15/11/2024

TP3 Base de données Réparties

Création de la Base de Données Reparties

Mamadou Baïlo BARRY
M1 MIAGE
UNIVERSITE TOULOUSE 3 – PAUL SABATIER

Sommaire

1. **Introduction** 3
2. **Fragmentation de la base centralisée (Scripts)** 3
   - **Création des fragments sur les trois bases :** 3
   - **Vérification de la présence des segments sur les trois bases :** 6
   - **Destruction des tables initiales centralisées :** 6
3. **Interconnexion des bases** 7
   - **Création des DATABASE LINK :** 7
   - **Test des liens sur chaque base :** 8
4. **Reconstruction des C.I. sur fragments** 9
   - **Pour les clés primaires :** 9
5. **Création des objets globaux de la base répartie** 17
   - **Créations des VIEW et SYNONYM :** 17
     - **Sur *ETUPRE* :** 17
     - **Sur *ETUSEC* :** 18
     - **Sur *ETUTER* :** 19
   - **Requêtes d'interrogation (SELECT) :** 20
6. **Conclusion** 21

1. Introduction

L'objectif de ce TP est de nous faire comprendre la construction d'une BD reparties en fragmentant les tables sur différentes Base de données. Ensuite reconstruire ces fragments (avec les contraintes d'intégrités) en vue de rendre l'écriture des requêtes transparente pour l'utilisateur.

2. Fragmentation de la base centralisée (Scripts)

➢ Création des fragments sur les trois bases :

```
------------------------Pays--------------------------------
-- Création de la fragmentation de Pays sur etuter
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etuter-
REPLACE Pays_3-
USING SELECT *-
    FROM Pays;

------------------------Sport--------------------------------
-- Création de la fragmentation Vertical de Sport sur etupre
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
REPLACE Sport_1-
USING SELECT cds, noms-
    FROM Sport;

-- Création de la fragmentation Vertical de Sport sur etusec
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@brehat.univ-tlse3.fr:1521:etusec-
REPLACE Sport_2-
USING SELECT cds, urls-
    FROM Sport;

------------------------Discipline--------------------------
-- Création de la fragmentation Horizontale de Discipline sur etupre
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
REPLACE Discipline_1-
USING SELECT ndd, nomd, typed, cds-
    FROM Discipline-
    WHERE genred = 'M';

-- Création de la fragmentation Horizontale de Discipline sur etusec
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@brehat.univ-tlse3.fr:1521:etusec-
REPLACE Discipline_2-
USING SELECT ndd, nomd, typed, cds-
    FROM Discipline-
    WHERE genred = 'W';
```

```
-- Création de la fragmentation Horizontale de Discipline sur etuter
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etuter-
REPLACE Discipline_3-
USING SELECT ndd, nomd, typed, genred, cds-
    FROM Discipline-
    WHERE genred IN('X', 'O');

-------------------------Athlete---------------------------
-- Création de la fragmentation Horizontale de Athlete sur etupre
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
REPLACE Athlete_1-
USING SELECT nda, ncomplet, taille, poids, daten, villen, paysn, cio-
    FROM Athlete-
    WHERE genre = 'M';

-- Création de la fragmentation Horizontale de Athlete sur etusec
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@brehat.univ-tlse3.fr:1521:etusec-
REPLACE Athlete_2-
USING SELECT nda, ncomplet, taille, poids, daten, villen, paysn, cio-
    FROM Athlete-
    WHERE genre = 'F';

-----------------------Pratiquer-------------------------------
-- Création de la fragmentation Horizontale de Pratiquer sur etupre
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
REPLACE Pratiquer_1-
USING SELECT P.*-
    FROM Pratiquer P-
    JOIN Discipline D ON P.ndd = D.ndd-
    WHERE genred = 'M';

-- Création de la fragmentation Horizontale de Pratiquer sur etusec
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@brehat.univ-tlse3.fr:1521:etusec-
REPLACE Pratiquer_2-
USING SELECT P.*-
    FROM Pratiquer P-
    JOIN Discipline D ON P.ndd = D.ndd-
    WHERE genred = 'W';

-- Création de la fragmentation Horizontale de Pratiquer sur etuter
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etuter-
```

```
REPLACE Pratiquer_3-
USING SELECT P.*-
    FROM Pratiquer P-
    JOIN Discipline D ON P.ndd = D.ndd-
    WHERE genred IN('X', 'O');

------------------------Gagner_I-----------------------------
-- Création de la fragmentation Horizontale de Pratiquer sur etupre
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
REPLACE Gagner_I_1-
USING SELECT G_I.*-
    FROM Gagner_I G_I-
    JOIN Discipline D ON G_I.ndd = D.ndd-
    WHERE genred = 'M' AND typed = 'P';

-- Création de la fragmentation Horizontale de Pratiquer sur etusec
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@brehat.univ-tlse3.fr:1521:etusec-
REPLACE Gagner_I_2-
USING SELECT G_I.*-
    FROM Gagner_I G_I-
    JOIN Discipline D ON G_I.ndd = D.ndd-
    WHERE genred = 'W' AND typed = 'P';

-- Création de la fragmentation Horizontale de Pratiquer sur etuter
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etuter-
REPLACE Gagner_I_3-
USING SELECT G_I.*-
    FROM Gagner_I G_I-
    JOIN Discipline D ON G_I.ndd = D.ndd-
    WHERE genred IN('X', 'O') AND typed = 'P';

------------------------Gagner_E-----------------------------
-- Création de la fragmentation de Gagner_E sur etuter
COPY FROM BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etupre-
    TO BRM2996A/"B@rry1999"@telline.univ-tlse3.fr:1521:etuter-
REPLACE Gagner_E_3-
USING SELECT *-
    FROM Gagner_E;
```

➢ Vérification de la présence des segments sur les trois bases :

- **Pour *etupre***

	TNAME	TABTYPE
1	SPORT_1	TABLE
2	SPORT	TABLE
3	PRATIQUER_1	TABLE
4	PRATIQUER	TABLE
5	PAYS	TABLE
6	GAGNER_I_1	TABLE
7	GAGNER_I	TABLE
8	GAGNER_E	TABLE
9	DISCIPLINE_1	TABLE
10	DISCIPLINE	TABLE
11	ATHLETE_1	TABLE
12	ATHLETE	TABLE

- **Pour *etusec***

	TNAME	TABTYPE
1	ATHLETE_2	TABLE
2	DISCIPLINE_2	TABLE
3	GAGNER_I_2	TABLE
4	PRATIQUER_2	TABLE
5	SPORT_2	TABLE

- **Pour *etuter***

	TNAME	TABTYPE
1	DISCIPLINE_3	TABLE
2	GAGNER_E_3	TABLE
3	GAGNER_I_3	TABLE
4	PAYS_3	TABLE
5	PRATIQUER_3	TABLE

➢ Destruction des tables initiales centralisées :

```
DROP TABLE Pratiquer;
DROP TABLE Gagner_E;
DROP TABLE Gagner_I;
DROP TABLE Athlete;
DROP TABLE Discipline;
DROP TABLE Sport;
DROP TABLE Pays;
```

3. Interconnexion des bases

➢ Création des DATABASE LINK :

```
-- Sur la base ETUPRE
DROP DATABASE LINK dbl_etupre;
DROP DATABASE LINK dbl_etusec;
DROP DATABASE LINK dbl_etuter;

CREATE DATABASE LINK dbl_etupre
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUPRE';

CREATE DATABASE LINK dbl_etusec
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUSEC';

CREATE DATABASE LINK dbl_etuter
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUTER';

-- Vérification
SELECT * FROM TAB@dbl_etupre;
SELECT * FROM TAB@dbl_etusec;
SELECT * FROM TAB@dbl_etuter;

-------------------------------------------------------------
-- Sur la base ETUSEC
DROP DATABASE LINK dbl_etupre;
DROP DATABASE LINK dbl_etusec;
DROP DATABASE LINK dbl_etuter;

CREATE DATABASE LINK dbl_etupre
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUPRE';

CREATE DATABASE LINK dbl_etusec
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUSEC';

CREATE DATABASE LINK dbl_etuter
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUTER';
```

```
-- Vérification
SELECT * FROM TAB@dbl_etupre;
SELECT * FROM TAB@dbl_etusec;
SELECT * FROM TAB@dbl_etuter;

------------------------------------------------------------
-- Sur la base ETUTER
DROP DATABASE LINK dbl_etupre;
DROP DATABASE LINK dbl_etusec;
DROP DATABASE LINK dbl_etuter;

CREATE DATABASE LINK dbl_etupre
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUPRE';

CREATE DATABASE LINK dbl_etusec
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUSEC';

CREATE DATABASE LINK dbl_etuter
    CONNECT TO BRM2996A
    IDENTIFIED BY "B@rry1999"
    USING 'VERS_ETUTER';

-- Vérification
SELECT * FROM TAB@dbl_etupre;
SELECT * FROM TAB@dbl_etusec;
SELECT * FROM TAB@dbl_etuter;
```

➢ **Test des liens sur chaque base :**

- **Pour *etupre***

	TNAME	TABTYPE
1	ATHLETE_1	TABLE
2	DISCIPLINE_1	TABLE
3	GAGNER_I_1	TABLE
4	PRATIQUER_1	TABLE
5	SPORT_1	TABLE

- **Pour *etusec***

	TNAME	TABTYPE
1	SPORT_2	TABLE
2	PRATIQUER_2	TABLE
3	GAGNER_I_2	TABLE
4	DISCIPLINE_2	TABLE
5	ATHLETE_2	TABLE

- **Pour *etuter***

	TNAME	TABTYPE
1	PRATIQUER_3	TABLE
2	PAYS_3	TABLE
3	GAGNER_I_3	TABLE
4	GAGNER_E_3	TABLE
5	DISCIPLINE_3	TABLE

4. Reconstruction des C.I. sur fragments

➢ Pour les clés primaires :

```
------------------------- CLES ETRANGERES ------------------------
-------------- Sur les fragments de Discipline---------------
--------------Discipline_1 et Sport_1
ALTER TABLE Discipline_1
ADD CONSTRAINT fk_discipline_1_sport_1 FOREIGN KEY (cds) REFERENCES Sport_1 (cds);

--------------Discipline_2 et Sport_2
ALTER TABLE Discipline_2
ADD CONSTRAINT fk_discipline_2_sport_2 FOREIGN KEY (cds) REFERENCES Sport_2 (cds);

--------TRIGGER 'iu' Discipline_3 et Sport_1----------
CREATE OR REPLACE TRIGGER t_iu_discipline_3_cds
BEFORE INSERT OR UPDATE OF cds
ON Discipline_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Sport_1@dbl_etupre
    WHERE cds = :NEW.cds;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'cds inexistant dans Sport_1');
    END IF;
END;

--------TRIGGER 'du' Sport_1 et Discipline_3----------
CREATE OR REPLACE TRIGGER t_du_sport_1_cds
BEFORE DELETE OR UPDATE OF cds
ON Sport_1
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Discipline_3@dbl_etuter
```

```
    WHERE cds = :OLD.cds;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'cds existant dans Discipline_3');
    END IF;
END;

--------------Sur les fragments de Athlete---------------
--------TRIGGER 'iu' Athlete_1 et Pays_3----------
CREATE OR REPLACE TRIGGER t_iu_athlete_1_cio
BEFORE INSERT OR UPDATE OF cio
ON Athlete_1
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Pays_3@dbl_etuter
    WHERE cio = :NEW.cio;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'cio inexistant das Pays_3');
    END IF;
END;

--------TRIGGER 'iu' Athlete_2 et Pays_3----------
CREATE OR REPLACE TRIGGER t_iu_athlete_2_cio
BEFORE INSERT OR UPDATE OF cio
ON Athlete_2
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Pays_3@dbl_etuter
    WHERE cio = :NEW.cio;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'cio inexistant das Pays_3');
    END IF;
END;

--------TRIGGER 'du' Pays_3 et Athlete_1----------
CREATE OR REPLACE TRIGGER t_du_pays_3_athelete_1_cio
BEFORE DELETE OR UPDATE OF cio
ON Pays_3
FOR EACH ROW
DECLARE
```

```
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Athlete_1@dbl_etupre
    WHERE cio = :OLD.cio;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'cio existant dans Athlete_1');
    END IF;
END;

--------TRIGGER 'du' Pays_3 et Athlete_2----------
CREATE OR REPLACE TRIGGER t_du_pays_3_athlete_2_cio
BEFORE DELETE OR UPDATE OF cio
ON Pays_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Athlete_2@dbl_etusec
    WHERE cio = :OLD.cio;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'cio existant dans Athlete_2');
    END IF;
END;

--------------Sur les fragments de Pratiquer---------------
--------------Pratiquer_1 et Athlete_1
ALTER TABLE Pratiquer_1
ADD CONSTRAINT fk_pratiquer_1_nda FOREIGN KEY (nda) REFERENCES Athlete_1 (nda);

--------------Pratiquer_2 et Athlete_2
ALTER TABLE Pratiquer_2
ADD CONSTRAINT fk_pratiquer_2_nda FOREIGN KEY (nda) REFERENCES Athlete_2 (nda);

--------------Pratiquer_1 et Discipline_1
ALTER TABLE Pratiquer_1
ADD CONSTRAINT fk_pratiquer_1_ndd FOREIGN KEY (ndd) REFERENCES Discipline_1 (ndd);

--------------Pratiquer_2 et Discipline_2
ALTER TABLE Pratiquer_2
ADD CONSTRAINT fk_pratiquer_2_ndd FOREIGN KEY (ndd) REFERENCES Discipline_2 (ndd);

--------------Pratiquer_3 et Discipline_3
ALTER TABLE Pratiquer_3
ADD CONSTRAINT fk_pratiquer_3_ndd FOREIGN KEY (ndd) REFERENCES Discipline_3 (ndd);
```

```
--------TRIGGER 'iu' Pratiquer_3 et Athlete_1----------
CREATE OR REPLACE TRIGGER t_iu_pratiquer_3_athlete_1_nda
BEFORE INSERT OR UPDATE OF nda
ON Pratiquer_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Athlete_1@dbl_etupre
    WHERE nda = :NEW.nda;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda inexistant dans Athlete_1');
    END IF;
END;

--------TRIGGER 'iu' Pratiquer_3 et Athlete_2----------
CREATE OR REPLACE TRIGGER t_iu_pratiquer_3_athlete_2_nda
BEFORE INSERT OR UPDATE OF nda
ON Pratiquer_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Athlete_2@dbl_etusec
    WHERE nda = :NEW.nda;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda inexistant dans Athlete_2');
    END IF;
END;

--------TRIGGER 'du' Athlete_1 et Pratiquer_3----------
CREATE OR REPLACE TRIGGER t_du_ath_1_prat_3_nda
BEFORE DELETE OR UPDATE OF nda
ON Athlete_1
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Pratiquer_3@dbl_etuter
    WHERE nda = :OLD.nda;

    IF(n > 0) THEN
```

```
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Pratiquer_3');
    END IF;
END;

--------TRIGGER 'du' Athlete_2 et Pratiquer_3----------
CREATE OR REPLACE TRIGGER t_du_ath_2_prat_3_nda
BEFORE DELETE OR UPDATE OF nda
ON Athlete_2
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Pratiquer_3@dbl_etuter
    WHERE nda = :OLD.nda;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Pratiquer_3');
    END IF;
END;

--------------Sur les fragments de Gagner_I---------------
--------------Gagner_I_1 et Discipline_1
ALTER TABLE Gagner_I_1
ADD CONSTRAINT fk_gagner_i_1_ndd FOREIGN KEY (ndd) REFERENCES Discipline_1 (ndd);

--------------Gagner_I_2 et Discipline_2
ALTER TABLE Gagner_I_2
ADD CONSTRAINT fk_gagner_i_2_ndd FOREIGN KEY (ndd) REFERENCES Discipline_2 (ndd);

--------------Gagner_I_3 et Discipline_3
ALTER TABLE Gagner_I_3
ADD CONSTRAINT fk_gagner_i_3_ndd FOREIGN KEY (ndd) REFERENCES Discipline_3 (ndd);

--------------Gagner_I_1 et Athlete_1
ALTER TABLE Gagner_I_1
ADD CONSTRAINT fk_gagner_i_1_nda FOREIGN KEY (nda) REFERENCES Athlete_1 (nda);

--------------Gagner_I_2 et Athlete_2
ALTER TABLE Gagner_I_2
ADD CONSTRAINT fk_gagner_i_2_nda FOREIGN KEY (nda) REFERENCES Athlete_2 (nda);

--------TRIGGER 'iu' Gagner_I_3 et Athlete_1----------
CREATE OR REPLACE TRIGGER t_iu_gagner_i_3_athlete_1_nda
BEFORE INSERT OR UPDATE OF nda
ON Gagner_I_3
FOR EACH ROW
DECLARE
```

```
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Athlete_1@dbl_etupre
    WHERE nda = :NEW.nda;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda inexistant dans Athlete_1');
    END IF;
END;

--------TRIGGER 'iu' Gagner_I_3 et Athlete_2----------
CREATE OR REPLACE TRIGGER t_iu_gagner_i_3_athlete_2_nda
BEFORE INSERT OR UPDATE OF nda
ON Gagner_I_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Athlete_2@dbl_etusec
    WHERE nda = :NEW.nda;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda inexistant dans Athlete_2');
    END IF;
END;

--------TRIGGER 'du' Athlete_1 et Gagner_I_3----------
CREATE OR REPLACE TRIGGER t_du_athelete_1_gagner_i_3_nda
BEFORE DELETE OR UPDATE OF nda
ON Athlete_1
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Gagner_I_3@dbl_etuter
    WHERE nda = :OLD.nda;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Gagner_I_3');
    END IF;
END;

--------TRIGGER 'du' Athlete_2 et Gagner_I_3----------
CREATE OR REPLACE TRIGGER t_du_athelete_2_gagner_i_3_nda
BEFORE DELETE OR UPDATE OF nda
```

```
ON Athlete_2
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Gagner_I_3@dbl_etuter
    WHERE nda = :OLD.nda;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Gagner_I_3');
    END IF;
END;

--------------Sur les fragments de Gagner_E---------------
--------------Gagner_E_3 et Pays_3
ALTER TABLE Gagner_E_3
ADD CONSTRAINT fk_gagner_e_3_cio FOREIGN KEY (cio) REFERENCES Pays_3 (cio);

--------TRIGGER 'iu' Gagner_E_3 et Discipline----------
CREATE OR REPLACE TRIGGER t_iu_gagner_e_3_discipline_ndd
BEFORE INSERT OR UPDATE OF ndd
ON Gagner_E_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Discipline_1@dbl_etupre
    WHERE ndd = :NEW.ndd;

    SELECT COUNT(*) INTO n
    FROM Discipline_2@dbl_etusec
    WHERE ndd = :NEW.ndd;

    SELECT COUNT(*) INTO n
    FROM Discipline_3@dbl_etuter
    WHERE ndd = :NEW.ndd;

    IF(n = 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda inexistant dans Discipline_1');
    END IF;
END;

--------TRIGGER 'du' Discipline_1 et Gagner_E_3----------
CREATE OR REPLACE TRIGGER t_du_discip_1_gagner_e_3_ndd
BEFORE DELETE OR UPDATE OF ndd
ON Discipline_1
```

```
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Gagner_E_3@dbl_etuter
    WHERE ndd = :OLD.ndd;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Gagner_E_3');
    END IF;
END;

--------TRIGGER 'du' Discipline_2 et Gagner_E_3----------
CREATE OR REPLACE TRIGGER t_du_discip_2_gagner_e_3_ndd
BEFORE DELETE OR UPDATE OF ndd
ON Discipline_2
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Gagner_E_3@dbl_etuter
    WHERE ndd = :OLD.ndd;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Gagner_E_3');
    END IF;
END;

--------TRIGGER 'du' Discipline_3 et Gagner_E_3----------
CREATE OR REPLACE TRIGGER t_du_discip_3_gagner_e_3_ndd
BEFORE DELETE OR UPDATE OF ndd
ON Discipline_3
FOR EACH ROW
DECLARE
    n NUMBER;
BEGIN
    SELECT COUNT(*) INTO n
    FROM Gagner_E_3@dbl_etuter
    WHERE ndd = :OLD.ndd;

    IF(n > 0) THEN
        RAISE_APPLICATION_ERROR(-20000, 'nda existant dans Gagner_E_3');
    END IF;
END;
```

5. Création des objets globaux de la base répartie

➢ Créations des VIEW et SYNONYM :

- **Sur *ETUPRE* :**

```
--------Vue pour Sport
CREATE OR REPLACE VIEW Sport AS
    SELECT s1.*, s2.urls
    FROM Sport_1@dbl_etupre s1,
        Sport_2@dbl_etusec s2
    WHERE s1.cds = s2.cds;

--------Vue pour Discipline
CREATE OR REPLACE VIEW Discipline AS
    SELECT ndd, nomd, typed, 'M' AS genred, cds
    FROM Discipline_1@dbl_etupre
    UNION
    SELECT ndd, nomd, typed, 'W' AS genred, cds
    FROM Discipline_2@dbl_etusec
    UNION
    SELECT *
    FROM Discipline_3@dbl_etuter;

--------Vue pour Athlete
CREATE OR REPLACE VIEW Athlete AS
    SELECT nda, ncomplet, 'M' AS genre, taille, poids, daten, villen, paysn, cio
    FROM Athlete_1@dbl_etupre
    UNION
    SELECT nda, ncomplet, 'F' AS genre, taille, poids, daten, villen, paysn, cio
    FROM Athlete_2@dbl_etusec;

--------Vue pour Pratiquer
CREATE OR REPLACE VIEW Pratiquer AS
    SELECT *
    FROM Pratiquer_1@dbl_etupre
    UNION
    SELECT *
    FROM Pratiquer_2@dbl_etusec
    UNION
    SELECT *
    FROM Pratiquer_3@dbl_etuter;

--------Vue pour Gagner_I
CREATE OR REPLACE VIEW Gagner_I AS
    SELECT *
    FROM Gagner_I_1@dbl_etupre
    UNION
    SELECT *
    FROM Gagner_I_2@dbl_etusec
    UNION
```

```
    SELECT *
    FROM Gagner_I_3@dbl_etuter;

--------Vue pour Pays
CREATE SYNONYM Pays
FOR Pays_3@dbl_etuter;

--------Vue pour Gagner_E
CREATE SYNONYM Gagner_E
FOR Gagner_E_3@dbl_etuter;
```

- **Sur *ETUSEC* :**

```
--------Vue pour Sport
CREATE OR REPLACE VIEW Sport AS
    SELECT s1.*, s2.urls
    FROM Sport_1@dbl_etupre s1,
        Sport_2@dbl_etusec s2
    WHERE s1.cds = s2.cds;

--------Vue pour Discipline
CREATE OR REPLACE VIEW Discipline AS
    SELECT ndd, nomd, typed, 'M' AS genred, cds
    FROM Discipline_1@dbl_etupre
    UNION
    SELECT ndd, nomd, typed, 'W' AS genred, cds
    FROM Discipline_2@dbl_etusec
    UNION
    SELECT *
    FROM Discipline_3@dbl_etuter;

--------Vue pour Athlete
CREATE OR REPLACE VIEW Athlete AS
    SELECT nda, ncomplet, 'M' AS genre, taille, poids, daten, villen, paysn, cio
    FROM Athlete_1@dbl_etupre
    UNION
    SELECT nda, ncomplet, 'F' AS genre, taille, poids, daten, villen, paysn, cio
    FROM Athlete_2@dbl_etusec;

--------Vue pour Pratiquer
CREATE OR REPLACE VIEW Pratiquer AS
    SELECT *
    FROM Pratiquer_1@dbl_etupre
    UNION
    SELECT *
    FROM Pratiquer_2@dbl_etusec
    UNION
    SELECT *
    FROM Pratiquer_3@dbl_etuter;
```

```
--------Vue pour Gagner_I
CREATE OR REPLACE VIEW Gagner_I AS
    SELECT *
    FROM Gagner_I_1@dbl_etupre
    UNION
    SELECT *
    FROM Gagner_I_2@dbl_etusec
    UNION
    SELECT *
    FROM Gagner_I_3@dbl_etuter;

--------Vue pour la séquence sur Discipline
CREATE SYNONYM seq_discipline
FOR seq_discipline@dbl_etupre;

--------Vue pour Pays
CREATE SYNONYM Pays
FOR Pays_3@dbl_etuter;

--------Vue pour Gagner_E
CREATE SYNONYM Gagner_E
FOR Gagner_E_3@dbl_etuter;
```

- **Sur *ETUTER* :**

```
--------Vue pour Sport
CREATE OR REPLACE VIEW Sport AS
    SELECT s1.*, s2.urls
    FROM Sport_1@dbl_etupre s1,
        Sport_2@dbl_etusec s2
    WHERE s1.cds = s2.cds;

--------Vue pour Discipline
CREATE OR REPLACE VIEW Discipline AS
    SELECT ndd, nomd, typed, 'M' AS genred, cds
    FROM Discipline_1@dbl_etupre
    UNION
    SELECT ndd, nomd, typed, 'W' AS genred, cds
    FROM Discipline_2@dbl_etusec
    UNION
    SELECT *
    FROM Discipline_3@dbl_etuter;

--------Vue pour Athlete
CREATE OR REPLACE VIEW Athlete AS
    SELECT nda, ncomplet, 'M' AS genre, taille, poids, daten, villen, paysn, cio
    FROM Athlete_1@dbl_etupre
    UNION
    SELECT nda, ncomplet, 'F' AS genre, taille, poids, daten, villen, paysn, cio
```

```
    FROM Athlete_2@dbl_etusec;

--------Vue pour Pratiquer
CREATE OR REPLACE VIEW Pratiquer AS
    SELECT *
    FROM Pratiquer_1@dbl_etupre
    UNION
    SELECT *
    FROM Pratiquer_2@dbl_etusec
    UNION
    SELECT *
    FROM Pratiquer_3@dbl_etuter;

--------Vue pour Gagner_I
CREATE OR REPLACE VIEW Gagner_I AS
    SELECT *
    FROM Gagner_I_1@dbl_etupre
    UNION
    SELECT *
    FROM Gagner_I_2@dbl_etusec
    UNION
    SELECT *
    FROM Gagner_I_3@dbl_etuter;

--------Vue pour la séquence sur Discipline
CREATE SYNONYM seq_discipline
FOR seq_discipline@dbl_etupre;

--------Vue pour Pays
CREATE SYNONYM Pays
FOR Pays_3@dbl_etuter;

--------Vue pour Gagner_E
CREATE SYNONYM Gagner_E
FOR Gagner_E_3@dbl_etuter;
```

- **Requêtes d'interrogation (SELECT) :**

```
SELECT COUNT(*) FROM Pays;
SELECT COUNT(*) FROM Sport;
SELECT COUNT(*) FROM Discipline;
SELECT COUNT(*) FROM Athlete;
SELECT COUNT(*) FROM Gagner_E;
SELECT COUNT(*) FROM Gagner_I;
SELECT COUNT(*) FROM Pratiquer;
```

- Exemple d'exécution de la requête qui compte le nombre de tuples pour Pratiquer :

```
Nombre de Pratiquants
---------------------
                14464
```

6. Conclusion

En conclusion, ce TP nous a permis de mettre en pratique les connaissances théoriques sur les fragmentations vue en cours (horizontale et verticale). Ensuite il nous a permis aussi de savoir comment créer des liens entres les Base de données (DATABASE LINK) pour faciliter l'accès aux objets distant de manière uniforme, et, aussi de reconstruire les contraintes d'intégrités (clés primaire et clés étrangères). Enfin, de reconstituer virtuellement les tables centralisées qui ont été fragmentés à travers des VIEW et des SYNONYM.