

## TP Bases de Données Réparties sur Oracle

**Rim Moussa**  
**Univ. 7 Nov. à Carthage**

### Read me

Il est possible de faire le TP sur le même ordinateur, et tester toutes les fonctionnalités. Créer une nouvelle BD en passant par *OracleHome ...>Database Administration>Oracle DB Config Assistant*.

Pour consulter la nouvelle BD dans *Oracle DBA Studio* → Menu : *File>Add database to tree*.

Environnement de Travail

*OracleHome ...>Database Administration>SQL\*Plus Worksheet*

*OracleHome...>Database Administration>Oracle DBA Studio*.

Pour chaque instance de BD, ouvrez un *SQL\*Plus Worksheet*.

Si ce message s'affiche '*oracle not available*', connectez-vous à l'instance avec le compte *system/manager* connect as *SYSDBA*, et tapez *STARTUP*; ou bien allez dans *Oracle DBA Studio*, et cliquez sur *instance*, puis *database* et sélectionner *open*. Le fichier d'initialisation est *C:\OraHome1\DATABASE\initBDTest.ora*.

A chaque fois que vous passez par *Oracle DBA studio* pour créer un objet ... cliquez sur le tab *SHOW SQL*, pour voir le code SQL.

### 1 Création d'un Nouveau Compte

Se connecter avec le compte *sys/change\_on\_install* ou *system/manager*; Créer un compte *user1/passwd1* et un compte *user2/passwd2*, respectivement sur S1 et S2, et leur accorder tous les privilèges

```
CREATE USER user1 IDENTIFIED BY passwd1;  
GRANT ALL PRIVILEGES TO user1;
```

```
-- se déconnecter: disconnect;
```

```
-- se connecter dorénavant sur les deux serveurs avec les nouveaux comptes créés
```

### 2 Création d'un database link

\* Affichage de paramètres

```
SELECT name, value          FROM v$parameter
```

WHERE name IN ('db\_name', 'db\_domain', 'service\_names', 'instance\_name', 'global\_names');

Ces paramètres sont visibles du fichier *init.ora* de nom de l'instance

```
db_name = "test"
db_domain = rim.tn
instance_name = test
service_names = test.rim.tn
```

Pour pouvoir nommer librement les database links, exécuter à chaque fois que vous lancez *SQL\*Plus Worksheet* : `ALTER system SET global_names = false ;`

\* Jeter un coup d'œil sur les fichiers de configuration réseau *tnsnames.ora* (client, càd S1) et *listener.ora* (serveur, càd S2). Les se trouvent dans `\orahome1\network\admin\`.

Passer par *Net8 Assistant* : pour ce cliquer sur *Service Naming* et le + (de couleur verte à votre gauche). L'ajout d'un nouveau *nom service réseau* (ang. *net service name*) modifie le fichier

\* Créer un database link : *lienBD\_S2*

```
CREATE PUBLIC DATABASE LINK lienBD_S2
CONNECT TO user2 IDENTIFIED BY passwd2
USING 'hellobd.sami.tn'; --'hellobd.sami.tn' est le nom de service réseau.
```

Vous pouvez également passer par *Oracle DBA Studio*, dans le schéma de *user1*, cliquer avec la droite de votre souris sur *database links*, et *create*. Il est à remarquer que si vous déclarez le lien public, vous le retrouverez dans le schéma de *public*.

\* Tester le lien

Afficher la date de S2 : `SELECT sysdate FROM dual@lienBD_S2;`

### 3 Création d'un jeu de données

\* Sur le Serveur 1, Créer une table *ClientsTunis*, *CommandesTunis*. De la même manière créer les tables *ClientsBizerte* et *CommandesBizerte* sur le Serveur 2.

<pre>CREATE TABLE ClientsTunis (   num NUMBER(4) PRIMARY KEY,   nom VARCHAR2(20),   prenom VARCHAR2(20),   adresse VARCHAR2(50),   ville VARCHAR2(15) NOT NULL, --attr de ('o','n')) ; partitionnement   CA NUMBER(10,3), --chiffre d'affaires   TR NUMBER(4,2) -- Taux de réduction );</pre>	<pre>CREATE TABLE CommandesTunis (   num NUMBER(4) PRIMARY KEY,   datec DATE,   numClt NUMBER(4) NOT NULL,   livraison CHAR(1) CHECK (livraison IN   COMMIT;</pre>
---	--

## 4 Créer des Synonymes

Créer sur le serveur S1 des synonymes aux tables hébergées sur le serveur 2, afin de cacher la localisation des tables distantes (vous pouvez le faire sur *Oracle DBA Studio*, les liens sont disponibles dans un combo box).

```
CREATE PUBLIC SYNONYM ClientsBizerte FOR ClientsBizerte@lienBD_S2;  
CREATE PUBLIC SYNONYM CommandesBizerte FOR CommandesBizerte@lienBD_S2;
```

## 5 Transparence pr à la Fragmentation

\* Vues

Créer une vue sur le serveur 1 qui fait l'union des tables clients, et une qui fait l'union des tables commandes.

```
CREATE VIEW Clients AS (SELECT * FROM ClientsTunis)  
UNION (SELECT * FROM ClientsBizerte);
```

\* Procédures (Oracle DBA Studio>schema>user1>procedures)

Ecrire une procédure stockée qui insère un nouveau client, et une seconde qui insère une nouvelle commande (sur les 2 serveurs).

```
CREATE OR REPLACE PROCEDURE "USER1"."INSERERCLIENT" (cnum IN NUMBER, cnom IN  
VARCHAR, cprenom IN VARCHAR, cadr IN VARCHAR, cville IN VARCHAR, cca IN NUMBER, ctr  
IN NUMBER)AS  
BEGIN  
    IF (cville = 'tunis') THEN INSERT INTO ClientsTunis VALUES (cnum, cnom, cprenom, cadr,  
cville, cca, ctr);  
    ELSIF (cville = 'bizerte') THEN INSERT INTO ClientsBizerte VALUES (cnum, cnom, cprenom,  
cadr, cville, cca, ctr);  
    ELSE DBMS_OUTPUT.PUT_LINE('La ville du client est soit tunis ou bizerte!');  
    END IF;  
    COMMIT;  
END;
```

--à ameliorer

```
CREATE OR REPLACE PROCEDURE "USER1"."INSERERCOMMANDE" (cnum IN NUMBER,  
cdate IN DATE, cnumclt IN NUMBER, cliv IN CHAR)AS  
clt NUMBER;  
BEGIN  
    clt := 0;  
    -- chercher un client avec le numero cnumclt dans la table ClientsTunis  
    SELECT num INTO clt FROM ClientsTunis WHERE (num = cnumclt);  
    IF SQL%FOUND THEN  
        INSERT INTO CommandesTunis VALUES (cnum, cdate, cnumclt, cliv);  
        COMMIT;  
    END IF;
```

```
EXCEPTION
WHEN NO_DATA_FOUND THEN
    -- chercher un client avec le numero cnumclt dans la table ClientsBizerte
    SELECT num INTO clt FROM ClientsBizerte WHERE (num = cnumclt);
    IF SQL%FOUND THEN
        INSERT INTO CommandesBizerte VALUES (cnum, cdate, cnumclt, cliv);
        COMMIT;
    END IF;
END;
END;
```

Que pensez-vous de la complexité d'insertion d'un tuple dans une table DHP-ed ? Suggérez un autre attribut de partitionnement pour la relation clients pour faciliter les insertions dans commandes?

Peupler les tables clients d'abord puis les tables commandes des 2 serveurs à partir du serveur 1.

Pour exécuter une procédure : EXECUTE nom\_procedure (arguments) ; (EXECUTE ou EXEC)

```
EXECUTE insererclient(1, 'moussa', 'rim', '14.rue 20', 'bizerte', 1000, 5);
EXECUTE inserercommande(1, sysdate-100, 1, 'o');
```

Vérifier vos insertions en interrogeant la vue *clients*: SELECT num, nom, prenom, ville FROM clients ;

Et également la table *ClientsBizerte* distante : SELECT num, nom, prenom, ville FROM *ClientsBizerte*; (remarquez que vous utilisez le synonyme de *ClientsBizerte*@ lienBD\_S2.

Merci de tenir compte des suggestions suivantes, pour répondre rapidement aux requêtes.

- Insérer des clients sur le site de Tunis et sur le site de Bizerte qui ont le même nom de famille.
- Insérer des clients, dont le CA est supérieure à 2000.

## 6 Plan d'exécution d'une Requête

SET TIMING ON ; --permet d'afficher le temps d'exécution des requêtes

ANALYZE TABLE nom\_table COMPUTE STATISTICS; -- ne peut pas s'exécuter sur une table distante!

Commentez brièvement le contenu des scripts suivants: *utlxplan.sql* et *utlxpls.sql*.

Afin d'exécuter *utlxplan.sql* tapez: @c:\orahome1\RDBMS\ADMIN\utlxplan

Regardez le poly p. 44 pour déterminer les requêtes qui se sont exécutées sur le site distant (démarche : utlxplan, EXPLAIN PLAN FOR requete, utlxpls, interroger plan\_table : SELECT other FROM plan\_table WHERE operation = 'REMOTE');

## 7 Requetes

Analyser (et joindre au compte rendu) le plan d'exécution de chacune des requêtes suivantes, en commentant l'exécution de la requête : (Transfert de données ? Transfert de requêtes ?)

Toutes les requêtes sont émises à partir du serveur 1.  
(utilisez les synonymes pour répondre aux requêtes R1, R2, R3)

**(R1)** Afficher les clients bizertins et leurs commandes,

**(R2)** Afficher les clients de Bizerte et de Tunis qui ont le même nom de famille.

Que pensez-vous des performances en cas d'une table *ClientsBizerte* volumineuse, que suggériez-vous ? (indic. Use de DRIVING\_SITE, ré-afficher le plan d'exécution pour vérifier le plan d'exécution)

```
SELECT /*+DRIVING_SITE(cb)*/ ct.num, ct.prenom, ct.nom, cb.num, cb.prenom
FROM clientstunis ct, clientsbizerte cb
WHERE ct.nom=cb.nom;
```

**(R3)** Afficher les commandes de clients bizertins non encore livrées.

**(R4)** Afficher les clients tunisois dont le CA est > 2000 (à exécuter sur la vue)

## 9 Réplication

Les *Materialized Views* et les *MV logs* peuvent être créées de *Oracle...>Database Administration>Oracle DBA Studio>schema>user1>*

En ce qui concerne les *Snapshots* voir *Oracle...>Extended Administration>Replication Manager*.

Merci de joindre à votre compte rendu le script SQL de création du *snapshot* ou de la *MV*. Pour cela si vous utilisez *Oracle DBA Studio* ou *Replication Manager*, cliquer sur le tab *SHOW SQL*.

### 8.1 COPY TO

Retourner à SQL\*Plus Worksheet ☺

Utiliser la commande *COPY TO* pour copier sur le serveur S1 les clients de Bizerte qui ont un CA > 2000 sur une nouvelle table *ClientsBizertins\_CA\_sup\_2K*. Puis, interroger la nouvelle table créée.

### 8.2 SNAPSHOT

Créer un snapshot (éventuellement un snapshot log) sur le serveur 1, qui instancie les requêtes concernant le serveur S2 suivantes :

(R1) Nombre de commandes de clients bizertins par client.

Choisissez toutes les 5 minutes et FAST REFRESH, est-ce possible ?

Ajouter une commande à un client bizertin et afficher le contenu du snapshot, que remarquez-vous ?

(R2) Les clients Bizertins (dont le CA est > 2000) et leurs commandes non livrées.

Choisissez toutes les 5 minutes et FAST REFRESH, est-ce possible ? pourquoi ?