TP Jointures réparties

mars 2023

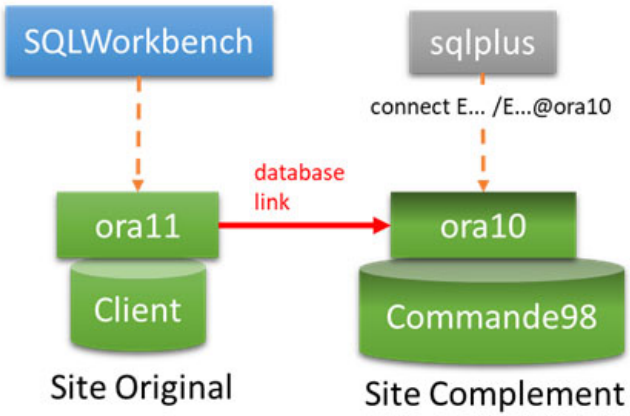
Objectifs:

L'objectif de ce TME est de comprendre l'évaluation d'une requête de jointure entre 2 relations qui sont situées sur deux sites distincts.

- Définir le schéma global qui offre un accès transparent à des données de plusieurs bases,
- Formuler une requête répartie,
- Comprendre l'ordre et l'emplacement des opérations permettant d'évaluer une requête répartie : savoir expliquer quel site traite quelles opérations.

Préparation

On dispose de 2 SGBD sur les sites nommés SiteOriginal et SiteComplement.



Pour les distinguer plus facilement, on accède à chaque site avec un client différent :

- Le site Original est celui que vous utilisez habituellement. Accès par le client SQLWorkbench.
- Le site Complémentaire : accès par le client textuel sqlplus depuis un terminal.

Si vous travaillez sur votre ordinateur perso, vous pouvez accéder au client sqlplus à partir de la passerelle de l'université. Pour cela, vous ouvrez une fenêtre de terminal (ou un powershell) et vous vous connectez à la passerelle avec la commande :

ssh votre_login@ssh.ufr-info-p6.jussieu.fr avec votre login sur la passerelle étant votre numéro d'étudiant par le vôtre. Ensuite il ne vous reste plus qu'à saisir les 2 commandes indiquées ci-dessous.

Sur le site Complement (via le terminal sur ora10)

Ouvrir un terminal et saisir

```
source /Infos/bd/config11
```

sqlplus Enumero/Enumero@ora10 (remplacer le numéro Enuméro par E suivi de votre numéro étudiant)

Vous obtenez une invite SQL> vous permettant de saisir des commandes SQL.

Création d'un **lien** entre les bases. Le lien permet au site Complement de lire les données du site Original.

Nom du lien : **siteOriginal.fr**

```
drop database link siteOriginal.fr ;   -- pour supprimer un lien si nécessaire

CREATE DATABASE LINK siteOriginal.fr CONNECT TO E1234567 IDENTIFIED BY "E1234567"
USING 'ora11';        -- (remplacer le numéro par le vôtre)
```

Vérifier que le lien fonctionne :

```
select count(*) from Orders@siteOriginal.fr
select count(*) from Orders@siteOriginal.fr
ou
desc Orders@siteOriginal.fr
desc Commande@siteOriginal.fr
```

Créer la table Commande98 :

```
create table Commande98 (
   numCde        Number       not null,
   numClient     Number       not null,
   etat          varchar(1)   not null,
   prixC         Number(15,2) not null,
   dateC         Date         not null,
   priorite      Char(15)     not null,
   vendeur       Char(15)     not null,
   commentaire   Varchar2(100)
 );
```

Remplir la table Commande98 avec les données de juillet 98 :

```
insert into Commande98 (
 select
   o_orderkey as numCde,
   o_custkey  as numClient,
   o_orderstatus as etat,
   o_totalprice as prixC,
   o_orderdate as dateC,
   o_orderpriority as priorite,
   o_clerk as vendeur,
   o_comment as commentaire
 from Orders@siteOriginal.fr
 where extract(year from o_orderdate) = 1998
 and extract(month from o_orderdate) = 7
);
```

commit; -- pour valider les données insérées

Vérifier que vous pouvez lire Commande98 localement

```
select *
from Commande98
where vendeur like '%001';
```

Tester une requête de jointure **répartie** entre Commande98 (locale sur le site Complement) et Client@siteOriginal (table distante car elle est située sur le site original)

```
select *
from Commande98 c, Client@SiteOriginal.fr co
where c.numClient = co.numClient
and c.vendeur like '%001' and;
```

Sur le site Original (via sqlWorkbench sur ora11)

Créer un lien qui permet au site *Original* de lire les données du site *Complement* :
Nom du lien : **siteComplement.fr**

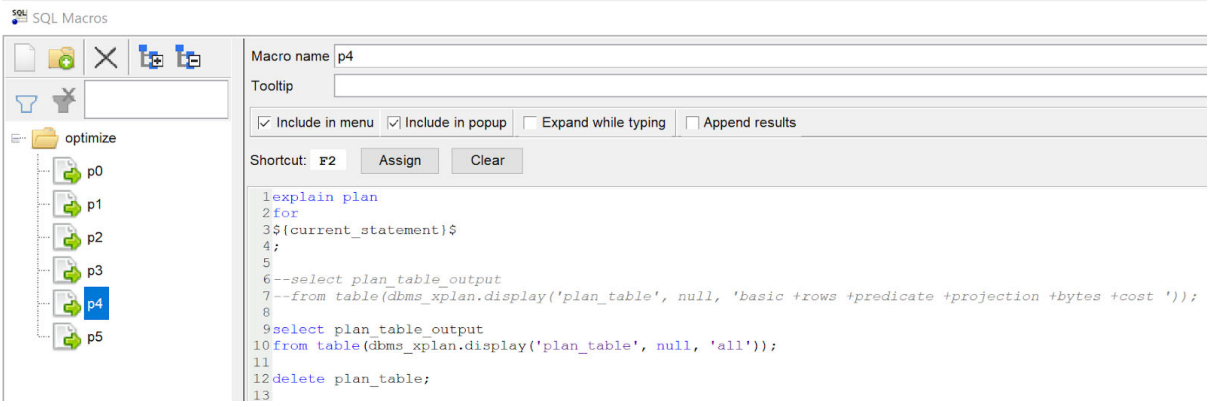
```
-- drop database link siteComplement.fr;    -- si nécessaire

CREATE DATABASE link siteComplement.fr CONNECT TO E1234567 IDENTIFIED BY "Enn"
USING 'ora10';          -- (remplacer Enn par E suivi de votre numéro)
```

Vérifier que le lien fonctionne : remarque, la commande *desc* n'est pas supportée par le client SQLWorkbench donc au on pose une requête pour vérifier que le lien fonctionne : select count(*) from Commande98@siteComplement.fr;

Poser une requête de jointure **répartie** entre la table Client (locale sur le site original) et la table Commande98 (distante sur le site Complement)

Afficher le plan de chaque requête. Repérer les opérateurs REMOTE et expliquer quelles sont les données échangées entre les sites et quelles sont les sous-requêtes traitées sur le site distant.

Pour cela aller dans (Macro → Manage Macro) et modifier la macro p4 et remplacer le dernier paramètre de la fonction display *'basic …. +cost'* par le mot clé **'all'**



Par la suite, quand vous afficherez le plan d'une requête, vous pourrez lire la partie « Remote SQL Information » qui indique quelle est **la sous-requête SQL posée sur le site distant.**

Requêtes réparties

On demande d'expliquer le plus précisément possible le traitement des requêtes suivantes.

Exercice 1 : Requête traitée entièrement sur un site distant

R1:

```
select *
from Commande98@siteComplement.fr c1
where c1.prixC = 5000;
```

Réponse : le plan est:

```
SELECT Statement REMOTE
  TABLE ACCESS FULL
```

La requête R1 ne concerne que des données distantes : donc toute la requête est déléguée au site distant qui se charge de la traiter. Puis le résultat est retourné vers le site Original.
Avantage : cela évite de transférer toutes les Commandes vers le site Original.

Exercice 2 : Jointure locale utilisant des données distantes

S'il n'est pas déjà défini, ajouter un index sur Client(numClient), puis expliquer R2 :

R2 : jointure NL

```
select /*+ USE_NL(c1 c) ordered */    *
from Commande98@siteComplement.fr c1, Client c
where c.numClient = c1.numClient;
```

R3: jointure Hash

```
select /*+ use_hash(c1 c)*/ *
from Commande98@siteComplement.fr c1, Client c
where c.numClient = c1.numClient;
```

R4: : Montrer que la sélection du prixC est « déléguée » au site distant :

```
select /*+ use_hash(c1 c)*/ *
from
Commande98@siteComplement.fr c1, Client c
where c.numClient = c1.numClient
AND c1.prixC= 1000;
```

Exercice 3 : Jointure entre une « petite table » locale et une « grande table » distante

R5:
Jointure sur toutes les commandes de 98 et le client dont le nom se termine par '001'

```
select *
from Commande98@siteComplement.fr c1, Client c
where c.numClient = c1.numClient
and c.nom like  '%001';
```

→ toutes les données de Commande 98 sont envoyées, ce n'est pas optimal en termes de quantité de données transférées.

La directive ORDERED

R6 : directive ORDERED

```
select /*+ ordered*/    *
from Client c, Commande98@siteComplement.fr c1
where c.numClient = c1.numClient and c.nom like '%001';
```

La directive DRIVING_SITE(…) pour contrôler quel site traite la jointure

R7 :

```
select  /*+   DRIVING_SITE( c1) */   *
from Commande98@siteComplement.fr c1, Client c
where c.numClient = c1.numClient and c.nom like '%001';
```

La jointure est traitée à distance sur le site complément puis le résultat est renvoyé un site original.
Le site distant traite lui-même une requête répartie car il doit obtenir les clients se trouvant sur le site Original qui est considéré comme étant « à distance » pour le site Complément.

Réponse attendue : Détailler le plan afin de préciser les transferts de données entre les sites

Exercice 4 : Réduire la quantité de données transférées

La requête globale étudiée est :

```
Select c.profile, co.commentaires
From Client cl, Commande98 co
Where cl.numClient = co.numClient
and cl.numPays = 13
```

On cherche à réduire les transferts des tuples de Clients qui sont supposés être très volumineux. On commence par augmenter la taille de chaque tuple à lire dans la table client, en ajoutant l'attribut *profile* qui est une chaine de 4000 caractères.

alter table Client add **profile** varchar2(4000);

On remplit le champ *profile* pour les clients sélectionnés dans la requête

update Client

```
set profile = lpad('hello', 4000, '*')
where numPays=13;
commit;
```

Proposer une solution qui réduise encore au maximum les données transférées entre les deux sites. L'objectif est de transférer vers le site Complément les valeurs de l'attribut *numClient* pour les seuls clients dont le nom satisfait la condition de la requête. Les autres attributs de Client ne sont pas transférés.
Indications :

Définir dans le siteComplement, la vue Commande13 qui sélectionne les Commandes des clients dont le pays est 13.

Définir dans le siteOriginal la requête R8 qui effectue la jointure entre Client et la vue distante Commande13.

Exercice 5 : Vue globale

Dans le site Original définir une vue nommée VueCommande98 correspondant aux commandes situées à distance sur le site Complément.
Cette vue permet de « masquer » la localisation des données.

Proposer une requête répartie qui ne mentionne pas explicitement de nom de site (c'est-à-dire, une requête sans @ dans la clause from). En particulier, proposer la jointure entre la vue VueCommande98 et la table Client. Expliquer le plan obtenu.

Exercice 6 : Durée des transferts

On s'intéresse à la durée des transferts. Afin d'obtenir des durées significatives observables (de l'ordre de la seconde), on chronomètre les transferts générés par plusieurs exécutions successives d'une requête : on définit une procédure contenant une boucle pour exécuter 20 fois une requête.

De plus, pour mesurer principalement les transferts de données entre les sites et non la durée d'affichage du résultat, on modifie légèrement les requêtes : le résultat d'une requête est collecté dans un tableau (avec l'instruction bulk collect) sans être affiché. Le temps d'affichage devient négligeable face à la durée des transferts de données entre les sites.

On crée les types suivants pour collecter le résultat de la requête :

```
create type RES1 as object (
  com varchar2(4000),
  profile varchar2(4000)
);
/

create type LRES as table of res1;
/
```

Comparer 3 exécutions différentes de la requête.

Plan1 : jointure sur le site original

```
create or replace procedure test1 as
  res LRES;
BEGIN
  FOR i IN 1 .. 20 LOOP
     SELECT RES1 (c.profile, c1.commentaire)
     bulk collect INTO res
     FROM Client c, Commande98@siteComplement.fr c1
     where c.numClient = c1.numClient
     and c.numPays = 13;
  END LOOP;
END;
/

exec test1;
```

Plan 2 : jointure sur le siteComplément

Plan 3 : semi jointure sur le siteComplément pour connaitre les commandes13 puis jointure avec les clients sur le site original.

Quelle exécution est la plus rapide ? Expliquer l'écart de durée. Proposer une formule pour estimer approximativement cette durée à partir de la taille des données.

Exercice 7 Fragmentation

a) Proposer un schema global tel que les clients sont fragmentés en deux parties : les clients ayant du numPays dans [1,10] sont sur le site Original. Les autres clients sont sur le site Complement.

b) Définir dans le site original une vue VueClient réunissant les deux fragments. Indication : utiliser l'opérateur UNION.

c) Expliquer le plan de la requête affichant les clients du segment AUTOMOBILE.

d) Expliquer le plan d'une requête de jointure entre VueCLient et Commande98.

Diverses questions et erreurs rencontrées

1) Est-ce que la commande **desc** permet de décrire une table distante ? Non pas si vous utilisez le client SQLWorkbench

2) "Je rencontre cette erreur" : Error at line 2: ORA-01882: **timezone region not found** : C'est lié à un pb de date sur votre ordinateur personnel. Résolution : Utiliser le client textuel (sqplplus) pour accéder aux 2 bases.