INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

2017

Compte-rendu de TP : Bases de données distribuées

SGBD 2017 – TP 3

B3412 LUCAS POISSE - ZIGGY VERGNE
B3407 HUBERT HAMELIN - CYRIL POTTIEZ
B3414 JULIEN CHARLES-NICOLAS - HORIA BURCA

2017 – GROUPE 4 – B3412/B3407/B3414

COMPTE-RENDU DE TP : BASES DE DONNEES DISTRIBUEES

Table des matières

B3412 Lucas POISSE – Ziggy VERGNE
B3407 Hubert HAMELIN – Cyril POTTIEZ
B3414 Julien CHARLES-NICOLAS – Horia BURCA

I. Rappel rapide du contexte et des objectifs du TP

Ryori est une entreprise implantée dans plusieurs grandes régions distantes : les Amériques, l'Europe du Nord et du Sud. L'entreprise décide de fragmenter sa base de données en plusieurs morceaux afin de les répartir sur chacun de ses sites. L'objectif est de reproduire le comportement habituel de la base centralisée du point de vue de l'utilisateur et des applications, mais avec les données de la base réparties sur les trois sites.

Ainsi, notre objectif est d'effectuer cette répartition tout en maintenant la cohérence des données et les propriétés de la base centralisée. Ainsi, nous devons maintenir les contraintes de type « clé primaire » et « clé étrangère » sur l'ensemble des bases même si les informations dont nous avons besoin pour vérifier ces contraintes ne sont pas sur la même base de données que la nôtre. Nous devons donc effectuer des traitements spéciaux à l'insertion, la mise-à-jour et la suppression des tuples sur la base. De plus, nous devons veiller à attribuer chaque tuple à un seule et unique site et s'assurer que tous les tuples ont un site attribuée. De manière générale, la réunion des bases doit former la base originale.

II. Présentation du groupe de travail et des rôles de chacun

Notre groupe est constitué des binômes suivants :

- B3412, composé de Lucas Poisse et de Ziggy Vergne
- B3407, composé de Hubert Hamelin et de Cyril Pottiez
- B3414, composé de Julien Charles-Nicolas et Horia Burca

Après discussion, nous avons décidé de nous répartir les sites à gérer de la maniére suivante :

- Lucas Poisse et de Ziggy Vergne ont reçu l'Europe du Sud
- Hubert Hamelin et de Cyril Pottiez ont reçu l'Europe du Nord
- Julien Charles-Nicolas et Horia Burca ont reçu les Amériques

Naturellement, nous avons spécifié des personnes responsables des éléments clés du TP. Ainsi, la coordination générale fût confiée à Lucas Poisse, et la responsabilité de la documentation et du rapport fût confiée à Ziggy Vergne.

III. Fragmentation (Analyse globale et commune à tous les sites)

A. Détermination des fragments

Avant toute chose, nous allons déterminer les usages qui sont faits sur la base de données. Les applications utilisant la base sont MakeIt, DesignIt, SellIt, RH. Après

consultation de l'énoncé, nous avons extrait les informations suivantes sur l'utilisation de la base par les applications :

MakeIT :

- N'est utilisé qu'en Europe du Nord
- Lit, écrit et supprime des tuples dans la table Stock quand ils portent sur des objets localisés en Allemagne (et donc en Europe du Nord)
- Lit, écrit et met à jour des tuples dans la tables Fournisseurs

DesignIT :

- N'est utilisé qu'en Europe du Sud
- Lit, écrit et met à jour des tuples dans la tables Categories
- Lit, écrit et met à jour des tuples dans la table Produit
- C'est la seule application pouvant faire modifier Catégories et Produits

SellIT :

- Est déployée sur tous les sites
- Lit, écrit et met à jour des tuples dans les tables Commandes/DetailsCommandes pour les commandes portant sur les clients locaux
- Lit, écrit et met à jour des tuples dans Clients uniquement quand ce sont des clients locaux
- Lit, écrit et met à jour des tuples dans Stock concernent le stock local
- Lit (mais rarement) les tuples dans Stock concernant le stock des autres sites
- Lit les tuples de la table Produit/Categories/Fournisseurs/Employes

RH :

- Est localisée uniquement aux Amériques
- Lit, écrit et met à jour des tuples dans Employes

Nous allons nous préoccuper de l'application SellIt. Elle est présente sur les trois sites, il existe donc trois versions de l'application :

SellIt-US : La version de l'application SellIt pour le site des Amériques

SellIt-EN : La version de l'application SellIt pour le site d'Europe du Nord

SellIt-ES : La version de l'application SellIt pour le site d'Europe du Sud

Nous pouvons donc remarquer la présence de prédicats discriminants, liés à la localisation géographique du pays donc le nom est contenu dans l'attribut « Pays » des tuples de la table « Client » et « Stock ». Il nous est donné les informations suivantes vis-à-vis de la constitution des continents :

- Europe du Nord : Norvège, Suède, Danemark, Islande, Finlande, Royaume-Uni, Irlande, Belgique, Luxembourg, Pays-Bas, Allemagne, Pologne

- Europe du Sud : Espagne, Portugal, Andorre, France, Gibraltar, Italie, Saint Marin, Vatican, Malte, Albanie, Bosnie-Herzégovine, Croatie, Grèce, Macédoine, Monténégro, Serbie, Slovénie, Bulgarie
- Amériques : Antigua-et-Barbuda, Argentine, Bahamas, Barbade, Belize, Bolivie, Brésil, Canada, Chili, Colombie, Costa Rica, Cuba, République dominicaine, Dominique, Équateur, États-Unis, Grenade, Guatemala, Guyana, Haïti, Honduras, Jamaïque, Mexique, Nicaragua, Panama, Paraguay, Pérou, Saint-Christophe-et-Niévès, Sainte Lucie, Saint-Vincent-et-les Grenadines, Salvador, Suriname, Trinité-et-Tobago, Uruguay, Venezuela.

Après consultation de la base de données, nous remarquons qu'il existe des tuples dans la table Clients ayant pour Pays « Autriche » et « Suisse » alors que ces pays n'apparaissent dans aucun des ensembles de pays précédemment décrits. Après concertation, nous avons décidé les inclure dans "Europe du Sud".

Voici donc une conjonction de toutes les combinaisons possibles vis-à-vis de la valeur que peut prendre l'attribut Pays.

Possible ?	Pays IN EuropeSud	Pays IN EuropeNord	Pays IN Ameriques
Oui	Non	Non	Non
Oui	Non	Non	Oui
Oui	Non	Oui	Non
Non	Non	Oui	Oui
Oui	Oui	Non	Non
Non	Oui	Non	Oui
Non	Oui	Oui	Non
Non	Oui	Oui	Oui
Non	null	null	null

Nous remarquons que les intersections des 3 ensembles (Europe du Sud, Europe du Nord, Amériques) sont nulles entre elles. Il ne peut donc pas exister de pays référencés dans deux groupes. De même, la contrainte "NOT NULL" est appliquée sur les attributs Pays dans la base de données, le pays est donc forcément connu. Par contre, nous pouvons voir qu'un pays peut faire partie d'aucun des continents, son continent est donc considéré comme inconnu.

Nous pouvons donc voir que séparer les tuples par pays nous donne quatre fragments. Un pour les pays appartenant à l'Europe du Nord, un pour les pays appartenant à l'Europe du Sud, un pour les pays appartenant aux Amériques et un dernier pour les pays n'ayant pas de continents attribués.

N'ayant pas identifié d'autre prédicats discriminants et n'ayant pas besoin d'effectuer une fragmentation verticale des tables, nous allons donc procéder à la fragmentation des tables de la base.

Tout d'abord, nous remarquons que les seuls tuples faisant mention de « Pays » sont ceux dans la table « Clients » et « Stock ». Si nous fragmentons ces tables, il convient de fragmenter également les tables ayant des clefs étrangères référençant les tuples de ces deux tables ainsi que les tuples ayant des clefs étrangères sur ces dernières. Ainsi, en plus de fragmenter les tables « Stock » et « Client » nous allons également fragmenter les tables «Commandes », « DetailCommandes ».

Voici la base après fragmentation :

- Clients-US si Clients.Pays est dans l'ensemble "Amériques"
- Clients-ES si Clients.Pays est dans l'ensemble "Europe du Sud"
- Clients-EN si Clients.Pays est dans l'ensemble "Europe du Nord"
- Clients-OI si le continent d'origine du client est inconnu, c'est-à-dire si le pays du Client n'est pas contenu dans "Amériques", "Europe du Nord" ou "Europe du Sud".
- Stock-US si Stock.Pays est dans l'ensemble "Amériques"
- Stock-ES si Stock.Pays est dans l'ensemble "Europe du Sud"
- Stock-EN si Stock.Pays est dans l'ensemble "Europe du Nord"
- Stock-OI si le continent de Stock.Pays est inconnu, c'est-à-dire s'il n'est pas contenu dans "Amériques", "Europe du Nord" ou "Europe du Sud".
- Commandes-US si la commande a été passé par un Client contenu dans Clients-US
- Commandes-ES si la commande a été passé par un Client contenu dans Clients-ES
- Commandes-EN si la commande a été passé par un Client contenu dans Clients-EN
- Commandes-OI si la commande a été passé par un Client contenu dans Clients-OI
- DétailsCommandes-US si la commande associée est dans Commandes-US
- DétailsCommandes-ES si la commande associée est dans Commandes-ES
- DétailsCommandes-EN si la commande associée est dans Commandes-EN
- DétailsCommandes-OI si la commande associée est dans Commandes-OI
- Produits
- Catégories
- Fournisseurs
- Employes

Nous pouvons constater que la fragmentation s'effectue sans perte d'information et qu'elle permet la reconstruction de la base d'origine par une simple union. La

fragmentation ayant été effectuée, nous allons répartir les fragments sur les différents sites.

B. Placement des fragments sur les sites (sans réplication)

Nous allons tout d'abord voir la répartition des applications sur les différents sites. Nous avons choisi de considérer que l'application SellIt-OI allait être utilisée sur le site en Europe du Nord.

Application/Site	Europe du Nord	Europe du Sud	Amériques
MakeIt	X		
DesignIt		X	
SellIt-US			X
SellIt-ES		X	
SellIt-EN	X		
SellIt-OI	X		
RH			X

A partir des informations de l'énoncé sur l'utilisation des données, nous établissons un tableau d'utilisation des données de la base par les applications. Ce tableau nous permettra de répartir les fragments en fonction de leur utilisation. R signifie une utilisation en lecture et W une utilisation en écriture, la majuscule différencie les petites et les grandes utilisations.

Fragment/Site	MakeIt	DesignIt	RH	SellIt-US	SellIt-ES	SellIt-EN	SellIt-OI
Commandes-US				R/W			
Commandes-ES					R/W		
Commandes-EN						R/W	
Commandes-OI							R/W
DétailsCommandes-US				R/W			
DétailsCommandes-ES					R/W		
DétailsCommandes-EN						R/W	
DétailsCommandes-OI							R/W
Clients-US				R/W			
Clients-ES					R/W		

Clients-EN						R/W	
Clients-OI							R/W
Stock-US				R/W	r	r	r
Stock-ES				r	R/W	r	r
Stock-EN	RW			r	r	R/W	r
Stock-OI				r	r	r	R/W
Produits		RW					
Catégories		RW					
Fournisseurs	RW			R	R	R	R
Employés			R/W	R	R	R	R

A partir des deux tableaux, nous avons déduit la répartition des différents fragments manière à ne pas dupliquer l'information et de rapprocher les données des lieux les lisant le plus. Si plusieurs lieux lisent les mêmes données, nous plaçons le fragment sur le lieu pouvant éditer les données. Voici notre choix de répartition :

Fragment/Site	Europe du Nord	Europe du Sud	Amériques
Commandes-US			X
Commandes-ES		X	
Commandes-EN	X		
Commandes-OI	X		
DétailsCommandes-US			X
DétailsCommandes-ES		X	
DétailsCommandes-EN	X		
DétailsCommandes-OI	X		
Clients-US			X
Clients-ES		X	
Clients-EN	X		
Clients-OI	X		
Stock-US			X
Stock-ES		X	

Stock-EN	X		
Stock-OI	X		
Produits		X	
Catégories		X	
Fournisseurs	X		
Employés			X

Maintenant que chacun des fragments a trouvé un site, nous allons pouvoir passer à la mise en place de la base sans réplication.

C Mise en œuvre de la base sans réplication

1 Europe du Nord

Nous allons maintenant mettre en œuvre la base sans réplication pour le site de l'Europe du Nord.

a) Binôme responsable

Le binôme responsable de l'Europe du Nord est le binôme B3407, composé de Hubert Hamelin et de Cyril Pottiez.

b) Création des liens entre les bases

Nous créons les liens vers les bases de données des autres binômes. Veuillez noter que nous utilisons la machine de départ pour héberger le site de l'Europe du Nord, DB1, car la machine initialement prévue à cet effet, DB2, est non-fonctionnelle.

Nous appellons LinkToDBES le lien vers la base de données d'Europe du Sud et LinkToDBUS le lien vers la base des Amériques.

```
CREATE DATABASE LINK LinkToDBES CONNECT TO hhamelin IDENTIFIED BY mdporacle
USING 'DB3';
CREATE DATABASE LINK LinkToDBUS CONNECT TO hhamelin IDENTIFIED BY mdporacle
USING 'DB4';
```

d) Création des tables & peuplement des tables

```
-- création de la table pour les clients de l'europe du nord
CREATE TABLE clientsEN AS
  SELECT *
  FROM ryori.clients
  WHERE
    pays IN ('Suede', 'Norvege', 'Danemark', 'Finlande', 'Belgique',
'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne');

-- création de la table pour les clients d'un nouveau site
CREATE TABLE clientsOI AS
  SELECT *
  FROM ryori.clients
```

```
  WHERE pays = 'babla';

-- création table des commandes pour les clients de l'europe du nord
CREATE TABLE commandesEN AS
  SELECT r.*
  FROM clientsEN c, ryori.commandes r
  WHERE c.code_client = r.code_client;

-- création table des commandes pour les clients d'un nouveau site
CREATE TABLE commandesOI AS
  SELECT r.*
  FROM clientsOI c, ryori.commandes r
  WHERE c.code_client = r.code_client;

-- création table de détails des commande pour les clients de l'europe du
nord
CREATE TABLE details_commandesEN AS
  SELECT r.*
  FROM commandesEN c, ryori.details_commandes r
  WHERE c.no_commande = r.no_commande;

-- création table de détails des commande pour les clients d'un nouveau
site
CREATE TABLE details_commandesOI AS
  SELECT r.*
  FROM commandesOI c, ryori.details_commandes r
  WHERE c.no_commande = r.no_commande;

--création table de stock des sites de l'europe du nord
CREATE TABLE stockEN AS
  SELECT DISTINCT r.*
  FROM clientsEN c, ryori.stock r
  WHERE c.pays = r.pays;

--création table de stock des nouveaux sites
CREATE TABLE stockOI AS
  SELECT DISTINCT r.*
  FROM clientsOI c, ryori.stock r
  WHERE c.pays = r.pays;

-- table fournisseurs
CREATE TABLE fournisseurs AS
  SELECT *
  FROM ryori.fournisseurs;
```

c) Contraintes d'intégrité

Tout d'abord, nous ajoutons les contraintes pouvant être gérées en local, c'est-à-dire les contraintes de type clef primaire et certaines contraintes de type clef étrangères.

```
-- contraintes de clé primaire
ALTER TABLE ClientsEN
  ADD CONSTRAINT ClientsENPk PRIMARY KEY (code_client);

ALTER TABLE ClientsOI
```

```
  ADD CONSTRAINT ClientsOIPk PRIMARY KEY (code_client);

ALTER TABLE commandesEN
  ADD CONSTRAINT CommandesENPk PRIMARY KEY (no_commande);

ALTER TABLE commandesOI
  ADD CONSTRAINT CommandesOIPk PRIMARY KEY (no_commande);

ALTER TABLE details_commandesEN
  ADD CONSTRAINT Details_commandesENPk PRIMARY KEY (no_commande,
ref_produit);

ALTER TABLE details_commandesOI
  ADD CONSTRAINT Details_commandesOIPk PRIMARY KEY (no_commande,
ref_produit);

ALTER TABLE fournisseurs
  ADD CONSTRAINT FournisseursPk PRIMARY KEY (no_fournisseur);

ALTER TABLE stockEN
  ADD CONSTRAINT StockENPk PRIMARY KEY (ref_produit, pays);

ALTER TABLE stockOI
  ADD CONSTRAINT StockOIPk PRIMARY KEY (ref_produit, pays);

--contraintes de clé étrangère

ALTER TABLE commandesEN
  ADD CONSTRAINT Fk_ClientsEN FOREIGN KEY (code_client) REFERENCES
clientsEN (code_client);

ALTER TABLE commandesOI
  ADD CONSTRAINT Fk_ClientsOI FOREIGN KEY (CODE_CLIENT) REFERENCES
CLIENTSOI (CODE_CLIENT);


ALTER TABLE DETAILS_COMMANDESEN
  ADD CONSTRAINT Fk_CommandesEN FOREIGN KEY (NO_COMMANDE) REFERENCES
COMMANDESEN (NO_COMMANDE);

ALTER TABLE DETAILS_COMMANDESOI
  ADD CONSTRAINT Fk_CommandesOI FOREIGN KEY (NO_COMMANDE) REFERENCES
COMMANDESOI (NO_COMMANDE);
```

De plus, nous devons ajouter des contraintes supplémentaire dans les tables CommandesEN, CommandesOI, DetailsCommandesEN, DetailsCommandesOI, StockEN, StockOI et Fournisseurs car ces tables ont besoin de vérifier l'états de tables distantes pour maintenir leurs contraintes de clef étrangére.

```
-- trigger pour vérifier la présence de l'employé à l'insertion ou maj dans
CommandesEN
CREATE OR REPLACE TRIGGER trg_check_employesEN
BEFORE INSERT OR UPDATE ON COMMANDESEN
FOR EACH ROW
  DECLARE
    Employee NUMBER(6);
  BEGIN
    SELECT e.no_employe
```

```
    INTO employee
    FROM hcburca.Employes@LinkToDBUS e
    WHERE e.no_employe = :new.no_employe;

    EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001,'Invalid no_employe : There is no
Employe with this no_employe in the company !');

  END;

-- trigger pour vérifier la présence de l'employé à l'insertion ou maj dans
CommandesOI
CREATE OR REPLACE TRIGGER trg_check_employesOI
BEFORE INSERT OR UPDATE ON COMMANDESOI
FOR EACH ROW
  DECLARE
    Employee NUMBER(6);
  BEGIN
    SELECT e.no_employe
    INTO employee
    FROM hcburca.Employes@LinkToDBUS e
    WHERE e.no_employe = :new.no_employe;

    EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001,'Invalid no_employe : There is no
Employe with this no_employe in the company !');

  END;

--  trigger pour vérifier la présence du produit à insérer dans
DETAILS_COMMANDESEN
CREATE OR REPLACE TRIGGER trg_check_CEN_produits
BEFORE INSERT OR UPDATE ON DETAILS_COMMANDESEN
FOR EACH ROW
  DECLARE
    produit NUMBER(6);
  BEGIN
    SELECT p.ref_produit
    INTO produit
    FROM lpoisse.produits@LinkToDBES p
    WHERE p.ref_produit = :new.REF_PRODUIT;

    EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Invalid ref_produit : There is no
product with this ref_produit in the company !');
  END;

--  trigger pour vérifier la présence du produit à insérer dans
DETAILS_COMMANDESOI
CREATE OR REPLACE TRIGGER trg_check_COI_produits
BEFORE INSERT OR UPDATE ON DETAILS_COMMANDESOI
FOR EACH ROW
  DECLARE
    produit NUMBER(6);
  BEGIN
```

```
    SELECT p.ref_produit
    INTO produit
    FROM lpoisse.produits@LinkToDBES p
    WHERE p.ref_produit = :new.REF_PRODUIT;

    EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001,'Invalid ref_produit : There is no
ref_produit with this ref_produit in the company !');
  END;

--  trigger pour vérifier si le produit inséré est bien référencé dans
stockEN
CREATE OR REPLACE TRIGGER trg_check_stockEN_produits
BEFORE INSERT OR UPDATE ON STOCKEN
FOR EACH ROW
  DECLARE
    produit NUMBER(6);
  BEGIN
    SELECT p.ref_produit
    INTO produit
    FROM lpoisse.produits@LinkToDBES p
    WHERE p.ref_produit = :new.REF_PRODUIT;

    EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Invalid ref_produit : There is no
ref_produit with this ref_produit in the company !');
  END;

--  trigger pour vérifier si le produit inséré est bien référencé dans
STOCKOI
CREATE OR REPLACE TRIGGER trg_check_stockoi_produits
BEFORE INSERT OR UPDATE ON STOCKOI
FOR EACH ROW
  DECLARE
    produit NUMBER(6);
  BEGIN
    SELECT p.ref_produit
    INTO produit
    FROM lpoisse.produits@LinkToDBES p
    WHERE p.ref_produit = :new.REF_PRODUIT;

    EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Invalid ref_produit : There is no
ref_produit with this ref_produit in the company !');
  END;

-- vérifier si le fournisseurs à supprimer/mettre à jour n'est déjà
référencé dans la table produits
CREATE OR REPLACE TRIGGER chk_Fournisseurs
BEFORE DELETE OR UPDATE ON FOURNISSEURS
FOR EACH ROW
  DECLARE
    number_of_rows NUMBER; -- nombre de tuples trouvés dans table produit
pour le fournisseurs qui doit être supprimé
```

```
BEGIN

  SELECT count(*)
  INTO number_of_rows
  FROM lpoisse.produits@LinkToDBES
  WHERE NO_FOURNISSEUR = :old.NO_FOURNISSEUR;

  IF (DELETING)
  THEN
    IF number_of_rows <> 0
    THEN
      raise_application_error(-20002,'Erreur : le fournisseur à supprimer
est déjà référencé dans la table produits en europe du sud');
    END IF;
  END IF;

  IF (UPDATING)
  THEN
    IF number_of_rows <> 0
    THEN
      raise_application_error(-20002, 'Erreur : le fournisseur à mettre à
jour est déjà référencé dans la table produits en europe du sud');
    END IF;
  END IF;

END;
```

d) Droits d'accès

Nous donnons les droits d'accès en lecture, écriture et suppressions sur les tables de notre base aux autres membres du groupe.

```
GRANT SELECT, update, insert, delete ON stockEN TO lpoisse;
GRANT SELECT, update, insert, delete ON stockEN TO zvergne;
GRANT SELECT, update, insert, delete ON stockEN TO hcburca;
GRANT SELECT, update, insert, delete ON stockEN TO jcharlesni;
GRANT SELECT, update, insert, delete ON stockEN TO hhamelin;

GRANT SELECT, update, insert, delete ON stockOI TO lpoisse;
GRANT SELECT, update, insert, delete ON stockOI TO zvergne;
GRANT SELECT, update, insert, delete ON stockOI TO hcburca;
GRANT SELECT, update, insert, delete ON stockOI TO jcharlesni;
GRANT SELECT, update, insert, delete ON stockOI TO hhamelin;

GRANT SELECT, update, insert, delete ON fournisseurs TO lpoisse;
GRANT SELECT, update, insert, delete ON fournisseurs TO zvergne;
GRANT SELECT, update, insert, delete ON fournisseurs TO hcburca;
GRANT SELECT, update, insert, delete ON fournisseurs TO jcharlesni;
GRANT SELECT, update, insert, delete ON fournisseurs TO hhamelin;

GRANT SELECT, update, insert, delete ON COMMANDESEN TO lpoisse;
GRANT SELECT, update, insert, delete ON COMMANDESEN TO zvergne;
GRANT SELECT, update, insert, delete ON COMMANDESEN TO hcburca;
GRANT SELECT, update, insert, delete ON COMMANDESEN TO jcharlesni;
GRANT SELECT, update, insert, delete ON COMMANDESEN TO hhamelin;
```

```
GRANT SELECT, update, insert, delete ON commandesoi TO lpoisse;
GRANT SELECT, update, insert, delete ON commandesoi TO zvergne;
GRANT SELECT, update, insert, delete ON commandesoi TO hcburca;
GRANT SELECT, update, insert, delete ON commandesoi TO jcharlesni;
GRANT SELECT, update, insert, delete ON commandesoi TO hhamelin;

GRANT SELECT, update, insert, delete ON details_commandesoi TO lpoisse;
GRANT SELECT, update, insert, delete ON details_commandesoi TO zvergne;
GRANT SELECT, update, insert, delete ON details_commandesoi TO hcburca;
GRANT SELECT, update, insert, delete ON details_commandesoi TO jcharlesni;
GRANT SELECT, update, insert, delete ON details_commandesoi TO hhamelin;

GRANT SELECT, update, insert, delete ON details_commandesEn TO lpoisse;
GRANT SELECT, update, insert, delete ON details_commandesEn TO zvergne;
GRANT SELECT, update, insert, delete ON details_commandesEn TO hcburca;
GRANT SELECT, update, insert, delete ON details_commandesEn TO jcharlesni;
GRANT SELECT, update, insert, delete ON details_commandesEn TO hhamelin;

GRANT SELECT, update, insert, delete ON clientsen TO lpoisse;
GRANT SELECT, update, insert, delete ON clientsen TO zvergne;
GRANT SELECT, update, insert, delete ON clientsen TO hcburca;
GRANT SELECT, update, insert, delete ON clientsen TO jcharlesni;
GRANT SELECT, update, insert, delete ON clientsen TO hhamelin;

GRANT SELECT, update, insert, delete ON clientsoi TO lpoisse;
GRANT SELECT, update, insert, delete ON clientsoi TO zvergne;
GRANT SELECT, update, insert, delete ON clientsoi TO hcburca;
GRANT SELECT, update, insert, delete ON clientsoi TO jcharlesni;
GRANT SELECT, update, insert, delete ON clientsoi TO hhamelin;
```

2 Europe du Sud

a) Binôme responsable

Le binôme responsable de l'Europe du Sud est le binôme B3412, constitué de Lucas Poisse et de Ziggy Vergne.

b) Création des liens entre les bases

Nous créons les liens vers les bases de données des autres binômes. Veuillez noter que nous utilisons la machine de départ pour héberger le site de l'Europe du Nord, DB1, car la machine initialement prévue à cet effet, DB2, est non-fonctionnelle.

Nous appellons dblinkMain le lien vers la base de données d'Europe du Nord et DBlinkUS le lien vers la base des Amériques.

```
--Liens de BDD
create database link dblinkMain CONNECT TO lpoisse IDENTIFIED BY mdporacle
USING 'DB1';
CREATE DATABASE LINK dbLinkUS CONNECT TO lpoisse IDENTIFIED BY mdporacle
USING 'DB4';
```

c) Création des tables & Peuplement des tables

Nous procédons à la création des différentes tables sur notre base et nous les remplissons aussitôt grâce à la commande CREATE TABLE AS. Le remplissage se fait en sélectionnant les données en fonction des prédicats définis précédemment.

```
--Création de la table clients de l'Europe du Sud AVEC Autriche/Suisse
--Nous nommons cette table « clientsES »
CREATE TABLE clientsES as
(SELECT * FROM ryori.clients@dblinkMain
WHERE pays IN ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-
Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro', 'Serbie',
'Slovenie', 'Bulgarie', 'Autriche', 'Suisse'));

--Création de la table commandes de l'europe du Sud (CommandesES)
--Pour que une commande soit sélectionnée il faut qu'elle soit relative à
--un client du continent Europe du Sud

CREATE TABLE commandesES as
(
SELECT * from ryori.commandes@dblinkMain WHERE NO_COMMANDE IN (
SELECT NO_COMMANDE FROM (
SELECT * FROM ryori.commandes@dblinkMain com NATURAL JOIN
ryori.clients@dblinkMain cli
WHERE cli.pays IN ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-
Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro', 'Serbie',
'Slovenie', 'Bulgarie', 'Autriche', 'Suisse'))
));


-- Création de la table détails commande de l'europe du Sud
-- (details_CommandesES)
-- Pour que une commande soit sélectionnée il faut qu'elle soit relative à
-- un client du continent Europe du Sud
CREATE TABLE details_commandesES as
(
SELECT * from ryori.details_commandes@dblinkMain WHERE NO_COMMANDE IN (
SELECT NO_COMMANDE FROM (
SELECT * FROM ryori.commandes@dblinkMain com NATURAL JOIN
ryori.clients@dblinkMain cli
WHERE cli.pays IN ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-
Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro', 'Serbie',
'Slovenie', 'Bulgarie', 'Autriche', 'Suisse'))
));



 -- Création de la table stock de l'europe du Sud (stockES)
CREATE TABLE stockES as
(
SELECT * from ryori.stock@dblinkMain
WHERE pays IN ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-
```

```
Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro', 'Serbie',
'Slovenie', 'Bulgarie', 'Autriche', 'Suisse'));


 --Création de la table Produits à partir de l'originale
CREATE TABLE produits as
(
SELECT * from ryori.produits@dblinkMain);



 --Création de la table Categories à partir de l'originale
CREATE TABLE CATEGORIES as
(
SELECT * from ryori.CATEGORIES@dblinkMain);
```

e) Contraintes d'intégrité

Nous devons désormais appliquer des contraintes d'intégrités à nos tables. Ces contraintes doivent coller aux contraintes présentes sur la base originale. Les contraintes de la base originale sont de type « clef primaire », « clef etrangere », « not null ». L'écriture des contraintes peut donc se séparer en deux parties : la partie « locale » qui ne concerne que les tuples présents en local et la partie déportée qui elle fait intervenir des Triggers qui sont nécessaires pour regarder et éditer le contenu des autres bases de données pour maintenir la cohérence des données en cas d'insertion, de modification et suppression.

Voici la partie « locale » :

Clefs primaires :

```
      ALTER TABLE clientsES ADD CONSTRAINT pk_clientsES PRIMARY KEY
(CODE_CLIENT);
      ALTER TABLE commandesES ADD CONSTRAINT pk_commandesES PRIMARY KEY
(NO_COMMANDE);
      ALTER TABLE details_commandesES ADD CONSTRAINT pk_detailsCommandesES
PRIMARY KEY (NO_COMMANDE, REF_PRODUIT);
      ALTER TABLE stockES ADD CONSTRAINT pk_StockES PRIMARY KEY
(REF_PRODUIT, PAYS);
      ALTER TABLE produits add constraint pk_produits PRIMARY KEY
(REF_PRODUIT);
      ALTER TABLE categories ADD CONSTRAINT pk_Categories PRIMARY KEY
(CODE_CATEGORIE);
```

Not null :

```
      ALTER TABLE CommandesES ADD CONSTRAINT chk_ccnotnull CHECK
(CODE_CLIENT IS NOT NULL);
      ALTER TABLE CommandesES ADD CONSTRAINT chk_noempnotnull CHECK
(NO_EMPLOYE IS NOT NULL);
      ALTER TABLE CommandesES ADD CONSTRAINT chk_datecnotnull CHECK
(DATE_COMMANDE IS NOT NULL);

      ALTER TABLE CLIENTSES ADD CONSTRAINT chk_socnotnull CHECK (societe IS
NOT NULL);
```

```
      ALTER TABLE CLIENTSES ADD CONSTRAINT chk_adrnull CHECK (adresse IS
NOT NULL);
      ALTER TABLE CLIENTSES ADD CONSTRAINT chk_villenotnull CHECK (ville IS
NOT NULL);
      ALTER TABLE CLIENTSES ADD CONSTRAINT chk_cpnotnull CHECK (code_postal
IS NOT NULL);
      ALTER TABLE CLIENTSES ADD CONSTRAINT chk_paysClientnotnull CHECK
(pays IS NOT NULL);
      ALTER TABLE CLIENTSES ADD CONSTRAINT chk_telnotnull CHECK (telephone
IS NOT NULL);

      ALTER TABLE DETAILS_Commandeses ADD CONSTRAINT chk_nocom CHECK
(no_commande IS NOT NULL);
      ALTER TABLE DETAILS_Commandeses ADD CONSTRAINT chk_refpdtnotnull
CHECK (REF_PRODUIT IS NOT NULL);
      ALTER TABLE DETAILS_Commandeses ADD CONSTRAINT chk_unprixnotnull
CHECK (PRIX_UNITAIRE IS NOT NULL);
      ALTER TABLE DETAILS_Commandeses ADD CONSTRAINT chk_quantnotnull CHECK
(quantite IS NOT NULL);
      ALTER TABLE DETAILS_Commandeses ADD CONSTRAINT chk_remisenotnull
CHECK (remise IS NOT NULL);

      ALTER TABLE stockes ADD CONSTRAINT chk_stockrefpdtnotnull CHECK
(ref_produit IS NOT NULL);
      ALTER TABLE stockes ADD CONSTRAINT chk_stockespays CHECK (pays IS NOT
NULL);
```

Clefs étrangères :

```
--FK possibles pour assurer les clés étrangères locales

      alter table details_commandeses add constraint
fk_detailscmdesproduits foreign key (REF_PRODUIT) REFERENCES Produits;
      alter table details_commandeses add constraint fk_detailsCmdeCmde
foreign key (no_commande) references commandeses;
      alter table stockES add constraint fk_stockESproduits foreign key
(REF_PRODUIT) REFERENCES Produits;
      alter table Produits add constraint fk_ProduitsCategories foreign key
(code_categorie) references categories;
      alter table Commandeses add constraint FK_CommandesesClientses
foreign key (code_client) references clientses;
```

Partie déportée (Triggers)

Pour la table Produits:

Notre trigger est chargé de vérifier si le fournisseur du produit existe bien dans la table fournisseur située sur le site en Europe du Nord en cas d'ajout et de modification de la table Produits. De plus, en cas de suppression, il est chargé de vérifier si il n'existe pas de tuples dans les autres bases référençant le produit qu'on tente de supprimer. Si c'est le cas, la suppression est refusée.

```
create or replace TRIGGER chk_Produits BEFORE INSERT OR UPDATE OR DELETE ON
Produits
FOR EACH ROW
DECLARE
```

```
idFourn number; --Id du fournisseur renseigné à vérifier
any_rows_found NUMBER;  --variable indiquant si un produit à supprimer
existe dans une table secondaire

BEGIN
      IF INSERTING OR UPDATING THEN
        SELECT NO_FOURNISSEUR INTO idFourn
        from cpottiez.Fournisseurs@dblinkMain rel
        where rel.NO_FOURNISSEUR = :NEW.NO_FOURNISSEUR;
      END IF;

  IF DELETING THEN

    SELECT count(*) into any_rows_found
    from cpottiez.details_commandesEN@dblinkMain
    where ref_produit = :NEW.REF_PRODUIT;

    IF any_rows_found <> 0 THEN
      raise_application_error(-20002, 'Erreur : le produit à supprimer est
déjà référencé dans la table DétailsCommandes en Europe du Nord');
    end if;

    SELECT count(*) INTO any_rows_found
    FROM cpottiez.details_commandesOI@dblinkMain
    WHERE ref_produit = :NEW.REF_PRODUIT;

    IF any_rows_found <> 0 THEN
      raise_application_error(-20003, 'Erreur : le produit à supprimer est
déjà référencé dans la table DétailsCommandes pour un pays inconnu');
    end if;

    SELECT count(*) INTO any_rows_found
    FROM hcburca.Details_Commandes_AM@dbLinkUS
    WHERE ref_produit = :NEW.REF_PRODUIT;

    IF any_rows_found <> 0 THEN
      raise_application_error(-20004, 'Erreur : le produit à supprimer est
déjà référencé dans la table DétailsCommandes en Amérique');
    end if;

    SELECT count(*) INTO any_rows_found
    FROM cpottiez.stockEN@dblinkMain
    WHERE ref_produit = :NEW.REF_PRODUIT;

    IF any_rows_found <> 0 THEN
      raise_application_error(-20005, 'Erreur : le produit à supprimer est
déjà référencé dans la table Stocks en Europe du Nord');
    end if;

    SELECT count(*) INTO any_rows_found
    FROM cpottiez.stockOI@dblinkMain
    WHERE ref_produit = :NEW.REF_PRODUIT;

    IF any_rows_found <> 0 THEN
      raise_application_error(-20006, 'Erreur : le produit à supprimer est
déjà référencé dans la table Stocks pour un pays inconnu');
    end if;
```

```
    SELECT count(*) INTO any_rows_found
    FROM hcburca.Stock_AM@dbLinkUS
    WHERE ref_produit = :NEW.REF_PRODUIT;

    IF any_rows_found <> 0 THEN
      raise_application_error(-20007, 'Erreur : le produit à supprimer est
déjà référencé dans la table Stocks en Amérique');
    end if;
  END IF;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20002, 'Erreur : tout fournisseur référencé
doit exister dans la table des fournisseurs');
END;
```

Pour la table CommandesES :

Notre trigger sur Commandes ES est chargé de vérifer que la commande que l'on cherche à modifier ou insérer fait bien référence à un employé existant. Pour cela, le trigger regarde si un empoyé comportant cet id existe bien dans la base Employe située aux Amériques

```
create or replace TRIGGER chkInsert_Commandes BEFORE INSERT OR UPDATE ON
CommandesES
FOR EACH ROW
DECLARE
idEmp number;

BEGIN
  SELECT No_employe INTO idEmp
  from hcburca.Employes@dbLinkUS rel
  where rel.no_employe = :NEW.no_employe;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Erreur : tout employé référencé doit
exister dans la table des employés');
END;
```

e) Droits d'accès

Nous donnons les droits d'accès à notre base aux autres membres du groupe. Nous rappelons que, comme en centralisé, nous donnons les droits de modification, suppression, insertion et consultation à tous les autres sites.

```
/*
  Permissions accordées : STOCKES (lecture/mise à
jour/insertion/suppression depuis les applications externes)
*/
GRANT SELECT, update, insert, delete ON stockES to cpottiez;
GRANT SELECT, update, insert, delete ON stockES to hhamelin;
GRANT SELECT, update, insert, delete ON stockES to zvergne;
GRANT SELECT, update, insert, delete ON stockES to jcharlesni;
GRANT SELECT, update, insert, delete ON stockES to hcburca;
```

```
/*
  Permissions accordées : PRODUITS (lecture/mise à
jour/insertion/suppression depuis les applications externes)
*/
GRANT SELECT, update, insert, delete ON produits to cpottiez;
GRANT SELECT, update, insert, delete ON produits to hhamelin;
GRANT SELECT, update, insert, delete ON produits to zvergne;
GRANT SELECT, update, insert, delete ON produits to jcharlesni;
GRANT SELECT, update, insert, delete ON produits to hcburca;

/*
  Permissions accordées : CATEGORIES (lecture/mise à
jour/insertion/suppression depuis les applications externes)
*/
GRANT SELECT, update, insert, delete ON categories to cpottiez;
GRANT SELECT, update, insert, delete ON categories to hhamelin;
GRANT SELECT, update, insert, delete ON categories to zvergne;
GRANT SELECT, update, insert, delete ON categories to jcharlesni;
GRANT SELECT, update, insert, delete ON categories to hcburca;

/*
  Permissions accordées : DETAILS_COMMANDESES (lecture/mise à
jour/insertion/suppression depuis les applications externes)
*/
GRANT SELECT, update, insert, delete ON details_commandeses to cpottiez;
GRANT SELECT, update, insert, delete ON details_commandeses to hhamelin;
GRANT SELECT, update, insert, delete ON details_commandeses to zvergne;
GRANT SELECT, update, insert, delete ON details_commandeses to jcharlesni;
GRANT SELECT, update, insert, delete ON details_commandeses to hcburca;

/*
  Permissions accordées : COMMANDESES (lecture/mise à
jour/insertion/suppression depuis les applications externes)
*/
GRANT SELECT, update, insert, delete on commandeses TO cpottiez;
GRANT SELECT, update, insert, delete on commandeses TO hhamelin;
GRANT SELECT, update, insert, delete on commandeses TO zvergne;
GRANT SELECT, update, insert, delete on commandeses TO jcharlesni;
GRANT SELECT, update, insert, delete on commandeses TO hcburca;

/*
  Permissions accordées : CLIENTSES (lecture/mise à
jour/insertion/suppression depuis les applications externes)
*/
GRANT SELECT, update, insert, delete on clientses TO cpottiez;
GRANT SELECT, update, insert, delete on clientses TO hhamelin;
GRANT SELECT, update, insert, delete on clientses TO zvergne;
GRANT SELECT, update, insert, delete on clientses TO jcharlesni;
GRANT SELECT, update, insert, delete on clientses TO hcburca;
```

f) Définition de synonymes et de vues pour interrogation de la base comme si elle était en centralisé.

Afin de rendre les changements transparents pour les applications, nous allons désormais ajouter des vues à notre base de données. Les vues correspondront à l'union des fragments dispersés sur les trois sites.

Notez l'utilisation du mot « WHERE » pour indiquer au SGBD le contenu des vues et ainsi lui permettre d'optimiser son plan d'exécution.

```
-- Vue "Stock", création avec WHERE pour optimiser le plan d'exécution
CREATE OR REPLACE VIEW Stock
AS
(SELECT * FROM StockES where StockES.PAYS in
('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie', 'Saint-
Marin', 'Vatican',
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')
UNION ALL
SELECT * FROM cpottiez.stockEN@dblinkMain where pays in ('Suede',
'Norvege', 'Danemark', 'Finlande', 'Belgique', 'Irlande', 'Pologne',
'Royaume-Uni', 'Allemagne', 'Islande', 'Luxembourg', 'Pays-Bas')
UNION ALL
SELECT * FROM cpottiez.stockOI@dblinkMain where pays not in ('Antigua-et-
Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela') and pays not in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')
  and pays not in ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican',
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')
UNION ALL
SELECT * FROM hcburca.stock_am@dbLinkUS where pays in ('Antigua-et-
Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela')
);

--Vue 'Clients', création avec WHERE pour optimiser le plan d'exécution

CREATE OR REPLACE VIEW Clients
AS
(SELECT * FROM ClientsES where PAYS in
('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie', 'Saint-
Marin', 'Vatican',
```

```
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')
UNION ALL
SELECT * FROM cpottiez.clientsEN@dblinkMain where pays in ('Suede',
'Norvege', 'Danemark', 'Finlande', 'Belgique', 'Irlande', 'Pologne',
'Royaume-Uni', 'Allemagne', 'Islande', 'Luxembourg', 'Pays-Bas')
UNION ALL
SELECT * FROM cpottiez.clientsOI@dblinkMain where pays not in ('Antigua-et-
Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela') and pays not in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')
  and pays not in ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican',
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')
UNION ALL
SELECT * FROM hcburca.clients_am@dbLinkUS where pays in ('Antigua-et-
Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela')
);

--Vue 'Commandes'
CREATE OR REPLACE VIEW Commandes
AS
(SELECT * FROM Commandeses
UNION ALL
SELECT * FROM cpottiez.commandesEN@dblinkMain
UNION ALL
SELECT * FROM cpottiez.commandesOI@dblinkMain
UNION ALL
SELECT * FROM hcburca.Commandes_AM@dbLinkUS
);

-- Vue 'Details_Commandes'
CREATE OR REPLACE VIEW details_commandes
AS
(SELECT * FROM details_commandeses
UNION ALL
SELECT * FROM cpottiez.details_commandesEN@dblinkMain
```

```
UNION ALL
SELECT * FROM cpottiez.details_commandesOI@dblinkMain
UNION ALL
SELECT * FROM hcburca.Details_Commandes_AM@dbLinkUS
);

--Vue fournisseurs
CREATE OR REPLACE VIEW Fournisseurs
AS
(SELECT * FROM cpottiez.fournisseurs@dblinkmain);

-- Vue employés
CREATE OR REPLACE VIEW employes
AS
(SELECT * FROM hcburca.Employes@dbLinkUS
);
```

Nous devons rediriger les requêtes de mise à jour, d'insertion et de suppression en fonction de la localisation des tuples que nous souhaitons modifier. Veuillez noter que nous acceptons qu'une application située sur un autre site puisse modifier des tuples présents sur un site tiers, ce qui signifie que l'application SellIT-ES peut modifier des données du stock d'Amériques alors que ces derniers sont utilisés par l'application SellIT-US. Ceci a été fait pour calquer le comportement de la base centralisée sur la base distribuée. L'ajout de l'interdiction des sites de modifier les tuples des autres sites pourra être ajouté si besoin par une simple modification des triggers suivants.

Pour la vue stock :

Ce trigger est chargé de rediriger les modifications, insertions ou suppressions de tuples vers la base de données contenant ce tuples. Veuillez noter que le cas de modification du pays de l'élément du stock, le trigger déplace le stock vers la base correspondant à l'appartenance du pays nouvellement associé si besoin.

```
CREATE OR REPLACE TRIGGER modify_Stocks INSTEAD OF
  UPDATE OR
  INSERT OR
  DELETE ON Stock
FOR EACH ROW
BEGIN
IF INSERTING THEN --Insertion à contrôler
      IF (:NEW.pays IN ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')) THEN
  INSERT
  INTO cpottiez.STOCKEN@dblinkMain VALUES
      (
      :new.ref_produit,
      :new.pays,
      :new.unites_stock,
      :new.unites_commandees,
      :new.indisponible
      );
ELSIF (:NEW.pays IN ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
```

```
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) THEN
  INSERT
  INTO stockes VALUES
      (
      :new.ref_produit,
      :new.pays,
      :new.unites_stock,
      :new.unites_commandees,
      :new.indisponible
      );
ELSIF (:NEW.pays IN('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine', 'Dominique',
'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) THEN
  INSERT
  INTO hcburca.stock_am@dbLinkUS VALUES
      (
      :new.ref_produit,
      :new.pays,
      :new.unites_stock,
      :new.unites_commandees,
      :new.indisponible
      );
ELSE
  INSERT
  INTO cpottiez.STOCKOI@dblinkMain VALUES
      (
      :new.ref_produit,
      :new.pays,
      :new.unites_stock,
      :new.unites_commandees,
      :new.indisponible
      );
END IF;
END IF;
IF UPDATING THEN
  if(:New.pays <> :OLD.pays) then
     --RAISE_APPLICATION_ERROR(-20013,'Attention, vous n''êtes pas autorisé
à changer le pays du stock');
     Delete from stock WHERE ref_produit = :old.ref_produit AND pays =
:old.pays;
       INSERT
     INTO stock VALUES
    (
      :new.ref_produit,
      :new.pays,
      :new.unites_stock,
      :new.unites_commandees,
      :new.indisponible
      );
```

```
  elsIF (:NEW.pays IN ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')) THEN
      UPDATE cpottiez.STOCKEN@dblinkMain
      SET ref_produit   = :new.ref_produit,
      PAYS              = :NEW.PAYS,
      UNITES_STOCK      = :new.UNITES_STOCK,
      UNITES_COMMANDEES = :new.UNITES_COMMANDEES,
      INDISPONIBLE      = :new.INDISPONIBLE
      WHERE ref_produit   = :old.ref_produit
      AND pays          = :old.PAYS;
  ELSIF (:NEW.pays IN ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) THEN
      UPDATE stockes
      SET ref_produit   = :new.ref_produit,
      PAYS              = :NEW.PAYS,
      UNITES_STOCK      = :new.UNITES_STOCK,
      UNITES_COMMANDEES = :new.UNITES_COMMANDEES,
      INDISPONIBLE      = :new.INDISPONIBLE
      WHERE ref_produit   = :old.ref_produit
      AND pays          = :old.PAYS;
  ELSIF (:NEW.pays IN('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine', 'Dominique',
'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) THEN
      UPDATE hcburca.stock_am@dbLinkUS
SET ref_produit   = :new.ref_produit,
      PAYS              = :NEW.PAYS,
      UNITES_STOCK      = :new.UNITES_STOCK,
      UNITES_COMMANDEES = :new.UNITES_COMMANDEES,
      INDISPONIBLE      = :new.INDISPONIBLE
      WHERE ref_produit   = :old.ref_produit
      AND pays          = :old.PAYS;
  ELSE
      UPDATE  cpottiez.STOCKOI@dblinkMain
      SET ref_produit   = :new.ref_produit,
      PAYS              = :NEW.PAYS,
      UNITES_STOCK      = :new.UNITES_STOCK,
      UNITES_COMMANDEES = :new.UNITES_COMMANDEES,
      INDISPONIBLE      = :new.INDISPONIBLE
      WHERE ref_produit   = :old.ref_produit
      AND pays          = :old.PAYS;
  END IF;
END IF;

IF DELETING THEN
  IF (:old.pays IN ('Suede', 'Norvege', 'Danemark', 'Finlande', 'Belgique',
'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande', 'Luxembourg',
'Pays-Bas')) THEN
      DELETE FROM cpottiez.STOCKEN@dblinkMain WHERE ref_produit =
:old.ref_produit AND pays = :old.pays;
```

```
  ELSIF (:old.pays IN ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) THEN
      DELETE
      FROM stockes
      WHERE ref_produit = :old.ref_produit AND pays= :old.pays;
  ELSIF (:old.pays IN('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine', 'Dominique',
'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) THEN
      DELETE
      FROM hcburca.stock_am@dbLinkUS
      WHERE ref_produit = :old.ref_produit AND pays= :old.pays;
  ELSE
      DELETE FROM  cpottiez.STOCKOI@dblinkMain WHERE ref_produit =
:old.ref_produit AND pays = :old.pays;
  END IF;
END IF;
END;
```

Pour la vue Commandes :

Ce trigger est chargé de rediriger les suppressions, modifications et insertions de tuples depuis la vue Commandes vers la base de données correspondante en fonction de l'appartenance à un continent du pays du client ayant passé la commande.

```
Create Or Replace TRIGGER MODIFY_COMMANDES
INSTEAD OF DELETE OR INSERT OR UPDATE ON COMMANDES
FOR EACH ROW
DECLARE
  paysclient$ varchar2(15);
BEGIN
  if (inserting or updating) then
    SELECT DISTINCT Pays INTO paysclient$
    FROM CLIENTS
    WHERE code_client = :new.code_client;
  elsif (deleting) then
    SELECT DISTINCT Pays INTO paysclient$
    FROM CLIENTS
    WHERE code_client = :old.code_client;
  end if;

  if (inserting) then
          if (paysclient$ in ('Antigua-et-Barbuda', 'Argentine',
      'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada',
      'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine',
      'Dominique','Equateur', 'Etats-Unis', 'Grenade', 'Guatemala',
      'Guyana', 'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua',
      'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-
```

```
      Lucie', 'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname',
      'Trinite-et-Tobago', 'Uruguay', 'Venezuela')) then

            INSERT INTO hcburca.Commandes_AM@dblinkus VALUES
(:new.no_commande, :new.code_client, :new.no_employe, :new.date_commande,
:new.date_envoi, :new.port);

    elsif (paysclient$ in ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) then

      INSERT INTO commandesES VALUES (:new.no_commande, :new.code_client,
:new.no_employe, :new.date_commande, :new.date_envoi, :new.port);

    elsif (paysclient$ in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')) then
      INSERT INTO cpottiez.commandesEN@dblinkmain VALUES (:new.no_commande,
:new.code_client, :new.no_employe, :new.date_commande, :new.date_envoi,
:new.port);
    else
      INSERT INTO cpottiez.commandesOI@dblinkmain VALUES (:new.no_commande,
:new.code_client, :new.no_employe, :new.date_commande, :new.date_envoi,
:new.port);
    end if;

ELSIF (updating) then
    if (paysclient$ in ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine',
'Dominique','Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique','Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie','Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay','Venezuela')) then

      UPDATE hcburca.Commandes_AM@dblinkus SET
      no_commande = :new.no_commande,
      code_client = :new.code_client,
      no_employe = :new.no_employe,
      date_commande = :new.date_commande,
      date_envoi = :new.date_envoi,
      port = :new.port
      WHERE no_commande = :old.no_commande;

      ELSIF (paysclient$ in ('Espagne', 'Portugal', 'Andorre', 'France',
      'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
      'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
      'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) then

      UPDATE commandesES SET
      no_commande = :new.no_commande,
      code_client = :new.code_client,
      no_employe = :new.no_employe,
      date_commande = :new.date_commande,
      date_envoi = :new.date_envoi,
      port = :new.port
```

```
      WHERE no_commande = :old.no_commande;

       ELSIF (paysclient$ in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne',
'Islande','Luxembourg', 'Pays-Bas')) then

      UPDATE cpottiez.commandesEN@dblinkmain SET
      no_commande = :new.no_commande,
      code_client = :new.code_client,
      no_employe = :new.no_employe,
      date_commande = :new.date_commande,
      date_envoi = :new.date_envoi,
      port = :new.port
    WHERE no_commande = :old.no_commande;
    ELSE

      UPDATE cpottiez.commandesOI@dblinkmain SET
no_commande = :new.no_commande,
code_client = :new.code_client,
no_employe = :new.no_employe,
date_commande = :new.date_commande,
date_envoi = :new.date_envoi,
port = :new.port
      WHERE no_commande = :old.no_commande;
    end if;

ELSIF (deleting) then
    if (paysclient$ in ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine',
'Dominique','Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) then

      DELETE FROM hcburca.Commandes_AM@dblinkus WHERE no_commande =
:old.no_commande;

    elsif (paysclient$ in ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican','Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie','Autriche', 'Suisse')) then

      DELETE FROM commandesES WHERE no_commande = :old.no_commande;

    elsif (paysclient$ in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
                           'Luxembourg', 'Pays-Bas')) then
      DELETE FROM cpottiez.commandesEN@dblinkmain WHERE no_commande =
:old.no_commande;

    else
      DELETE FROM cpottiez.commandesOI@dblinkmain WHERE no_commande =
:old.no_commande;
    end if;
  end if;
```

```
EXCEPTION
  WHEN no_data_found then
  raise_application_error(-20011, 'Erreur : le client indiqué est
inconnu.');
END;
```

Pour la vue Details_Commandes :

Ce trigger est chargé de rediriger les suppressions, modifications et insertions de tuples depuis la vue Details_Commandes vers la base de données correspondante en fonction de l'appartenance à un continent du pays du client ayant passé la commande référencée dans le tuple.

```
create or replace TRIGGER modify_DetailsCommandes INSTEAD OF
  UPDATE OR
  INSERT OR
  DELETE ON DETAILS_COMMANDES FOR EACH ROW
  DECLARE
  paysTest VARCHAR2(24);
  BEGIN
  if(updating or inserting) then
    SELECT pays into paystest
    FROM CLIENTS natural join COMMANDES
    where no_commande = :new.no_commande;
    end if;

  IF INSERTING THEN          --Insertion à contrôler

    IF (paysTest   IN ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')) THEN
      INSERT
      INTO cpottiez.DETAILS_COMMANDESEN@dblinkmain VALUES
        (
          :new.no_commande,
          :new.ref_produit,
          :new.prix_unitaire,
          :new.quantite,
          :new.remise
        );
    ELSIF (paysTest IN ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) THEN
      INSERT
      INTO details_commandesES VALUES
        (
          :new.no_commande,
          :new.ref_produit,
          :new.prix_unitaire,
          :new.quantite,
          :new.remise
        );
    ELSIF (paysTest IN('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine', 'Dominique',
'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
```

```
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) THEN
      INSERT
      INTO hcburca.Details_Commandes_AM@dbLinkUS VALUES
        (
          :new.no_commande,
          :new.ref_produit,
          :new.prix_unitaire,
          :new.quantite,
          :new.remise
        );
    ELSE
      INSERT
      INTO cpottiez.details_commandesOI@dblinkmain VALUES
        (
          :new.no_commande,
          :new.ref_produit,
          :new.prix_unitaire,
          :new.quantite,
          :new.remise
        );
    END IF;
  END IF;
  IF UPDATING THEN

    IF(:old.ref_produit= :new.ref_produit AND :old.NO_COMMANDE=
:new.no_commande) then
      IF (paysTest IN ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')) THEN
        UPDATE cpottiez.DETAILS_COMMANDESEN@dblinkmain
        SET ref_produit   = :new.ref_produit,
          NO_COMMANDE     = :new.no_commande,
          PRIX_UNITAIRE   = :new.PRIX_UNITAIRE,
          QUANTITE        = :new.QUANTITE,
          REMISE          = :new.REMISE
        WHERE ref_produit = :old.ref_produit
        AND no_commande   = :old.no_commande;
      ELSIF (paysTest IN ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) THEN
        UPDATE details_commandesES
        SET ref_produit   = :new.ref_produit,
          NO_COMMANDE     = :new.no_commande,
          PRIX_UNITAIRE   = :new.PRIX_UNITAIRE,
          QUANTITE        = :new.QUANTITE,
          REMISE          = :new.REMISE
        WHERE ref_produit = :old.ref_produit
        AND no_commande   = :old.no_commande;
      ELSIF (paysTest IN('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine', 'Dominique',
'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
```

```
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) THEN
        UPDATE hcburca.Details_Commandes_AM@dbLinkUS
        SET ref_produit    = :new.ref_produit,
          NO_COMMANDE      = :new.no_commande,
          PRIX_UNITAIRE    = :new.PRIX_UNITAIRE,
          QUANTITE         = :new.QUANTITE,
          REMISE           = :new.REMISE
        WHERE ref_produit = :old.ref_produit
        AND no_commande    = :old.no_commande;
      ELSE
        UPDATE cpottiez.details_commandesOI@dblinkmain
        SET ref_produit    = :new.ref_produit,
          NO_COMMANDE      = :new.no_commande,
          PRIX_UNITAIRE    = :new.PRIX_UNITAIRE,
          QUANTITE         = :new.QUANTITE,
          REMISE           = :new.REMISE
        WHERE ref_produit = :old.ref_produit
        AND no_commande    = :old.no_commande;
      END IF;
    END IF;
  END IF;
  IF DELETING THEN

--    DBMS_OUTPUT.PUT_LINE(:old.no_commande||' - ' || :old.ref_produit);
    SELECT pays into paysTest
    FROM CLIENTS natural join COMMANDES
    where no_commande = :old.no_commande;

   IF (paysTest  IN ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas')) THEN
      DELETE
      FROM cpottiez.DETAILS_COMMANDESEN@dblinkmain
      WHERE ref_produit = :old.ref_produit
      AND no_commande    = :old.no_commande;

      ELSIF (paysTest IN ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzegovine', 'Croatie', 'Grece', 'Macedoine', 'Montenegro',
'Serbie', 'Slovenie', 'Bulgarie', 'Autriche', 'Suisse')) THEN

      DELETE
      FROM details_commandesES
      WHERE ref_produit = :old.ref_produit
      AND no_commande    = :old.no_commande;
    ELSIF (paysTest IN('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie','Costa Rica', 'Cuba', 'Republique dominicaine', 'Dominique',
'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique', 'Mexique', 'Nicaragua', 'Panama',
'Paraguay', 'Perou', 'Saint-Christophe-et-Nieves', 'Sainte-Lucie', 'Saint-
Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-Tobago',
'Uruguay', 'Venezuela')) THEN
      DELETE
      FROM hcburca.Details_Commandes_AM@dbLinkUS
      WHERE ref_produit = :old.ref_produit
      AND no_commande    = :old.no_commande;
```

```
    ELSE
      DELETE
      FROM cpottiez.details_commandesOI@dblinkmain
      WHERE ref_produit = :old.ref_produit
      AND no_commande   = :old.no_commande;
    END IF;
  END IF;

  exception
    when no_data_found then
      raise_application_error(-20012, 'La commande indiquée est inconnue');
END;
```

Pour la vue Clients :

Ce trigger est chargé de rediriger les suppressions, modifications et insertions de tuples depuis la vue Clients vers la base de données correspondante en fonction de l'appartenance à un continent du pays du client. Notez que on interdit le changement de pays du Clients, l'opération de transfert des données associées à ce client et situées dans d'autres tables étant jugé trop complexe.

```
create or replace TRIGGER modify_Clients INSTEAD OF UPDATE OR INSERT or
delete ON Clients
FOR EACH ROW
BEGIN
IF INSERTING then

IF (:NEW.pays in ('Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican',
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')) THEN

      INSERT INTO Clientses
      VALUES (:new.code_client,:new.societe, :new.adresse, :new.ville,
:new.code_postal, :new.pays, :new.telephone, :new.fax);

   ELSIF (:new.pays in  ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela')) then
         INSERT INTO hcburca.Clients_am@dbLinkUS
        VALUES (:new.code_client,:new.societe, :new.adresse, :new.ville,
:new.code_postal, :new.pays, :new.telephone, :new.fax);

      ELSIF  (:new.pays in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas'))
       then
```

```
         INSERT INTO cpottiez.ClientsEN@dblinkMain
         VALUES (:new.code_client,:new.societe, :new.adresse, :new.ville,
:new.code_postal, :new.pays, :new.telephone, :new.fax);

      else
         INSERT INTO cpottiez.ClientsOI@dblinkMain
          VALUES (:new.code_client,:new.societe, :new.adresse, :new.ville,
:new.code_postal, :new.pays, :new.telephone, :new.fax);
  end if;
      END IF;

  if UPDATING THEN

  if(:New.pays <> :OLD.pays) then
     RAISE_APPLICATION_ERROR(-20010,'Attention, vous n''êtes pas autorisé à
changer le pays de l''entreprise');
    end if;
     IF (:NEW.pays in ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican',
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')) THEN
    update clientses
        set
        code_client = :new.code_client,
        societe = :NEW.societe,
        adresse = :new.adresse,
        ville = :new.ville,
        code_postal = :new.code_postal,
        pays = :new.pays,
        telephone = :new.telephone,
        fax = :new.fax
      WHERE code_client = :old.code_client;
      ELSIF  (:new.pays in ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela')) then
        update hcburca.clients_am@dblinkUS
        set
        code_client = :new.code_client,
        societe = :NEW.societe,
        adresse = :new.adresse,
        ville = :new.ville,
        code_postal = :new.code_postal,
        pays = :new.pays,
        telephone = :new.telephone,
        fax = :new.fax
      WHERE code_client = :old.code_client;
      ELSIF  (:new.pays in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique', 'Irlande',
```

```
      'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande', 'Luxembourg',
'Pays-Bas')) then
       update cpottiez.clientsen@dbLinkMain
        set
        code_client = :new.code_client,
        societe = :NEW.societe,
        adresse = :new.adresse,
        ville = :new.ville,
        code_postal = :new.code_postal,
        pays = :new.pays,
        telephone = :new.telephone,
        fax = :new.fax
      WHERE code_client = :old.code_client;
      else
        update cpottiez.clientsoi@dbLinkMain
        set
        code_client = :new.code_client,
        societe = :NEW.societe,
        adresse = :new.adresse,
        ville = :new.ville,
        code_postal = :new.code_postal,
        pays = :new.pays,
        telephone = :new.telephone,
        fax = :new.fax
      WHERE code_client = :old.code_client;
      end if;
  end if;

  IF DELETING THEN

    IF  (:old.pays in ('Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican',
    'Malte', 'Albanie', 'Bosnie-Herzegovine', 'Croatie', 'Grece',
'Macedoine', 'Montenegro', 'Serbie', 'Slovenie', 'Bulgarie',
    'Autriche', 'Suisse')) THEN
    delete from clientses where code_client = :old.code_client;

    ELSIF (:old.pays in  ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil',
  'Canada', 'Chili', 'Colombie','Costa Rica', 'Cuba', 'Republique
dominicaine', 'Dominique',
  'Equateur', 'Etats-Unis', 'Grenade', 'Guatemala', 'Guyana',
'Haiti','Honduras', 'Jamaique',
  'Mexique', 'Nicaragua', 'Panama', 'Paraguay', 'Perou', 'Saint-Christophe-
et-Nieves', 'Sainte-Lucie',
  'Saint-Vincent-et-les Grenadines', 'Salvador', 'Suriname', 'Trinite-et-
Tobago', 'Uruguay',
  'Venezuela')) then
       delete from hcburca.clients_AM@dbLinkUS where code_client =
:old.code_client;
   ELSIF (:old.pays in ('Suede', 'Norvege', 'Danemark', 'Finlande',
'Belgique',
   'Irlande', 'Pologne', 'Royaume-Uni', 'Allemagne', 'Islande',
'Luxembourg', 'Pays-Bas'))then
        delete from cpottiez.clientsEN@dbLinkMain where code_client =
:old.code_client;
    else
```

```
        delete from cpottiez.clientsOI@dbLinkMain where code_client=
:old.code_client;
    end if;
  END IF;
      end;
```

Ainsi, ces triggers sur les vues Commandes, DétailsCommandes, Stock et Clients garantissent une intégrité lors de l'insertion, modification et suppression des données lorsqu'on effectue des opérations dessus.

h) Nettoyages éventuels

Nous supprimons les différents tuples ajoutés dans la table afin de tester son bon fonctionnement.

i) Tests de vérification du bon fonctionnement

Pour des raisons de lisibilité, nous avons placé l'ensemble de nos tests en Annexe, à la fin de ce document. En effet, ces derniers font cinq pages.