PROJET Système d'information et base de données 2
Base de données d'un tournois sportif

Groupe 8

22010415 - Lisa SAVY
21809267 - Éric GILLES
22009176 - Morgan NAVEL
22010416 - Romain GALLERNE

16 décembre 2022

Table des matières

1 Description du Sujet

1.1 Mise en contexte

Nous proposons de modéliser des compétitions sportives, nous représenterons donc les compétitions avec leurs matchs ainsi que les équipes qui y participent et les différents acteurs de la compétitions (joueurs, arbitres, sponsors, clubs) modulo leur présence.

1.2 Spécifications

Personne
Une Personne est représentée par un numéro, un nom, un prenom, une nationalité et une date de naissance.
Les Joueurs, les Arbitres, les Coachs sont des Personnes.

Arbitre
Un arbitre peut arbitrer plusieurs matchs (il faudra faire attention à ce qu'il n'arbitre pas 2 matchs qui se déroulent en même temps).

Coach
Un Coach est affecté à au plus une Equipe.

Joueur
Chaque Joueur possède un classement en fonction des résultats de ses matchs.
Un Joueur joue dans un unique Club, et plusieurs joueurs peuvent jouer dans ce Club.
Un Joueur peut jouer dans plusieurs Equipes (exigence : le joueur ne pourra être affecté qu'à une seule équipe d'une compétition).
Un Joueur ne peut être soutenu que par au plus un Sponsor.

Equipe
Une Equipe est identifiée par un numéro d'équipe et possède un nom.
Une Equipe ne peut être sponsorisée que par au plus un Sponsor.

Club
Un Club est identifié par un numéro de Club et est caractérisé par un nom, par une date de création, une adresse, un nombre de victoires en compétition.

Sponsor
Un Sponsor est identifié par un numéro de Sponsor, un nom et une date de création.
Un Sponsor sponsorise une ou plusieurs Equipes.
Un Sponsor soutient un ou plusieurs Joueurs.

Match
Un Match est composée d'un identifiant d'une date et des scores des deux Equipes.
Deux Equipes participent à un Match.
Un Match est arbitré par un Arbitre.

Compétition
Une Compétition est composée d'un identifiant, d'un nom et de la saison où elle se déroule.
Une Comptétition est composée de plusieurs Matchs.

2 Dictionnaire des Données

Nom Symbolique	Description	Type de données	Longueur	Nature
NUM_PERSONNE	numéro d'identification d'une personne	Entier	6	clé primaire
NOM	nom d'une personne	Chaîne de caractères	15	non nul
PRENOM	prénom d'une personne	Chaîne de caractères	15	non nul
NATIONALITE	nationalité d'une personne	Chaîne de caractères	15	non nul
DATE_NAISSANCE	numéro d'identification d'une personne	Date		facultatif
NUM_JOUEUR	numéro d'identification d'un joueur	Entier	6	clé primaire et clé etrangère
CLASSEMENT	classement du joueur	Entier		unique
NUM_ARBITRE	numéro d'identification d'un arbitre	Entier	6	clé primaire et clé etrangère
NUM_COACH	numéro d'identification d'un coach	Entier	6	clé primaire et clé etrangère
NUM_CLUB	numéro d'identification d'un club	Entier	6	clé primaire
NOM_CLUB	nom d'un club	Chaîne de caractères	15	non nul
DATE_DE_CREATION	date de création du club	Date		facultatif
ADRESSE	adresse du club	Chaîne de caractères	25	facultatif
NOMBRE_DE_VICTOIRE	nombre de victoire du club	Entier		facultatif
PAYS	pays du club	Chaîne de caractères	20	non nul
NUM_EQUIPE	numéro d'identification d'une équipe	Entier	6	clé primaire
NOM_EQUIPE	nom d'une équipe	Chaîne de caractères	15	non nul
NUM_SPONSOR	numéro d'identification d'un sponsor	Entier	6	clé primaire
Sponsor.NOM	nom d'un sponsor	Chaîne de caractères	15	non nul
DATE_CREATION	date de création du sponsor	Date		facultatif
NUM_COMPETITION	numéro d'identification d'une compétition	Entier	6	clé primaire
NOM_COMPETITION	nom de la compétition	Chaîne de caractères	15	non nul
SAISON	saison pendant laquelle se déroule la compétition	Chaîne de caractères	4	non nul
NUM_MATCH	numéro d'identification d'un match	Entier	6	clé primaire
SCORE_E1	score de l'équipe 1	Entier	1	non nul
SCORE_E2	score de l'équipe 2	Entier	1	non nul
DATE_MATCH	date du match	Date		facultatif

FIGURE 1 – Dictionnaire des Données

3 Schéma Entité-Association

On choisit dans ce schéma de représenter Arbitre, Coach et Joueur comme des personnes. On précise cependant que ces catégories sont disjointes (par exemple, un coach ne peut pas être un joueur). Lorsque la lecture des associations est ambiguë, le sens de lecture a été indiqué à côté de l'association. Concernant les cardianalités, on remarque entre autre qu'un match n'est arbitré que par un seul et unique arbitre, à l'inverse un arbitre arbitre plusieurs matchs. Une compétition est composé de plusieurs matchs, mais un match n'appartient qu'à une seule compétition.
Nous avons volontairement choisi de faire apparaitre le plus de cas différents pour les cardinalités (0 :1-1 :1, 0 :1-1 :N, 0 :N-1 :N)

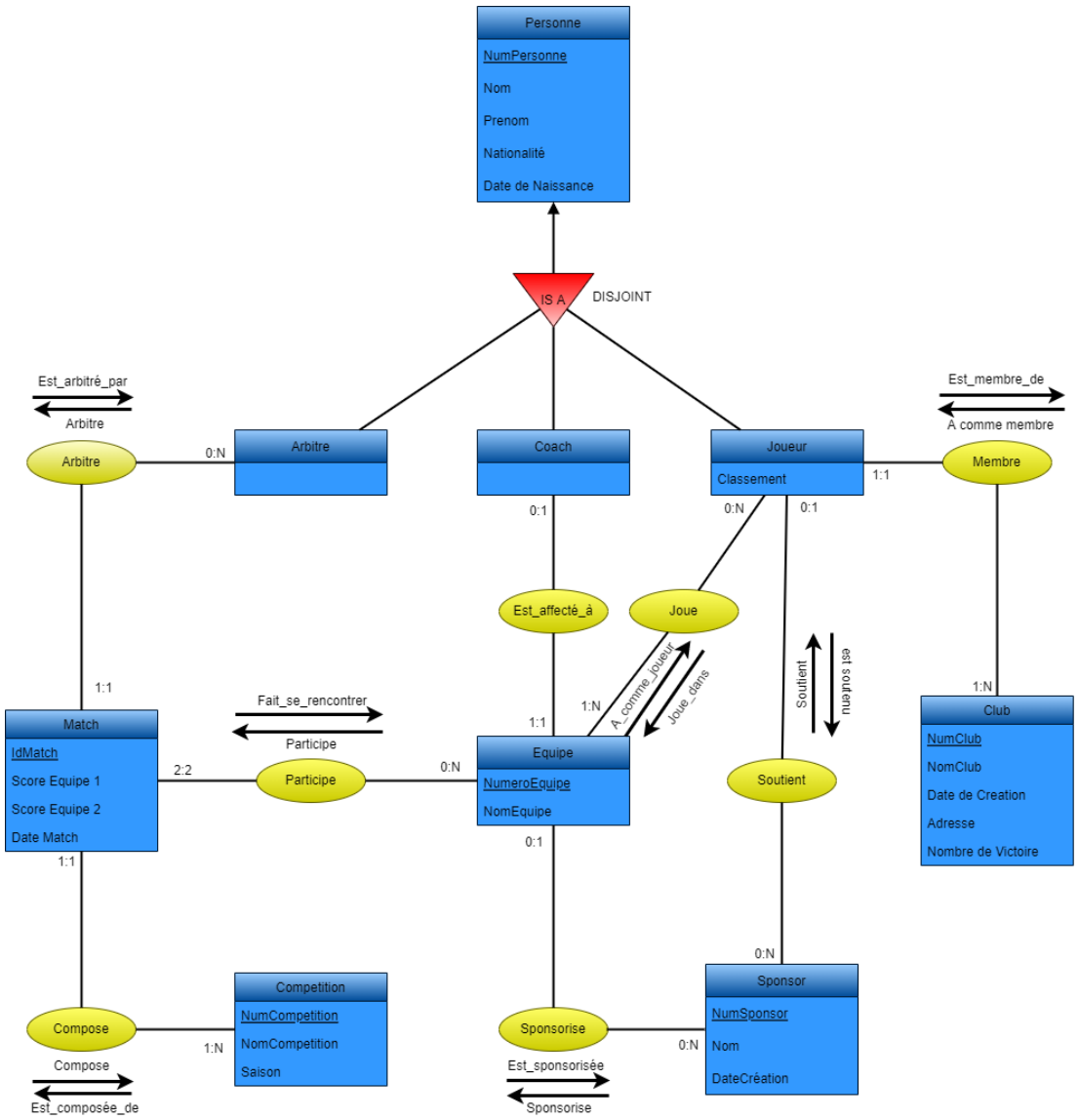

FIGURE 2 – Modèle Entité/Association du projet

4 Schéma Relationnel

Par convention les clés primaires sont soulignées et les clés étrangères sont indiquées en italique et sont précédées d'un dièse (#).

PERSONNE (NUM_PERSONNE,NOM,PRENOM,NATIONALITE,DATE_NAISSANCE)
ARBITRE (*#NUM_ARBITRE*) Ref Personne.NUM_PERSONNE
COACH (*#NUM_COACH*) Ref Personne.NUM_PERSONNE
JOUEUR (*#NUM_JOUEUR*,CLASSEMENT,*#NUM_CLUB,#NUM_SPONSOR*) Ref Personne.NUM_PERSONNE
CLUB (NUM_CLUB,NOM_CLUB,DATE_CREATION,ADRESSE,NOMBRE_VICTOIRE)
SPONSOR (NUM_SPONSOR,NOM_SPONSOR,DATE_DE_CREATION)
EQUIPE (NUM_EQUIPE,NOM_EQUIPE,*#NUM_COACH,#NUM_SPONSOR*)
JOUE (*#NUM_EQUIPE,#NUM_JOUEUR*)
COMPETITION (NUM_COMPETITION,NOM_COMPETITION,SAISON)
MATCH (NUM_MATCH,SCORE_E1,SCORE_E2,DATE_MATCH,*#NUM_E1,#NUM_E2,#NUM_ARBITRE, #NUM_COMPETITION*)

5 Schéma Physique

```
CREATE TABLE PERSONNE(
NUM_PERSONNE NUMERIC(6,0),
NOM VARCHAR(20) CONSTRAINT C_NOM NOT NULL,
PRENOM VARCHAR(20) CONSTRAINT C_PRENOM NOT NULL,
NATIONALITE VARCHAR(20) CONSTRAINT C_NATIONALITE NOT NULL,
DATE_NAISSANCE DATE,
CONSTRAINT Pk_NUM_PERSONNE PRIMARY KEY(NUM_PERSONNE)
);

CREATE TABLE CLUB(
NUM_CLUB NUMERIC(6,0),
NOM_CLUB VARCHAR(30) CONSTRAINT CLUB_NOM NOT NULL,
DATE_CREATION DATE,
ADRESSE VARCHAR(80),
NOMBRE_VICTOIRE INT,
PAYS VARCHAR(20) CONSTRAINT PAYS_NOM NOT NULL,
CONSTRAINT Pk_NUM_CLUB PRIMARY KEY(NUM_CLUB)
);

CREATE TABLE SPONSOR(
NUM_SPONSOR NUMERIC(6,0),
NOM VARCHAR(20) CONSTRAINT SPNOM NOT NULL,
DATE_DE_CREATION DATE,
CONSTRAINT Pk_NUM_SPONSOR PRIMARY KEY(NUM_SPONSOR)
);

CREATE TABLE COACH(
NUM_COACH NUMERIC(6,0),
CONSTRAINT Pk_NUM_COACH PRIMARY KEY (NUM_COACH),
CONSTRAINT Fk_NUM_COACH FOREIGN KEY (NUM_COACH) REFERENCES PERSONNE(NUM_PERSONNE)
);

CREATE TABLE ARBITRE(
NUM_ARBITRE NUMERIC(6,0),
CONSTRAINT Pk_NUM_ARBITRE PRIMARY KEY(NUM_ARBITRE),
CONSTRAINT Fk_NUM_ARBITRE FOREIGN KEY(NUM_ARBITRE) REFERENCES PERSONNE(NUM_PERSONNE)
);

CREATE TABLE JOUEUR(
NUM_JOUEUR NUMERIC(6,0),
NUM_SPONSOR NUMERIC(6,0),
CLASSEMENT INT,
NUM_CLUB NUMERIC(6,0) CONSTRAINT JOUEUR_CLUB NOT NULL,
CONSTRAINT Pk_NUM_JOUEUR PRIMARY KEY(NUM_JOUEUR),
CONSTRAINT Fk_NUM_SPONSOR_JOUEUR FOREIGN KEY(NUM_SPONSOR) REFERENCES
SPONSOR(NUM_SPONSOR),
CONSTRAINT Fk_NUM_CLUB FOREIGN KEY (NUM_CLUB) REFERENCES CLUB(NUM_CLUB),
CONSTRAINT Fk_NUM_JOUEUR FOREIGN KEY(NUM_JOUEUR) REFERENCES PERSONNE(NUM_PERSONNE),
CONSTRAINT UNIQUE_CLASSEMENT UNIQUE(CLASSEMENT)
);
```

```
CREATE TABLE EQUIPE(
NUM_EQUIPE NUMERIC(6,0),
NOM_EQUIPE VARCHAR(20),
NUM_COACH NUMERIC(6,0) CONSTRAINT EQUIPE_COACH NOT NULL,
NUM_SPONSOR NUMERIC(6,0),
CONSTRAINT Pk_NUM_EQUIPE PRIMARY KEY(NUM_EQUIPE),
CONSTRAINT Fk_NUM_COACH_EQUIPE FOREIGN KEY(NUM_COACH) REFERENCES COACH(NUM_COACH),
CONSTRAINT Fk_NUM_SPONSOR_EQUIPE FOREIGN KEY(NUM_SPONSOR) REFERENCES
SPONSOR(NUM_SPONSOR)
);

CREATE TABLE JOUE(
NUM_JOUEUR NUMERIC(6,0),
NUM_EQUIPE NUMERIC(6,0) CONSTRAINT JOUE_NUM_EQUIPE NOT NULL,
CONSTRAINT Pk_JOUE PRIMARY KEY(NUM_JOUEUR,NUM_EQUIPE),
CONSTRAINT Fk_NUM_JOUEUR_JOUE FOREIGN KEY(NUM_JOUEUR) REFERENCES JOUEUR(NUM_JOUEUR),
CONSTRAINT Fk_NUM_EQUIPE FOREIGN KEY(NUM_EQUIPE) REFERENCES EQUIPE(NUM_EQUIPE)
);

CREATE TABLE COMPETITION(
NUM_COMPETITION NUMERIC(6,0),
NOM_COMPETITION VARCHAR(20) CONSTRAINT COMPETITION_NOM NOT NULL,
SAISON VARCHAR(12) CONSTRAINT COMPETITION_SAISON NOT NULL,
CONSTRAINT Pk_NUM_COMPETITION PRIMARY KEY(NUM_COMPETITION)
);

CREATE TABLE MATCH(
NUM_MATCH NUMERIC(6,0),
SCORE_E1 NUMERIC(1,0) CONSTRAINT MATCH_SCORE_E1 NOT NULL,
SCORE_E2 NUMERIC(1,0) CONSTRAINT MATCH_SCORE_E2 NOT NULL,
DATE_MATCH DATE CONSTRAINT MATCH_DATE NOT NULL NUM_E1 NUMERIC(6,0) CONSTRAINT
MATCH_NUM_E1 NOT NULL,
NUM_E2 NUMERIC(6,0) CONSTRAINT MATCH_NUM_E2 NOT NULL,
NUM_ARBITRE NUMERIC(6,0) CONSTRAINT MATCH_NUM_ARBITRE NOT NULL,
NUM_COMPETITION NUMERIC(6,0) CONSTRAINT MATCH_NUM_COMPETITION NOT NULL,
CONSTRAINT Pk_NUM_MATCH PRIMARY KEY(NUM_MATCH),
CONSTRAINT Fk_NUM_E1 FOREIGN KEY(NUM_E1) REFERENCES EQUIPE(NUM_EQUIPE),
CONSTRAINT Fk_NUM_J2 FOREIGN KEY(NUM_E2) REFERENCES EQUIPE(NUM_EQUIPE),
CONSTRAINT Fk_NUM_ARBITRE_MATCH FOREIGN KEY(NUM_ARBITRE)
REFERENCES ARBITRE(NUM_ARBITRE),
CONSTRAINT Fk_NUM_COMPETITION FOREIGN KEY(NUM_COMPETITION) REFERENCES COMPETI-
TION(NUM_COMPETITION)
);
```

6 Requêtes

Nous avons décidé de faire 5 requêtes afin de tester notre codes. Ces requêtes comportent des group by, sous-requêtes, sous-requêtes corrélatives et une division, les voici dans l'ordre d'apparition des figures :

1. Le nombre de joueur sponsorisé dans chaque match dans chaque compétition
2. Les arbitres ayant arbitré au moins un match de chaque compétitions
3. Pour chaque compétition, les Equipes ayant gagné au moins 2 matchs
4. Les sponsors qui ne sponsorient que des joueurs français
5. Pour chaque compétition, l'équipe vainqueur

Ci-dessous, le code SQL correspondant à chacune des requêtes.

```sql
--Le nombre de joueur sponsorisé dans chaque match dans chaque compétition
select match.NUM_COMPETITION,match.NUM_MATCH,count(joueur.NUM_SPONSOR) as NbJoueursSponso from match
  join equipe on (match.NUM_E1=equipe.NUM_EQUIPE or match.NUM_E2=equipe.NUM_EQUIPE)
    join joue on (equipe.NUM_EQUIPE=joue.NUM_EQUIPE)
    join joueur on (joue.NUM_JOUEUR=joueur.NUM_JOUEUR)
    group by match.NUM_COMPETITION,match.NUM_MATCH
    order by match.NUM_COMPETITION,match.NUM_MATCH;
```

FIGURE 3 – Le nombre de joueur sponsorisé dans chaque match dans chaque compétition

```sql
--Les arbitres ayant arbitré au moins un match de chaque compétitions
select arbitre.num_arbitre,personne.prenom,personne.nom,personne.nationalite from arbitre
  join personne on (arbitre.num_arbitre=personne.num_personne)
    where not exists(
      select * from competition where not exists(
        select * from match where (
          match.NUM_ARBITRE = arbitre.NUM_ARBITRE
        and match.NUM_COMPETITION = competition.NUM_COMPETITION
      )
      )
    );
```

FIGURE 4 – Les arbitres ayant arbitré au moins un match de chaque compétitions

```sql
--Pour chaque compétition, les Equipes ayant gagné au moins 2 matchs
select match.NUM_COMPETITION,e.NUM_EQUIPE,e.NOM_EQUIPE from equipe e
  join match on (NUM_E1=e.NUM_EQUIPE or NUM_E2=e.NUM_EQUIPE)
  where ((NUM_E1=e.NUM_EQUIPE and SCORE_E1 > SCORE_E2)
    or (NUM_E2=e.NUM_EQUIPE and SCORE_E2 > SCORE_E1))
    group by match.NUM_COMPETITION,e.NUM_EQUIPE,e.NOM_EQUIPE
    having (
      2 <= (
        select count(*) from match where
        (NUM_E1=e.NUM_EQUIPE and SCORE_E1 > SCORE_E2)
        or (NUM_E2=e.NUM_EQUIPE and SCORE_E2 > SCORE_E1))
    );
```

FIGURE 5 – Pour chaque compétition, les Equipes ayant gagné au moins 2 matchs

```sql
--Les sponsors qui ne sponsorient que des joueurs français
select sponsor.NUM_SPONSOR,sponsor.NOM,p.NATIONALITE from sponsor
  join joueur j on sponsor.NUM_SPONSOR=j.NUM_SPONSOR
    join personne p on p.NUM_PERSONNE=j.NUM_JOUEUR
  where (p.NATIONALITE = 'FRANCAIS')
    group by sponsor.NUM_SPONSOR,sponsor.NOM;
```

FIGURE 6 – Les sponsors qui ne sponsorient que des joueurs français

```sql
--Pour chaque compétition, l'équipe vainqueur
select c.NUM_COMPETITION,c.NOM_COMPETITION,c.SAISON,e.NUM_EQUIPE,e.NOM_EQUIPE from competition c
  join match m on m.NUM_COMPETITION=c.NUM_COMPETITION
    join equipe e on (e.NUM_EQUIPE=m.NUM_E1 or e.NUM_EQUIPE=m.NUM_E2)
    where (
      (select count(*) from match where(
        ((NUM_E1=e.NUM_EQUIPE and SCORE_E1 > SCORE_E2) or (NUM_E2=e.NUM_EQUIPE and SCORE_E2 > SCORE_E1))
        and match.NUM_COMPETITION = m.NUM_COMPETITION
      ))
      >=
      (select max(NbVictoire) from(
      select e2.NUM_EQUIPE,count(*) NbVictoire from equipe e2
      join match on (NUM_E1=e2.NUM_EQUIPE or NUM_E2=e2.NUM_EQUIPE)
        where(
            ((NUM_E1=e2.NUM_EQUIPE and SCORE_E1 > SCORE_E2) or (NUM_E2=e2.NUM_EQUIPE and SCORE_E2 > SCORE_E1))
            and match.NUM_COMPETITION = m.NUM_COMPETITION
        )
        group by e2.NUM_EQUIPE)
      )
    )
    group by c.NUM_COMPETITION,c.NOM_COMPETITION,c.SAISON,e.NUM_EQUIPE,e.NOM_EQUIPE;
```

FIGURE 7 – Pour chaque compétition, l'équipe vainqueur

7 Explication des procédures, fonctions et triggers

7.1 Procédure

La procédure de la FIGURE 8 s'appelle equipe_max_victoire et elle déclare deux variables : nb_victoire de type NUMÉRO et num_equipe de type NUMÉRO.
Elle sélectionne le nombre maximum de victoires dans la table EQUIPE et le stocke dans la variable nb_victoire en utilisant la fonction MAX et la fonction nb_victoire_equipe() voir la fonction ci-dessus.
Elle sélectionne dans le tableau EQUIPE le numéro d'équipe (num_equipe) dont le nombre de victoires est égal au nombre maximum de victoires stocké dans la variable nb_victoire et le stocke dans la variable num_equipe.
La procédure envoie ensuite un message à l'utilisateur, qui affiche le numéro de l'équipe et le nombre de victoires de l'équipe ayant le plus de victoires.
Cette procédure stockée peut être appelée à partir d'un programme ou d'une application pour récupérer l'équipe ayant le plus de victoires dans la table EQUIPE et afficher un message avec le numéro de l'équipe et le nombre de victoires.

```sql
CREATE OR REPLACE PROCEDURE equipe_max_victoire
IS
    nb_victoire NUMBER;
    num_equipe NUMBER;
    CURSOR c_equipe IS SELECT NUM_EQUIPE FROM EQUIPE WHERE nb_victoire_equipe(NUM_EQUIPE) = nb_victoire;
BEGIN
    SELECT MAX(nb_victoire_equipe(NUM_EQUIPE)) INTO nb_victoire FROM EQUIPE;
    SELECT NUM_EQUIPE INTO num_equipe FROM EQUIPE WHERE nb_victoire_equipe(NUM_EQUIPE) = nb_victoire;
    DBMS_OUTPUT.PUT_LINE('L''équipe qui a le plus de victoires est l''équipe ' || num_equipe || ' avec ' || nb_victoire || ' victoires.');
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Il y a aucune équipe avec des victoires dans la base de données.');
    WHEN TOO_MANY_ROWS THEN
        FOR i IN c_equipe LOOP
            DBMS_OUTPUT.PUT_LINE('L''équipe ' || i.NUM_EQUIPE || 'avec ' || nb_victoire || ' victoires.');
        END LOOP;
END;
/
```

FIGURE 8 – Procédure

7.2 Fonction

La fonction de la FIGURE 9 permet de retourner le nombre de victoire en fonction du numéro d'équipe passé en paramètre à la fonction. Cette fonction prend un paramètre de type NUMBER et retourne un résultat de type NUMBER. La fonction déclare une variable de type NUMBER, qui sera utilisée pour stocker le résultat final de la requête SQL effectué.

```
CREATE OR REPLACE FUNCTION nb_victoire_equipe(p_num_equipe NUMBER) RETURN NUMBER
IS
    nb_victoire NUMBER;
BEGIN
    SELECT COUNT(*) INTO nb_victoire FROM MATCH WHERE (NUM_E1=p_num_equipe AND SCORE_E1 > SCORE_E2) OR (NUM_E2=p_num_equipe AND SCORE_E2 > SCORE_E1);
    RETURN nb_victoire;
END;
/
```

FIGURE 9 – Fonction du nombre de victoires par équipe

7.3 Triggers

Le trigger de la FIGURE 10 s'exécute avant qu'un nouveau tuple (ou une mise à jour d'un tuple existant) soit inséré dans la table MATCH. Il vérifie si l'arbitre que l'on souhaite affecter à un match a déjà été affecté à un autre match le même jour. Si c'est le cas, il lève une erreur avec un message d'erreur personnalisé.
Le trigger utilise une exception (nommée ARBITRE dans ce cas) pour gérer le cas où l'arbitre est déjà affecté à un autre match le même jour. Cela permet d'intercepter l'erreur et de fournir un message plus explicite à l'utilisateur.

Le trigger de la FIGURE 11 s'exécute avant qu'un nouveau tuple (ou une mise à jour d'un tuple existant) soit inséré dans la table MATCH. Il vérifie si les deux équipes impliquées dans le match sont les mêmes. Si c'est le cas, il lève une erreur avec un message d'erreur personnalisé en utilisant une exception.

Le trigger de la FIGURE 12 s'exécute avant qu'un tuple soit inséré dans la table JOUE. Il permet de vérifié si le joueur que l'on veut ajouter à l'équipe donnée ne joue pas dans les mêmes compétitions mais pour une équipe différentes. Si on en trouve alors dans ce cas, il lève une exception avec une message l'informant de l'erreur.

```
CREATE OR REPLACE TRIGGER arbitre_dispo
BEFORE INSERT OR UPDATE ON MATCH FOR EACH ROW
DECLARE
    ARBITRE EXCEPTION;
    nb_match NUMBER;
BEGIN
    SELECT COUNT(*) INTO nb_match FROM MATCH WHERE NUM_ARBITRE=:NEW.NUM_ARBITRE AND DATE_MATCH=:NEW.DATE_MATCH;
    IF nb_match > 0 THEN
        RAISE ARBITRE;
    END IF;
EXCEPTION
    WHEN ARBITRE THEN RAISE_APPLICATION_ERROR(-20500,'Cet arbitre est déjà assigné à un match ce jour.');
END;
/
```

FIGURE 10 – Trigger : affectation des arbitres

```
CREATE OR REPLACE TRIGGER diff_equipes
BEFORE INSERT OR UPDATE ON MATCH FOR EACH ROW
DECLARE EQUIPES EXCEPTION;
BEGIN
    IF :NEW.NUM_E1 = :NEW.NUM_E2 THEN
        RAISE EQUIPES;
    END IF;
EXCEPTION
    WHEN EQUIPES THEN RAISE_APPLICATION_ERROR(-20501,'Une équipe ne peut pas jouer contre elle même.');
END;
/
```

FIGURE 11 – Trigger : affectation des équipes pour un match

```
CREATE OR REPLACE TRIGGER Joueur_Autre_Equipe
BEFORE INSERT OR UPDATE ON JOUE FOR EACH ROW
DECLARE
    JOUEUR EXCEPTION;
    nb_competitions NUMBER;
BEGIN
  SELECT COUNT(*) INTO nb_competitions
  FROM MATCH M1 WHERE M1.NUM_E1 = :NEW.NUM_EQUIPE OR M1.NUM_E2 = :NEW.NUM_EQUIPE AND
  M1.NUM_COMPETITION IN (
    SELECT DISTINCT M2.NUM_COMPETITION
    FROM JOUE J2 JOIN MATCH M2 ON M2.NUM_E1 = J2.NUM_EQUIPE OR M2.NUM_E2 = J2.NUM_EQUIPE
    WHERE J2.NUM_EQUIPE != :NEW.NUM_EQUIPE AND J2.NUM_JOUEUR = :NEW.NUM_JOUEUR);
  IF nb_competitions > 0 THEN
    RAISE JOUEUR;
  END IF;
EXCEPTION
    WHEN JOUEUR THEN RAISE_APPLICATION_ERROR(-20502,'Ce joueur participe déjà à une même compétition dans une autre équipe.');
END;
/
```

FIGURE 12 – Trigger : Affectation d'un joueur à une équipe

8 Tests Effectués

Après l'execution des triggers précédents, voici les résultats obtenus dans l'ordre où les triggers ont été présentés.

```
INSERT INTO MATCH VALUES (000012,0,3,000008,000005,000003,010101,'30-01-2019')
            *
ERREUR a la ligne 1 :
ORA-20500: Cet arbitre est deja assigne a un match ce jour.
ORA-06512: a "E20200007055.ARBITRE_DISPO", ligne 10
ORA-04088: erreur lors d'execution du declencheur 'E20200007055.ARBITRE_DISPO'
```

FIGURE 13 – Trigger : affectation des arbitres

```
INSERT INTO MATCH VALUES (000011,1,3,000001,000001,000012,010101,'25-09-2040')
            *
ERREUR a la ligne 1 :
ORA-20501: Une equipe ne peut pas jouer contre elle-meme.
ORA-06512: a "E20200007055.DIFF_EQUIPES", ligne 7
ORA-04088: erreur lors d'execution du declencheur 'E20200007055.DIFF_EQUIPES'
```

FIGURE 14 – Trigger : affectation des équipes pour un match

```
INSERT INTO JOUE VALUES (000002,000001)
            *
ERREUR a la ligne 1 :
ORA-20502: Ce joueur participe deja a une meme competition dans une autre equipe.
ORA-06512: a "E20200007055.JOUEUR_AUTRE_EQUIPE", ligne 15
ORA-04088: erreur lors d'execution du declencheur 'E20200007055.JOUEUR_AUTRE_EQUIPE'
```

FIGURE 15 – Trigger : Affectation d'un joueur à une équipe

```
SELECT nb_victoire_equipe(000002) FROM DUAL

NB_VICTOIRE_EQUIPE(000002)
--------------------------
                         3
```

FIGURE 16 – Fonction : Affichage du nombre de victoires en fonction du numéro de l'équipe

```
EXEC equipe_max_victoire
L equipe 2 avec 3 victoires.
L equipe 5 avec 3 victoires.
L equipe 8 avec 3 victoires.
```

FIGURE 17 – Procédure : Affichage des équipes avec le plus grand nombre de victoires