IFT2935 – Projet noté

Tina Liu Lee (20092684)

Bojan Odobasic (952514)

Jean-Marc Prud’homme (20137035)

Jean-Daniel Toupin (20046724)

Michel Boyer

Université de Montréal

16 avril 2020

**Tâche 1. La modélisation**

**Diagram

Description automatically generated**

**Tâche 2. La transformation**

Coupe\_du\_monde (**edition**, date\_debut, date\_fin)

* Chaque coupe du monde va augmenter de 1 à chaque évènement (édition), alors nous allons utiliser cela comme clé primaire. Quel pays elle accueille cette année-là, la date début et de fin.

Equipe (**nation**, **#edition\_coupe,** #entraineur\_id, placement)

* Une équipe va changer à chaque coupe du monde. Donc, chaque équipe est définie par sa nation et quelle édition de la coupe du monde qu’elle assiste. Chaque équipe fait partie d’une coupe, a un entraineur chef, représente un pays et a un placement final dans le tournoi.
* Puisque la relation est 1..1 pour equipe et participe, nous pouvons mettre la clé de Coupe\_du\_monde comme clé étrangère dans cette relation. Association de dimension 2 avec lien 1:1

Personne (**personne\_id**, nom, prenom, ddn, pays\_natal, sexe)

* Nous avons choisi de faire de l’héritage pour les personnes. Chaque profession a leurs propres attributs importants à eux. Les personnes ont des attributs assez généraux : nom, prénom, date de naissance, pays natal, genre. Nous avons dû donner un ID à personne, car il n'y avait pas vraiment de combinaison d’attributs qui donnait une clé absolument unique.

Joueur (#**personne\_id**, joueur\_depuis)

* Le premier match professionnel dans une League professionnelle.

Entraineur (#**personne\_id***, entraineur\_depuis*)

* La date de leur premier match en tant qu’entraineur

Collaborateur (#**personne\_id**, expertise, collaborateur\_depuis)

* Expertise d’un collaborateur. Ex. Medecin, physio, diététiste, etc.

Arbitre (#**personne\_id***, arbitre\_depuis*)

* Date de début de carrière arbitre

Stade (**nom**, **ville**, pays\_stade, capacite, annee\_construction)

Un nom (ex. Centre Bell) et une ville qui le détermine uniquement. Un pays, une capacité maximale et une année de construction.

*Sanction (***sanction\_id,** *#j*oueur\_id, #arbitre\_id, #nation\_equipe\_1, #nation\_equipe\_2, #date\_match, couleur*)*

* Une sanction est donnée par un arbitre à un joueur dans un match en particulier. La clé primaire est une incrémentation car il n’y a pas d’autre façon unique de définir une sanction car un même joueur peut recevoir plusieurs sanctions dans le même match par le même arbitre de même couleur

Match (**date, *#nation\_equipe\_1***, **#nation\_equipe\_2**, #edition\_coupe, rang, *#nom\_stade*, #ville\_stade, score\_equipe\_1, score\_equipe\_2)

* Pour déterminer un match, nous allons devoir considérer les 2 équipes et la date du match. Un match a une date, un rang dans le tournoi (ex. 1e round, demi final, final, etc), équipe1 plus son score, équipe 2 plus son score. Nous avons pu absorber les associations accueil, joue, organise car elles sont des associations de dimension 2 avec lien 1:1.

Collaborateur\_equipe ***(#collaborateur\_id , #nation\_equipe*, #edition\_coupe**)

* Cette relation représente les collaborateurs qui assistent l’équipe

Joueur\_equipe ( ***#joueur\_id***, **#nation\_equipe**, **#edition\_coupe** , position, numero\_dossard, equipe\_ligue\_professionnelle)

* Cette relation représente quel joueur a joué pour quelle équipe. Avec Equipe et Joueur, nous pouvons accéder à d’autres informations comme sa position au sein de l’équipe, son numéro de joueur assigné au sein de l’équipe (peut être différent de celui qu’il a dans son équipe professionnelle pour éviter des doublons) et son équipe professionnelle lors de cette coupe.

Arbitre\_match (***#arbitre\_id***, ***#date\_match, #nation\_equipe\_1, #nation\_equipe\_2,*** type\_arbitre)

* Représente qui a arbitré quel match. Avec Arbitre et Match, nous pouvons savoir s’il était arbitre principal ou assistant.

Pays\_coupe (***#edition\_coupe***, ***pays***)

* Permet d’avoir une coupe dans plusieurs pays à la fois.

**Tâche 3. La normalisation**

Coupe\_du\_monde (edition, date\_debut, date\_fin)

Df = { (edition -> date\_debut, date\_fin), (date\_debut -> edition, date\_fin), (date\_fin -> edition, date\_debut) }

Il n’y a toujours qu’une seule coupe du monde en même temps. edition, date\_debut et date\_fin sont donc toutes des clés de la relation. Puisque l’édition des coupes est un numéro débutant à 1 et qui incrémenté à chaque nouvelle coupe, c’est un candidat idéal comme clé primaire qui sera plus facile à utiliser que les dates. Nous prenons pour acquis que les dates sont des attributs simples en postgres. La table est déjà en FNBC car tous les déterminants sont des super-clés.

Equipe (nation, edition\_coupe**,** entraineur\_id, placement)

Df = { (nation, edition\_coupe-> entraineur\_id, placement), (entraineur\_id, edition\_coupe-> nation, placement), (placement, edition\_coupe-> nation, entraineur\_id) }

Un pays est représenté par une seule équipe dans une édition de coupe donnée. Son entraîneur est unique, tout comme son placement. On remarque alors que n’importe quel attribut associé avec l’édition de coupe détermine une équipe précise et donc le reste des attributs. Comme clé primaire, nous choisissons nation et edition\_coupe car c’est habituellement de cette manière qu’on identifie les équipes de la coupe du monde. La table est déjà en FNBC car tous les déterminants sont des super-clés.

Personne ( personne\_id, nom, prenom, ddn, pays\_natal, sexe)

Df ={ personne\_id -> nom, prenom, ddn, pays\_natal, sexe)

Comme un SSN aux yeux de la fédération/ notre BDSM une personne a un identificateur unique. Chaque personne a un nom, prénom, adresse, date de naissance, pays natal et un sexe.

Joueur ( personne\_id, joueur\_depuis)

Df = { personne\_id -> joueur\_depuis)

Un joueur hérite de Personne alors dans la relation Joueur, avec son identificateur de personne nous pouvons savoir quand il a commencé à jouer au foot professionnellement.

Entraineur ( personne\_id, entraineur\_depuis)

Df = { personne\_id -> entraineur\_depuis }

Comme joueur sauf que l’attribut est la date de son premier match en tant que coach. De cette façon même un ancien joueur peut être entraineur et être dans la même BDD

Collaborateur ( personne\_id, expertise, collaborateur\_depuis)

Df = { personne\_id -> expertise, collaborateur\_depuis)

Suit la même logique Joueur et entraineur.

Arbitre ( personne\_id, arbitre\_depuis )

Df = { personne\_id -> arbitre\_depuis)

Même logique que les 3 précédents.

Stade ( nom, ville, capacite, pays\_stade, annee\_construction)

Df = { nom, ville -> capacite, pays\_stade, annee\_construction)

Le nom et la ville déterminent uniquement un stade. Car une ville peut avoir 2 stades et il pourrait avoir 2 stades avec le même nom.

Match (date, nation\_equipe\_1, nation\_equipe\_2, rang, edition\_coupe, *nom\_stade, ville\_stade*, score\_equipe\_1, score\_equipe\_2)

Df= { date, nation\_equipe\_1, nation\_equipe\_2 --> rang, edition\_coupe, nom\_stade, ville\_stade, score\_equipe\_1, score\_equipe\_2 }

Il est tout à fait possible que deux équipes s’affrontent à nouveau dans une même coupe du monde dans le même rang. Cependant, deux équipes ne s’affronteront jamais deux fois dans la même journée. Il s’agit de la seule clé.

*Sanction (sanction\_id, j*oueur\_id, arbitre\_id, nation\_equipe\_1, nation\_equipe\_2, date\_match, couleur*)*

Df = {sanction\_id -> joueur\_id, arbitre\_id, nation\_equipe\_1, nation\_equipe\_2, date\_match, couleur}

Nous avons besoin d’un id unique car un même joueur peut recevoir plusieurs sanctions dans le même match par le même arbitre de même couleur. Alors, c’est la seul façon d’identifier les sanctions

*Collaborateur\_equipe (collaborateur id , nation\_equipe, edition\_coupe)*

*Df = { }*

*Aucune DF car il n’y a pas d’attribut particulier à cette association. Cependant nous avons besoin de cette relation pour joindre les collaborateurs au reste de la BDD par jointure.*

*Joueur\_equipe (*joueur\_id, *nation\_equipe, edition\_coupe,* position, numero\_dossard, equipe\_ligue\_professionnelle)

Df = { joueur\_id, nation\_equipe, edition\_coupe -> position, numero\_dossard, equipe\_ligue\_professionnelle }

La raison nous avons besoin des 3 attributs pour définir position, dossard et équipe prof est parce qu’un même joueur peut jouer dans plusieurs équipes et une équipe a plusieurs joueurs. La raison pour laquelle nous avons besoin de joueur\_id + equipe pour définir position, dossard, et équipe professionnelle est que la position, le dossard et l’équipe professionnel actuelle au moment de la coupe peuvent changer entre chaque coupe.

Arbitre\_match (*arbitre\_id*, *date\_match, nation\_equipe\_1, nation\_equipe\_2,* type\_arbitre)

Df = { *arbitre\_id*, *date\_match, nation\_equipe\_1, nation\_equipe\_2 ->* type\_arbitre }

Un peu dans la même optique que Joueur\_equipe. Il y a plusieurs arbitres dans un match et un arbitre peu arbitrer plusieurs matchs. Alors nous avons besoin de définir match et arbitre au complet avant de savoir quel type d’arbitre il a été dans ce match. À partir de ça, nous pouvons savoir si l’arbitre était principal ou assistant.

Pays\_coupe (*edition\_coupe*, *pays*)

*Df = { }*

Nous avons fait attention de créer notre modèle entité-association le plus normal possible pour rendre la table le plus maintenable possible. Il n’y a pas de normalisation particulière à faire à partir des DF défini plus haut.

1NF :

Nous pouvons remarquer que nos relations sont toutes 1NF car il n’y a aucuns attributs multivalués.

2NF :

La majorité de nos relations ont une clé simple. Ceux qui ont une clé composite, tous les attributs ont besoin de la clé complète pour les définir. Ce qui rend nos relations 2NF

3NF:

Il n’y a aucune transitivité dans nos relations. Nous avons pris soin de mettre les clés étrangères pour nos permettre d’accéder à l'information pertinente à nos entités.

BCNF:

Tous les relations n’ont qu’une clé, la seule relation qui aurait pu causer trouble était celle de Match, cependant, nous avons enlevé la dépendance fonctionnelle qui causait problématique.

***Schema relationnel final***

Coupe\_du\_monde (**edition**, date\_debut, date\_fin)

Equipe (**nation**, **#edition\_coupe,** #entraineur\_id, placement)

Personne (**personne\_id**, nom, prenom, ddn, pays\_natal, sexe)

Joueur (#**personne\_id,** joueur\_depuis)

Entraineur (#**personne\_id**, entraineur\_depuis)

Collaborateur (#**personne\_id**, expertise, collaborateur\_depuis)

Arbitre (#**personne\_id**, arbitre\_depuis)

Stade (**nom**, **ville**, pays\_stade, capacite, annee\_construction)

Sanction (**sanction\_id,** #joueur\_id, #arbitre\_id, #nation\_equipe\_1, #nation\_equipe\_2, #date\_match, couleur)

Match (**date, *#nation\_equipe\_1*, #nation\_equipe\_2,** rang, #edition\_coupe, *#nom\_stade*, #ville\_stade, score\_equipe\_1, score\_equipe\_2)

Collaborateur\_equipe **(#collaborateur\_id, #nation\_equipe, #edition\_coupe**)

Joueur\_equipe (**#joueur\_id**, **#nation\_coupe**, **#edition\_coupe,** position, numero\_dossard, equipe\_ligue\_professionnelle)

Arbitre\_match (**#arbitre\_id**, **#date\_match, #nation\_equipe\_1, #nation\_equipe\_2,** type\_arbitre)

Pays\_coupe (***#edition\_coupe***, ***pays***)

Nous pouvons remarquer que nous pouvons accéder à tous les informations de la BDD par jointure entre les relations. En partant de la relation Equipe, nous avons accès à n’importe quelle information que nous voulons par jointure.

**Tâche 4. L’implémentation**

**Tâche 5. Questions/réponses**

**Tâches 6 et 7. Interaction avec Java et utilisation d’un ORM**

Pour exécuter l’application, il est nécessaire d’avoir une installation de Java 15 ainsi que de JavaFX 16. À partir du terminal, naviguer vers le dossier où se trouve le ficher JAR et exécuter :

java --module-path {CHEMIN\_VERS\_JAVAFX} --add-modules javafx.controls,javafx.fxml -jar IFT2935-Projet.jar

En remplaçant {CHEMIN\_VERS\_JAVAFX} par la localisation du dossier lib de JavaFX (inclusivement).

Pour compiler l’application, il faut avoir également Maven d’installé pour la gestion des dépendances. À partir du terminal, naviguer vers le dossier où se trouve le fichier *pom.xml* et exécuter :

mvn clean package

Le fichier résultant *IFT2935-Projet-1.0-SNAPSHOT-jar-with-dependencies.jar* peut être exécuté avec les instructions ci-haut.

Maven peut également être utilisé pour compiler et lancer l’application sans passer par un fichier JAR en exécutant plutôt :

mvn javafx:run

À noter qu’il est possible de changer les versions de Java et de JavaFX utilisées en modifiant les lignes 13 et 14 (pour Java) et les lignes 21 et 26 (pour JavaFx) du fichier *pom.xml*.

À l’ouverture de l’application, remplacer les informations de connexion par défaut si nécessaire :