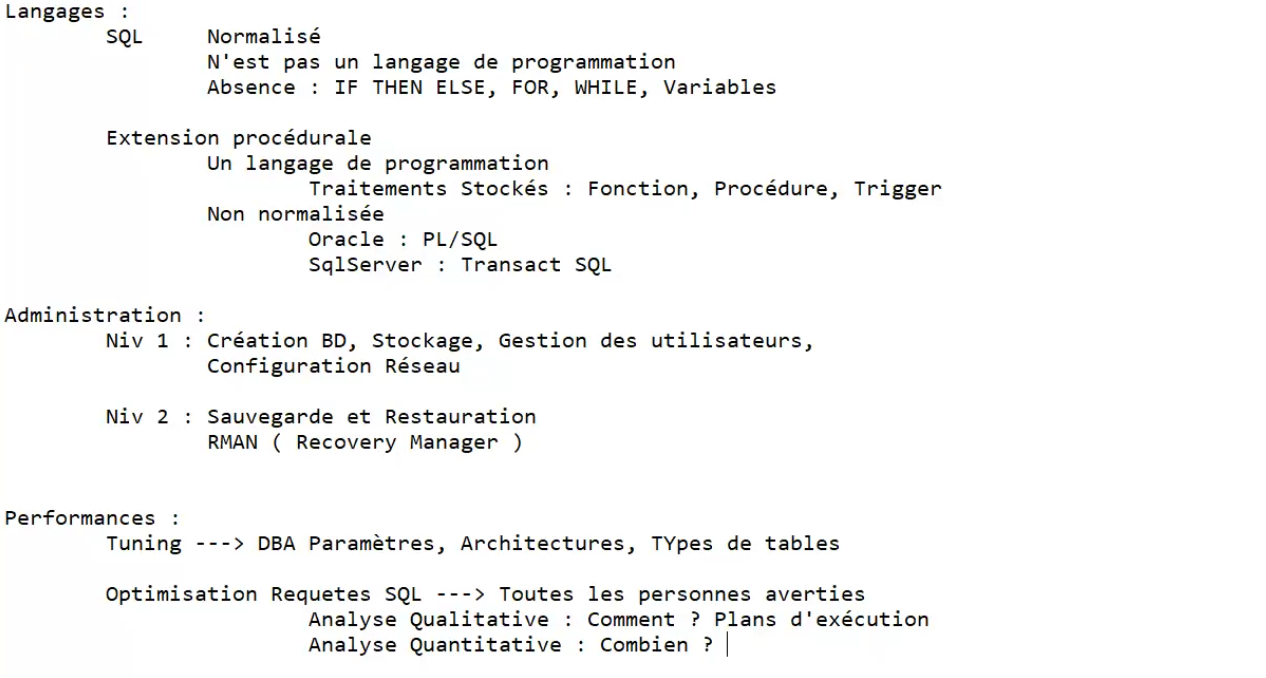
**ORACLE opération et administration**

**Notes : 17/04/2023**

**INTRODUCTION**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**PLAN DU PROGRAMME :**

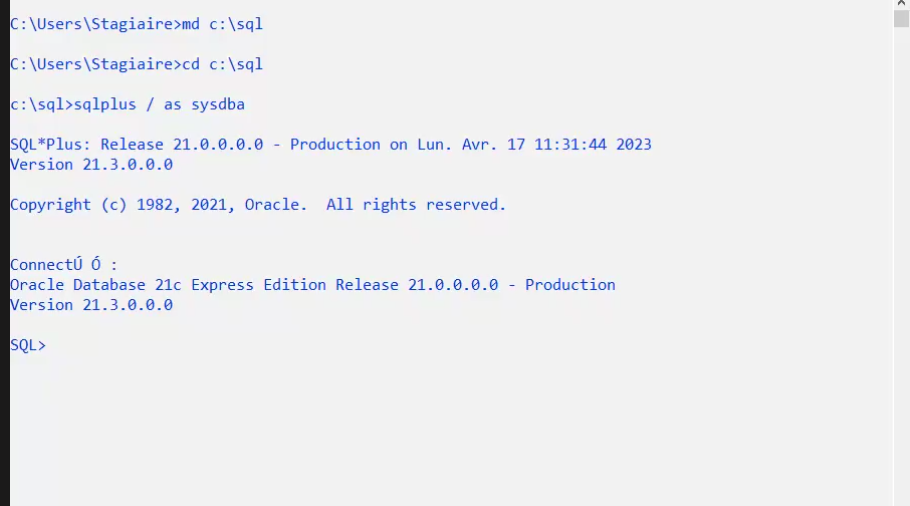
Oracle Opération et Administration :

SQL : Consultation

Administration : Architecture, Manipulation des fichiers

Sauvegarde et Restauration avec RMAN

**Dans cmd :**



**NB :**

Le mots clés de la commande en maj juste pour la bonne pratique mais pas nécessaire.

Dans les tables c’est essentiel de faire la différence entre min et maj

Le virgule c’est une séparation pour les listes de commande

Le ; pour finir et exécuter

GRANT : Ouvrir session – création de structure en allouant des espaces

**Création d’un USER c##demo avec un mot de passe sql:**

CREATE USER c##demo IDENTIFIED BY sql ;

Moindre privilège au début pour ce USER

=>

GRANT

CONNECT, -- ouverture de session

RESOURCE, -- création de structures

UNLIMITED TABLESPACE -- allocation illimitée

TO

c##demo ;

SHOW USER

**Modifier un Script:**

ed %ORACLE\_HOME%\rdbms\admin\utlsampl -- Un fichier sample fourni par Oracle

Enter

Sélectionner depuis le début jusqu’à create table dept et supprimer.

7788 SCOTT : On remplace par : 13-7-1987 dd-mm-yyyy et on supprime -85

7876 ADAMS : Pareil

On supprime la commande EXIT a la fin du fichier

On enregistre

Et on relance le script :

@%ORACLE\_HOME%\rdbms\admin\utlsampl

On vérifie :

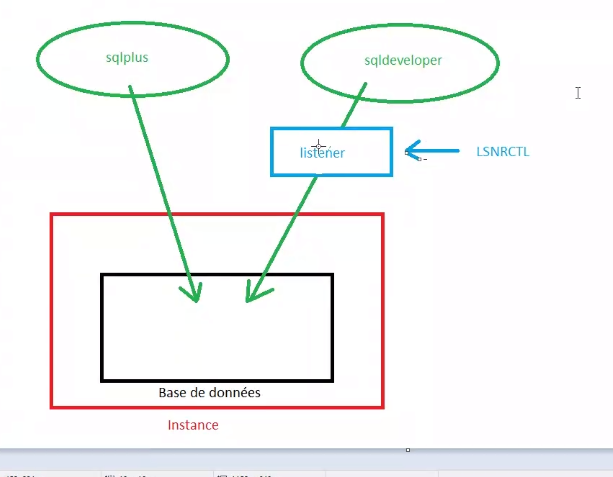
Select \* from dept ;

On installe SQL Developer: [Identifiant Oracle - Connexion SSO](https://login.oracle.com/mysso/signon.jsp)

Une instance est considéré comme un driver vis-à-vis de la base de donnée, elle nous permet de travailler avec la base de donné, comme avec l’outil SQL+

Pour plus de confort on travaille on utilisera l’outil sqldeveloper, mais ce dernier il ne se connecte pas directement à la base de donnée, il a besoin d’une brique intermédiaire qu’on appelle le listener, ce listner on le contrôle via une commande dans SQL+ =>

$lsnrctl status



Sur SQL+ on lance cette commande :

$lsnrctl status

PUIS :

**Sur SQL developpeur :**

Sur + vert :

Name : formation

Type de base de données : Oracle ( mais on peut changer)

Nom d’utilisateur : c##demo

M de P : sql

On coche : Enregistrer

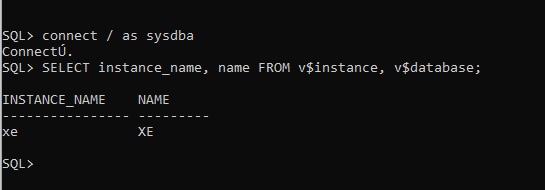
Boutton : tester

=> On essaye (Test, Connexion...) et si ça marche pas on change le Localhost pour IP locale ( ip config)

Les Base de données en Oracle accepte le .XML et .JSON

SGBDR gère plusieurs types de données

Sur SQL+:



On se connecte en tant que Admin :

Connet / as sysdba

Pour trouver notre instance :

SELECT instance\_name, name

FROM v$instance, v$database;

Taches administration :

------ > dictionnaire de données

-> toutes les définitions

---- > v$ ….

Dans notre exemple : L’Instance qui pilote une base de données qui s’appelle XE ( ce n’est pas nécessaire qu’elle porte le même nom)

------- L’admin pourra changer physiquement les données sans avoir un impact sur le code et les requetes donc partie logique

Le modèle relationnel =>

Une Table = Une relation

Null : information inconnue en BDD

Chaque table possède un identifiant qui est un Primary Key \_ Clé primaire, pas obligatoire mais pour une bonne pratique

Cette clé permet de trouver une table facilement et rend plus facile l’écriture des requetes. Donc vaut mieux avoir toujours des clés ( c’est une contrainte )

----------------

Index. Et table contraintes pour stockage

---------------

Les vues = Un Select ou Requête enregistré

---------------

CREATE

ALTER

DROP

RENAME

TRUNCATE = Pour vider

--------------

SELECT

--------------

INSERT

UPDATE

DELETE

-----------

**GESTION DE TRANSACTION:**

COMMIT

ROLLBACK

SAVEPOINT

----------

GRANT

REVOKE

-----------

**Règles de gestion ou contraintes :**

**Sur SQLDEVELOPER :**

**Exemple de contraintes :**

Ex1 : S’assurer que le nom de l’employé soit renseigné = >

Ajouter une règle de gestion ou contrainte sur la table, on agit sur la structure de la table, celle-là consiste à obliger d’avoir un nom pour les employés :

ALTER TABLE emp ADD

CONSTRAINT nn\_ename\_emp

CHECK ( ename IS NOT NULL );

On essaie d’ajouter un employé dans la table emp sans un NOM :

INSERT INTO emp ( empno )

VALUES ( 8000);

On aura un message d’erreur:

Erreur commençant à la ligne: 1 de la commande -

INSERT INTO emp ( empno )

VALUES ( 8000)

Rapport d'erreur -

ORA-02290: violation de contraintes (C##DEMO.NN\_ENAME\_EMP) de vérification

EX2 : Interdire les doublons dans la colonne NAME

ALTER TABLE emp ADD

CONSTRAINT u\_ename\_emp

UNIQUE ( ename ) ;

On essaie d’insérer un employé dont le nom est SMITH qui existe déjà :

INSERT INTO emp ( empno, ename )

VALUES ( 8000, ‘SMITH’);

On aura un message d’erreur:

Erreur commençant à la ligne: 1 de la commande -

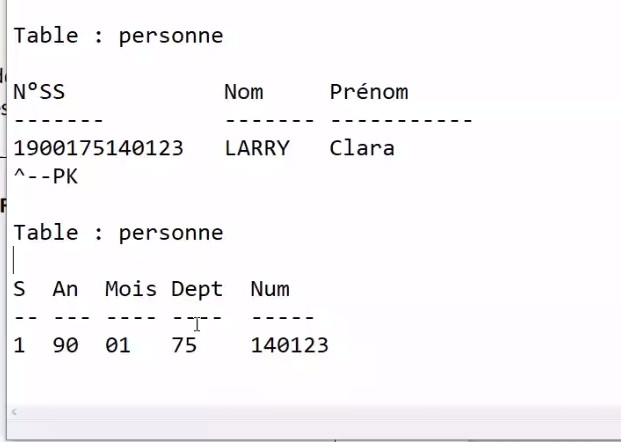
INSERT INTO emp ( empno )

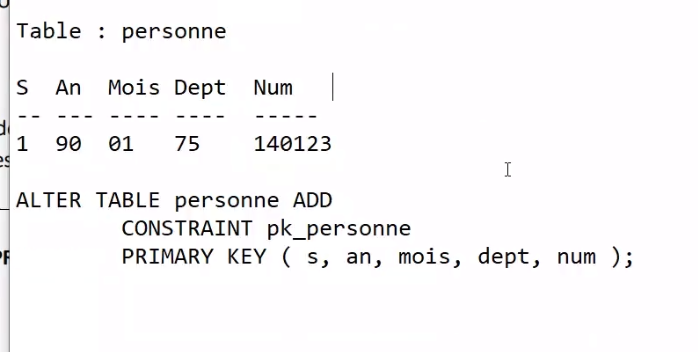
VALUES ( 8000)

Rapport d'erreur -

ORA-02290: violation de contraintes (C##DEMO.NN\_ENAME\_EMP) de vérification

**EX3 :** Contrainte avec Clé primaire :

****

****

ALTER TABLE salgrade ADD

CONSTRAINT pk\_salgrade

PRIMARY KEY ( grade );

**On essaie :**

INSERT INTO salgrade

VALUES ( 1, 1000, 2000 ) ;

Erreur commençant à la ligne: 1 de la commande -

ALTER TABLE salgrade ADD

CONSTRAINT pk\_salgrade

PRIMARY KEY ( grade )

Rapport d'erreur -

ORA-02437: impossible de valider (C##DEMO.PK\_SALGRADE) - violation de la clé primaire

02437. 00000 - "cannot validate (%s.%s) - primary key violated"

\*Cause: attempted to validate a primary key with duplicate values or null

values.

**Une clé primaire** est une colonne ou un ensemble de colonnes qui identifie de manière unique chaque enregistrement dans une table. La clé primaire est utilisée pour garantir l'intégrité des données et pour permettre des recherches efficaces dans une table.

Chaque table ne peut avoir qu'une seule clé primaire, et chaque valeur de clé primaire doit être unique et non nulle. Les clés primaires sont souvent utilisées pour créer des relations entre des tables et pour assurer l'unicité des enregistrements.

Pour définir une clé primaire dans SQL, vous pouvez utiliser la clause PRIMARY KEY lors de la création de la table, ou vous pouvez ajouter une contrainte de clé primaire à une table existante en utilisant la commande ALTER TABLE.

**Une clé unique** est une contrainte qui garantit qu'une colonne ou un ensemble de colonnes d'une table a une valeur unique pour chaque enregistrement, mais contrairement à la clé primaire, elle peut accepter les valeurs NULL.

Cela signifie qu'une colonne avec une contrainte de clé unique peut contenir des valeurs NULL en plus des valeurs non NULL, mais toutes les valeurs non NULL doivent être uniques.

Une clé unique peut être utilisée pour garantir l'unicité d'une colonne ou d'un groupe de colonnes, même si cette colonne ou ce groupe de colonnes ne sera pas utilisé comme identifiant principal de la table. Par exemple, une table "utilisateurs" pourrait avoir une colonne "email" avec une contrainte de clé unique pour garantir que chaque adresse e-mail est unique.

EX4 : S’assurer que les noms des employés soient toujours en majuscule.

ALTER TABLE emp ADD

CONSTRAINT ck\_maj\_ename\_emp

CHECK ( ename = upper( ename ) ) ;

On essaie :

INSERT INTO emp ( empno, ename )

VALUES ( 8000, ‘larry’);

On aura un message d’erreur :

Erreur commençant à la ligne: 1 de la commande -

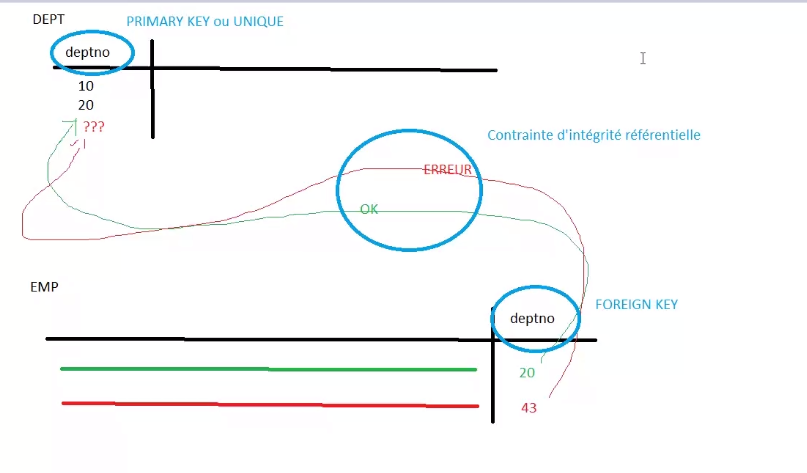
INSERT INTO emp ( empno, ename )

VALUES ( 8000, 'larry')

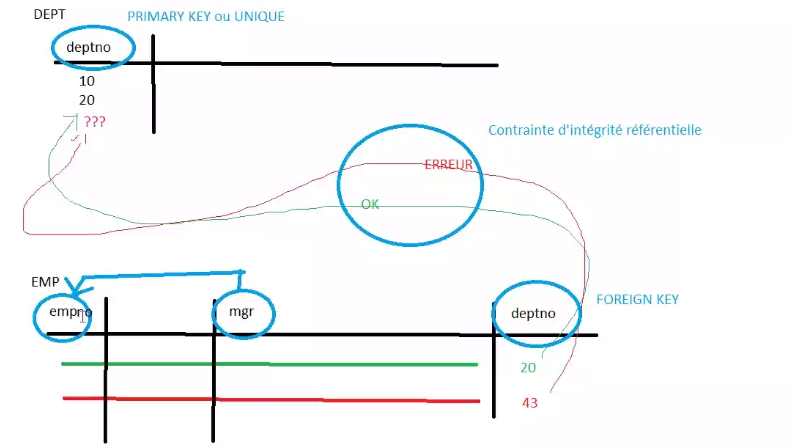
Rapport d'erreur -

ORA-02290: violation de contraintes (C##DEMO.CK\_MAJ\_ENAME\_EMP) de vérification

EX5 : Contrainte d’intégrité référentielle :



Application sur notre table : S’assurer qu’un manager soit au moins un employé de la société. ( mgr 1234 dans notre exemple ).



ALTER TABLE emp ADD

CONSTRAINT fk\_emp\_emp

FOREIGN KEY( mgr )

REFERENCES emp( empno );

INSERT INTO emp ( empno, ename, mgr)

VALUES ( 8000, ‘LARRY’, 1234);

On aura un message d’eureur :

Erreur commençant à la ligne: 1 de la commande -

INSERT INTO emp ( empno, ename, mgr)

VALUES ( 8000, 'LARRY', 1234)

Rapport d'erreur -

ORA-02291: violation de contrainte d'intégrité (C##DEMO.FK\_EMP\_EMP) - clé parent introuvable

EX6 : **Les triggers :**

Permet de faire des audits, propager l’info dans une table, interdire des actions sur une table après une certaine heure …

Objectif de l’exemple: On va transformer automatiquement le nom de l’employé en majuscules :

CREATE OR REPLACE TRIGGER insemp

BEFORE INSERT ON emp

FOR EACH ROW

BEGIN

:NEW.ename := upper( :NEW.ename );

END;

/

On essaie :

INSERT INTO emp ( empno, ename )

VALUES ( 8000, ‘larry’);

Ça Fonctionne !!! avec un message :

1 ligne inséré.

**LE NULL : 100 + NULL = NULL**

**Ex:**

SELECT

ename,

sal,

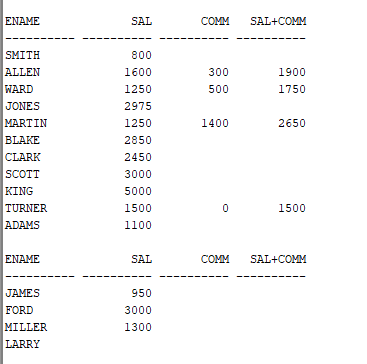
comm,

sal+comm

FROM

emp;

Résultat :



**NVL et COALESCE:** Donne une valeur alternative quand on rencontre un NULL

**COALESCE** est la préférée car NVL est made in ORACLE

Les Fonctions normalement ne sont pas normalisées, sauf que COALESCE est normalisée

On préfère toujours les normalisées car elles fonctionnent dans plusieurs type de BDD

Exemple d’Application de **COALESCE :**

**NB : On vérifie si les colonnes acceptent les NULL alors on les traite avec cette commande COALESCE**

SELECT

ename,

sal,

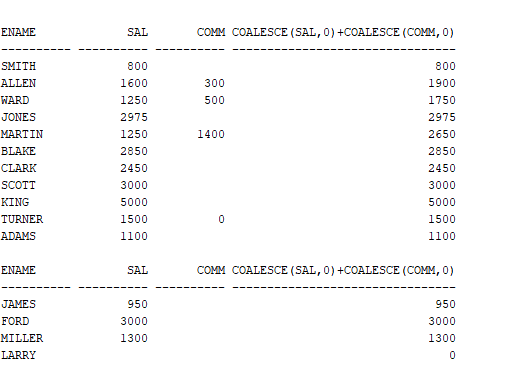
comm,

coalesce(sal,0)+ coalesce(comm,0)

FROM

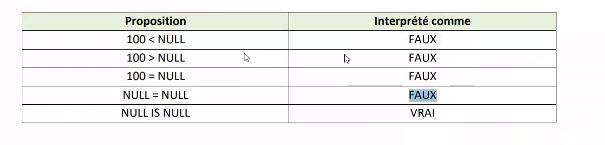
emp;

Résultat :



Avec les chaines de caractère le NULL sera un vide ‘a’ || ‘NULL’ = ‘a’

Comment se comporte le NULL lors de la comparaison :



L’opérateur « **IS »**le seul qui sait lire le NULL, donc si on veut repérer un NULL dans une table on utiliser le **IS**

**EXEMPLE :**

SELECT ename

FROM emp

WHERE comm = NULL;

Résultat:



SELECT ename

FROM emp

WHERE comm IS NULL;

Résultat:



**TABLE DE DICTIONNAIRE DE DONNEE :**

Il contient les informations concernant les objets ( Table, View, Sequence ). SQLdeveloper interroge cette TABLE de DD pour afficher nos requêtes.

Pour le consulter Oracle a mis à disposition un Views du D.D :

Pour chaque objet on aura : user\_tables, all\_tables,dba\_tables / user\_view, all\_views, dba\_views etc ……

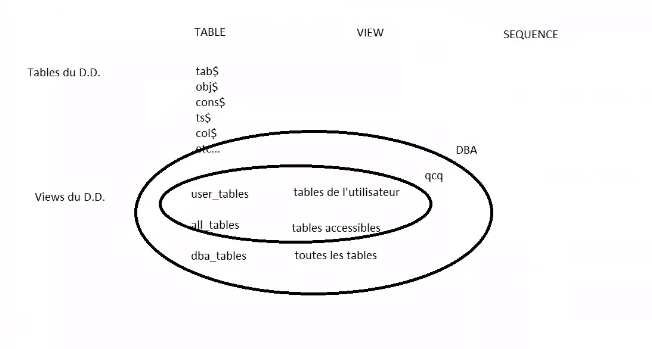
user\_tables : tables de l’utilisateur

all\_tables : tables accessibles

dba\_tables : toutes les tables

Le user a accès à user\_tables  et all\_tables :

L’admin a accès à user\_tables  et all\_tables et dba\_tables



**EXEMPLE :**

SELECT table\_name

FROM user\_tables;

SELECT column\_name

FROM user\_tab\_columns

WHERE table\_name = 'EMP';

SELECT constraint\_name, constraint\_type

FROM user\_constraints

WHERE table\_name = 'EMP';

SELECT table\_name

FROM user\_tab\_columns

WHERE column\_name = 'DEPTNO';

**La consultation avec SELECT :**

SELECT : Affichage ou projection de donnée

FROM

WHERE

GROUP BY : une fonction qui s’applique sur des groupes de données, comme les fonctions min max sum abg count

HAVING : filtre sur les fonctions de groupe

ORDER BY : Trier

Il faut respecter l’ordre de ces mots

Exercice 1 :

Afficher pour chaque employé la phrase :

xxx a pour rémunération annuelle yyy

ou XXX est le nom de l’employé

YYY sa rémunération annuelle

Sal --- > mensuel

Comm -> trimestrielle

SELECT ename ||

' a pour rémunération annuelle ' ||

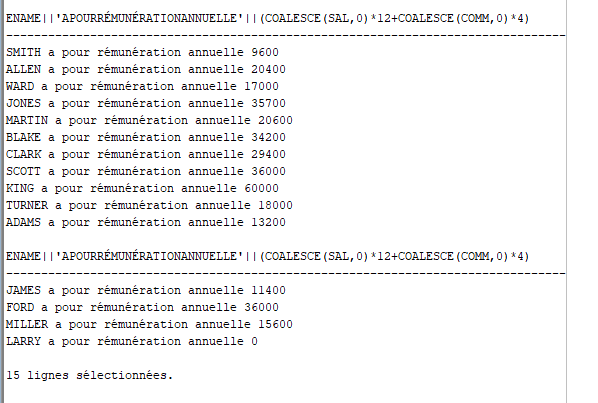
( coalesce(sal,0)\*12 +

coalesce(comm,0) \* 4 )

FROM

emp;

Résultat :



Les || pour coller des chaines de caractère

SUM est utilisé sur des colonnes pas sur des lignes.

**LES ALIAS :**

On peut renommer les colonnes :

SELECT ename ||

' a pour rémunération annuelle ' ||

( coalesce(sal,0)\*12 +

coalesce(comm,0) \* 4 )

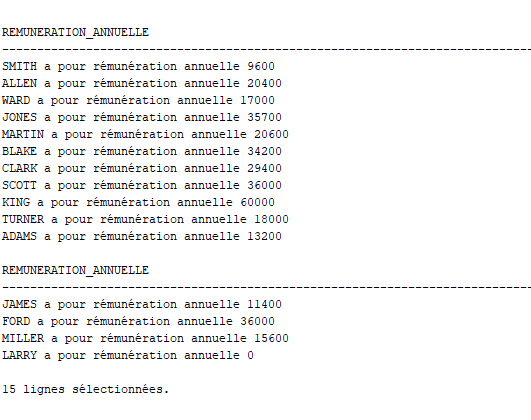
AS remuneration\_annuelle

FROM

emp;

Le “ \_ “ à la place de l’espace sinon entre deux guillemets

Résultats :



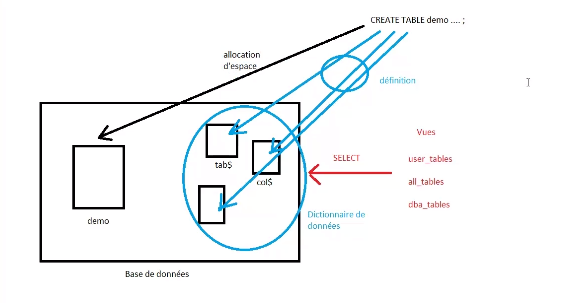
**Notes : 18/04/2023**

Rappel Dictionnaire de donnée

SELECT constraint\_name, table\_name

FROM user\_constraints

WHERE constraint\_type = ‘P’ – - primary key



**ALL / DISTINCT**

Ex1: Quelles sont les départements avec employée ?

SELECT deptno

FROM emp;

---- > ne répond pas à ma question car il y a des doublons, car il fait le ALL

SELECT DISTINCT deptno

FROM emp;

----- > Répond sans doublons!

**Source des informations:**

( ORACLE)

SELECT 5 \* 3

FROM

------ > OK

SELECT DISTINCT 5 \* 3

FROM emp;

------ > OK sans doublons

SELECT 5 \* 3

FROM dual;

Dual: Une table dans ORACLE, avec une seule colonne et une seule ligne, pour respecter le syntaxe de ORACLE qui exige d’avoir un FROM après le SELECT

SELECT sysdate

FROM dual ;

SELECT \*

FROM dual;

Sur autre SGBDR comme SQLserver et Postgresql on peut faire =>

SELECT 5 \* 3 sans le FROM

**Critères de selection**

WHERE

Ex2 : Afficher pour chaque employé dont le job est SALESMAN :

- Son nom

- l’écart entre son salaire et sa commission ramenés à une année.

SELECT ename

AS "employe",

( coalesce(sal,0)\*12 -

coalesce(comm,0) \* 4 )

AS " ecart salaire commission"

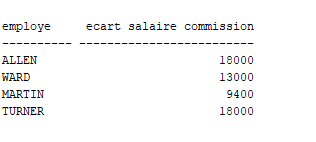
FROM

emp

WHERE

job = 'SALESMAN';

Résultat:



Ex3: Afficher pour chaque employé dont le salaire est supérieur à 1000 :

- son nom

- la moyenne mensuelle salaire commission

SELECT ename

AS "employe",

round ( ( coalesce(sal,0)\*12 +

coalesce(comm,0) \* 4 )/ 12 )

AS " Moyenne Mensuelle Salaire Commission"

FROM

emp

WHERE

sal > 1000;

-------- le round pour arrondir le résultat

Cas particulier : Afficher les noms des employés travaillant

dans le département SALES

Dans la table DEPT les SALES correspondent au Numéro 30

SELECT ename

FROM emp

WHERE deptno = 30 ;

---- > OK comme résultat mais A NE PAS FAIRE, car le numéro 30 pourra changer, en plus c’est nous qui l’a cherché

DONC on procède comme suivant ( SELECT imbriqué):

SELECT ename

FROM emp

WHERE deptno = ( SELECT deptno

FROM dept

WHERE dname = ‘SALES’;

---- > OK comme résultat et c’est la bonne méthode

----- > il existe une alternative à base de jointure

Ex4 APPLICATION du cas particulier ci-dessus:

Afficher pour chaque collègue de SMITH :

- le nom du collègue

- son salaire

- sa commission

Deux personnes sont collègues si elles travaillent dans le même département

On admet que SMITH est son propre collègue.

SELECT

ename,

sal,

comm

FROM

emp

WHERE deptno = ( SELECT deptno

FROM

emp

WHERE

ename = 'SMITH' );

------------------------------------------------------------------

Opérateur : IN – Between – LIKE –

Ex5 sur l’opérateur LIKE:

SELECT ename FROM emp WHERE ename LIKE

-- 'A%' -- qui commence par A

-- '%S' -- qui finit par S

-- '%E%'-- qui contient E

-- '\_A%' -- un A en deuxième position

-- '%E\_' -- un E en avant dernière position

'\_\_M%' -- un M en troisième position

;

SELECT ename FROM emp

WHERE substr ( ename, 3, 1 ) = 'M'

;

**Opérateur EXISTS:** Teste l’existence d’un retour d’un select imbriqué

Ex6 :

SELECT

‘Il existe des SALESMAN’ AS info

FROM

dual

WHERE

EXISTS ( SELECT

ename ---- n’importe que ce soit le nom ici, on met un « 1 » dans les entreprises ou NULL et on aura le meme résultat

FROM

emp

WHERE

Job = ‘SALESMAN’ ) ;

**Combinaison de critères:**

NB: le AND est prioritaire au OR donc on utilise des ( ) pour le OR

Trier

----------------------------

**JOINTURE :**

Afficher pour chaque employé :

- son nom

- le nom de son département

SELECT

ename,

dname

FROM

emp,

dept ;

------ > Est-ce que le résultat est conforme à mes attentes ? NON, RESULTATS INCORRECT

------ > Pas d’erreur de syntaxe,

------ > Il fait un **produit cartésien** si on ne précise pas une condition ( le produit cartésien n’est pas interdit mais il faut savoir ou et quand l’utiliser)

POUR AVOIR UN BON RESULTAT ON UTILISE **UNE JOINTURE :**

SELECT

ename,

dname

FROM

emp,

dept ;

**WHERE**

**Emp.deptno = dept.deptno ;**

MIEUX METTRE DES ALLIAS DANS CE CODE =>

SELECT

e.ename,

d.dname

FROM

emp e,

dept d

WHERE

e.deptno = d.deptno;

AUTRE SYNTAXE POUR LES JOINTURES : MIEUX UTILISER CELLE Là SI NOUS ON ECRIT LE CODE

SELECT

e.ename,

d.dname

FROM

emp e,

JOIN

dept d

ON

e.deptno = d.deptno;

-----------------------------------------------------------------------------------

EX7 : Afficher pour chaque employé :

- son nom

- sa rémunération annuelle

Sal -- > mensuel

Comm 🡪 trimestrielle

- son lieu de travail

A condition que ce dernier soit NEW YORK ou DALLAS

Trier par lieu, rémunération décroissante et nom.

SELECT

e.ename

AS "EMPLOYE",

( coalesce(sal,0)\*12 +

coalesce(comm,0) \* 4 )

AS "Renumération Annuelle",

d.LOC

FROM

emp e

JOIN

dept d

ON

e.deptno = d.deptno

WHERE

d.LOC IN ( 'NEW YORK', 'DALLAS' )

ORDER BY

3, 2 DESC, 1;

NB : ORDER ne s’imbrique pas, ici le 3 et 2 et 1 c’est suivant qu’on les a citer dans le select

CAS avec FK et PK

SELECT

e.ename,

d.dname

FROM

emp e,

JOIN

dept d

ON

e.deptno = d.deptno

--- (FK) (PK) - - - C.I.R ( Contrainte)

;

Ex8 : APPLICATION ou la jointure n’est pas une égalité ( THETA) :

Afficher pour chaque employé :

- son nom

- son niveau de salaire

SELECT

e.ename,

s.grade

FROM

emp e

JOIN

salgrade s

ON

e.sal BETWEEN s.losal AND s.hisal;

---------------------------------------------------------

Ex9: Afficher les employés qui ont le même salaire et le même job que SCOTT excepté lui-même.

SELECT

e.ename

FROM

emp s --- considérée comme inclut que SCOTT

JOIN

emp e --- les employés ( sans SCOTT)

ON

s.sal = e.sal AND s.job = e.job

WHERE

s.ename = 'SCOTT'

AND

e.ename != 'SCOTT'

;

---------------------------------------------------------------------------------------------

Ex10 APPLICATION du cas particulier ci-dessus:

Afficher pour chaque collègue de SMITH :

- le nom du collègue

- l’écart de salaire entre celui

Du collègue et celui de SMITH

Deux personnes sont collègues si elles travaillent dans le même département

SMITH n’est pas son propre collègue.

SELECT

c.ename AS "Collegue",

coalesce(c.sal,0) -

coalesce(s.sal,0)

AS "Ecart salaires"

FROM

emp c ---- Comme collègue

JOIN

emp s ---- Comme SMITH

ON

c.deptno = s.deptno

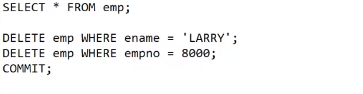
WHERE

s.ename = 'SMITH'

AND

c.ename != 'SMITH'

Quelques Commandes :



--------------------------------------------

**JOINTURE EXTERNE :**

Si on fait cette requete:

------ > le département OPERATIONS n’apparait pas

-----🡪 car pas d’employé

SELECT

e.ename,

d.dname

FROM

emp e

INNER JOIN

dept d

ON

e.deptno = d.deptno;

Pour autre besoin comme afficher les departements sans employé

Alors on fait une **jointure externe** :

RIGHT

SELECT

e.ename,

d.dname

FROM

**emp e RIGHT OUTER JOIN dept d**

ON

e.deptno = d.deptno;

OU LEFT

SELECT

e.ename,

d.dname

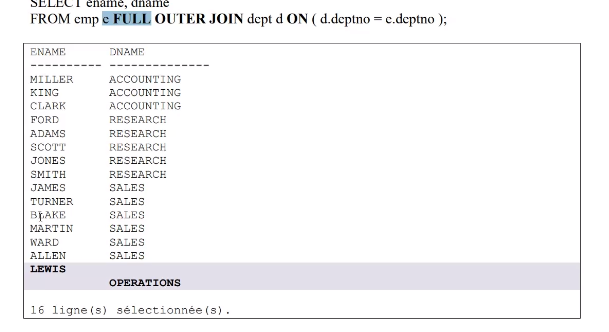
FROM

**dept d LEFT OUTER JOIN emp e**

ON

e.deptno = d.deptno;

**JOINTURE EXTERNE FULL :** Les deux tables sont directrices



Quand on veut mettre deux tables ensembles on utilise 5 opérations :

Produit cartésien, jointure interne, externe droite, gauche et totale

Suivant le besoin => Les employés avec leurs départements

---- > INNER JOIN

En plus les départements sans employé

-----> RIGHT OU LEFT OUTER JOIN

En plus les départements sans département

-----> RIGHT OU LEFT OUTER JOIN

« tout »

---- > FULL OUTER JOIN

Ex11 : APPLICATION JOINTURE EXTERNE :

Afficher pour chaque employé ayant un niveau identifié :

- son nom

- son niveau

Plus les niveaux sans employé. Pour cela, la mention ‘Sans employé’ sera affichée dans la colonne « Nom »

SELECT

Coalesce( e.ename, 'Sans Employe')

as "Nom",

s.grade

as "Grade"

FROM

emp e

RIGHT OUTER JOIN

salgrade s

ON

e.sal BETWEEN s.losal AND s.hisal;

Rendre le contexte moins favorable :

ALTER TABLE emp DROP

CONSTRAINT nn\_ename\_emp;

INSERT INTO emp ( empno, sal)

VALUES ( 9001, 2000 );

Si on passe la commande précédente :

SELECT

Coalesce( e.ename, 'Sans Employe')

as "Nom",

s.grade

as "Grade"

FROM

emp e

RIGHT OUTER JOIN

salgrade s

ON

e.sal BETWEEN s.losal AND s.hisal;

---- > RESULTAT INCOHERENT



DANS ce cas on utilise le **CASE** =>

**CASE : affichage conditionnel :**

SELECT

ename,

sal,

-- debranchement simple

CASE WHEN sal = 3000 THEN ‘<--‘

ELSE NULL

END

AS «  Qui gagne 3000 ? »

FROM

emp ;

SELECT

ename,

sal,

-- debranchement multiple

CASE WHEN sal > 3000 THEN ‘PLUS’

WHEN sal < 3000 THEN ‘MOINS’

WHEN sal = 3000 THEN ‘IDEM’

ELSE NULL

END

AS «  Qui gagne 3000 ? »

FROM

emp ;

SELECT

CASE WHEN e.empno IS NULL THEN ‘Sans Employe’

ELSE

Coalesce( e.ename, Nom Inconnu' )

END

as "Nom",

s.grade

as "Grade"

FROM

emp e

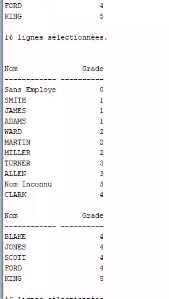
RIGHT OUTER JOIN

salgrade s

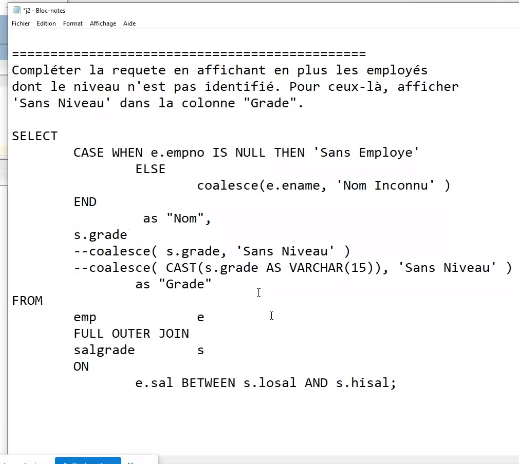
ON

e.sal BETWEEN s.losal AND s.hisal;

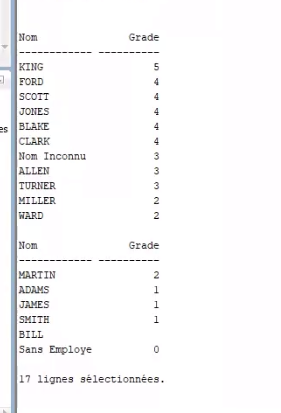
----- > RESULTAT COHÉRENT



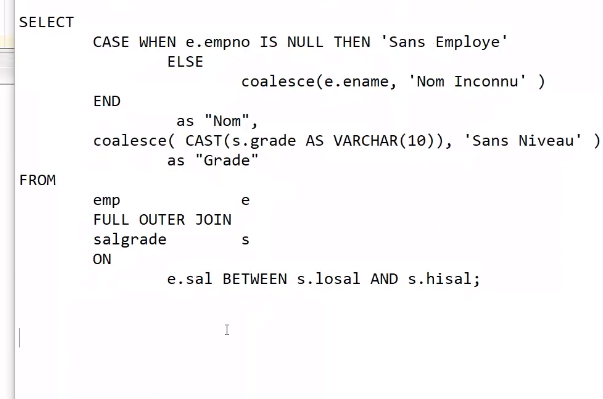
Compléter la requete précedante en affichant en plus les employés dont le niveau n’est pas identifié. Pour ceux-là, affichier ‘Sans Niveau’ dans la colonne « Grade » :



RESULTAT :



Pour avoir un « sans niveau » à coté de BILL : On utilise le CAST pour transformer s.grade en Texte



*On supprime pour continuer les exercices :*

DELETE emp WHERE empno IN ( 9000, 9001);

DELETE salgrade WHERE grade = 0

----------------------------------

EX 12 : Joindre plusieurs tables : Afficher pour chaque employé :

- son nom

- le nom de son département

- son niveau de salaire

SELECT

e.ename,

d.dname,

s.grade

FROM

emp e

JOIN

dept d

ON

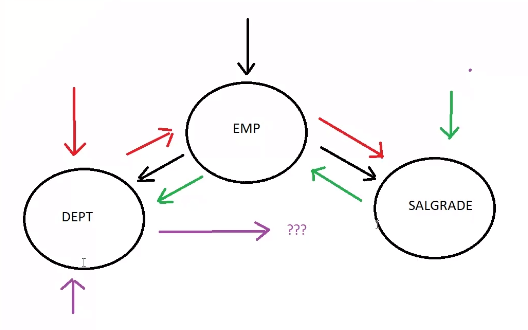
e.deptno = d.deptno

JOIN

salgrade s

ON

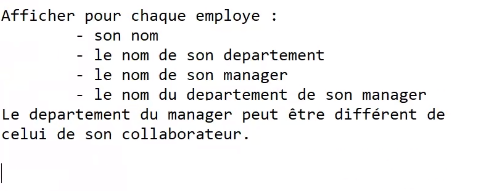
e.sal BETWEEN s.losal AND s.hisal;



ORDRE de JOINTURE entre plusieurs tables ci-dessus

------------------------------------------------------------

Ex 13 :



SELECT

E.ename AS "Employe",

DE.dname AS "Dept.E.",

M.ename AS "Manager",

DM.dname AS "Dept.M."

FROM

emp E ------- employe

JOIN

emp M -------- manager

ON

E.mgr = M.empno

JOIN

dept DE ----- departement de l’employe

ON

E.deptno = DE.deptno

JOIN

dept DM ----- departement du manager

ON

M.deptno = DM.deptno;

**FONCTION DE GROUPE**

**AGREGAT IMPLICITE :**

SELECT count(empno), sum(sal)

FROM emp;

( Une table est un groupe pour le SGBDR / Fonctions de groupe)

EXEMPLE ---- Qui gagne le plus petit salaire de la société ?

SELECT

ename

FROM

emp

WHERE

sal = min ( sal);

ERREUR CAR une approche ligne par ligne et en groupe en meme temps =>

**SOLUTION un SELECT imbriqué** :

SELECT

ename

FROM

emp

WHERE

sal = ( SELECT

min ( sal )

FROM

emp );

DANS WHERE on ne peut pas mettre les fonctions de groupe que en SELECT imbriqué

EXEMPLE : Selection :

SELECT count(empno), sum(sal)

FROM emp;

WHERE job IN ( ‘ANALYST’, ‘CLERK’, ‘SALESMAN’);

------- > Là c’est une sélection d’une partie de la table donc sélection du groupe table

------ Qui du dept 10 gagne le plus petit salaire de son departement ?

SELECT

ename

FROM

emp

WHERE

sal = ( SELECT

min(sal)

FROM

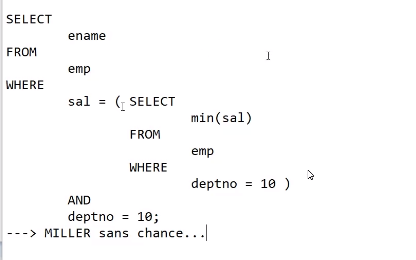
emp

WHERE

deptno = 10) ;

---------- > bonne réponse mais avec de la chance =>

LE CORRECT :



Il FAUT Ajouter le AND, la selection dans le SELECT imbriquée n’est pas vu dans le SELECT mère , donc faut répéter avec le AND

--------------------------

EXEMPLE:



SELECT min(sal), max(sal)

FROM emp

GROUP BY deptno;

------- > OK mais le resultat est peu exploitable, car dans le resultat pas de nbre de département

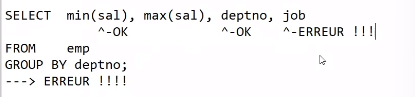
ALORS on ajoute le deptno =>

SELECT min(sal), max(sal), deptno

FROM emp

GROUP BY deptno;

------ > OK et exploitable



Il faut ajouter le job dans GROUPE BY car on est dans une approche de groupe avec les fonctions, mais on change le sens de la requête =>

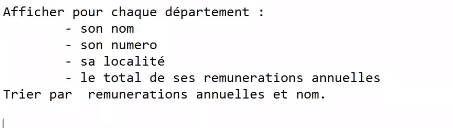
SELECT min(sal), max(sal), deptno, job

FROM emp

GROUP BY deptno, job;

---- > OK

Exercice 14 :



SELECT

d.dname,

d.deptno,

d.loc,

sum( coalesce(e.sal,0)\*12 +

coalesce(e.comm,0) \* 4 ) AS " TOTAL"

FROM

emp e

JOIN

dept d

ON

e.deptno = d.deptno

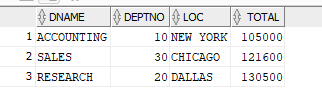
GROUP BY

d.dname, d.deptno, d.loc

ORDER BY

4, 1;

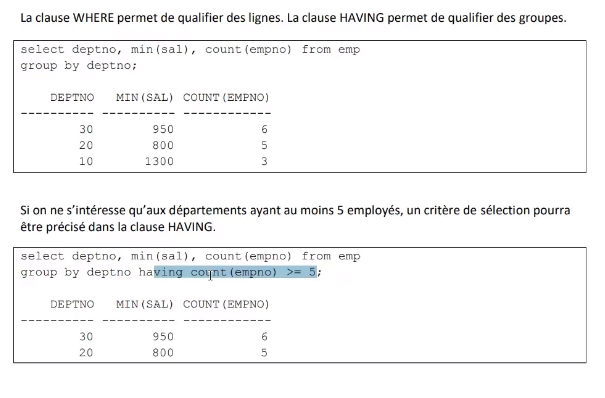
**RESULTAT :**



Le SUM est une fonction de groupe donc il faut utiliser le GROUP BY pour pouvoir l’exécuter

UNE CONDITION SUR UNE FONCTION DE GROUPE AVEC :

**Having sur le GROUP BY ( filtrer ? ) :**



**Notes : 19/04/2023**

**Scripts SQL**

Sur un BLOC NOTE :

On crée un fichier et on l’appelle demo1.sql, on efface le type de fichier ( on enregistre avec option tous les fichiers), on enregistre dans le dossier sql.

On écrit les lignes de commande suivants :

SELECT

\*

FROM

emp

WHERE

deptno=10;

On le lance sur sql developer :

@c:\sql\demo1

**Script interactif : On enregistre sous : demo1i.sql On modifie les lignes :**

WHERE

deptno = &numdept;

Le & indique dans Oracle que c’est une variable =>

Sur sql developer on execute ce script =>

@c:\sql\demo1i

Une fenêtre apparait pour choisir le dept qui nous intéresse.

On modifie le Script =>

CLEAR SCREEN

ACCEPT numdept PROMPT "Saisir un numero de departement S.V.P. :

SELECT

\*

FROM

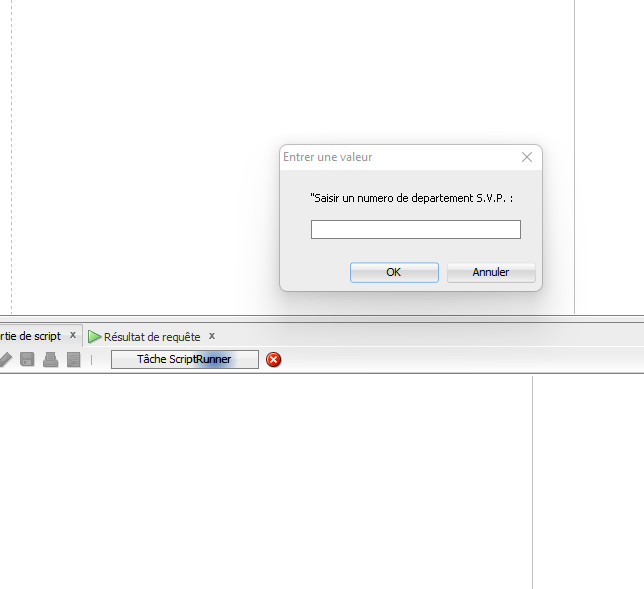
emp

WHERE

deptno = &numdept;

On relance le Script =>

Ce script nettoye ou efface les résultats des variables avant on ajout une phrase sur la fenetre qui apparait :



On ajoute au script une ligne SET VERIFY OFF pour effacer les commandes antérieur ou le script ancienne version :

SET VERIFY OFF

CLEAR SCREEN

ACCEPT numdept PROMPT "Saisir un numero de departement S.V.P. :

SELECT

\*

FROM

emp

WHERE

deptno = &numdept;

**Script orienté Bash**

**On enregistre le fichier sous : demo1b.sql On modifie les lignes :**

Dans WHERE on met un &1 => argument 1

SET VERIFY OFF

CLEAR SCREEN

SELECT

\*

FROM

emp

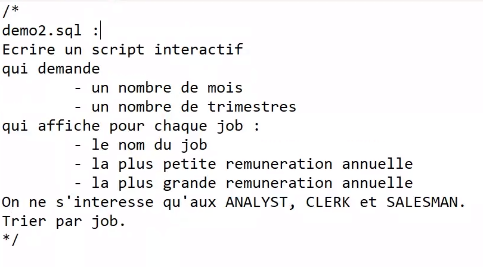
WHERE

deptno = &1;

On lance le script avec un argument 10 ( departement )

@c:\sql\demo1b **10**

Exercice1 :

****

**RESULTAT : Fichier**

demo2.sql

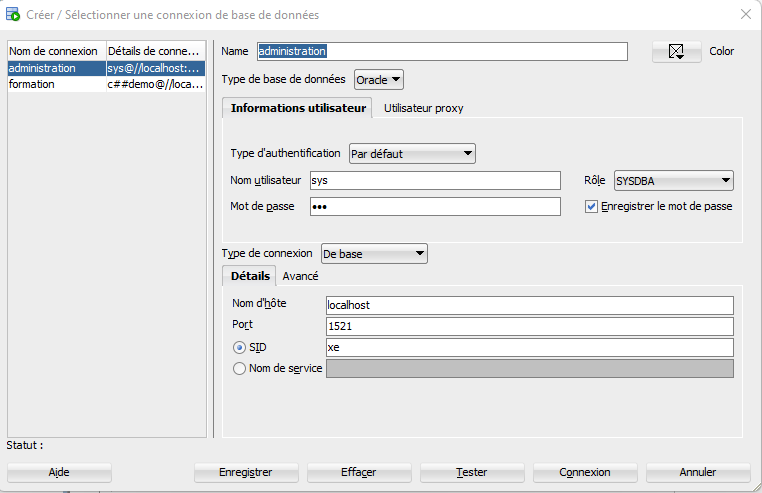
**ADMINISTRATION BDD ORACLE :**

Super utilisateur : SYS

On va lui modifier son mot de passe : ( sur cmd) : On se connecte dans sql en tant que sys ( administrateur) :



On revient vers sqldevelopeur pour paramétrer une nouvelle connexion sur le + vert:



On vérifie sur sqldevelopper sur quelle base de donnée on travaille:

SELECT instance\_name, name

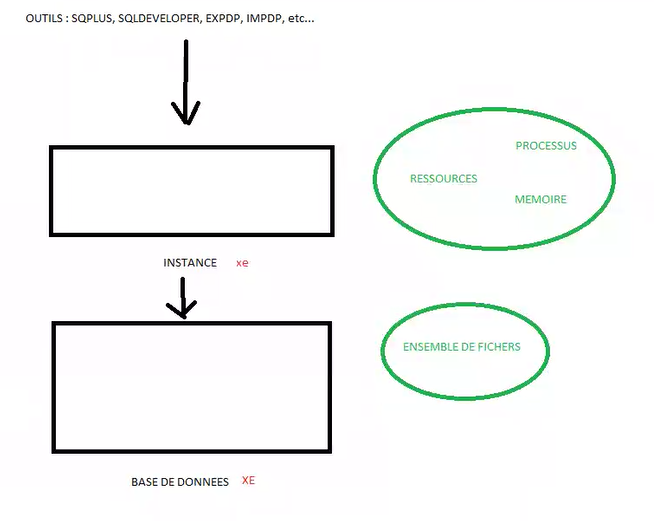
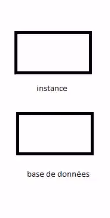
FROM v$instance, v$database;

Résultat :



Instance NAME ( Pilote ) : xe

Base de Donnée Name : xe

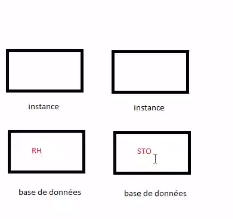
 

**BDD :** ensemble de fichier ( physique

**Pilote dans Oracle : Instance :** Ressources = > processus et mémoire

Les Outils adresse l’instance, l’instance adresse la BDD

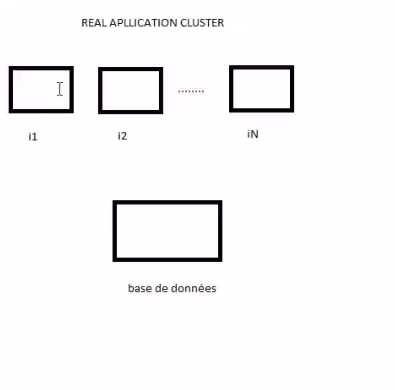
Cas 1 : **une application RH et une application de gestion de Stock :**



Deux gestions différentes de performance des deux applications.

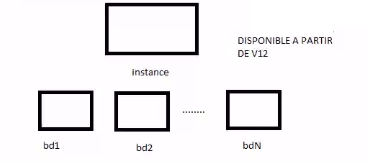
Cas 2 : **REAL APPLICATION CLUSTER ( RAC) :**

3 instances qui accèdent les trois BDD : Système à tolérance de panne :



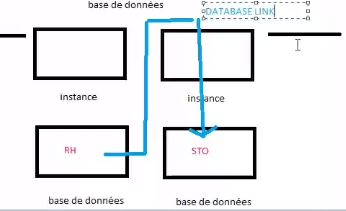
On peut paramétrer : Une instance traite la lecture, une instance pilote la mise à jour etc ..

Cas 3 : **Une instance qui pilote plusieurs BDD à partir de la version 12, avant c’était impossible**

****

On choisi cette possibilité en créant la BDD au début

Cas 4 : **Accèder à une autre BDD à travers l’instance**

****

Accéder d’un BDD RH à une BDD STOCK à travers un Objet qui s’appelle Database Link

**La Base de Donnée : ( ensemble de fichiers) dont les fichiers de données**

Une requête qui liste les fichiers de données de la base de donnée là ou on trouvera tous le définitions et objets de la base de donnée

SELECT con\_id, name

FROM v$datafile;

Ces fichiers sont regroupés en container : 1, 2, 3

1 : racine

2 : seed ( pour générer d’autre container)

3 : premier container utilisateur

Au sein de ces container les fichiers sont aussi regroupés en tablespace :

SELECT con\_id, name

FROM v$tablespace ;

On fait une jointure entre datafile et tablespace :

SELECT

t.con\_id AS container,

t.name AS tablespace,

f.name AS fichier

FROM

v$tablespace t

JOIN

v$datafile f

ON

t.con\_id = f.con\_id

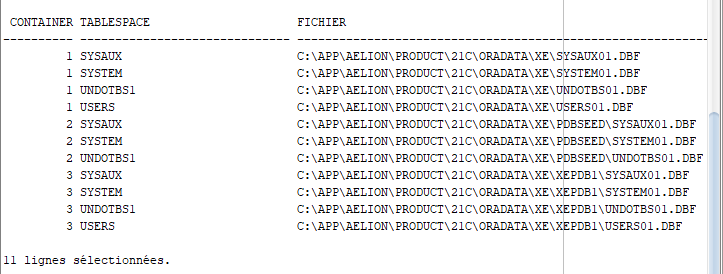
AND

t.ts# = f.ts#

ORDER BY

1,2;

**RESULTATS :**



Quand on crée une table, on se pose la question dans quelle tablespace on range notre table

Tablespace SYSTEM : Dictionnaire de données

Tablespace SYSAUX : Auxiliaires ( statistiques, …)

Tablespace UNDOTBS1 : Transaction

Tablespace USERS : Par défaut pour les utilisateurs

---------------------------------

On crée un TABLESPACE appli, on copie le chemin de la requete précédente

CREATE TABLESPACE appli

DATAFILE ‘C:\APP\AELION\PRODUCT\21C\ORADATA\XE\appli.dbf’

SIZE 10 M ;

On passe la requete précedente on aura comme résultat :



On crée une TABLE projet dans ce TableSpace :

CREATE TABLE projet

(

id NUMBER,

libelle VARCHAR(20)

)

TABLESPACE appli;

Pour chercher ou se trouve cette TABLE =>

SELECT tablespace\_name

FROM user\_tables

WHERE table\_name = 'PROJET';

**RESULTAT :**



Par default ce TABLESPACE est enregistré dans le Container 1 si on ne précise rien.

----------------------------------------------------------------------

**La base de donnée est un ensemble de fichiers :**

**Fichiers de données**

**Fichiers Journaux :** Enregistre tout se qui se passe au niveau de la BDD

**Fichiers de Contrôle**

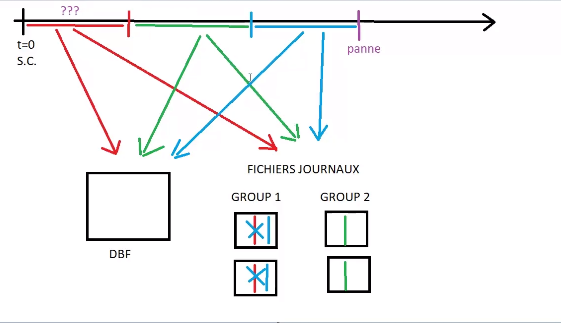
**Fichiers temporaires**

**Fichiers Journaux :**

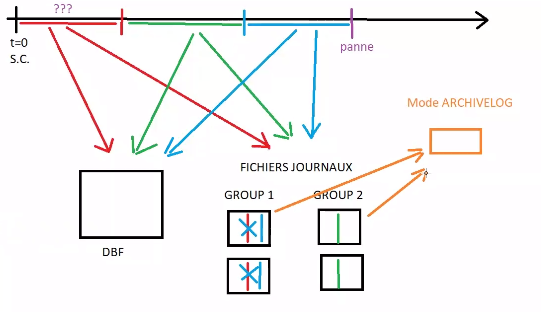
Dans ORACLE pas de nom de fichier ni de noms d’extensions réservés

A t = 0 Tout se qui se passe dans la BDD sont enregistrés dans les fichiers de données sous l’extension DBF et dans les fichiers de journaux qui sont divisés en Groupes ex : Groupe 1 et 2 , et chaque groupe peut être constitué de plusieurs fichiers ou membres

Les fichiers journaux sont de taille fixe, donc quand Groupe 1 est saturé, les enregistrements passe dans le GROUPE 2. Quand le GROUP 2 est Plein, l’enregistrement reprend sur GROUP 1 en écrasant ce qui était avant.



Pour résoudre ce problème on met en place un Mode ARCHIVELOG :



Donc quand l’espace est plein les journaux sont archivés

Exemple sur sqldevelopper :

SELECT

group#, member

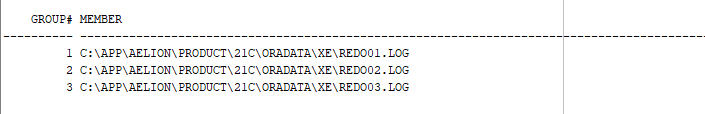
FROM

v$logfile

ORDER BY

1;

RESULTAT



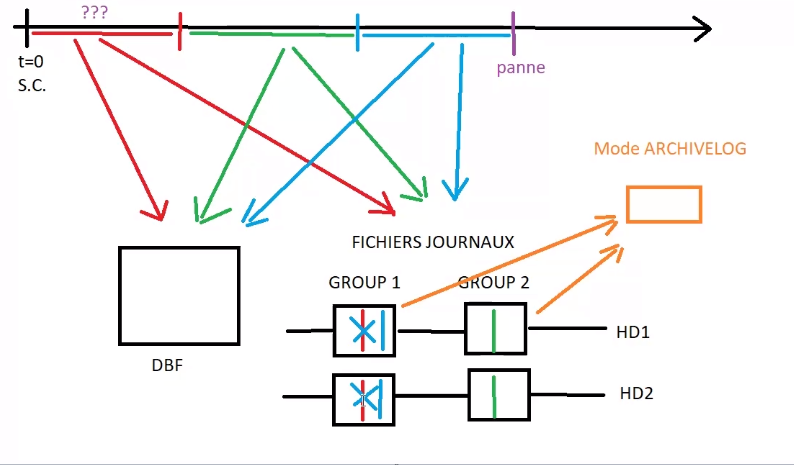
Vérifier s’il y a un mode archive:

SELECT log\_mode FROM v$database;

RESULTAT : NO ARCHIVE :



Deux membres identique par groupe aux cas ou un membre n’est plus dispo. ON les met sur des Disque dur différents. ( MIRRORING)



On va créer des membres et un Groupe 4 :

ALTER DATABASE ADD LOGFILE

GROUP 4

(

'C:\APP\AELION\PRODUCT\21C\ORADATA\XE\RED041.LOG',

'C:\APP\AELION\PRODUCT\21C\ORADATA\XE\RED042.LOG'

)

SIZE 100 M;

Pour voir le nouveau groupe et membre :

SELECT

group#, member

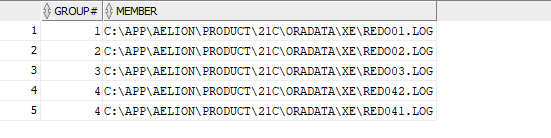
FROM

v$logfile

ORDER BY

1;

RESULTAT:



**FICHIERS DE CONTROLE**

SELECT name FROM v$controlfile;

On trouve deux fichiers de CONTROL :



Les fichiers de contrôle c’est la carte d’identité de la base de données. Ils contiennent la liste de tous les fichiers qui constituent la base. Ils contiennent les propriétés des fichiers de la base de donnée : la date de création et de mises à jours des fichiers

Deux membres identique par groupe aux cas ou un membre n’est plus dispo. ON les met sur des Disque dur différents

Deux fichiers de contrôles identiques minimum pour avoir une haute disponibilité.

**FICHIERS TEMPORAIRES :**

Le contenu est temporaire, mais les fichiers sont présents toujours !!!

Pour les tri ( ORDER BY) , les jointures et autres, c’est une espace de travail pour la base de donnée qui utilise quand la mémoire est saturée ( comme le swap)

Pour les voir :

SELECT

t.con\_id AS container,

t.name AS tablespace,

f.name AS fichier

FROM

v$tablespace t

JOIN

v$tempfile f

ON

t.con\_id = f.con\_id

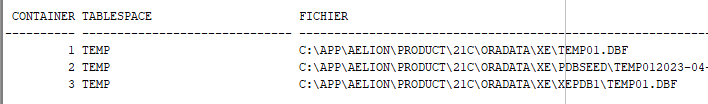
AND

t.ts# = f.ts#

ORDER BY

1,2;

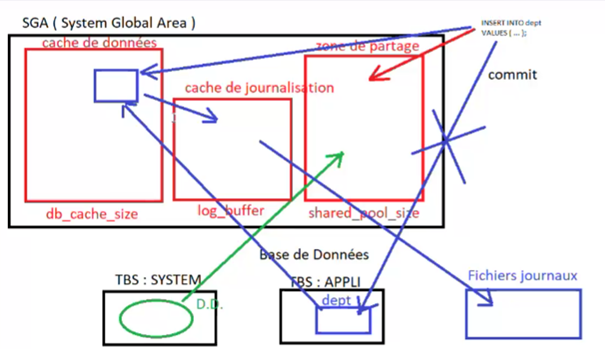
RESULTAT :



Les 4 types de fichiers ne sont pas modifiables, si on essaye de les modifier directement ils seront corrompus

Les fichiers ont des marqueurs spécifiques, on peut jamais déplacer des fichiers d’une BDD à une autre

**INSTANCE :**



Supposons qu’on fait un INSERT INTO dept,

**Dans l’instance il y a une zone de mémoire la SGA ( System Global Area )** : pour insérer et traiter cette opération dans la zone da partage.

**Zone de partage => Cache de données => Cache de Journalisation**

**Zone de partage** sert à accueillir les instructions qu’on passe et analyser ces requêtes.

Oracle veut vérifier avant cette requête, le syntaxe etc, donc Oracle va monter une partie de la BDD ( dictionnaire de donnée) dans la zone de partage.

Un paramètre de l’instance : shared\_pool\_size c’est la taille de la zone de partage :

SHOW PARAMETER shared\_pool\_size

ou

SELECT value FROM v$parameter

WHERE name = 'shared\_pool\_size';

Resultat Zero , c a d demande auto tuning ( ce n‘est pas la valeur zero) , en auto tuning c’est ORACLE qui gère la taille de zone de partage

V$sysstat / V$sesstat / Etc.. Oracle ou l’admin check ces fichiers de stat pour pouvoir gérer la taille de la zone de partage ( et les autres zones ?)

**Le cache de données :**

Oracle monte la table DEPT dans le cache de données et le INSERT se fait dans ce cache pour une question de performance c’est plus rapide, une fois on fait COMMIT les données vont etre enregistrées dans le TBS : APPLI

Commande pour voir la zone cache de données :

SHOW PARAMETER db\_cache\_size

OU

SELECT value FROM v$parameter

WHERE name = 'db\_cache\_size';

Résultat Zéro car ORACLE le gère

**Le cache de Journalisation :**

Enregistre tout ce qui se passe dans le cache de donnée

Pour être transféré ensuite dans les fichiers de journaux

Pour voir ce cash :

SHOW PARAMETER log\_buffer

OU

SELECT value/1024/1024 FROM v$parameter

WHERE name = 'log\_buffer';

( value en octet, on divise deux fois par 1024 pour avoir un résultat en M)

RESULTAT A peu près 6 M :



Instance = mémoire ( SGA) et des processus

SHOW PARAMETER MEMORY\_TARGET

--------- > 0

--------- > le mode AMM n’a pas été activé

SGA plus PGA

( mémoire process utilisateurs)

SHOW PARAMETER sga\_target

------ > 1536 M

------- > Le mode ASMM a été activé

Uniquement le SGA

*Résumé :*

*On travaille avec une base de données. Celle-ci est pilotée par une instance. Une instance c’est de la mémoire (SGA), et des processus qui tournent en tâche de fond. Cette instance va piloter une base de données. La BDD est un ensemble de fichier. Ces fichiers sont :*

*- Les fichiers de données : les objets de la base*

*- Les fichiers journaux : enregistrent les actions sur la base. (Permet de restaurer en cas de panne*

*- Les fichiers de contrôles : CI de la base, liste des fichiers de toute la base (nom, emplacement, taille, date de création, mise à jour)*

*- Les fichiers temporaires (opération de tri)*

*Aucun de ces fichiers n’est éditable (propriétaire ORACLE)*

**On crée un Script :**

Mémoire.sql et on le lance pour verifier quelle mode de mémoire :

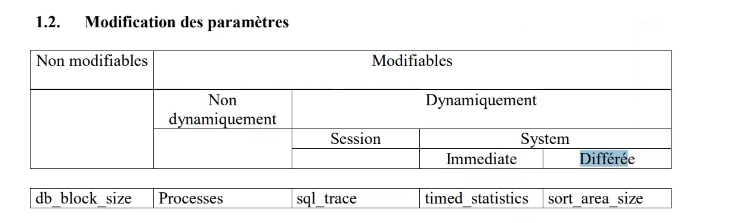
@c:\sql\memoire



---------------------------------------

**MODULE 6**

Des paramètres non modifiables | Modifiables Non dynamiquement et Dynamiquement :



select isses\_modifiable, issys\_modifiable from v$parameter 2> where name = ‘nom\_parametre’;

Ces paramètres se trouvent dans :

pfile ou spfile

**PFILE :** fichier texte donc éditable

**SPFILE :** fichier format propriétaire donc NON éditable => Avantage : On modifie avec commande SQL avec la commande : ALTER SYSTEM, on a 3 SCOPE, mémoire juste pour la session, sur SPFILE c’est pris en compte qu’au redémarrage, et BOTH c a d mémoire + SPFILE

**Sur cmd SQL :**

On tape la commande

Shutdown immediate ------- > plus d’attente

Autres type de Shutdown :

Shutdown normal --------- > attente de la fin des sessions

Shutdown transactional ------ > attente de la fin des transactions

Dernière synchronisation entre la mémoire et le disque

Shutdown abort ------- > Pas de dernière synchronisation donc potentiellement il y aura une perte de donnée

**Pour démarrer en 3 étapes On tape:**

Select status from v$instance ;

----- > ORACLE NOT AVAILABLE

**1 -** startup nomount

-------- > On démarre juste l’instance c.a.d process ( DBWR, LGWR, …) démarrés, allocation de la SGA

Select status from v$instance ;

--------- > STARTED = juste l’instance a été démarrée, la base de donnée n’est pas en ligne donc on ne peut pas créer des tables

**2 -** Alter database mount ;

-------- > lire le fichier de contrôle, c’est là ou l’instance prend connaissance des fichiers de contrôle qu’elle doit gérer

Select status from v$instance ;

------- > MOUNTED : On ne peut toujours pas créer de table car on a tjs pas accès à la base de donnée

**3 -** alter database open ;

--------- > On a accès aux fichiers données et journaux

Select status from v$instance ;

---------- > OPEN c a d tout est cohérent ( on check aussi si tout va bien ) sinon un échec

Pour démarrer en une seule étape : Shutdown immediate

Puis startup tout court

startup

Les trois étapes sont exécuté en une seule commande startup

SI on a eu des problème : sqlplus / as sysdba puis shutdown abort

**Comment savoir si on travail avec un PFILE ou SPFILE ?**

SHOW PARAMETER spfile =>

-------------------- > C:\APP\AELION\PRODUCT\21C\DATABASE\SPFILEXE.ORA

Si pas de résultat c a d on est en PFILE

-----------------------------------------------------

**MODULE 7 :**

**On ajoute un nouveau fichier de donnée dans le TABLESPACE appli**

ALTER TABLESPACE appli ADD

DATAFILE C:\APP\AELION\PRODUCT\21C\ORADATA\XE\appli2.dbf’

SIZE 15 M ;

On vérifie :

SELECT name FROM v$datafile;



On agrandit notre tablespace :

en ajoutant un fichier ( comme dans notre exemple) – Modifier la taille du fichier – Ou on met le fichier en auto extension ( avec une taille maximale)

Taille de tablespace = La somme de taille des fichiers qui constituent le tablespace

-------------------------------------------------------

**Simuler un disque dur par un répertoire : SUR cmd SQL>**

$md C:\APP\AELION\PRODUCT\HD1 ------------ > md = make directory

Objectif : Déplacer les DBFs du tablespace APPLI sur HD1 :

1. Mettre le TABLESPACE offline

ALTER TABLESPACE appli OFFLINE ;

1. Déplacer nos fichiers physiquement :

$move -

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\APPLI.DBF -

C:\APP\AELION\PRODUCT\HD1

On vérifie :

$dir C:\APP\AELION\PRODUCT\HD1

$move -

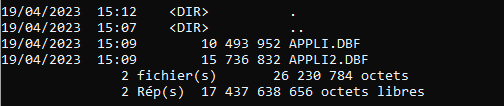
C:\APP\AELION\PRODUCT\21C\ORADATA\XE\APPLI2.DBF -

C:\APP\AELION\PRODUCT\HD1

On vérifie :

$dir C:\APP\AELION\PRODUCT\HD1

Résultats:



1. Déplacer nos fichiers dans la base de donnée ( Comme un COMMIT pour que la base de donnée met à jour son dictionnaire de donnée

**Sur SQLDEVELOPPER :**

ALTER TABLESPACE appli

RENAME DATAFILE

‘C:\APP\AELION\PRODUCT\21C\ORADATA\XE\APPLI.DBF’,

‘C:\APP\AELION\PRODUCT\21C\ORADATA\XE\APPLI2.DBF’

TO

‘C:\APP\AELION\PRODUCT\HD1\APPLI.DBF’,

‘C:\APP\AELION\PRODUCT\HD1\APPLI2.DBF’;

On vérifie : En remettant le TABLESPACE online car c’était OFFLINE

ALTER TABLESPACE appli ONLINE;

/// Si jamais on a eu un problème à lancer SQL+ sur cmd on fait cette commande :

SET ORACLE\_SID=XE ///

**Comment connaitre l’espace libre et l’espace occupé dans un TABLESPACE :**

CREATE TABLE essai ( n NUMBER ) TABLESPACE appli;

--------- > allocation d’espace pour la TABLE => un segment de type TABLE

SELECT segment\_name, segment\_type

FROM dba\_segments

WHERE tablespace\_name = ‘APPLI’ ;

Résultats :



Un segment est constitué d’EXTENTS

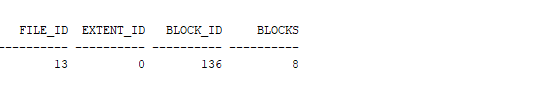
Composition exact du segment ( représentation physique) :

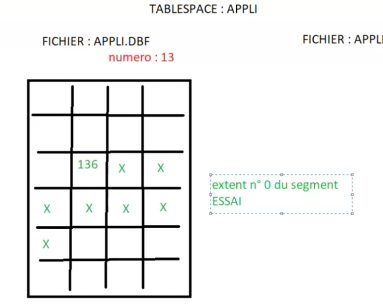
SELECT file\_id, extent\_id, block\_id, blocks

FROM dba\_extents

WHERE segment\_name = ‘ESSAI’;

Résultat: longueur 8,





On ajoute des lignes dans notre table essai : sur SQL + :

BEGIN

FOR i IN 1..10000

LOOP

INSERT INTO essai VALUES ( I );

END LOOP;

END;

/

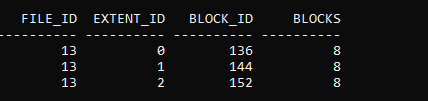
Puis On regarde de nouveau =>

SELECT file\_id, extent\_id, block\_id, blocks

FROM dba\_extents

WHERE segment\_name = ‘ESSAI’;

Resultat :



Oracle a alloué automatiquement de l’espace à la table essai

**ESPACE LIBRE sur TABLESPACE name :**

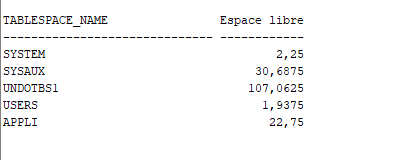
SELECT tablespace\_name,

sum(bytes)/1024/1024 AS "Espace libre"

FROM dba\_free\_space

GROUP BY tablespace\_name;

Resultat :



**ESPACE OCCUPE sur TABLESPACE name :**

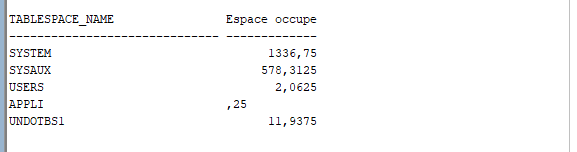
SELECT tablespace\_name,

sum(bytes)/1024/1024 AS "Espace occupe"

FROM dba\_segments

GROUP BY tablespace\_name;

Resultat:



/// Si on eu un problème on check le status du tablespace :

SELECT tablespace\_name, status

FROM dba\_tablespaces;

--------------------------------------------------------------

**Modification sur les fichiers journaux:**

SELECT group#, member FROM v$logfile;

Ajouter un member au groupe 1

ALTER DATABASE ADD LOGFILE MEMBER

'C:\APP\AELION\PRODUCT\21C\ORADATA\XE\REDO01b.LOG'

TO GROUP 1;

On vérifie :

SELECT group#, member FROM v$logfile ORDER BY 1;

-----------------------------

Vérifier le status des différents fichiers log :

SELECT group#, member, status FROM v$logfile ORDER BY 1;

--------------- > nouveau member INVALID



INVALID : 2 Raisons : Ou bien problème ou bien retard de synchronisation

On passe cette commande pour verifier les 4 fichiers et ainsi les synchroniser par la suite !!!

Mieux le faire que d’attendre la synchro au moment de la saturation d’un fichier

ALTER SYSTEM SWITCH LOGFILE;

ALTER SYSTEM SWITCH LOGFILE;

ALTER SYSTEM SWITCH LOGFILE;

ALTER SYSTEM SWITCH LOGFILE;

Puis on check:

SELECT group#, member, status FROM v$logfile ORDER BY 1;

Le nouveau member n’est plus INVALID :



On ne peux pas changer les tailles des fichiers membres, que en créant des nouveaux groupes avec une nouvelle taille ceci écrase ce qui est avant donc il faut déplacer les membres du groupe vers un disk par exemple =>

APPLICATION :

Objectif : Déplacer les membres du groupe 4 sur HD1 :

Sure SQL + :

Shutdown immediate

rollback ;

startup mount

////// SET ORACLE\_SID=XE ou XE est ici le nom de la base à utiliser

Et se connecter normalement avec sqlplus / as sysdba

Sinon shutdown abort puis recommancer les memes étapes puis startup mount

1. On déplace physiquement :

$move -

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\RED041.LOG -

C:\APP\AELION\PRODUCT\HD1

$move -

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\RED042.LOG -

C:\APP\AELION\PRODUCT\HD1

2 - Mise à jour de la base de donnée pour qu’elle prend en compte le déplacement physique :

ALTER DATABASE RENAME FILE

‘C:\APP\AELION\PRODUCT\21C\ORADATA\XE\RED041.LOG’,

‘C:\APP\AELION\PRODUCT\21C\ORADATA\XE\RED042.LOG’;

TO

‘C:\APP\AELION\PRODUCT\HD1\RED041.LOG’,

‘C:\APP\AELION\PRODUCT\HD1\RED042.LOG’;

3 - On démarre la base de donnée pour vérifier qu’il n’y a pas d’erreur et elle démarre :

NB : On a déplacé les fichier avec la BDD montée mais non démarrée

ALTER DATABASE open;

Vérification du status de la BDD :

SELECT STATUS from v$instance;

Startup mount ( sql + ) = Alter database mount ( sqldeveloper)

**Notes : 20/04/2023**

On veut voir les fichiers de contrôle en créant un script :

listefic.sql dans le dossier C:\sql =>

SELECT

con\_id AS "Container",

'CTL' AS"Type",

name AS"Fichier",

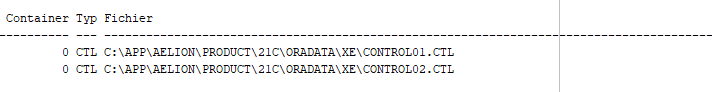
NULL AS"Information"

FROM

v$controlfile;

On l’execute dans sql developer : @c:\sql\listefic

Résultat :



On va compléter ce Script - listefic.sql - pour lister tous nos fichiers de la base de donnée :

Un script listfic2.sql – pour lister des info de plus dans la quatrième colonne

------------------------------------------------------------------------------

**Effectuer des modifications sur les fichiers contrôle :**

CTL déclarés dans le PFILE ou le SPFILE via le parametre control\_files

On regarde nos fichiers de contrôle :

show PARAMETER control\_files

select value from v$parameter

where name = 'control\_files';

RESULTAT:

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL01.CTL,

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL02.CTL

----------------------------------------------------------------------------------------------------

**Commande:**

CREATE CONTROLEFILE :

Cette commande Regénère des fichiers de contrôle dans le cas de perte complète des fichiers de contrôle, pour génere cette commande on passe la commande suivante :

ALTER DATABASE BACKUP CONTROLFILE TO TRACE;

🡺 Il a généré un CREATE CONTROLFILE dans .trc

Ou ?

Show parameter diagnostic\_dest

-------- >C:\app\Aelion\product\21C

--------- > C:\app\Aelion\product\21c\diag\rdbms\xe\xe\trace

Dans le dossier trace on trouve tous les traces générées par les différents Process de ORACLE

Dans le fichier alerte on trouve à la fin :

C:\APP\AELION\PRODUCT\21C\diag\rdbms\xe\xe\trace\xe\_ora\_21272.trc

On ouvre le fichier : xe\_ora\_21272.trc

A l’intérieur on trouve la commande : CREATE CONTROLEFILE en deux FORMES :

1 - CREATE CONTROLFILE REUSE DATABASE "XE" NORESETLOGS NOARCHIVELOG

1. CREATE CONTROLFILE REUSE DATABASE "XE" RESETLOGS NOARCHIVELOG

On récupère celui du NORESETLOGS

--------------------------------------------

**Ajouter un fichier de contrôle :**

**On déclare le fichier Control 03 =>**

ALTER SYSTEM SET control\_files=

'C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL01.CTL',

'C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL02.CTL',

'C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL03.CTL'

SCOPE=SPFILE;

**Sur cmd SQL + on de connecte en admin : sqlplus / as sysdba :**

On arrête la base de donnée

Shutdown immediate

Puis on va créer le fichier control 03 en copiant le fichier control 01 :

$copy –

C :\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL01.CTL –

C :\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL03.CTL

Puis

startup

Sur sql developer on vérifie l’ajout du fichier control 03 avec notre script listefic2 :

@c:\sql\listefic2

Resultat :



Exercice : Déplacer les fichiers de contrôle sur

C:\APP\AELION\PRODUCT\HD1:

On declare le déplacement des fichier dans le nouveau répertoire :

ALTER SYSTEM SET control\_files=

'C:\APP\AELION\PRODUCT\HD1\CONTROL01.CTL',

‘C:\APP\AELION\PRODUCT\HD1\CONTROL02.CTL',

‘C:\APP\AELION\PRODUCT\HD1'\CONTROL03.CTL'

SCOPE=SPFILE;

**Sur cmd sql+ on déplace physiquement :**

$move -

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL01.CTL -

C:\APP\AELION\PRODUCT\HD1

$move -

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL02.CTL -

C:\APP\AELION\PRODUCT\HD1

$move -

C:\APP\AELION\PRODUCT\21C\ORADATA\XE\CONTROL03.CTL

C:\APP\AELION\PRODUCT\HD1

**On vérifie le répertoire HD1 :**

$dir -

C:\APP\AELION\PRODUCT\HD1

Puis on fait un :

startup

Sur sqldeveloper on lacnce le script :

@c:\sql\listefic2

On remarquera que les trois fichiers contrôles sont déplacés

Des commandes sur cmd sql+ pour debuger //// show parameter control\_files - Alter database mount – Alter database open

---------------------------------------------

**Les fichiers JOUNRAUX**

Ou archiver les fichiers journaux ?

--------- > Dans le répertoire FRA ( Fast Recovery Area )

Il accueillera tous les fichiers relatifs aux sauvegardes et restaurations

**Sur cmd sql+ :**

On crée un dossier FRA ( on peut le nommer différemment)

$md C:\ APP\AELION\PRODUCT\FRA

On peut avoir plusieurs base de données qui ont le même Archive File Journaux =>

On déclare la taille de la part de la database allouée dans le fichier FRA et la destination de tous les recovery fichiers dans cette databse dans le même fichier FRA,

ALTER SYSTEM SET

db\_recovery\_file\_dest\_size=4G

db\_recovery\_file\_dest= ‘C:\ APP\AELION\PRODUCT\FRA’ ;

Activer l’archivage c.à.d. passer en mode ARCHIVELOG : Toujours sur cmd sql+

Shutdown immediate

Startup mount

Alter database archivelog;

Alter database open;

Sous Windows on ouvre le dossier :

C:\app\Aelion\product\FRA\XE\ARCHIVELOG\2023\_04\_20

Il est vide donc on passe les commandes sur sql+

Alter system switch logfile;

Alter system switch logfile;

Alter system switch logfile;

Alter system switch logfile;

Alter system switch logfile;

On regarde de nouveau le dossier :

C:\app\Aelion\product\FRA\XE\ARCHIVELOG\2023\_04\_20

On trouvera nos fichiers log

**On va simuler des pannes pour faire des Restauration**

Au début comme prérequis =>

Vérifier si la base de donnée est en mode archive :

Archive log list

Exemple1 : Restauration d’un DBF à partir d’une sauvegarde à froid.

CREATE TABLESPACE crash1

DATAFILE 'C:\APP\AELION\PRODUCT\HD1\crash1.dbf'

SIZE 10 M;

Puis

CREATE TABLE democrash1

( c VARCHAR(100) )

TABLESPACE crash1;

Puis

INSERT INTO democrash1

VALUES ( 'Avant sauvegarde' );

COMMIT;

Puis sauvegarde à froid sur **sql+**:

Shutdown immediate

On copie notre fichier en une autre extension ( à titre d’exemple)

$copy -

C:\APP\AELION\PRODUCT\HD1\crash1.dbf -

C:\APP\AELION\PRODUCT\HD1\crash1.sau

Puis

startup

Puis on passe cette commande sur sql developer ou sql +

INSERT INTO democrash1

VALUES ( 'Avant sauvegarde' );

COMMIT;

On arrete la base :

Shutdown immediate

On simule la panne :

$erase -

C:\APP\AELION\PRODUCT\HD1\crash1.dbf

Startup

---------- > instance démarrée ( étape PFILE ou SPFILE)

---------- > base de donnée montée ( accès fichier controle réussi)

----------- > ECHEC à l’ouverture de la BDD

Pour régler le problème on recopie notre fichier qu’on a sauvegardé :

$copy -

C:\APP\AELION\PRODUCT\HD1\crash1.sau -

C:\APP\AELION\PRODUCT\HD1\crash1.dbf

Puis on essaie d’ouvrir notre base de donnée :

alter database open;

----------- > ECHEC =>



Donc il faut chercher les fichiers journaux qui sont nécessaire à la reconstruction du fichier donnée :

alter database recover automatic database ;

alter database open;

------------ > OK

Pour lire nos fichiers et vérifier :

select \* from democrash1;

Exemple2 : Restauration d’un DBF à partir d’une sauvegarde à chaud.

**Sur sql+ Ou sur sqldeveoper**

CREATE TABLESPACE crash2

DATAFILE 'C:\APP\AELION\PRODUCT\HD1\crash2.dbf'

SIZE 10 M;

Puis

CREATE TABLE democrash2

( c VARCHAR(100) )

TABLESPACE crash2;

Puis

INSERT INTO democrash2

VALUES ( 'Avant début sauvegarde a chaud' );

COMMIT;

On va indiquer à la base qu’on va débuter une sauvegarde à chaud :

ALTER TABLESPACE crash2 BEGIN BACKUP ;

Puis

INSERT INTO democrash2

VALUES ( 'Après début sauvegarde a chaud' );

COMMIT;

Puis on fait la sauvegarde physiquement :

$copy -

C:\APP\AELION\PRODUCT\HD1\crash2.dbf -

C:\APP\AELION\PRODUCT\HD1\crash2.sau

Puis:

INSERT INTO democrash2

VALUES ( 'Avant fin sauvegarde a chaud' );

COMMIT;

Puis

ALTER TABLESPACE crash2 END BACKUP;

Puis

INSERT INTO democrash2

VALUES ( 'Après fin sauvegarde a chaud' );

COMMIT;

Puis:

Shutdown immediate

Puis:

$erase -

C:\APP\AELION\PRODUCT\HD1\crash2.dbf

Puis:

Startup

---------- > instance démarrée ( étape PFILE ou SPFILE)

---------- > base de donnée montée ( accès fichier controle réussi)

----------- > ECHEC à l’ouverture de la BDD

Pour régler le problème on recopie notre fichier qu’on a sauvegardé :

$copy -

C:\APP\AELION\PRODUCT\HD1\crash2.sau -

C:\APP\AELION\PRODUCT\HD1\crash2.dbf

alter database open;

----------- > ECHEC =>



Donc il faut chercher les fichiers journaux qui sont nécessaire à la reconstruction du fichier donnée :

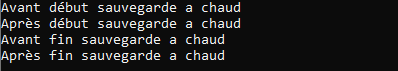
alter database recover automatic database ;

alter database open;

------------ > OK

Pour lire nos fichiers et vérifier :

select \* from democrash2;



NB: Les données qu’on restaurent qui n’étaient pas sauvegardées comme les insertions des lignes après la copie de sauvegarde et après la fin de sauvegarde sont enregistrées en action dans les journaux et c’est comme ça qu’elles seront restaurées après la récupération de la copie de sauvegarde puis

Exemple3 : Restauration d’un DBF à partir sans sauvegarde :

Dans sql+

CREATE TABLESPACE crash3

DATAFILE 'C:\APP\AELION\PRODUCT\HD1\crash3.dbf'

SIZE 10 M;

CREATE TABLE democrash3

( c VARCHAR(100) )

TABLESPACE crash3;

INSERT INTO democrash3

VALUES ( 'Avant panne' );

COMMIT;

On simule la panne : (Shutdown sans sauvegarde)

Shutdown immediate

Puis:

$erase -

C:\APP\AELION\PRODUCT\HD1\crash3.dbf

Puis:

Startup

---------- > instance démarrée ( étape PFILE ou SPFILE)

---------- > base de donnée montée ( accès fichier controle réussi)

----------- > ECHEC à l’ouverture de la BDD

**SOLUTION :**

On ne peut pas exercer les fichiers journaux sur le vide normalement on le fait sur le dbf restauré

**DEUX VARIANTES ou CAS** :

**Variante 1 :** désignation du fichier par son numéro ( 17) et non plus par son nom

**Variante 2** : mettre temporairement offline un fichier ( par exemple si on a 1000 fichiers dans notre BDD et on a eu ce crush avec un seul fichier et les fichiers journaux qui lui appartiennent ne sont pas dispo, mais on ne veut pas bloquer toute la base à cause d’un seul fichier !!

POUR LA VARIANTE 2 :

ALTER DATABASE DATAFILE 17 OFFLINE;

Donc on peut ouvrir la base :

ALTER DATABASE OPEN ;

La base de donnée est ouverte sans que le fichier 17 soit restauré

VARIANTE 1 :

On crée le fichier vierge dbf

ALTER DATABASE CREATE DATAFILE 17 ;

On restaure les fichiers journaux sur ce fichier seulement car la database est déjà ouverte :

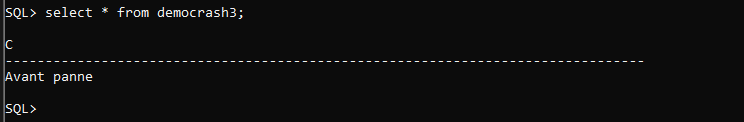
ALTER DATABASE RECOVER AUTOMATIC DATAFILE 17 ;

On remet le fichier 17 online :

ALTER DATABASE DATAFILE 17 ONLINE;

On vérifie:

select \* from democrash3;



ATTENTION: Ce cas là est possible que si on dispose de toutes les données de journalisation depuis la création du fichier. ( On peut verifier ceci en regardant les fichiers journaux et on regarde si leur date est juste antérieur à la création du fichier)

Bonne pratique : Ce qu’il faut faire c’est des plans de sauvegarde à chaud pour ne jamais arriver au 3 ème cas.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**SIMULER UNE PANNE GENERALE :**

**Simuler la perte d’un disque sur lequel se trouveront des CTLs, des RDOs et des DBFs.**

On crée un répertoire qui simule un disque :

$md C:\APP\AELION\PRODUCT\HD2

On ajoute un quatrième CTL sur HD2 :

ALTER SYSTEM SET control\_files=

'C:\APP\AELION\PRODUCT\HD1\CONTROL01.CTL',

‘C:\APP\AELION\PRODUCT\HD1\CONTROL02.CTL',

‘C:\APP\AELION\PRODUCT\HD1\CONTROL03.CTL’,

‘C:\APP\AELION\PRODUCT\HD2\CONTROL04.CTL'

SCOPE=SPFILE;

Shutdown immediate

$copy -

C:\APP\AELION\PRODUCT\HD1\CONTROL03.CTL -

C:\APP\AELION\PRODUCT\HD2\CONTROL04.CTL

Startup

On crée 2 groupes de 2 membres chacun un sur HD1 un sur HD2 ( On suit le prérequis de la bonne pratique en créant des membres sur HD1 et HD2 pour qu’on puisse après simuler la panne générale )

ALTER DATABASE ADD LOGFILE

GROUP 10

(

'C:\APP\AELION\PRODUCT\HD1\REDO10a.RDO',

‘C:\APP\AELION\PRODUCT\HD2\ REDO10b.RDO'

) SIZE 100 M,

GROUP 11

( 'C:\APP\AELION\PRODUCT\HD1\REDO11a.RDO',

‘C:\APP\AELION\PRODUCT\HD2\ REDO11b.RDO'

) SIZE 100 M;

On fait un nettoyage : On supprime les anciens groupes :

SELECT group#, status FROM v$log;

Résultat:



On ne peut supprimer que les GROUP innactive : !!!!!!!!!!!!!!!!!!

ALTER DATABASE DROP LOGFILE GROUP 1, GROUP 3, GROUP 4;

SELECT group#, status FROM v$log;

Résultat:



GROUP 2 --- > CURRENT

ALTER SYSTEM SWITCH LOGFILE ;

GROUP 2 -- > ACTIVE

ALTER SYSTEM FLUSH BUFFER\_CACHE ;

GROUP 2 devient inactive , on essai de le supprimer =>

ALTER DATABASE DROP LOGFILE GROUP 2;

---------- > ERREUR

SHUTDOWN IMMEDIATE

STARTUP

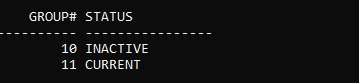
On essaie de nouveau de supprimer le GROUP 2

ALTER DATABASE DROP LOGFILE GROUP 2;

On regarde :

SELECT group#, status FROM v$log;

On aura toujours 2 GROUP un Innactive et un Current



On crée des données sur le HD2

CREATE TABLESPACE crash4

DATAFILE 'C:\APP\AELION\PRODUCT\HD2\crash4.dbf'

SIZE 10 M;

CREATE TABLE democrash4

( c VARCHAR(100) )

TABLESPACE crash4;

INSERT INTO democrash4

VALUES (‘Essai' );

COMMIT;

On fait :

Shutdown immediate

On commence la simulation de panne en détruisant le disk HD2 :

PERTE de HD2 , RESTAURER en HD3

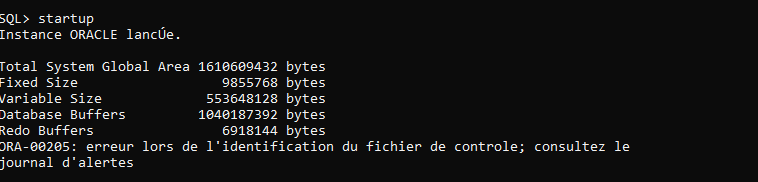
$rd / s / q C:\APP\AELION\PRODUCT\HD2

On vérifie la suppression de HD2 :

$dir C :\APP\AELION\PRODUCT

On tente de redémarrer:

Startup



--------- > instance démarée

--------- > ECHEC au montage

On regarde **le fichier journal d’alerte** comme mentionné dans le message d’erreur :

ed C:\app\Aelion\product\21c\diag\rdbms\xe\xe\trace\alert\_xe.log

**SOLUTION :**

On crée un nouveau disk HD3

$md C :\APP\AELION\PRODUCT\HD3

On copie le fichier CONTROL 03 en 04 sur le HD3 physiquement

$copy -

C:\APP\AELION\PRODUCT\HD1\CONTROL03.CTL -

C:\APP\AELION\PRODUCT\HD3\CONTROL04.CTL

On declare le fichier CONTROL 04 à la base de donnée

ALTER SYSTEM SET control\_files=

'C:\APP\AELION\PRODUCT\HD1\CONTROL01.CTL',

‘C:\APP\AELION\PRODUCT\HD1\CONTROL02.CTL',

‘C:\APP\AELION\PRODUCT\HD1\CONTROL03.CTL’,

‘C:\APP\AELION\PRODUCT\HD3\CONTROL04.CTL' --- Nouveau

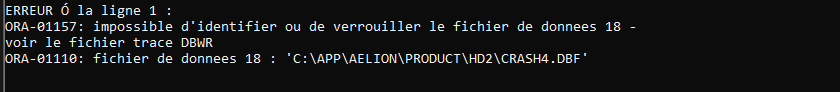
SCOPE=SPFILE;

PUIS

Shutdown immediate

PUIS

Startup



------- > instance démarrée

------- > bd montée ( problème CTL resolu )

--------- > ECHEC à l’ouverture

On met le fichier 18 offline pour ouvrir la Database

ALTER DATABASE DATAFILE 18 OFFLINE ;

Puis

ALTER DATABASE OPEN ;

On crée un fichier de donnée 18 vierge en modifiant le PATH

ALTER DATABASE CREATE DATAFILE 18

AS ‘C:\APP\AELION\PRODUCT\HD3\crash4.dbf’;

On reconstruit notre fichier de donnée 18 en récupérant le contenu des journaux sur HD1

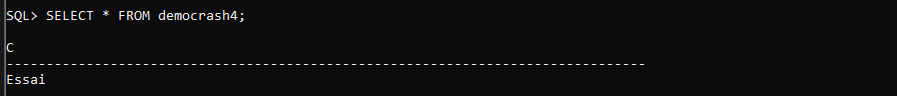
ALTER DATABASE RECOVER AUTOMATIC DATAFILE 18 ;

On remet en ligne le fichier donnée 18

ALTER DATABASE DATAFILE 18 ONLINE ;

On vérifie :

SELECT \* FROM democrash4;



NB: ON n’a pas réparé les JOURNAUX, On les a juste utilisé ( ceux sur HD1) pour reconstruire le fichier de donnée avec la commande recover automatic

On regarde le statut des fichiers log:

SELECT member, status FROM v$logfile;

On doit recréer les fichiers log sur HD3

SHUTDOWN IMMEDIATE

Puis

$copy -

C:\APP\AELION\PRODUCT\HD1\REDO10A.RDO -

C:\APP\AELION\PRODUCT\HD3\REDO10B.RDO

$copy -

C:\APP\AELION\PRODUCT\HD1\REDO11A.RDO -

C:\APP\AELION\PRODUCT\HD3\REDO11B.RDO

On déclare ces fichiers de LOG au début on fait un startup :

STARTUP MOUNT

Puis on déclare avec ALTER :

ALTER DATABASE RENAME FILE

‘C:\APP\AELION\PRODUCT\HD2\ REDO10B.RDO’, ----- Attention il y a un espace \ REDO10…

‘C:\APP\AELION\PRODUCT\HD2\ REDO11B.RDO’ ----- Attention il y a un espace \ REDO11…

TO

‘C:\APP\AELION\PRODUCT\HD3\REDO10B.RDO’,

‘C:\APP\AELION\PRODUCT\HD3\REDO11B.RDO’ ;

On démarre la Database:

ALTER DATABASE OPEN;

On relance le script @c:\sql\listefic2 sur sqldeveloper pour vérifier les fichiers de la BDD on trouvera ainsi les fichiers log dans HD3, les données crash4, le fichier de control 04