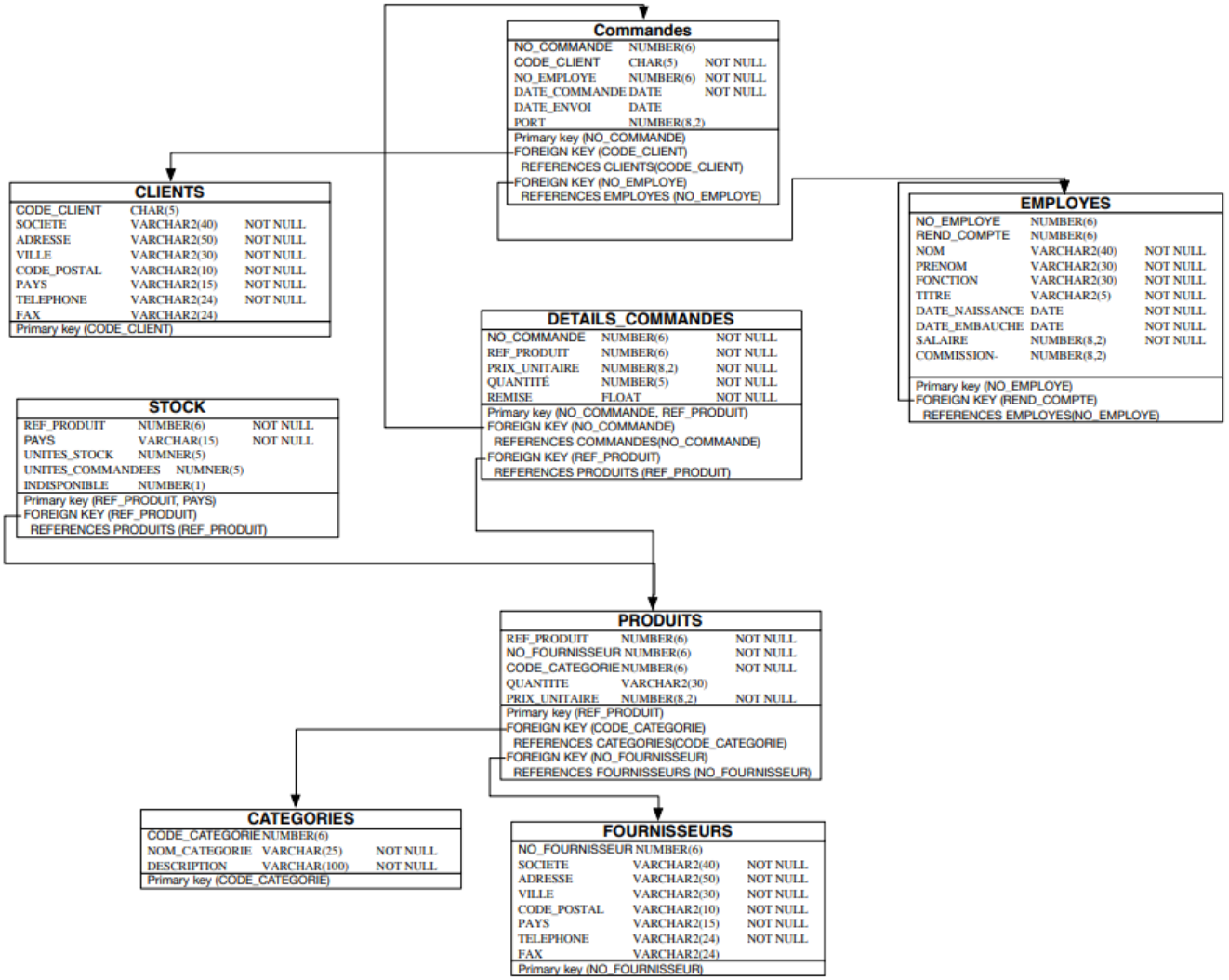
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

TP3

Bases de données distribuées



B3110 : Erabhaoui - Iich - Lemseffer
B3107 : El Karchouni - Bellargui
B3103 : Bachet - Vlad

I. Rappel rapide du contexte et des objectifs du TP.

Le TP vise à transformer la base de données centralisée de l'entreprise "Ryori" en un système distribué sur trois sites (Europe du Nord, Europe du Sud, Amériques), adaptés à leurs applications spécifiques. Les objectifs sont de concevoir la fragmentation des données, les répartir sur les sites sans réplication initiale, implémenter les bases localement avec intégrité et droits d'accès, et mettre en place des réplications pour optimiser le système.

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

A. Groupe de binômes

B3110 : Erabhaoui - Iich - Lemseffer
B3107 : El Karchouni - Bellargui
B3103 : Bachet - Vlad

B. Identification des tâches à réaliser et répartition

- Un binôme responsable par site à gérer.
- Une personne responsable de la coordination générale.
- Une personne responsable de la documentation et du rapport à rendre.

III. Fragmentation

A. Détermination des fragments

1. Fragmentation Horizontale

Etant donné la dimension internationale de cette base de données et les usages locaux, nous allons fragmenter cette table selon des prédicats géographiques. Nos prédicats sont les suivants :

- P1 : Pays ∈ {Europe du Nord}
- P2 : Pays ∈ {Europe du Sud}
- P3 : Pays ∈ {Amérique}

De la sorte, nous identifions les cas suivants :

P1	P2	P3	Fragment
V	F	F	EdN
F	V	F	EdS
F	F	V	A
V	V	F	F (France)
F	F	F	O (Others)

Tableau 1 : Détermination des fragments selon les prédicats discriminants

Ainsi, on a fragmenté horizontalement les tables Commandes, D_Commandes, Clients, et Stock en fonction du pays auquel elles font référence.

La table Fournisseurs n'est pas fragmentée, car bien que cette dernière possède une colonne pays, son seul usage fréquent est en Europe du Nord, comme l'indique l'application **MakeIt**.

Cette fragmentation horizontale nous offre comme avantages un stockage local pour plusieurs données dont on a rarement recours selon le site consulté, ainsi donner des responsabilités selon la situation géographique pour faciliter l'accès aux données et leur modification et aussi pour la particularité de chaque site: Par exemple, Le site américain est responsable des ressources humaines, et l'isolement de ces données facilite sa gestion ou encore Le site d'Europe du Sud qui gère les catégories et les produits via l'application **DesignIt**.

Du coup pour le choix des fragments, elle est de la manière suivante:

Commandes est divisé comme suit :

- Commandes_EdN pour l'Europe du Nord
- Commandes_EdS pour l'Europe du Sud
- Commandes_F pour la France
- Commandes_A pour l'Amérique
- Commandes_O pour les zones n'appartenant pas au tables mentionnés précédemment

Précision: D'après l'énoncé, la France appartient à l'Europe du Nord mais on a fait comme choix de traiter ce cas séparément pour éviter les conflits de l'Europe du Nord et l'Europe du Sud.

On applique une fragmentation horizontale dérivée à Détails Commandes qui est divisée comme ceci:

- D_Commandes EdN pour l'Europe du Nord

● D_Commandes_EdS pour l'Europe du Sud
● D_Commandes_F pour la France
● D_Commandes_A pour l'Amérique
● D_Commandes_O pour les zones n'appartenant pas au tables mentionnés précédemment.

Ensuite, Stock est divisé comme suit :

● Stock_EdN pour l'Europe du Nord
● Stock_EdS pour l'Europe du Sud
● Stock_F pour la France
● Stock_A pour l'Amérique
● Stock_O pour les zones n'appartenant pas au tables mentionnés précédemment.

Finalement, Clients est divisé comme suit :

● Clients_EdN pour l'Europe du Nord
● Clients_EdS pour l'Europe du Sud
● Clients_F pour la France
● Clients_A pour l'Amérique
● Clients_O pour les zones n'appartenant pas au tables mentionnés précédemment

Les autres tables (Catégories, Produits, Fournisseurs et Employés) ne sont pas fragmentées, car elles sont principalement accédées et modifiées depuis un seul et unique site. Il n'est donc pas pertinent de les fragmenter.

2. Fragmentation Verticale

La fragmentation verticale n'a pas été appliquée dans ce contexte car les tables nécessitent une consultation complète de leurs colonnes.

3. Fragmentation mixte

Une fragmentation mixte (horizontale + verticale) n'a pas été nécessaire ici, car les critères horizontaux couvrent tous les besoins de séparation.

4. Bilan de la fragmentation

Après fragmentation, les fragments suivants sont définis pour chaque table :

a. **Commandes** :

Commandes_EdN, Commandes_EdS, Commandes_F, Commandes_A, Commandes_O.

b. **Détails_Commandes** :

Details_Commandes_EdN, Details_Commandes_EdS, Details_Commandes_F, Details_Commandes_A, Details_Commandes_O.

c. **Stock** :

Stock_EdN, Stock_EdS, Stock_F, Stock_A, Stock_O.

d. **Clients** :

Clients_EdN, Clients_EdS, Clients_F, Clients_A, Clients_O.

Les tables **Catégories**, **Produits**, **Fournisseurs**, et **Employés** restent non fragmentées.

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

1. Analyse

Chaque site gère les données locales correspondant à sa région géographique à l'exception de la France qui est traitée séparément, se trouvant en Europe du Nord et du Sud.

Le siège social de l'entreprise se trouvant en Allemagne (Europe du Nord), elle hébergera les données des sites sur lesquels n'ont pas été encore déployé un site de base de données donc ceci concerne les pays qui n'appartiennent pas aux trois sites au moment de son intégration (Autres) . La production est prise en charge par l'Europe du Nord, c'est pour cela qu'on l'a choisi pour accéder et modifier la table Fournisseurs.

Le site Amérique est chargé des ressources humaines de l'entreprise,ainsi que les informations sur les tables Stock, Clients et Commandes. C'est pour cela qu'on va la mettre en place dans le site Amerique.

L'application **DesignIt** permet de modifier les tables concernant les produits fabriqués et disponibles à la vente ainsi que leurs catégories. Cette application n'est disponible que sur le site d'Europe du Sud.Par contre, les données créées par cette application sont accédées depuis tous les sites, mais seule l'Europe du Sud peut les modifier. C'est pour cela que ces données sont déployées en Europe du Sud.

Et comme mentionné dans le sujet, les fragments pour les pays ne faisant partie d'aucune région géographique où est implantée l'entreprise sont pris en charge par le site Europe du Nord (où se trouve le siège social de l'entreprise).

Finalement, les fragments du site français sont pris en charge par le site Europe du Sud. En effet, étant donné que les pays ne faisant partie d'aucune région géographique où est implantée l'entreprise sont pris en charge par le site d'Europe du Nord, nous avons

décidé de placer nos fragments en Europe du Sud où la charge de travail risque d'être plus faible.

2. Bilan

Fragments	Amérique	EdN	EdS
Clients_A	1	0	0
Clients_EdN	0	1	0
Clients_EdS	0	0	1
Clients_F	0	0,5	0,5
Clients_O	0	1	0
Commandes_A	1	0	0
Commandes_EdN	0	1	0
Commandes_EdS	0	0	1
Commandes_F	0	0,5	0,5
Commandes_O	0	1	0
D_Commandes_A	1	0	0
D_Commandes_EdN	0	1	0
D_Commandes_EdS	0	0	1
D_Commandes_F	0	0,5	0,5
D_Commandes_O	0	1	0
Stock_A	1-ε_2-ε_3	ε_2	ε_3
Stock_EdN	ε_1	1-ε_1-ε_3	ε_3
Stock_EdS	ε_1	ε_2	1-ε_1-ε_2
Stock_F	0	0,5	0.5
Stock_O	0	1	0
Produits	ε_1	ε_2	1-ε_1-ε_2
Fournisseurs	ε_1	1-ε_1-ε_3	ε_3
Categories	ε_1	ε_2	1-ε_1-ε_2
Employés	1	0	0

Tableau 2 : Probabilités d'usage des fragments sur les différents sites

- Les valeurs `1` montrent la localisation principale.
- Les valeurs fractionnaires (`0.5`) reflètent une gestion partagée.
- Les ε représentent les probabilités d'usage.

Fragments	Amérique	EdN	EdS
Clients_A	x		
Clients_EdN		x	
Clients_EdS			x
Clients_F			x
Clients_O		x	
Commandes_A	x		
Commandes_EdN		x	
Commandes_EdS			x
Commandes_F			x
Commandes_O		x	
D_Commandes_A	x		
D_Commandes_EdN		x	
D_Commandes_EdS			x
D_Commandes_F			x
D_Commandes_O		x	
Stock_A	x		
Stock_EdN		x	
Stock_EdS			x
Stock_F			x
Stock_O		x	
Produits			x
Fournisseurs		x	
Categories			x
Employés	x		

Tableau 3 : Répartition des fragments sur les différents sites

C. Mise en œuvre de la base sans réplication

1. Site Europe du Nord

a) Binôme Responsable

B3110: ERABHAOUI-LEMSEFFER-IICH

b) Création de liens entre les bases

On crée un lien DB1_Centrale avec la base de données centrale DB1 :

```
CREATE DATABASE LINK DB1_Centrale
CONNECT TO mlemseffer
IDENTIFIED BY MDPORACLE
USING 'DB1';
```

On crée un lien DB3_EdS avec la base de données responsable de l'europe du sud DB3 :

```
CREATE DATABASE LINK DB3_EdS
CONNECT TO mlemseffer
IDENTIFIED BY MDPORACLE
USING 'DB3';
```

On crée un lien DB4_A avec la base de données responsable de l'amérique DB4 :

```
CREATE DATABASE LINK DB4_A
CONNECT TO mlemseffer
IDENTIFIED BY MDPORACLE
USING 'DB4';
```

c) Création et peuplement des tables

Création de la table Clients_EdN en se basant sur le fait que le pays qui doit faire partie de l'europe du nord :

```
CREATE TABLE Clients_EdN AS
Select *
from Ryori.clients@DB1_Centrale
Where PAYS IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne');
```

Création de la table Commandes_EdN via une jointure naturelle entre la table Clients_EdN et la table commande de la base de donnees centralisee, cela permet de récupérer les commandes de l'europe du nord :

```
CREATE TABLE Commandes_EdN AS
Select *
```

```
FROM Ryori.commandes@DB1_Centrale
NATURAL JOIN
(Select CODE_CLIENT
from clients_EdN
Where PAYS IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne'));
```

De même, on a créé la table D_Commandes_EdN des détails des commande via une jointure entre Commandes_EdN et la table détails commandes de la base de donnees centralisee :

```
CREATE TABLE D_Commandes_EdN AS
Select *
FROM Ryori.DETAILS_COMMANDES@DB1_Centrale
Natural JOIN
(Select NO_COMMANDE
from Commandes_EdN);
```

Création de la table Stock_EdN en se basant sur le fait que le pays doit faire partie de l'europe du nord :

```
CREATE TABLE Stock_EdN AS
Select *
from Ryori.stock@DB1_Centrale
Where PAYS IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne');
```

On crée la table FOURNISSEURS en l'important de la base de donnees centralisee :

```
CREATE TABLE FOURNISSEURS AS
Select *
from Ryori.fournisseurs@DB1_Centrale;
```

Clients_O :

```
CREATE TABLE Clients_O AS
Select *
from Ryori.clients@DB1_Centrale
Where PAYS NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
```

```
'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique', 'Équateur', 'Etats-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');
```

Commandes_O :

```
CREATE TABLE Commandes_O AS
Select *
FROM Ryori.commandes@DB1_Centrale
NATURAL JOIN
(Select CODE_CLIENT
from clients_O
Where PAYS NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique', 'Équateur', 'Etats-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'));
```

D_Commandes_O :

```
CREATE TABLE D_Commandes_O AS
Select *
FROM Ryori.DETAILS_COMMANDES@DB1_Centrale
Natural JOIN
(Select NO_COMMANDE
from Commandes_O);
```

Stock_O :

```
CREATE TABLE Stock_O AS
Select *
from Ryori.stock@DB1_Centrale
Where PAYS NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique', 'Équateur', 'Etats-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');
```

d) Contraintes d'intégrités

On a commencé par ajouter les contraintes **clés primaires** :

```
ALTER TABLE CLIENTS_EDN ADD CONSTRAINT pk_Clients_EdN
PRIMARY KEY (Code_Client);

ALTER TABLE CLIENTS_O ADD CONSTRAINT pk_Clients_O
PRIMARY KEY (Code_Client);

ALTER TABLE COMMANDES_EDN ADD CONSTRAINT pk_Commandes_EdN
PRIMARY KEY (NO_COMMANDE);

ALTER TABLE COMMANDES_O ADD CONSTRAINT pk_Commandes_O
PRIMARY KEY (NO_COMMANDE);

ALTER TABLE D_COMMANDES_EDN ADD CONSTRAINT pk_D_Commandes_EdN
PRIMARY KEY (NO_COMMANDE, REF_PRODUIT);

ALTER TABLE D_COMMANDES_O ADD CONSTRAINT pk_D_Commandes_O
PRIMARY KEY (NO_COMMANDE, REF_PRODUIT);

ALTER TABLE FOURNISSEURS ADD CONSTRAINT pk_Fournisseurs
PRIMARY KEY (NO_FOURNISSEUR);
```

```
ALTER TABLE STOCK_EDN ADD CONSTRAINT pk_STOCK_EDN
PRIMARY KEY (REF_PRODUIT, PAYS);

ALTER TABLE STOCK_O ADD CONSTRAINT pk_STOCK_O
PRIMARY KEY (REF_PRODUIT, PAYS);
```

On a ajouté après les contraintes sur les **clés étrangère** :

```
ALTER TABLE COMMANDES_EDN ADD CONSTRAINT fk_Commandes_Clients_EdN
FOREIGN KEY (CODE_CLIENT) REFERENCES Clients_EdN(CODE_CLIENT);

ALTER TABLE COMMANDES_O ADD CONSTRAINT fk_Commandes_Clients_O
FOREIGN KEY (CODE_CLIENT) REFERENCES Clients_O(CODE_CLIENT);
```

On s'est rendu compte que pour la clé étrangère sur les tables COMMANDES_EDN et employes de la base de donnée de l'amérique (DB4_A), il fallait faire un trigger avant l'insertion et la mise à jour des données car les deux tables sont séparées sur deux bases de données différentes, rendant impossible l'utilisation d'une clé étrangère :

```
CREATE OR REPLACE TRIGGER fk_commande_employe_EDN
before INSERT OR UPDATE ON COMMANDES_EDN
for each row
DECLARE
   nbEmp number;
BEGIN
    Select count(*)
    into nbEmp
    FROM ebachet.employes@DB4_A
    where NO_EMPLOYE = :NEW.NO_EMPLOYE;
    IF nbEMP = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_EMPLOYE invalide');
    END IF;
END;
/
```

Il faut faire la même chose entre la table COMMANDES_O et la table employes de la base de donnée de l'amérique (DB4_A) :

```
CREATE OR REPLACE TRIGGER fk_commande_employe_O
before INSERT OR UPDATE ON COMMANDES_O
for each row
DECLARE
```

```
   nbEmp number;
BEGIN
    Select count(*)
    into nbEmp
    FROM ebachet.employes@DB4_A
    where NO_EMPLOYE = :NEW.NO_EMPLOYE;
    IF nbEMP = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_EMPLOYE invalide');
    END IF;
END;
/
```

On ajoute ces deux contraintes de **clés étrangères** :

```
ALTER TABLE D_COMMANDES_EDN ADD CONSTRAINT fk_D_Commandes_Commandes_EdN
FOREIGN KEY (NO_COMMANDE) REFERENCES COMMANDES_EDN(NO_COMMANDE);

ALTER TABLE D_COMMANDES_O ADD CONSTRAINT fk_D_Commandes_Commandes_O
FOREIGN KEY (NO_COMMANDE) REFERENCES COMMANDES_O(NO_COMMANDE);
```

On ajoute encore un Trigger pour la clé étrangère D_Commandes_EDN et la table Produits appartenant à la base de donnée qui gère l'europe du sud (DB3_EdS) :

```
CREATE OR REPLACE TRIGGER fk_D_Commandes_EDN_Produits
before INSERT OR UPDATE ON D_Commandes_EDN
for each row
DECLARE
   nbProd number;
BEGIN
    Select count(*)
    into nbProd
    FROM helkarchou.Produits@DB3_EdS
    where REF_PRODUIT = :NEW.REF_PRODUIT;
    IF nbProd = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_Produit invalide');
    END IF;
END;
/
```

On crée de même un Trigger pour la clé étrangère D_Commandes_O et la table Produits appartenant à la base de donnée qui gère l'europe du sud (DB3_EdS) :

```
CREATE OR REPLACE TRIGGER fk_D_Commandes_O_Produits
before INSERT OR UPDATE ON D_Commandes_O
for each row
DECLARE
   nbProd number;
BEGIN
    Select count(*)
    into nbProd
    FROM helkarchou.Produits@DB3_EdS
    where REF_PRODUIT = :NEW.REF_PRODUIT;
    IF nbProd = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_Produit invalide');
    END IF;
END;
/
```

On crée maintenant un Trigger pour la clé étrangère D_Stock_EdN et la table Produits appartenant à la base de donnée qui gère l'europe du sud (DB3_EdS) :

```
CREATE OR REPLACE TRIGGER fk_D_Stock_EdN_Produits
before INSERT OR UPDATE ON Stock_EdN
for each row
DECLARE
   nbProd number;
BEGIN
    Select count(*)
    into nbProd
    FROM helkarchou.Produits@DB3_EdS
    where REF_PRODUIT = :NEW.REF_PRODUIT;
    IF nbProd = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_Produit invalide');
    END IF;
END;
/
```

On crée de même un Trigger pour la clé étrangère D_Stock_O et la table Produits appartenant à la base de donnée qui gère l'europe du sud (DB3_EdS) :

```
CREATE OR REPLACE TRIGGER fk_D_Stock_O_Produits
before INSERT OR UPDATE ON Stock_O
for each row
DECLARE
   nbProd number;
BEGIN
    Select count(*)
    into nbProd
    FROM helkarchou.Produits@DB3_EdS
    where REF_PRODUIT = :NEW.REF_PRODUIT;
    IF nbProd = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_Produit invalide');
    END IF;
END;
/
```

On ajoute une **contrainte** qui vérifie si le pays est bien un pays de l'europe du nord :

```
ALTER TABLE CLIENTS_EdN ADD CONSTRAINT Pays_EdN
CHECK (PAYS IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne'));
```

On ajoute une contrainte qui vérifie si les autres pays n'appartiennent ni à l'amérique ni à l'europe de sud ni à l'europe :

```
ALTER TABLE CLIENTS_O ADD CONSTRAINT Pays_O
CHECK (PAYS NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine', 'Croatie',
'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie', 'Bulgarie',
'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie',
'Bresil', 'Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba', 'République
dominicaine', 'Dominique', 'Équateur', 'Etats-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'));
```

On implémente un trigger pour **supprimer** correctement des tuples de la table fournisseur :

```
CREATE OR REPLACE TRIGGER supp_FOURNISSEURS
BEFORE DELETE ON FOURNISSEURS
for each row
DECLARE
    nbF number;
BEGIN
    select count(*)
    Into nbF
    from helkarchou.produits@DB3_EDS
    where NO_FOURNISSEUR = :OLD.NO_FOURNISSEUR;
    IF nbF > 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'On ne peut supprimr un fournisseur
qui vend encore des produits');
    END IF;
END;
/
```

On implémente un trigger pour vérifier que le Code_Client contient la bonne lettre pour les sites europe du nord(EDN) et autres(O) pour les tables CLIENTS_EDN, CLIENTS_O, COMMANDES_EDN et COMMANDES_O :

```
CREATE OR REPLACE TRIGGER insertion_code_cle_client_EDN
BEFORE INSERT OR UPDATE ON CLIENTS_EDN
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'N' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"N".');
    END IF;
END;
/

CREATE OR REPLACE TRIGGER insertion_code_cle_client_O
BEFORE INSERT OR UPDATE ON CLIENTS_O
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'O' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"O".');
```

```
    END IF;
END;
/

CREATE OR REPLACE TRIGGER insertion_code_cle_commandes_EDN
BEFORE INSERT OR UPDATE ON COMMANDES_EDN
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'N' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"N".');
    END IF;
END;
/

CREATE OR REPLACE TRIGGER insertion_code_cle_commandes_O
BEFORE INSERT OR UPDATE ON COMMANDES_O
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'O' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"O".');
    END IF;
END;
/
```

e) Droits d'accès

Droits d'accès pour l'utilisateur ebachet :

```
GRANT SELECT ON CLIENTS_EDN TO ebachet;
GRANT SELECT ON CLIENTS_O TO ebachet;
GRANT SELECT ON COMMANDES_EDN TO ebachet;
GRANT SELECT ON COMMANDES_O TO ebachet;
GRANT SELECT ON D_COMMANDES_EDN TO ebachet;
GRANT SELECT ON D_COMMANDES_O TO ebachet;
GRANT SELECT ON FOURNISSEURS TO ebachet;
GRANT SELECT ON STOCK_EDN TO ebachet;
GRANT SELECT ON STOCK_O TO ebachet;
```

Droits d'accès pour l'utilisateur helkarchou :

```
GRANT SELECT ON CLIENTS_EDN TO helkarchou;
GRANT SELECT ON CLIENTS_O TO helkarchou;
GRANT SELECT ON COMMANDES_EDN TO helkarchou;
GRANT SELECT ON COMMANDES_O TO helkarchou;
GRANT SELECT ON D_COMMANDES_EDN TO helkarchou;
GRANT SELECT ON D_COMMANDES_O TO helkarchou;
GRANT SELECT ON FOURNISSEURS TO helkarchou;
GRANT SELECT ON STOCK_EDN TO helkarchou;
GRANT SELECT ON STOCK_O TO helkarchou;
```

f) Définition de synonymes et de vues pour interrogation de la base comme si elle était centralisée

- **Création des synonymes**

```
CREATE OR REPLACE SYNONYM Commandes_EDS for helkarchou.Commandes_EDS@DB3_EdS;
CREATE OR REPLACE SYNONYM Commandes_F for helkarchou.Commandes_F@DB3_EdS;
CREATE OR REPLACE SYNONYM Commandes_A for ebachet.Commandes_A@DB4_A;

CREATE OR REPLACE SYNONYM D_Commandes_EDS for
helkarchou.D_Commandes_EDS@DB3_EdS;
CREATE OR REPLACE SYNONYM D_Commandes_F for helkarchou.D_Commandes_F@DB3_EdS;
CREATE OR REPLACE SYNONYM D_Commandes_A for ebachet.D_Commandes_A@DB4_A;

CREATE OR REPLACE SYNONYM Clients_EDS for helkarchou.Clients_EDS@DB3_EdS;
CREATE OR REPLACE SYNONYM Clients_F for helkarchou.Clients_F@DB3_EdS;
CREATE OR REPLACE SYNONYM Clients_A for ebachet.Clients_A@DB4_A;

CREATE OR REPLACE SYNONYM STOCK_EDS for helkarchou.STOCK_EDS@DB3_EdS;
CREATE OR REPLACE SYNONYM STOCK_F for helkarchou.STOCK_F@DB3_EdS;
CREATE OR REPLACE SYNONYM STOCK_A for ebachet.STOCK_A@DB4_A;

CREATE OR REPLACE SYNONYM EMPLOYES for ebachet.employes@DB4_A;
CREATE OR REPLACE SYNONYM CATEGORIES for helkarchou.categories@DB3_EDS;
CREATE OR REPLACE SYNONYM PRODUITS for helkarchou.Produits@DB3_EDS;
```

- Création des vues :

```
CREATE OR REPLACE VIEW COMMANDES AS
(
    (SELECT Commandes_EDN.*
    from Commandes_EDN, Clients_EDN
```

```
    where Commandes_EDN.Code_Client = Clients_EDN.Code_Client
    AND Clients_EDN.Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas',
'Allemagne', 'Pologne'))
    UNION ALL
    (SELECT Commandes_O.*
    from Commandes_O, Clients_O
    where Commandes_O.Code_Client = Clients_O.Code_Client
    AND Clients_O.Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas',
'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie',
'Slovénie', 'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie',
'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique', 'Équateur',
'Etats-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'))
    UNION ALL
    (SELECT Commandes_EdS.*
    FROM Commandes_EdS, Clients_EdS
    where Commandes_EdS.Code_Client = Clients_EdS.Code_Client
    AND Clients_EDS.Pays IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie'))
    UNION ALL
    (SELECT Commandes_F.*
    FROM Commandes_F, Clients_F
    where Commandes_F.Code_Client = Clients_F.Code_Client
    AND Clients_F.Pays = 'France')
    UNION ALL
    (SELECT Commandes_A.*
    FROM Commandes_A, Clients_A
    where Commandes_A.Code_Client = Clients_A.Code_Client
    AND Clients_A.Pays IN ('Antigua-et-Barbuda', 'Argentine',
'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique',
```

```
'Équateur', 'Etats-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'))
);

CREATE OR REPLACE VIEW D_COMMANDES AS
(
    (SELECT D_Commandes_EDN.*
    from Commandes_EDN, Clients_EDN, D_Commandes_EDN
    where Commandes_EDN.Code_Client = Clients_EDN.Code_Client
    AND Commandes_EDN.no_commande = D_Commandes_EDN.no_commande
    AND Clients_EDN.Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne', 'Pologne'))
    UNION ALL
    (SELECT D_Commandes_O.*
    from Commandes_O, Clients_O, D_Commandes_O
    where Commandes_O.Code_Client = Clients_O.Code_Client
    AND Commandes_O.no_commande = D_Commandes_O.no_commande
    AND Clients_O.Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas',
'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie',
'Slovénie', 'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie',
'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique', 'Équateur',
'Etats-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'))
    UNION ALL
    (SELECT D_Commandes_EdS.*
    FROM Commandes_EdS, Clients_EdS, D_Commandes_EdS
    where Commandes_EdS.Code_Client = Clients_EdS.Code_Client
    AND Commandes_EDS.no_commande = D_Commandes_EDS.no_commande
    AND Clients_EDS.Pays IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
```

```
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie'))
    UNION ALL
    (SELECT D_Commandes_F.*
    FROM Commandes_F, Clients_F, D_Commandes_F
    where Commandes_F.Code_Client = Clients_F.Code_Client
    AND Commandes_F.no_commande = D_Commandes_F.no_commande
    AND Clients_F.Pays = 'France')
    UNION ALL
    (SELECT D_Commandes_A.*
    FROM Commandes_A, Clients_A, D_Commandes_A
    where Commandes_A.Code_Client = Clients_A.Code_Client
    AND Commandes_A.no_commande = D_Commandes_A.no_commande
    AND Clients_A.Pays IN ('Antigua-et-Barbuda', 'Argentine',
'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique',
'Équateur', 'Etats-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'))
    );

CREATE OR REPLACE VIEW STOCK AS
(
    (SELECT *
    FROM STOCK_EDN
    where Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne'))
    UNION ALL
    (SELECT *
    from STOCK_O
    where Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique', 'Équateur', 'Etats-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'))
    UNION ALL
    (SELECT *
    FROM STOCK_EDS
    where Pays IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine', 'Croatie',
'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie', 'Bulgarie'))
    UNION ALL
    (SELECT *
    FROM STOCK_F
    where Pays = 'France')
    UNION ALL
    (SELECT *
    FROM STOCK_A
    where Pays IN ('Antigua-et-Barbuda', 'Argentine',
'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique',
'Équateur', 'Etats-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'))
    );

CREATE OR REPLACE VIEW CLIENTS AS
(
    (SELECT *
    FROM CLIENTS_EDN
    where Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne'))
    UNION ALL
    (SELECT *
    from CLIENTS_O
    where Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne',
'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie', 'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
```

```
'Bolivie', 'Bresil', 'Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique', 'Équateur', 'Etats-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'))
    UNION ALL
    (SELECT *
    FROM CLIENTS_EDS
    where Pays IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine', 'Croatie',
'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',  'Bulgarie'))
    UNION ALL
    (SELECT *
    FROM CLIENTS_F
    where Pays = 'France')
    UNION ALL
    (SELECT *
    FROM CLIENTS_A
    where Pays IN ('Antigua-et-Barbuda', 'Argentine',
'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil', 'Canada', 'Chili',
'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique',
'Équateur', 'Etats-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'))
    );
```

g) Nettoyages éventuels

Sachant que toutes les données de la base Ryori ont été distribuées, on pourra supprimer le lien avec la Base DB1 une fois les tests de vérification du bon fonctionnement terminés :

```
DROP DATABASE LINK DB1_centrale;
```

h) Tests de vérification du bon fonctionnement

- **Vérification des contraintes**

Dans cette partie, nous allons vérifier le bon fonctionnement des contraintes, notamment celles implémentées par un Trigger :

- insertion_code_cle_commandes_O :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO COMMANDES_O
VALUES ('KERNHS', 10795, 1, TO_DATE('1997-12-24', 'YYYY-MM-DD'), TO_DATE('1998-01-20', 'YYYY-MM-DD'), 633.3)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: Le code client doit commencer par "O".
ORA-06512: à "MLEMSEFFER.INSERTION_CODE_CLE_COMMANDES_O", ligne 3
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.INSERTION_CODE_CLE_COMMANDES_O'
```

- insertion_code_cle_commandes_EdN :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO COMMANDES_EdN
VALUES ('KERNHS', 10795, 1, TO_DATE('1997-12-24', 'YYYY-MM-DD'), TO_DATE('1998-01-20', 'YYYY-MM-DD'), 633.3)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: Le code client doit commencer par "N".
ORA-06512: à "MLEMSEFFER.INSERTION_CODE_CLE_COMMANDES_EDN", ligne 3
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.INSERTION_CODE_CLE_COMMANDES_EDN'
```

- insertion_code_cle_client_O :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO Clients_O
VALUES ('KPICCO', 'Piccolo und mehr', 'Geislweg 14', 'Salzburg', '5020', 'Autriche', '6562-9722', '6562-9723')
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: Le code client doit commencer par "O".
ORA-06512: à "MLEMSEFFER.INSERTION_CODE_CLE_CLIENT_O", ligne 3
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.INSERTION_CODE_CLE_CLIENT_O'
```

- insertion_code_cle_client_EdN :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO Clients_EdN
VALUES ('KPICCO', 'Piccolo und mehr', 'Geislweg 14', 'Salzburg', '5020', 'Autriche', '6562-9722', '6562-9723')
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: Le code client doit commencer par "N".
ORA-06512: à "MLEMSEFFER.INSERTION_CODE_CLE_CLIENT_EDN", ligne 3
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.INSERTION_CODE_CLE_CLIENT_EDN'
```

- supp_FOURNISSEURS :

```
Erreur commençant à la ligne: 1 de la commande -
DELETE FROM Fournisseurs
WHERE NO_Fournisseur = 1
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: On ne peut supprimr un fournisseur qui vend encore des produi
ORA-06512: à "MLEMSEFFER.SUPP_FOURNISSEURS", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.SUPP_FOURNISSEURS'
```

- fk_commande_employe_EDN :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO COMMANDES_EDN
VALUES ('NQUICK', 10788, 195, TO_DATE('22/12/1997', 'DD/MM/YYYY'), TO_DATE('19/01/1998', 'DD/MM/YYYY'), 213.5)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: NO_EMPLOYE invalide
ORA-06512: à "MLEMSEFFER.FK_COMMANDE_EMPLOYE_EDN", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.FK_COMMANDE_EMPLOYE_EDN'
```

- fk_commande_employe_O :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO COMMANDES_O
VALUES ('OQUICK', 10788, 195, TO_DATE('22/12/1997', 'DD/MM/YYYY'), TO_DATE('19/01/1998', 'DD/MM/YYYY'), 213.5)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: NO_EMPLOYE invalide
ORA-06512: à "MLEMSEFFER.FK_COMMANDE_EMPLOYE_O", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.FK_COMMANDE_EMPLOYE_O'
```

- fk_D_Commandes_EDN_Produits :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO D_COMMANDES_EdN
VALUES (10623,  140,    116.25, 21, 0)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: NO_Produit invalide
ORA-06512: à "MLEMSEFFER.FK_D_COMMANDES_EDN_PRODUITS", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.FK D COMMANDES EDN PRODUITS'
```

- fk_D_Commandes_O_Produits :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO D_COMMANDES_O
VALUES (10623,  140,    116.25, 21, 0)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: NO_Produit invalide
ORA-06512: à "MLEMSEFFER.FK_D_COMMANDES_O_PRODUITS", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.FK_D_COMMANDES_O_PRODUITS'
```

- fk_D_Stock_EdN_Produits :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO STOCK_EdN
VALUES (2785,'Allemagne',17, 40, 0)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: NO_Produit invalide
ORA-06512: à "MLEMSEFFER.FK_D_STOCK_EDN_PRODUITS", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.FK_D_STOCK_EDN_PRODUITS'
```

- fk_D_Stock_O_Produits :

```
Erreur commençant à la ligne: 1 de la commande -
INSERT INTO STOCK_O
VALUES (2785,'Allemagne',17, 40, 0)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: NO_Produit invalide
ORA-06512: à "MLEMSEFFER.FK_D_STOCK_O_PRODUITS", ligne 10
ORA-04088: erreur lors d'exécution du déclencheur 'MLEMSEFFER.FK_D_STOCK_O_PRODUITS'
```

- **Vérification de la fragmentation**

Dans cette partie, nous allons comparer le nombre de tuples dans les tables de la base Ryori et nos vues. Nous obtenons le même nombre de tuples.

```
//Test de la View COMMANDES : Même nombre de tuples (830)

Select *
From COMMANDES;
SELECT *
FROM Ryori.Commandes@DB1_Centrale;

//Test de la View D_COMMANDES : Même nombre de tuples (2155)

Select *
From D_COMMANDES;
SELECT *
FROM Ryori.DETAILS_Commandes@DB1_Centrale;

//Test de la View STOCK : Même nombre de tuples (231)

Select *
From STOCK;
SELECT *
FROM Ryori.STOCK@DB1_Centrale;

//Test de la View CLIENTS : Même nombre de tuples (91)

Select *
From CLIENTS;
SELECT *
FROM Ryori.CLIENTS@DB1_Centrale;
```

2. Site Europe du Sud

a) Binôme responsable

Le binôme responsable du site Europe du Sud est le B3107, composé de Hamza El Karchouni et Yliess Bellargui.

b) Création de liens entre les bases

```
CREATE DATABASE LINK DB1_Centrale
CONNECT TO helkarchou
IDENTIFIED BY MDPORACLE
USING 'DB1';
```

```
CREATE DATABASE LINK DB2_EdN
CONNECT TO helkarchou
IDENTIFIED BY MDPORACLE
USING 'DB2';

CREATE DATABASE LINK DB4_A
CONNECT TO helkarchou
IDENTIFIED BY MDPORACLE
USING 'DB4';
```

Ces liens permettent de connecter la base Europe du Sud aux autres sites (Europe du Nord, Amérique) et à la base centralisée.

c) Création et peuplement des tables

Les tables suivantes ont été créées :

```
-- Création de la table Clients_EDS
CREATE TABLE Clients_EdS AS
Select *
from Ryori.clients@DB1_Centrale
Where PAYS IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie','Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie');


-- Création de table Produits
CREATE TABLE Produits AS (
    SELECT * FROM Ryori.Produits@DB1_Centrale
);

-- Création de table Catégories
CREATE TABLE Categories AS (
    SELECT * FROM Ryori.Categories@DB1_Centrale
);

-- Création de la table Commandes_EdS
CREATE TABLE Commandes_EdS AS
Select *
FROM Ryori.commandes@DB1_Centrale
NATURAL JOIN
(Select CODE_CLIENT
from clients_EdS);

-- Création de la table Détails des commandes_EdS
```

```
CREATE TABLE D_Commandes_EdS AS
Select *
FROM Ryori.DETAILS_COMMANDES@DB1_Centrale
Natural JOIN
(Select NO_COMMANDE
from Commandes_EdS);

-- Création de la table Stock_EdS
CREATE TABLE Stock_EdS AS
Select *
from Ryori.stock@DB1_Centrale
Where PAYS IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie',  'Bulgarie');

-- Création de la table Clients_F
CREATE TABLE Clients_F AS
Select *
from Ryori.clients@DB1_Centrale
Where PAYS IN ('France');

-- Création de la table Commandes_F
CREATE TABLE Commandes_F AS
Select *
FROM Ryori.commandes@DB1_Centrale
NATURAL JOIN
(Select CODE_CLIENT
from clients_F);

-- Création de la table Détails des commandes_F
CREATE TABLE D_Commandes_F AS
Select *
FROM Ryori.DETAILS_COMMANDES@DB1_Centrale
Natural JOIN
(Select NO_COMMANDE
from Commandes_F);

-- Création de la table Stock_F
CREATE TABLE Stock_F AS
Select *
from Ryori.stock@DB1_Centrale
Where PAYS IN ('France');
```

d) Contraintes d'intégrités

1) *Modification Code-Client, afin qu'ils commencent par un 'S' pour Europe de Sud, et un 'F' pour France :*

```
ALTER TABLE Clients_EDS MODIFY code_client CHAR (7 byte);
UPDATE Clients_EDS SET CODE_CLIENT = CONCAT('S', SUBSTR(code_client, 0
,5));

ALTER TABLE Clients_F MODIFY code_client CHAR (7 byte);
UPDATE Clients_F SET CODE_CLIENT = CONCAT('F', SUBSTR(code_client, 0 ,5));

ALTER TABLE Commandes_EDS MODIFY code_client CHAR (7 byte);
UPDATE Commandes_EDS SET CODE_CLIENT = CONCAT('S', SUBSTR(code_client, 0
,5));

ALTER TABLE Commandes_F MODIFY code_client CHAR (7 byte);
UPDATE Commandes_F
SET CODE_CLIENT = 'F' || SUBSTR(CODE_CLIENT, 1, 5);
```

2) *Clés primaires :*

```
ALTER TABLE Categories
ADD PRIMARY KEY ( Code_categorie );

ALTER TABLE Produits
ADD PRIMARY KEY ( Ref_produit );

ALTER TABLE Clients_EdS ADD CONSTRAINT pk_Clients_EDS
PRIMARY KEY ( Code_client );

ALTER TABLE Commandes_EdS
ADD PRIMARY KEY ( no_commande );

ALTER TABLE D_Commandes_EdS
ADD PRIMARY KEY (No_commande , Ref_produit );

ALTER TABLE Stock_EdS
ADD PRIMARY KEY (Pays , Ref_produit );

ALTER TABLE Clients_F ADD CONSTRAINT PK_Clients_F
PRIMARY KEY ( Code_client );

ALTER TABLE Commandes_F ADD CONSTRAINT PK_Commandes_F
PRIMARY KEY ( no_commande );

ALTER TABLE D_Commandes_F
ADD PRIMARY KEY (No_commande , Ref_produit );
```

```
ALTER TABLE Stock_F
ADD PRIMARY KEY (Pays , Ref_produit );
```

3) Clés étrangères :

```
ALTER TABLE Commandes_EdS
ADD CONSTRAINT FK_Commandes_Clients_EdS
FOREIGN KEY (CODE_CLIENT)
REFERENCES Clients_EdS (CODE_CLIENT);

ALTER TABLE D_Commandes_EdS
ADD CONSTRAINT FK_D_Commandes_Commandes_EdS
FOREIGN KEY (NO_COMMANDE)
REFERENCES Commandes_EdS (NO_COMMANDE);

ALTER TABLE D_Commandes_EdS
ADD CONSTRAINT FK_D_Commandes_Produits
FOREIGN KEY (REF_PRODUIT)
REFERENCES Produits (REF_PRODUIT);

ALTER TABLE Stock_EdS
ADD CONSTRAINT FK_Stock_Produits_EdS
FOREIGN KEY (REF_PRODUIT)
REFERENCES Produits (REF_PRODUIT);

ALTER TABLE Commandes_F
ADD CONSTRAINT FK_Commandes_Clients_F
FOREIGN KEY (CODE_CLIENT)
REFERENCES Clients_F (CODE_CLIENT);


ALTER TABLE D_Commandes_F
ADD CONSTRAINT FK_D_Commandes_Commandes_F
FOREIGN KEY (NO_COMMANDE)
REFERENCES Commandes_F (NO_COMMANDE);


ALTER TABLE D_Commandes_F
ADD CONSTRAINT FK_D_Commandes_Produits_F
FOREIGN KEY (REF_PRODUIT)
REFERENCES Produits (REF_PRODUIT);

ALTER TABLE Stock_F
ADD CONSTRAINT FK_Stock_Produits_F
FOREIGN KEY (REF_PRODUIT)
REFERENCES Produits (REF_PRODUIT);
```

```
ALTER TABLE Produits
ADD CONSTRAINT FK_Produits_Categories
FOREIGN KEY (CODE_CATEGORIE)
REFERENCES Categories (CODE_CATEGORIE);
```

4) Contraintes de validation :

```
ALTER TABLE CLIENTS_EdS ADD CONSTRAINT Pays_EdS
CHECK (PAYS IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie'));

ALTER TABLE CLIENTS_F ADD CONSTRAINT Pays_F
CHECK (PAYS ='France');
```

5) Triggers pour contraintes distantes :

```
-- Ce trigger garantit que tout employé référencé dans une commande existe
dans la base distante avant l'insertion ou la mise à jour dans
Commandes_EdS.

CREATE OR REPLACE TRIGGER fk_commande_employe_EDS
before INSERT OR UPDATE ON COMMANDES_EDS
for each row
DECLARE
   nbEmp number;
BEGIN
    Select count(*)
    into nbEmp
    FROM ebachet.employes@DB4_A
    where NO_EMPLOYE = :NEW.NO_EMPLOYE;
    IF nbEMP = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_EMPLOYE invalide');
    END IF;
END;
/

-- Ce trigger garantit que tout employé référencé dans une commande existe
dans la base distante avant l'insertion ou la mise à jour dans
Commandes_F.

CREATE OR REPLACE TRIGGER fk_commande_employe_F
before INSERT OR UPDATE ON COMMANDES_F
for each row
```

```
DECLARE
   nbEmp number;
BEGIN
    Select count(*)
    into nbEmp
    FROM ebachet.employes@DB4_A
    where NO_EMPLOYE = :NEW.NO_EMPLOYE;
    IF nbEMP = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_EMPLOYE invalide');
    END IF;
END;
/
-- Ce trigger valide que chaque produit est associé à un fournisseur
valide avant toute opération d'insertion ou de mise à jour.

CREATE OR REPLACE TRIGGER fk_produits_fournisseurs
BEFORE INSERT OR UPDATE ON Produits
FOR EACH ROW
DECLARE
   nbFournisseurs NUMBER;
BEGIN
   SELECT COUNT(*)
   INTO nbFournisseurs
   FROM mlemseffer.fournisseurs@DB2_EdN
   WHERE NO_FOURNISSEUR = :NEW.NO_FOURNISSEUR;

   IF nbFournisseurs = 0 THEN
      RAISE_APPLICATION_ERROR(-20001, 'NO_FOURNISSEUR invalide.');
   END IF;
END;
/

-- Ce trigger empêche la suppression d'un produit dans Produits s'il est
référencé dans au moins une commande des tables distantes D_Commandes_A,
D_Commandes_EDN, ou D_Commandes_O.

CREATE OR REPLACE TRIGGER Produits_Commandes_Dist
BEFORE DELETE ON Produits
FOR EACH ROW
DECLARE
      nbA number;
      nbEDN number;
      nbO number;
      nbTot number;
BEGIN
      SELECT COUNT(*)
```

```
        INTO nbA
        FROM ebachet.D_Commandes_A@DB4_A
        where ref_produit = :OLD.ref_produit;

        select count(*)
        into nbEDN
        from mlemseffer.D_Commandes_EDN@DB2_EDN
        where ref_produit = :OLD.ref_produit ;

        select count(*)
        into nbO
        from mlemseffer.D_Commandes_O@DB2_EDN
        where ref_produit = :OLD.ref_produit ;

nbTot := nbA + nbEDN+nbO;
If nbTot > 0 then
        RAISE_APPLICATION_ERROR ( -20001 , 'Commande using this product');
END IF;
END;
/
-- Ce trigger empêche la suppression d'un produit dans Produits s'il est
référencé dans au moins une commande des tables distantes Stock_A,
Stock_EDN, ou Stock_O.

CREATE OR REPLACE TRIGGER Produits_Stock_Dist
BEFORE DELETE ON Produits
FOR EACH ROW
DECLARE
    nbA number;
    nbEDN number;
    nbO number;
    nbTot number;
BEGIN
    SELECT COUNT(*)
    INTO nbA
    FROM ebachet.Stock_A@DB4_A
    where ref_produit = :OLD.ref_produit;

    select count(*)
    into nbEDN
    from mlemseffer.Stock_EDN@DB2_EDN
    where ref_produit = :OLD.ref_produit ;

    select count(*)
    into nbO
    from mlemseffer.Stock_O@DB2_EDN
    where ref_produit = :OLD.ref_produit ;
```

```
nbTot := nbA+nbEDN+nbO;
If nbTot > 0 then
      RAISE_APPLICATION_ERROR ( -20001 , 'Commande using this product');
END IF;
END;
/
```

- Nous avons aussi décidé de mettre en place des triggers qui vérifient avant l'insertion ou la modification que les code_clients commencent bien avec l'indice de la zone (S pour Europe de Sud et F pour France).

```
---- Pour Clients_F et Clients_EDS
CREATE OR REPLACE TRIGGER insertion_code_cle_client_F
BEFORE INSERT OR UPDATE ON CLIENTS_F
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'F' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"N".');
    END IF;
END;
/

CREATE OR REPLACE TRIGGER insertion_code_cle_client_EDS
BEFORE INSERT OR UPDATE ON CLIENTS_EDS
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'S' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"N".');
    END IF;
END;
/

---- Pour Commandes_F et Commandes_EDS
CREATE OR REPLACE TRIGGER insertion_code_cle_commandes_F
BEFORE INSERT OR UPDATE ON COMMANDES_F
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'F' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"N".');
    END IF;
END;
/

CREATE OR REPLACE TRIGGER insertion_code_cle_commandes_EDS
```

```
BEFORE INSERT OR UPDATE ON COMMANDES_EDS
for each row
BEGIN
    IF SUBSTR(:NEW.code_client, 1, 1) != 'S' THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le code client doit commencer par
"N".');
    END IF;
END;
/
```

e) Droits d'accès

Les droits suivants ont été attribués pour ebachet (DB4) et mlemseffer (DB2) :

```
GRANT SELECT ON Produits TO ebachet;
GRANT SELECT ON Categories TO ebachet;
GRANT SELECT ON Stock_F TO ebachet;
GRANT SELECT ON D_Commandes_F TO ebachet;
GRANT SELECT ON Commandes_F TO ebachet;
GRANT SELECT ON Clients_F TO ebachet;
GRANT SELECT ON Stock_EdS TO ebachet;
GRANT SELECT ON D_Commandes_EdS TO ebachet;
GRANT SELECT ON Commandes_EdS TO ebachet;
GRANT SELECT ON Clients_EdS TO ebachet;

GRANT SELECT ON Produits TO mlemseffer;
GRANT SELECT ON Categories TO mlemseffer;
GRANT SELECT ON Stock_F TO mlemseffer;
GRANT SELECT ON D_Commandes_F TO mlemseffer;
GRANT SELECT ON Commandes_F TO mlemseffer;
GRANT SELECT ON Clients_F TO mlemseffer;
GRANT SELECT ON Stock_EdS TO mlemseffer;
GRANT SELECT ON D_Commandes_EdS TO mlemseffer;
GRANT SELECT ON Commandes_EdS TO mlemseffer;
GRANT SELECT ON Clients_EdS TO mlemseffer;
```

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

1) *Synonymes créés pour accéder à des tables externes :*

```
CREATE OR REPLACE SYNONYM Commandes_EDN for
mlemseffer.Commandes_EDN@DB2_EdN;
CREATE OR REPLACE SYNONYM Commandes_O for mlemseffer.Commandes_O@DB2_EdN;
CREATE OR REPLACE SYNONYM Commandes_A for ebachet.Commandes_A@DB4_A;
```

```
CREATE OR REPLACE SYNONYM D_Commandes_EDN for
mlemseffer.D_Commandes_EDN@DB2_EdN;
CREATE OR REPLACE SYNONYM D_Commandes_O for
mlemseffer.D_Commandes_O@DB2_EdN;
CREATE OR REPLACE SYNONYM D_Commandes_A for ebachet.D_Commandes_A@DB4_A;

CREATE OR REPLACE SYNONYM Clients_EDN for mlemseffer.Clients_EDN@DB2_EdN;
CREATE OR REPLACE SYNONYM Clients_O for mlemseffer.Clients_O@DB2_EdN;
CREATE OR REPLACE SYNONYM Clients_A for ebachet.Clients_A@DB4_A;

CREATE OR REPLACE SYNONYM STOCK_EDN for mlemseffer.STOCK_EDN@DB2_EdN;
CREATE OR REPLACE SYNONYM STOCK_O for mlemseffer.STOCK_O@DB2_EdN;
CREATE OR REPLACE SYNONYM STOCK_A for ebachet.STOCK_A@DB4_A;

CREATE OR REPLACE SYNONYM EMPLOYES FOR ebachet.EMPLOYES@DB4_A;
CREATE OR REPLACE SYNONYM FOURNISSEURS FOR
mlemseffer.FOURNISSEURS@DB2_EDN;
```

2) *Vues pour regrouper les tables fragmentées et optimiser les recherches par région :*

```
CREATE OR REPLACE VIEW Stock AS(
      SELECT *
      FROM Stock_EDS
      WHERE Pays IN (
      'Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
      'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
      'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
      'Bulgarie'
))
UNION ALL
(     SELECT *
      FROM Stock_F
      WHERE Pays = 'France')
UNION ALL
(     SELECT *
      FROM Stock_EDN
      WHERE Pays IN (
    'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Pologne', 'Allemagne'
))
UNION ALL
(     SELECT *
      FROM Stock_O
      WHERE Pays NOT IN (
      'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas',
```

```
'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie', 'Bulgarie','Antigua-et-Barbuda', 'Argentine',
'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada', 'Chili',
'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine',
'Dominique','Équateur', 'Etats-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'
))
UNION ALL
(     SELECT *
      FROM Stock_A
      WHERE Pays IN (
      'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique','Équateur', 'Etats-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'
));


CREATE OR REPLACE VIEW Commandes AS(
      SELECT Commandes_EdS.*
      FROM Commandes_EDS, Clients_EDS
      where Commandes_EdS.Code_Client = Clients_EdS.Code_Client
      AND Clients_EDS.Pays IN (
    'Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie'
))
UNION ALL
(     SELECT Commandes_F.*
      FROM Commandes_F, Clients_F
      where Commandes_F.Code_Client = Clients_F.Code_Client
      AND Clients_F.Pays ='France')
UNION ALL
(     SELECT Commandes_EDN.*
      FROM Commandes_EDN, Clients_EDN
      where Commandes_EDN.Code_Client = Clients_EDN.Code_Client
      AND Clients_EDN.Pays IN (
```

```
    'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Pologne', 'Allemagne'
))
UNION ALL
(     SELECT Commandes_O.*
      FROM Commandes_O, Clients_O
      where Commandes_O.Code_Client = Clients_O.Code_Client
      AND Clients_O.Pays NOT IN (
    'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne', 'Pologne',
'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie','Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade',
'Belize', 'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie', 'Costa Rica',
'Cuba', 'République dominicaine', 'Dominique','Équateur', 'Etats-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'
))
UNION ALL
(     SELECT Commandes_A.*
      FROM Commandes_A, Clients_A
      where Commandes_A.Code_Client = Clients_A.Code_Client
      AND Clients_A.Pays IN (
    'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique','Équateur', 'Etats-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'
));


CREATE OR REPLACE VIEW D_COMMANDES AS
(
    (SELECT D_Commandes_EDN.*
    from Commandes_EDN, Clients_EDN, D_Commandes_EDN
    where Commandes_EDN.Code_Client = Clients_EDN.Code_Client
    AND Commandes_EDN.no_commande = D_Commandes_EDN.no_commande
    AND Clients_EDN.Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg',
'Pays-Bas', 'Pologne', 'Allemagne'
))
```

```
UNION ALL
    (SELECT D_Commandes_O.*
    from Commandes_O, Clients_O, D_Commandes_O
    where Commandes_O.Code_Client = Clients_O.Code_Client
    AND Commandes_O.no_commande = D_Commandes_O.no_commande
    AND Clients_O.Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg',
'Pays-Bas', 'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre',
'France', 'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte',
'Albanie', 'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine',
'Monténégro', 'Serbie', 'Slovénie', 'Bulgarie','Antigua-et-Barbuda',
'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada',
'Chili', 'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine',
'Dominique','Équateur', 'Etats-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'))
 UNION ALL
    (SELECT D_Commandes_EdS.*
    FROM Commandes_EdS, Clients_EdS, D_Commandes_EdS
    where Commandes_EdS.Code_Client = Clients_EdS.Code_Client
    AND Commandes_EDS.no_commande = D_Commandes_EDS.no_commande
    AND Clients_EDS.Pays IN ('Espagne', 'Portugal', 'Andorre',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie', 'Bulgarie'))
UNION ALL
    (SELECT D_Commandes_F.*
    FROM Commandes_F, Clients_F, D_Commandes_F
    where Commandes_F.Code_Client = Clients_F.Code_Client
    AND Commandes_F.no_commande = D_Commandes_F.no_commande
    AND Clients_F.Pays = 'France')
UNION ALL
    (SELECT D_Commandes_A.*
    FROM Commandes_A, Clients_A, D_Commandes_A
    where Commandes_A.Code_Client = Clients_A.Code_Client
    AND Commandes_A.no_commande = D_Commandes_A.no_commande
    AND Clients_A.Pays IN ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie',
'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique','Équateur',
'Etats-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'
```

```
))
    );


CREATE OR REPLACE VIEW CLIENTS AS
(
    (SELECT *
    FROM CLIENTS_EDN
    where Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Pologne',
'Allemagne'))
UNION ALL
    (SELECT *
    from CLIENTS_O
    where Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg',
'Pays-Bas', 'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre',
'France', 'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte',
'Albanie', 'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine',
'Monténégro', 'Serbie', 'Slovénie', 'Bulgarie','Antigua-et-Barbuda',
'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada',
'Chili', 'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine',
'Dominique','Équateur', 'Etats-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'))
UNION ALL
    (SELECT *
    FROM CLIENTS_EDS
    where Pays IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie', 'Bulgarie'))
 UNION ALL
    (SELECT *
    FROM CLIENTS_F
    where Pays = 'France')
UNION ALL
    (SELECT *
    FROM CLIENTS_A
    where Pays IN ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie',
'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique','Équateur',
'Etats-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'
))
    );
```

g) Nettoyages éventuels

Aucun nettoyage spécifique n'a été effectué. Les tables temporaires ont été supprimées automatiquement après le traitement.

h) Tests de vérification du bon fonctionnement

On vérifie dans cette partie la fragmentation en comparant le nombre de tuples dans les tables de la base Ryori et les vues que nous avons créé :

```
Select * From D_Commandes;
```

- 2155 tuples sur 2155 initialement.

```
Select * From Clients;
```

- 91 tuples sur 91 initialement.

```
Select * From Commandes;
```

- 830 tuples sur 830 initialement.

```
Select * From Stock;
```

- 231 tuples sur 231 initialement.

```
Select * From Fournisseurs;
```

- 29 tuples sur 29 initialement.

```
Select * From Employes;
```

- 9 tuples sur 9 initialement.

—> On a donc ni pertes, ni doublons.

On vérifie aussi le bon fonctionnement des contraintes, notamment celles des triggers. Par exemple :

```
INSERT INTO commandes_f
VALUES ('RERFDD', 10000, 1, To_date('2004-11-29','YYYY-MM-DD'),
To_date('2005-11-29','YYYY-MM-DD'),500);
```

```
Erreur commençant à la ligne: 1 de la commande -
insert into commandes_f
values ('RERFDD', 10000, 1, To_date('2004-11-29','YYYY-MM-DD'), To_date('2005-11-29','YYYY-MM-DD'),500)
Erreur à la ligne de commande: 1 Colonne: 13
Rapport d'erreur -
Erreur SQL : ORA-20001: Le code client doit commencer par "N".
ORA-06512: à "HELKARCHOU.INSERTION_CODE_CLE_COMMANDES_F", ligne 3
ORA-04088: erreur lors d'exécution du déclencheur 'HELKARCHOU.INSERTION_CODE_CLE_COMMANDES_F'
```

—> En testant toutes les contraintes de clés étrangères ainsi que celles implémentées via des triggers sur Oracle à l'aide de requêtes, nous constatons qu'elles fonctionnent correctement.

3. Site Amérique

a) Binôme responsable

Le binôme responsable du site Amérique est le B3103, composé d'Andreea-Christiana Vlad et Elise Bachet

b) Création de liens entre les bases

```
CREATE DATABASE link DBL_Centrale
CONNECT TO ebachet
IDENTIFIED BY MDPORACLE
USING 'DB1';

CREATE DATABASE link DBL_EdN
CONNECT TO ebachet
IDENTIFIED BY MDPORACLE
USING 'DB2';

CREATE DATABASE link DBL_EdS
CONNECT TO ebachet
IDENTIFIED BY MDPORACLE
USING 'DB3';
```

Ces liens permettent de connecter la base Amérique aux autres sites (Europe du Nord, Europe du Sud) et à la base centralisée.

c) Création et peuplement des tables

```
-- Création de la table Clients
CREATE TABLE Clients_A AS (select * From Ryori.clients @DBL_Centrale where
PAYS='Antigua-et-Barbuda' OR PAYS='Argentine' OR PAYS='Bahamas' OR
PAYS='Barbade' OR PAYS='Belize' OR PAYS='Bolivie' OR PAYS='Bresil' OR
PAYS='Canada'
```

```
OR PAYS='Chili' OR PAYS='Colombie' OR PAYS='Costa Rica' OR PAYS='Cuba' OR
PAYS='Republique dominicaine' OR PAYS='Dominique' OR PAYS='Equateur' OR
PAYS='Etats-Unis' OR PAYS='Grenade' OR PAYS='Guatemala'
OR PAYS='Guyana' OR PAYS='Haiti' OR PAYS='Honduras' OR PAYS='Jamaique' OR
PAYS='Mexique' OR PAYS='Nicaragua' OR PAYS='Panama' OR PAYS='Paraguay' OR
PAYS='Perou' OR PAYS='Saint-Christophe-et-Nieves'
OR PAYS='Sainte-Lucie' OR PAYS='Saint-Vincent-et-les-Grenadines' OR
PAYS='Salvador' OR PAYS='Suriname' OR PAYS='Trinite-et-Tobago' OR
PAYS='Uruguay' OR PAYS='Venezuela') ;

-- Création de la table Commande
CREATE TABLE Commandes_A AS (select * from Ryori.Commandes @DBL_Centrale
natural join clients_a);
ALTER TABLE commandes_A DROP COLUMN SOCIETE;
ALTER TABLE commandes_A DROP COLUMN VILLE;
ALTER TABLE commandes_A DROP COLUMN SOCIETE;
ALTER TABLE commandes_A DROP COLUMN adresse;
ALTER TABLE commandes_A DROP COLUMN code_postal;
ALTER TABLE commandes_A DROP COLUMN pays;
ALTER TABLE commandes_A DROP COLUMN telephone;
ALTER TABLE commandes_A DROP COLUMN fax;

-- Création de la table Détails-Commande
CREATE TABLE D_Commandes_A AS (SELECT * FROM Ryori.Details_commandes
@DBL_Centrale NATURAL JOIN (select NO_Commande from Commandes_a));

-- Création de la table Employes
CREATE TABLE Employes AS (SELECT * FROM Ryori.Employes @DBL_Centrale);
-- Création de la table Stock
CREATE TABLE Stock_A AS (select * From Ryori.Stock @DBL_Centrale where
PAYS='Antigua-et-Barbuda' OR PAYS='Argentine' OR PAYS='Bahamas' OR
PAYS='Barbade' OR PAYS='Belize' OR PAYS='Bolivie' OR PAYS='Bresil' OR
PAYS='Canada'
OR PAYS='Chili' OR PAYS='Colombie' OR PAYS='Costa Rica' OR PAYS='Cuba' OR
PAYS='Republique dominicaine' OR PAYS='Dominique' OR PAYS='Equateur' OR
PAYS='Etats-Unis' OR PAYS='Grenade' OR PAYS='Guatemala'
OR PAYS='Guyana' OR PAYS='Haiti' OR PAYS='Honduras' OR PAYS='Jamaique' OR
PAYS='Mexique' OR PAYS='Nicaragua' OR PAYS='Panama' OR PAYS='Paraguay' OR
PAYS='Perou' OR PAYS='Saint-Christophe-et-Nieves'
OR PAYS='Sainte-Lucie' OR PAYS='Saint-Vincent-et-les-Grenadines' OR
PAYS='Salvador' OR PAYS='Suriname' OR PAYS='Trinite-et-Tobago' OR
PAYS='Uruguay' OR PAYS='Venezuela') ;
```

d) Contraintes d'intégrités

Modification Code-Client, afin qu'ils commencent par un 'A', pour Amérique

```
ALTER TABLE Clients_A
MODIFY code_client CHAR (7 byte);
ALTER TABLE Commandes_A
MODIFY code_client CHAR (7 byte);

UPDATE CLIENTS_A
SET code_client=CONCAT('A', SUBSTR(code_client, 0, 5));
UPDATE Commandes_A
SET code_client=CONCAT('A', SUBSTR(code_client, 0, 5));
ALTER TABLE Clients_A
ADD CHECK (SUBSTR(code_client, 0, 1)='A');
ALTER TABLE Commandes_A
ADD CHECK (SUBSTR(code_client, 0, 1)='A');
```

Clés Primaires

```
ALTER TABLE Clients_A
ADD CONSTRAINT pkCodeClient
PRIMARY KEY (CODE_CLIENT);

ALTER TABLE Commandes_A
ADD CONSTRAINT pkNoCommande
PRIMARY KEY (NO_COMMANDE);

ALTER TABLE D_Commandes_A
ADD CONSTRAINT pkNoCommandeRef
PRIMARY KEY (NO_COMMANDE,REF_PRODUIT);

ALTER TABLE EMPLOYES
ADD CONSTRAINT pkNoEmployes
PRIMARY KEY (NO_EMPLOYE);
```

Clés étrangères

```
ALTER TABLE EMPLOYES
ADD CONSTRAINT fkNoEmployes
FOREIGN KEY(rend_compte)
```

```
REFERENCES EMPLOYES(NO_EMPLOYE);

ALTER TABLE COMMANDES_A
ADD CONSTRAINT fkCodeClient
FOREIGN KEY(CODE_CLIENT)
REFERENCES CLIENTS_A(CODE_CLIENT);

ALTER TABLE COMMANDES_A
ADD CONSTRAINT fkNOEMPLOYE
FOREIGN KEY (NO_EMPLOYE)
REFERENCES EMPLOYES (NO_EMPLOYE);

ALTER TABLE D_COMMANDES_A
ADD CONSTRAINT fkNOCOMMANDE
FOREIGN KEY (NO_COMMANDE)
REFERENCES COMMANDES_A(NO_COMMANDE);
```

Contraintes de validation :

```
ALTER TABLE Clients_A
ADD CONSTRAINT Pays_Valide
CHECK (PAYS='Antigua-et-Barbuda' OR PAYS='Argentine' OR PAYS='Bahamas' OR
PAYS='Barbade' OR PAYS='Belize' OR PAYS='Bolivie' OR PAYS='Bresil' OR
PAYS='Canada'
OR PAYS='Chili' OR PAYS='Colombie' OR PAYS='Costa Rica' OR PAYS='Cuba' OR
PAYS='Republique dominicaine' OR PAYS='Dominique' OR PAYS='Equateur' OR
PAYS='Etats-Unis' OR PAYS='Grenade' OR PAYS='Guatemala'
OR PAYS='Guyana' OR PAYS='Haiti' OR PAYS='Honduras' OR PAYS='Jamaique' OR
PAYS='Mexique' OR PAYS='Nicaragua' OR PAYS='Panama' OR PAYS='Paraguay' OR
PAYS='Perou' OR PAYS='Saint-Christophe-et-Nieves'
OR PAYS='Sainte-Lucie' OR PAYS='Saint-Vincent-et-les-Grenadines' OR
PAYS='Salvador' OR PAYS='Suriname' OR PAYS='Trinite-et-Tobago' OR
PAYS='Uruguay' OR PAYS='Venezuela');


ALTER TABLE D_Commandes_A
ADD CONSTRAINT Quantite_Positive
CHECK (QUANTITE > 0);

ALTER TABLE D_Commandes_A
ADD CONSTRAINT Remise_Existante
CHECK (REMISE IS NOT NULL);
```

```
ALTER TABLE Commandes_A
ADD CONSTRAINT Date_Commande_Valide
CHECK (DATE_COMMANDE IS NOT NULL);

ALTER TABLE Clients_A
ADD CONSTRAINT Telephone_Renseigne
CHECK (TELEPHONE IS NOT NULL);
```

Triggers pour contraintes distantes :

```
-- Trigger vérifiant que le produit de la commande est disponible
CREATE OR REPLACE TRIGGER fk_D_Commandes_A_Produits
before INSERT OR UPDATE ON D_Commandes_A
for each row
DECLARE
   nbProd number;
BEGIN
    Select count(*)
    into nbProd
    FROM helkarchou.Produits@DBL_EdS
    where REF_PRODUIT = :NEW.REF_PRODUIT;
    IF nbProd = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_Produit invalide');
    END IF;
END;

-- Trigger vérifiant que le produit est en stock
CREATE OR REPLACE TRIGGER fk_STOCK_A
before INSERT OR UPDATE ON STOCK_A
for each row
DECLARE
   nbProd number;
BEGIN
    Select count(*)
    into nbProd
    FROM helkarchou.Produits@DBL_EdS
    where REF_PRODUIT = :NEW.REF_PRODUIT;
    IF nbProd = 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'NO_Produit invalide');
```

```
    END IF;
END;

-- Triger pour vérifier si la suppression d'un employé est faisable

CREATE OR REPLACE TRIGGER Commandes_Employes
before DELETE ON EMPLOYES
for each row
DECLARE
   nbComEdS number;
   nbComEdN number;
   nbComO number;
   nbComF number;
   nbTotCom number;
BEGIN
    Select count(*)
    into nbComEdS
    FROM helkarchou.COMMANDES_EDS@DBL_EdS
    where NO_Employe = :OLD.NO_EMPLOYE;

    Select count(*)
    into nbComF
    FROM helkarchou.Commandes_F@DBL_EdS
    where NO_Employe = :OLD.NO_EMPLOYE;

    Select count(*)
    into nbComEdN
    FROM mlemseffer.Commandes_EdN@DBL_EdN
    where NO_Employe = :OLD.NO_EMPLOYE;

    Select count(*)
    into nbComO
    FROM mlemseffer.Commandes_O@DBL_EdN
    where NO_Employe = :OLD.NO_EMPLOYE;

    nbTotCom:=nbComEdS+nbComEdN+nbComO+nbComF;
    IF nbTotCom> 0
    THEN
        RAISE_APPLICATION_ERROR(-20001, 'Employe qui s_occupe d_une
commande');
    END IF;
END;
```

```
/
```

e) Droits d'accès

```
GRANT SELECT Employes TO mlemseffer;
GRANT SELECT ON Clients_A TO mlemseffer;
GRANT SELECT ON Commandes_A TO mlemseffer;
GRANT SELECT ON D_Commandes_A TO mlemseffer;
GRANT SELECT ON Employes TO helkarchou;
GRANT SELECT ON Clients_A TO helkarchou;
GRANT SELECT ON Commandes_A TO helkarchou;
GRANT SELECT ON D_Commandes_A TO helkarchou;
```

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

Synonymes créés pour accéder à des tables externes :

```
CREATE OR REPLACE SYNONYM Commandes_EDN for
mlemseffer.Commandes_EDN@DBL_EdN;
CREATE OR REPLACE SYNONYM Commandes_O for mlemseffer.Commandes_O@DBL_EdN;
CREATE OR REPLACE SYNONYM Commandes_EDS for
helkarchou.Commandes_EDS@DBL_EdS;
CREATE OR REPLACE SYNONYM Commandes_F for helkarchou.Commandes_F@DBL_EdS;

CREATE OR REPLACE SYNONYM D_Commandes_EDN for
mlemseffer.D_Commandes_EDN@DBL_EdN;
CREATE OR REPLACE SYNONYM D_Commandes_O for
mlemseffer.D_Commandes_O@DBL_EdN;
CREATE OR REPLACE SYNONYM D_Commandes_EDS for
helkarchou.D_Commandes_EDS@DBL_EdS;
CREATE OR REPLACE SYNONYM D_Commandes_F for
helkarchou.D_Commandes_F@DBL_EdS;


CREATE OR REPLACE SYNONYM Clients_EDN for mlemseffer.Clients_EDN@DBL_EdN;
CREATE OR REPLACE SYNONYM Clients_O for mlemseffer.Clients_O@DBL_EdN;
CREATE OR REPLACE SYNONYM Clients_EDS for helkarchou.Clients_EDS@DBL_EdS;
CREATE OR REPLACE SYNONYM Clients_F for helkarchou.Clients_F@DBL_EdS;

CREATE OR REPLACE SYNONYM STOCK_EDN for mlemseffer.STOCK_EDN@DB2_EdN;
CREATE OR REPLACE SYNONYM STOCK_O for mlemseffer.STOCK_O@DB2_EdN;
```

```
CREATE OR REPLACE SYNONYM STOCK_EDS for helkarchou.Stock_EDS@DBL_EdS;
CREATE OR REPLACE SYNONYM STOCK_F for helkarchou.Stock_F@DBL_EdS;


CREATE OR REPLACE SYNONYM FOURNISSEURS FOR
mlemseffer.FOURNISSEURS@DB2_EDN;
CREATE OR REPLACE SYNONYM CATEGORIES for helkarchou.categories_F@DBL_EdS;
CREATE OR REPLACE SYNONYM PRODUITS for helkarchou.produits_F@DBL_EdS;
```

Vues pour regrouper les tables fragmentées et optimiser les recherches par région :

```
CREATE OR REPLACE VIEW Stock AS(
      SELECT *
      FROM Stock_EDS
      WHERE (Pays IN (
      'Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
      'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
      'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
      'Bulgarie'
))
UNION ALL
(     SELECT *
      FROM Stock_F
      WHERE Pays = 'France')
UNION ALL
(     SELECT *
      FROM Stock_EDN
      WHERE Pays IN (
    'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Pologne', 'Allemagne'
))
UNION ALL
(     SELECT *
      FROM Stock_O
      WHERE Pays NOT IN (
      'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas',
'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre', 'France',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie', 'Bulgarie','Antigua-et-Barbuda', 'Argentine',
'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada', 'Chili',
'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine',
'Dominique','Équateur', 'Etats-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'
))
UNION ALL
(     SELECT *
      FROM Stock_A
      WHERE Pays IN (
      'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique','Équateur', 'Etats-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'
));


CREATE OR REPLACE VIEW Commandes AS(
      SELECT Commandes_EdS.*
      FROM Commandes_EDS, Clients_EDS
      where Commandes_EdS.Code_Client = Clients_EdS.Code_Client
      AND Clients_EDS.Pays IN (
    'Espagne', 'Portugal', 'Andorre', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie'
))
UNION ALL
(     SELECT Commandes_F.*
      FROM Commandes_F, Clients_F
      where Commandes_F.Code_Client = Clients_F.Code_Client
      AND Clients_F.Pays ='France')
UNION ALL
(     SELECT Commandes_EDN.*
      FROM Commandes_EDN, Clients_EDN
      where Commandes_EDN.Code_Client = Clients_EDN.Code_Client
      AND Clients_EDN.Pays IN (
    'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Pologne', 'Allemagne'
))
UNION ALL
(     SELECT Commandes_O.*
      FROM Commandes_O, Clients_O
      where Commandes_O.Code_Client = Clients_O.Code_Client
      AND Clients_O.Pays NOT IN (
    'Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande', 'Royaume-Uni',
'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Allemagne', 'Pologne',
```

```
'Espagne', 'Portugal', 'Andorre', 'France', 'Gibraltar', 'Italie',
'Saint-Marin', 'Vatican', 'Malte', 'Albanie', 'Bosnie-Herzégovine',
'Croatie', 'Grèce', 'Macédoine', 'Monténégro', 'Serbie', 'Slovénie',
'Bulgarie','Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade',
'Belize', 'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie', 'Costa Rica',
'Cuba', 'République dominicaine', 'Dominique','Équateur', 'Etats-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'
))
UNION ALL
(     SELECT Commandes_A.*
      FROM Commandes_A, Clients_A
      where Commandes_A.Code_Client = Clients_A.Code_Client
      AND Clients_A.Pays IN (
    'Antigua-et-Barbuda', 'Argentine', 'Bahamas', 'Barbade', 'Belize',
'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie', 'Costa Rica', 'Cuba',
'République dominicaine', 'Dominique','Équateur', 'Etats-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'
));


CREATE OR REPLACE VIEW D_COMMANDES AS
(
    (SELECT D_Commandes_EDN.*
    from Commandes_EDN, Clients_EDN, D_Commandes_EDN
    where Commandes_EDN.Code_Client = Clients_EDN.Code_Client
    AND Commandes_EDN.no_commande = D_Commandes_EDN.no_commande
    AND Clients_EDN.Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg',
'Pays-Bas', 'Pologne', 'Allemagne'
))
UNION ALL
    (SELECT D_Commandes_O.*
    from Commandes_O, Clients_O, D_Commandes_O
    where Commandes_O.Code_Client = Clients_O.Code_Client
    AND Commandes_O.no_commande = D_Commandes_O.no_commande
    AND Clients_O.Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg',
'Pays-Bas', 'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre',
'France', 'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte',
'Albanie', 'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine',
```

```
'Monténégro', 'Serbie', 'Slovénie', 'Bulgarie','Antigua-et-Barbuda',
'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada',
'Chili', 'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine',
'Dominique','Équateur', 'Etats-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'))
 UNION ALL
    (SELECT D_Commandes_EdS.*
    FROM Commandes_EdS, Clients_EdS, D_Commandes_EdS
    where Commandes_EdS.Code_Client = Clients_EdS.Code_Client
    AND Commandes_EDS.no_commande = D_Commandes_EDS.no_commande
    AND Clients_EDS.Pays IN ('Espagne', 'Portugal', 'Andorre',
'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie', 'Bulgarie'))
UNION ALL
    (SELECT D_Commandes_F.*
    FROM Commandes_F, Clients_F, D_Commandes_F
    where Commandes_F.Code_Client = Clients_F.Code_Client
    AND Commandes_F.no_commande = D_Commandes_F.no_commande
    AND Clients_F.Pays = 'France')
UNION ALL
    (SELECT D_Commandes_A.*
    FROM Commandes_A, Clients_A, D_Commandes_A
    where Commandes_A.Code_Client = Clients_A.Code_Client
    AND Commandes_A.no_commande = D_Commandes_A.no_commande
    AND Clients_A.Pays IN ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie',
'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique','Équateur',
'Etats-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'
))
    );


CREATE OR REPLACE VIEW CLIENTS AS
(
    (SELECT *
    FROM CLIENTS_EDN
```

```
    where Pays IN ('Norvege', 'Suede', 'Danemark', 'Islande', 'Finlande',
'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg', 'Pays-Bas', 'Pologne',
'Allemagne'))
UNION ALL
    (SELECT *
    from CLIENTS_O
    where Pays NOT IN ('Norvege', 'Suede', 'Danemark', 'Islande',
'Finlande', 'Royaume-Uni', 'Irlande', 'Belgique', 'Luxembourg',
'Pays-Bas', 'Allemagne', 'Pologne', 'Espagne', 'Portugal', 'Andorre',
'France', 'Gibraltar', 'Italie', 'Saint-Marin', 'Vatican', 'Malte',
'Albanie', 'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine',
'Monténégro', 'Serbie', 'Slovénie', 'Bulgarie','Antigua-et-Barbuda',
'Argentine', 'Bahamas', 'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada',
'Chili', 'Colombie', 'Costa Rica', 'Cuba', 'République dominicaine',
'Dominique','Équateur', 'Etats-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'))
UNION ALL
    (SELECT *
    FROM CLIENTS_EDS
    where Pays IN ('Espagne', 'Portugal', 'Andorre', 'Gibraltar',
'Italie', 'Saint-Marin', 'Vatican', 'Malte', 'Albanie',
'Bosnie-Herzégovine', 'Croatie', 'Grèce', 'Macédoine', 'Monténégro',
'Serbie', 'Slovénie', 'Bulgarie'))
 UNION ALL
    (SELECT *
    FROM CLIENTS_F
    where Pays = 'France')
UNION ALL
    (SELECT *
    FROM CLIENTS_A
    where Pays IN ('Antigua-et-Barbuda', 'Argentine', 'Bahamas',
'Barbade', 'Belize', 'Bolivie', 'Bresil','Canada', 'Chili', 'Colombie',
'Costa Rica', 'Cuba', 'République dominicaine', 'Dominique','Équateur',
'Etats-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'
))
    );
```

g) Nettoyages éventuels

Aucun nettoyage spécifique.

h) Tests de vérification du bon fonctionnement

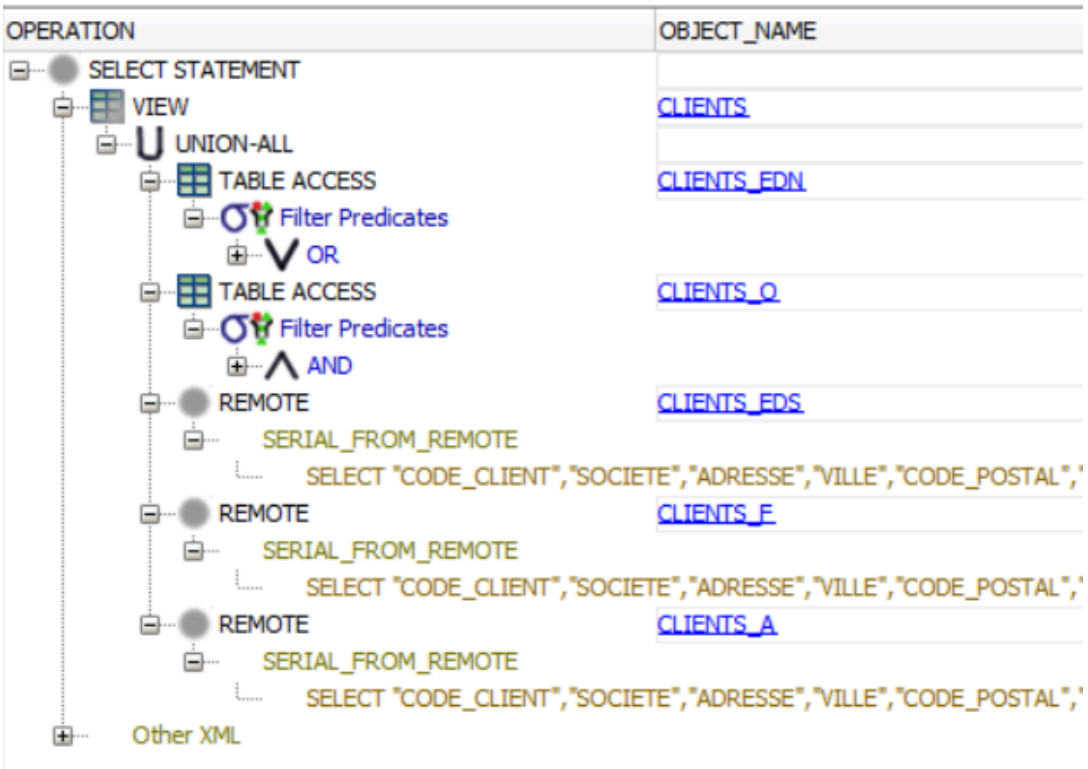
```
Select * From D_Commandes;
Select * From Clients;
Select * From Commandes;
Select * From Stock;
Select * From Fournisseurs;
Select * From Employes;
```

Le nombre de tuples renvoyé est le même que le nombre initial de tuples pour chaque requête.

IV. Tests de requête distribuées et optimisations

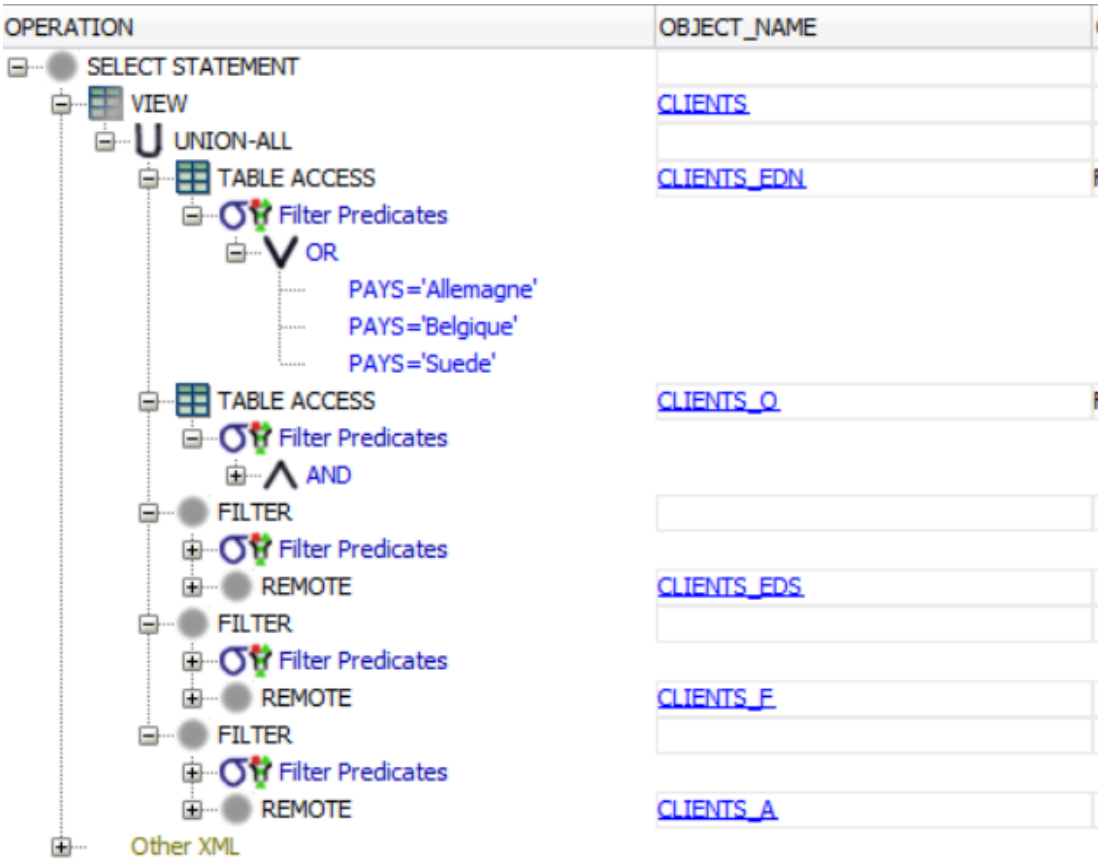
A. Site Europe du Nord

1. **Requête 1 :** `select * from clients;`



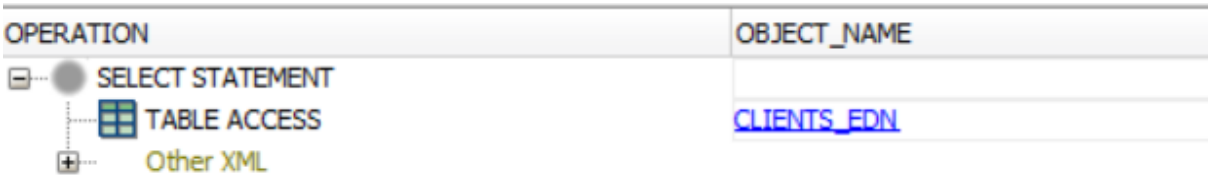
On remarque dans ce plan d'exécution que le SGBD réalise un UNION ALL entre tous les fragments pour obtenir la table clients. Ceci est conforme avec la VIEW que l'on a créé pour la table clients.

2. **Requête 2 :** `select * from clients where pays in ('Allemagne', 'Suede', 'Belgique');`



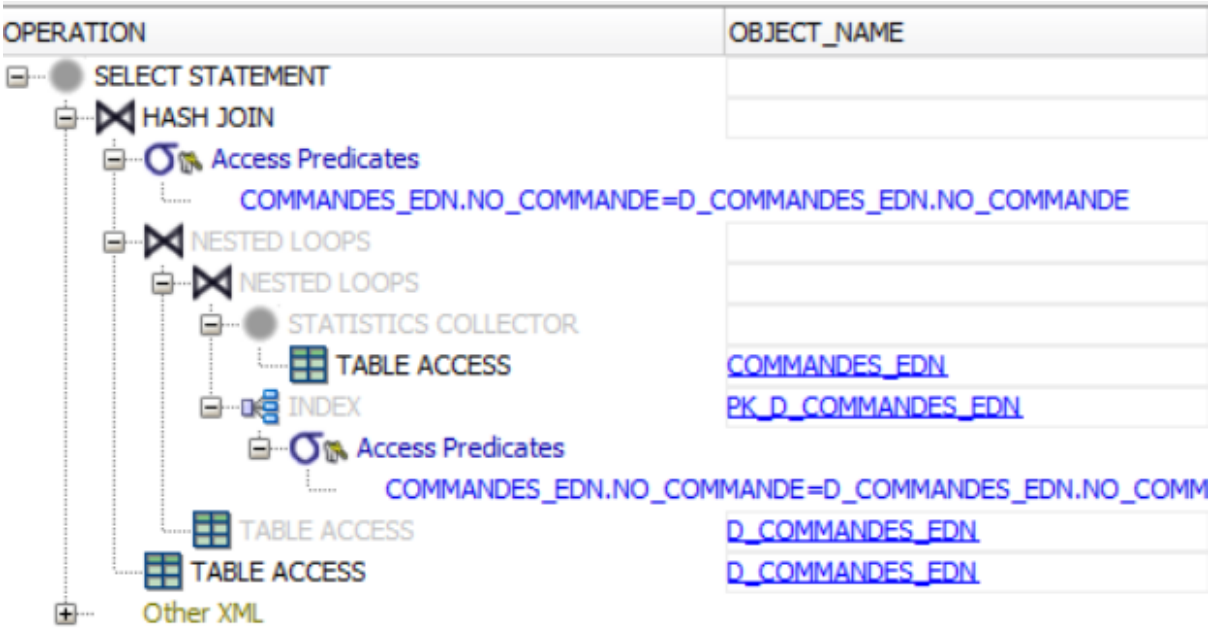
Ce plan d'exécution ressemble beaucoup à celui de la requête précédente, mais au lieu d'avoir tous les pays dans le OR de Client EdN, on a que les pays qui nous intéressent. Oracle ne cherche pas dans les tables qui ne sont pas en Europe du Nord, en utilisant comme filtre de sélection NULL is NOT NULL, ce qui est impossible.

3. **Requête 3 :** `select * from clients_edn;`



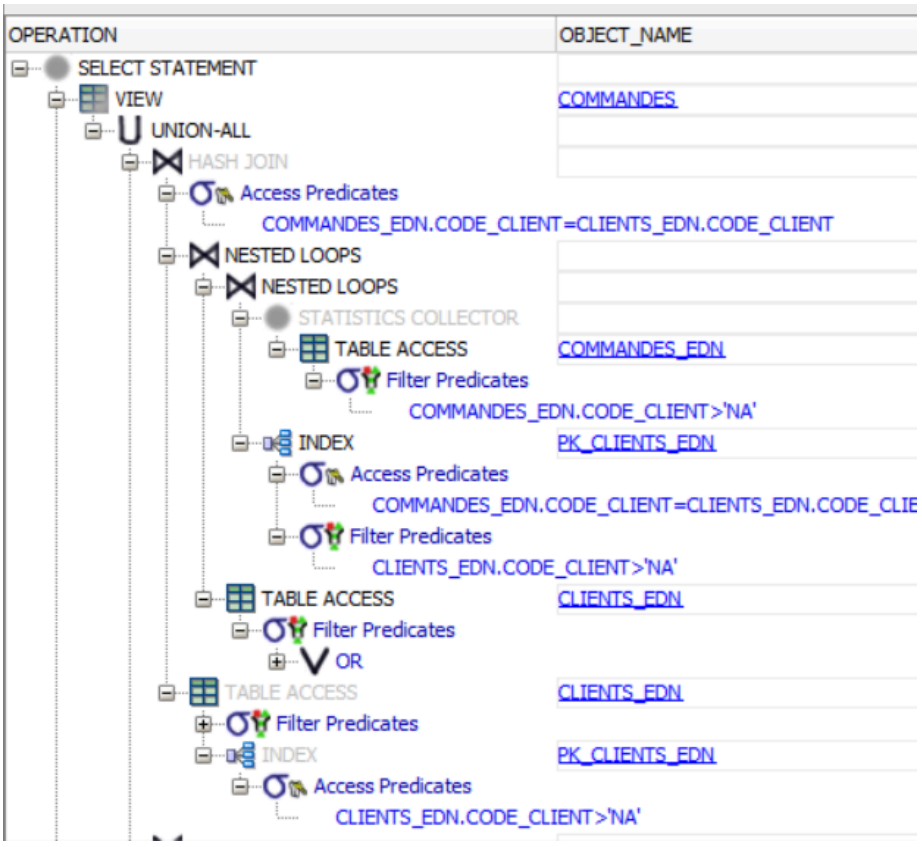
Ce plan d'exécution représente la sélection de tuples dans une table locale, Clients_EdN qui se fait par un simple TABLE ACCESS étant donné que l'on sélectionne tous les tuples.

4. **Requête 4 :** `Select * from Commandes_EDN NATURAL JOIN D_Commandes_EDN;`



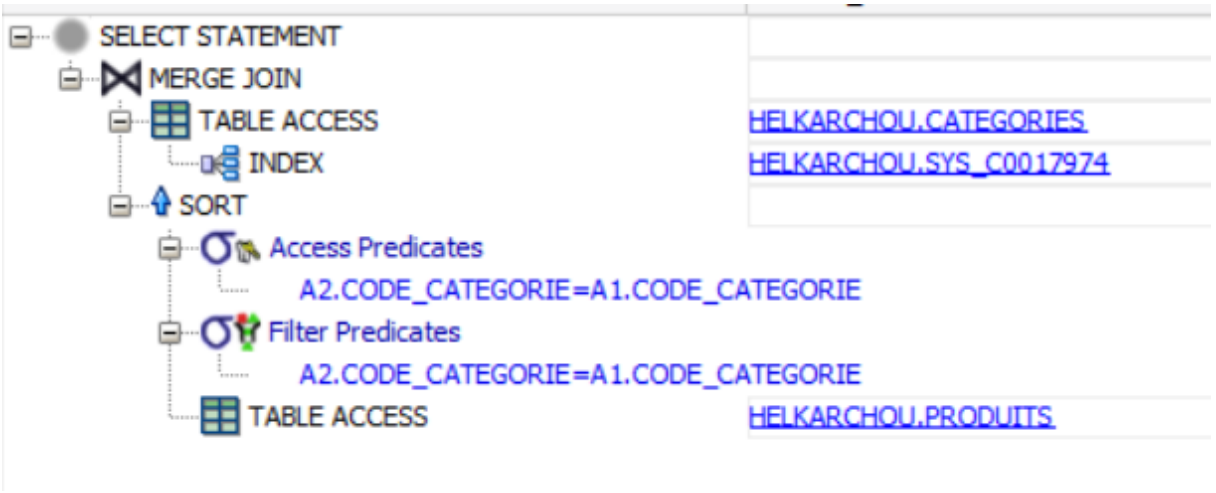
Pour cette requête, nous avons décidé de réaliser une jointure naturelle, afin de bien vérifier que la jointure se fait par l'usage d'un index. Cela est bien le cas, avec l'usage de l'index sur la clé primaire de la table D_Commandes_EdN.

5. **Requête 5 :** `Select * from Commandes WHERE Code_CLIENT > 'NA';`



Dans ce cas, nous avions décidé de faire une sélection sur la vue commandes avec une condition sur la clé, pour vérifier que le SGBD allait bien utiliser un index B_Arbre par défaut (très performant pour les requêtes d'inégalités). Ici, on a bien l'usage d'un index. La requête est bien optimisée par le SGBD.

6. **Requête 6 :** `SELECT * FROM PRODUITS NATURAL JOIN CATEGORIES;`



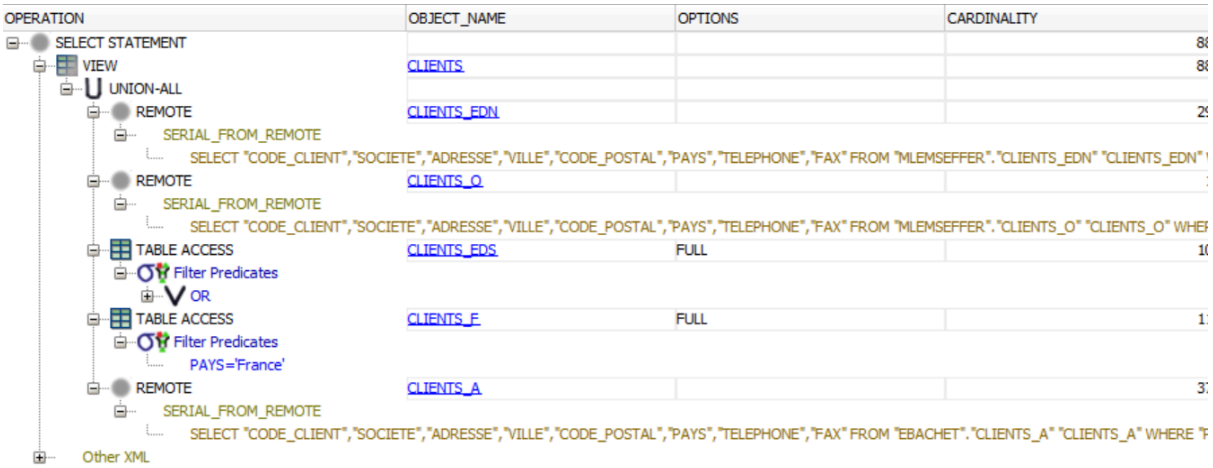
Nous avons choisi de représenter cette requête pour la comparer avec l'avant dernière requête (cf. Requêtes distribuées : tests et optimisations) où nous n'utilisons plus des vues mais des vues matérialisées.

Cette requête est optimale, car en plus de l'usage d'un index pour la jointure, on a un tri (SORT), ce qui permet à la jointure de se faire plus vite.

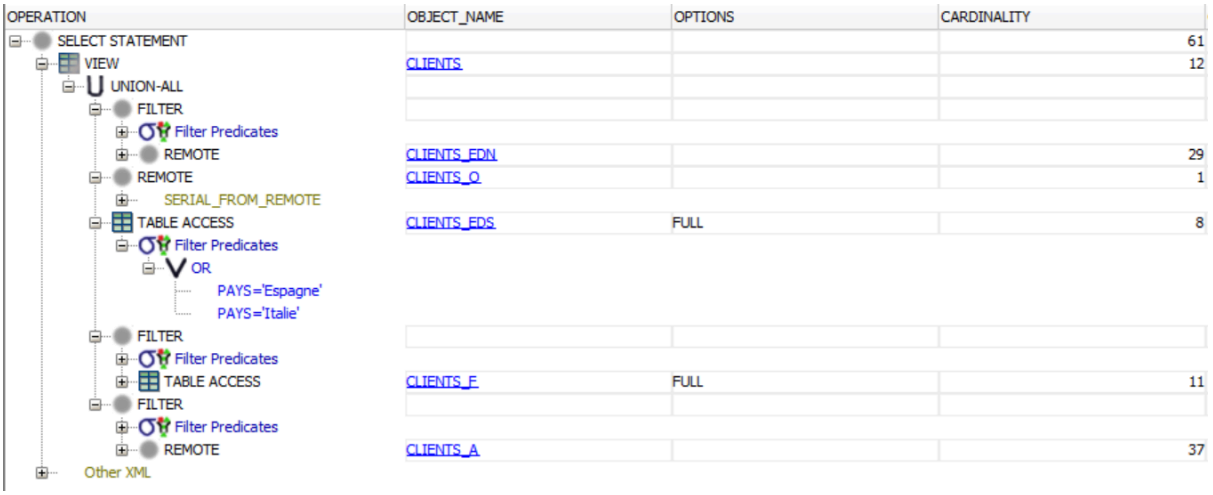
B. Site Europe du Sud

Dans cette partie, nous allons essayer différentes requêtes et analyser leur plan d'exécution.

- **Requête 1 :** `SELECT * FROM CLIENTS;`



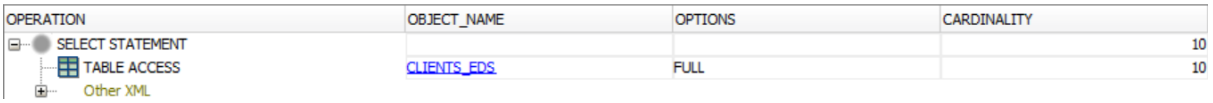
- **Requête 2 :** `SELECT * FROM CLIENTS WHERE PAYS IN ('Italie', 'Espagne');`



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY
SELECT STATEMENT			61
VIEW	CLIENTS		12
UNION-ALL			
FILTER			
Filter Predicates			
REMOTE	CLIENTS_EDN		29
REMOTE	CLIENTS_O		1
SERIAL_FROM_REMOTE			
TABLE ACCESS	CLIENTS_EDS	FULL	8
Filter Predicates			
OR			
PAYS='Espagne'			
PAYS='Italie'			
FILTER			
Filter Predicates			
TABLE ACCESS	CLIENTS_F	FULL	11
FILTER			
Filter Predicates			
REMOTE	CLIENTS_A		37
Other XML			

—> On remarque que Oracle cherche directement dans la table de l'Europe de Sud.

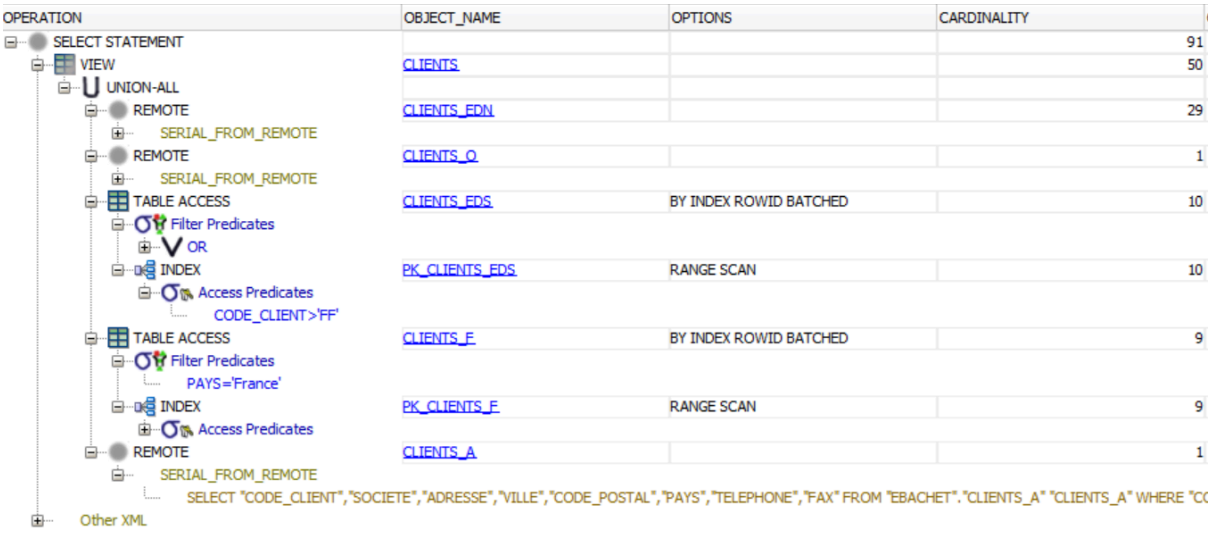
- **Requête 3 :** `SELECT * FROM CLIENTS_EDS;`



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY
SELECT STATEMENT			10
TABLE ACCESS	CLIENTS_EDS	FULL	10
Other XML			

—> On a effectué un select sur une table de notre base, il n'y a rien à remarquer.

- **Requête 4 :** `SELECT * FROM CLIENTS WHERE (Code_CLIENT > 'SA' OR Code_CLIENT > 'FF')`

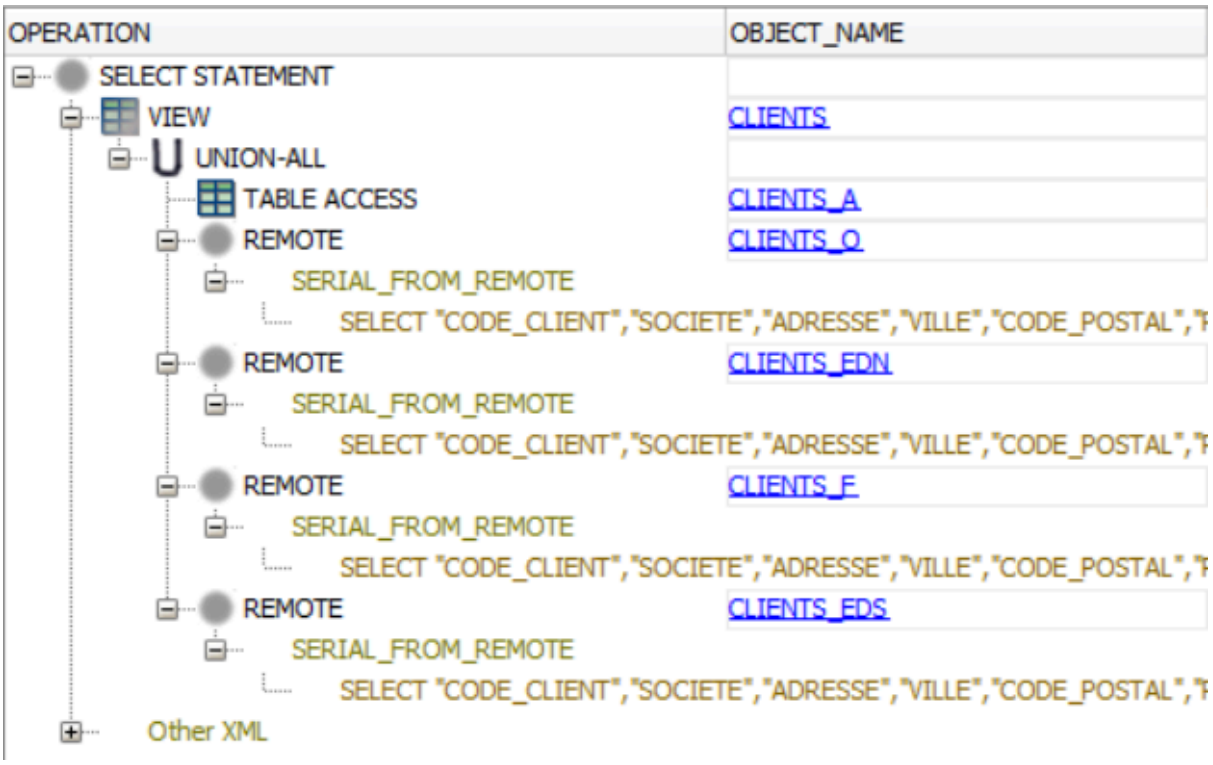


OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY
SELECT STATEMENT			91
VIEW	CLIENTS		50
UNION-ALL			
REMOTE	CLIENTS_EDN		29
SERIAL_FROM_REMOTE			
REMOTE	CLIENTS_O		1
SERIAL_FROM_REMOTE			
TABLE ACCESS	CLIENTS_EDS	BY INDEX ROWID BATCHED	10
Filter Predicates			
OR			
INDEX	PK_CLIENTS_EDS	RANGE SCAN	10
Access Predicates			
CODE_CLIENT>'FF'			
TABLE ACCESS	CLIENTS_F	BY INDEX ROWID BATCHED	9
Filter Predicates			
PAYS='France'			
INDEX	PK_CLIENTS_F	RANGE SCAN	9
Access Predicates			
REMOTE	CLIENTS_A		1
SERIAL_FROM_REMOTE			
SELECT "CODE_CLIENT","SOCIETE","ADRESSE","VILLE","CODE_POSTAL","PAYS","TELEPHONE","FAX" FROM "EBACHET"."CLIENTS_A" "CLIENTS_A" WHERE "CC			
Other XML			

—> Le plan d'exécution montre une vue utilisant un UNION ALL pour combiner des données de sources locales et distantes, optimisées avec des scans indexés sur les tables locales et des accès distants via SERIAL_FROM_REMOTE, réduisant ainsi les lectures inutiles.

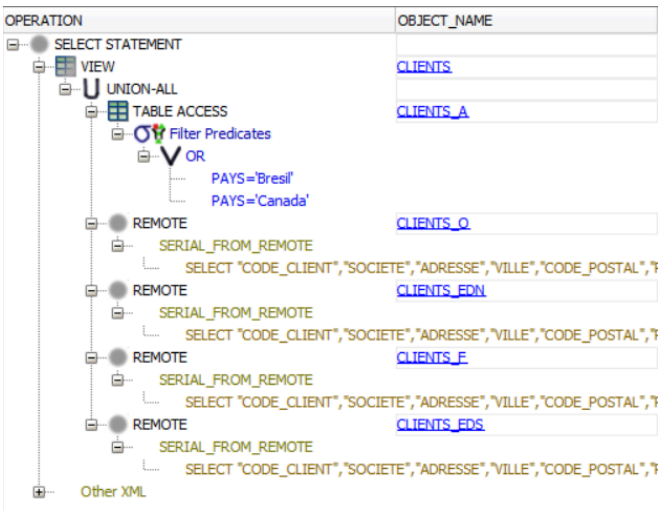
C. Site Amérique

Requête 1 : `SELECT * FROM Clients;`



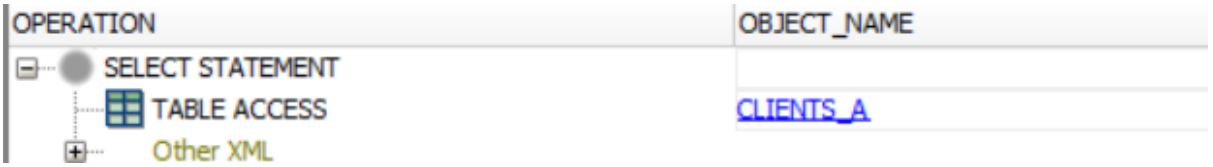
Le SGBD réalise un UNION ALL entre tous les fragments pour obtenir la table clients, ce qui est conforme avec la VIEW qui a été créé pour la table clients.

Requête 2 : `SELECT * FROM Clients WHERE PAYS IN ('Canada','Bresil');`



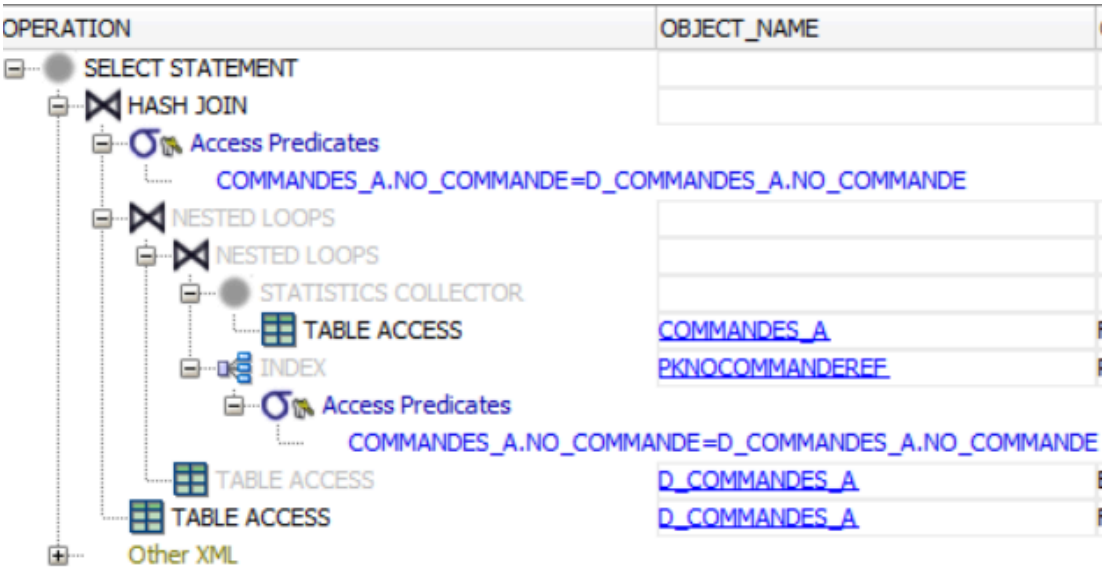
Oracle se sert de la condition pour chercher les tuples qui nous intéressent dans la table Clients_A, ce qui optimise la recherche.

Requête 3 :`SELECT * FROM Clients_A;`



Dans cette requête, nous faisons la sélection d'une table locale complète, ainsi, il y a un TABLE ACCESS qui est un parcours simple de la table.

Requête 4 :`SELECT * FROM Commandes_A NATURAL JOIN D_Commandes_A;`



Lors de cette requête, nous joignons à chaque commande ses détails. La jointure se fait avec un INDEX, la requête est donc optimale.

V. Réplications

A. Mise en œuvre des réplications sur le site Europe du Nord

1. Rappel du binôme responsable

B3110: ERABHAOUI-LEMSEFFER-IICH

2. Objectifs

L'objectif de la réplication est d'optimiser les requêtes et transactions locales. De même, la réplication permet d'avoir des sauvegardes des données de toute la base en cas de panne sur un autre site.

3. Liste des replications prévues

Sur notre site, nous n'avons ni la table **CATEGORIES**, ni la table **PRODUITS,** ni **EMPLOYÉS**. Nous allons ainsi répliquer ces trois tables sur notre base.

4. Analyse

Catégories : Cette table ne contenant que 8 tuples, nous allons faire un refresh complet et ce toutes les 24 heures. On suppose qu'une nouvelle catégorie ne sera déployée que le lendemain.

Produits : Cette table contient 77 tuples. Ainsi, nous allons faire un refresh fast nécessitant un log de la part du site Europe du Sud. On suppose qu'un nouveau produit ne sera déployé que le lendemain.

Employés : Cette table contient 9 tuples. Nous allons donc faire un refresh complet, et ce toutes les 24 heures. En effet, une personne embauchée commencera à travailler au minimum le lendemain.

5. Mise en œuvre des réplications pour les besoins locaux

```
CREATE MATERIALIZED VIEW MV_Produits
REFRESH FAST
NEXT sysdate +(1)
AS SELECT * FROM PRODUITS;

CREATE MATERIALIZED VIEW MV_Employes
REFRESH COMPLETE
NEXT sysdate +(1)
AS SELECT * FROM EMPLOYES;

CREATE MATERIALIZED VIEW MV_Categories
REFRESH COMPLETE
NEXT sysdate +(1)
AS SELECT * FROM CATEGORIES;
```

6. Demandes d'autres sites portant sur des fragments gérés localement

La table FOURNISSEURS ne contient actuellement que 29 tuples, ainsi,

```
CREATE MATERIALIZED VIEW LOG ON fournisseurs;
GRANT SELECT ON MLOG$_fournisseurs TO helkarchou;
GRANT SELECT ON MLOG$_fournisseurs TO ebachet;
```

B. Mise en œuvre des réplications sur le site Europe du Su

1. Rappel du binôme responsable

Le binôme responsable du site Europe du Sud est le B3107, composé de Hamza El Karchouni et Yliess Bellargui

2. Objectifs

L'objectif principal des réplications est d'améliorer l'accès local aux données critiques, d'optimiser les performances des transactions locales et de garantir la synchronisation inter-sites pour assurer l'intégrité et le partage des données.

3. Liste des replications prévues

Les tables Produits et Categories sont sur notre site, donc nous devons répliquer uniquement les tables Fournisseurs et Employes.

4. Analyse

- La table Fournisseurs contient un grand nombre de tuples, c'est pourquoi nous optons pour une réplication Fast afin d'éviter le transfert complet de la table. De plus, un rafraîchissement quotidien sera mis en place.
- La table Employés contient peu de tuples, ce qui justifie une réplication complète plutôt qu'un système de log, jugé plus coûteux. Un rafraîchissement quotidien sera également mis en place.

5. Mise en œuvre des réplications pour les besoins locaux

- **Fournisseurs :**

```
CREATE MATERIALIZED VIEW MV_Fournisseurs
REFRESH FAST
NEXT sysdate +(1)
AS SELECT * FROM mlemseffer.Fournisseurs@DB2_EDN;
```

- **Employes :**

```
CREATE MATERIALIZED VIEW MV_Employes
REFRESH COMPLETE
NEXT sysdate + (1)
AS
SELECT * FROM ebachet.Employes@DB4_A;
```

6. Demandes d'autres sites portant sur des fragments gérés localement

D'autres sites (Europe du Nord, Amérique) ont des besoins spécifiques pour accéder à certaines données gérées sur le site Europe du Sud, notamment Produits.
En effet, cette table contient beaucoup de tuples, ce qui est préférable d'utiliser la réplication en Fast.

```
CREATE MATERIALIZED VIEW LOG ON Produits;
GRANT SELECT ON MLOG$_Produits TO mlemseffer;
GRANT SELECT On MLOG$_Produits TO ebachet;
```

C. Mise en œuvre des réplications sur le site Amerique

1. Rappel du binôme responsable

B3103 : Andreea-Critiana Vlad, Elise Bachet

2. Objectifs

Il y a deux objectifs majeurs à la réplication :
- Optimiser le temps d'exécution des transactions
- Améliorer l'accès local aux données les plus utilisées.

3. Liste des replications prévues

La table *Employes* étant dans notre base, nous allons répliquer les tables Fournisseurs, Produits et Categories

4. Analyse

- La table Fournisseurs contient beaucoup de tuples, donc il vaut mieux faire une réplication Fast pour ne pas répliquer toute la table.
- La table Produits contient beaucoup de tuples, donc il vaut mieux faire une réplication Fast pour ne pas répliquer toute la table.
- La table Categories ne contient pas beaucoup de tuples, donc il vaut mieux faire une réplication complète pour ne pas utiliser de LOG.

5. Mise en œuvre des réplications pour les besoins locaux

- **Fournisseurs :**

```
CREATE MATERIALIZED VIEW MV_Fournisseurs
REFRESH FAST
NEXT sysdate +(1)
AS SELECT * FROM mlemseffer.Fournisseurs@DBL_EDN;
```

- **Produits :**

```
CREATE MATERIALIZED VIEW MV_Produits
REFRESH FAST
NEXT sysdate +(1)
AS SELECT * FROM helkarchou.produits@DBL_EDS;
```

- **Catégories :**

```
CREATE MATERIALIZED VIEW MV_Categories
REFRESH COMPLETE
NEXT sysdate + (1)
AS
SELECT * FROM helkarchou.categories@DBL_EDS;
```

6. Demandes d'autres sites portant sur des fragments gérés localement

La table Employes n'a pas beaucoup de tuples, il est donc plus judicieux de la répliquer de manière complète.

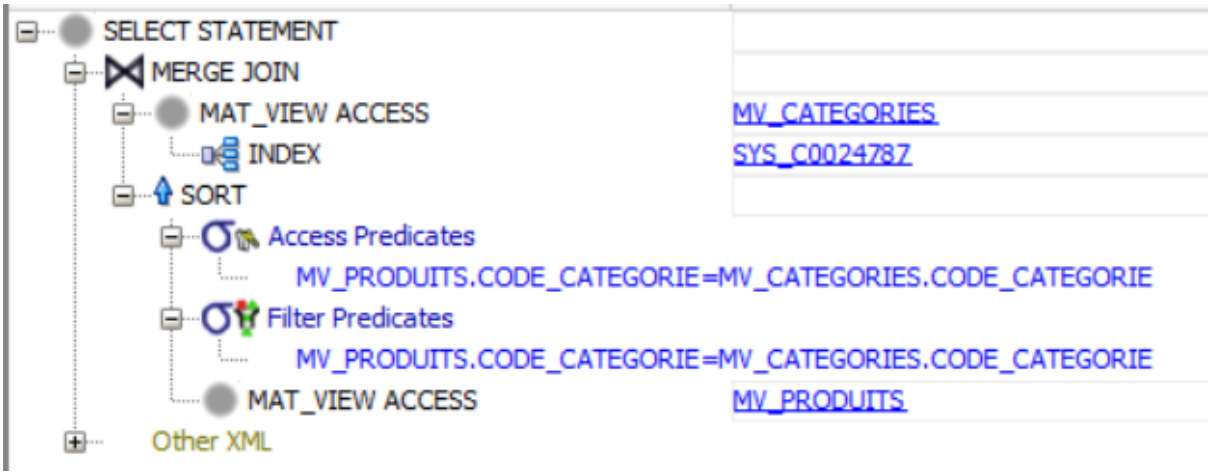
D. Bilan global des réplications mises en œuvre sur les différents sites.

In fine, grâce aux différentes réplications et vues sur chaque site, nous avons toutes les données de la table Ryori dans son état initial. Ainsi, chaque site est désormais peut travailler indépendamment sur ses données et suivre les mises-à-jours réalisées par les autres sites grâce aux vues matérilisées.

VI. Requêtes distribuées : tests et optimisations

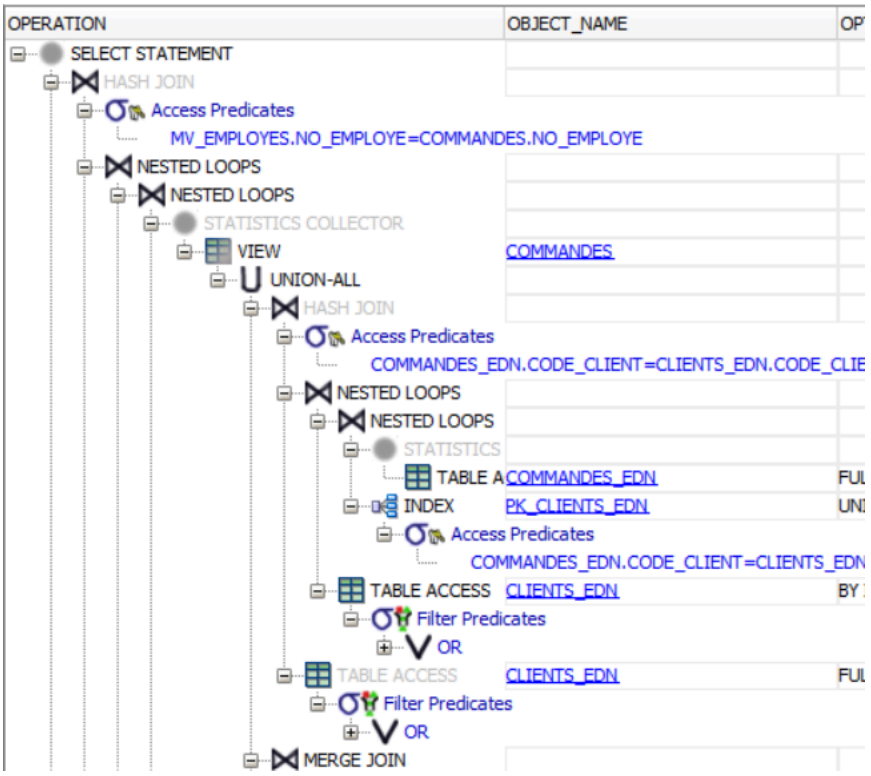
A. Site Europe du Nord

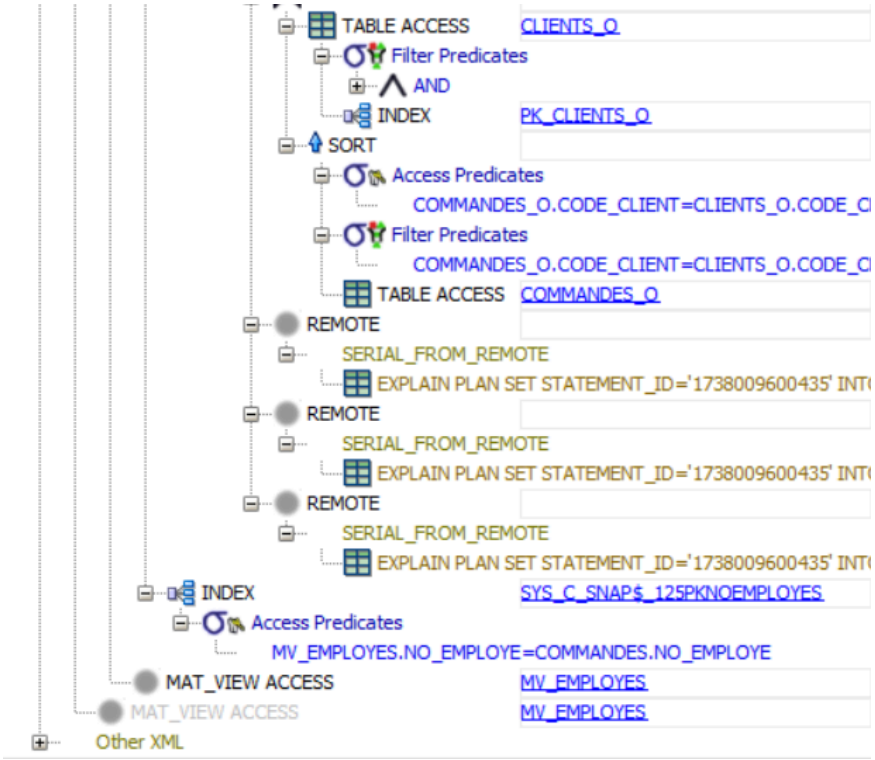
1. **Requête 1 :** `SELECT * FROM MV_PRODUITS NATURAL JOIN MV_CATEGORIES;`



Cette requête ne diffère pas énormément de la **requête 6** testée précédemment sauf qu'à la place des "TABLE ACCESS", nous avons des "MAT_VIEW_ACCESS". Cela n'a pas pour effet d'altérer la complexité algorithmique de la jointure.

2. **Requête 2 :** `SELECT * FROM MV_EMPLOYES NATURAL JOIN Commandes;`





Finalement, nous avons choisi de faire une jointure naturelle entre la vue matérialisée MV_EMPLOYES et la vue COMMANDES. Dans ce cas, le SGBD utilise un index pour sélectionner les tuples vérifiant MV_EMPLOYES.NO_EMPLOYE = COMMANDES.NO_EMPLOYE. La requête est donc optimale.

B. Site Europe du Sud

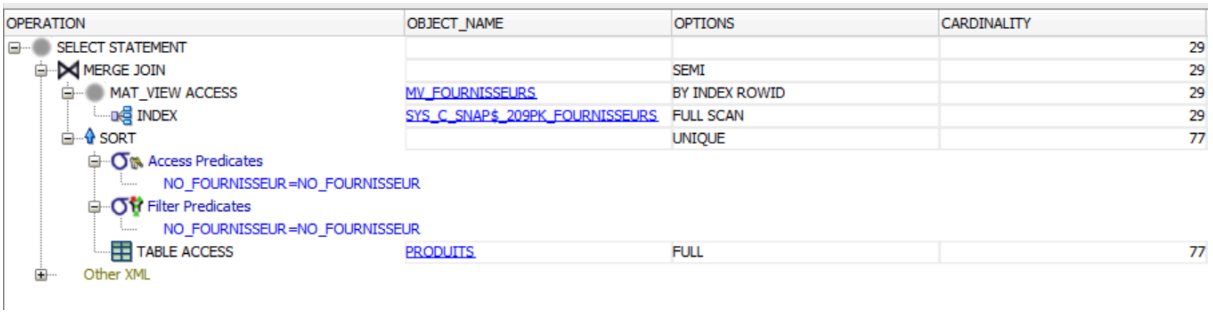
- Nous avons effectué une requête qui fait une jointure entre la table Fournisseurs qui a été répliquée et la table Produits.

La requête est la suivante :

```
SELECT * FROM MV_Fournisseurs
WHERE NO_Fournisseur
IN ( SELECT NO_Fournisseur FROM Produits);
```

—>On obtient 29 tuples (aucune perte ou doublons) .

- En analysant le plan d'exécution de la requête ci-dessous :

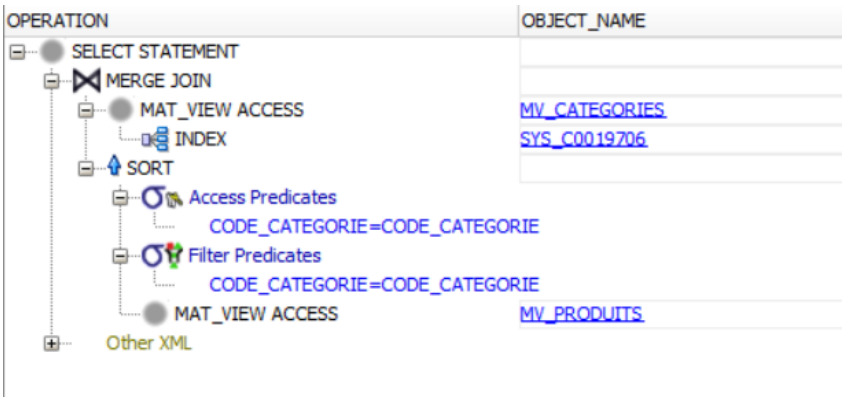


On remarque que la vue matérialisée est bien utilisée.

C. Site Amérique

Nous avons utilisé une requête qui fait une jointure entre les table Categories et Produits, répliquées de deux sites différents.

```
SELECT * FROM MV_Categories
WHERE code_categorie
IN (SELECT code_categorie FROM MV_Produits);
```



Ainsi, on constate qu'on utilise bel et bien la vue matérialisée.

VII. Conclusion

Pour conclure, durant ce TP, nous avons fragmenté la base de données Ryori selon des critères géographiques afin de la distribuer sur trois sites : Europe du Nord, Europe du Sud et Amérique. Chaque site a été configuré pour gérer localement les données qui lui sont pertinentes, avec des fragments spécifiques attribués pour optimiser les performances et l'accès. Nous avons également mis en place des contraintes d'intégrité, des triggers pour gérer les dépendances distantes, et des droits d'accès pour sécuriser les opérations. Enfin, des tests approfondis ont validé la cohérence des vues centralisées et le bon fonctionnement du système distribué.