❶❶ FFB™ – NoSQL : Programmation Java

A l'approche des jeux olympiques de LA, la Fédération Française de Billes (FFB) a décidé de remodeler son système d'information. Le président de la FFB a donc tout naturellement demandé à ChatGPT quel était le meilleur SGBD. Et celui-ci lui a répondu que "MongoDB est un SGBD NoSQL orienté document, très apprécié des Ninjas de l'IUT de Montpellier, et adapté aux applications nécessitant une gestion flexible et évolutive des données non structurées". N'ayant rien compris à la réponse de ChatGPT, le président de la FFB a décidé de migrer les données de la fédération vers une base de données MongoDB.

Les informations qui sont manipulées par la FFB concernent tous les clubs de la fédération, les joueurs de ces clubs, les tournois organisés par les clubs et les résultats obtenus par les joueurs dans ces tournois. Le modèle entités-associations suivant permet d'avoir une vue conceptuelle de ces données.

Actuellement, ces données sont stockées dans une base de données Cassandra qui possède deux tables : `joueurs_par_id_club` et `resultats_par_id_joueur`. Vous trouverez sur Moodle un script CQL permettant de créer et de remplir ces tables. La structure de ces tables est décrite ci-dessous.

joueurs_par_id_club		
id_club	**varchar**	**K**
nom_club	varchar	S
ville_club	varchar	S
id_joueur	varchar	C ↑
nom_joueur	varchar	
prenom_joueur	varchar	
date_naissance	timestamp	
ville_joueur	varchar	

resultats_par_id_joueur		
id_joueur	**varchar**	**K**
nom_joueur	varchar	S
prenom_joueur	varchar	S
id_tournoi	varchar	C ↑
nom_tournoi	varchar	
classement	int	

Après avoir recensé les requêtes qu'il faudra réaliser sur la future base de données MongoDB, l'informaticien de la FFB (qui n'a pas encore été remplacé par ChatGPT), a décidé que sous MongoDB on pouvait stocker les données dans une seule collection `joueurs`.
Vous pourrez trouver sur Moodle le script de création de la collection `joueurs` avec le schéma associé. Mais cette collection sera vide, et on vous demande donc d'écrire un programme Java qui va migrer les données de la BD Cassandra dans la collection `joueurs` de la BD MongoDB.

1ère partie : Programmer la migration à l'arrache.

Dans un premier temps, on souhaite réaliser une application Java qui permet de migrer les données de la BD Cassandra vers la BD MongoDB sans trop se soucier de la conception de l'application.

On essayera tout de même d'écrire le code le plus clair possible, avec des méthodes pas trop longues (si vos méthodes sont trop longues, vous pourrez éventuellement les raccourcir en introduisant des méthodes privées). Mais, dans cette partie on ne se préoccupera pas d'avoir une conception 'ouvert-fermé' permettant d'ajouter facilement des extensions (comme par exemple, des bases de données d'un autre type que Cassandra ou MongoDB). Dans cette partie il n'est donc pas nécessaire d'utiliser des interfaces de programmation ou bien des Design Patterns que vous avez vu en 2ème année. Vous ferrez éventuellement cela dans la troisième partie du TD où on vous demandera de refactorer votre application avant de rajouter des nouvelles fonctionnalités.

Dans cette partie, au niveau de la conception, la seule chose qui vous est imposée est que la migration doit se dérouler en deux temps :

- d'abord, votre application doit se connecter à Cassandra, récupérer les données qui se trouvent dans les tables pour instancier des objets de classes métier (`Club`, `Joueur`, `Tournoi`, `Resultat`). Il faudra bien sûr se déconnecter de Cassandra à la fin de cette étape.
- ensuite, votre application doit se connecter à la base de données MongoDB et récupérer les données qui se trouvent dans les objets des classes métier pour les mettre dans la collection `joueurs`. Là aussi, il faudra se déconnecter de MongoDB à la fin de cette étape.

Vous trouverez sur gitlab une ébauche de cette application (Migration FFB). Dans ce projet, on vous fournit déjà les méthodes qui permettent de se connecter et se déconnecter à Cassandra et MongoDB. Vous trouverez dans la classe `Main` la logique de la conception qui est décrite ci-dessus.

Travail à faire :

1. Avec votre IDE, se connecter à Cassandra et exécuter le script `FFB.cql` qui se trouve sur le Moodle.
2. Se connecter à MongoDB et exécuter le script `joueurs.json`.
3. Forker le projet Migration FFB que vous trouverez sur gitlab.
4. Dans le projet, implémenter les 4 classes métier qui se trouvent dans le package `metier`. Il faut déterminer la structure qui sera la plus judicieuse afin de faciliter la migration des données des objets des classes métier vers la collection `joueurs` de MongoDB. Pour cela il est conseillé de bien regarder le schéma proposé dans `joueurs.json`.
5. Implémenter le plus proprement possible les méthodes des classes de migration (`MongoDestinationMigration` et `CassandraSourceMigration`).
6. Implémenter la classe `Main` avec la logique décrite précédemment.

2ème partie : Réaliser des requêtes sous MongoDB avec le client Studio 3T ou Robo 3T.

R1 : Le nombre de joueurs (attention, sous Robo 3T l'opération `countDocuments()` n'est pas disponible).

Cette requête doit retourner 14

R2 : L'identifiant des joueurs qui ont participé à 4 tournois.

Cette requête doit retourner J1

R3 : Parmi les joueurs qui ont participé au moins à un tournoi, le nom et prénom des joueurs qui n'ont pas de classement à un tournoi.

Cette requête doit retourner Draté Daisy

R4 : L'identifiant et le nom des joueurs du club 'CrocoBilles' qui ont participé au tournoi 'T4'.

Cette requête doit retourner J8 Draté ; J9 Diccule

R5 : L'identifiant et le nom des joueurs qui ont participé à la fois aux tournois 'Billes en fête' et au tournoi 'Les billes babillent'.

Cette requête doit retourner J1 Ouzy ; J2 Bricot

R6 : L’identifiant des joueurs qui se trouvent dans un club qui n'est ni à Montpellier, ni à Nimes, ni à Béziers.

Cette requête doit retourner J13 ;J14

R7 : L’identifiant des joueurs nés un 15 août ou qui sont dans un club qui se trouve à Nimes.

Cette requête doit retourner J7 ; J8 ; J9 ; J10 ; J11 ; J13

R8 : Le nom du joueur qui est second au classement du tournoi 'T1'.

Cette requête doit retourner Bricot

R9 : L’identifiant des joueurs qui habitent 'Montpellier' ou qui sont dans un club qui se trouve à 'Montpellier' mais pas les deux à la fois.

Cette requête doit retourner J2 ; J4 ; J6 ; J12

R10 : L'identifiant des 3 joueurs les plus vieux.

Cette requête doit retourner J1 ; J2 ; J10 ; J11

R11 : Le nombre de joueurs de chaque club.

Cette requête doit retourner C1–4 ; C3–3 ; C4–3 ; C2–2 ; C5–2 ;

R12 : L'identifiant des joueurs qui ont participé à plus de 2 tournois.

Cette requête doit retourner J1 ; J2 ; J3

R13 : Le nom du tournoi qui a le plus de participants.

Cette requête doit retourner T1 ; T4

R14 : L'identifiant joueurs qui habitent dans la ville où se trouve leur club.

Cette requête doit retourner J1 ; J3 ; J5 ; J7 ; J9 ; J10 ; J13