ENSIAS

Bases de Données Réparties

Compte Rendu des TPs

Par :
BOUKROUH Insaf
DAOUDI Wissal
GL1

Sous la direction de :
Monsieur M.Nassar

Année Universitaire : 2015/2016

TP1

Soit la base de données relationnelle Comptes composée des relations suivantes :

Client (#No, Nom, Prénom, Adresse, Ville)
Agence (#No, Nom, Adresse, Ville)
Compte (#No, Type_Compte_No, DateOuverture, Decouvert_autorise, Solde, Client_No, Agence_No)
Type_Compte (#No, Nom, Description)
Operation (#No, Type_Operation_No, Compte_No)
Type_Operation (#No, Description)

La création de la base de donnée Comptes avec ses tables, leurs clés primaires et étrangères, et l'insertion de quelques lignes dans les tables se font en exécutant les scripts contenus dans creationBaseComptes.sql

1 & 2 - Répartition des données : définition et création des fragments sur les deux sites

A partir de la base Comptes centralisée déjà mise en œuvre sur la base ENSIAS1 de la machine Serveur1, on désire construire une base de données répartie sur les deux sites : Serveur1 et Serveur2.

Les règles de répartition ou de fragmentation ont été définies en fonction de certains critères d'utilisation et de manipulation des données.

a- Fragmentation horizontale pour la table Client :

- Sur Serveur1 : la table Client_1 contenant les clients de Casablanca sans la colonne Ville.
  CREATE TABLE Client_1 (Nom, Prénom, Adresse CONSTRAINT pk1 PRIMARY KEY(No)) AS SELECT No,Nom, Prénom, Adresse FROM Client WHERE ville = 'Casablanca';
- Sur Serveur2 : la table Client_2 contenant les clients de Rabat sans la colonne Ville.
  COPY FROM Inwis/azerty@ensias1 TO Inwis/azerty@ensias2 REPLACE Client_2 (No,Nom, Prénom, Adresse) USING SELECT No,Nom, Prénom, Adresse FROM Client WHERE ville = 'Rabat';

b- Fragmentation horizontale pour la table Compte :

- Sur Serveur1 : la table Compte_1 avec les comptes appartenant aux clients de Casablanca.
  CREATE TABLE Compte_1 (No, Type_Compte_No, DateOuverture, Decouvert_autorise, Solde, Client_No, Agence_No, CONSTRAINT pk2 PRIMARY KEY(No)) AS SELECT No, Type_Compte_No, DateOuverture, Decouvert_autorise,

Solde, Client_No, Agence_No FROM Compte WHERE Client_No IN (SELECT No FROM Client_1);

- Sur Serveur2 : la table Compte_2 avec les comptes appartenant aux clients de Rabat.
  COPY FROM Inwis/azerty@ensias1 TO Inwis/azerty@ensias2 REPLACE TABLE Compte_2 (No, Type_Compte_No, DateOuverture, Decouvert_autorise, Solde, Client_No, Agence_No) USING SELECT No, Type_Compte_No, DateOuverture, Decouvert_autorise, Solde, Client_No, Agence_No FROM Compte WHERE Client_No IN (SELECT No FROM Client WHERE ville = 'Rabat');

c- Fragmentation horizontale pour la table Operation :

- Sur Serveur1 : la table Operation_1 correspondant aux opérations des comptes de la table Compte_1.
  CREATE TABLE Operation_1 (No, Type_Operation_No, Compte_No, CONSTRAINT pk3 PRIMARY KEY(No)) AS SELECT No, Type_Operation_No, Compte_No FROM Operation WHERE Compte_No IN (SELECT Compte_No FROM Compte_1);
- Sur Serveur2 : la table Operation_2 correspondant aux opérations des comptes de la table Compte_2.
  COPY FROM Inwis/azerty@ensias1 TO Inwis/azerty@ensias2 REPLACE Operation_2(No, Type_Operation_No, Compte_No) USING SELECT No, Type_Operation_No, Compte_No FROM Operation Where Compte_No IN (SELECT Compte_No FROM Compte WHERE Client_No IN (SELECT No FROM Client WHERE ville = 'Rabat');

d - Déplacement complet de la table Type_Compte sur Serveur2 : Type_Compte_2

COPY FROM Inwis/azerty@ensias1 TO Inwis/azerty@ensias2 REPLACE Type_Compte_2(no,libelle_compte,description) USING SELECT no,libelle_compte, description FROM Type_Compte;
DROP TABLE Type_Compte;

e - Déplacement complet de la table Type_Operation sur Serveur2 : Type_Operation_2

COPY FROM Inwis/azerty@ensias1 TO Inwis/azerty@ensias2 REPLACE Type_Operation_2(no, libelle_operation, decription) USING SELECT no, libelle_operation, description from Type_Operation;
DROP TABLE Type_Operation;

f - Les Séquences restent sur Serveur1.

Rien à faire.

3 - Création du lien inter – base (database link)

Sur chaque base (Serveur1 et Serveur2), créer deux database link (dbl_Ensias1 et dbl_Ensias2) permettant d'accéder aux objets distants.

CREATE DATABASE LINK dbl_ensias1 CONNECT TO Inwis IDENTIFIED BY azerty USING 'ensias1';
CREATE DATABASE LINK dbl_ensias2 CONNECT TO Inwis IDENTIFIED BY azerty USING 'ensias2';

Tester les liens établis sur chaque base en accédant aux objets distants dans les deux sens.

```
CONNECT Inwis/azerty@ensias2;
SELECT * FROM Client_1@dbl_ensias1;

CONNECT Inwis/azerty@ensias1;
SELECT * FROM Client_2@dbl_ensias2;
```

4 - Ajout des contraintes de base

Ajouter, sur chaque fragment, les contraintes initiales qui ont disparues :

(1) Les contraintes de clé primaire,

```
ALTER TABLE Client_2 ADD PRIMARY KEY (No);
ALTER TABLE Agence_2 ADD PRIMARY KEY (No);
ALTER TABLE Compte_2 ADD PRIMARY KEY (No);
ALTER TABLE Type_Compte_2 ADD PRIMARY KEY (No);
ALTER TABLE Operation_2 ADD PRIMARY KEY (No);
ALTER TABLE Type_Operation_2 ADD PRIMARY KEY (No);
```

(2) Les contraintes de références classiques si la table 'mère' est sur le même site,

```
ALTER TABLE Compte_2 ADD CONSTRAINT fk1 FOREIGN KEY(Client_No)
REFERENCES Client_2(No);
ALTER TABLE Operation_2 ADD CONSTRAINT fk2 FOREIGN KEY(Compte_No)
REFERENCES Compte_2(No);
```

(3) Les contraintes de références par 'trigger' si la table 'mère' est sur un site distant

Deux triggers :

→ un trigger sur la 'fille' remplaçant la FOREIGN KEY,

→ un trigger sur la 'mère' interdisant de supprimer une ligne référencée.

Pour la table Agence :

```
CONNECT Inwis/azerty@ensias1;

CREATE OR REPLACE TRIGGER trig_agence_mother
BEFORE DELETE OR UPDATE OF No ON Agence
FOR EACH ROW
DECLARE
x number := 0;
BEGIN
SELECT count(*) INTO x from Compte_2@dbl_ensias2
WHERE Agence_No = :OLD.No;
IF x>0 THEN
RAISE_APPLICATION_ERROR(-20175,'Cette agence est utilisée.');
END IF;
END;
/
```

```
CONNECT Inwis/azerty@ensias2;

CREATE OR REPLACE TRIGGER trig_agence_daughter
BEFORE INSERT OR UPDATE OF Agence_No ON Compte_2
FOR EACH ROW
DECLARE
x number := 0;
BEGIN
SELECT count(*) INTO x FROM Agence@dbl_ensias1
WHERE No = :NEW.Agence_No;
IF x=0 THEN
RAISE_APPLICATION_ERROR(-20175,'Cette agence est inexistante.');
END IF;
END;
/
```

5 - Création de la base répartie

(a) Ecrire les requêtes permettant de créer les objets virtuels répartis (view ou synonym) dans les deux dictionnaires : Serveur1 et Serveur2.

```
CONNECT Inwis/azerty@ensias1;

CREATE VIEW Client (No,Nom,Prenom,Adresse,Ville) AS
SELECT No,Nom,Prenom,Adresse,'Casablanca' FROM Client_1
UNION
SELECT No,Nom,Prenom,Adresse,'Rabat' FROM Client_2@dbl_ensias2;

CONNECT Inwis/azerty@ensias2;

CREATE VIEW Client (No,Nom,Prenom,Adresse,Ville) AS
SELECT No,Nom,Prenom,Adresse,'Casablanca' FROM Client_1@dbl_ensias1
UNION
SELECT No,Nom,Prenom,Adresse,'Rabat' FROM Client_2;
```

(b) Ecrire les requêtes de consultation de ces objets virtuels sur Serveur1 et sur Serveur2.

```
CONNECT Inwis/azerty@ensias1;
SELECT * FROM Client;

CONNECT Inwis/azerty@ensias2;
SELECT * FROM Client;
```

6 - Mise à jour en réparti : le commit (ou rollback) à deux phases

(a) Sur la base de Serveur1, insérer deux nouveaux clients : un dans la table Client_1 et un dans la table Client_2 distante.

INSERT INTO Client_1 VALUES('BOUKROUH', 'INSAF', 'Adr1');

INSERT INTO Client_2@dbl_ensias2 VALUES('DAOUDI', 'WISSAL', 'Adr2');

Vérifier dans chaque table la présence du nouveau client.

SELECT * FROM Client_1;

SELECT * FROM Client_2@dbl_ensias2;

(b) Sur la base de Serveur2, insérer deux autres clients : un dans la table Client_2 et un dans la table Client_1 distante.

INSERT INTO Client_2 VALUES('BOUKROUH', 'WISSAL', 'Adr3');

INSERT INTO Client_1@dbl_ensias1 VALUES('DAOUDI', 'INSAF', 'Adr4');

Vérifier dans chaque table la présence du nouveau client.

SELECT * FROM Client_2;

SELECT * FROM Client_1@dbl_ensias1;

(c) Sur la base de Serveur1, faites un COMMIT.

COMMIT;

(d) Sur la base de Serveur2, faites un ROLLBACK.

ROLLBACK;

Vérifier dans les deux tables des deux sites et commenter.

Le ROLLBACK simule une panne.

Toutes les insertions des 4 clients étaient annulées à cause de ce ROLLBACK, même s'il n'a été fait que sur la base de Serveur2.

TP2
Réplication synchrone et Réplication asynchrone

I - Mise en œuvre de la réplication synchrone

1- Copier la table centrale APPAREIL dans le site de Rabat (Serveur2).

```
CONNECT Inwis/azerty@ensias1;
CREATE TABLE Appareil(no_appareil number(7) PRIMARY KEY, designation varchar(30), prix number(7,2), caracteristiques_techniques varchar(50));
CONNECT Inwis/azerty@ensias2;
CREATE TABLE Appareil_Copy AS SELECT * FROM Appareil@dbl_ensias1;
```

2- Ecrire un trigger sur la base du siège (Serveur1) qui permet d'assurer que toute modification au niveau de la table centrale APPAREIL soit répercutée immédiatement vers l'image de cette table à Rabat.

Pour l'insertion :

```
CREATE TRIGGER insert_app
AFTER INSERT ON Appareil
FOR EACH ROW
BEGIN
INSERT INTO Appareil_Copy@dbl_ensias2 VALUES(:NEW.no_appareil, :NEW.designation, :NEW.prix, :NEW.caracteristiques_techniques);
END;
/
```

Pour la suppression :

```
CREATE TRIGGER delete_app
AFTER DELETE ON Appareil
FOR EACH ROW
BEGIN
DELETE FROM Appareil_Copy@dbl_ensias2 WHERE no_appareil = :OLD.no_appareil;
```

```
END;
/
```

Pour la mise à jour :

```
CREATE TRIGGER update_app
AFTER UPDATE ON Appareil
FOR EACH ROW
BEGIN
UPDATE Appareil_Copy @dbl_ensias2
SET no_appareil = :NEW.no_appareil, designation = :NEW.designation, prix = :NEW.prix,
caracteristiques_techniques = :NEW.caracteristiques_techniques
WHERE no_appareil = :OLD.no_appareil;
END;
/
```

3- Tester.

Tester le trigger de l'insertion :

```
INSERT INTO Appareil(1,'app1', 5300, 'cara1');
SELECT * FROM Appareil_Copy WHERE No = 1;
```

Tester le trigger de la suppression :

```
DELETE FROM Appareil WHERE No = 1;
SELECT * FROM Appareil_Copy WHERE No = 1;
```

Tester le trigger de la mise à jour :

```
UPDATE Appareil SET prix = 7100 WHERE No = 1;
SELECT * FROM Appareil_Copy WHERE No = 1;
```

II - Mise en œuvre de la réplication asynchrone

1- Créer une image (cliché) de la table centrale APPAREIL dans chacun des autres sites. Le rafraîchissement doit être rapide et sa mise à jour doit être effectuée toutes les 30 minutes.

```
CREATE SNAPSHOT LOG ON Appareil;
CREATE SNAPSHOT image_appareil
REFRESH FAST
```

START WITH SYSDATE

NEXT SYSDATE + 30

AS SELECT * FROM Appareil@dbl_ensias1;

2- Tester.

Pour pouvoir faire le test, on crée une image qui fait un rafraîchissement chaque 2 minutes par exemple.