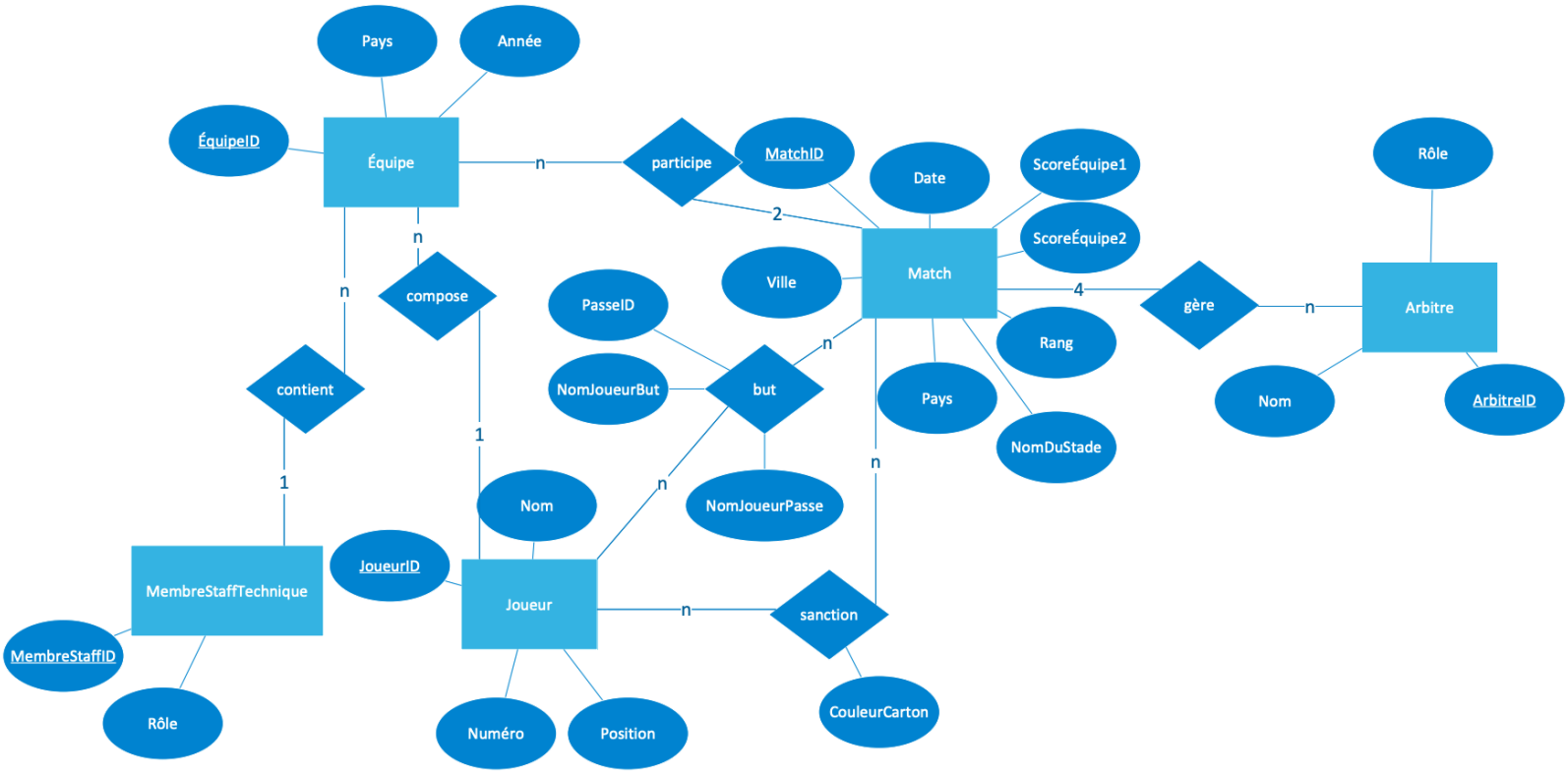
Projet – IFT 2935

- **Martin Medina** *(20235219)*
- **Étienne Mitchell-Bouchard** *(20243430)*
- **Vincent Clouâtre** *(20248263)*
- **Melzi Ibrahim** *(20066033)*

→ **Sujet choisi** : La coupe du monde des nations du football

1. Modélisation



Contraintes :

- Les 4 arbitres sont composés de 1 principal et 3 assistants, donc Rôle peut être “Principal” ou “Assistant”.
- CouleurCarton doit être “Jaune” ou “Rouge”.
- La Position d’un joueur peut être “Attaquant” ou “Defenseur” ou “Gardien” ou “Milieu”
- Le Rang d’un match est un entire qui va de 0 à 4 où 0 correspond à la phase de groupes, 1 aux 8ème de finale, 2 aux quarts de finale, 3 aux demi-finales et 4 à la finale.
- Le Numero d’un joueur est un entier allant de 1 à 99.
- Le Score d’une équipe est un entier positif (>0).

2. Modèle relationnel

Équipe (ÉquipeID, Pays, Année)

Match(MatchID, Date, ScoreÉquipe1, ScoreÉquipe2, Rang, Pays, Ville, NomDuStade)

Arbitre(ArbitreID, Nom, Rôle)

Joueur(JoueurID, Nom, Numéro, Position, #ÉquipeID)

MembreStaffTechnique(MembreStaffID, Rôle, Nom, #ÉquipeID)

Participe(#MatchID, #ÉquipeID)

Gère(#MatchID, #ArbitreID)

But(#MatchID, #JoueurID, #PasseID, NomJoueurBut, NomJoueurPasse)

Sanction(#MatchID, #JoueurID, CouleurCarton)

Règles du passage :

- Lorsqu'il y a une relation entre une entité de cardinalité N et une entité de cardinalité 1, l'entité avec la cardinalité 1 prend comme clé étrangère la clé primaire de l'entité de cardinalité N. **(Relations : Équipe-Joueur et Équipe-Membre_staff _technique)**
- Lorsqu'il y a une relation entre deux entités de cardinalité N, la relation devient une entité et contient comme clés primaires et étrangères les clés primaires des deux entités. Il n'y aucun échange de clés entre les deux entités de départ. **(Relations : Joueur-Match avec les relations But et Sanction)**
- Lorsqu'il y a une relation entre une entité de cardinalité >1 (constante c, c>1) et une entité de cardinalité N, la cardinalité c est équivalente à une cardinalité N, donc il faut traiter la relation comme une relation entre deux entités de cardinalité N. Alors, la relation devient une entité et contient comme clés primaires et étrangères les clés primaires des deux entités. Il n'y aucun échange de clés entre les deux entités de départ. **(Relations : Équipe-Match et Match-Arbitre)**

3. Normalisation

Dépendances pour chaque table :

- Équipe = {ÉquipeID -> Pays ; ÉquipeID -> Année}

  Avec l'identificateur unique de l'équipe, on peut en déduire son pays et son année.

- Match = {MatchID -> Date ; MatchID -> ScoreÉquipe1 ; MatchID -> ScoreÉquipe2 ; MatchID -> Rang ; MatchID -> Pays ; MatchID -> Ville ; MatchID -> NomDuStade ; Ville -> Pays ; NomDuStade -> Ville ; NomDuStade -> Pays}

  Avec l'identificateur unique du match, on peut en déduire tous ses attributs. De plus, sachant qu'un stade est situé dans pays et dans une ville, on peut les déduire. De même, sachant qu'une ville est situé dans un pays, on peut le déduire.

- Arbitre = {ArbitreID -> Nom ; ArbitreID -> Rôle}

  Avec l'indentificateur unique de l'arbitre, on peut en déduire son nom et son rôle.

- Joueur = {JoueurID -> Nom ; JoueurID -> Numéro ; JoueurID -> Position ; Nom,#ÉquipeID -> Numéro}

Avec l'identificateur unique du joueur, on peut en déduire tous ses attributs. De plus, avec son nom et son équipe, on peut déduire son numéro.

- MembreStaffTechnique = {MembreStaffID -> Rôle ; MembreStaffID -> Nom}

  Avec l'identificateur unique du membre du staff, on peut en déduire son rôle et son nom.

- Participe = {}
- Gère = {}
- But = {#MarqueurID -> NomJoueurBut, #PasseID -> NomJoueurPasse}

  On peut déduire les noms des joueurs qui ont marqués et fait la passe décisive avec leurs identificateurs uniques.

- Sanction = {}

Tables normalisées :

- Équipe = Déjà BCNF
- Match = À normaliser
- Arbitre = Déjà BCNF
- Joueur = À normaliser
- MembreStaffID = Déjà BCNF
- Participe = Déjà BCNF
- Gère = Déjà BCNF
- But = À normaliser
- Sanction = Déjà BCNF

- Table Match : Nous avons les dépendances fonctionnelles Ville -> Pays, NomDuStade -> Ville, NomDuStade -> Pays.

  Pour résoudre ce problème, nous pouvons créer une nouvelle table Stade(NomDuStade, Ville, Pays) et modifier la table Match pour supprimer les attributs Ville et Pays. La table Match devient Match(MatchID, Date, ScoreÉquipe1, ScoreÉquipe2, Rang, NomDuStade).

- Table Joueur : Nous avons la dépendance fonctionnelle Nom,#ÉquipeID -> Numéro.

Pour résoudre ce problème, nous pouvons créer une nouvelle table NuméroJoueur(Nom, #ÉquipeID, Numéro) et modifier la table Joueur pour supprimer l'attribut Numéro. La table Joueur devient Joueur(JoueurID, Nom, Position, #ÉquipeID).

- Table But : Nous avons les dépendances fonctionnelles #MarqueurID -> NomJoueurBut, #PasseID -> NomJoueurPasse.

  Pour résoudre ce problème, nous pouvons créer deux nouvelles tables Marqueur(#MarqueurID, NomJoueurBut) et Passeur(#PasseID, NomJoueurPasse), et modifier la table But pour supprimer les attributs NomJoueurBut et NomJoueurPasse. La table But devient But(#MatchID, #JoueurID, #PasseID).

Vérifier si la décomposition à préserver les informations :

- Table Match : Nous avons décomposé cette table en Match(MatchID, Date, ScoreÉquipe1, ScoreÉquipe2, Rang, NomDuStade) et Stade(NomDuStade, Ville, Pays). La jointure naturelle de ces deux tables sur NomDuStade donne la table Match originale. Donc, la décomposition de Match a préservé les informations.

- Table Joueur : Nous avons décomposé cette table en Joueur(JoueurID, Nom, Position, #ÉquipeID) et NuméroJoueur(Nom, #ÉquipeID, Numéro). La jointure naturelle de ces deux tables sur Nom et #ÉquipeID donne la table Joueur originale. Donc, la décomposition de Joueur a préservé les informations.

- Table But : Nous avons décomposé cette table en But(#MatchID, #JoueurID, #PasseID), Marqueur(#MarqueurID, NomJoueurBut) et Passeur(#PasseID, NomJoueurPasse). La jointure naturelle de ces trois tables sur #MarqueurID et #PasseID donne la table But originale. Donc, la décomposition de But a préservé les informations.

Schéma final de la base de données :

Équipe (ÉquipeID, Pays, Année)

Match (MatchID, Date, ScoreÉquipe1, ScoreÉquipe2, Rang, #NomDuStade)

Arbitre (ArbitreID, Nom, Rôle)

Joueur (JoueurID, Nom, Position, #ÉquipeID, #NuméroJoueurID)

MembreStaffTechnique (MembreStaffID, Rôle, Nom, #ÉquipeID)

Participe (ParticipeID, #MatchID, #ÉquipeID)

Gère (GèreID, #MatchID, #ArbitreID)

But (ButID, #MatchID, #MarqueurID, #PasseID)

Sanction (SanctionID, #MatchID, #JoueurID, CouleurCarton)

Stade (NomDuStade, Ville, Pays)

NuméroJoueur (NuméroJoueurID, Nom, #ÉquipeID, Numéro)

Marqueur (MarqueurID, NomJoueurBut)

Passeur (PasseID, NomJoueurPasse)

4. Implémentation

Regarder le fichier « Script.sql »

5. Question/Réponse

Requêtes :

- **Requête 1** : Classe les arbitres par le nombre de cartons données. Donne toutes les informations de l'arbitre ainsi que son nombre total de carton donnée

- **Requête 2** : Montre les 3 meilleur joueur qui ont marqué des buts avec leur information et l'information de leur équipe. Donne le nom, le numéro et la position du joueur ainsi que le pays et l'année du pays et le nombre total de but marqué

- **Requête 3** : Montre les équipes en ordre alphabétique de pays avec leur nombre de but et de passe. Donne les infos de l'équipe ainsi que le total des buts et le total des passes.

- **Requête 4** : Affiche les informations techniques de chaque match. Donne la date, le score de l'Équipe 1, le score de l'Équipe 2, le rang, le nom (nom arbitre), le nom du stade, le pays et la ville.

Algèbre relationnelle :

A = Fonction d'agrégation

- **Requête 1** :
  J1 <- Arbitre $\bowtie_{ArbitreID=ArbitreID}$ Gère
  J2 <- J1 $\bowtie_{MatchID=MatchID}$ Sanction
  Result <- $\rho_{(ArbitreID,\ total\_carton,\ Nom,\ Rôle)}$(Arbitre.ArbitreID $A_{Count(Sanction.SantionID)}$, Arbitre.Nom, Arbitre.Rôle (J2))

- **Requête 2** :
  J1 <- Joueur $\bowtie_{ÉquipeID=ÉquipeID}$ Équipe
  J2 <- J1$\bowtie_{JoueurID=JoueurID}$ But
  Result <- $\rho_{(Nom,\ Numéro,\ Position,\ Pays,\ Année,\ total\_but)}$(Joueur.Nom, Joueur.Numéro, Joueur.Position, Équipe.Pays, Équipe.Année, (Joueur.JoueurID, But.JoueurID) $A_{Count(But.ButID)}$ (J2))

- **Requête 3** :
  J1 <- Équipe $\bowtie_{ÉquipeID=ÉquipeID}$ Joueur
  J2 <- J1$\bowtie_{JoueurID=JoueurID}$ But
  Result <- $\rho_{(Pays,\ Année,\ ÉquipeID,\ total\_but,\ total\_passe)}$(Équipe.Pays, Équipe.Année, (Équipe.ÉquipeID) $A_{Count(But.ButID)}$, (Équipe.ÉquipeID) $A_{Count(But.PasseID)}$ (J2))

- **Requête 4** :
  J1 <- Match $\bowtie_{MatchID=MatchID}$ Gère
  J2 <- J1 $\bowtie_{ArbitreID=ArbitreID\ AND\ Rôle="Principal"}$ Arbitre

J3 <- J2 $\bowtie_{Stade=NomDuStade}$ Stade
Result <- $\rho_{(Date, ScoreÉquipe1, ScoreÉquipe2, Rang, Nom, NomDuStade, Pays, Ville)}$(Match.Date, Match.ScoreÉquipe1, Match.ScoreÉquipe2, Match.Rang, Arbitre.Nom, Stade.NomDuStade, Stade.Pays, Stade.Ville (J3))

SQL :

Regarder le fichier « Script.sql »

6. Java + Bonus

- **Mini-guide utilisateur : https://youtu.be/q7JRWJUTBgc**

  Avant d'exécuter le JAR :

  Le JAR fonctionne sans avoir setup une base de données. Cependant, s'il y a une erreur de connexion de la part de PostgreSQL, un message d'erreur s'affichera dans la zone de texte lorsque vous cliquez sur un des boutons. Pour que tout fonctionne, il faut avoir créer une DB avec pgAdmin4. Exécutez donc le code SQL de la création des tables et de l'insertion des données (fichier « *Script.sql* ») pour créer votre DB.

  Pour se connecter il faut entrer ses informations dans le fichier « *SQLCredentials.txt* ». Il faut tout d'abord que le nom de la DB que vous avez créer dans pgAdmin4 corresponde à celui dans le fichier. Ensuite, il faut entrer son **user** et son **password** pour pgAdmin4 dans les endroits indiqués dans le fichier. Sauvegardez les modifications et passez à l'étape suivante (Exécution du JAR). L'application .jar ne crée aucune table et n'insère aucune donnée, elle ne fait qu'effectuer les requêtes et afficher leur résultat.

  Exécution du JAR :

  Pour exécuter le fichier .jar, ouvrez la console et déplacez-vous avec la commande *cd* jusqu'au dossier où se trouve le fichier. Ensuite, écrivez la commande :

  « *java -jar ProjetIFT2935.jar* »

  L'interface s'ouvrira et vous verrez 4 boutons qui correspondent à l'exécution des 4 requêtes. Lorsque vous cliquez sur un des boutons, le résultat de la requête choisie

s’affichera dans la zone de texte placée à droite des boutons. Pour exécuter une autre commande, vous devez seulement cliquer sur le bouton de la requête. Le programme effacera le résultat de la requête précédente et affichera celui de la nouvelle.

- **Bonus :** Veuillez noter que nous avons utilisé JPA (Jakarta Persistence) à la place de JDBC, donc nous avons effectué le bonus (utilisation d’un ORM).

Bon été ☺