Module: Bases de données à large échelle

NOSQL

Enseignant: Adel Jebali

adel.jbali@fst.utm.tn
adel.jebali@esprit.tn

Plan du cours

- Introduction
- Présentation de NOSQL
- Définition
- Bases de données NOSQL
- Les Bases de Données Clé-valeur
- Les Bases de Données colonnes
- Les Bases de Données documents
- Les Bases de Données Graphes
- Atelier

Introduction (1/3)

- L'essor de très grandes plateformes Web (Google, Amazon, …) et réseaux sociaux (Facebook, twitter, …) a mis en évidence des nouveaux besoins:
  - **Volume important de données** à gérer par ces applications nécessitant leur distribution et leur traitement sur de nombreux serveurs (DataCenters).
  - **Scalabilité:** Besoin d'extensibilité énorme
  - Données souvent associées à des objets **complexes** (composés d'autres objets) et **hétérogènes** (structures différentes).
  - **Données interconnectées:** liées (statuts / Twits, commentaires, …)

Introduction (2/3)

- Les SGBD classiques, basés sur le modèle relationnel, sont:
  - Incapables de gérer de très grands volumes de données
  - Incapables de gérer des débits extrêmes (requêtes par seconde)
  - peu adaptés au stockage et à l'interrogation de certains types de données (hiérarchiques, faiblement structurées, semi-structurées)
  - système centralisé, avec possibilités limitées de distribution

Introduction (3/3)

- D'où le besoin d'une nouvelle approche de stockage et de gestion des données :
  - permettant une meilleure scalabilité (extensibilité) dans des contextes fortement distribués.
  - permettant une gestion d'objets complexes et hétérogènes sans déclarer tous les champs représentant un objet.
  - ne se substituant pas aux SGBD Relationnels mais les complétant en comblant leurs faiblesses ➔ « Not Only SQL » (**NOSQL**)

Présentation de NOSQL

- L'origine de l'accronyme NoSQL remonte à 1998, lorsque **Carlo Strozzi** propose un SGBD(Strozzi NoSQL) s'éloignant du système relationnel (Fichiers textes interrogés par des scripts Linux).
- Plusieurs développeurs ensuite ont développé leurs solutions propres comme alternatives au modèle SQL
- Exemples: Facebook (Cassandra), Google (BigTable),etc

Présentation de NOSQL

- Une rencontre en 2009 à San Francisco a regroupé une grande partie de ces développeurs pour élaborer et adopter enfin le concept des bases de données NOSQL (Voldemort de LinkedIn, Cassandra de Facebook, Hbase, HyperTable, CouchDB et MongoDB).
- Cette adoption des bases de données NOSQL a abouti à une variété de solutions développées ce qui permet de répondre aux différents besoins.

Définition

- NOSQL (Not Only SQL) est une approche de la conception des bases de données et de leur administration particulièrement utile pour un grand volume de données distribuées.
- NOSQL englobe une gamme étendue de technologies et d'architectures, afin de résoudre les problèmes de performances que les bases de données relationnelles ne sont pas conçues pour affronter.

Bases de données NOSQL: BASE vs ACID

- Les transactions des BD relationnelles et Objets respectent les propriétés ACID (Atomicité, Cohérence, Isolation, Durabilité):
  - **Atomicité :** Une transaction s'effectue entièrement ou pas du tout
  - **Cohérence :** Le contenu doit être cohérent au début et à la fin d'une transaction (exp: vis-à-vis des contraintes d'intégrité)
  - **Isolation** : Les modifications d'une transaction ne sont visibles ou modifiables que quand celle-ci a été validée (concurrence)
  - **Durabilité :** Une fois la transaction validée, l'état de la base est permanent (non affecté par les pannes ou autre)
- Toutefois, ces propriétés ne sont pas applicables dans un contexte fortement distribué .

Bases de données NOSQL: BASE vs ACID

- Pour cela, les bases de données NoSQL ont adopté d'autres propriétés qui sont regroupés par l'acronyme **BASE:**
  - **B*asically Available*** (Globalement disponible): quelle que soit la charge de la base de données, le système garantie un taux acceptable de disponibilité de la donnée
  - ***Soft-state*** *(état flexible)* **:** Une BD NoSQL n'a pas à être cohérente à tout instant
  - ***Eventually consistent*** (Finalement cohérent)**:** À terme, la BD atteindra un état cohérent

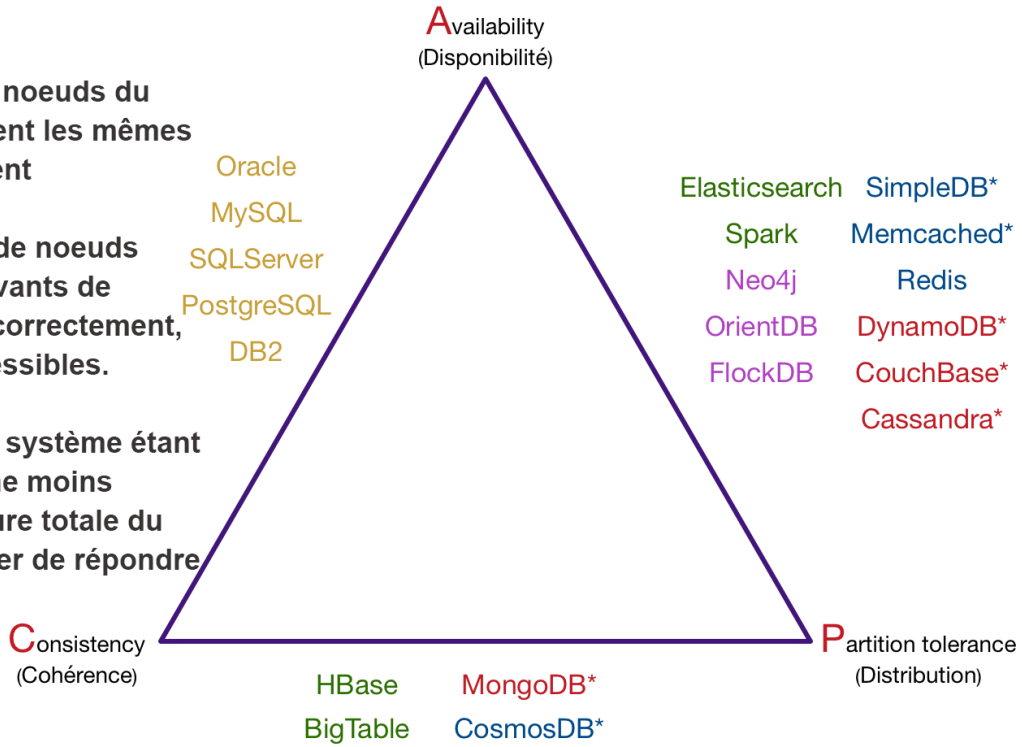
Bases de données NOSQL:
Théorème de CAP

- Enoncé par « Eric Brewer » (scientifique américain) en 1999.
- **Principe:** impossible sur un système informatique distribué de garantir simultanément les 3 contraintes suivantes:

C(Coherence): tous les noeuds du système voient exactement les mêmes données au même moment

A(Availability): la perte de noeuds n'empêche pas les survivants de continuer à fonctionner correctement, les données restent accessibles.

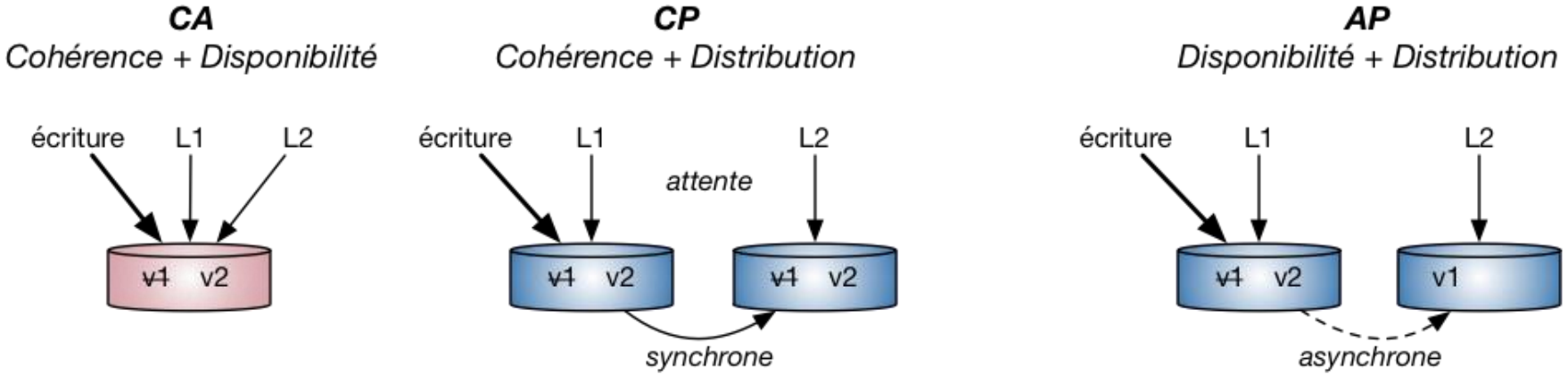
P(Partition tolerance): le système étant partitionné, aucune panne moins importante qu'une coupure totale du réseau, ne doit l'empêcher de répondre correctement.



- Les SGBD classiques favorisent la cohérence et la disponibilité

Bases de données NOSQL:
Théorème de CAP

- Les BD **NoSQL** tentent à privilégier:
  - soit la **disponibilité** et la tolérance au partitionnement.
  - soit la **cohérence** et la tolérance au partitionnement.

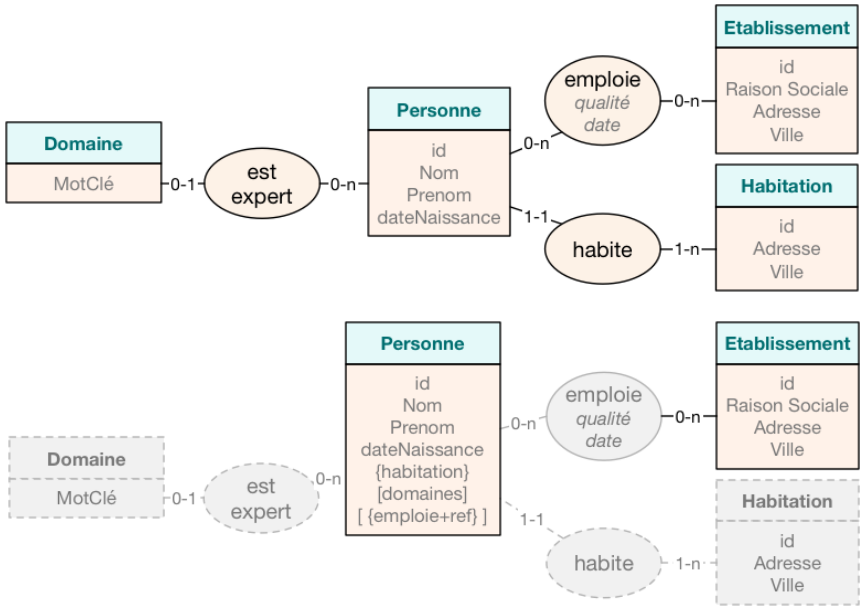


Exemple: Il peut être préférable que 2 personnes faisant la même recherche sur Google, ou consultant une liste de commentaires sur Facebook, obtiennent des résultats différents que de n'avoir aucune réponse.

Bases de données NOSQL: Dénormalisation

- Dans une base de données NoSQL, il n'y a pas de schéma fixe ni de jointure.
- Le passage du relationnel à NOSQL se fait par le processus de **dénormalisation** qui consiste à regrouper plusieurs tables liées par des références, en une seule table, en réalisant statiquement les opérations de jointure adéquates.

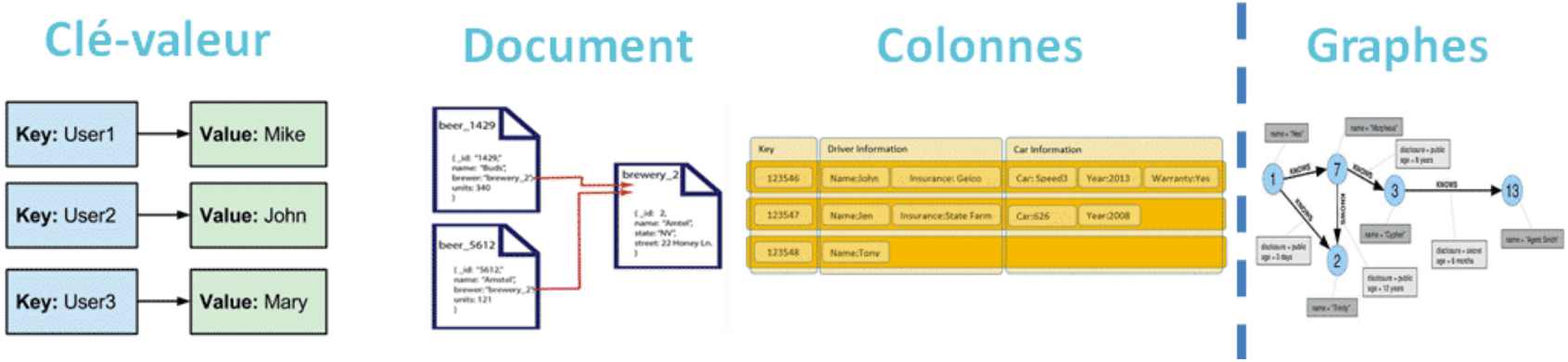
➔ Données imbriquées




```
{
  "_id" : 1,    "nom" : "Travers",    "prenom" : "Nicolas",
  "domaines" : ["SGBD", "NoSQL", "RI", "XML"],
  "emplois" : [
    {"id_etablissement" : "100", "qualité" : "Maître de Conférences",
        "date" : "01/09/2007"},
    {"id_etablissement" : "101", "qualité" : "Vacataire",
        "date" : "01/09/2012"}
  ],
  "Habite" : {"adresse" : "292 rue Saint Martin", "ville" : "Paris"}
}
```

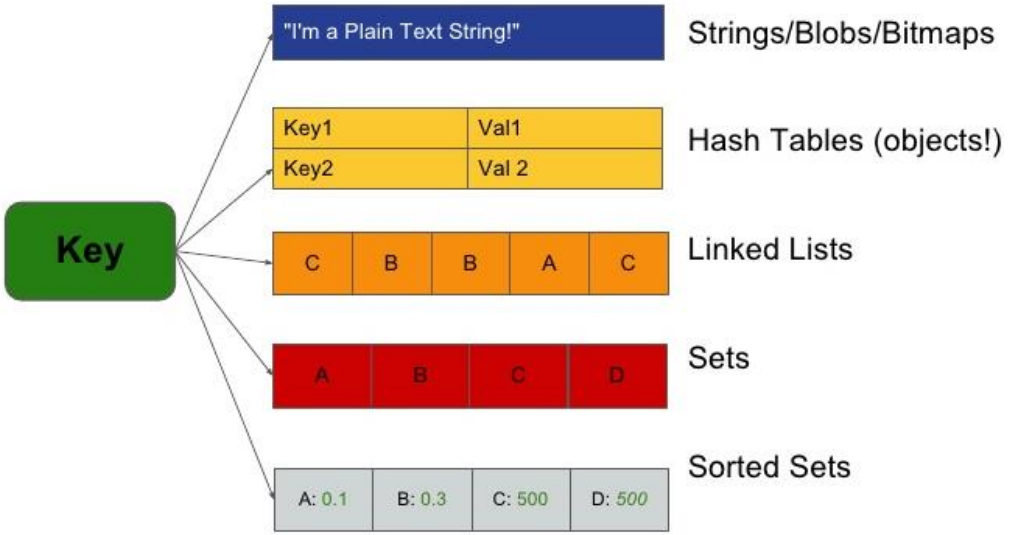
Bases de données NOSQL: Typologie

- Les bases des données NOSQL peuvent être classifiées en quatre grandes catégories:
  - **Les bases de données orientés Clé-Valeur**
  - **Les bases de données orientés colonnes**
  - **Les bases de données orientés documents**
  - **Les bases des données orientées graphes**



Les BD Clé-Valeur (1/4)

- Cette catégorie de bases fonctionne comme une table associative
- Les données sont simplement représentées par une paire clé/valeur .
- Simple à mettre en place et permet un accès rapide aux informations.
- La clé sert comme identifiant unique de type chaine de caractères et la valeur peut être:
  - une simple chaîne de caractères,
  - composée de couples clé-valeur.
  - une liste ordonnée
  - ou une liste non ordonnée



Les BD Clé-Valeur (2/4)

- Elles sont très facilement scalables .
- Elles sont capables de gérer des millions – voire des milliards – d'entrées avec un temps de réponse très bas.
- Le système de stockage ne connait pas la structure de l'information qu'il manipule . la structure de l'objet est libre, et laissé à la charge du développeur de l'application.
- Le requêtes se font uniquement sur les clés.

Les BD Clé-Valeur (3/4)

- Utilisations principales des BD NoSQL type «Clés-Valeurs» :
  - Les profils et les préférences d'utilisateurs,
  - Les données de panier d'achat,
  - Les données de capteurs,
  - Les logs d'applications,
  - Les données des flux de clics * (clickstream data),
  - etc.

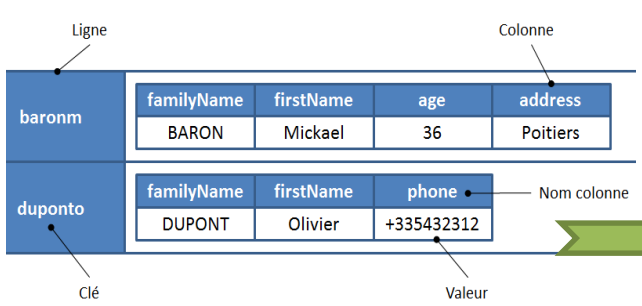
(*): **Flux de clics:** Technique qui enregistre les clics des utilisateurs pour l'analyse de l'activité Internet, les tests de logiciels, la recherche de marché et pour l'analyse de la productivité des employés.

Les BD Clé-Valeur (4/4)

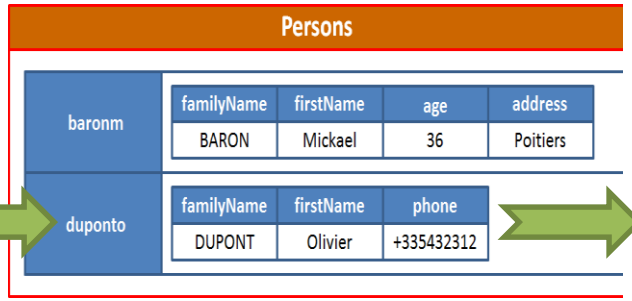
- Implémentations les plus connues :
  - **Redis**: projet sponsorisé par VMWare
  - **Amazon Dynamo** (Propriétaire à Amazon)
  - **Riak:** distribué sous license Apache et inspiré de Dynamo
  - **Voldemort** (développé par LinkedIn puis passage en open source).

Les BD colonnes (1/3)

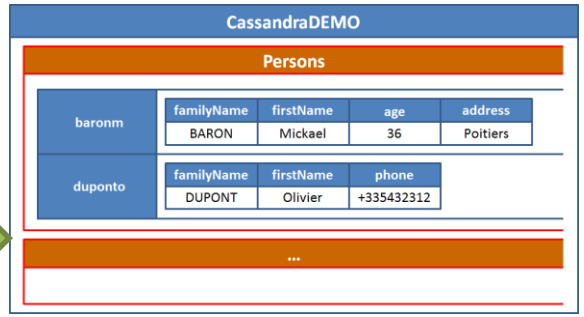
- Modèle proche d'une table dans un SGBDR, mais avec un nombre de colonnes qui peut varier d'un enregistrement à un autre
- Structure hiérarchique:
  - **Colonnes**: entité de base représentant un champ de donnée.
  - **Familles de colonnes (Table)**: ressemble le plus aux tables en SQL
  - **Super Famille de colonne (Keyspace)**: permet d'ajouter un niveau d'imbrication.



SuperColonne: composé de colonnes



Famille de colonnes/Table



Super famille de colonnes

Les BD colonnes (2/3)

- Les BD orientées colonnes sont utilisés pour:
  - L'enregistrement et l 'analyse de l'activité des clients (Exp: Netflix),
  - L'optimisation de la recherche (Exp: Ebay),
  - Analyse et calcul des parts d'audience et les votes des spectateurs pour les chaines tv,
  - La journalisation des évènements pour les applications,
  - etc.

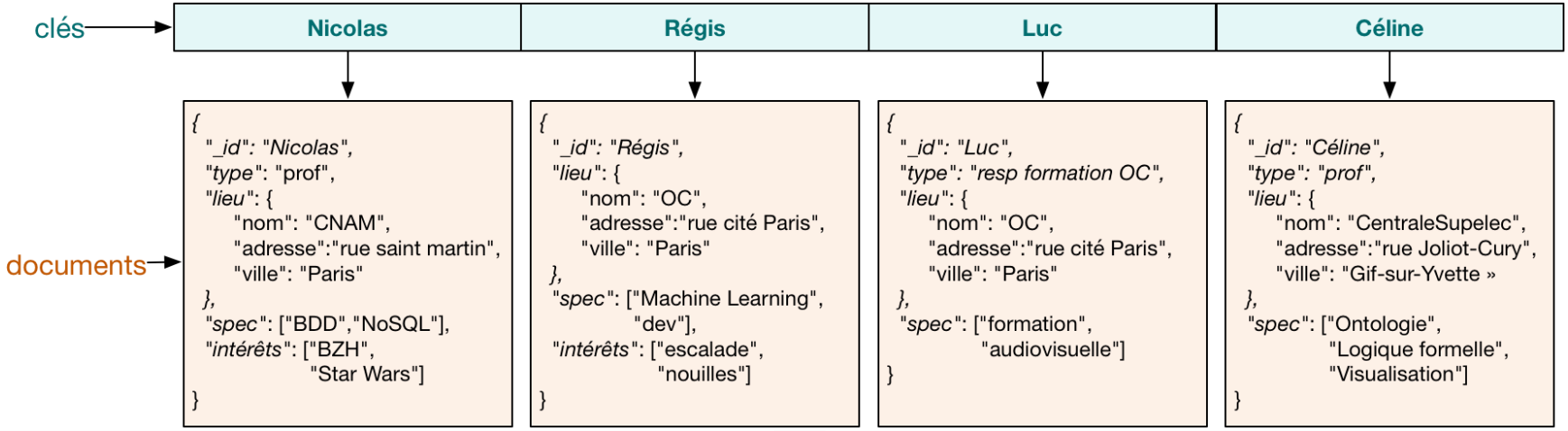
Les BD colonnes (3/3)

- Implémentations les plus connues :
  - **Cassandra** (Initialement développée par Facebook puis diffusé en Open Source en Juin 2008)
  - **HBase** (Open Source de BigTable de Google utilisé pour l'indexation des pages web, Google Earth, Google analytics, ...)
  - **Hypertable** (Open Source inspiré du BigTable de Google)

HYPERTABLE

Les BD documents (1/5)

- Elles stockent une collection de "documents"
- Un document est composé de champs et des valeurs associées. Ces valeurs peuvent être:
  - soit d'un type simple (entier, chaine de caractère, date, ...)
  - soit elles mêmes composées de plusieurs couples clé/valeur.



Les BD documents (2/5)

- Permettent d'effectuer des requêtes sur le contenu des documents (ou des objets) ce qui n'est pas possible avec les BD clés/valeurs.
- Deux langages sont principalement utilisés pour représenter les documents structurés : XML et JSON.
- La forme des requêtes est dépendante du langage de représentation des documents (XML ➔ XPath et Xquery).

Les BD documents (3/5)

- **JSON** est un format de représentation logique de données, hérité de la syntaxe de création d'objets en JavaScript. Il est facilement lisible par les humains, facilement « *parsable » par les machines, et indépendant des* langages qui l'utilisent.
- **Format:** Des couples de type "nom": valeur, comme l'on peut en trouver dans les tableaux associatifs.

```
1  {
2      "nom" : "Norris",
3      "prenom" : "Chuck",
4      "age" : 73,
5      "etat" : "Oklahoma"
6  }
```

- Il ne doit exister qu'un seul élément père par document contenant tous les autres : un élément racine
- Tout fichier JSON bien formé doit être :
  - ✓ soit un objet commençant par { et se terminant par },
  - ✓ soit un tableau commençant par [et terminant par].
- Cependant ils peuvent être vides, ainsi [] et {} sont des JSON valides.
- Les séparateurs utilisés entre deux paires/valeurs sont des virgules.
- Un objet JSON peut contenir d'autres objets JSON.

Les BD documents (4/5)

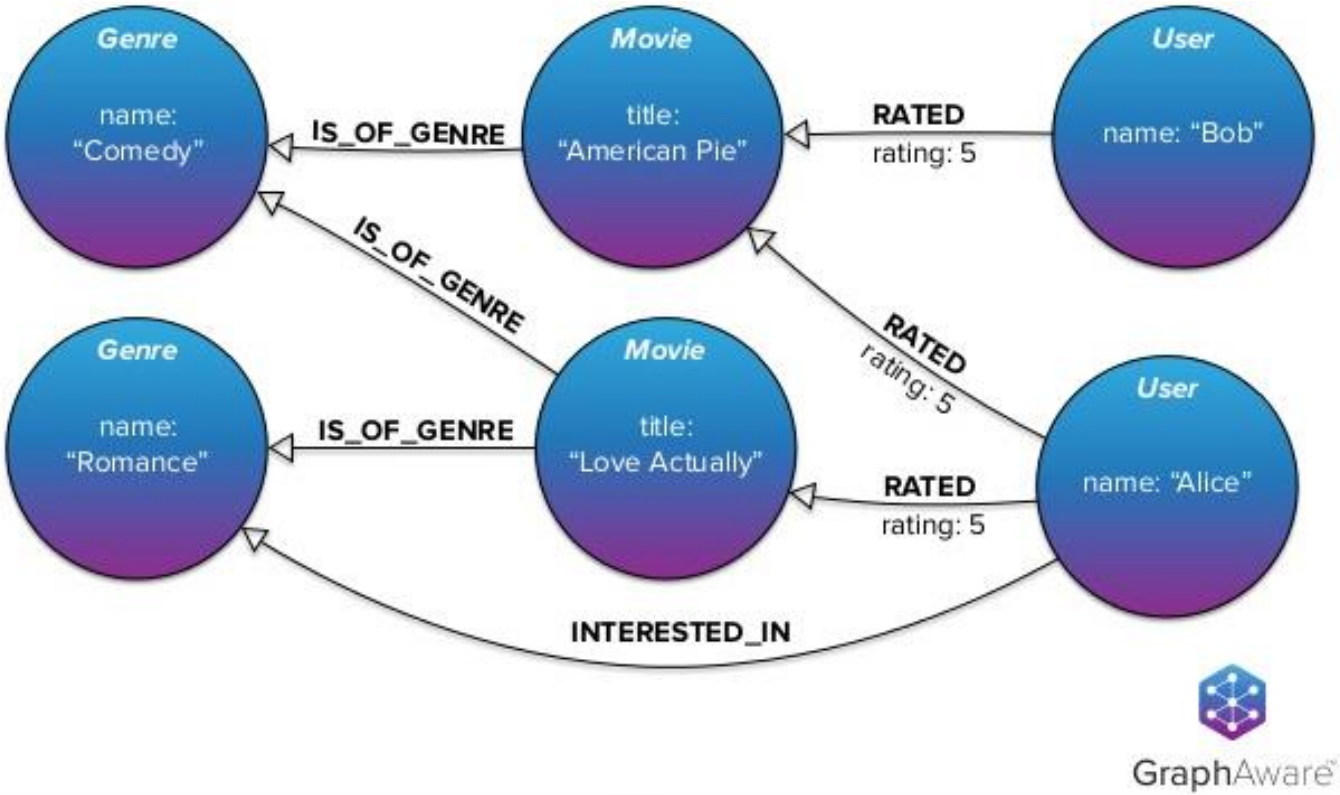
- Les BD orientées documents sont utilisés pour:
  - Enregistrement d'événements
  - Systèmes de gestion de contenu (CMS)
  - Web analytique ou analytique temps-réel (Google Analytics)
  - Catalogue de produits
  - Systèmes d'exploitation
  - …

Les BD documents (5/5)

- Implémentations les plus connues :
  - **BigTable:** Propriétaire à Google
  - **MangoDB:** license GPL + Apache (2009)
  - **CouchDB** : license Apache (2005)
  - **RavenDB** : Open Source pour plateformes « .NET/Windows »


Les BD graphes (1/5)

- Modèle de représentation des données basé sur la théorie des graphes.



Les BD graphes (2/5)

- Leur conception s'appuie sur les notions de noeuds, de relations et de propriétés qui leur sont rattachées.
- Elles permettent la modélisation, le stockage et la manipulation de données complexes liées par des relations variables.
- Elles sont adaptées à la manipulation d'objets complexes organisés en réseaux : cartographie, réseaux sociaux, ..

Les BD graphes (3/5)

- Elles utilisent :
  - Un moteur de stockage pour les objets (similaire à une base documentaire, chaque entité de cette base étant nommée nœud)
  - Un mécanisme de description d'arcs (relations entre les objets), arcs orientés et avec propriétés (nom, date, ...)

Les BD graphes (4/5)

- Les BD orientées «Graphe» sont utilisées pour :
  - **Moteurs de recommandation** (programmes qui effectuent en temps réel des recommandations produits sur un site marchand)
  - **Informatique décisionnelle**
  - **Web Sémantique** (Extension du web par Le W3C)
  - **Social computing** (domaine de l'informatique qui s'intéresse à l'intersection du comportement social et des systèmes informatiques)
  - **Données géospatiales** (formes et localisation des objets)
  - **Généalogie**
  - **L'internet des objets** (Internet of things)
  - **Catalogue des produits**
  - **Services de routage, d'expédition (colis) et de géolocalisation**
  - **Services financiers** : gestion des risques, détection des fraudes,…

Les BD graphes (5/5)

- Implémentations les plus connues :
  - **Neo4J:** license GPL, écrit en JAVA (2007)
  - **OrientDB**: license Apache, écrit en JAVA
  - **TITAN:** Projet Open Source sous license Apache

Récap

	Clé-Valeur	Colonne	Document	Graphe
Principe	Données représentées par des couples clé/valeur.	Représentation par colonnes et non par lignes. Données hiérarchisées .	Données stockées sous forme de collections de documents.	Données représentés sous forme de nœuds, de liens et des leurs propriétés.
Avantages	+ Modèle de données simple + Scalabilité	+ Supporte les données semi-structurées +Pas de stockage de valeurs nulles	+ Modèle de données simple mais puissant (structures imbriquées)	+ Adapté pour les données complexes et interconnectées + Algorithmes des graphes applicables
Inconvénients	-Très limité pour les données complexes -Interrogation sur clé seulement	- Structure un peu difficile à comprendre - Non-adaptée aux données interconnectées	- Non adaptée aux données interconnectées	- Non performants pour les grandes agrégations des données

Classement db-engines

- Classement «db-engines.com» (Décembre 2018) des SGBD selon leur popularité:

341 systems in ranking, December 2018

Rank Dec 2018	Rank Nov 2018	Rank Dec 2017	DBMS	Database Model	Score Dec 2018	Score Nov 2018	Score Dec 2017
1.	1.	1.	Oracle	Relational DBMS	1283.22	-17.89	-58.32
2.	2.	2.	MySQL	Relational DBMS	1161.25	+1.36	-156.82
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1040.34	-11.21	-132.14
4.	4.	4.	PostgreSQL	Relational DBMS	460.64	+20.39	+75.21
5.	5.	5.	MongoDB	Document store	378.62	+9.14	+47.85
6.	6.	6.	IBM Db2	Relational DBMS	180.75	+0.87	-8.83
7.	7.	↑ 8.	Redis	Key-value store	146.83	+2.66	+23.59
8.	8.	↑ 10.	Elasticsearch	Search engine	144.70	+1.24	+24.92
9.	9.	↓ 7.	Microsoft Access	Relational DBMS	139.51	+1.08	+13.63
10.	10.	↑ 11.	SQLite	Relational DBMS	123.02	+0.31	+7.82
11.	11.	↓ 9.	Cassandra	Wide column store	121.81	+0.07	-1.40

Références

- NoSQL Les bases de données et le Big Data, Rudi Bruchez, Eyrolles 2015
- Introduction aux systèmes NoSQL, Bernard ESPINASSEn (Professeur à l'Ecole Polytechnique Universitaire de Marseille)
- Introduction à la gestion de données, Pierre Senellart (Ecole normale supérieure, France)