Les familles NoSQL

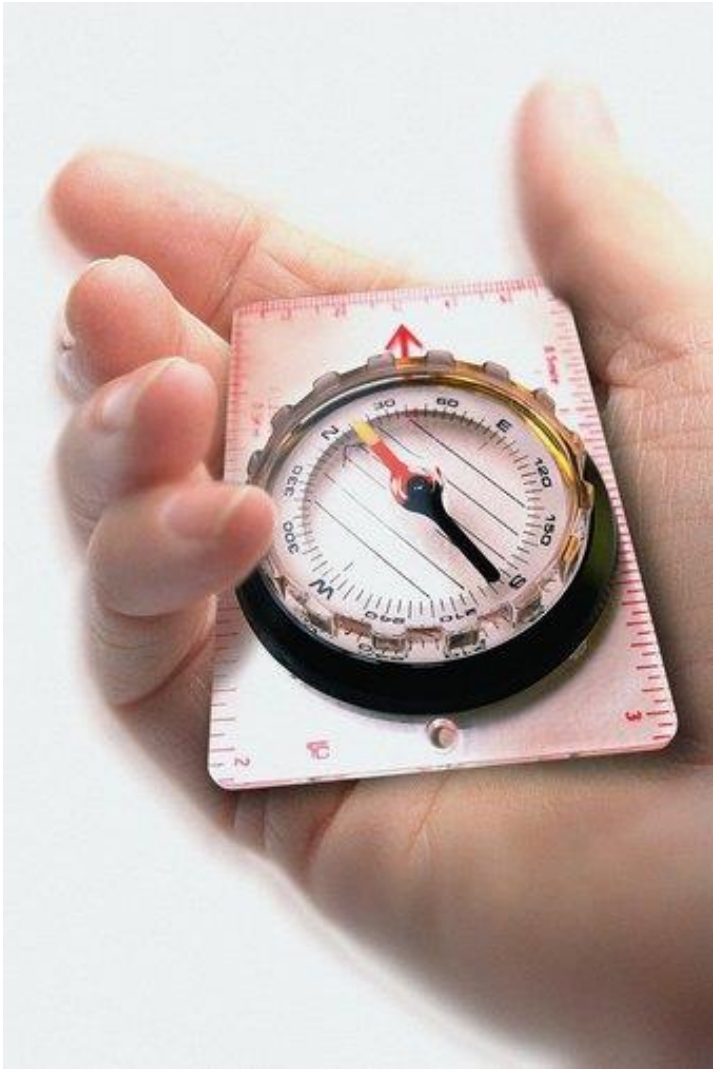
Cours 2

Clé-valeur, colonne, document et graphe

Wilfried Y. H. Adoni

Université Internationale de Casablanca

Objectif



- Établir le rôle de NoSQL
- Stockage orienté Key-Value
- Stockage orienté document
- Stockage en familles de colonnes
- Stockage orienté graphe

Établir le rôle de NoSQL

Université Internationale de Casablanca

Contenu du chapitre

En suivant ce cours, vous pourrez:

- Présenter les différents types de stockage de données Big Data
  - Key-value
  - Document
  - Column Family
  - Graph
- Acquérir de l'expérience en utilisant les frameworks Big Data, y compris
- Hadoop
- Neo4j

Stockage Big Data

Le dilemme du stockage massif

- Grâce au stockage peu coûteux, les entreprises ont la possibilité de stocker plus de données que jamais auparavant.
- Ils ont également accès à plus de types de données
  - Structuré
  - Semi-Structuré
  - Non structuré

Université Internationale de Casablanca

Le dilemme du stockage

- Relever les défis du Big Data est un défi technique complexe
  - Le volume
  - Variété
  - Rapidité
  - Véracité
- La solution de stockage doit satisfaire un certain nombre de demandes

Université Internationale de Casablanca

La demande de stockage de données Big Data

- Capacité
  - Être capable de supporter l'évolution rapide des données
- Sécurité
  - Une grande partie des données est conforme aux normes de sécurité
    - Financier, médical
  - L'intégration pour l'analyse en temps réel augmente le défi de la sécurité

La demande de stockage de données Big Data

- Latence
  - De nombreuses solutions ont une composante temps réel
  - Besoin d'évoluer et toujours fournir une faible latence
- Flexibilité
  - La conception doit permettre à l'infrastructure de stockage d’être agile
  - Incorporer les possibilités de migration de données

Université Internationale de Casablanca

La demande de stockage de données Big Data

- Persistance
  - Longévité souvent nécessaire pour une analyse temporelle
  - La conformité réglementaire exige souvent que les données soient sauvegardées pendant de nombreuses années.
  - Les périphériques de stockage peuvent nécessiter des fonctionnalités de fiabilité à long terme intégrées
- Coût
  - Principalement construit autour de matériel de base peu coûteux
  - Les coûts peuvent toujours être importants lorsque le volume augmente

Université Internationale de Casablanca

La demande de stockage de données Big Data

- Les bases de données relationnelles constituent le cœur du stockage des données d'application depuis de nombreuses années
  - Et continuera à l'être
- Les bases de données relationnelles constituent le cœur du stockage des données d'application depuis de nombreuses années
  - Structure bien définie avec des schémas à tables fixes
  - Les propriétés des données peuvent être définies au début
  - Les relations entre les données sont clairement définies
  - Les données sont denses et généralement uniformes
  - Les index peuvent être définis pour tirer parti de l'interrogation rapide

Université Internationale de Casablanca

Mise à l'échelle des bases de données relationnelles

- Les bases de données relationnelles sont conçues pour s'exécuter sur des machines uniques
- La mise à l'échelle signifie généralement que vous utilisez une machine plus grande
  - Des approches alternatives ont été développées:
    - Clustering, telles que Oracle Real Application Clusters (RAC)
    - Sharding — différents jeux de données sur différentes machines

Université Internationale de Casablanca

Mise à l'échelle des bases de données relationnelles

- L’inadéquation entre les bases de données relationnelles et les clusters a entraîné l’émergence de solutions
  - Google
  - Amazone
- Ces deux organisations disposent de données à une échelle dépassant celle de la plupart des organisations Big Data!

Université Internationale de Casablanca

Émergence des données de stockage alternatifs

- Un nouvel ensemble de données de stockage a émergé au cours des dernières années
  - La motivation principale était de supporter de grands ensembles de données
- Les caractéristiques communes de ces entrepôts de données incluent:
  - Échelle pour les grands ensembles de données
  - Ne pas avoir de modèle relationnel
  - Pas de schéma fixe permettant de stocker des données
  - Big Data demande de stockage
  - Exécuter sur des cluster de machines à faible coût

NoSQL

- Le NoSQL ne remplacent pas les bases de données SQL
  - Ils les complètent
- Il existe maintenant un choix de plusieurs technologies de stockage de données disponibles
  - Devrait être choisi en fonction de la manière dont les données sont utilisées
  - Conduit à ce qu'on appelle la persistance polyglotte

Université Internationale de Casablanca

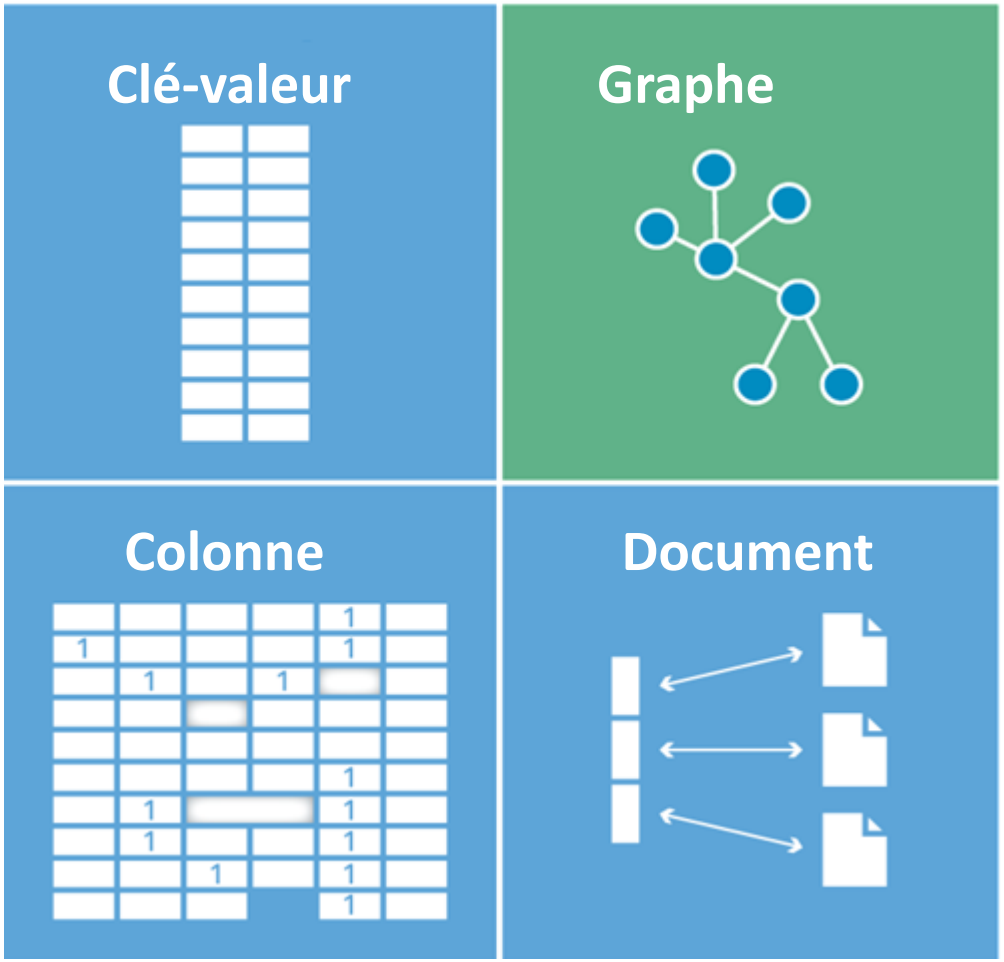
Les différentes familles du modèle NoSQL

Le modèle NoSQL est regroupés en 4 familles.

- Chacune de ces familles répond à des besoins très spécifiques, avec pour but *in fine*, de vous permettre de choisir votre solution NoSQL.

Les familles NoSQL

- ❑ Stockage clés-Valeurs
- ❑ Stockage orienté colonne
- ❑ Stockage orienté document
- ❑ Stockage orienté graphe



Stockage orienté Key–Value

Stockage Key-Value

- Ce sont les plus simples magasins de données NoSQL
- Certains sont persistants, d'autres sont en mémoire seulement
- Soutenir trois opérations principales
  - Insérer une valeur pour une clé
  - Obtenir une valeur associée à une clé
  - Supprimer une valeur

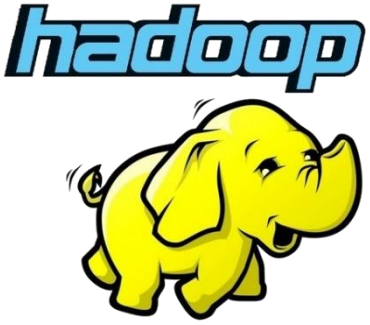
Stockage Key-Value

- Il existe un certain nombre de magasins de données de clé-valeur disponibles
  - Hadoop
  - Riak
  - Redis
  - Amazon S3 et DynamoDB
  - Memcached
  - Infinispan
- Nous utiliserons **Hadoop** dans nos exemples pour représenter les types de fonctionnalités à attendre

Université Internationale de Casablanca

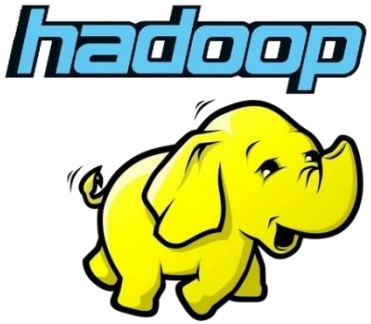
Hadoop

- Framework open source
- Distribution Apache
- Disponible sur https://hadoop.apache.org/
- Les valeurs stockées peuvent être:
  - Strings
  - Lists
  - Hashes
  - Sets (y compris les ensembles triés)



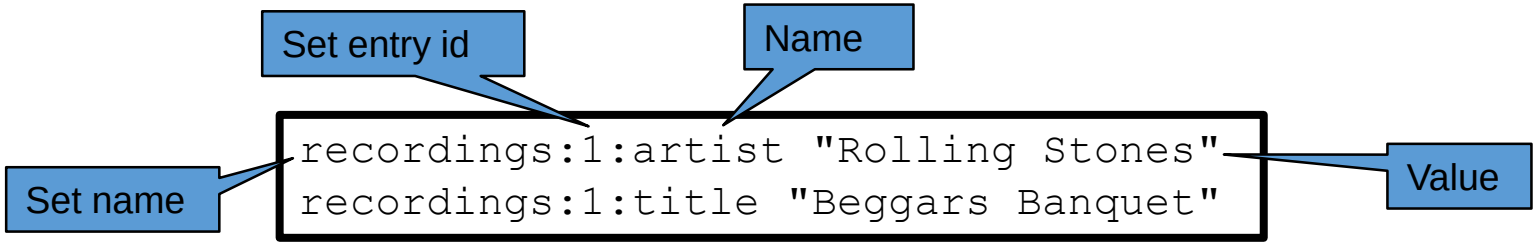
Hadoop

- Prise en charge de nombreux langages de programmation
  - Java (Native Language)
  - C/C++
  - Python



Hadoop

- Chaque enregistrement de la série aura un identifiant unique
  - Généralement un nombre
  - Assure que chaque enregistrement de musique est dans le jeu de données qu'une seule fois
- Les données de chaque enregistrement de l'ensemble sont constituées de paires nom-valeur.



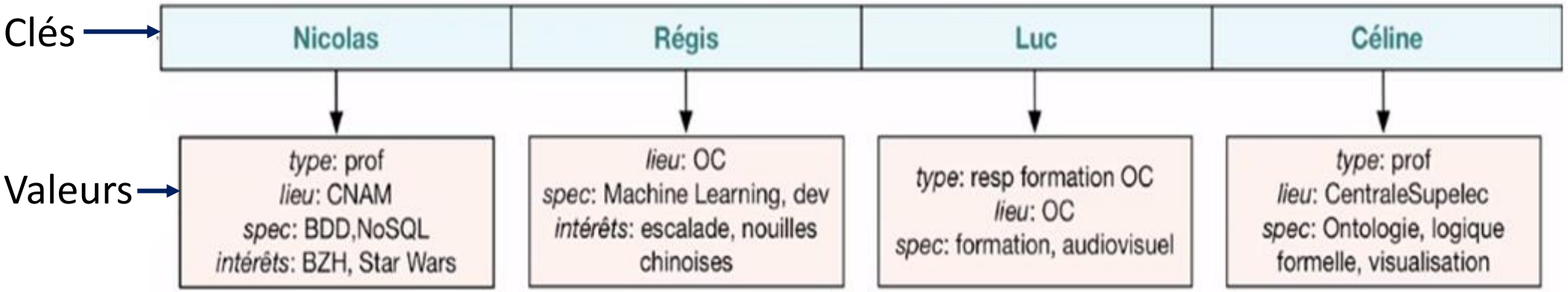
Université Internationale de Casablanca

NoSQL: orienté clés-valeurs

Modèle SQL

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	

Modèle NoSQL: clés-valeurs

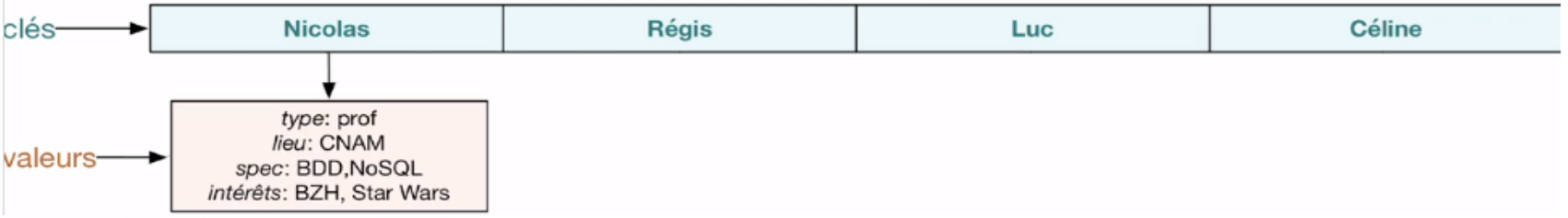


NoSQL: orienté clés-valeurs

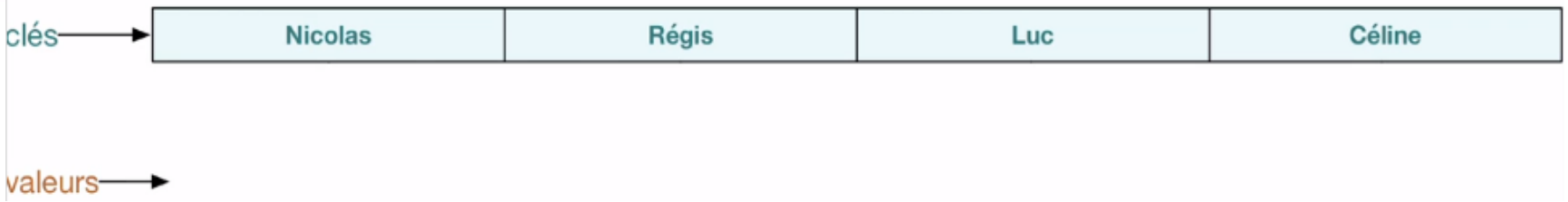
Seules les opérations de type CRUD peuvent être utilisées :

- CREATE (clé, valeur)
  - CREATE ("Nicolas", "type:'prof',lieu:'CNAM',spec:'BDD,NoSQL',interets:'BZH,Star Wars' ") → OK

- READ(clé)
  - READ("Nicolas") → "type:'prof',lieu:'CNAM',spec:'BDD,NoSQL',interets:'BZH,Star Wars' "
- UPDATE(clé, valeur)
  - UPDATE("Nicolas", "type:'prof',lieu:'CNAM,CS',spec:'BDD,NoSQL' ") → OK

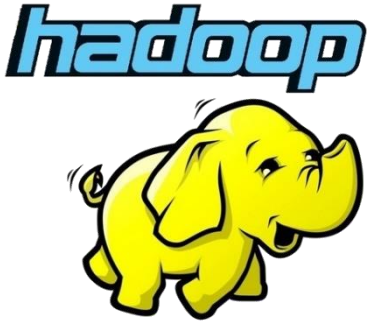


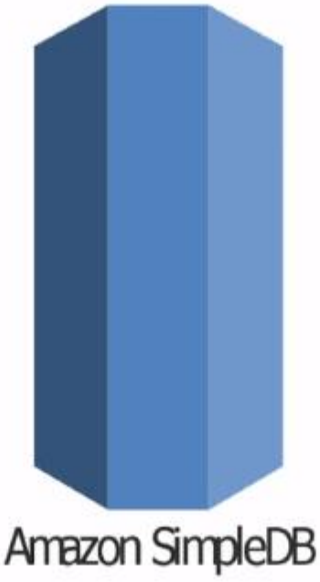
- DELETE(clé)
  - DELETE("Nicolas") → OK



Université Internationale de Casablanca

Solutions basées sur le modèle clés-valeurs





Efficacité

Facilité de mise en œuvre

Stockage orienté Document

Stockage orienté document

- Permet le stockage d'informations semi-structurées (orientées document)
- Les documents contiennent des données codées
  - Données texte: XML, YAML, JSON, BSON
  - Données binaires: PDF, MS Word, etc.
- Les documents de données sont adressés par une clé unique
  - Peut également être récupéré en fonction de leur contenu
  - Par exemple, un enregistrement de musique auquel on a accédé par le titre de l'enregistrement

Université Internationale de Casablanca

Stockage orienté document

- Considérez les deux documents suivants:
  - Ils n'adhèrent à aucun schéma / structure prédéfinie

```
{ artist: "Rolling Stones",
  title: "Beggars Banquet",
  price: 9.99
}
```

JSON document

```
{ artist: "Rolling Stones",
  title: "Beggars Banquet",
  tracks:[
   {title: "Dear Doctor"},
   {title: "Factory Girl" }
  ]
}
```

Colon separates keys from values

BSON = Binary JSON
JSON = JavaScript Object Notation
PDF = Portable Document Format

XML = extensible markup language
YAML = Yet Another Multicolumn Layout

Université Internationale de Casablanca

Stockage orienté document

- De nombreux données de documents sont disponibles
  - CouchDB
  - MongoDB
  - Terrastore
  - OrientDB
  - RavenDB
  - Jackrabbit
- Nous utiliserons MongoDB dans nos exemples pour représenter les types de fonctionnalités à attendre

Université Internationale de Casablanca

MongoDB

- Documents open source: www.mongodb.org
- Les principales caractéristiques comprennent:
  - Modèle de données enrichi non contraint par le schéma
  - Mise à l'échelle facile par mise à l'échelle
  - Réplication et haute disponibilité
  - Interrogation riche
  - Agrégation flexible et traitement de données

MongoDB

- Prise en charge de nombreux langages de programmation
  - C/C++
  - Java
  - JavaScript
  - .NET
  - Node.js
  - PHP
  - Python
  - Ruby

mongoDB.

Travailler avec MongoDB

- Les documents sont stockés dans la base de données MongoDB
  - Une base de données comprend une ou plusieurs collections
    - Chaque collection peut avoir plusieurs documents
- Les données stockées sont identifiés par leur nom
  - Sélectionnez un magasin de données: `use <data-store-name>`
    - Par exemple, `use rainforest`
    - Créé s'il n'existe pas
- Une fois le magasin de données sélectionné, il est référencé sous le nom `db`

Interrogation avec MongoDB

- L'exemple ajoutera un document d'enregistrement musical au stockage de données: ajouté à la collection `recordings`:

```
use rainforest

album = {
        id: 1,
        artist: "Rolling Stones",
        title: "Beggars Banquet",
        price: 9.99
}

db.recordings.insert(album)

db.recordings.find()

db.recordings.remove({id:1})
```

Use `rainforest` data store

Create document named `album`

Insert document into collection `recordings`

Remove document with id `1` from `recordings`

Interrogation avec MongoDB

- La méthode `find()` permet d'effectuer une recherche dans les documents
  - Par exemple, pour trouver tous les enregistrements dont l'artiste est Rolling Stones

```
db.recordings.find({artist : "Rolling Stones"})
```

Université Internationale de Casablanca

Interrogation avec MongoDB

- Les requêtes peuvent également spécifier des critères tels que des plages et des combinaisons logiques

  - Par exemple, pour trouver tous les enregistrements dont le prix est compris entre 5,00 $ et 7,00 $

```
db.recordings.find({price: {"$gte":5.00, "$lte": 7.00}})
```

Range specified as document

Interrogation avec MongoDB

- L'opérateur non égal autorise la correspondance sur des clés différentes de la valeur spécifiée

```
db.recordings.find({artist: {"$ne":"Rolling Stones"}})
```

All artists except Rolling Stones

Université Internationale de Casablanca

NoSQL: orienté documents

Modèle SQL

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	

Modèle NoSQL: documents

Clés → Nicolas | Régis | Luc | Céline

Documents →

```
{
  "_id": "Nicolas",
  "type": "prof",
  "lieu": {
      "nom": "CNAM",
      "adresse":"rue saint martin",
      "ville": "Paris"
  },
  "spec": ["BDD","NoSQL"],
  "intérêts": ["BZH",
               "Star Wars"]
}
```

```
{
  "_id": "Régis",
  "lieu": {
      "nom": "OC",
      "adresse":"rue cité Paris",
      "ville": "Paris"
  },
  "spec": ["Machine Learning",
           "dev"],
  "intérêts": ["escalade",
               "nouilles"]
}
```

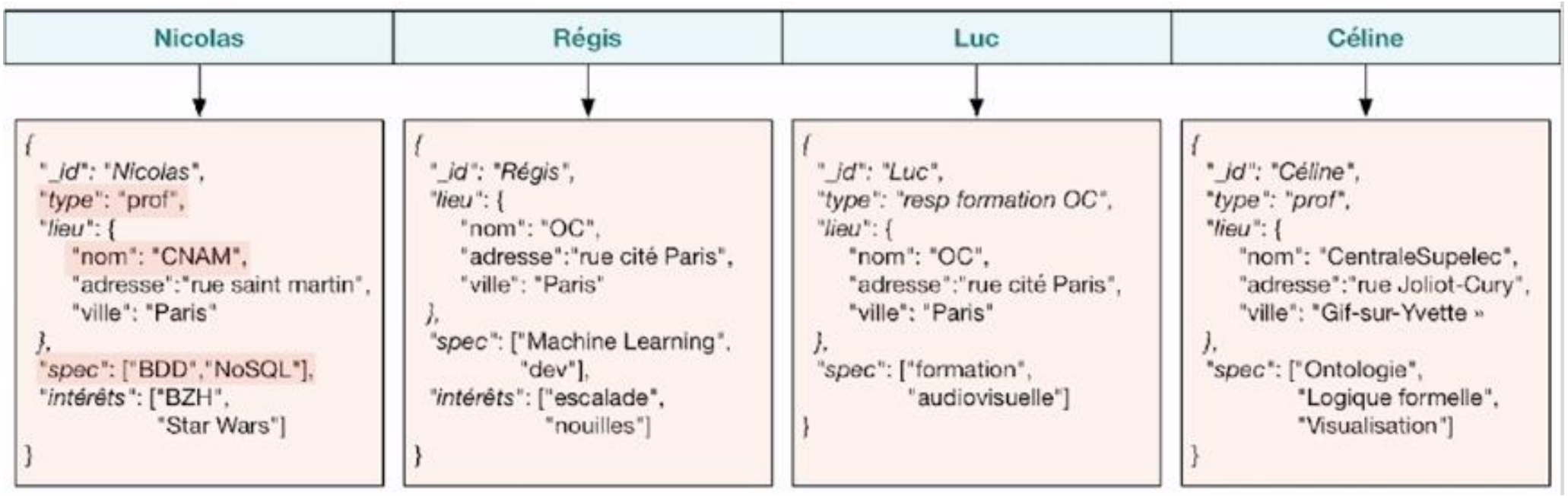
```
{
  "_id": "Luc",
  "type": "resp formation OC",
  "lieu": {
      "nom": "OC",
      "adresse":"rue cité Paris",
      "ville": "Paris"
  },
  "spec": ["formation",
           "audiovisuelle"]
}
```

```
{
  "_id": "Céline",
  "type": "prof",
  "lieu": {
      "nom": "CentraleSupelec",
      "adresse":"rue Joliot-Cury",
      "ville": "Gif-sur-Yvette »
  },
  "spec": ["Ontologie",
           "Logique formelle",
           "Visualisation"]
}
```

NoSQL: orienté documents

Seules les requêtes sur le contenu des documents sont possibles :

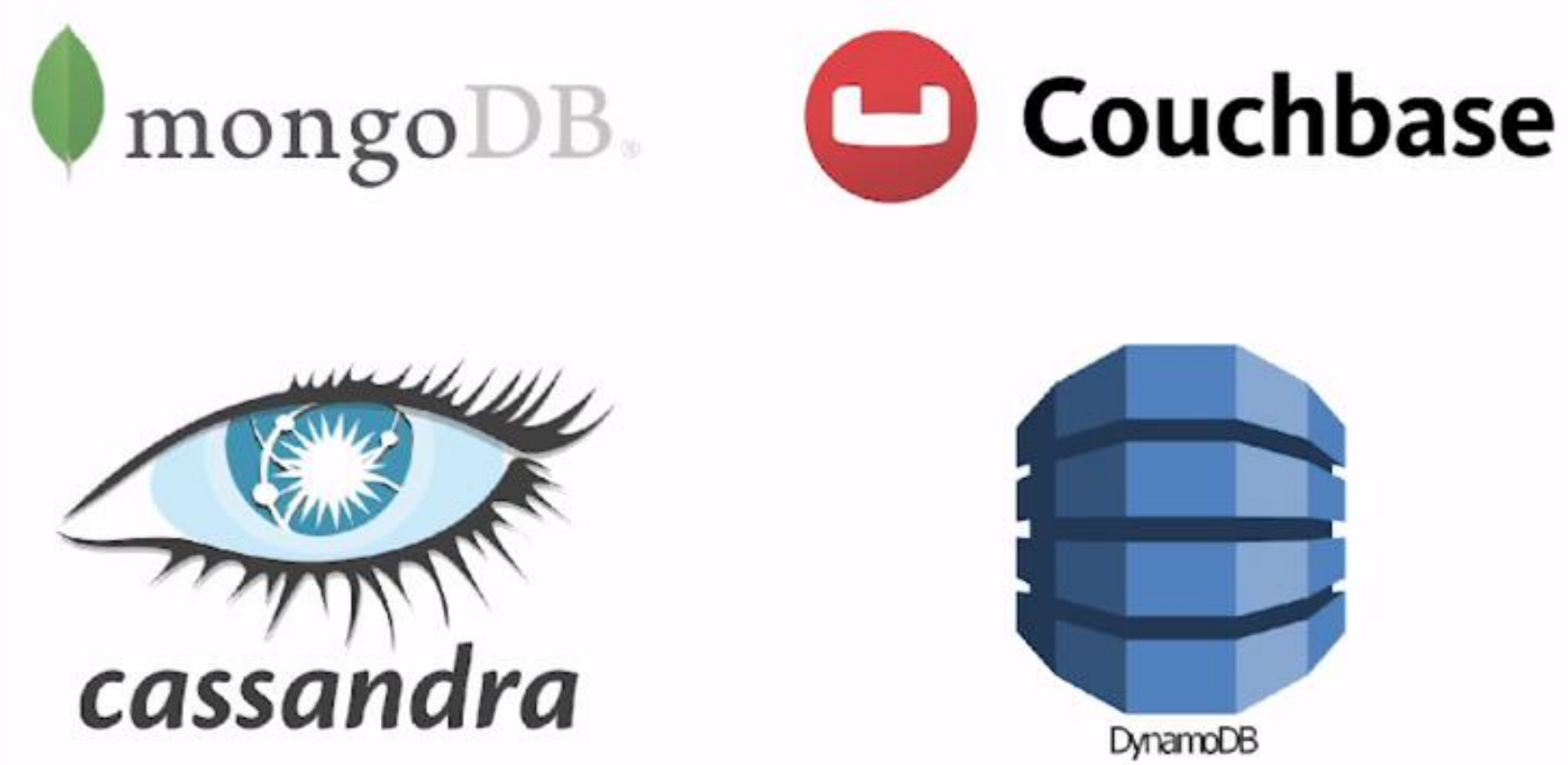
- Etablissement (lieu.nom) des professeurs (type) spécialisé en BDD (*in* spec)



Interrogation avec MongoDB

- Applications qui stockent des données dans des fichiers plats
  - Les données fournies avec une fonctionnalité de recherche structurée
- Les données sont complexes à modéliser dans une base de données relationnelle
- L'application a de gros volumes d'appariement d'enregistrements
  - Compensation commerciale
  - Détection de fraude
  - Rapprochement des transactions

Solutions basées sur le modèle documents



Requêtes riches

Gestion d'objets

Stockage orienté Famille de Colonne

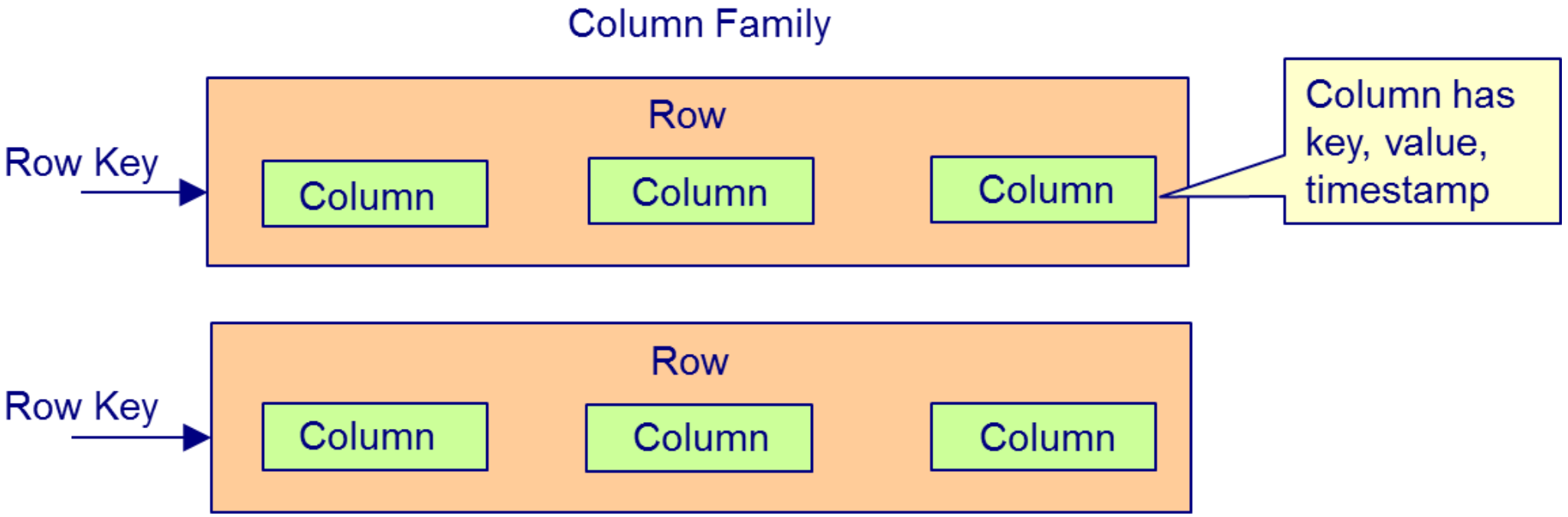
Université Internationale de Casablanca

Stockage orienté famille de colonne

- Stocker les données sous forme de tableau mais sans schéma formel
  - L'unité de stockage principale est un *keyspace*
  - Chaque ligne a une clé unique *key*
- Chaque ligne peut contenir plusieurs colonnes
  - Connu comme une *column family*
    - Paires clé-valeur associées stockées dans une seule ligne de la famille de colonnes

Université Internationale de Casablanca

Stockage orienté famille de colonne



Cassandra

- Utilisation principale: pour les très grands ensembles de données
  - Les lignes peuvent avoir des millions de colonnes
  - Variable / flexible quant aux données à stocker
- Nombre de magasins de données de familles de colonnes disponibles
  - Cassandra
  - HBase
  - Hypertable
  - Amazon SimpleDB

- Nous allons utiliser Cassandra dans nos exemples

Université Internationale de Casablanca

Cassandra

- Prise en charge de nombreux langages de programmation
  - Java
  - Python
  - PHP
  - Perl
  - C #

Université Internationale de Casablanca

Travailler avec Cassandra

- Cassandra a trois niveaux d'adressage
  - *keyspace* contenant des familles de colonnes
  - *Column Family* contenant des colonnes
  - *Column* contenant des valeurs
- Travailler avec Cassandra nécessite
  - Créer un *keyspace*
  - Créer une *Column Family*
  - Ajout de données à une *Column*

Travailler avec Cassandra

- L'exemple ci-dessous montre la création d'un *keyspace*:

```
CREATE KEYSPACE rainforest;
```

- Création d'une famille de colonnes = définition du schéma de DB
  - Le schéma de *Column Family* peut changer dynamiquement

```
CREATE COLUMN FAMILY recordings
WITH comparator = UTF8Type
AND key_validation_class = UTF8Type
AND column_metadata = [
{column_name: Artist, validation_class: UTF8Type}
{column_name: Price, validation_class: DoubleType}
];
```

For searching

Key will be string

Data type of column

Université Internationale de Casablanca

Lecture et Ecriture des données

- Les commandes de base **GET**, **SET** et **DEL** peuvent être exécutées à l'aide d'un client en ligne de commande

```
SET recordings['Beggars Banquet']['Price'] = '9.55';
SET recordings['Beggars Banquet']['Artist'] = 'Rolling Stones';
```

Column family

Row key

Column

```
GET recordings['Beggars Banquet'];
```

Get column data for Beggars Banquet

Université Internationale de Casablanca

Lecture et Ecriture des données

- Tous les enregistrements peuvent être listés en utilisant la commande **LIST**

```
LIST recordings;
```

- Les *keyspace* et les *Column Family* peuvent être supprimés à l'aide de la commande **DROP**

```
DROP columnfamily recordings;
```

Remove column family `recordings`

Cassandra Query Language (CQL)

- Semblable au SQL pour créer des familles de colonnes et accéder aux données
- Pour créer un *keyspace*, utilisez:

```
CREATE KEYSPACE rainforest
WITH replication={'class':'SimpleStrategy',
'replication_factor':'1'};
```

Data is in one data center only

Only replicate to one other node

- Sélectionner un *keyspace* à l'aide de la commande **USE**

```
USE rainforest;
```

Université Internationale de Casablanca

Lecture et écriture de données avec CQL

- Une famille de colonnes peut être créée avec la commande **CREATE COLUMNFAMILY**

```
CREATE COLUMNFAMILY recordings(
Title varchar PRIMARY KEY,
Artist varchar,
Price double);
```

Will be title of recording

- Pour insérer des données, utilisez la commande **INSERT**

```
INSERT INTO  recordings(Title, Artist, Price) values
('Beggars Banquet', 'Rolling Stones', 9.99);
```

Université Internationale de Casablanca

Lecture et écriture de données avec CQL

- Pour récupérer des données, CQL fournit la commande **SELECT**

```
SELECT * FROM  recordings;
```

Retrieve all data from `recordings` column family

- Les espaces de clés et les familles de colonnes peuvent être supprimés à l'aide de la commande **DROP**

```
DROP COLUMNFAMILY recordings;
```

Université Internationale de Casablanca

NoSQL: orienté colonnes

Modèle SQL

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	UIC
Céline	prof	UIC	Ontologie, logique formelle, visualisation	

Modèle NoSQL: colonnes

id	type
Nicolas	prof
Céline	prof
Luc	resp formation OC

id	lieu
Céline	UIC
Nicolas	CNAM
Régis	OC
Luc	OC

id	spec
Nicolas	BDD
Nicolas	NoSQL
Régis	Machine Learning
Régis	Dev
Luc	formation
Luc	audiovisuel
Céline	Ontologie
Céline	logique formelle

id	intérêts
Nicolas	BZH
Nicolas	Star Wars
Régis	escalade
Régis	nouilles chinoises

NoSQL: orienté colonnes

Seules les requêtes sur les colonnes sont possibles :

- Combien de professeurs (type) à UIC (lieu)

id	type
Nicolas	prof
Céline	prof

id	lieu
Céline	UIC

id	spec
Nicolas	BDD
Nicolas	NoSQL
Régis	Machine Learning
Régis	Dev
Luc	formation
Luc	audiovisuel
Céline	Ontologie
Céline	logique formelle

id	intérêts
Nicolas	BZH
Nicolas	Star Wars
Régis	escalade
Régis	nouilles chinoises

Université Internationale de Casablanca

Solutions basées sur le modèle colonnes

Agrégations

Corrélations

Université Internationale de Casablanca

Cas d’application des familles de colonnes

- Enregistrement d'événements
- Les erreurs d'application peuvent être stockées dans Cassandra
- Permet de traiter les données pour analyse à l'aide de Hadoop
- Permet une recherche rapide sur les tags

Stockage orienté Graphe

Université Internationale de Casablanca

Stockage en graphe

- Les bases de données orientés graphes peuvent être interrogées de différentes manières
  - Par exemple, obtenez tous les nœuds de personnes qui vivent à New York et qui aiment The Killers
- Connu comme recherche de plus chemin dans un graphe
- Fournir une grande flexibilité dans l'exploration d'un graphe
- Aussi très haute performance

Stockage en graphe

- Ajouter de nouvelles relations à une base de données graphes est facile
  - La base de données relationnelle nécessite des modifications de schéma et un transfert de données
- Il existe un certain nombre de base de données graphes disponibles.
  - FlockDB
  - InfiniteGraph
  - Neo4j
  - OrientDB
- Nous allons utiliser Neo4j dans nos exemples

Neo4j

- Une base de données de graphes open-source: www.neo4j.org
- Les principales caractéristiques comprennent:
  - Durable
  - Support fiable pour les transactions ACID
  - Massivement évolutif
  - Hautement disponible
  - Rapide - Requêtes graphiques à grande vitesse
  - Simple à utiliser

Neo4j

- Prise en charge de nombreux langages de programmation
  - Java
  - Python
  - JRuby
  - Scala
  - Clojure

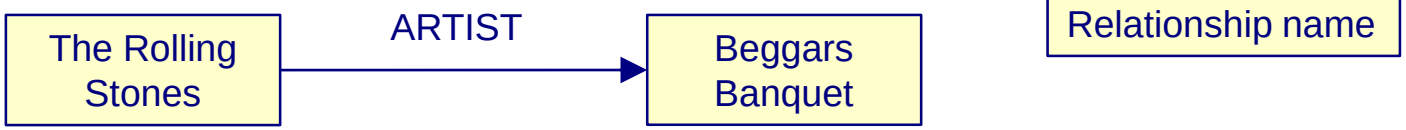
Travailler avec Neo4j

- L'exemple de code suivant crée deux nœuds
  - Ajoute une relation entre eux

Accesses music recording database

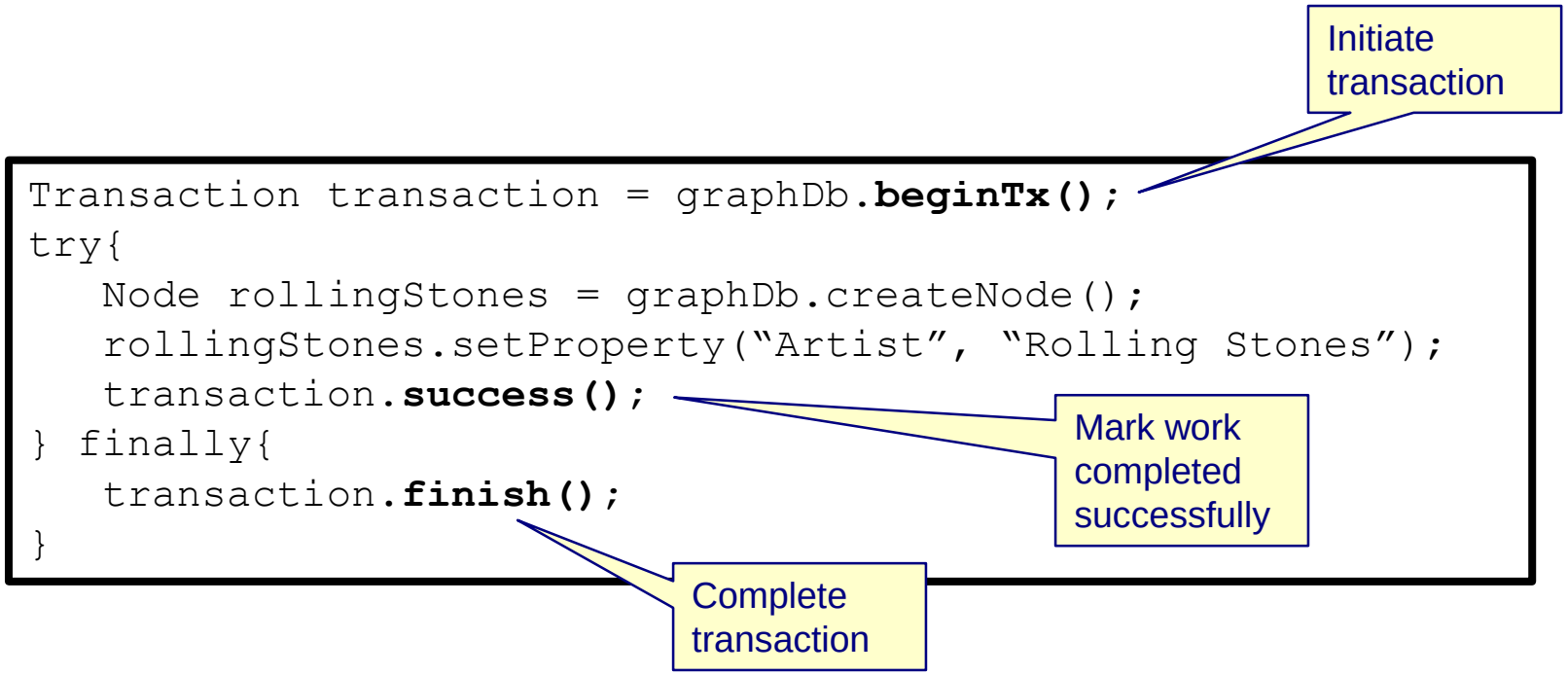
```
Node rollingStones = graphDb.createNode();
rollingStones.setProperty("Artist", "Rolling Stones");

Node beggarsBanquet = graphDb.createNode();
beggarsBanquet.setProperty("Title", "Beggars Banquet");

rollingStones.createRelationship(beggarsBanquet, ARTIST);
```

Relationship name



Transactions

- Neo4j est compatible ACID
  - Tous les changements doivent être effectués dans une transaction.
    - Sinon, une erreur se produira
  - La lecture peut être effectuée sans transaction



Caractéristiques de la requête Neo4j

- Fournit le langage de requête Cypher pour interroger le graphe
- La structure générale de Cypher est

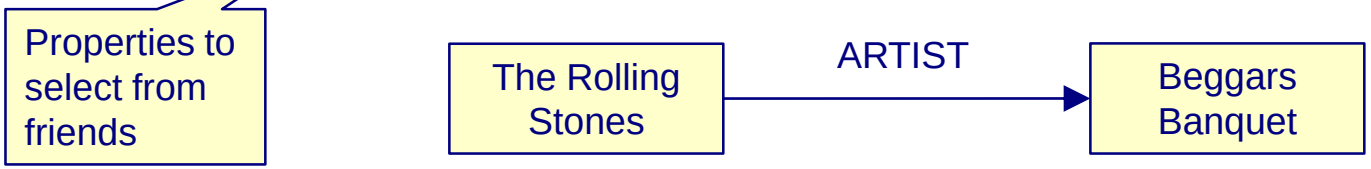
```
START beginningNode = (beginning node specification)
MATCH (relationship, pattern matches)
WHERE (filtering condition: on data in nodes and
relationships)
RETURN (what to return: nodes, relationships, properties)
ORDER BY (properties to order by)
SKIP (nodes to skip from top)
LIMIT (limit results)
```

Université Internationale de Casablanca

Caractéristiques de la requête Neo4j

- Lister tous les titres des enregistrements dont l'artiste est Rolling Stones

```
START rolling_stones = node(nodeId)
MATCH (rolling_stones)-[:ARTIST]-(recordings)
RETURN recordings.Title;
```



Exécution de requêtes à partir de Java

- Utilisez **`ExecutionEngine`** pour exécuter les requêtes Cypher

```
String query = "START rolling_stones = node(rollingStones.getId())" +
"MATCH (rolling_stones)-[:ARTIST]-(recordings)" +
"RETURN recordings.Title";

ExecutionEngine engine = new ExecutionEngine(graphDb);
ExcutionResult result = engine.execute(query);

// Process results here
```

Identify node by id

Execute query from Java code

Université Internationale de Casablanca

Cas d'utilisation des bases de données Graphe

- Données connectées
  - Représenter les employés et leurs compétences
  - Projets travaillés sur
- Services d'expédition pour la livraison au détail
  - Chaque livreur et la livraison est un noeud
  - Les relations peuvent avoir une propriété de distance

Université Internationale de Casablanca

Cas d'utilisation des bases de données Graphe

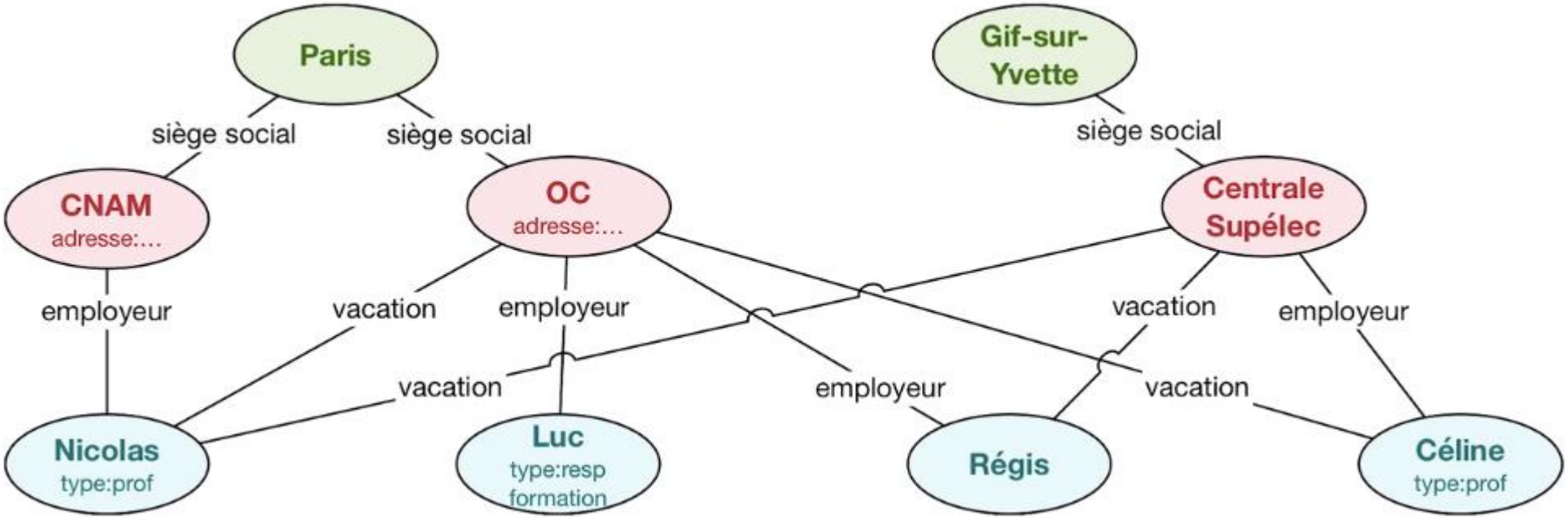
- Moteurs de recommandation
  - Toute application d'information avec les utilisateurs peut utiliser cette
    - Les nœuds «clients» sont liés aux nœuds «hôtels réservés»
    - Recommander des hôtels similaires aux clients en fonction de l'historique des achats précédents
- Pouvez-vous ajouter plus à la liste ci-dessus?

NoSQL: orienté graphes

Modèle SQL

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	

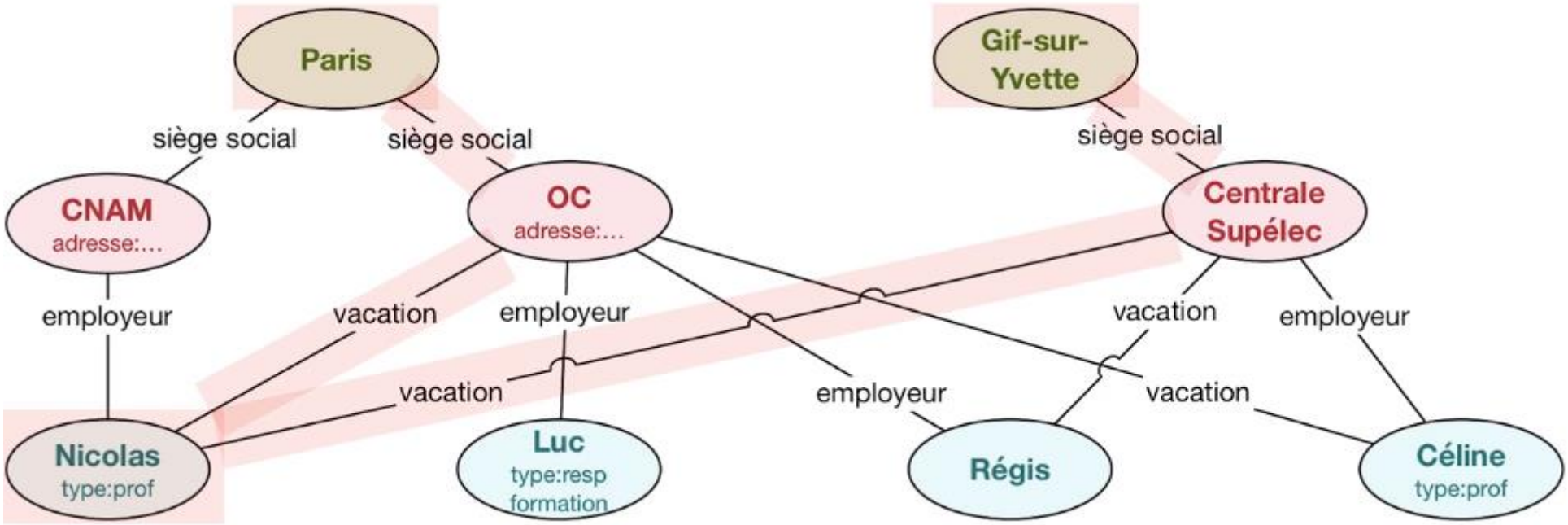
Modèle NoSQL: graphes



NoSQL: orienté graphes

Les requêtes sont des requêtes de type graphe :

- Personne fessant des vacations à Paris et Gif-sur-Yvette



Solutions basées sur le modèle graphes

Azure Cosmos DB:

Exploitation des réseaux

Recommandations

Université Internationale de Casablanca

Ressources

- Documentation officielle :
  - https://www.ibm.com/support/knowledgecenter/SSPT3X_3.0.0/com.ibm.swg.im.infosphere.biginsights.product.doc/doc/c0057605.html
  - https://insidebigdata.com/category/whitepapers/
  - https://spark.apache.org/
  - https://hadoop.apache.org/
  - https://hive.apache.org/

- Livre :
  - "*Les bases de données NoSQL et le Big Data: Comprendre et mettre en oeuvre*" par Rudi Bruchez.
  - *"Big Data white paper" par* Arzu Barske