ANNEXE :
Conception d’une base de données NoSQL

Issu de https://mediaserver.lecnam.net/permalink/v125f3595297febdcm2m/

Les documents structurés ne sont pas soumis aux contraintes de normalisation.

Un attribut peut avoir **plusieurs valeurs** (en utilisant la structure de tableau)

```
{
  "title": "Pulp fiction",
  "year": "1994",
  "genre": ["Action", "Policier", "Comedie"]
  "country": "USA"
}
```

En relationnel, il faudrait ajouter deux tables.

Il est également facile de représenter des données régulières.

id	nom	prénom
37	Tarantino	Quentin
167	De Niro	Robert
168	Grier	Pam

Facile à représenter sous forme d'un document (très régulier).

```
[
  artiste: {"id": 37, "nom": "Tarantino", "prenom": "Quentin"},
  artiste: {"id": 167, "nom": "De Niro", "prenom": "Robert"},
  artiste: {"id": 168, "nom": "Grier", "prenom": "Pam"}
]
```

Table relationnelle = arbre de hauteur constante. Trois niveaux : ligne, attribut, valeur.

Documents structurés = imbrication des structures

Grâce à l'imbrication des structures, il est possible de représenter dans une même unité un film et son metteur en scène.

```
{
    "title": "Pulp fiction",
    "year": "1994",
    "genre": "Action",
    "country": "USA",
    "director": {
    "last_name": "Tarantino",
    "first_name": "Quentin",
    "birth_date": "1963"
    }
}
```

On cherche à regrouper les informations dans un document autonome

Représentation sous forme de document structuré

On peut coder toutes les informations sur un film.

```
{
    "_id": "movie:17",
    "title": "Pulp Fiction",
    "year": "1994",
    "director":  {
    "last_name": "Tarantino", "first_name": "Quentin",
    "birth_date": "1963"
    },
    "actors": [
       {
         "first_name": "John", "last_name": "Travolta",
         "birth_date": "1954",  "role": "Vindent Vega"
       },
       {
        "first_name": "Bruce", "last_name": "Willis",
        "birth_date": "1955", "role": "Butch Coolidge"
       },
       {
        "first_name": "Quentin", "last_name": "Tarantino",
        "birth_date": "1963", "role": "Jimmy Dimmick"
       }
     ]
```

Les avantages de la représentation par document structuré.

- **Plus besoin de jointure (?)** : il est inutile de faire des jointures pour reconstituer l'information puisqu'elle n'est plus dispersée, comme en relationnel, dans plusieurs tables.
- **Plus besoin de transaction (?)** : une écriture (du document) suffit ; une lecture suffit pour récupérer l'ensemble des informations.
- **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

Les inconvénients du modèle

La représentation par document a deux inconvénients forts.

- **Chemin d'accès privilégié** : les films apparaissent près de la racine des documents, les artistes sont enfouis dans les profondeurs ;
  L'accès aux films est donc privilégié
- **Les entités ne sont plus autonomes** : pas moyen de créer un réalisateur s'il n'y a pas au moins un film.
- **Redondance** : la même information doit être représentée plusieurs fois, ce qui est tout à fait fâcheux (Quentin Tarantino est représenté deux fois).

La redondance mène à des incohérences.

Privilégier un chemin d'accès est bon pour certaines applications, mauvais pour d'autres.

Comment faire pour obtenir la liste des films de Quentin Tarantino avec la représentation précédente ?

Pas de langage de requête ?
Il faut faire un programme pour tout, même la moindre mise à jour !

Pas de schéma ?
On peut mettre n'importe quoi dans la base ; c'est l'application qui doit faire les contrôles et les corrections.

Pas de transaction ?
Ne convient pas pour beaucoup d'applications (commerce électronique).