

Projet du Base du données réparties

ORACLE®

Réalisé par :

- Sousaid Abdellah

A.U : 2022/2023

Contenu

Création d'un réseau virtuel :

Création de Nouveaux Comptes :.....

Créer des database link:

Création d'une BDD centrale:.....

Création des fragments de la BDD centrale:

Créer des Synonymes :.....

Synchronisation CRUD :

Les Requêtes réparties:

Les contraintes:

Les requêtes:

1)- Création d'un réseau virtuel

Un réseau interne virtuel a été créé dans votre machine physique. Ce réseau est constitué de trois serveurs Oracle, représentés par les machines virtuelles suivantes : « Centre », « Site1 » et « Site2 ». Le serveur central a été configuré avec l'adresse IP 10.111.100.11, tandis que le site1 dispose des adresses IP 10.111.100.100 et 10.111.100.10 pour les machines du site2. Chaque machine virtuelle a été configurée avec les adresses IP correspondantes, permettant ainsi une communication interne sécurisée et efficace entre les serveurs Oracle. Les données et les ressources peuvent désormais être partagées au sein de ce réseau virtuel.

```
C:\Users\Oracle>ping 10.111.100.100

Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Oracle>ping 10.111.100.100

Envoi d'une requête 'Ping' 10.111.100.100 avec 32 octets de données :
Réponse de 10.111.100.100 : octets=32 temps=2 ms TTL=128
Réponse de 10.111.100.100 : octets=32 temps=1 ms TTL=128
Réponse de 10.111.100.100 : octets=32 temps=2 ms TTL=128
Réponse de 10.111.100.100 : octets=32 temps=1 ms TTL=128

Statistiques Ping pour 10.111.100.100:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
    Durée approximative des boucles en millisecondes :
        Minimum = 1ms, Maximum = 2ms, Moyenne = 1ms

C:\Users\Oracle>

C:\Users\Oracle>PING 10.111.100.10

Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Oracle>PING 10.111.100.10

Envoi d'une requête 'Ping' 10.111.100.10 avec 32 octets de données :
Réponse de 10.111.100.10 : octets=32 temps=1 ms TTL=128
Réponse de 10.111.100.10 : octets=32 temps=1 ms TTL=128
Réponse de 10.111.100.10 : octets=32 temps=1 ms TTL=128
Réponse de 10.111.100.10 : octets=32 temps=2 ms TTL=128

Statistiques Ping pour 10.111.100.10:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
    Durée approximative des boucles en millisecondes :
        Minimum = 1ms, Maximum = 2ms, Moyenne = 1ms

C:\Users\Oracle>_
```

```

C:\Users\Oracle>ping 10.111.100

Envoi d'une requête 'Ping' 10.111.0.100 avec 32 octets de données :
Réponse de 10.111.100.11 : Impossible de joindre l'hôte de destination.
Réponse de 10.111.100.11 : Impossible de joindre l'hôte de destination.
Réponse de 10.111.100.11 : Impossible de joindre l'hôte de destination.
    
```

2)- Création de Nouveaux Comptes

Connectez-vous à chaque machine en tant qu'administrateur en utilisant la commande suivante :conn / as sysdba, Une fois connecté, vous pouvez créer de nouveaux comptes en utilisant la syntaxe suivante :CREATE USER nom_utilisateur IDENTIFIED BY mot_de_passe;

Remplacez "nom_utilisateur" par le nom que vous souhaitez donner à l'utilisateur et "mot_de_passe" par le mot de passe désiré pour ce compte.

Dans notre cas :

- Machine centrale : conn admin/admin ;

VM-Centrale - VMware Workstation 15 Player (Non-commercial use only)

```

Player | || | |
-----|-----
C:\> Invite de commandes - sqlplus / as sysdba;

SQL> select name from v$database;

NAME
-----
BDDGLOBA

SQL> conn / as sysdba;
SP2-0306: Option non valide.
Syntaxe : CONNECT [ <logon> [/ <proxy> ] AS <SYSDBA|SYSOPER|SYSASM> ] [ edition=value
] ]
o" <logon> ::= <username> [ / <password> ] [ @ <connect_identifier> ]
<proxy> ::= <proxyuser> [ <username> ] [ / <password> ] [ @ <connect_identifier> ]
SQL> conn / as sysdba;
Connecté.
SQL> create user admin identified by admin;

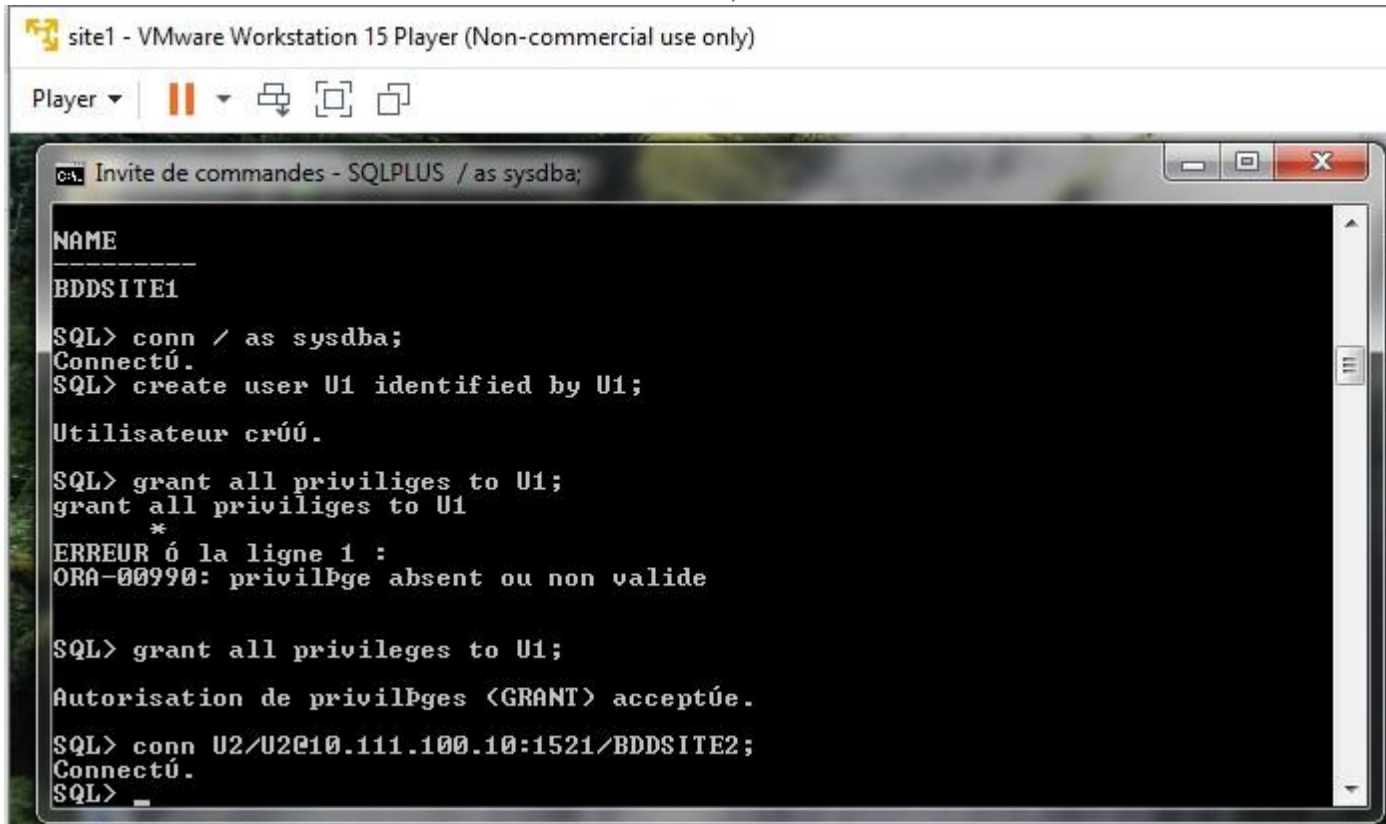
Utilisateur créé.

SQL> grant all privileges to admin;

Autorisation de privilèges <GRANT> acceptée.

SQL> conn U2/U2@10.111.100.10:1521/BDDSITE2;
Connecté.
SQL>
    
```

- Machine Site1 : conn U1/U1 ;



site1 - VMware Workstation 15 Player (Non-commercial use only)

Player ▾ || ▾ □ □ □

```

C:\ Invite de commandes - SQLPLUS / as sysdba;

NAME
-----
BDDSITE1

SQL> conn / as sysdba;
Connecté.
SQL> create user U1 identified by U1;

Utilisateur créé.

SQL> grant all privileges to U1;
grant all privileges to U1
      *
ERREUR ó la ligne 1 :
ORA-00990: privilège absent ou non valide

SQL> grant all privileges to U1;
Autorisation de privilèges (GRANT) acceptée.

SQL> conn U2@10.111.100.10:1521/BDDSITE2;
Connecté.
SQL>
    
```

- Machine Site2 : conn U2/U2 ;

```

SQL> create user U2 identified by U2;

Utilisateur créé.

SQL>
SQL> grant all privileges to U2.
2 grant all privileges to U2;
grant all privileges to U2.
      *
ERREUR ó la ligne 1 :
ORA-00933: la commande SQL ne se termine pas correctement

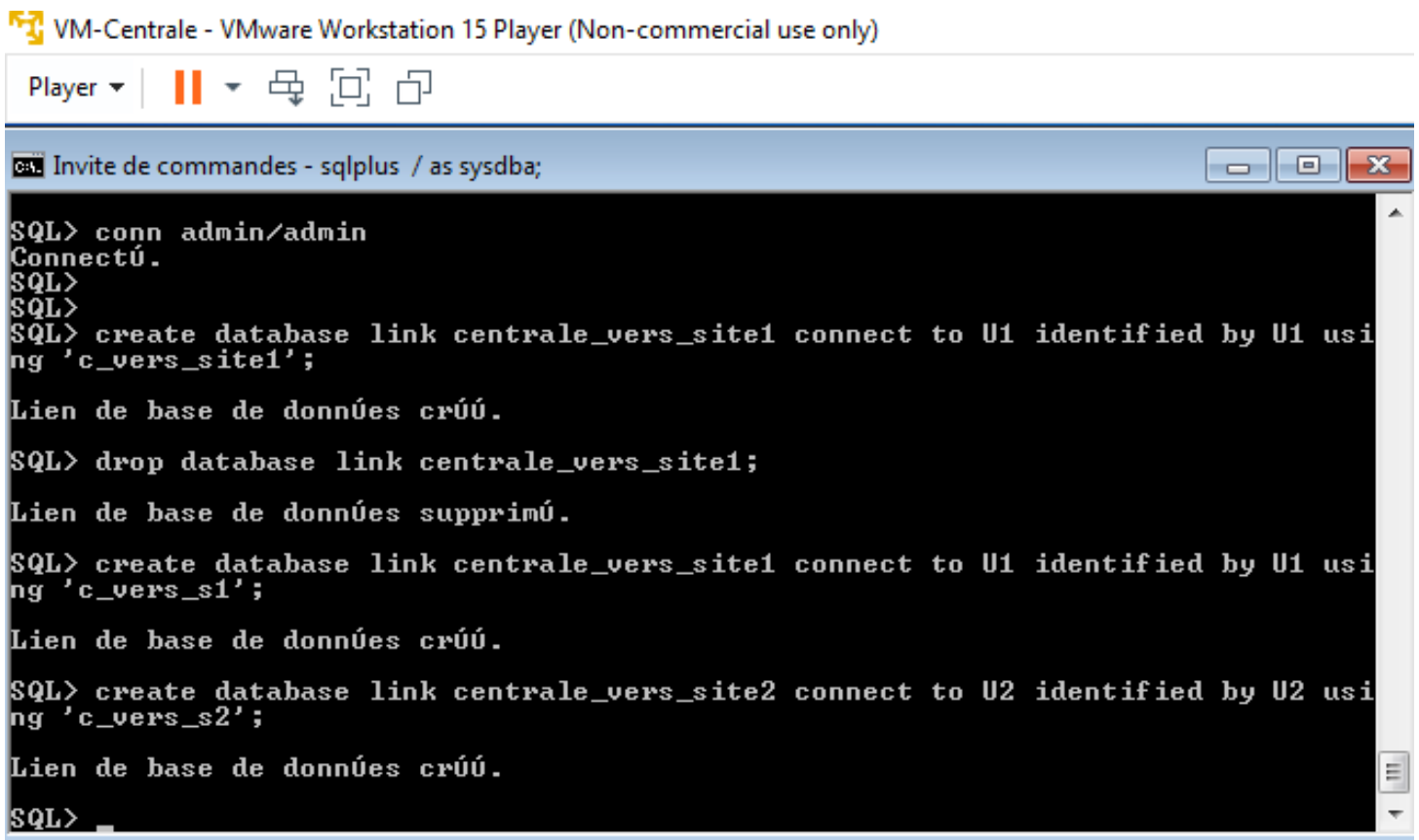
SQL> grant all privileges to U2;
Autorisation de privilèges (GRANT) acceptée.

SQL> select username from dba_users;

USERNAME
    
```

3)- Créer des database link :

Le rôle de créer des liens de base de données (database links) dans Oracle est de permettre la communication et l'accès à des bases de données distantes à partir d'une base de données locale. Les liens de base de données permettent à une base de données Oracle de se connecter et d'interagir avec d'autres bases de données Oracle situées sur des serveurs distincts.



```
VM-Centrale - VMware Workstation 15 Player (Non-commercial use only)
Player | || | |
C:\ Invite de commandes - sqlplus / as sysdba;
SQL> conn admin/admin
Connecté.
SQL>
SQL>
SQL> create database link centrale_vers_site1 connect to U1 identified by U1 using 'c_vers_site1';
Lien de base de données créé.
SQL> drop database link centrale_vers_site1;
Lien de base de données supprimé.
SQL> create database link centrale_vers_site1 connect to U1 identified by U1 using 'c_vers_s1';
Lien de base de données créé.
SQL> create database link centrale_vers_site2 connect to U2 identified by U2 using 'c_vers_s2';
Lien de base de données créé.
SQL> _
```

site 2 - VMware Workstation 15 Player (Non-commercial use only)

Player ▾ |  ▾ |   

```

C:\ Invite de commandes - SQLPLUS / AS SYSDBA.
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> conn U2/U2
Connecté.
SQL> create database link site2_vers_centrale connect to admin identified by ad
min using 's2_vers_centrale';

Lien de base de données créé.

SQL> create database link site2_vers_site1 connect to U1 identified by U1 using
'versSite1';

Lien de base de données créé.

SQL> select count(*) from client@site2_vers_centrale;

  COUNT(*)
-----
       1000

SQL>
    
```

site 1 - VMware Workstation 15 Player (Non-commercial use only)

Player ▾ |  ▾ |   

```

C:\ Invite de commandes - SQLPLUS / as sysdba;
ORA-01017: invalid username/password; logon denied

Avertissement : vous n'êtes plus connecté à ORACLE.
SQL> conn u1/u1;
ERROR:
ORA-01017: invalid username/password; logon denied

SQL> conn U1/U1;
Connecté.
SQL> create database link site1_vers_site2 connect to U2 identified by U2 using
'versSite2';

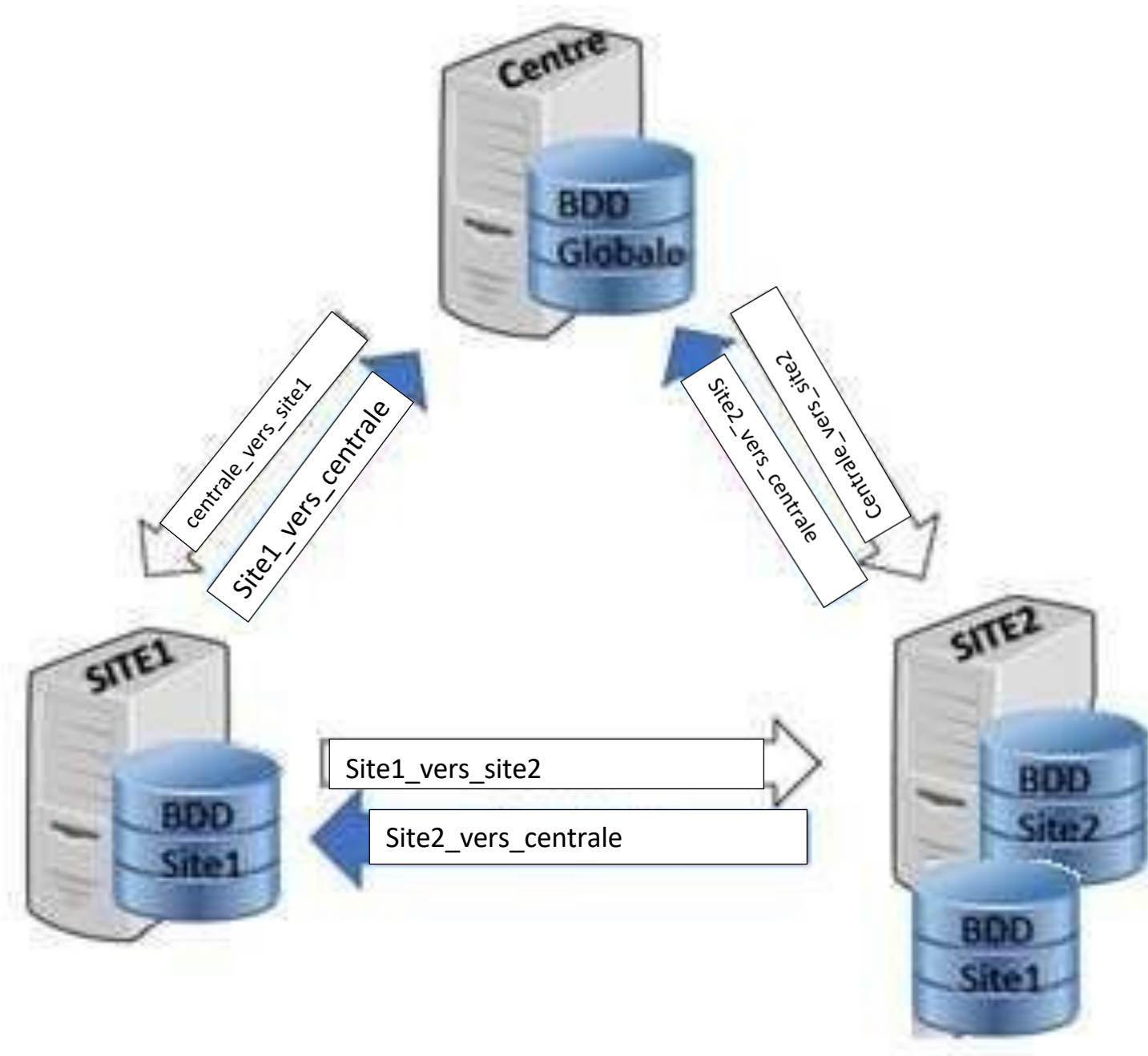
Lien de base de données créé.

SQL> create database link site1_vers_centrale connect to admin identified by adm
in using 's1_vers_centrale';

Lien de base de données créé.

SQL> select count(*) from client@site1_vers_centrale;

  COUNT(*)
-----
    
```



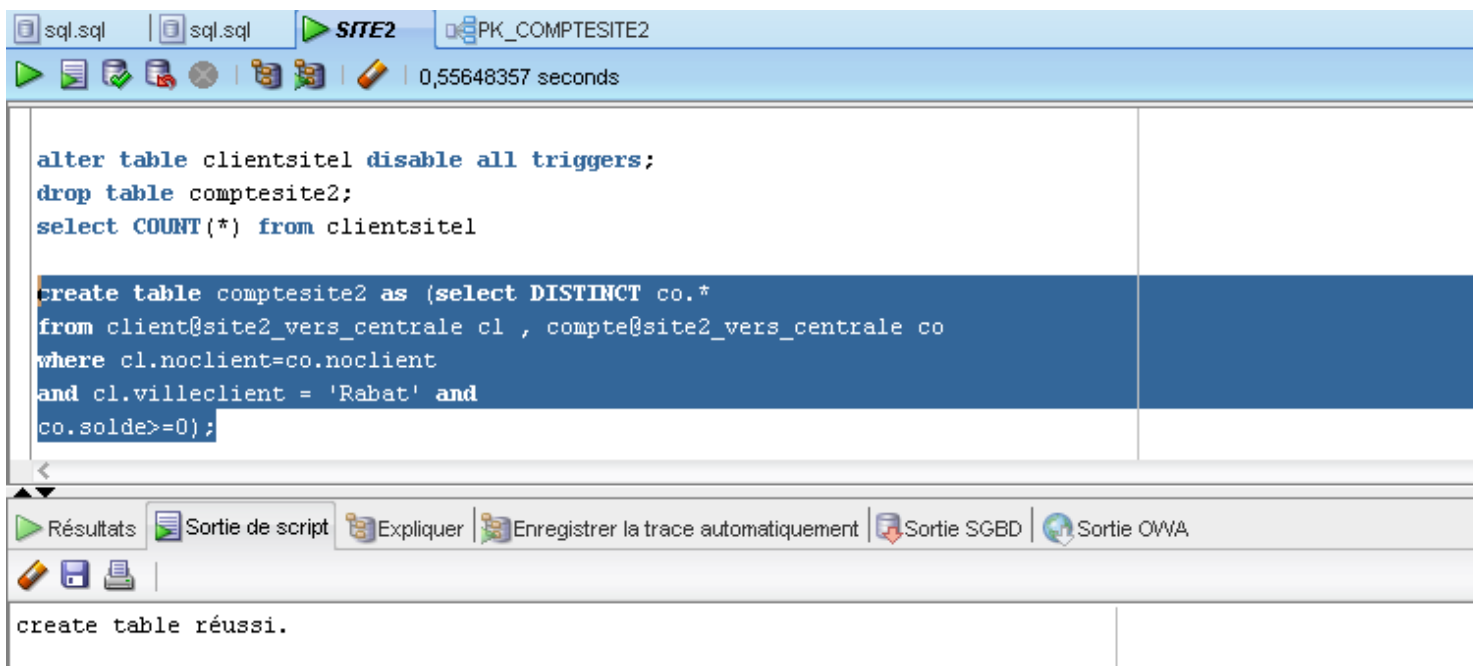
4)- Création des fragments de la BDD centrale :

Création des bases de données BDDGLOBALE, BDDSITE1, et BDDSITE2 dans les machines respectivement machine centrale , machine 1 et machine 2

fragmenter la base de donne globale sur les deux sites1 et 2 selon les cratères suivant :

$$R_1 = \sigma_{VilleClient = 'Casablanca'} \text{ AND } Solde < 0 (Clients \bowtie Comptes)$$

$$R_2 = \sigma_{VilleClient = 'Rabat'} \text{ AND } Solde \geq 0 (Clients \bowtie Comptes)$$



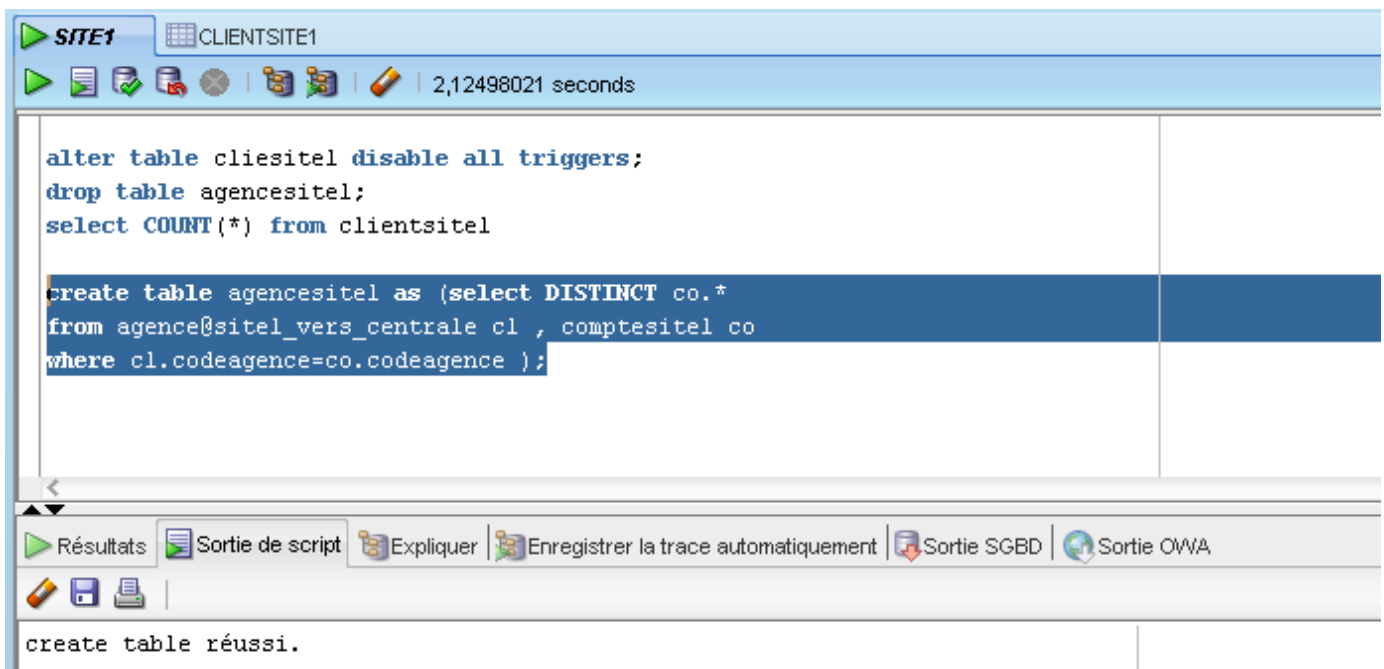
sql.sql | sql.sql | **SITE2** | PK_COMPESITE2
0,55648357 seconds

```
alter table clientsitel disable all triggers;
drop table comptesite2;
select COUNT(*) from clientsitel

create table comptesite2 as (select DISTINCT co.*
from client@site2_vers_centrale cl , compte@site2_vers_centrale co
where cl.noclient=co.noclient
and cl.villeclient = 'Rabat' and
co.solde>=0);
```

Résultats | Sortie de script | Expliquer | Enregistrer la trace automatiquement | Sortie SGBD | Sortie OWA

create table réussi.



SITE1 | CLIENTSITE1
2,12498021 seconds

```
alter table cliesitel disable all triggers;
drop table agencesitel;
select COUNT(*) from clientsitel

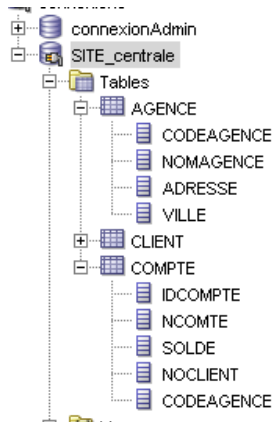
create table agencesitel as (select DISTINCT co.*
from agence@site1_vers_centrale cl , comptesitel co
where cl.codeagence=co.codeagence );
```

Résultats | Sortie de script | Expliquer | Enregistrer la trace automatiquement | Sortie SGBD | Sortie OWA

create table réussi.

5)- Créer des Synonymes :

En Oracle, les synonymes sont des objets de base de données qui permettent de fournir un autre nom ou une autre référence à un objet existant, qu'il s'agisse d'une table, d'une vue, d'une procédure stockée, d'une fonction, d'un package ou d'un autre synonyme. Les synonymes offrent un moyen pratique de simplifier l'accès aux objets de base de données et d'améliorer la lisibilité et la compréhension du code.

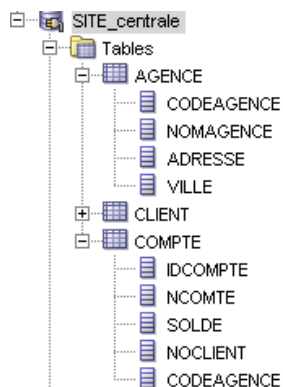


```

CREATE PUBLIC SYNONYM ClientsSite2 FOR ClientSite2@centrale_vers_site2;
CREATE PUBLIC SYNONYM ComptesSite2 FOR ComptesSite2@centrale_vers_site2;
CREATE PUBLIC SYNONYM AgenceSite1 FOR AgencesSite1@centrale_vers_site2;
    
```

Résultats | Sortie de script | Expliquer | Enregistrer la trace automatiquement | Sortie SGBD | Sortie OWA

CREATE PUBLIC réussi.
 CREATE PUBLIC réussi.
 CREATE PUBLIC réussi.



```

CREATE PUBLIC SYNONYM ClientsSite1 FOR ClientSite1@centrale_vers_site1;
CREATE PUBLIC SYNONYM ComptesSite1 FOR ComptesSite1@centrale_vers_site1;
CREATE PUBLIC SYNONYM AgenceSite1 FOR AgencesSite1@centrale_vers_site1;
    
```

Résultats | Sortie de script | Expliquer | Enregistrer la trace automatiquement | Sortie SGBD | Sortie OWA

CREATE PUBLIC réussi.

6)- Les contraintes :

```

ALTER TABLE client
ADD CONSTRAINT pk_Client
PRIMARY KEY(noclient);
    
```

Résultats | Sortie de script | Expliquer | Enregistrer la trace automatiquement | Sortie SGBD | Sortie OWA

ALTER TABLE client réussi.

```

ALTER TABLE comptesite2
ADD CONSTRAINT PK_Compte2 PRIMARY KEY (idcompte);

ALTER TABLE clientsitel
ADD CONSTRAINT PK_Clientl1 PRIMARY KEY (noclient);

ALTER TABLE comptesitel
ADD CONSTRAINT PK_Comptel1 PRIMARY KEY (idcompte);

ALTER TABLE comptesitel
ADD CONSTRAINT FKS_CLIENT_COMPTEsitlrr
FOREIGN KEY (noclient)
REFERENCES clientsitel (noclient)
ON DELETE CASCADE;
    
```

```

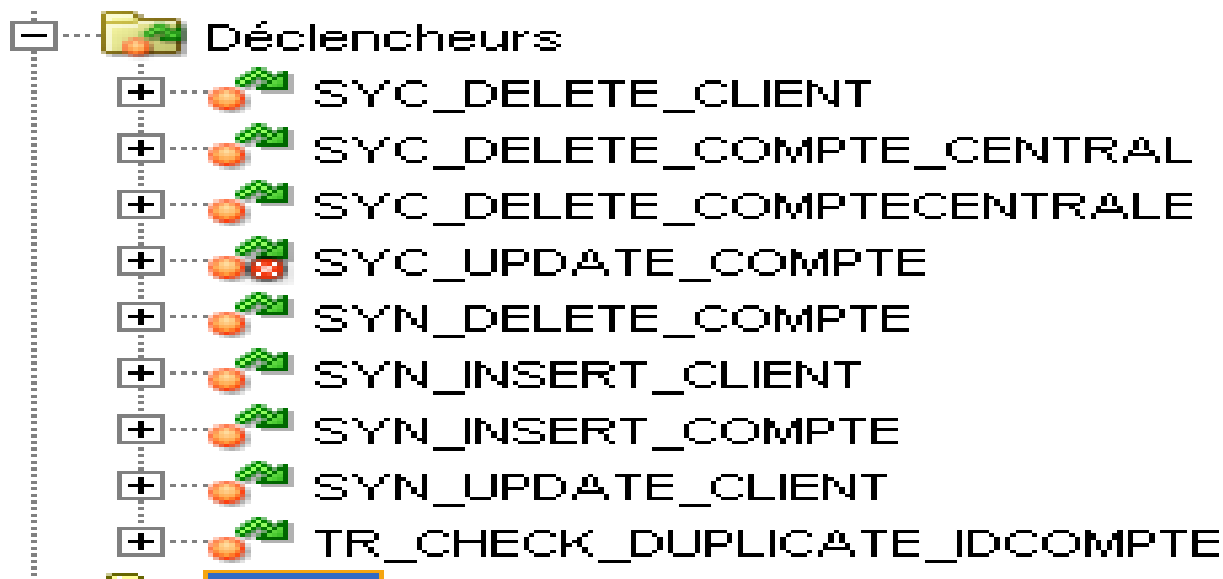
ALTER TABLE clientsitel
ADD CONSTRAINT PK_Clientl PRIMARY KEY (noclient);
    
```

Résultats | Sortie de script | Expliquer | Enregistrer la trace automatiquement | Sortie SGBD | Sortie OWA

ALTER TABLE comptesitel réussi.
 ALTER TABLE clientsitel réussi.

7)- Synchronisation CRUD :

Triggers de la machine



Triggers : syn_insert_compte :

```

create or replace trigger syn_insert_compte
before insert on compte
for each row
begin
    declare
        v_count NUMBER := 0;
        v_idcompte compte.idcompte%TYPE;
        CURSOR c_compte IS
            SELECT idcompte
            FROM compte;
        V client.villeclient%type;
        S compte.solde%type:=:new.solde;
        nb integer;
    begin

        -- Récupérer la nouvelle valeur de idcompte
        v_idcompte := :NEW.idcompte;

        -- Vérifier si l'idcompte existe déjà
    
```

<pre> FOR rec IN c_compte LOOP IF v_idcompte = rec.idcompte THEN v_count := v_count + 1; END IF; END LOOP; -- Si l'idcompte existe déjà, lever une exception IF v_count > 0 THEN RAISE_APPLICATION_ERROR(-20001, 'La clé primaire (idcompte) existe déjà dans la table "compte".'); ROLLBACK; END IF; select cl.villeclient into V FROM client cl where cl.noclient=:new.noclient; if(V='Casablanca' AND S<0) then select count(noclient) into nb FROM clientsitel@centrale_vers_sitel where noclient=:new.noclient; if nb=0 then DBMS_OUTPUT.PUT_LINE('idclient n existe pas'); ROLLBACK; ELSE insert into comptesitel@centrale_vers_sitel values (:new.idcompte, :new.Ncompte, </pre>	
<pre> : new.solde,:new.noclient, :new.codeagence); end if; elsif(V='Rabat' AND S >= 0) then select count(noclient) into nb FROM clientsite2@centrale_vers_site2 where noclient=:new.noclient; if nb=0 then DBMS_OUTPUT.PUT_LINE('idclient n existe pas'); ROLLBACK; else insert into comptesite2@centrale_vers_site2 values (:new.idcompte, :new.Ncompte, : new.solde,:new.noclient, :new.codeagence); end if; end if; END ; END ; </pre>	

Trigger : syn_insert_client :

```
create or replace TRIGGER syn_insert_CLIENT
AFTER INSERT ON CLIENT
FOR EACH ROW
DECLARE
    V CLIENT.VILLECLIENT%TYPE := :NEW.VILLECLIENT;
    NBR_CN NUMBER;
    NBR_CP NUMBER;
BEGIN
    SELECT COUNT(*) INTO NBR_CN FROM compte WHERE noclient = :NEW.NOCLIENT AND solde < 0;
    SELECT COUNT(*) INTO NBR_CP FROM compte WHERE noclient = :NEW.NOCLIENT AND solde > 0;

    IF V = 'Casablanca' AND NBR_CN > 0 THEN
        INSERT INTO CLIENTSITE1@centrale_vers_sitel VALUES (:NEW.NOCLIENT, :NEW.NOMCLIENT,
        :NEW.PRENOMCLIENT, :NEW.VILLECLIENT, :NEW.AGE);
    END IF;

    IF V = 'Rabat' AND NBR_CP > 0 THEN
        INSERT INTO CLIENTSITE2@centrale_vers_site2 VALUES (:NEW.NOCLIENT, :NEW.NOMCLIENT,
        :NEW.PRENOMCLIENT, :NEW.VILLECLIENT, :NEW.AGE);
    END IF;
END;
```

Trigger : syn_update_client :

```
create or replace TRIGGER syn_update_client
BEFORE UPDATE ON client
FOR EACH ROW
DECLARE
    OV client.villeclient%type := :old.villeclient;
    NV client.villeclient%type := :new.villeclient;
    R compte%rowtype;
    n1 number;
    n2 number;

    -- les comptes de client solde < 0
    CURSOR cur1 IS
        SELECT *
        FROM compte
        WHERE Noclient = :new.Noclient AND solde < 0;

    -- les comptes de client solde >= 0
    CURSOR cur2 IS
        SELECT *
        FROM compte
        WHERE Noclient = :new.Noclient AND solde >= 0;
BEGIN
    -- nbr comptes de client solde < 0
    SELECT count(idcompte) INTO n1 FROM compte WHERE Noclient = :new.Noclient AND solde < 0;
```

```

ELSIF OV = 'Rabat' THEN
    IF n2 > 0 THEN
        IF NV = 'Rabat' THEN
            UPDATE clientsite2@centrale_vers_site2
            SET NomClient = :new.NomClient, PrenomClient = :new.PrenomClient, VilleClient = :new.VilleClient, Age = :new.Age
            WHERE Noclient = :new.Noclient;
        ELSE
            DELETE clientsite2@centrale_vers_site2 WHERE Noclient = :new.Noclient;

            IF NV = 'Casablanca' THEN
                IF n1 > 0 THEN
                    INSERT INTO clientsitel@centrale_vers_sitel VALUES (:new.NoClient,:new.NomClient, :new.PrenomClient, :new.VilleClient, :new.Age);
                    FOR n IN curl LOOP
                        R := n;
                        INSERT INTO comptesitel@centrale_vers_sitel VALUES R;
                    END LOOP;
                END IF;
            END IF;
        END IF;
    END IF;
ELSE
    IF n2 > 0 THEN
        IF NV = 'Rabat' THEN
            INSERT INTO clientsite2@centrale_vers_site2 VALUES (:new.NoClient,:new.NomClient, :new.PrenomClient, :new.VilleClient, :new.Age);
            IF n2 > 0 THEN
                SELECT count(idcompte) INTO n1 FROM compte WHERE Noclient = :new.Noclient AND solde < 0;
                -- nbr comptes de client solde <= 0
                SELECT count(idcompte) INTO n2 FROM compte WHERE Noclient = :new.Noclient AND solde >= 0;

                IF OV = 'Casablanca' THEN
                    IF n1 > 0 THEN
                        IF NV = 'Casablanca' THEN
                            UPDATE clientsitel@centrale_vers_sitel
                            SET NomClient = :new.NomClient, PrenomClient = :new.PrenomClient, VilleClient = :new.VilleClient, Age = :new.Age
                            WHERE Noclient = :new.Noclient;
                        ELSE
                            DELETE clientsitel@centrale_vers_sitel WHERE Noclient = :new.Noclient;

                            IF NV = 'Rabat' THEN
                                IF n2 > 0 THEN
                                    INSERT INTO clientsite2@centrale_vers_site2 VALUES (:new.NoClient,:new.NomClient, :new.PrenomClient, :new.VilleClient, :new..
                                    FOR n IN cur2 LOOP
                                        R := n;
                                        INSERT INTO comptesite2@centrale_vers_site2 VALUES R;
                                    END LOOP;
                                END IF;
                            END IF;
                        END IF;
                    END IF;
                ELSE
                    ELSIF OV = 'Rabat' THEN

```



```

IF NV = 'Rabat' THEN
    INSERT INTO clientsite2@centrale_vers_site2 VALUES (:new.NoClient,:new.NomClient, :new.PrenomClient, :new.VilleClient, :new.Age);
    IF n2 > 0 THEN
        FOR n IN cur2 LOOP
            R := n;
            INSERT INTO comptesite2@centrale_vers_site2 VALUES R;
        END LOOP;
    END IF;
END IF;
END IF;
IF n1 > 0 THEN
    IF NV = 'Casablanca' THEN
        INSERT INTO clientsitel@centrale_vers_sitel VALUES (:new.NoClient,:new.NomClient, :new.PrenomClient, :new.VilleClient, :new.Age);
        FOR n IN curl LOOP
            R := n;
            INSERT INTO comptesitel@centrale_vers_sitel VALUES R;
        END LOOP;
    END IF;
END IF;
END IF;
END;

```

Trigger : syn_update_compte

```

create or replace TRIGGER SYC_DELETE_CLIENT
AFTER DELETE ON client
FOR EACH ROW
DECLARE
    OC client.noclient%TYPE := :old.noclient;
    ns1 NUMBER;
    ns2 NUMBER;
BEGIN
    SELECT COUNT(*) INTO ns1
    FROM clientsitel@centrale_vers_sitel
    WHERE noclient = OC;

    SELECT COUNT(*) INTO ns2
    FROM clientsite2@centrale_vers_site2
    WHERE noclient = OC;

    IF ns1 = 1 THEN
        DELETE FROM clientsitel@centrale_vers_sitel WHERE noclient = OC;
    END IF;

    IF ns2 = 1 THEN
        DELETE FROM clientsite2@centrale_vers_site2 WHERE noclient = OC;
    END IF;
END;

```

Trigger : syn_update_compte

```
create or replace trigger sync_update_compte
before update on compte
for each row
begin
declare
OS compte.solde%type:=:old.solde;
NS compte.solde%type:=:new.solde;
V client.villeclient%type;
G client.age%type;
n integer;
R client%rowtype;
begin
select * into R from client where noclient=:new.noclient;
select villeclient into V from client where noclient=:new.noclient;
if(V='Casablanca' AND NS<0) then
update comptesitel@centrale_vers_sitel set solde=NS;
elsif(V='Rabat' AND NS>=0) then
update comptesite2@centrale_vers_site2 set solde=NS;
end if;
end;
end;
```

Trigger : syn_delete_compte

```
create or replace trigger syn_Delete_compte
before delete on compte
for each row
begin
declare
V client.villeclient%type;
S compte.solde%type:=:old.solde;
nb integer;
begin
select cl.villeclient into V
FROM client cl where cl.noclient=:old.noclient;
if (V='Casablanca' AND S<0) then
delete comptesitel@centrale_vers_sitel where idcompte=:old.idcompte;
DBMS_OUTPUT.PUT_LINE('compte est deleted');
elsif (V='Rabat' AND S >= 0) then
delete comptesite2@centrale_vers_site2 where idcompte=:old.idcompte;
DBMS_OUTPUT.PUT_LINE('compte est deleted');
end if;
END ;
END ;
```

Trigger : syn_insert_comptesitel

```
create or replace trigger syn_insert_comptesitel
before insert on comptesitel
for each row
begin
declare
V clientsitel.villeclient%type;
S comptesitel.solde%type:=:new.solde;
v_count NUMBER := 0;
v_idcompte comptesitel.idcompte%TYPE;
CURSOR c_compte IS
SELECT idcompte
FROM comptesitel;
nb integer;
begin
-- Récupérer la nouvelle valeur de idcompte
v_idcompte := :NEW.idcompte;
-- Vérifier si l'idcompte existe déjà
FOR rec IN c_compte LOOP
IF v_idcompte = rec.idcompte THEN
v_count := v_count + 1;
```

```

END IF;
END LOOP;

-- Si l'idcompte existe déjà, lever une exception
IF v_count > 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'La clé primaire (idcompte) existe déjà dans la table "comptesitel".');
    ROLLBACK;
END IF;

```

```

select cl.villeclient into V
FROM clientsitel cl where cl.noclient=:new.noclient;
if (V ='Casablanca') THEN
    IF ($<0) then
        select count(noclient) into nb FROM clientsitel
        where noclient=:new.noclient;
        if nb=0 then
            DBMS_OUTPUT.PUT_LINE('idclient n existe pas');
            ROLLBACK;
        ELSE

```

```

            DBMS_OUTPUT.PUT_LINE('compte est bien inserée');
            insert into compte@sitel_vers_centrale values (:new.idcompte, :new.Ncompte,
: new.solde,:new.noclient, :new.codeagence);
            end if;

        ELSE
            DBMS_OUTPUT.PUT_LINE('le solde est pas négatif !!!');
            ROLLBACK;
        end if;
    else
        DBMS_OUTPUT.PUT_LINE('la ville client n est Casablanca!!!');
        ROLLBACK;
    END IF;
END ;
END ;

```

Triggers de site1 :



Trigger : syn_delete_clientDIBLC :

```

create or replace TRIGGER SYNC_DELETE_CLIENTDIBLC
AFTER DELETE ON clientsitel
FOR EACH ROW
DECLARE
    OC clientsitel.noclient%TYPE := :old.noclient;
    nsl NUMBER;
BEGIN
    DELETE FROM clientsitel@site1_vers_site2 WHERE noclient = OC;
END;
```

Trigger : syn_insert_clientdibl :

```
create or replace TRIGGER syn_insert_CLIENTDB
AFTER INSERT ON CLIENTsitel
FOR EACH ROW
DECLARE
    .

BEGIN

    INSERT INTO CLIENTSITE1@sitel_vers_site2 VALUES (:NEW.NOCLIENT, :NEW.NOMCLIENT,
    :NEW.PRENOMCLIENT, :NEW.VILLECLIENT, :NEW.AGE);

END;
```

Trigger : syn_iinsert_compteDB :

```
create or replace TRIGGER syn_insert_CompteDB
after insert ON comptesitel
FOR EACH ROW
DECLARE

BEGIN

    insert into comptesitel@sitel_vers_site2 values (:new.idcompte, :new.Ncompte,
    :new.solde, :new.noclient, :new.codeagence);

END;
```

Trigger : syn_update_compteDB :

```
create or replace TRIGGER syn_update_CompteDB
after UPDATE ON comptesitel
FOR EACH ROW
DECLARE

BEGIN

    UPDATE comptesitel@sitel_vers_site2
    SET Idcompte =:new.Idcompte,NCompte =:new.NCompte,solde =:new.solde,Noclient =:new.Noclient ,CodeAgence =:new.CodeAgence

    WHERE Idcompte = :new.Idcompte;

END;
```

Trigger : syn_update_clientDB :

```
create or replace TRIGGER syn_update_clientDB
after UPDATE ON clientsitel
FOR EACH ROW
DECLARE

BEGIN

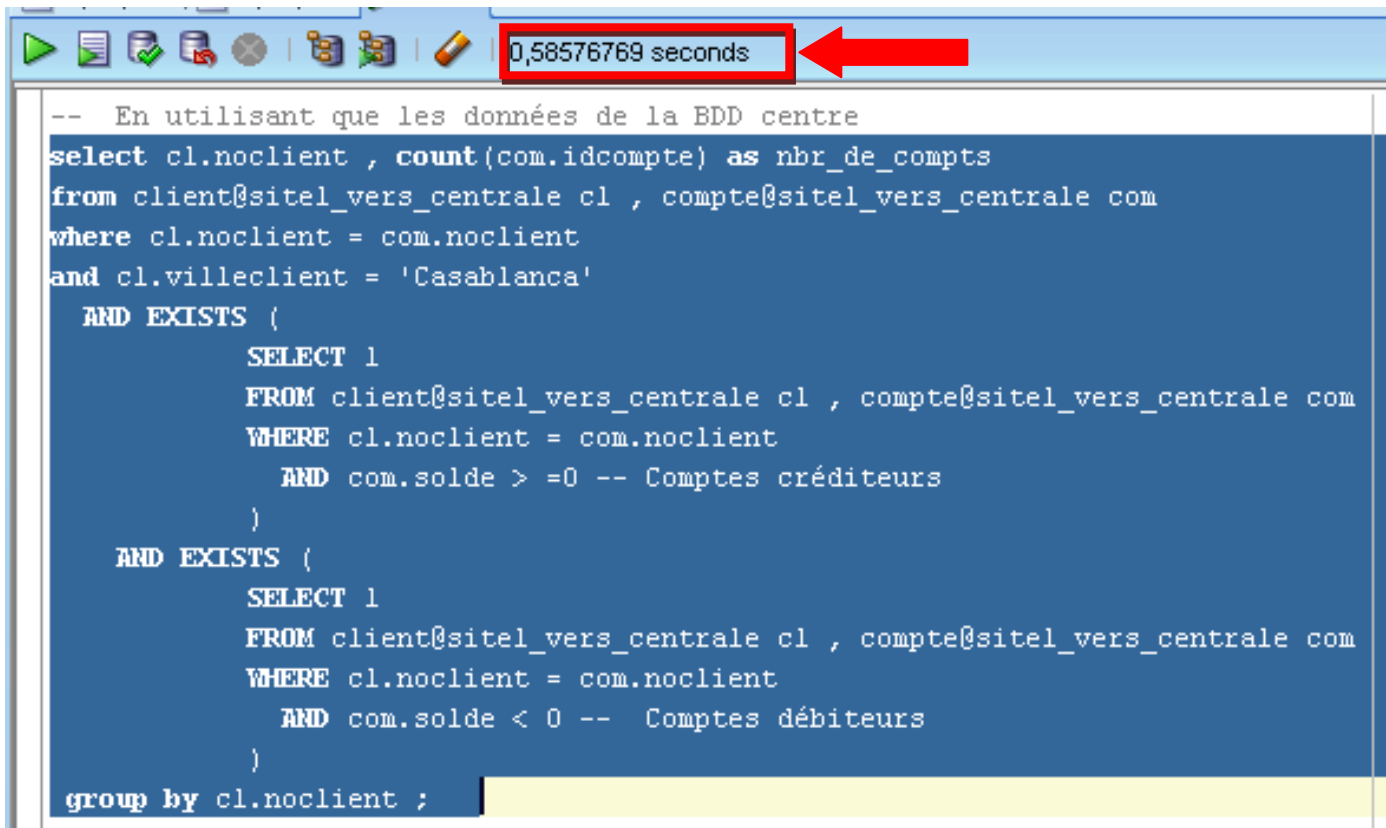
    UPDATE clientsitel@sitel_vers_site2
    SET NomClient = :new.NomClient, PrenomClient = :new.PrenomClient, VilleClient = :new.VilleClient, Age = :new.Age
    WHERE Noclient = :new.Noclient;

END;
```

8)- Les Requêtes réparties :

L'utilisateur du site1(*UI*) veut connaître la liste des clients de 'Casablanca' qui possèdent des comptes débiteurs et créditeurs. On doit créer cette requête en Procédant de deux manières :

- En utilisant que les données de la BDD centre :



```

-- En utilisant que les données de la BDD centre
select cl.noclient , count(com.idcompte) as nbr_de_compts
from client@sitel_vers_centrale cl , compte@sitel_vers_centrale com
where cl.noclient = com.noclient
and cl.villeclient = 'Casablanca'
  AND EXISTS (
    SELECT 1
    FROM client@sitel_vers_centrale cl , compte@sitel_vers_centrale com
    WHERE cl.noclient = com.noclient
      AND com.solde > =0 -- Comptes créditeurs
  )
  AND EXISTS (
    SELECT 1
    FROM client@sitel_vers_centrale cl , compte@sitel_vers_centrale com
    WHERE cl.noclient = com.noclient
      AND com.solde < 0 -- Comptes débiteurs
  )
group by cl.noclient ;
    
```

- En utilisant une jointure entre les données stockées dans la BDD centre et celle stockées dans la BDD site-1



The screenshot shows a SQL query execution window. At the top, a status bar displays the execution time: 0,56273293 seconds, which is highlighted with a red box and a red arrow. Below the status bar, the SQL query is displayed in a blue background. The query is a complex join between two tables, 'clientsitel' and 'comptesitel', with a subquery in the 'where' clause. The query is as follows:

```
-- En utilisant une jointure entre les données stockées dans la BDD centre et cell
select cl.noclient , count(com.idcompte) as nbr_de_compts
from clientsitel cl , comptesitel com
where cl.noclient = com.noclient
and cl.noclient in
(
    SELECT cl.noclient
    FROM client@sitel_vers_centrale cl , compte@sitel_vers_centrale com
    WHERE cl.noclient = com.noclient
    AND cl.villeclient = 'Casablanca'
    AND com.solde > =0 -- Comptes créditeurs
)
group by cl.noclient;
```

Cependant, en général, la requête utilisant uniquement les données de la base de données "centre" peut être plus rapide puisqu'elle évite la nécessité de la jointure entre les deux bases de données, ce qui peut entraîner une latence supplémentaire. Si les données nécessaires se trouvent uniquement dans la base de données "centre", il est préférable d'utiliser cette approche pour optimiser les performances.