactionnelles modernes
  - Nouvelles définitions du principe de transaction
  - Utilisation des propriétés BASE au lieu des propriétés ACID

Bases de données NOSql

- Adaptation des BD NOSQL au BigData

Volume	Vitesse	Variété	Véracité
			
données très nombreuses (To et Po)	beaucoup de données qui arrivent rapidement, à partir de plusieurs sources	données structurées, semi-structurées ou non-structurées	données stockées et gérées à plusieurs endroits et data centers.

Les systèmes NoSQL : CAP

- Dans un environnement distribué, il n'est pas possible de respecter simultanément toutes les contraintes. On peut, en revanche, respecter deux.

Consistency

Si j'écris une donnée dans un nœud et que je la lis à partir d'un autre nœud dans un système distribué, je retrouve ce que j'ai écris sur le premier.

Availability

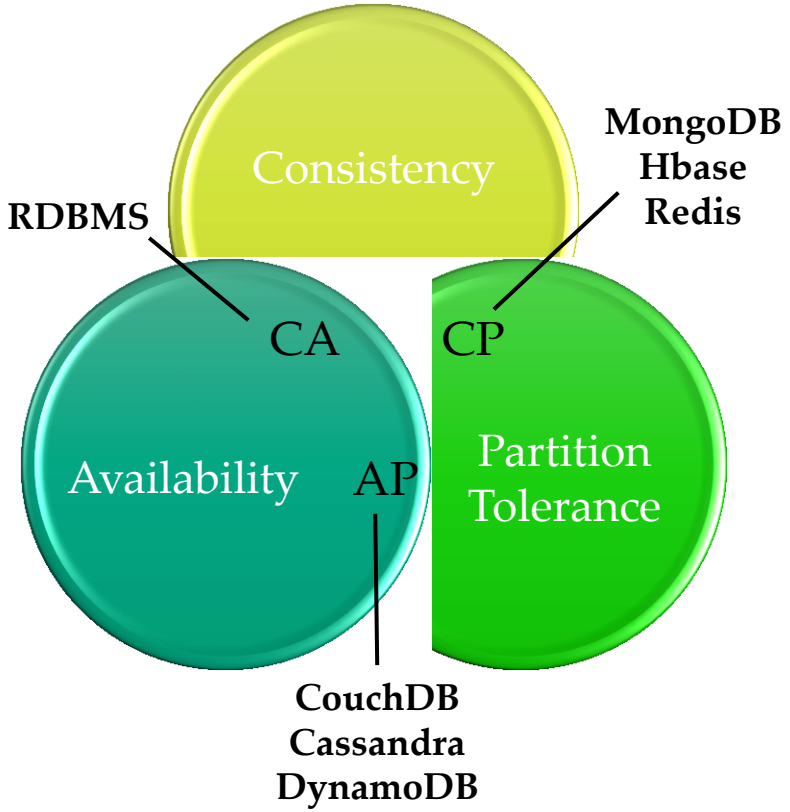
A tout moment, pour chaque requête, la réponse est garantie. Même en cas de panne, les données restent accessibles

Partition Tolerance

Les données peuvent être partitionnées sur différents supports sans souci de localisation.
Les activités continuent sans interruption lors de la modification du système et en cas de chute du réseau

Les systèmes NoSQL : CAP

- Présentation des trois combinaisons CA, CP, AP :
  - CA – un seul cluster. Ainsi tous les nœuds sont en contact. Quant on ajoute une partition le système est bloqué
  - CP – des données peuvent être non accessible mais le reste est cohérent
  - AP - la valeur lue par un nœud peu être incohérente avec le contenu des autres nœuds



NOSQL : Propriétés BASE

- BASE est plus flexible que ACID, accepte certaines erreurs, dont l'occurrence est assez rare

Basically Available

Le système garantit la disponibilité, comme définie dans le théorème CAP

Soft-State

L'état du système peut changer dans le temps, même sans nouvelles entrées, et ce à cause du principe de consistance éventuelle

Eventual Consistency

Le système deviendra cohérent au fil du temps, en supposant que le système ne reçoit pas des entrées pendant ce temps la.

NOSQL et BDR : Récapitulatif

BRD	NOSQL
Consistance forte	Consistance éventuelle
Grandes quantités de données	Enorme quantité de données
Évolutivité possible	Evolutivité facile
SQL	
Bonne disponibilité	Très haute disponibilité

Synthèse

- Les SGBDR sont maintenant une option parmi d'autres. Leur généricité leur permet d'être utilisés dans de nombreux cas mais les rends peu performants et complexes sur certains types de données (masse de données, attributs trop nombreux, …).
- Il faut savoir utiliser la solution la plus adaptée et pourquoi pas en utiliser plusieurs.
- Bases de données NOSQL
  - Performances sur de gros volumes de données
  - Performances sur des données non structurées
  - Evolutivité très importante, même pour de faibles volumes

Synthèse

- Les systèmes NoSQL ne remplaceront jamais les bases de données relationnelles. Il faut les voir comme une alternative après un quasi monopole du relationnel de plusieurs décennies.
- Cependant:
  - Technologie assez jeune et Manque d'outils la supportant
  - Encore en évolution, pas de standards
  - Pas de langage de requêtage commun comme SQL
  - On doit faire plus de travail au niveau du code, ce qui peut influer sur la performance

Types de Bases de données NOSql

Types de Base de données NOSql

- Types des bases de données NOSQL
  - Clé/valeur
  - Orientées colonnes
  - Orientées documents
  - Orientées graphes

1. Clé/Valeur

- stockage simple et direct d'un tableau associatif (hashtable) où tous les accès se font en utilisant la clé primaire
- Conçues pour sauvegarder les données sans définir de schéma
- Interrogation uniquement par la clé. On récupère une donnée associée à une clé.
- Ne fait que stocker des données en rapport avec une clé. Seulement 3 opérations possibles :
  - Put : donner une valeur à une clé
  - Get : récupérer la valeur d'une clé
  - Delete : effacer la clé et sa valeur
- Exemple:
  - DynamoDB(Amazon), Redis, Voldemort(LinkedIn)

1. Clé/Valeur

☑ A utiliser pour ...

- De l'information très volatile
  - Session utilisateur, données d'un panier d'achat
- De l'information très peu volatile et accéder très fréquemment
  - Descriptions produit, paramétrage applicatif

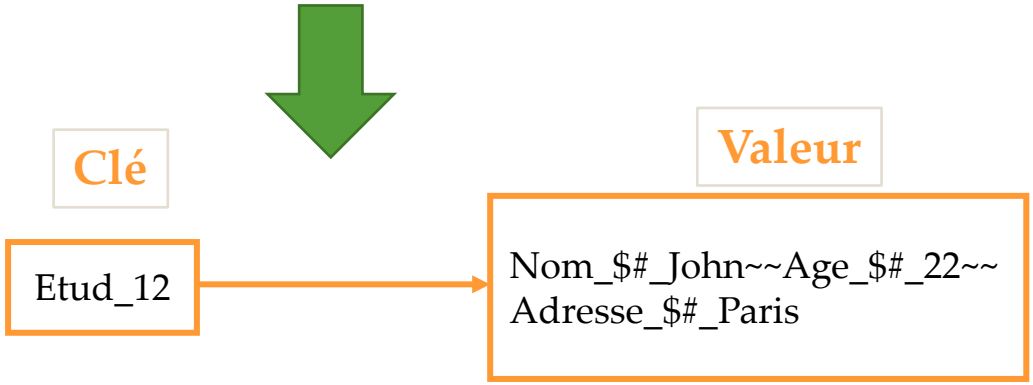
☒ A éviter pour ...

- Des données possédant des relations
  - Relations entre agrégats ou corrélation entre données de différents ensemble de clés
- Des opérations impliquant de multiples clés
- Des besoins de requêtage par les données

1. Clé/Valeur : Exemple

- Table étudiant :

Id	Nom	Age	Adresse
12	John	22	Paris
13	Ali	Null	Bruxelles
14	Alain	Null	Null



2. Orientée Documents

- Basé sur un système clé-valeur
- Mais la valeur est une structure que le système connaît et peut donc parcourir.
- Chaque document est un objet qui contient un ou plusieurs champs, et chaque champs contient une valeur typée (string, date….)
- Avantage : pouvoir récupérer, via une seule clé, un ensemble d'informations structurées de manière hiérarchique
  - Dans les bases relationnelles, cela impliquerait plusieurs jointures
- Exemples : Mongo DB (SourceForge), CouchDB (Apache), RavenDB…

2. Orientée Documents

☑ **A utiliser pour ...**

- Données avec partie structurée et partie non structurée
- Données de suivi temps réel ou analytiques

☒ **A éviter pour ...**

- Opérations nécessitant consistance sur plusieurs agrégats
- Des structures d'agrégat très changeantes avec des besoins de requêtage forts
  - Inconvénient du schemaless

2. Orientée Documents

Id	Nom	Age	Adresse
12	John	22	Paris
13	Ali	Null	Bruxelles
14	Alain	Null	Null

```
{
   {Nom: « John », Age: 22, Adresse: »Paris »},
   {Nom: « Ali », Adresse: »Paris »},
   {Nom: « Alain »}
}
```

3. Orientées Colonnes

- Évolution de la BD clé/valeur
- Ressemble aux SGBDR, mais avec un nombre de colonnes dynamiques, différent d'un enregistrement à un autre (pas de colonnes portant les valeurs NULL)
- Exemples: Hbase(Hadoop), Cassandra (Facebook, Twitter), BigTable(Google)

3. Orientées Colonnes

☑ **A utiliser pour ...**

- Données avec partie structurée et partie non structurée
- Données de publication variables
  - CMS, Blogging avec commentaires, contenu dynamique, etc …
- Compteurs et analytiques

☒ **A éviter pour ...**

- Des besoins de requêtage complexes
- Des besoins de calcul d'agrégation simples (nécessité de passer systématiquement par Map-Reduce aujourd'hui)

3. Orientées Colonnes

Id	Nom	Age	Adresse
12	John	22	Paris
13	Ali	Null	Bruxelles
14	Alain	Null	Null

12	Nom|John	Age|22	Adresse | Paris
13	Nom|Ali	Adresse | Bruxelles	
14	Nom|Alain		

4. Orienté Graphes

- On ne stocke plus un simple ensemble de données mais des relations.
- S'appuie sur les notions de nœuds, de relations et des propriétés qui leur sont rattachées
- Conçues pour les données dont les relations sont représentées comme graphes, et ayant des éléments interconnectés, avec un nombre indéterminé de relations entre elles.
- Le stockage d'un graphe sur plusieurs serveurs est un problème difficile (problème de performance même pour un simple calcul de chemin si le graphe est réparti sur plusieurs serveurs).
- Adapté aux traitements des données des réseaux sociaux
- Exemples: Neo4J et InfiniteGraph, OrientDB

4. Orienté Graphes

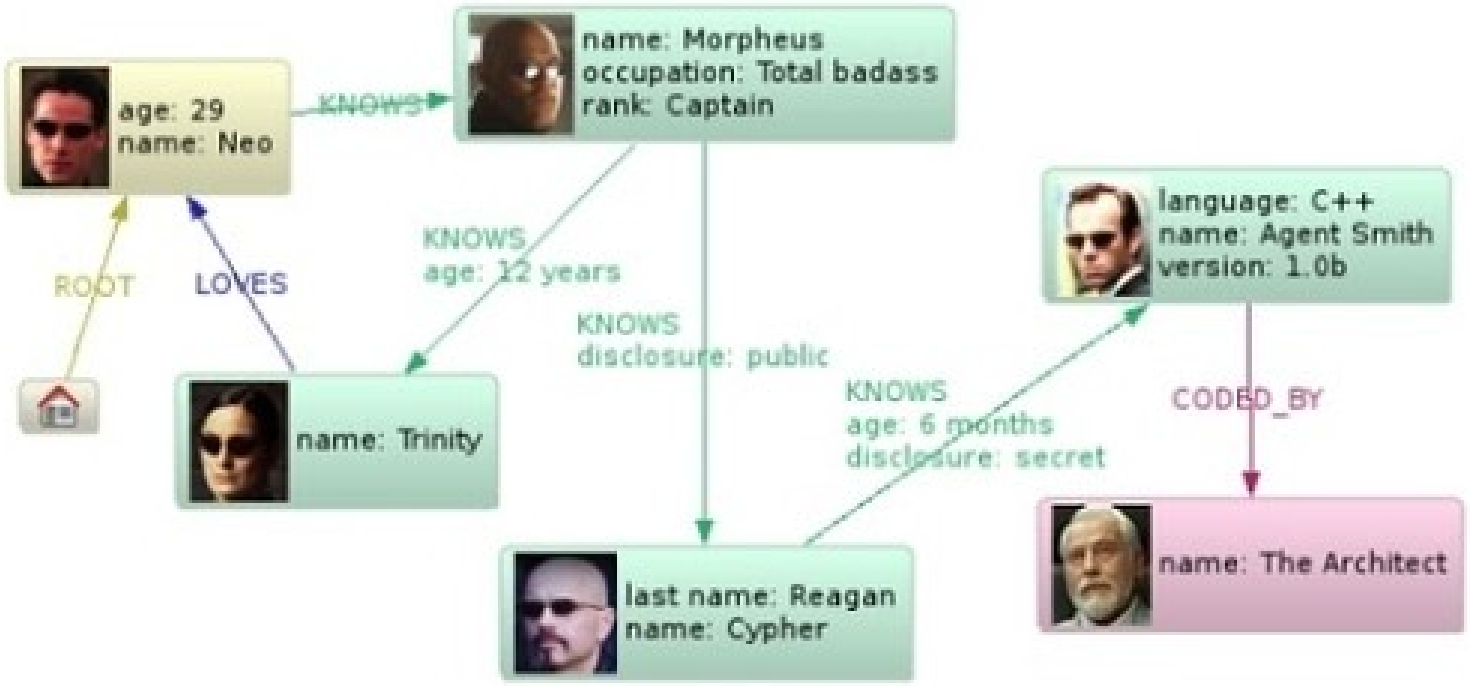
☑ **A utiliser pour ...**

- Les moteurs de recommandations
  - « Les autres clients ayant acheté ce produit ont aussi acheté ... »
- Les données naturellement connectées
  - Réseaux sociaux
- Les services basés sur la localisation ou le calcul d'itinéraires

☒ **A éviter pour ...**

- Les cas où de nombreux nœuds doivent être mis à jour

4. Orienté Graphes



Comparaison entre Types de BD NOSQL