Ministère de l'enseignement supérieur et de recherches scientifiques
Institut Supérieur des Etudes Technologiques de Radès
Département Technologies de l' information

Modélisation des Bases NoSQL
Modélisation dans MongoDB

Modélisation relationnelle

Schéma Entité/Association

→Transformation

Schéma relationnel

→Normalisation (si nécessaire)

→Schéma relationnel normalisé

→→ création de la base

Modélisation relationnelle

La table des films :

```
create table Film (idFilm integer not null,
                   titre varchar (50) not null,
                   annee integer not null,
                   idRealisateur integer not null,
                   genre varchar (20) not null,
                   resume varchar(255),
                   codePays varchar (4),
                   primary key (idFilm),
                   foreign key (idRealisateur) references Artiste,
                   foreign key (idGenre) references Genre );
```

Modélisation relationnelle

Pour garantir l'intégrité des donnes dans le relationnel il faut normaliser les relations.

Pourquoi normaliser ?

- pour limiter les redondances de données.
- pour limiter les pertes de données.
- pour limiter les incohérences au sein des données.
- pour améliorer les performances des traitements.

Modélisation relationnelle

Règles de normalisation(Rappel):

1ére FN : Une relation est en première forme normale si tous ses attributs sont atomiques (mono_valués) : (Contiennent des valeurs -Non décomposables-constantes dans le temps)

2éme FN : Une relation est en deuxième forme normale si et seulement si : Elle est en 1 FN,

Tout attribut non clé est totalement dépendant de toute la clé.

3ème FN : Une relation est en troisième forme normale si et seulement si : Elle est en 2 FN, tout attribut n'appartenant pas a une clé ne dépend pas d'un attribut non clé

Boyce-Codd FN : Une relation est en forme normale de Boyce-Codd si : Elle est en 3, Aucun attribut faisant partie de la clé ne dépend d'un attribut ne faisant pas partie de la clé primaire.

Et dans les bases NoSQL ?

- Comment les données sont représentées dans les bases documentaires?
- Les données sont représentées sous forme de documents.
- C’est quoi un document ?

Documents Structurés

Un *document structuré ou semi structuré* est un document qui possède une structure . Cette structure peut aller du très simple au très compliqué, ce qui permet de représenter de manière autonome des informations arbitrairement complexes.

- Les documents structurés sont représentés sous deux formats différents : XML et JSON.
- Dans le contexte du NoSQL, on parle de *documents* et de *collections* (de documents).

Documents Structurés

- Une base relationnelle peut être transformée sous forme de documents structurés, et chaque document pourrait être plus complexe structurellement qu'une ligne dans une table relationnelle.

- Cette représentation, pour des données *régulières*, n'est pas recommandée à cause de la redondance que peut générer cette description.

Documents Structurés

- Les documents structurés ne sont pas soumis aux contraintes de normalisation.
- Un attribut peut avoir plusieurs valeurs (en utilisant la structure de tableau en JSON)
- En relationnel, il faudrait ajouter trois tables qui représente la relation entre film , genre et film réalisateur .

Documents Structurés

Document Film

```
{
    "title": "Star Wars épisode VIII",
    "year": 2015,
    "genre": ["Action", "Science Fiction"],
    "country": "USA",
    "realisateur": [{
        "id": 168,
        "nom": "Grier",
        "prenom": "Pam"
    }]

}
```

Documents Structurés

Constat

- Dans une base relationnelle, les données sont liées, très contraintes, régulières et bien structurées.

Alors

- Une représentation arborescente XML / JSON est plus appropriée pour des données de structures complexes et / ou flexibles.

Documents Structurés et imbrication

- Grâce à l'imbrication des structures, il est possible avec un document structuré de représenter pour tous les films, leurs genres, ainsi que leurs réalisateurs dans le même document.
- *Grâce à cette représentation , un film est décrit dans une seule unité d'infomation(document)*

Avantages de la représentation imbriquée

- **Moins de jointure**: il est inutile de faire des jointures pour reconstituer l'information puisque toutes le informations sont rassemblées dans la même table.
- **Plus besoin de transaction :** les données sont créées et insérées en même temps, plus besoin de créer l'objet au préalable
- **Adaptation à la distribution:** Si les documents sont autonomes, il est très facile des les déplacer pour les répartir au mieux dans un système distribué; l'absence de lien avec d'autres documents donne la possibilité d'organiser librement la collection.

Inconvénients de la représentation imbriquée

- **Accès hiérarchique :** dans la représentation imbriquée l'accès à une entité est privilégié à l'accès à une autre entité.
- **Les entités ne sont pas autonomes :** Il y a des entités qui dépendent d'autres entités.
- **Redondance:** la même information peut être représentée plusieurs fois.

Schéma dans les bases NoSQL ?

- Les bases NoSQL (à quelques exceptions près, ex. Cassandra) ne proposent pas de schéma, ou en tout cas rien d'équivalent aux schémas relationnels. Il existe un gain apparent: on peut tout de suite, sans effectuer la moindre démarche de modélisation, commencer à insérer des documents. Mais rapidement la structure de ces documents peut changer, et on ne sait plus trop ce qu'on a mis dans la base de données.
- Si on veut éviter cela, c'est au niveau de l'application effectuant des insertions qu'il faut effectuer la vérification des contraintes qu'un système relationnel peut nativement prendre en charge. Il faut également, pour toute application exploitant les données, effectuer des contrôles puisqu'il n'y a pas de garantie de cohérence ou de complétude.

Comment choisir : NoSQL ou relationnel ?

Quand utiliser (ou pas) une base documentaire ?

- Des données très spécifiques, peu ou faiblement structurées (texte, données multimédia, graphes)
- Peu de mises à jour, beaucoup de lectures. ; la redondance ne pose pas de problème.
- on veut traiter de très gros volumes de manière “scalable”.
- De forts besoins en temps réel.
- Données distribuées

→ bases documentaires

Sites Utiles

- Tout sur JSON : http://json.org/
- Un validateur de documents JSON : http://jsonlint.com/
- Un générateur de schéma JSON : https://www.jsonschema.net/home