HMIN328 Bases de données réparties : Tp Oracle Db Link

1. Préambule

Les bases de données réparties répondent, notamment, aux besoins des entreprises multi-sites qui ont à gérer des données provenant de leurs différents sites. Dans un SGBD réparti, la base de données apparaît comme une base de données centralisée pour l'utilisateur, les données et les traitements sont cependant distribués sur plusieurs serveurs de données (qui gèrent chacun les données locales et qui participent aux transactions globales), reliés par un réseau. Les données peuvent être répliquées sur plusieurs sites pour en assurer une meilleure disponibilité. La répartition des données doit s'opérer de manière à garantir la performance des accès aux données (équilibrer la charge du système en répartissant les traitements sur les différents serveurs, rendre efficace la réponse aux requêtes, etc). La conception d'une base de données répartie peut être soit ascendante (à partir des schémas locaux pour produire un schéma global), soit descendante (à partir d'un schéma global pour produire des schémas locaux).
Oracle propose, un mécanisme traitant de la décomposition d'un schéma global (approche top-down) en différents sous-schémas sous-jacents (fragmentation horizontale comme verticale ou encore hybride), ou à l'inverse traitant de l'intégration de différents schémas au travers de vues dites intégrantes (approche bottom-up). Ce mécanisme repose en grande partie sur la notion de *database link*, qui va faciliter la construction et la manipulation de bases de données réparties.
Le travail pratique proposé s'inscrit dans un premier temps dans une démarche d'intégration de schémas de bases de données existants, puis dans un second temps, dans une démarche de décomposition.
Un lien de bases de données (database link) va permettre à partir d'une base de données (ou d'un schéma utilisateur de cette base de données) d'accéder très simplement à des tables, et à leurs contenus, provenant d'autres bases de données (locales ou à distance). Les liens de bases de données peuvent dépasser le cadre d'Oracle et mettre en relation des bases de données Oracle avec des bases de données relationnelles gérées sous d'autres SGBDs (à l'exemple de Mysql ou de Postgresql au travers d'un pont ODBC) pour mettre en place des serveurs de données dits hétérogènes.
Différents types de "database link" sont disponibles, selon si le lien est public, privé ou partagé, ou encore, si le lien porte sur l'ensemble de la base de données (global) ou sur un schéma utilisateur en particulier. Un lien reste cependant unidirectionnel dans tous les cas de figure. Des exemples de création de lien sont donnés :

— lien privé portant sur un schéma utilisateur (nomSchema) d'une base de données (nomService)
```
create database link nomDuLien connect to nomSchema identified by motPasseSchema
using 'nomService';
```
— lien public défini globalement sur la base de données
```
create public database link nomDuLien
using 'nomService';
```
— lien public portant sur le schéma utilisateur interne user1 et la BD master
```
create public database link to_user1_master connect to user1 identified by user1
using 'master';
```

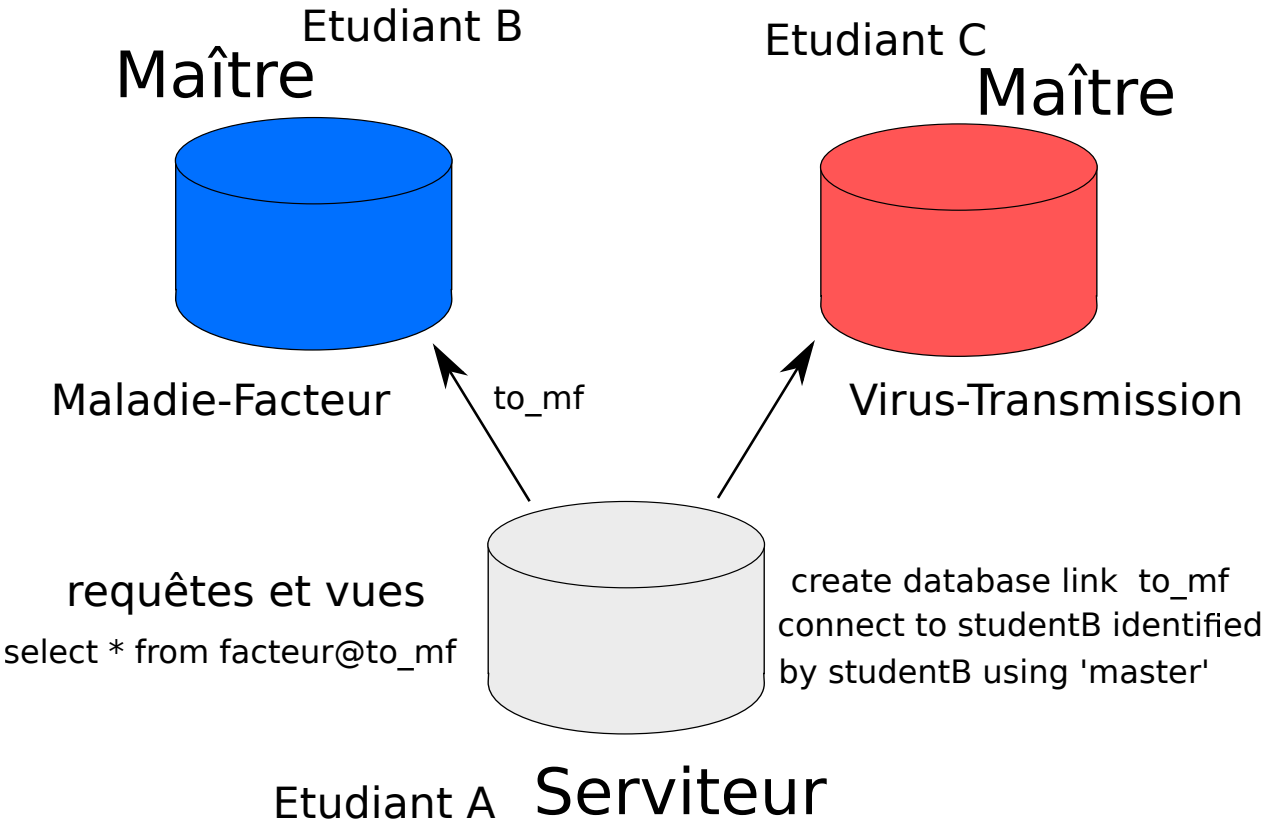

FIGURE 1 – Architecture étudiée dans le Tp

— lien privé portant sur le schéma utilisateur interne user1 et la BD licence
`create database link me_to_user1_licence connect to user1 identified by user1 using 'licence';`

Des vues du méta-schéma sont également consultables

— vues statiques user_db_links, all_db_links et dba_db_links
`ex : select owner, db_link, username from dba_db_links`
— vue dynamique v$dblink : renseigne sur les liens ouverts dans le contexte de la transaction en cours

Les requêtes se font simplement en rajoutant le nom du lien après la table ou la vue interrogée.

— consulter un schéma à distance via un lien (exemple monLien qui pointe sur la BD licence)
`select table_name from user_tables@to_user1_master ;`

Vous allez tout particulièrement travailler avec des liens (privés ou publiques) portant sur des schémas utilisateurs. Vous choisirez de travailler à trois (pour avoir trois schémas utilisateur). Nous allons considérer la situation illustrée dans la figure ci-dessous : 3 schémas utilisateur dont deux sont définis au sein de la même base de données *master* hébergée sous le serveur *venus* et un schéma utilisateur défini au sein de la base de données *licence* hébergée également sous le serveur *venus*.

L'idée est de manipuler deux schémas utilisateur en tant que schémas Maître (hébergés chacun sur une base de données locale) et un schéma serviteur qui hébergera essentiellement des vues construites à partir de liens portant sur les schémas maîtres. Vous travaillerez en trinôme ((étudiant A, étudiant B et étudiant C).

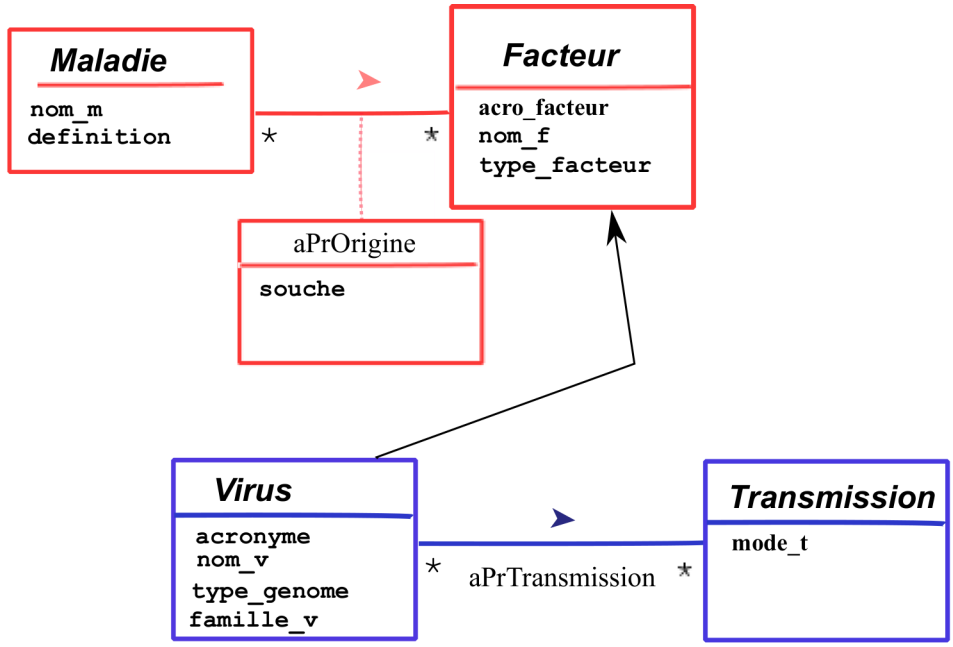

FIGURE 2 – Schéma global comme vue intégratrice des schémas locaux

2. Approche ascendante (global as view)

Le schéma utilisateur Serviteur (étudiant A) sur la BD licence ne possèdera, donc dans un premier temps, aucune donnée en propre et exploitera les données structurées au sein des tables des schémas Maître (étudiant B et étudiant C) sur la base de données master (ou éventuellement licence). A cet effet, vous implanterez deux schémas relationnels : Maladie-Facteur au sein du schéma utilisateur Maître de l'étudiant A et Virus-Transmission au sein du schéma utilisateur de l'étudiant B. Les scripts de création et d'alimentation des tables vous sont donnés.

2.0.1 Schéma Maladie-Facteur

Maladie (**nom_m** varchar(20), definition varchar(20))
Facteur (**acro_facteur** varchar(16), nom_f varchar(26), type_facteur varchar(15))
aprOrigine (**nom_m** varchar(20), **acro_facteur** varchar(16), **souche** varchar(5))
avec aprOrigine (nom_m) $\sqsubseteq$ Maladie(nom_m) et aprOrigine (acro_facteur) $\sqsubseteq$ Facteur(acro_facteur)

2.0.2 Schéma Virus-Transmission

Virus (**acronyme** varchar(5), nom_v varchar(20), type_genome varchar(3), famille_v varchar(18))
Transmission (**mode_t** varchar(20))
aPrTransmission (**acronyme** varchar(5), **mode_t** varchar(20))
avec aPrTransmission(acronyme) $\sqsubseteq$ Virus(acronyme) et aPrTransmission(mode_t) $\sqsubseteq$ Transmission(mode_t)

Vous créerez deux liens de bases de données à partir du compte Serviteur sur licence, et pointant respectivement sur les schémas des usagers Maître.

Vous construirez, au niveau du schéma utilisateur Serviteur (étudiant A), des vues "intégrantes" qui porteront soit sur l'un ou l'autre des schémas Maitre, soit sur les deux schémas Maître à la fois (vues complexes). L'idée est de pouvoir consulter l'ensemble de l'information contenue dans les deux schémas Maître et de pouvoir établir des passerelles entre ces deux schémas afin de pouvoir poser des questions qui nécessitent l'accès à la totalité de l'information.

A ce sujet, vous exprimerez au travers de vues (matérialisées ou non matérialisées) que vous définirez sur le schéma du compte Serviteur sur licence, les requêtes suivantes. Vous définirez également les requêtes en algèbre relationnelle (en précisant le placement des relations) :

- **Donner les informations générales sur les virus**
- **Donner les acronymes des virus, leurs familles virales et leur mode de transmission**
- **Donner les facteurs viraux**
- **Donner le nom des maladies bactériennes**
- **Donner les informations que vous pouvez obtenir de part et d'autre des deux schémas concernant les virus d'un côté et les facteurs viraux d'un autre côté : acronyme, nom du virus, famille virale, type de génome**
- **Donner les maladies virales qui se transmettent par voie aérienne**
- **Donner les facteurs viraux qui ne sont pas référencés en tant que virus dans la base de données Virus-Transmission**

Vous expérimenterez, également, la création de liens de bd, entre les schémas utilisateurs . Vous noterez que les liens sont unidirectionnels et qu'il vous faudra créer deux liens si vous souhaiter naviguer par exemple de user1 sur master à user1 sur licence et inversément.

2.1 Vues matérialisées

Vous pouvez également envisager, une démarche, faisant appel à de la réplication, au travers de vues matérialisées [1] que vous construirez également au niveau du schéma utilisateur Serviteur. Différents ordres de construction de vues matérialisées vous sont proposés en guise d'exemple, vous pourrez les tester en lieu et place des vues précédentes.

```
drop materialized view test;
create materialized view t as select * from maladie;

create materialized view log on maladie;
create materialized view test refresh fast on commit as select * from maladie;

create materialized view log on maladie@u1M
create materialized view test as select * from maladie@u1M;
create view test1 as  select * from maladie@u1M;
create materialized view test refresh fast as select * from maladie@u1M;

-- deux vues du dictionnaire
select mview_name from user_mviews;
select primary_key from user_mview_logs;

-- synchronisation
 execute DBMS_MVIEW.REFRESH('test');
```

1. Pour plus d'informations voir : http ://didier.deleglise.free.fr/dba/structure/mat_view.htm

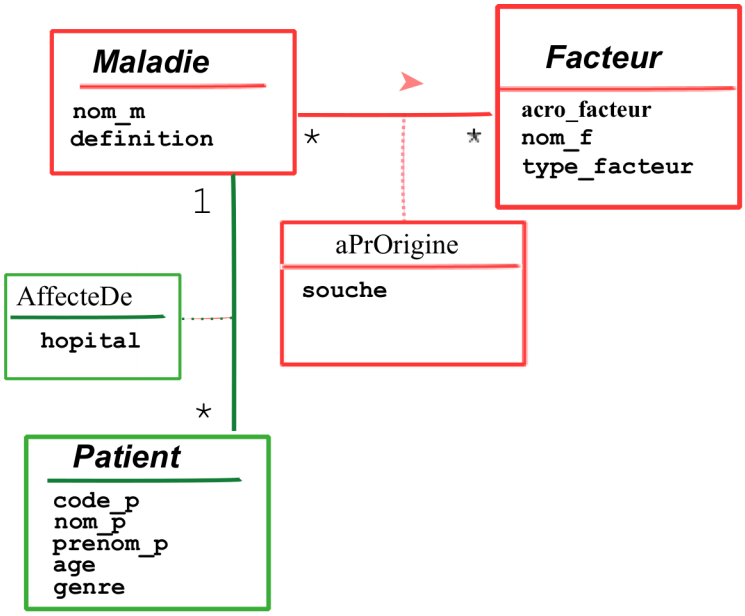

FIGURE 3 – Ajout Schéma Maladie et Facteur

2.2 Performances des accès

Vous construirez des plans d'exécution pour quelques requêtes listées ci dessus ou de votre choix. Vous réfléchirez au coût supplémentaire induit par les accès distants par comparaison avec une architecture classique (client-serveur et base de données centralisée). Un exemple de directive adaptée à dblink est donné ci-dessous, qui permet de désigner la table à partir de laquelle la jointure va s'opérer :

```
select /*+DRIVING_SITE(m)*/
m.a1, m.a2, v.a2 from user1.tablem@u1M m, user1.tablev@u1L v on m.a1 = v.a3;
```

3. Approche descendante

La fragmentation est une deuxième notion d'importance pour les BD réparties, tout autant que la réplication. La fragmentation et la réplication permettent de disposer de bases de données incorporant une forte volumétrie de manière efficace. L'objectif est d'inverser la vision précédente et de partir d'un schéma global qui sera ensuite partitionné. Les schémas locaux constituent cette fois-ci les vues du schéma global (local as view). A cet effet, vous travaillerez avec une table Patient qui vient se rajouter dans le schéma Maladie-Facteur.

> Patient (**code_p** varchar(12), nom_p varchar(15), prenom_p varchar(15), age integer, genre varchar(1), affection varchar(20), hopital varchar(15)))
> avec Patient (affection) $\sqsubseteq$ Maladie(nom_m)

L'information de la table provient de deux hôpitaux (SaintEloi et GDeChauliac). Vous proposerez différentes activités de fragmentation.

1. fragmentation horizontale : vous distribuerez l'information de Patient sur les deux schémas maître en fonction du centre de soin dans lequel le patient est traité. Donnez le schéma de fragmentation et le schéma d'allocation.

2. fragmentation verticale : vous construirez sur le schéma maître master une table à partir d'une projection sur les attributs : codeP, age, genre, affection à partir de Patient et qui sera destiné à des études épidémiologiques
3. fragmentation hybride : vous proposerez un exemple de fragmentation hybride qui vous semble pertinent. A cet effet, vous donnerez les schémas d'allocation et de fragmentation associés.