NoSQL

Master Professionnel en Systèmes d'Information

Option Ingénierie des Systèmes d'Information

Octobre 2019

SOMMAIRE

II. Le mouvement NoSQL

Le mouvement NoSQL

- **Modèle relationnel vs Modèle NoSQL**
  - SGBDR
  - ACID vs BASE
- **Le Théorème CAP**
- **Les systèmes NoSQL**
  - Orienté Clé/valeur
  - Orienté Colonne
  - Orienté Document
  - Orienté graphe

Modèle relationnel vs Modèle NoSQL

- **SGBDR**
  - Les SGBDR sont généralement transactionnels
  - Les transactions sont soumises aux contraintes **ACID**
    - **A**tomicité : Une transaction s'effectue entièrement ou pas du tout
    - **C**ohérence : Le contenu d'une base doit être cohérent au début et à la fin d'une transaction
    - **I**solation : Les modifications d'une transaction ne sont visibles/modifiables que quand celle-ci a été validée
    - **D**urabilité : Une fois la transaction validée, l'état de la base est permanent (non affecté par les pannes ou autre)

Modèle relationnel vs Modèle NoSQL

- **SGBDR - Limites**
  - Incapable de gérer de très grands volumes de données (de l'ordre du péta-octet)
  - Impossible de gérer des débits extrêmes (supérieur à quelques milliers de requêtes par seconde)
  - Les propriétés ACID entraînent de sérieux surcoûts en latence, accès disques, temps CPU (verrous, journalisation, etc.)

Modèle relationnel vs Modèle NoSQL

- **SGBDR - Limites**
  - Une plateforme de e-business à grande échelle comme Amazon, indisponible ne serait-ce que quelques minutes n'est pas acceptable car elle causerait la perte de dizaine de milliers de commandes.
  - Pour garantir la **performance** et la **haute disponibilté**, une telle plateforme doit être déployée sur une **architecture distribuée**.
  - Les contraintes **ACID** sont difficilement conciliables avec de bonnes performances dans un système distribué.

Modèle relationnel vs Modèle NoSQL

- **ACID vs BASE**
  - Le **NoSQL** ne respecte pas toutes les contraintes **ACID** et se caractérise beaucoup plus par des propriétés dites **BASE**.
    - **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

Modèle relationnel vs Modèle NoSQL

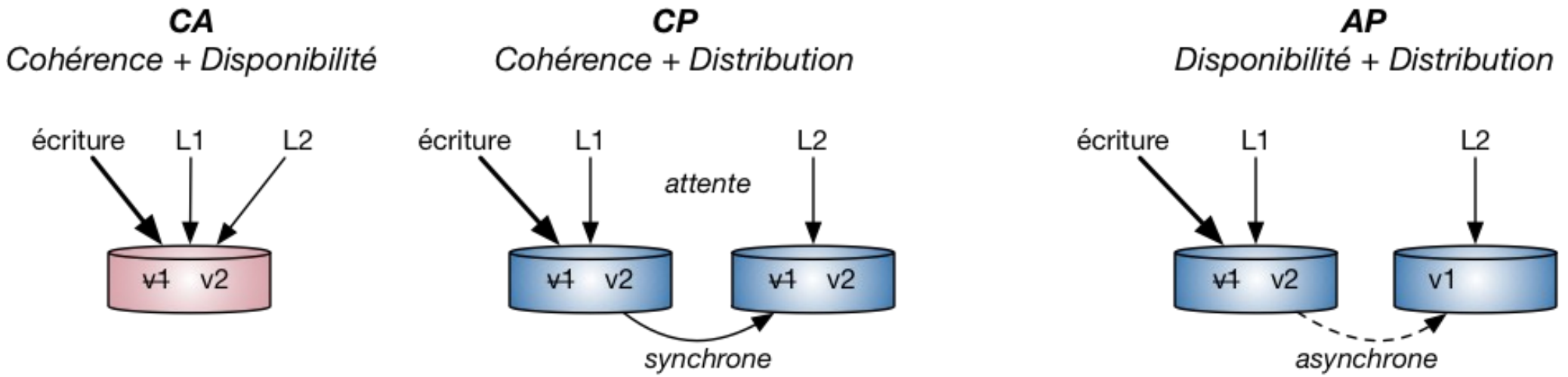
- **ACID vs BASE**

Le théorème CAP

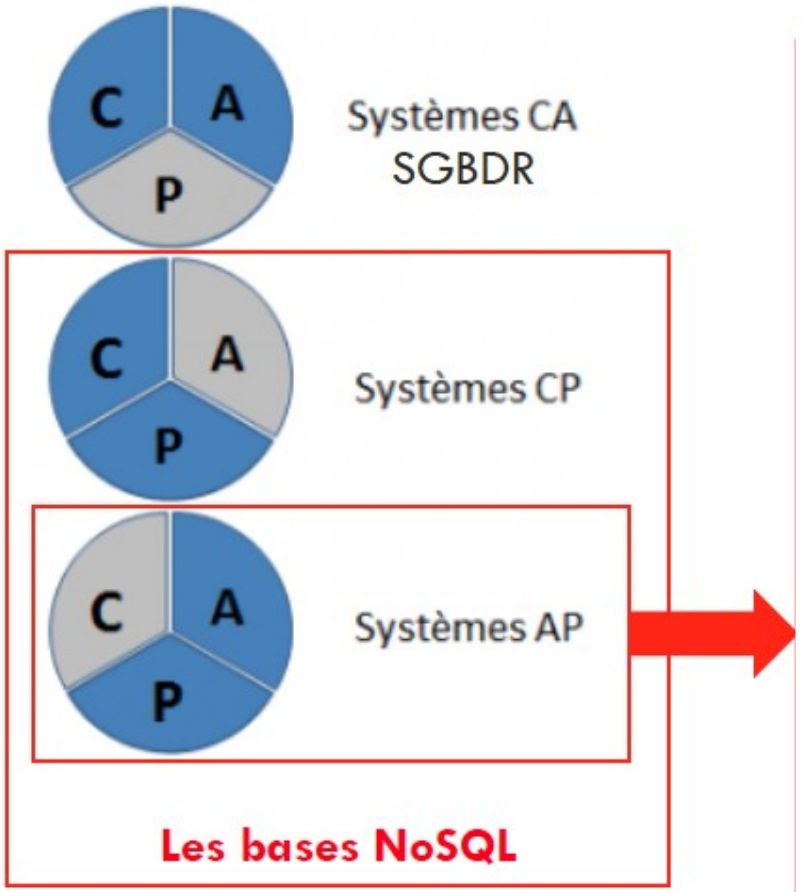
- En 2000, Eric A. Brewer a formalisé un théorème qui repose sur 3 propriétés fondamentales pour caractériser les bases de données (relationnelles, NoSQL et autres) :
  - **C**onsistency (**Cohérence**) : Une donnée n'a qu'un seul état visible quel que soit le nombre de réplicas
  - **A**vailability (**Disponibilité**) : Tant que le système tourne (distribué ou non), la donnée doit être disponible
  - **P**artition Tolerance (**Distribution**) : Quel que soit le nombre de serveurs, toute requête doit fournir un résultat correct

Le théorème CAP

- **Le théorème CAP** dit :
  - Dans toute base de données, vous ne pouvez respecter **au plus que 2 propriétés** parmi la **cohérence**, la **disponibilité** et la **distribution**.



Le théorème de CAP



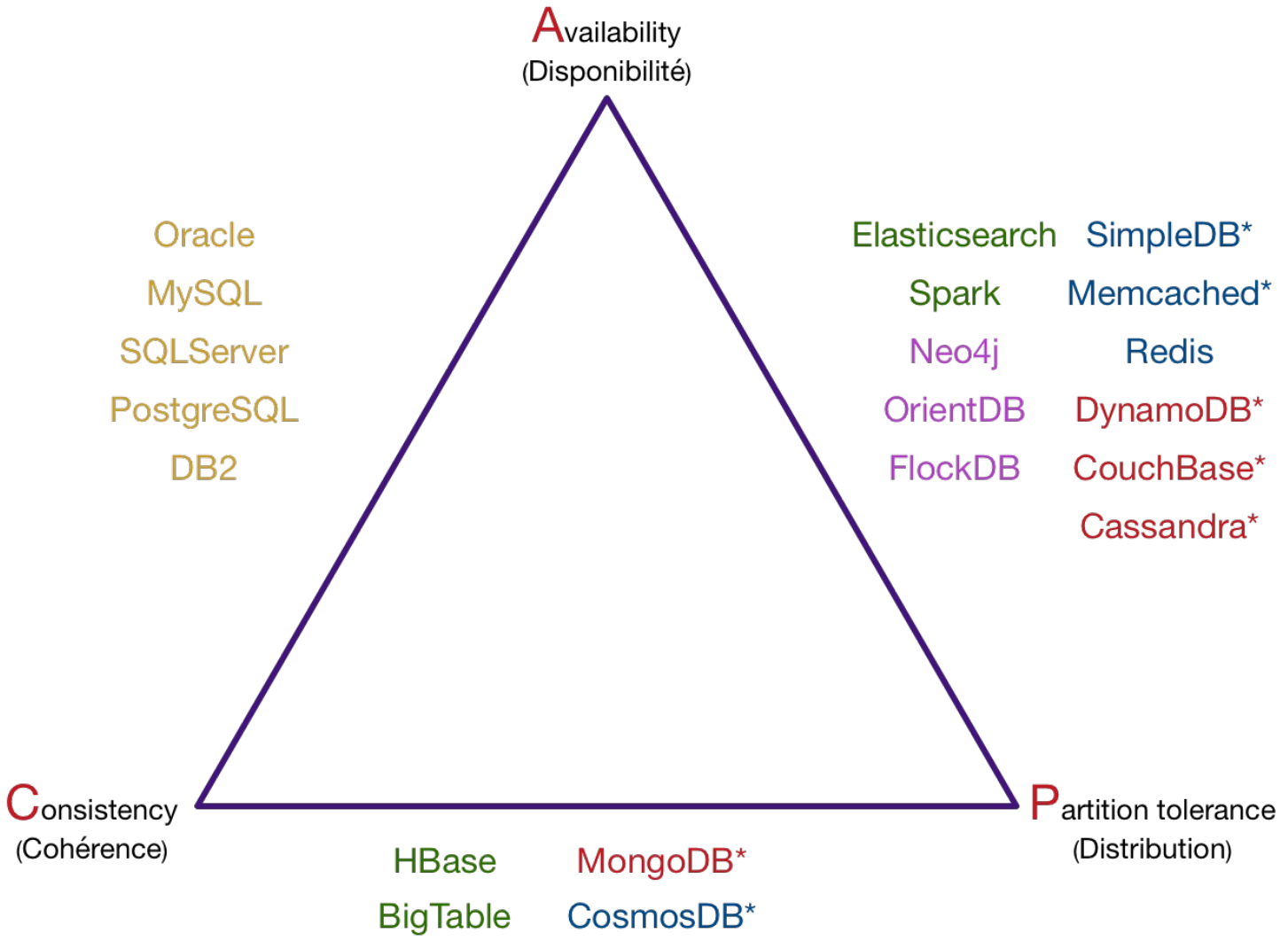
- Souvent ce sont les deux derniers principes qui sont choisis (A et P)
- C'est par exemple le cas de Amazon
- Le besoin en disponibilité et en partitionnement est souvent plus important que la cohérence.

Le théorème CAP

Modèle

Relationnel
Orienté Clé-Valeur
Orienté Colonne
Orienté Document
Orienté Graphe

* Possibilité de changer
le mode de cohérence
Cohérence <-> Disponibilité

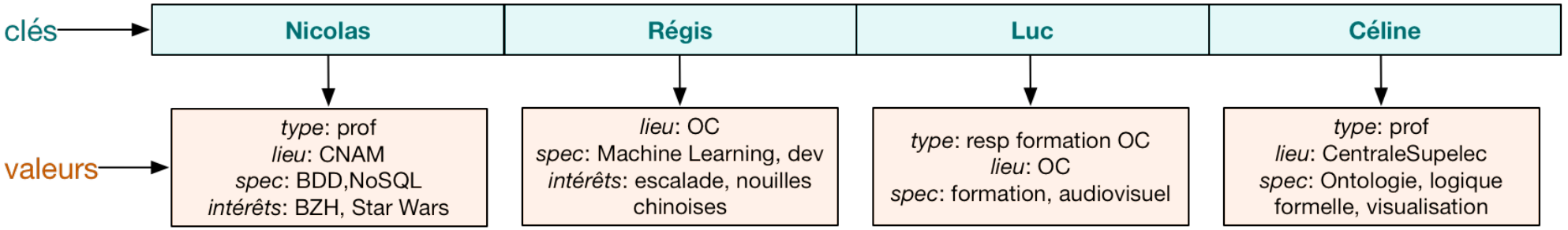


Les systèmes NoSQL

- **Orienté clé/valeur**
  - Base NoSQL la plus simple
  - La base peut être assimilée à une table de hachage distribuée
  - Les données sont simplement représentées par un couple (clé, valeur)
  - La valeur = chaîne de caractères, un objet sérialisé, …
  - Absence de structure ou typage

Les systèmes NoSQL

- **Orienté clé/valeur**
  - Chaque objet est identifié par une clé unique :
    - Seule façon de le requêter !
  - On ne peut pas effectuer de requête sur le contenu des objets stockés



Les systèmes NoSQL

- **Orienté clé/valeur**
  - L'exploitation des données est basée sur 4 opérations (CRUD) :
  - `Create(key,value)` : créer un nouveau couple (clé, valeur)
  - `Read(key)` : lire un objet à partir de sa clé
  - `Update(key,value)` : mettre à jour la valeur d'un objet à partir de sa clé
  - `Delete(key)` : supprimer un objet à partir de sa clé

Les systèmes NoSQL

- **Orienté clé/valeur**
  - **Exemples d'utilisations :**
    - Dépôt de données avec besoins de requêtage très simples
    - Profils, préférences utilisateur
    - Données de panier d'achat
    - Données de capteurs

Les systèmes NoSQL

- **Orienté clé/valeur**
  - **Exemples de base :**
    - **DynamoDB** : Utilisée pour gérer le panier d'achat sur Amazon
    - **Riak** : Implémentation open source d'Amazon DynamoDB. Utilisé par Mozilla, AOL, Yammer, WorkShare, …
    - **Voldemort** : Développée et utilisée par LinkedIn

Les systèmes NoSQL

- **Orienté clé/valeur**
  - **Avantages**
    - Modèle de données simple
    - Aucune maintenance
    - Performances élevées en lecture et en écriture
    - **Scalabilité horizontale** considérable
    - **Disponibilité**

Les systèmes NoSQL

- **Orienté clé/valeur**
  - **Inconvénients**
    - Modèle de données trop simple
    - Pas de langage de requêtes (uniquement sur clés)
    - Complexité déportée vers l'applicatif

Les systèmes NoSQL

- **Orienté colonne**
  - Les données sont stockées sous forme de table
  - Correspond beaucoup plus à un entrepôt de stockage de données
  - Les attributs sont regroupés en famille de colonnes
  - Deux attributs qui sont fréquement utilisés ensemble seront stockés au sein d'une même famille de colonnes.
  - Le nombre de colonne est dynamique, il varie d'une ligne (enregistrement) à l'autre. Ce qui évite de retrouver des colonne ayant une valeur null

Les systèmes NoSQL

- **Orienté colonne**

Représentation de ventes en relationnel

Table Sales

#ticket	#date	#book
1	01/01/16	2212121504
1	01/01/16	2212141556
2	01/01/16	2212141556

Table Book

#isbn	#title	#author
2212121504	Scenari	1
2212141556	NoSQL	2

Table Author

#id	surname	firstname
1		
2	Bruchez	Rudi

Représentation de ventes en colonne

Family Sales

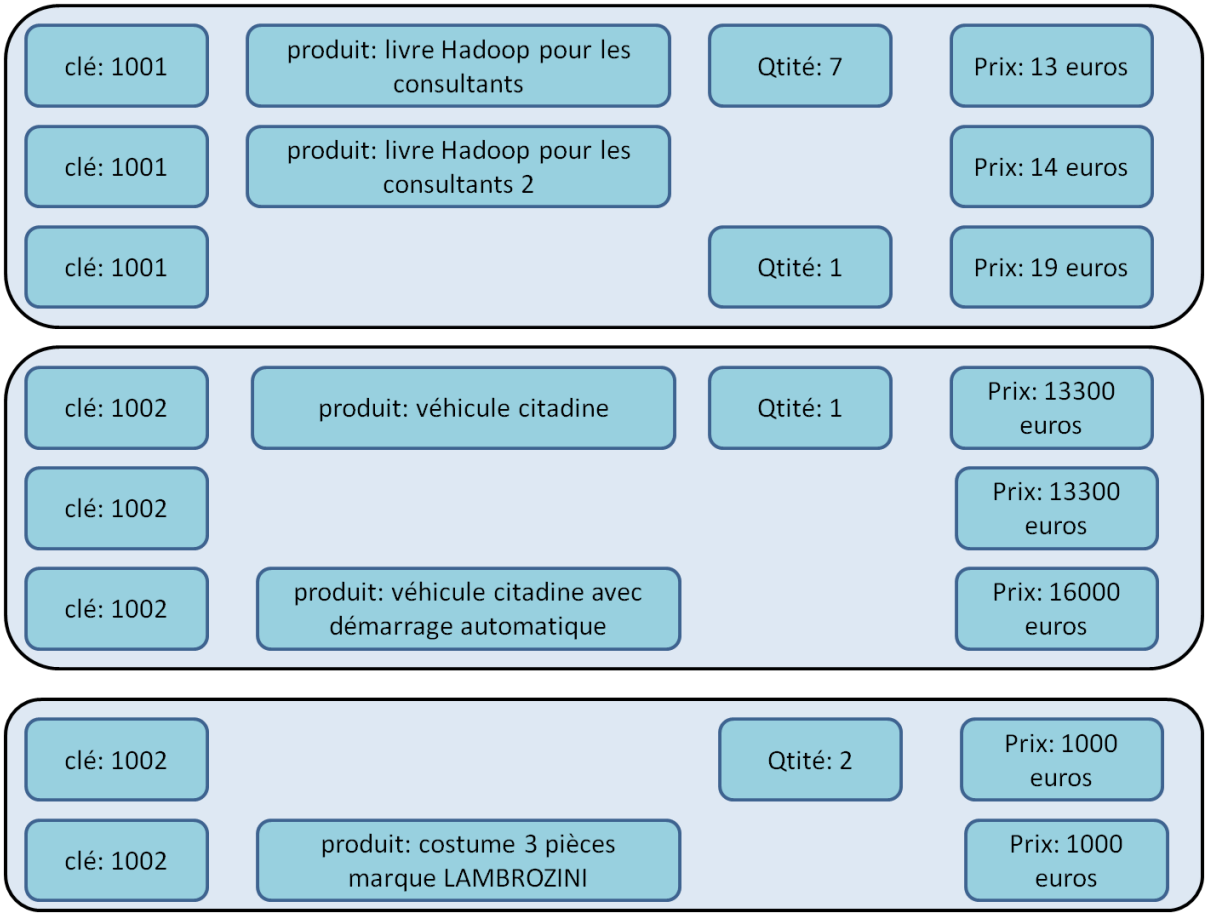
#ticket	date	books
1	01/01/16	2212121504 2212141556
2	01/01/16	2212141556

Family Book

#isbn	title	a-surname	a-firstname
2212121504	Scenari		
2212141556	NoSQL	Bruchez	Rudi

Les systèmes NoSQL

- **Orienté colonne – Cas de HBase**
  - Dynamisme des colonnes dans HBase



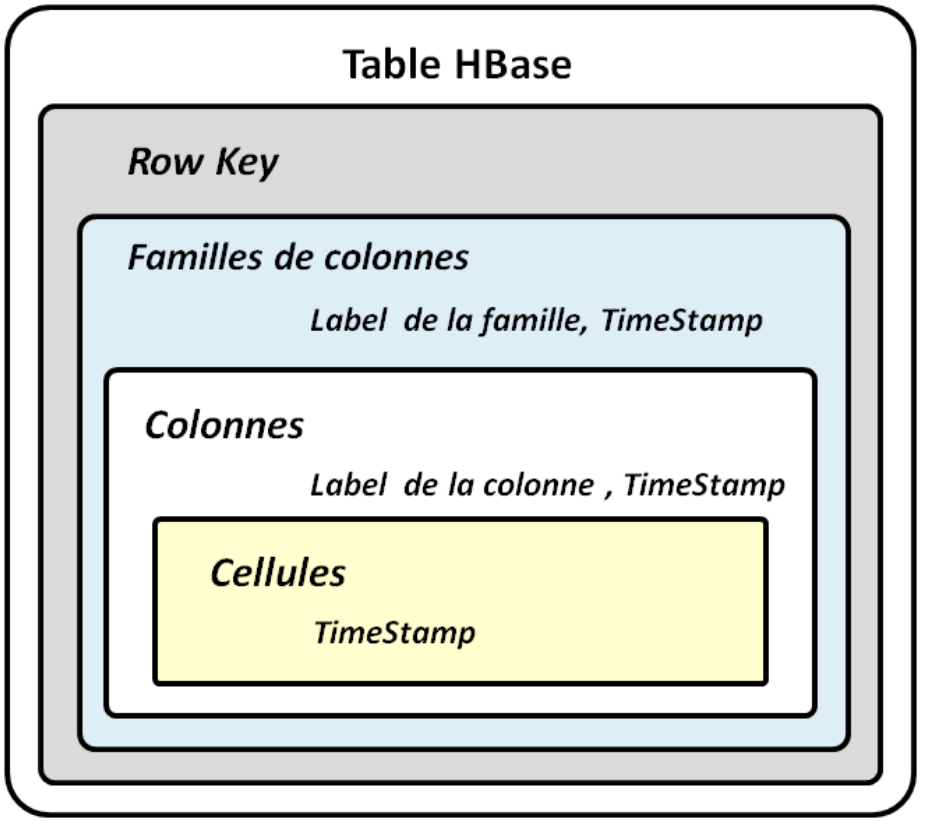
Les systèmes NoSQL

- **Orienté colonne – Cas de HBase**
  - Structure logigue d'une table Hbase

ROW KEY	TIMESTAMP	COLUMN FAMILY	COLUMN NAME	VALUE
Clé de la ligne	Date de création ou de mise à jour de la ligne	Famille : colonne	Nom de la colonne	Valeur ou donnée stockée dans la cellule

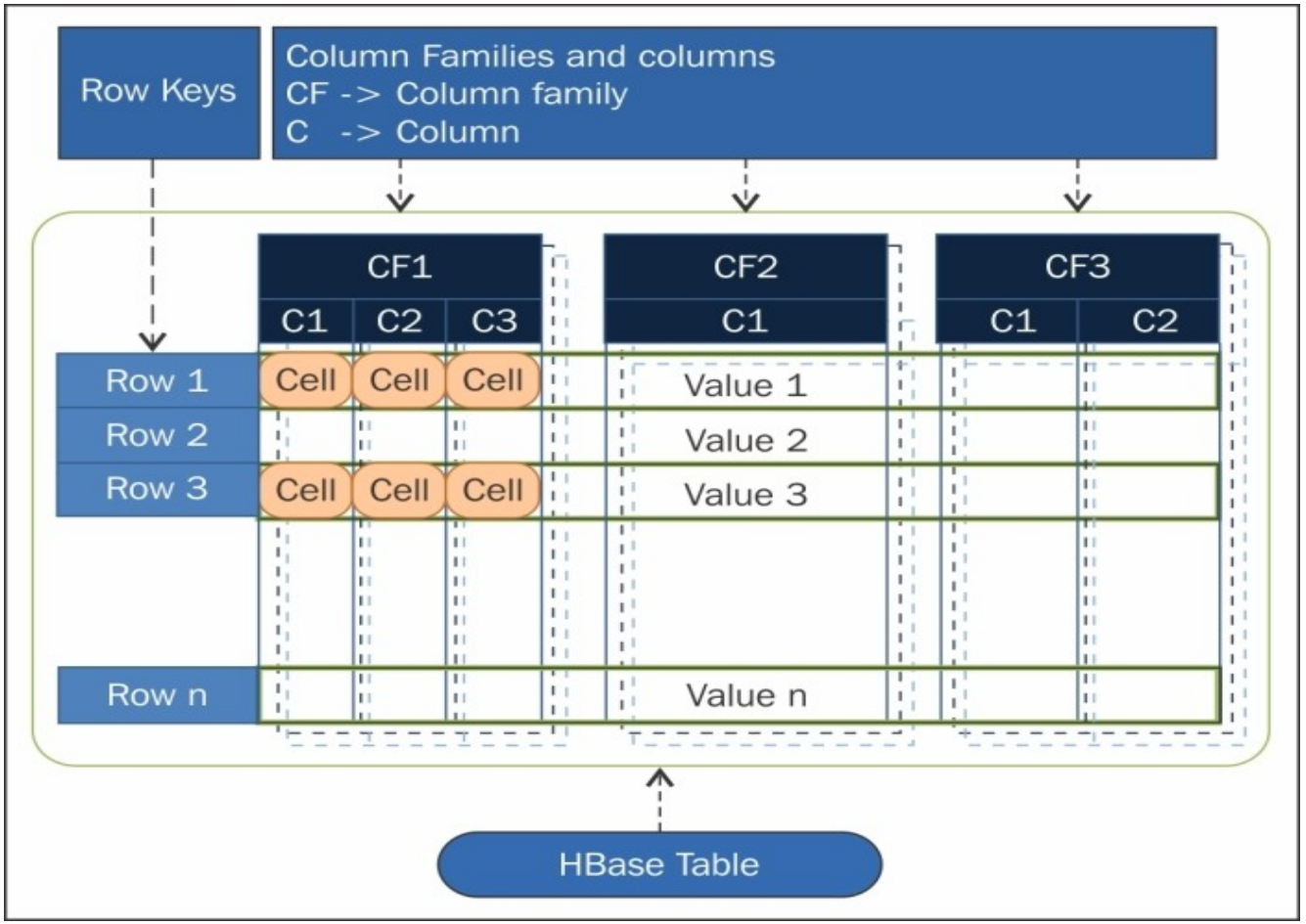
Les systèmes NoSQL

- **Orienté colonne – Cas de HBase**
  - Représentation hierarchique de la structure d'une table HBase



Les systèmes NoSQL

- **Orienté colonne – Cas de HBase**
  - Représentation des données dans une table HBase



Les systèmes NoSQL

- **Orienté colonne – Cas de HBase**
  - Représentation des données dans une table HBase

	Personnel		Contact	
Row Key	**Prénom**	**Age**	**téléphone**	**ville**
00001	Juvénal ***timestamp***: 10/09/2011 13:05:17:09	22 ***timestamp***: 10/09/2011 13:05:17:09	06 90 98 76 52 ***timestamp***: 10/09/2011 13:05:17:09	Douala ***timestamp***: 10/09/2011 13:05:17:09
		25 ***timestamp***: 20/09/2013 15:00:05:09		Lille ***timestamp***: 20/09/2013 15:00:05:09
				Paris ***timestamp***: 30/10/2016 16:18:50:10
00002	Paul ***timestamp***: 10/09/2011 13:10:05:09	30 ***timestamp***: 10/09/2011 13:10:05:09	07 90 94 86 52 ***timestamp***: 10/09/2011 13:10:05:09	Nancy ***timestamp***: 10/09/2011 13:10:05:09
00003	Jean ***timestamp***: 12/09/2011 11:30:20:09	34 ***timestamp***: 12/09/2011 11:30:20:09	06 74 98 76 25 ***timestamp***: 12/09/2011 11:30:20:09	Marseille ***timestamp***: 12/09/2011 11:30:20:09

Les systèmes NoSQL

- **Orienté colonne**
  - **Exemples d'utilisations :**
    - Utilisation pour le logging et l'analyse de la clientèle (Netflix)
    - Optimisation de la recherche (Ebay)
    - Utilisation par les sociétés TV pour analyser les audiences TV et gérer le vote des spectateurs
    - Bons outils d'analyse des données semi-structurées

Les systèmes NoSQL

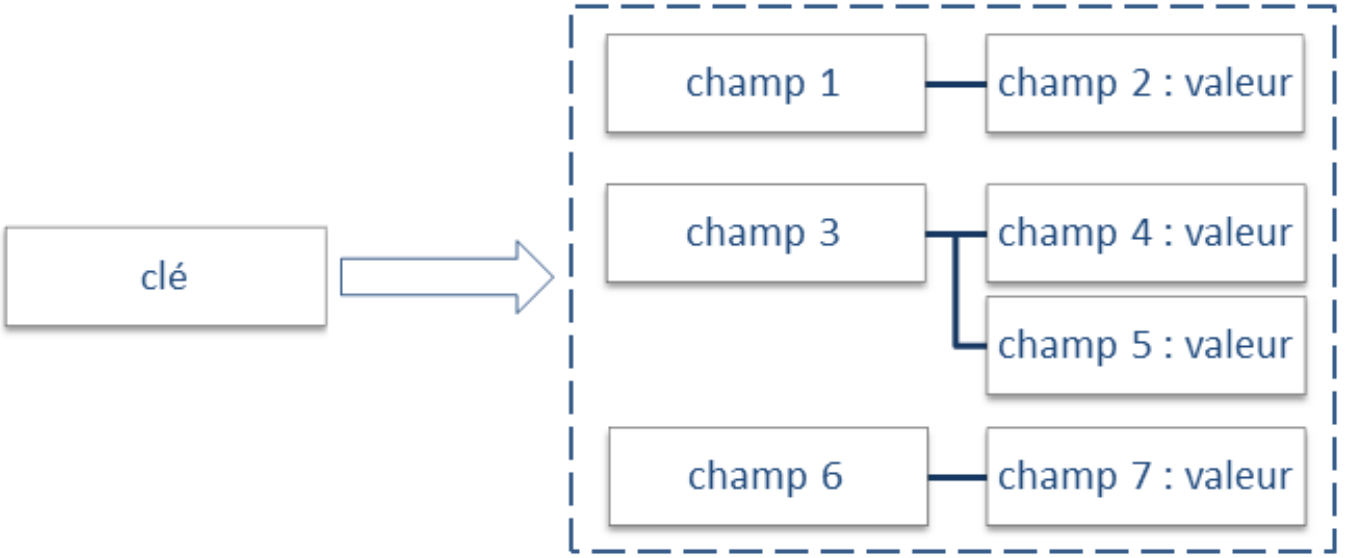
- **Orienté colonne**
  - **Exemples de bases :**
    - **HBase** (Apache, Hadoop)
    - **Cassandra** (FaceBook)
    - **BigTable** (Google)

Les systèmes NoSQL

- **Orienté colonne**
  - **Avantages :**
    - Données semi-structurés
    - Indexation incluse (colonnes)
    - Requêtage en temps-réel
    - **Scalabilité horizontale**
  - **Inconvénients :**
    - Peu adapté aux données complexes
    - Peu adapté aux données interconnectées

Les systèmes NoSQL

- **Orienté document**
  - Stockage d'une collection de documents
  - Basé sur le modèle (clé, valeur)
  - La valeur est un document en format semi-structuré
  - Semi-structuré = structure arborescente de type **JSON**



Les systèmes NoSQL

- **Orienté document**
  - Le format **JSON** (JavaScript Object Notation) facile l'échange ou la réutilisation des données

```
{
  "cle": "valeur",
  "cle2": ["tableau","d'","éléments"],
  "cle3" : {"js" :"JavaScript",
            "o" :"Object",
            "n" :"Notation"
           }
}
```

Les systèmes NoSQL

- **Orienté document**

```
{
    "titre":"Imitation Game",
    "summary":"Imitation Game retrace l'histoire tragique d'Alan Turing, \n
                un mathématicien britannique spécialisé dans la cryptologie, ou l'art \n
                de briser des codes secrets. Pendant la Seconde Guerre mondiale, Alan \n
                Turing s'est illustré en déchiffrant les codes générés par Enigma, la \n
                machine que les nazis utilisaient pour correspondre. (...)",
    "year":2015,
    "director":{"last_name": "Tyldum",
                "first_name": "Morten"},
    "actors":[
                {"last_name": "Cumberbatch", "first_name": "Benedict"},
                {"last_name": "Knightley", "first_name": "Keira"}
            ]
}
```

Les systèmes NoSQL

- **Orienté document**
  - **Exemples d'utilisations :**
    - Enregistrement d'événements
    - Systèmes de gestion de contenu
    - Web analytique ou analytique temps-réel
    - Catalogue de produits

Les systèmes NoSQL

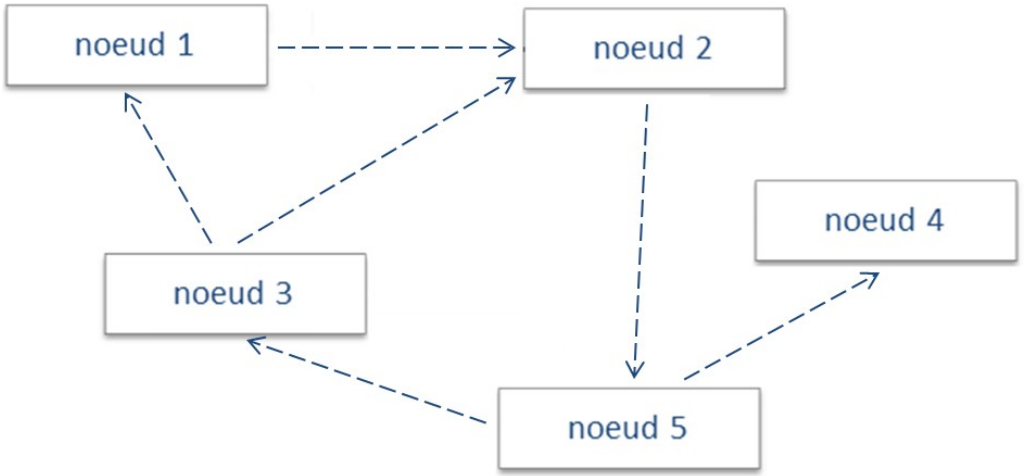
- **Orienté document**
  - **Exemples de bases :**
    - **CouchDB** : Fondation Apache
    - **RavenDB** : Pour plateformes .NET/Windows
    - **MongoDB** : Développé par 10gen

Les systèmes NoSQL

- **Orienté document**
  - **Avantages :**
    - Modèle simple mais à performances élevées
    - Bonne mise à l'échelle (**scalabilité**)
    - Efficace pour les interrogations par clé
  - **Inconvénients :**
    - Peu adapté aux données interconnectées
    - Requêtage limitée aux clés
    - Lent pour les grandes requêtes

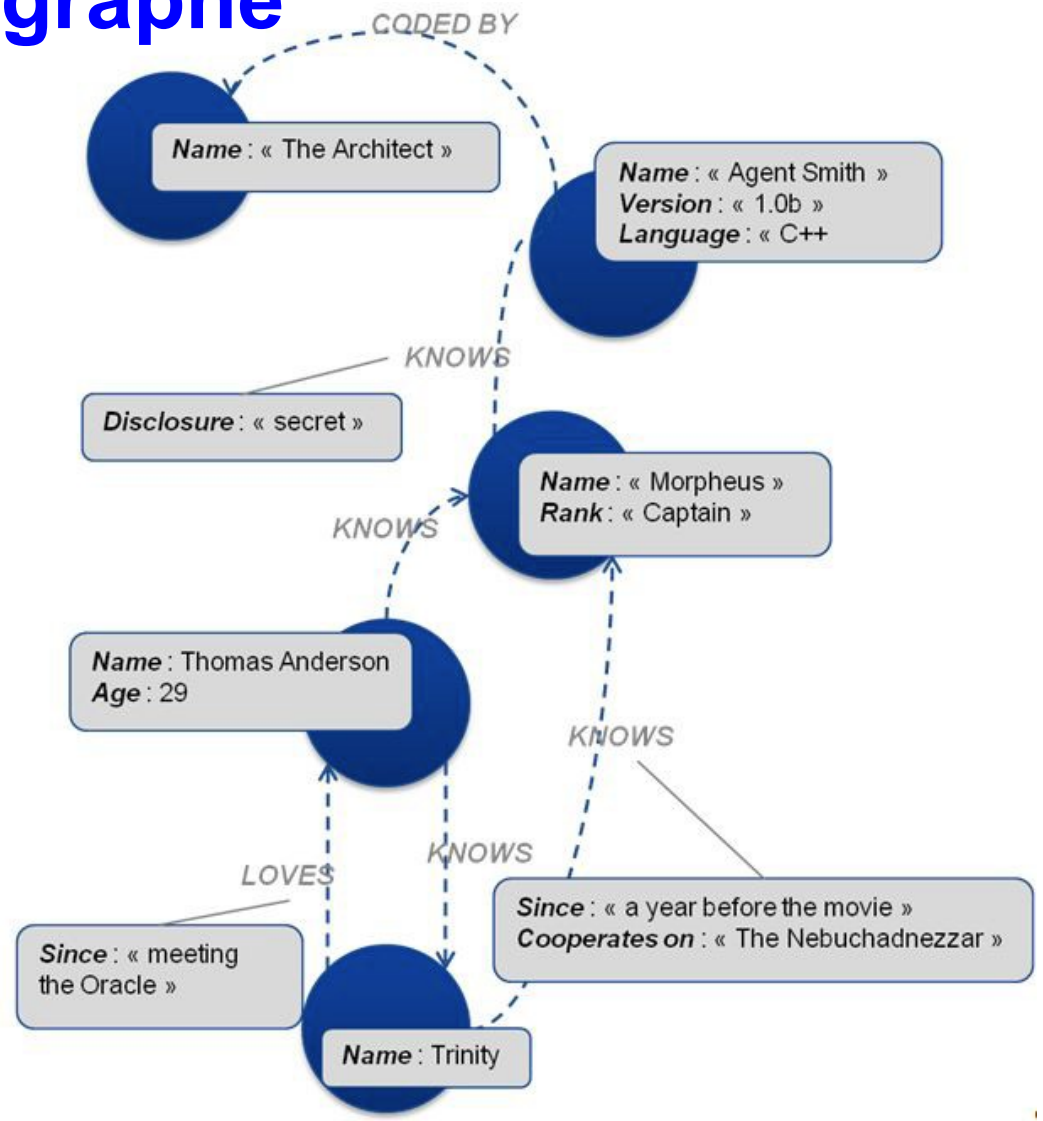
Les systèmes NoSQL

- **Orienté graphe**
  - Forme avancée du modèle (clé, valeur)
  - Fondées sur la théorie des graphes
  - Modélisation, stockage et manipulation de données complexes liées par des relations non-triviales ou variables
  - Notions de noeuds, relations et propriétés



Les systèmes NoSQL

- **Orienté graphe**



Les systèmes NoSQL

- **Orienté graphe**
  - un graphe est un ensemble de **sujets-prédicats-objets**
  - Modélisation grâce à trois blocs de base :
    - Le **noeud** ou sommet (node, vertex)
    - La **relation** ou arête (relationship, edge), avec une orientation et un type (orienté et marqué)
    - La **propriété** ou attribut (property, attribute), portée par un noeud ou une relation

Les systèmes NoSQL

- **Orienté graphe**
  - **Exemples d'utilisations :**
    - Web sémantique
    - Moteurs de recommandation
    - Données sociales
    - Cartographie
    - Données géo-spatiales
    - Détection de fraudes

Les systèmes NoSQL

- **Orienté graphe**
  - **Exemples de bases :**
    - **Neo4j** : Développée en java par NeoTechnology et distribuée sous licence AGPL
    - **OrientDB** : Développée en java et distribuée sous licence Apache

Les systèmes NoSQL

- **Orienté graphe**
  - **Avantages :**
    - Modèle de données puissant
    - Modèle de données riche
    - Modèle de données adapté aux situations où il faut modéliser beaucoup de relations
    - Modèle d'interrogations standardisés et performants
  - **Inconvénients :**
    - répartition des données peut être problématique pour de gros volumes de données