

L'objectif du TP est d'implémenter une base de données fédérée sur deux machines et de voir le paramétrage des communications distantes des bases de données Oracle. Une première machine contient une partie de la base, tandis que une seconde machine contient le complément de données. Un ensemble de vues sera créé afin de reconstituer une vision unique de la base complète.

**I/ Présentation.** On considère une BD CIRQUE dont le schema conceptuel :

PERSONNEL\_CIRQUE (NOM, ROLE) ;

NUMERO\_CIRQUE (TITRE, NATURE, RESPONSABLE) ;

ACCESSOIRE (NOM, COULEUR, VOLUME, RATELIER, CAMION) ;

UTILISATION (TITRE, UTILISATEUR, ACCESSOIRE) ;

CIRQUE est une base de données fédérée composée de deux Bases de données CIRQUE1, CIRQUE2 :

**Composition de la BD CIRQUE1 :**

PERSONNEL\_CIRQUE1, NUMERO\_CIRQUE1, ACCESSOIRE1, UTILISATION\_CIRQUE1.

NUMERO\_CIRQUE1 = NUMERO\_CIRQUE where

NUMERO\_CIRQUE.NOM = PERSONNEL\_CIRQUE1.NOM

UTILISATION\_CIRQUE1 = UTILISATION\_CIRQUE where

UTILISATION\_CIRQUE.NOM = PERSONNEL\_CIRQUE1.NOM

**Composition de la BD CIRQUE2 :**

PERSONNEL\_CIRQUE2, NUMERO\_CIRQUE2, ACCESSOIRE2, UTILISATION\_CIRQUE2.

NUMERO\_CIRQUE2 = NUMERO\_CIRQUE where

NUMERO\_CIRQUE.NOM = PERSONNEL\_CIRQUE2.NOM

UTILISATION\_CIRQUE2 = UTILISATION\_CIRQUE where

UTILISATION\_CIRQUE.NOM = PERSONNEL\_CIRQUE2.NOM

La table ACCESSOIRE est une table dupliquée en CIRQUE1 et en CIRQUE2.

**Le contenu de cette BD est le suivant :**

**PERSONNEL\_CIRQUE1**

NOM(40)	ROLE(20)
Clovis	Jongleur
Reine	Ecuyer
Louche	Clown
Benard	Equilibriste
Bignon	Musicien
Bordeau	Dompteur
Jerry	Clown
VASSEUR	Musicien
Fremez	Musicien
Dolores	Jongleur
Hebert	Jongleur
Sorel	Ecuyer
Sorel	Ecuyer
Loisel	Equilibriste
Jeanne	Jongleur
Sangtrespur	Dompteur

**PERSONNEL\_CIRQUE2**

NOM(40)	ROLE(20)
VASSEUR	Joncleur
LEBRAS	Equilibriste
LEJEUNE	Ecuyer
MARMOL	Clown
OLIVIER	Equilibriste
CALMETTES	Musicien
PORCHERON	Dompteur
QUENTIN	Clown
COLIGNON	Ecuyer
DA-SILVA	Musicien

**NUMERO\_CIRQUE1**

TITRE(30)	NATURE(20)	RESPONSABLE(20)
Les Zoupalas	Jonglerie	Clovis
Le coche infernal	Equitation	Reine
Les fauves	Clownerie	Louche

Les Smilers	Equilibre	Benard
Les Smilers	Equilibre	Clovis
les Smilers	Equilibre	VASSEUR
Les Fauves	Clownerie	Jerry
La passoire magique	Lion	Bordeau
Les Zozos	Clownerie	Jerry
Les Tartarins	Jonglerie	Demare

#### NUMERO\_CIRQUE2

TITRE(30)	NATURE(20)	RESPONSABLE(20)
Les Fauves	Clownerie	LEBRAS
Les Zoupalas	Jonglerie	VASSEUR
Les Zoupalas	Lion	LEJEUNE
Le coche infernal	Equitation	MARMOL
Les fauves	Clownerie	OLIVIER
Les Smilers	Equilibre	CALMETTES
Les Smilers	Lion	PORCHERON
les Smilers	Equilibre	COLIGNON
Les Fauves	Clownerie	DA-SILVA

NUMERO\_CIRQUE = NUMERO\_CIRQUE1 Union NUMERO\_CIRQUE2

#### ACCESSOIRE

NOM(30)	COULEUR(10)	VOLUME	RATELIER	CAMION
Ballon	Rouge	0.3	15	5
Barre	Blanc	0.6	19	5
Fouet	Marron	0.2	11	3
Bicyclette a elephant	Vert	0.4	27	8
Trompette	Rouge	0.2	2	1
Cercle	Magique Orange	0.2	1	1
Boule	Cristal	0.2	88	8
Cage a lions	Noir	10.0	0	2
Chaise longue de lion	Bleu	0.9	11	5
Peigne de chimpanze	Jaune	0.2	23	3
Etrier				

#### UTILISATION\_CIRQUE1

TITRE(30)	UTILISATEUR(20)	ACCESSOIRE(30)
Les Zoupalas	Dolores	Ballon
Les Zoupalas	Hebert	Ballon
Les Zoupalas	Dolores	Barre
Le coche infernal	Jeanne	Bicyclette a elephant
Le coche infernal	Sorel	Fouet
Les fauves	Jerry	Trompette
Les Smilers	Benard	Cercle magique
Les Smilers	Benard	Boule

#### UTILISATION\_CIRQUE2

TITRE(30)	UTILISATEUR(20)	ACCESSOIRE(30)
Les Smilers	VASSEUR	Bicyclette a elephant
La passoire magique	LEBRAS	Cage a lions
La passoire magique	DA-SILVA	Chaise longue de lion
Les Zoupalas	LEJEUNE	Ballon
Le coche infernal	MARMOL	Ballon
Le coche infernal	OLIVIER	Barre
Les fauves	CALMETTES	Bicyclette a elephant
Les Smilers	LEJEUNE	Fouet
Les Zoupalas	MARMOL	Trompette
Le coche infernal	OLIVIER	Cercle magique

UTILISATION\_CIRQUE = UTILISATION\_CIRQUE1 Union UTILISATION\_CIRQUE2

## II/ Travail à faire (Rappelons que dans ce tp on travail sur deux machines)

A. Utiliser le scripte disponible sur [http://nakechb.free.fr/M2\\_SIRES](http://nakechb.free.fr/M2_SIRES) pour créer sur chacun des deux machines la structure des quatre tables de la base en intégrant les contraintes d'intégrité (local) nécessaires et notamment les références (clé étrangère).

## B/ Peupler CIRQUE1 et CIRQUE2

**Remarque :** Oracle ne gère pas automatiquement les références décrites dans les scripts de création. La solution qu'on va adopter dans un premier temps est de corriger les erreurs de références entre Bases. On verra plus tard comment **Oracle assure l'intégrité référentielle des BDD avec l'utilisation des «trigger».**

**B.1/** Peupler les tables de la BD CIRQUE1 en exécutant le scripte créer\_cirque1 qui est disponible sur [http://nakechb.free.fr/M2\\_SIRES](http://nakechb.free.fr/M2_SIRES).

**B.2/** Peupler les tables de la BD CIRQUE2 en utilisant les données décrites dans les tables plus hautes. (penser aux contraintes d'intégrités en local).

**C/** L'objectif de cet exercice est de réer et tester un lien entre deux BDs une locale l'autre distante. Rappelons que le fichier **tnsnames.ora** décrit les différentes BD distantes qui sont accessibles par la BD locale. Ce fichier est dans le répertoire ORACLE\_HOME/network/admin

Dans la suite on suppose que les deux machines sont **al04-07** et **al04-06**, l'utilisateur "oracle" de la première machine est **aurelien**, et l'utilisateur "oracle" de la deuxième machine est **arnaud**.

Arnaud est un utilisateur oracle de la base 'BD\_AURLIEN' sa création peut se faire par la commande suivante :  
`create user arnaud identified by mot_passe`

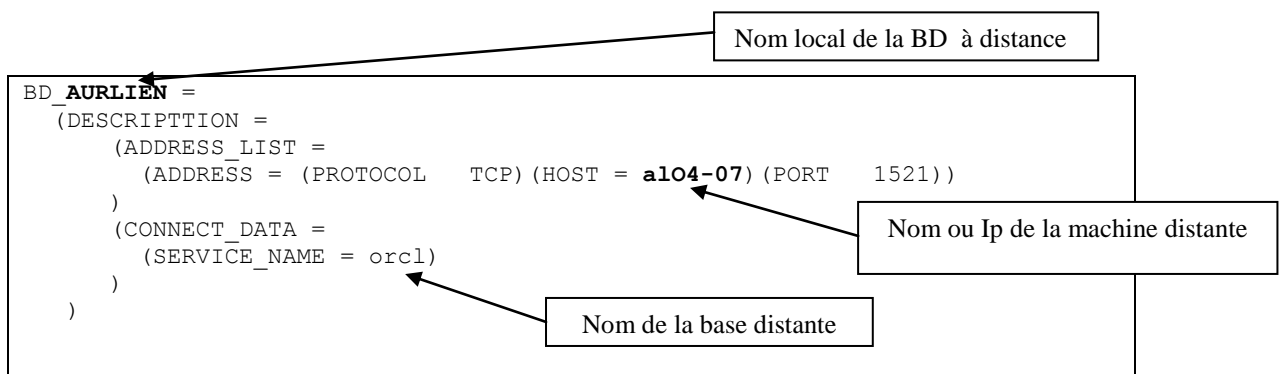
Pour donner à Arnaud les droits de connection et de pouvoir créer des tables :  
`grant connect, ressource to arnaud;`

Et pour donner à Arnaud le droit d'utiliser les tables de l'utilisateur Aurlien :  
`grant select on aurlien.personnel to arnaud;`

...

**C.1/ (Travail à faire par arnaud)** Éditer le fichier **tnsnames.ora** puis **ajouter** le bloc ci-dessous (**voir annexe pour un exemple complet**) :

`/usr/local/oracle/product/11g/network/admin/tnsnames.ora`



Cet ajout permet à *arnaud* d'accéder à la BD de AURLIEN (par la creation des alias).

**C.2/** Idem, aurelien doit également configurer l'accès à la BD distante de arnaud

**C.3/** Créer un liens entre la BD locale et la BD distante.

Command SQL :

```
create database link <nom> connect to <utilisateur de la base cible>
identified by <mdp dans DB cible> using <nom de la base cible>.
```

<nom de la base cible> doit être défini comme alias dans le fichier **tnsnames.ora** décrit plus haut.

La commande de création de lien que *arnaud* doit faire est donc :

```
create database link lien_a connect to arnaud
      identified by mot_passe using 'BD_AURLIEN' ;
```

C.4/ Tester ce lien par les requêtes SQL suivantes :

```
Desc USER_DB_LINKS
select db_link, username USER_DB_LINKS;
select * from aurlien.personnel@lien_a;
```

C.5/ Essayer d'afficher la structure d'une table de la base distante

( desc aurlien.personnel@lien\_a )

D/ Le but de cet exercice est de créer une vue unique permettant de d'afficher l'ensemble de données répartie (sur les deux machines) d'une table comme une table unique. **La création des vues passe par les étapes suivantes :**

D.1/ Rendre le lien qu'on vient de créer transparent par la création d'un synonyme :

```
create synonym personnel_distant for aurlien.personnel@lien_a;
```

D.2/ Une requête de création de vue pouvant être donc :

```
create view tout_personnel as
select * from personnel union select * from personnel_distant ;
```

D.3/ Vérifier : select \* from tout\_personnel

D.4/ Créer et tester les autres vues

E/ Le but de cette exercice est de faire quelques requêtes SQL pour manipuler la BDD.

E.1/ Tester une mise à jour directement sur le lien et sur la table local

```
update aurlien.accessoire@lien_a set volume = volume + 0.1 ;
```

...

Vérifier sur la machine local et sur la machine à distance ( Ne pas oublier les **commit**)

E.2/ Tester une mises à jour sur une vue totale . Votre conclusion ?

E.3/ Modifier la structure de la table accessoire en ajoutant une contrainte supplémentaire

```
alter accesoire (constran volume_ck (volume <=10 )) ;
alter aurlien.accessoire@lien_a (constran volume_ck (volume <=10 )) ;
```

On notera qu'Oracle n'autorise pas les commandes CREATE, ALTER et DROP de définition de données distantes.

E.4/ Tester maintenant des mises à jour violant les contraintes précédentes

F/ Créer quelques requêtes à base de jointures et/ou de requêtes imbriquées, par **exemples** :

- Afficher les personnels qui utilisent l'accessoire Ballon ;
- Afficher les accessoires utilisés par les Jongleurs ;
- Trouver les noms de personnels assurant au moins deux numéros
- Afficher le nombre des numéros utilisant des accessoires rouges
- Donner le nombre de personnels jouant chaque rôle
- Donner le nombre des numéros assures par chaque personne.

**CR : Dossier à rendre par quadri-nom par mail au [nakech@free.fr](mailto:nakech@free.fr) avec comme objet**

**m2BDD\_tp1\_votre\_nom.** Le CR comportant toutes les requêtes utilisées avec échantillonnage significatif d'exécution. **Date limite** : Mon prochain cours ??? .

**Pour faciliter la rédaction du CR le sujet est disponible sur : [http://nakech.free.fr/M2\\_SRO](http://nakech.free.fr/M2_SRO)**

## Annexe 1 : Création de la base de données

(Les scripts sont disponibles à l'url : [http://nakechb.free.fr/M2\\_SIRES](http://nakechb.free.fr/M2_SIRES))

```
create table personnel (nom varchar(40),role varchar(20),
                        constraint personnel_pri primary key(nom));

create table numero (titre varchar(30),nature varchar(20),responsable varchar(40),
                    constraint numero_pri primary key(titre),
                    constraint numero_etr foreign key(responsable) references personnel(nom));

create table accessoire (nom varchar(30),couleur varchar(10),
                        volume number(4,1),ratelier number(2),camion number(1),
                        constraint accessoire_pri primary key(nom));

create table utilisation (titre varchar(30),utilisateur varchar(40),
                        accessoire varchar(30),
                        constraint utilisation_pri primary key(titre, utilisateur, accessoire),
                        constraint utilisation_et1 foreign key(titre) references numero(titre),
                        constraint utilisation_et2 foreign key(utilisateur) references personnel(nom),
                        constraint utilisation_et3 foreign key(accessoire) references accessoire(nom));
```

### Saisie sur machine 1

```
insert into personnel values ('Clovis','Jongleur');
...
insert into numero values ('Les Zoupalas','Jonglerie','Clovis');
```

### Saisie sur machine 2

```
insert into personnel values ('LEBRAS', 'Joncleur') ;
...
insert into numero values ('Les Fauves', 'Clownerie', 'LEBRAS');
```

## Annexe2 : Rappel sur SQLPLUS-Oracle

### Notion de transaction en base de données

**Une transaction** est un ensemble de modifications de la base qui forme un tout indivisible. Il faut effectuer ces modifications entièrement ou pas du tout, sous peine de laisser la base dans un état incohérent. Les Systèmes de Gestion de Bases de Données permettent aux utilisateurs de gérer leurs transactions. Ils peuvent à tout moment :

- Valider la transaction en cours par la commande COMMIT. Les modifications deviennent définitives et visibles à tous les utilisateurs.
- Annuler la transaction en cours par la commande ROLLBACK. Toutes les modifications depuis le début de la transaction sont alors défaites.

En cours de transaction, seul l'utilisateur ayant effectué les modifications les voit. Ce mécanisme est utilisé par les systèmes de gestion de bases de données pour assurer l'intégrité de la base en cas de fin anormale d'une tâche utilisateur : il y a automatiquement ROLLBACK des transactions non terminées.

### Rappel de quelques instructions SQL utile pour ce TP :

**opérateur UNION :**

```
SELECT ...
UNION (SELECT ...)
```

**opérateur MINUS :**

```
SELECT ...
MINUS (SELECT ...)
```

### Mémento SQL\*PLUS

- `save filename.sql` : sauve le contenu du buffer dans un fichier de nom `filename.sql`
- `get filename.sql` : charge le buffer avec le contenu du fichier de nom `filename.sql`
- `start filename` : charge le buffer et lance l'exécution du fichier script `sql`
- `run` : lance l'exécution du contenu du buffer
- `spool filename.txt` : copie la sortie écran sur le fichier `filename.txt`
- `spool off` : suspend l'opération précédente
- `help commande` : pour obtenir de l'aide sur la commande donnée en argument

### Variables d'environnement (Mise en page SQLPLUS)

- `set long 1024` : pour voir la totalité des définitions de vues  
(exemple : `set long 1024 ; select * from USER_VIEWS ;`)
- `set pagesize 20` : formate la sortie écran par blocs de 20 lignes
- `set pause on` : ne visualise la sortie qu'après un 2e RC - bloc par bloc -
- `set timing [on|off]` : active ou désactive le chronomètre.

### Quelques Vues de "méta-base"

(dictionnaire de données ou tables système) **décrivant les objets utiles pour ce TP :**

- `desc[ribe] tablename` : donne le schéma de la relation `tablename`
- `all_catalog` : table système donnant toutes les tables accessibles
- `user_catalog` : table système donnant les seules tables du USER
- `all_objects` : table système donnant tous les objets accessibles
- `user_objects` : table système donnant les seuls objets du USER
- `user_sys_privs` : table système donnant les privilèges système du USER
- `user_tab_privs` : table système donnant les privilèges sur les objets accessibles
- `USER_TABLES` : Description des tables créées par l'utilisateur.
- `USER_TAB_COLUMNS` : Description des colonnes de chaque table ou vue créée par l'utilisateur courant. Chaque ligne de la vue `col` décrit une colonne.
- `USER_TAB_COMMENTS` : Commentaires sur les tables et les vues créés par l'utilisateur.
- `USER_TAB_GRANTS` : Droits sur ses objets, tables ou vues, donnés par l'utilisateur.
- `USER_USERS` : Informations sur l'utilisateur courant.
- `USER_VIEWS` : Texte des vues créées par l'utilisateur.

### Exemples :

```
desc all_db_links;
SELECT * FROM all_db_links;
```

- Pour obtenir une liste complète des vues statiques DBA utiliser la requête suivante :  

```
desc dict ;
select * from dict where table_name like 'DBA%'
```
- Synonymes : Les noms des vues étant assez longs, les synonymes suivants ont été définis :  
`COLS` : Synonyme pour `USER_TAB_COLUMNS`  
`MYPRIV` : Synonyme pour `USER_USERS`  
`TABS` : Synonyme pour `USER_TABLES`  
`OBJ` : Synonyme pour `USER_OBJECTS`  
`SYN` : Synonyme pour `USER_SYNONYMS`  
`cat` : synonyme pour `user_catalog`

### Exemples :

la requête : `select * from tabs;` Affiche une liste des tables de l'utilisateur.

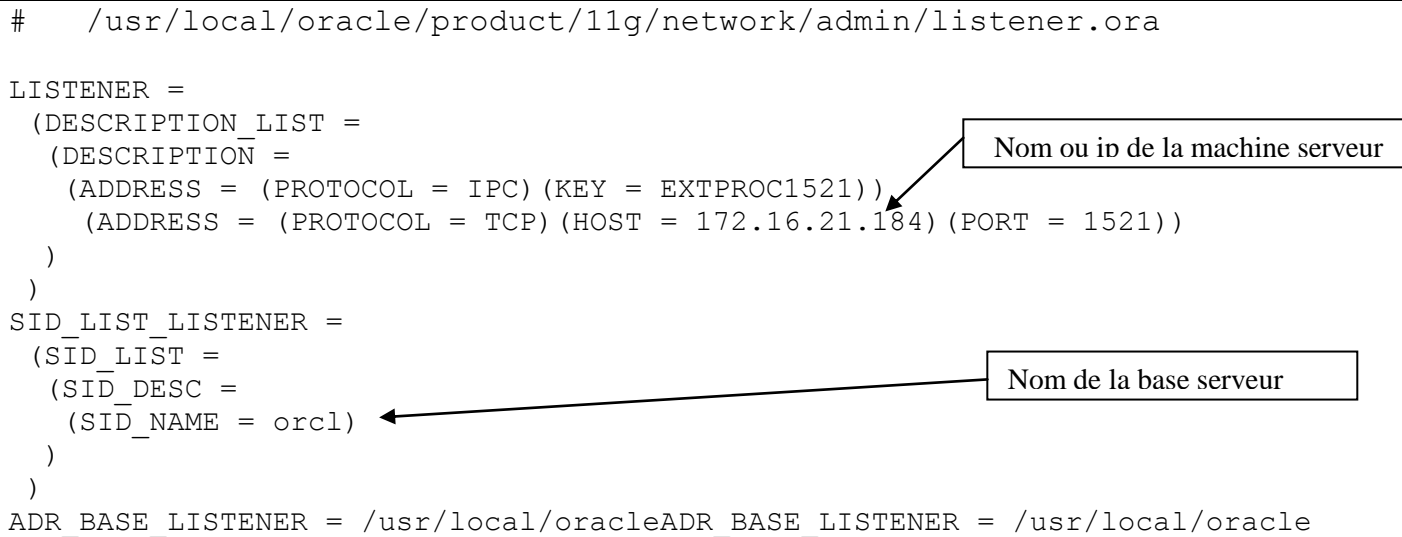
la requête : `select * from SYN ...`

### Annexe 3 : Exemples de fichiers de configuration

**listener.ora** Ce fichier gère les demande d'accès à distance.

```
# /usr/local/oracle/product/11g/network/admin/listener.ora

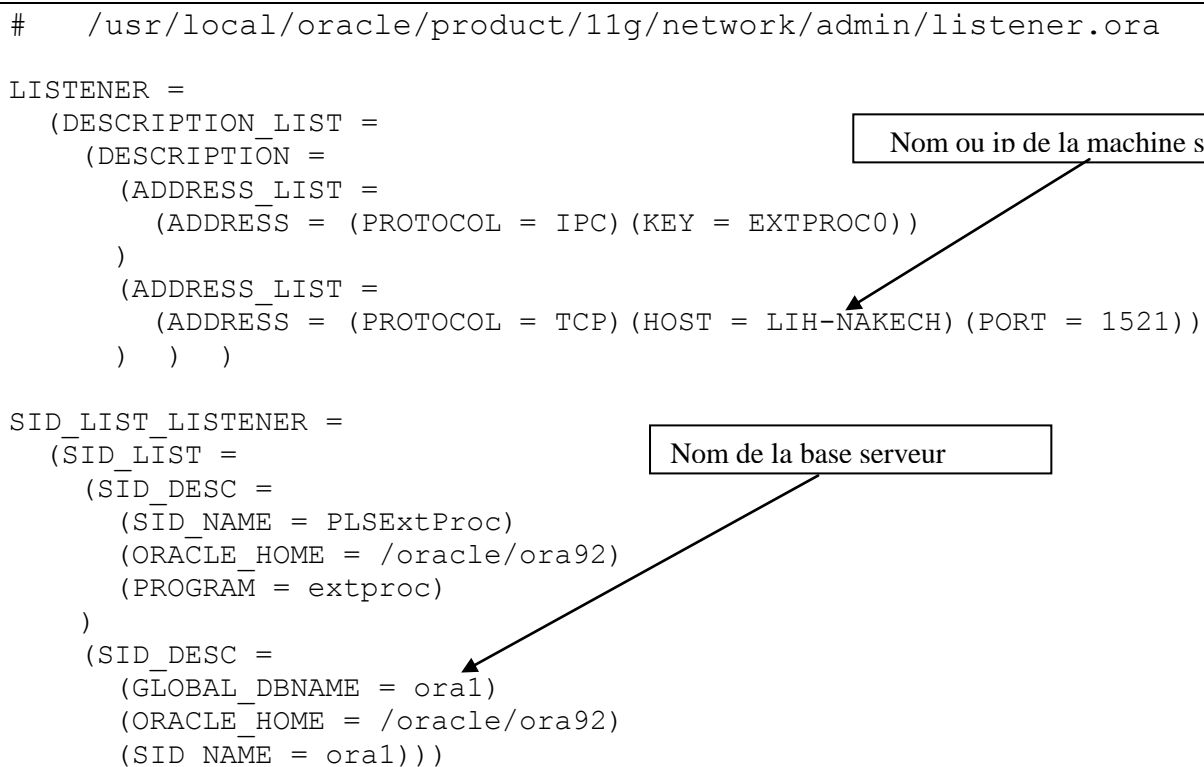
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = IPC) (KEY = EXTPROC1521))
      (ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21.184) (PORT = 1521))
    )
  )
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = orcl)
    )
  )
ADR_BASE_LISTENER = /usr/local/oracleADR_BASE_LISTENER = /usr/local/oracle
```



#### Un autre exemple du fichier **listener.ora**

```
# /usr/local/oracle/product/11g/network/admin/listener.ora

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC) (KEY = EXTPROC0))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP) (HOST = LIH-NAKECH) (PORT = 1521))
      )
    )
  )
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/ora92)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = oral)
      (ORACLE_HOME = /oracle/ora92)
      (SID_NAME = oral))
  )
```



## Exemple complet de 2 fichiers tnsnames.ora (site1 site2)

### # site 1 (de Pascal)

```
# tnsnames.ora
# Network Configuration File: /oracle/product/11g/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.
LISTENER_ORCL =
  (ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21.180) (PORT = 1521))
ORCL =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21.180) (PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = orcl)
    )
  )
ORCL_MICHEL = (DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21.186) (PORT = 1521))
)
(CONNECT_DATA =
(SERVICE_NAME = orcl)
))
```

Nom local de la BD à

Ip de la machine distante

La partie **en gras** est à ajouter sur le fichier tnsnames existant

### # site 2 (de MICHEL)

```
# tnsnames.ora Network Configuration File:
/oracle/product/11g/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.
LISTENER_ORCL =
  (ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21. 186) (PORT = 1521))
ORCL =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21. 186) (PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = orcl)
    )
  )
ORCL_Pascal = (DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = 172.16.21.180) (PORT = 1521))
)
(CONNECT_DATA =
(SERVICE_NAME = orcl)
))
```

Nom local de la BD à distance

Ip de la machine distante

### Commentaire :

- Alias (ORCL\_MICHEL) est entré dans le fichier de configuration tnsnames.ora du site1 (site de pascal). Il permet de référencer la base de données orcl hébergée sur la machine site2 ayant pour adresse IP 172.16.21.186. Aussi, cet alias permet de se connecter à cette base distante en utilisant le port "1521".
- De même pour que le site2 se connecte au site1, on procède de la même manière sauf il faut changer l'alias (ORCL\_Pascal) et les adresses IP sur le fichier tnsnames.ora