Installation & utilisation de BDD NoSQL
MongoDB (orientée Document)

MongoDB (bdd NoSQL orientée Document)

- Installation de la base de données MongoDB : en local, ou dans le cloud :
  - Utilisation de `MongoDB Atlas` dans le **cloud** :
    - créez un compte sur le site de MongoDB permettant d'utiliser `MongoDB Atlas`.
    - créez un cluster (FREE), indiquez le provider et la région, créez un utilisateur (system/manager), définissez les adresses IP pouvant accéder à votre bdd (0.0.0.0/0).
    - récupérez le string de connexion (Connect + Driver + Java) indispensable pour se connecter depuis Java.
  - *Utilisation de `MongoDB Community Edition` en local :*
    - *downloadez & installez `MongoDB Community Server`.*
    - *vous pouvez downloader & installer un outil d'administration : `Shell/Compass/…`*
    - *testez l'installation ainsi que l'accès à la bdd à l'aide de l'outil d'administration (`mongosh/Compass`)*
- Downloadez un driver `mongo-java-driver-3.12.14.jar` contenant les librairies MongoDB pour Java.
- Dans un projet IntelliJ, créez un sous-répertoire `lib`, copiez le driver dans `lib`, et définissez-le comme librairie.
- Créez une classe `Bdd` dans un package `dao`, et codez cette méthode de connexion :

```
public static MongoDatabase connectDB() {
  return MongoClients.create("mongodb://localhost").getDatabase("nomDeLaBase"); }
```

  - si vous utilisez `MongoDB Atlas` dans le cloud, remplacez `localhost` par le string de connexion fourni *(par exemple* `"mongodb+srv://system:manager@cluster0.xxxxxxx.mongodb.net"`).
- Vous pouvez dès lors récupérer une collection de documents ainsi :

```
MongoCollection<Document> coll = db.getCollection("nomDeLaCollection");
```

- Puis utilisez `coll.find`, `coll.insertOne/Many`, `coll.deleteMany,…` sur cette collection.
- Pour limiter les messages d'info dans le log : Logger.getLogger("org.mongodb.driver").setLevel(Level.SEVERE);

- Petite analogie « simpliste/basique » entre MongoDB et une bdd relationnelle :
  - une `Table` SQL devient ==> une `Collection` MongoDB
  - un `enregistrement` / une `ligne` ==> un `Document`
  - une `Colonne` ==> un `Champ`, une paire `clé-valeur`
  - une `Jointure` ==> un `Document imbriqué/intégré` *(ou une `référence`).*

TP-10 - MongoDB

- Créez-vous une bdd `MongoDB` en utilisant `MongoDB Atlas` dans le cloud *(ou en local sur votre machine si vous voulez).*
- Downloadez un driver MongoDB pour java *(par ex. depuis https://**jar-download.com**/?search_box=mongo-java-driver)*.
- Développez une petite **application Java** effectuant des lectures/écritures de données dans une bdd MongoDB *(par exemple, stockez et recherchez les clubs, les compétitions et les participants comme dans les TP précédents).*
- Utilisez `coll.find`, `coll.insertOne/Many`, `coll.deleteMany,…`
- Étudiez la gestion des **filtres** dans les instructions `find` : `coll.find(Filters.gte("nom","S"))`
- Ainsi que les façons de **parcourir** une collection de documents : `coll.find().cursor()` et `.into()`
- Créez *(instanciez)* des classes Java plutôt que de travailler uniquement avec du texte ou des `Document`.