Titre: Oracle Database - Le premier tiers de l'architecture Cocktail
Version: 1.4
Dernière modification: 2011/06/22 18:40
Auteur: Aurélien Minet <aurelien dot minet at cocktail dot org>
Statut: version finale

Remarques:

- les pré-requis sont de connaître le SQL, le shell et avoir la maîtrise de l'OS utilisé (ce document est orienté GNU/Linux et bash)
- le document est "orienté" 11gR2
  - certaines fonctionnalités n'existent pas en 9i
  - la 10gR2 (avec patchset 3: 10.2.0.4) est la version minimale supportée
  - certaines choses ont évoluées en 11g
  - la version recommandée est la 11gR2
    (voir cette page pour le changement de version)
- ce n'est qu'un document (orienté Cocktail):
  - et non une formation complète sur Oracle
  - il faut manipuler, tester, pratiquer pour être efficace !
  - il peut contenir des erreurs (je ne suis pas expert Oracle)
- au niveau de la typographie: le code sql, les lignes de commandes, le contenu des fichiers sont en italique
- la dernière version de ce document est disponible ici

Oracle Database

Le premier tiers de l'architecture Cocktail

Historique des versions:

date	version	observations
25/05/2008	1.0	version initiale
18/10/2010	1.2	Quelques de corrections, ajout, reformulations principalement pour la 11gR2 (init.ora) et pour l'environnement shell.
22/06/2011	1.4	Corrections et mises à jours mineures.

Objectif

avoir les connaissances minimales pour l'exploitation d'Oracle afin d'assumer le rôle de DBA

- Assimiler les concepts Oracle
- Comprendre le fonctionnement du SGBDR
- Pouvoir créer une base Oracle
- Savoir comment sauvegarder, dupliquer, restaurer
- Avoir des notions relatives à la "sécurité Oracle"
- Connaître certains outils de gestion des bases Oracle
- Être au fait des pratiques et méthodes d'exploitation

Pour simplifier: avoir "des billes" pour appréhender Oracle

Introduction

Que signifie être Administrateur de Base de données ?
(ie: être DBA Oracle)

- Installer Oracle (le moteur SGBDR)
- Créer, configurer des bases
- Créer, gérer les tablespaces et les comptes
- Assurer l'intégrité, la disponibilité des données
- Optimiser l'environnement d'exécution, au niveau du système et des instances
- Analyser, surveiller, diagnostiquer, collaborer ...

1. Installation

Oracle Database c'est quoi ?

- un système de gestion de base de données relationnelle
- logiciel propriétaire et payant
- toujours à la pointe :
    - complet
    - riche
- professionnel et donc pas forcément simple (le pourquoi de cette présentation ?) mais il tend à se simplifier.
- le SGBDR le plus vendu (chiffre d'affaire) mais pas forcément le plus utilisé (nombre de bases)

L'installation d'une version d'Oracle est:
(aussi appelée noyau / kernel)

- bien documentée: voir le portail pour la 11gR2
- simple et basique: pas à pas
- spécifique pour chaque OS !
- téléchargeable librement (nécessite un compte sur OTN)

mais:

- pour les updates, patchs il faut un accès Metalink (payer la maintenance annuelle)
- pour des configurations particulières (hugepages...) la documentation est sur MOS (anciennement Metalink) ou en partie sur le Web (ex: puschitz.com)

La grille de compatibilité entre les versions du SGBDR pour Linux x86-32 et pour Linux x86-64.

L'installation du sgbdr Oracle passe par des pré-requis :

voici un résumé de ce que l'on trouve dans la documentation Oracle:

- matériels :
  - espace disque : 10gr2 ~1.5 Go / 9ir2 2.5Go / 11g ~3.8Go
  - en fonctions des options
  - pendant l'installation espace temporaire ~400Mo en plus des fichiers d'installation !
  - prévoir un système de fichier / volume spécifique
  - partition de swap (pas besoin d'avoir le double de la RAM)
  - de la RAM: 1 Go au minimum (varie suivant le volume de données)
  - CPU (fonction de la licence)
  - disques rapides 10K ou 15K tr/m avec redondance

- Kernel: paramétrage spécifique via /etc/sysctl.conf

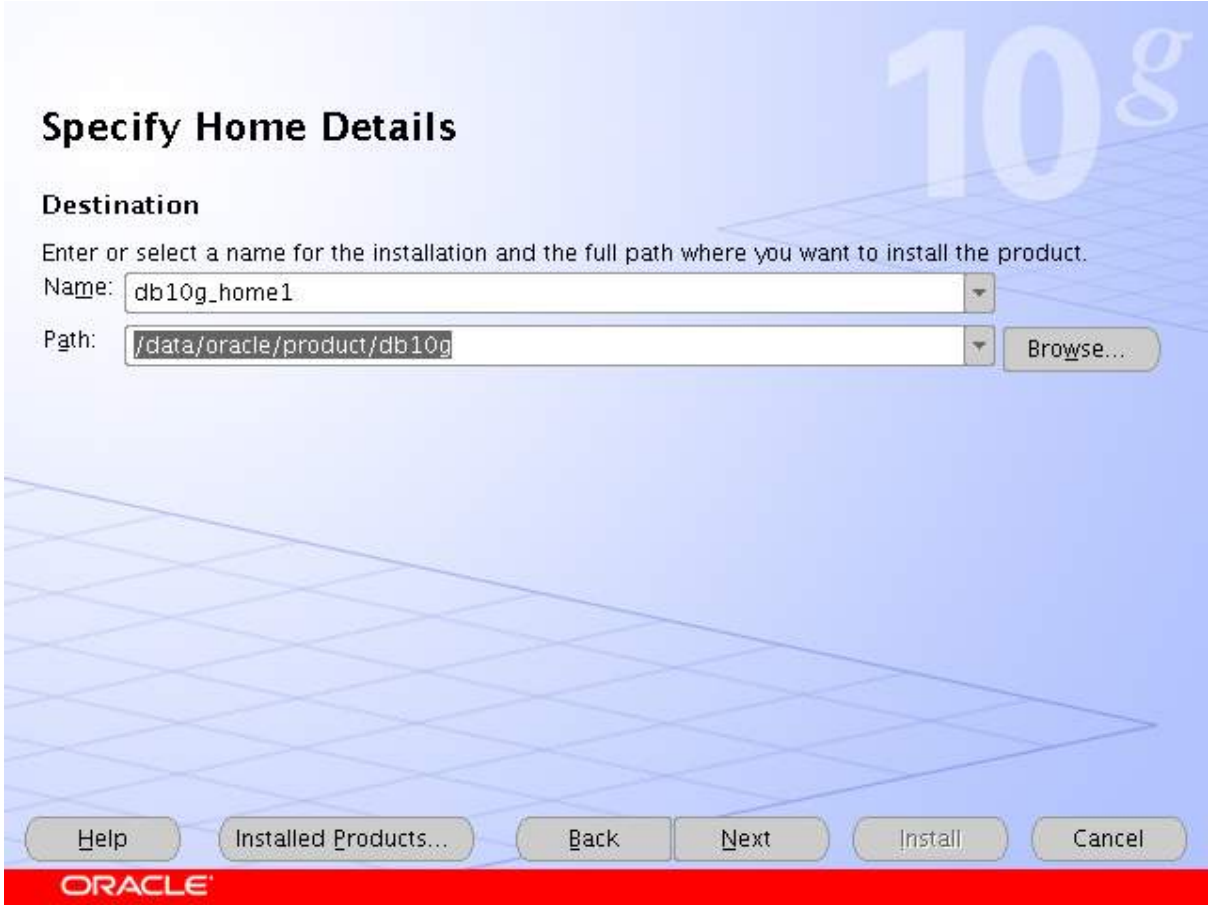
```
kernel.shmall = 2097152
kernel.shmmax = 536870912
kernel.shmmni = 4096
kernel.sem = 250 32000 100 128
fs.file-max = 65536
net.ipv4.ip_local_port_range=1024 65000
net.ipv4.tcp_rmem=4096 87380 8388608
net.ipv4.tcp_wmem=4096 65536 8388608
net.ipv4.tcp_mem=4096 4096 409
```

- OS:
  - différents packages (glibc, gcc, libaio, make ....)
  - groupes: dba, oinstall (oper)
  - users: oracle (ora9i, ora102 ...)
  - configuration pour les users dba
  - modification de /etc/security/limits.conf & /etc/pam.d/login
  - bash_profile: ulimits et exports ORACLE_HOME, ORACLE_... PATH (non nécessaire pour l'installation elle même)
  - systèmes de fichiers: bases, sauvegarde
  - ouverture de ports (firewall, acl)

L'installation en elle même :

- télécharger sur OTN
- Nouveauté: la 11gR2 Patchset 1 (11.2.0.2) est une version intégrale (et non une simple update), mais est à télécharger sur MOS.
- décompresser avec unzip/cpio/tar
- exécuter l'installeur : *Disk1/install/runInstaller* (besoin de X11)
  (*Disk1/install/runInstaller -silent -responseFile /.../...rsp*)

Voici l'une des écrans de l'interface d'installation, ici il faut faire attention au chemin (patch) que le runInstaller aura choisi, le modifier pour l'adapter à vos besoins.



L'installeur est OUI
(Oracle Universal Installer)

il a besoin de 3 répertoires:

- ORACLE_BASE: racine de l'arborescence Oracle
  ex: /u00/oracle
- oraInventory: entrepôt où les installations sont référencées
  généralement c'est oracle_base/oraInventory
- ORACLE_HOME: emplacement de l'installation
  ex: oracle_base/product/11.2.0/db_1

Partie spécifique à oracle dans le fichier ~/.bash_profile (user dba):

```
export ORACLE_HOME=/u00/oracle/product/11.2.0/db_1
export LD_LIBRARY_PATH= $ORACLE_HOME/lib:/lib:/usr/lib:/usr/
local/lib
export LANG=en_US.UTF-8
# pour l'exploitation
export NLS_LANG=american_AMERICA.AL32UTF8
# pour passer les patch
export NLS_LANG=french_FRANCE.AL32UTF8
export ORACLE_SID=test
export PATH=$PATH:$HOME/bin:$ORACLE_HOME/bin

ulimit -u 16384 -n 63536

PS1='\[\033[01;32m\]\u@\h\[\033[01;31m\]
[${ORACLE_SID}]\[\033[01;36m\] \w \$\[\033[00m\] '
```

Le prompt correspondant à la déclaration de PS1:

```
oracle@newgest [test] ~ $
```

il affiche en rouge et entre crochets la valeur d'ORACLE_SID donc l'instance sur laquelle potentiellement on travaille.

Remarques sur le choix des répertoires et de leur organisation
il y a plusieurs possibilités notamment l'OFA et "l'ASA"

L'OFA (Optimal Flexible Architecture) recommandée par Oracle :

- /ptmnt/type/user correspond à ORACLE_BASE, ex: /u01/app/oracle
- /ptmnt/type/user/products/v pour les installations, ex: /u01/app/oracle/products/11.2.0/
- /ptmnt/type/user/admin/sid/[adhoc|arch|?dump|create|exp|pfile] , ex: /u01/app/oracle/admin/TEST/adhoc (scripts sql spécifiques à la base)
- /ptmnt/oradata/sid fichier de la base (.ctl .rdo .dbf), ex: /ptmnt/oradata/TEST/sysaux01.dbf

un ptmnt (point de montage) correspond a un disque

L'ASA (architecture simplifiée actuelle): organisation personnelle

- /stru/[databases|oracle|backup] chaque stru correspond à une unité de stockage et chaque sous répertoire un filesystem/lun/vdisk.
- /stru/oracle/products/V/db_versionX/revision peut avoir plusieurs oracle_home: updates et patchs spécifiques
- /stru/databases/SID/[oradata|logs/?dump|arch|scripts]
  tout les fichiers de la base sont dans les sous-répertoires de SID
  il peut y a avoir plusieurs ptmnt pour le multiplexage :
  /u00/databases/TEST/oradata/redo01a.log
  /u01/databases/TEST/oradata/redo01b.log
- /stru/backup/[uman/rman/exp] pour les différentes sauvegardes.

un stru (storage unit) est un contrôleur RAID, un SAN ...

La différence entre ces 2 solutions est principalement historique et matérielle. Au début d'Oracle il n'y avait pas de RAID et encore moins de SAN, le matériel coûtait cher. Ainsi on passe de la gestion disque par disque à une gestion par moyen de stockage (contrôleur RAID ou SAN). Ce qui permet d'assurer la continuité de service lorsque l'on perd le serveur (il en faut un de secours relié au SAN) ou le SAN (il faut de la place sur le stockage local) si l'on a un backup et multiplexé les redologs, archivelogs, controlefiles sur le SAN et en local.

Vous êtes libre d'organiser les fichiers comme vous le souhaitez où utiliser l'une des deux organisations présentées mais dans tout les cas cela mérite réflexion.

2. Fonctionnement

Vocabulaire:

une base de données Oracle: l'ensemble des fichiers qui constituent l'espace de stockage des données.

une instance Oracle: l'ensemble des processus et de la mémoire qui permet d'accéder à une (seule) base de données (ie: aux données), faire des traitements SQL. Elle a un SID qui correspond à son nom.

un serveur Oracle: c'est une base de données, une instance avec en plus des processus non dédiés aux traitements SQL mais utiles pour les performances, la disponibilité des données (la réplication des données ...).

les fichiers constituants physiquement la base de données:

3 types de fichiers :

- les fichiers de données (data files): .dbf
  - ils contiennent le dictionnaire de données (métadata)
  - ils contiennent les données (rows)
    (certaines de ces données peuvent être temporaires ou servir pour l'annulation).
  - ils constituent les tablespaces (espaces logiques)
- les fichiers de journalisation (redologs)
  - ils contiennent les modifications (ordres ddl,dml) successives des fichiers de données.
  - ils sont utilisés pour restaurer ou corriger un problème sur des fichiers de données par re-jeux.
  - il y en a plusieurs utilisés de manière circulaire: lorsque le redolog courant est saturé on passe au suivant ...
  - ils peuvent être multiplexés (plusieurs groupe) afin de palier à une perte éventuelle.
- les fichiers de contrôle (controlfiles)
  - ils sont généralement multiplexés (au moins doublés) mais tous identiques.
  - ils sont utilisés à l'initialisation de l'instance pour déterminer la localisation des autres fichiers.
  - ils contrôlent (!) l'état des fichiers de données à partir du SCN inscrit dans l'en-tête de chaque fichiers de donnée.
    Le SCN: System Change Number correspond à l'horodatage séquentiel (interne a Oracle) d'un commit (n transactions) réalisé.
    Cela permet d'assurer la consistance de la base.

Les autres fichiers correspondent à l'instance:

- un fichier d'initialisation pfile ou spfile:
  (initSID.ora spfileSID.ora)
  contient les paramètres d'initialisation: taille de la mémoire, ressources système, emplacement des controlfiles ...
- un fichier de password pour les dba (partagé: orapw ou exclusif: orapwSID)
- des fichiers de logs, d'audit, core, trace dans les répertoires adump,bdump,cdump,udump. Dans bdump il y a notamment le fichier d'alertes qui contient les erreurs internes à Oracle.
- éventuellement les fichiers de journalisation archivés:

Il peut aussi y avoir les archivelogs qui sont des redologs archivés lorsque ces derniers sont remplis (ou que l'on passe au redolog suivant).

Les processus:

les principaux processus sont:

- DBWR: database writer
  Il a la gestion des buffer mémoire des fichiers de données
  Il écrit les données modifiées/ajoutées du buffer dans les fichiers de la base de données lorsqu'il y a un commit et libère le buffer d'annulation. (il écrit aussi lors d'un checkpoint, drop, truncate, begin backup, alter tablespace, plus de buffer, temps expiré, mise en lecture seul d'un tablespace)
  L'écriture et la lecture peuvent être parallélisées DBWn avec le paramètre d'initialisation DB_WRITERS
- LGWR: log writer
  Ecrit dans les redologs lorsqu'il y a un commit, lorsqu'un buffer a enregistré plus d'1 Mo de modification, qu'il est rempli a 33% ou toutes les 3 secondes.
  Ainsi il écrit immédiatement les données modifiées validées ou annulées avant le DBWR et ne valide une transaction (commit/rollback) qu'une fois les données écrites sur le disque (dans le redolog) avec un SCN.CKPT:
- CKPT: checkpoint
  Il y a asynchronicité entre les écritures dans les redologs et dans les datafiles même après un commit. Le CKPT force le DBWR à écrire les modifications de données régulièrement, cela met à jour l'en-tête des fichiers de données ainsi que des fichiers de contrôle. Ainsi le dernier checkpoint détermine quand on doit commencer la récupération lorsque l'instance n'a pas été fermée correctement (différence de SCN entre les fichiers de contrôle, les redologs et les fichiers de données.
- SMON: system monitor
  Vérifie à l'initialisation de l'instance via les SCN l'intégrité de la base de données et effectue une récupération si nécessaire à partir des transactions validées dans le redolog qui n'aurait pas été écrites dans les fichiers de données par le DBWR.
  Il annule les transactions non validées.
  Il rend la base de données accessible: ouverte (open).
- PMON: process monitor
  Il gère les échecs des processus (ex: crash du client distant) en annulation de la transaction en cours, en libérant les verrous ou les autres ressources, en 2 mots: il nettoie !
- ARCH: archiver (si la base est en archivelog)
  Historise les redologs lorsque qu'ils ne sont plus utilisés (le log writer écrit dans le suivant)

Ces processus sont dits d'arrière-plan (background process)

Ensuite il y a les processus correspondants aux connexions, ils peuvent être dédiés ou partagés (beaucoup de connexions pour une même instance, plus de 500).

exemple:

```
oracle   11035 17806  0 10:09 pts/1    00:00:00 sqlplus
oracle   11292 11035  0 10:09 ?   00:00:00 oraclecocktail
(DESCRIPTION=(LOCAL=YES)(ADDRESS=(PROTOCOL=beq)))
oracle   12791    1  0 Jun04 ?       00:01:24 oraclecocktail (LOCAL=NO)
oracle   12799    1  0 Jun04 ?       00:00:27 oraclecocktail (LOCAL=NO)
oracle   12801    1  0 Jun04 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   18097    1  0 May28 ?       00:00:10 ora_pmon_cocktail
oracle   18099    1  0 May28 ?       00:00:00 ora_psp0_cocktail
oracle   18101    1  0 May28 ?       00:00:00 ora_mman_cocktail
oracle   18103    1  0 May28 ?       00:00:28 ora_dbw0_cocktail
oracle   18105    1  0 May28 ?       00:00:32 ora_lgwr_cocktail
oracle   18107    1  0 May28 ?       00:01:22 ora_ckpt_cocktail
oracle   18109    1  0 May28 ?       00:01:08 ora_smon_cocktail
oracle   18111    1  0 May28 ?       00:00:00 ora_reco_cocktail
oracle   18113    1  0 May28 ?       00:00:30 ora_cjq0_cocktail
oracle   18115    1  0 May28 ?       00:02:11 ora_mmon_cocktail
oracle   18117    1  0 May28 ?       00:01:02 ora_mmnl_cocktail
oracle   18137    1  0 May28 ?       00:00:00 ora_qmnc_cocktail
oracle   18145    1  0 May28 ?       00:00:00 ora_q000_cocktail
oracle   18836    1  0 Jun10 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   19283    1  0 May29 ?       00:29:39 oraclecocktail (LOCAL=NO)
oracle   24827    1  0 09:36 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   27955    1  0 May29 ?       00:00:01 oraclecocktail (LOCAL=NO)
oracle   27957    1  0 May29 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   28718    1  0 05:51 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   28720    1  0 05:51 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   28810    1  0 May29 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   28812    1  0 May29 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   29282    1  0 08:46 ?       00:00:00 oraclecocktail (LOCAL=NO)
oracle   29284    1  0 08:46 ?       00:00:00 oraclecocktail (LOCAL=NO)
```

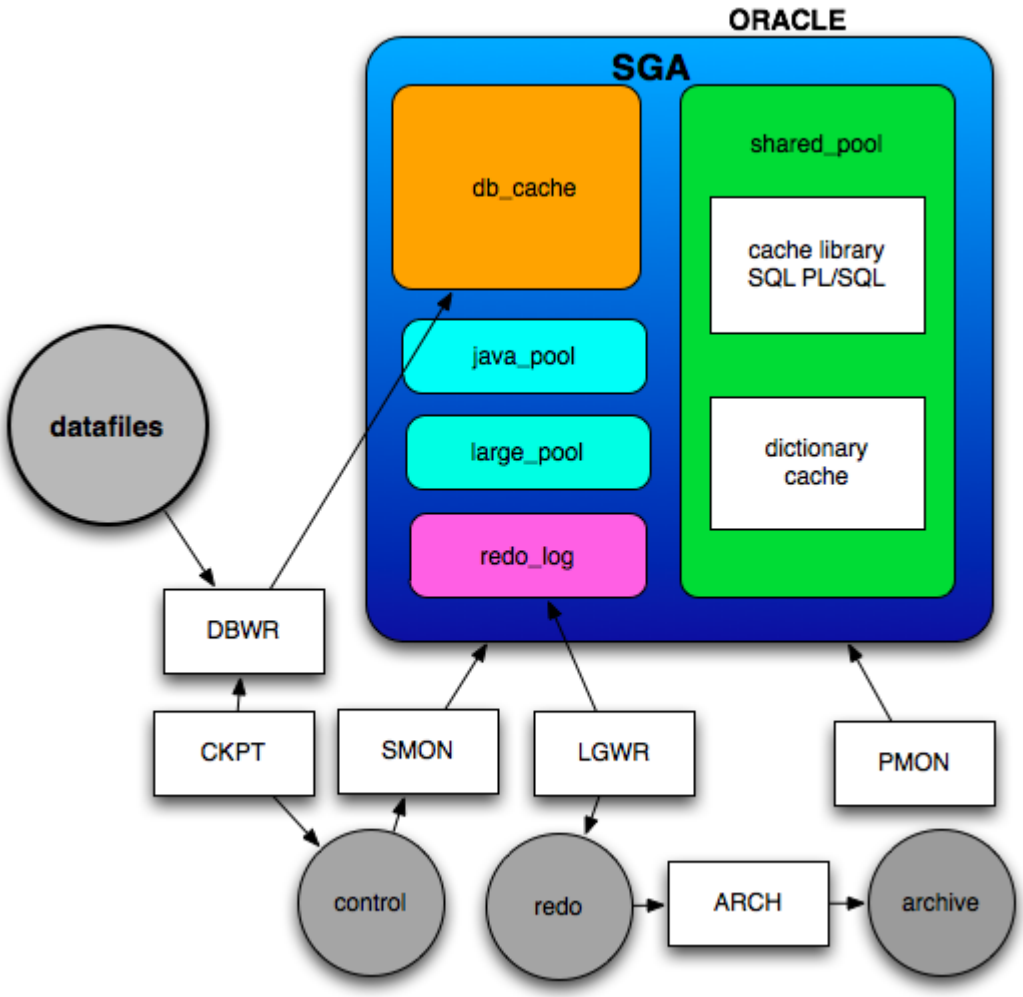
On remarque qu'il y a les 5 processus principaux d'arrière plan, une connexion locale (à noter le PID du parent) et 14 connexions distantes

La mémoire:

- SGA: system global area (show sga)
  Elle est allouée au démarage de l'instance (SGA_MAX_SIZE / SGA_TARGET & ASMM) et assure le partage des données: modifications, lectures, ordres sql...
  Elle est structurée en plusieurs zones:
  - cache buffer (DB_CACHE_SIZE,DB_KEEP_CACHE_SIZE, DB_RECYCLE_CACHE_SIZE, DB_nnK_CACHE_SIZE)
  - redolog buffer (note Metalink *351857.1)*
  - shared pool: cache library (dernières instructions sql ou pl/sql), data dictionnary cache (définitions des objects: tables, users, privilèges ...) (SHARED_POOL_SIZE)
  - java pool (JAVA_POOL_SIZE) procédures java (facultative)
  - large pool (LARGE_POOL_SIZE) facultative (non LRU)
    Elle est utilisé en mode serveurs partagés, lorsque RMAN utilise des bandes pour la sauvegarde (ou avec parallel_automatic_tuning=true) (elle peut soulager la shared pool)

- PGA: program global area
  Elle est allouée pour chaque processus "utilisateur" (connexion) et libérer la fin de ce dernier.
  En fonction des serveurs (dédiés,partagés) certaines allocation se font en SGA ou en PGA. Globalement la PGA est une mémoire SQL privée (bind, cursor, sort, join...) ce qui fait qu'elle est influencée par la sort_area_size hash_area_size mais aussi par les dblinks et le nombre de datafiles. Utilisation du paramètre pga_aggregate_target simplifie la configuration.
  (lorsqu'il n'y a plus de place pour les tris le tablespace temporaire est utilisé, à noter que les extents d'un tablespace temporaire restent toujours marqués comme utilisés mais sont recyclés lorsque cela est nécessaire ainsi les tris restent en cache et peuvent être réutilisés)

schéma sur l'interaction entre les processus, les fichiers et la mémoire:



Structure logique:

Une base oracle est composée de tablespaces, il y en a 2 au minimum :

- SYSTEM: dictionnaire de données interne d'oracle, à défaut d'un tablespace undo il a un rollback segment system, à défaut d'un tablespace temporaire il est le tablespace temporaire.
- SYSAUX: auxiliaire du tablespace system (depuis la 10g), il est le tablespace pour les outils oracle (awr,oem,text,logminer...) , tout ce qui est non sys.

Il y a ensuite 2 tablespaces qui ne sont pas totalement obligatoires mais presque car sinon tout va être fait dans le tablespace system donc de manière moins performante (les noms de ces 2 tablespaces sont libres puisque non absolument obligatoires)

- UNDO: tablespace pour gérer l'annulation, il remplace avantageusement les rollback segments depuis la 9i avec une gestion des données d'annulation automatisée (globalisée: non segmentée).
- TEMP: tablespace temporaire, utilisé lorsqu'il n'y a plus de place en mémoire PGA donc principalement pour les tris.

Hiérarchiquement:

- une base de données = n tablespaces (+redologs+controlfiles)
  (un tablespace ne peut être partagé entre plusieurs bases sauf s'il est en mode lecture seule et transportable ce qui est un cas exceptionnel)
- un tablespace = n fichiers de données
  (lecture écriture ou lecture seule)
- un fichier = n segment
  (n'appartient qu'à un seul tablespace)
- un segment = n extents = objets 'physique' contenant des données (table, index pas une vue, synonyme, trigers ...), il peut aussi être d'annulation ou temporaire.
  (peut être dans plusieurs fichiers d'un même tablespace)
- un extent = n blocks contigus
  (*db_block_size* doit être un multiple de la taille des blocs du FileSystem)

La gestion d'une instance:

Elle passe par l'initialisation

Elle se fait d'abord pour un fichier d'initialisation qui peut être sous 2 formes:

- pfile: fichier texte avec paramètre=valeur (les commentaires commencent par un #)
- spfile: fichier binaire non modifiable manuellement (pris en charge par rman), pour la modification d'un paramètre il faut utiliser: *alter system set param=valeur [comment 'text'] [scope=memory|spfile|both]*
  (on peut quand même afficher et reconnaître les paramètres

la conversion entre pfile et spfile se fait avec : *create [s]pfile="..." from [s]pfile="..." ;*
la lecture avec : *show parameters partie_du_nom_du_parametre* (ou via l'interrogation des vues: v$parameter, v$spparameter).

Le démarrage d'une instance se déroule en 3 étapes:

- nomount:
    - lecture du fichier d'initialisation dans $ORACLE_HOME/dbs
    - d'abord spfileSID.ora puis spfile.ora sinon initSID.ora
    - affectation de la SGA
    - lancement des processus d'arrière-plan: smon pmon ....
    - ouverture du fichier alert_SID.log

  On démarre en nomount que pour création de la base ou recréation des fichiers de contrôle
- mount: ouverture et lecture des fichiers de contrôle.
  Démarrer en mount est utile pour les opération de maintenance (récupération, renommer un fichier de données).
- open: les fichiers de données et de journalisation sont en ligne après vérification par SMON.
  Cela correspond à un démarrage normal.

Concrètement:

- pour se connecter a une instance, depuis un shell sur le serveur (avec un user dba):
  export ORACLE_SID=MonSID
  sqlplus /nolog
  connect / as sysdba

  (l'instance peut être inactive: arrêtée ou démarrée)
- pour démarrer une instance en nomount (1ière étape) avec un fichier d'initialisation spécifique (et non la lecture par défaut dans $ORACLE_HOME/dbs):
  startup nomount pfile=/...../initmydb.ora
- pour passer de l'étape nomount à mount:
  alter database mount;
- pour ouvrir complètement la base:
  alter database open;
- pour la démarrer tout simplement:
  startup
- pour un démarrage en mode restrict (les utilisateurs "standards" ne peuvent se connecter)
  startup restrict
- pour passer ou quitter le mode restrict
  alter system enable restricted session;
  alter system disable restricted session;

L'arrêt de la base peut être:

- normal (par défaut): on attend la fin des sessions en cours .....
- transactionnal: on attend la fin des transactions, chaque client est déconnecté à la fin de sa transaction
- immediate: on attend pas la fin des traitement sql en cours, les transactions actives sont annulées (le rollback peut prendre du temps), les utilisateurs déconnectés. L'utilisation la plus courante.
- abort: arrêt brutal (~kill -9): base non fermée ni démontée correctement ce qui implique une récupération automatique au prochain *startup open*.
  Cela peut laisser la base dans un état incohérent, ne faire qu'en cas de problème (le shutdown immediate ne fonctionne pas).

la commande est : *shutdown* [*normal* | *transactionnal* | *immediate* | *abort*]

connexions à l'instance:

Pour l'administration et les batchs les connexions sont locales mais pour l'utilisation les connexions sont distantes, ce sont les plus nombreuses.

Les connexions distantes doivent passer par un listener qui ouvre un port TCP pour recevoir les demandes de connexions pour lesquelles il lance un processus serveur dédié (ou un dispatcher utilisant un serveur partagé).

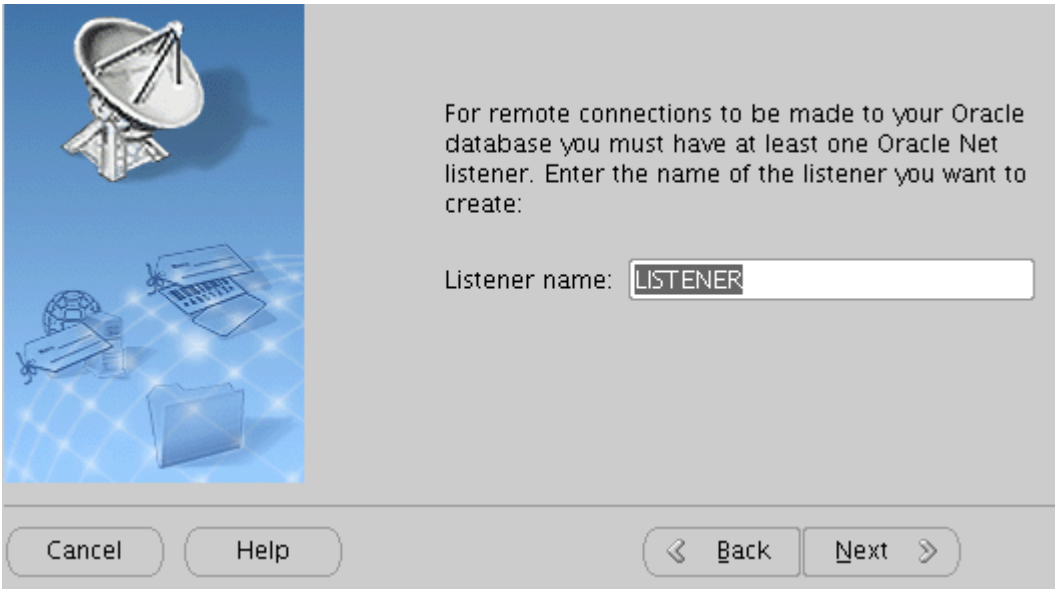
Lorsqu'il est le listener par défaut il reçoit aussi les enregistrements des PMON automatiquement (manuellement faire: *alter system register*), ainsi il sait quelles instances sont démarrées (il va regarder l'état de l'instance lorsque c'est nécéssaire)

On peut configurer les listeners via l'utilitaire netca ou en éditant le fichier $ORACLE_HOME/network/admin/listener.ora

lsnrctl est la commande qui permet de contrôler les divers listeners:

lsnrctl [*start* | *stop*| *status*] [*nom_du_listener*]

netca (net configuration assistant) est l'outil graphique, son utilisation est simple mais limitée



3. Base Cocktail

Pré-requis d'une base Oracle cocktail:

- Oracle Standard ou Enterprise
- versions:
  - 9ir2 doit toujours fonctionner (migration est à planifier au plus tôt)
  - 10gr2 est la version minimum (migration à envisager, plus de support standard)
  - 11gr2 est la version recommandée (patchset 1 de préférence)
- OS: nous conseillons GNU/Linux, vous pouvez utiliser ce que vous voulez, nous recommandons de rester dans la matrice de compatibilité Oracle. Nous déconseillons Microsoft Windows.
- Le paramétrage est libre, de même que les divers choix pour la création de la base et la gestion de l'instance !
  on impose les noms des tablespaces (et encore) et le nom des users et le jeux de caractères.

En 3 mots : vous êtes libres
(vous devez respecter la licence CeCCIL sous laquelle est distribuée le PGI Cocktail)

Pour la création de la base nous ne faisons que des recommandations.

La création:

Elle se déroule depuis une instance, donc il faut un fichier init pour pouvoir la démarrer, par défaut init.ora ou spfile.ora sont utilisés, il est donc préférable de créer son propre initSID.ora

```
# since 11g there is AMM for auto
 memory_target=512M

# automatic management memory
#sga_target=320M

# it auto tune :
# DB_CACHE_SIZE
# SHARED_POOL_SIZE
# LARGE_POOL_SIZE
# JAVA_POOL_SIZE
# pga_aggregate_target=192M  #defaut 20% SGA, 5% / sort
# v$db_cache_advice
# v$shared_pool_advice
# v$java_pool_advice
# v$mttr_target_advice
# v$pga_target_advice
# v$buffer_pool

# ----------------------

open_cursors=800
processes=300
job_queue_processes=10

# ----------------------

# DB version >= 10g
RECYCLEBIN = OFF
#alter system set recyclebin =OFF

# ----------------------

# max time to perform crash recovery
fast_start_mttr_target = 240

log_checkpoint_timeout=900

# ----------------------

#with hyper threading need to set cpu_count to the right value
#cpu_count=2
db_writer_processes=1 #nb volumes/fs but max=nb cpu

 # ----------------------

undo_management=AUTO
undo_tablespace=UNDO
```

```
# ----------------------

audit_trail='true'
audit_file_dest=/u00/databases/cocktail/logs/adump
# no background_dump_dest in 11g
#background_dump_dest=/u00/databases/cocktail/logs/bdump
core_dump_dest=/u00/databases/cocktail/logs/cdump
# no user_dump_dest in 11g
#user_dump_dest=/u00/databases/cocktail/logs/udump
diagnostic_dest=/u00/databases/cocktail/logs/trace #11g xml log
utl_file_dir=/u00/databases/cocktail/logs/utl

# ----------------------

# db version >= 10g
#db_recovery_file_dest=/u00/databases/cocktail/flash
#db_recovery_file_dest_size=512M

# ----------------------

control_files=('/u00/databases/cocktail/oradata/control01.ctl', '/u01/databases/
cocktail/oradata/control02.ctl')

log_archive_format=log%t_%s_%r.arch
log_archive_min_succeed_dest=1
# log_archive_start=true #for 9i
log_archive_dest=/u00/databases/cocktail/arch
log_archive_duplex_dest=/u01/databases/cocktail/arch
log_archive_max_processes=2

# ----------------------

# db version >= 10g
#dispatchers='(PROTOCOL=TCP) (SERVICE=XDB)'
#prevent DATA GUARD _XTP service cf Metalink note 339940.1
__dg_broker_service_names=' '

# ----------------------
db_domain='univxxxx.fr'  # no '-' cause it's not dns
db_name=cocktail

# ----------------------
nls_language='FRENCH'
nls_territory='FRANCE'

# ----------------------

compatible=11.2.0.2.0
remote_login_passwordfile='EXCLUSIVE'

# ----------------------

db_block_size=8192
#4096 is the minimum as it generally correspond to the filesystem block size
```

```
# new 11gr2 feature, by default at true which makes empty tables to not being exported:
DEFERRED_SEGMENT_CREATION=FALSE

# ----------------------

# TUNING PART
# CHANGE CAREFULLY

#sql_trace = true

timed_statistics=true
```

On peut noter que

- les fichiers de contrôle et les archivelogs sont multiplexés.
- le paramètre *compatible* doit correspondre à la version que vous utilisez (où une version antérieure dans certains cas).
- *sga_target*, *pga_aggregate_target*, les chemins, le nom de la base sont à adapter
- tous les répertoires indiqués dans le fichier d'initialisation doivent exister et le groupe dba doit avoir tout les droits dessus.
- certains paramètres d'initialisations peuvent changer complètement les plans d'exécution des requêtes SQL.
    - évaluer l'impact avant tout changement
    - tester (avec la même charge que la production)
    - toutes les bases sont différentes (données, OS, utilisateurs, matériel ....)
- certains paramétrages d'Oracle sont automatiques
    - gain en temps d'administration
    - moins de risques
    - plus simple

donc pourquoi s'en priver !

Passons à la création elle même :

dans votre shell positionnez la valeur ORACLE_SID au nom de l'instance puis lancez sqlplus

```
~$export ORACLE_SID=cocktail
~$sqlplus /nolog
```

il faut ensuite se connecter avec le rôle sysdba, comme l'on utilise une connexion locale avec un utilisateur qui est dans le groupe (unix) dba, on a pas besoin d'indiquer de login/password, ainsi par défaut on se connecte en tant que sys. On démarre l'instance sans la monter.

```
connect / as sysdba
startup nomount
```

Puis l'ordre sql *create database* va permettre de créer la base, avec le minimum nécessaire.

```
create database "cocktail"
maxinstances 1
maxlogfiles 16
maxlogmembers 4
maxdatafiles 32
character set "WE8ISO8859P15"
national character set al16utf16
archivelog
extent management local
datafile '/.../system01.dbf' size 128M autoextend on next 8M maxsize 512M
SYSAUX datafile '.../sysaux01.dbf' size 128M autoextend on next 8M maxsize
512M
undo tablespace UNDO datafile '.../undo01.dbf' size 256M
default temporary tablespace TEMP tempfile '.../temp01.tmp' size 512M
LOGFILE
        group 1 ( '.../redo01a.log' , '.../redo01b.log' ) size 16M,
        group 2 ( '.../redo02a.log' , '.../redo02b.log' ) size 16M,
        group 3 ( '.../redo03a.log' , '.../redo03b.log' ) size 16M;
```

A noter :

- que le jeux de caractères est ISO 8859-15
- que la base est en archive-log ce qui permet une plus grande sécurité (pour un coût assez réduit)
- que la gestion des extents est locale (et non centralisée dans le tablespace system)
- qu'il y a un undo tablespace et un tempory tablespace
- qu'il y a 3 groupes de redologs, composés de 2 membres (on peut en perdre un), ce qui est la configuration minimum.
  Si la base est en archivelog et que l'activité sur l'instance est telle que l'archivage (copie d'un redolog) est plus long que le remplissage de deux redologs, il faudra rajouter des groupes.

- qu'il n'y a que les tablespaces SYSAUX et SYSTEM qui sont extensibles et cela de manière limitée.
- que le tablespace temporaire a une taille fixe et qu'il sera utilisé à 100%, c'est normal (conservation des tris au cas où).
  Pour un établissement avec beaucoup de données et de connexions il peut dépasser 1 Go
- la taille du tablespace undo doit correspondre à l'activité qu'il y a sur l'instance (nombres d'utilisateurs modifiant ou ajoutant des données).
  256Mo doivent être suffisant pour un petit établissement, 1 Go pour un établissement de taille importante.

Ensuite installe le dictionnaire Oracle et le support des procédures stockées (PL/SQL)

```
@$ORACLE_HOME/rdbms/admin/catalog.sql;
@$ORACLE_HOME/rdbms/admin/catproc.sql;
```

On change les passwords par défaut et on verrouille les comptes non utilisés pour se connecter

```
alter user SYS identified by "**********";
alter user SYSTEM identified by "**********";
alter user OUTLN identified by "**********";
alter user DBSNMP identified by "**********";
alter user SYSMAN identified by "**********";
alter user OUTLN account lock;
```

On installe le seul package qui peut être nécessaire (cryptage de mot de passe)

```
@$ORACLE_HOME/javavm/install/initjvm.sql;
```

L'installation de l'outil d'explication du plan d'exécution est recommandée afin de pouvoir analyser comment oracle répond à une requête tout comme l'ajout des vues sur les verrous (locks)

```
@$ORACLE_HOME/rdbms/admin/catplan.sql;
@$ORACLE_HOME/rdbms/admin/catblock.sql;
```

L'installation de l'outil Statspack qui permet la prise de clichés de statistique, c'est une aide importante pour diagnostiquer certains problèmes de performances (il remplace ultbstats depuis la 9i)

```
DEFINE perfstat_password = "***********"
DEFINE default_tablespace = "SYSAUX"
DEFINE temporary_tablespace = "TEMP"
@$ORACLE_HOME/rdbms/admin/spcreate.sql
connect / as sysdba
```

On ajoute les extensions pour sqlplus (la table product_user_profile et l'aide)

```
connect system/**********
-- sqlplus product user profile
@$ORACLE_HOME/sqlplus/admin/pupbld.sql;
-- sqlplus help
@$ORACLE_HOME/sqlplus/admin/help/hlpbld.sql helpus.sql
connect / as sysdba
```

Enfin on reconstruit les objets invalides

```
connect / as sysdba
@$ORACLE_SID/rdbms/admin/utlrp.sql
```

et l'on vérifie les users existants

```
select username,account_status from dba_users;
```

Il reste quand même à ajouter les tablespaces

```
create tablespace DATA_GRHUM datafile
        '/u00/databases/$ORACLE_SID/oradata/data_grhum01.dbf'
        size 1024M autoextend on next 16M maxsize 1280M
        extent management local autoallocate
        segment space management  auto;
```

elle deux dernières lignes sont importantes et correspondent à des évolutions d'Oracle :

extent management: local & autoallocate

- Local (LMT): cela spécifie que la gestion de l'espace libre et des extents du tablespace est géré localement (via un b-tree/bitmap) et non par un dictionnaire (via du sql récursif ce qui implique des redologs et des rollbacks) situé dans le tablespace system.
  Cela est plus performant mais moins flexible: la taille des extents est automatique sauf l'initial extents. Ainsi NEXT,MINEXTENTS, PCTINCREASE, MAXEXTENTS, DEFAULT STORAGE se sont plus pris en compte. Aussi il n'est plus nécessaire de "coalescer" les extents libres.
- autoallocate vs uniform:
  - uniform: évite la fragmentation car la taille est fixe, utile s'il y a peut d'objets ou des objets de même taille (defaut 1Mo).
  - autoallocate: gestion automatique (taille 64k 1Mo 8Mo ...), une gestion manuelle plus fine pourra/ais être plus performante mais nécessite du temps pour l'administrer.

segment space management: auto

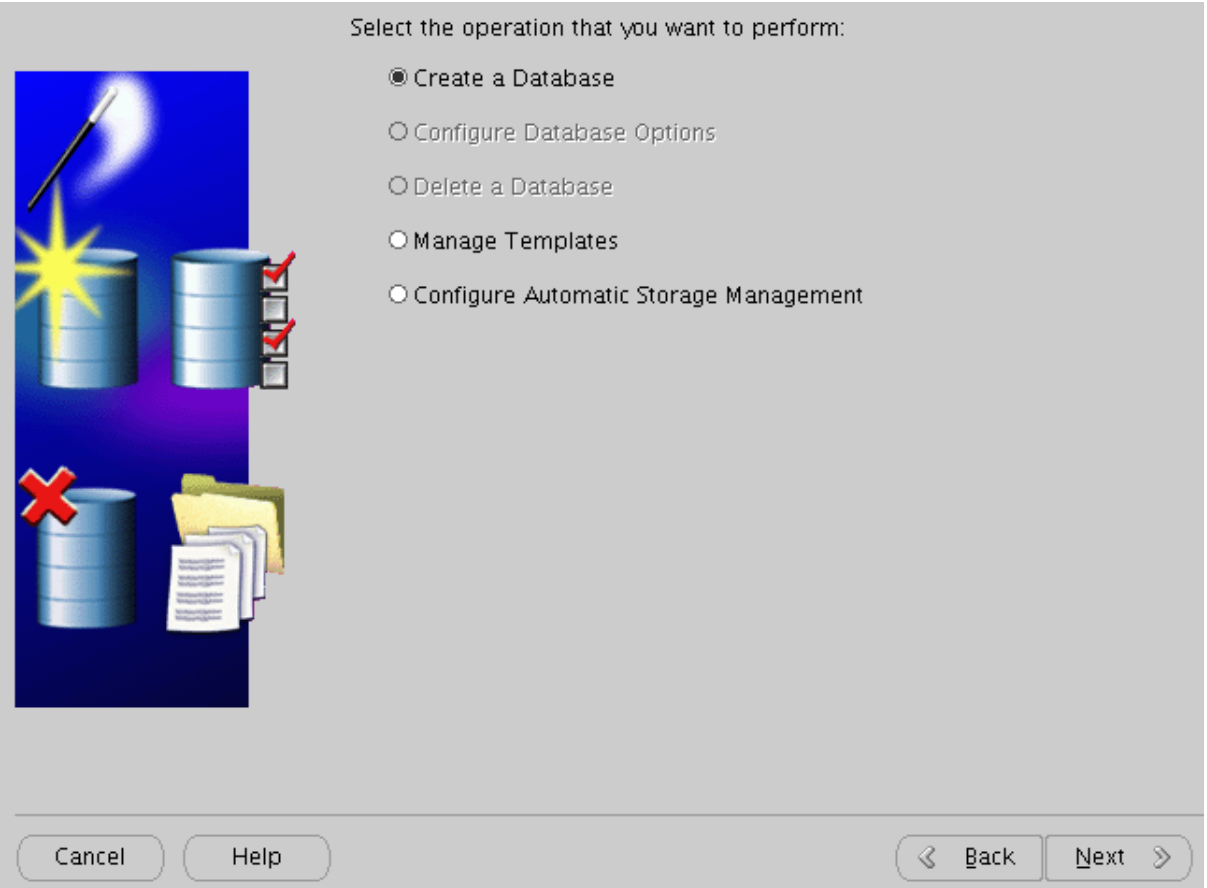
Automatic Segment (Free) Space Management: ASSM, automatise la gestion des freelist, ainsi PCTFREE, PCTUSED, FREELISTS, FREELIST GROUPS de la clause storage ne sont plus pris en compte. La gestion passe par un b-tree (bitmap) et non par des listes chaînées, cela favorise la concurrence et les objets dont les tailles de lignes

sont variables.
(le tablespace est forcément localy managed).

Les tablespaces "Cocktail" peuvent être en autoxentend, cela permet d'optimiser l'utilisation de l'espace sur le système de fichiers. Par contre fixer une taille limite est recommandé. Il faut néanmoins surveiller régulièrement l'espace libre disponible.

DBCA: database creation assistant

Est un outil qui permet graphiquement de créer des bases de données, à la fin des étapes de création il peut générer des scripts de créations. Ces derniers peuvent être utile pour comprendre les différentes options de création d'une base.



4. Sauvegarde, restauration, duplication

IL NE FAUT PRENDRE AUCUN RISQUE !

- ceinture et bretelles (2 sauvegardes de différents types)
- ceinture et bretelles à tous les niveaux
  (archivelogs, redologs, controlfile sont multiplexés sur différents systèmes de fichiers utilisant des contrôleurs distincts sur plusieurs disques en RAID 1, 5 ou 10 ...)
- vérifiez les sauvegardes !
- effectuez des simulations de crash !
- préparez des scénarios de restauration (testez les)

rien n'est plus important que les données, elles sont vitales pour le fonctionnement de votre établissement.

c'est votre tâche la plus importante

Sauvegarde à froid: copie de tous les fichiers de la base lorsqu'il n'y a pas d'instance de démarrée

basiquement:

il faut arrêter la base et copier les fichiers

```
export ORACLE_SID=mydb
sqlplus / as sysdba
        shutdown immediate
exit
cp /...../oradata/*  /..../uman
```

mais que se passe-t-il si quelqu'un a ajouté un fichier ailleurs que dans oradata, que la copie échoue ?
il faut donc envisager les divers cas

```
export ORACLE_SID=mydb

sqlplus / as sysdba
set head off    pagesize 0  linesize 1000  feedback off
set verify off   termout off   echo off

spool /..../cpall2backup.sh

select 'cp -p '||name||' /.../uman  || exit 1' from v$controlfile;
-- select 'cp -p '||member||' /.../uman  || exit 1' from v$logfile;
select 'cp -p '|| name ||' /.../uman/'|| FILE# ||'_'|| substr(name,instr(name,'/
',-2,1)+1) ||' || exit 1' from v$datafile;

spool off;
shutdown immediate;
exit;
/..../cpall2backup.sh
```

les redologs ne sont pas nécessaires si la base a été fermée correctement (il n'y a pas besoin de faire de récupération). Il peut être très problématique d'écraser ceux encore présent, si la sauvegarde se fait avec une base correctement fermée il n'y a aucune raison de sauvegarder les redologs, c'est pourquoi la ligne les concernant est commentaire.

On peut faire une pseudo sauvegarde des controlfiles sous forme de fichier texte (fichier instance_ora_pid.trc dans udump), il permet de recréer les controlfiles à la main avec une réincarnation de la base: remise à zéro des redologs (resetlogs) anciens archivelogs non utilisable

```
alter database backup controlfile to trace
```

(ce fichier peut être utile en cas de duplication de base)

pour trouver de manière automatisée le fichier .trc créé et le copier dans un répertoire de sauvegarde:

```
select 'cp '|| c.value || '/' || lower(instance) || '_ora_' ||
ltrim(to_char(a.spid,'fm99999999')) || '.trc /.../uman/controlfile.txt'
from v$process a, v$session b, v$parameter c, v$thread c
where a.addr = b.paddr
        and b.audsid = userenv('sessionid')
        and c.name = 'user_dump_dest';
```

<u>Sauvegarde à chaud:</u> des données et métadonnées (et non de la base de données)

les 2 outils de base pour les dumps

- export

```
exp \'/ as sysdba\' file=....dmp log=....log COMPRESS=N  STATISTICS=NONE
CONSISTENT=Y FULL=Y
```

(les séquences peuvent être en cache et sont alors non exportées à leur valeur réelle en cas d'utilisation)
(full= tout sauf les objets de SYS)

- import

```
imp \'/ as sysdba\'  file=....dmp ignore=Y log=fullimp.log full=Y commit=Y
rows=Y
```

depuis la 10g il y a en plus datapump:
- plus rapide (parallélise)
- utilise une DIRECTORY de la base (pas de dump local)
  (*create directory DumpDir as '/..../';*)

son utilisation reste sensiblement la même:

- export

```
expdp  \'/ as sysdba\' directory=FULLEXPORTS  dumpfile=bckcktl.dmp
logfile=bckcktl.log full=y    [  parallel=4 ]
```

- import

```
impdp test/test@db10g schemas=TEST directory=TEST_DIR
dumpfile=TEST.dmp logfile=TEST.log
```

Sauvegarde à chaud:

sauvegarde manuelle:
le principe est de passer les tablespaces en mode backup, de copier les fichiers et de remettre les tablespaces en mode normal.

```
alter system switch logfile ;
alter tablespace SYSTEM begin backup  ;
alter tablespace UNDOTBS1 begin backup  ;
....
cp /..../system01.tbs  /...../uman
cp /.../undo01.tbs /.../uman
...
alter tablespace SYSTEM end backup  ;
alter tablespace UNDOTBS1 end backup  ;
alter database backup controlfile to '.../uman/control.ctl' REUSE ;
alter system switch logfile ;
```

(monitoring via v$backup)

sauvegarde automatisée:
elle passe par l'utilisation de RMAN

ce dernier utilise un référentiel/catalogue c'est a dire une schéma dans une base (mais peut utiliser le fichier de contrôle de la base de données cible ce qui est encore moins pratique).
il faut donc avoir une base pouvant accueillir RMAN :

```
create tablespace rman_data datafile '...rman_data01.dbf' size 64M;
create user rman identified by ******  default tablespace rman_data;
grant connect, resource, recovery_catalog_owner to rman;
```

initialiser le catalogue

```
rman catalog rman/*****[@sid]
RMAN> create catalog tablespace rman_ts_data ;
```

(attention il existe un outil rman sous unix pour "décompiler" les pages de manuels)

on peut donc maintenant utiliser RMAN:

pour ajouter une cible

```
rman catalog rman/***
RMAN> connect target sys/***@TargetSID
RMAN>register database ;
```

pour resynchroniser une cible en cas d'ajout/changement de fichiers

```
RMAN> connect target sys/***@TargetSID
RMAN> resync catalog ;
```

d'autres commandes sont possibles:

```
RMAN> startup mount  # démarre la base cible

RMAN> configure retention policy to revocery windows of 10 days;
RMAN> configure retention policy to redundancy 2;
RMAN> configure backup optimization on;
RMAN> show all; # affiche toute la configuration

RMAN> list backup of database;
RMAN> list copy of tablespace "TOOLS"

RMAN> copy datafile '/../tools.dbf' to  '/.../tools.bck';
RMAN> backup copies datafile 1, datafile 2 format '/.../%U';
RMAN> backup database filesperset 3;
RMAN> configure controlfile autobackup;

RMAN> report need backup days 3;
RMAN> report obsolete; #liste la backups
RMAN> report schema;
```

script de sauvegarde:

```
RMAN>run {
   allocate channel t1 type disk;
   backup format '/.../rman/df_%d_%t_%s_%p' database;
   sql 'alter system switch logfile';
   backup format '/.../al_%d_%t_%s_%p'
   archivelog all delete input;
   }
```

(sauvegarde de la base, archivage d'un redolog, sauvegarde de se dernier puis effacement de tout les archivelogs puisque celui dont on a besoin est sauvegardé)

Restauration:

- avec RMAN:

```
RMAN>Run {
   allocate channel t1 type disk;
   restore database;
   recover database;
   sql "alter database open";
}
```

- à partir des fichiers:
  recopie des fichiers dans leurs répertoires d'origines puis *startup* ou *startup nomount* suivit de *create controlfiles* et ou *open resetlogs*

- à partir d'un dump :
  création de la base, des répertoires où sont les fichiers de données et éventuellement les tablespaces puis import full.
  (Un import full peut recréer les tablespaces, avec les fichiers correspondants mais pas les répertoires. Les tablespaces existants ne seront pas recréés)

Duplication:

- à partir d'un dump:
  c'est comme une restauration, sauf qu'il faut absolument créer les tablespaces.

- à partir de fichiers:
  création des répertoires, copie (depuis un backup manuel ou depuis rman) des fichiers de données dans ces répertoire, enfin sous sqlplus :

```
startup nomount
create controlfile reuse SET database "NEWSID" resetlogs  NOarchivelog
   maxlogfiles 10
   ....
logfile
  group 1 (
  ...
   datafile
        '.../...dbf',
        ....
       '.../.../dbf'
   character set we8iso8859p15;
alter database open resetlogs;
alter tablespace temp add tempfile '....dbf' size 128m  reuse autoextend off;
alter database rename global_name to NEWSID.DOMAIN
shutdown immediate;
startup
```

la sauvegarde du controlfile sous forme textuelle donne exactement cette commande avec resetlog et noresetlog saut il faut simplement ajouter SET et éventuellement mettre NO devant archivelog (si c'est une base de test). Une fois la nouvelle instance démarrée il est bon de changer global_name.

- à partir de RMAN la duplication est aussi possible, elle passe par la création d'une base auxiliaire et est moins simple
  (voir http://download.oracle.com/docs/cd/B19306_01/backup.102/b14191/rcmdupdb.htm)

La récupération
(à ne pas confondre avec restauration)

elle est automatique, utilise les redologs (et archivelogs), peut être base ouverte ou fermée (lorsque les tablespaces system ou undo sont non disponibles):

recover [database| tablespace | datafile]

En cas de perte d'un datafile: on recopie celui de la sauvegarde et la récupération va rejouer ce qu'il y a dans les redologs et archivelogs.

Il peut arriver que la récupération soit incomplète car il manque des redologs/ archivelogs :

- *recover database until cancel*
- *recover database until time 'yyyy-mm-jj-hh-mm-ss-cc'*
- *recover database until scn INTEGER*

puis *alter database open resetlogs*
(faire une sauvegarde immédiatement...)
Cela est un cas qui ne doit pas arriver étant donné que les redologs et les archivelogs doivent être multiplexés et ce sur différents systèmes de fichiers sur différents supports physiques.

Notes concernant RMAN:

L'utilisation ne semble pas simple, en tout cas la prise en main n'est pas instantanée. Cependant une fois l'outil maitrisé c'est très confortable. La restauration est simple (RMAN = Recovery MANager), sous rman après s'être connecté à la base cible "*RMAN> restore database*" suffit !
Mais il y a d'autres avantages:

- Il travaille au niveau blocs: vérification l'intégrité des blocs, restauration d'un bloc corrompu, sauvegarde plus rapide (blocs non déjà sauvegardé et non vide), sauvegarde incrémentale.
- Il peut appliquer des politiques de rétention, de purge automatique.
- Il connaît de facto les structures des bases donc ce qui faut sauvegarder, gère les réincarnations (resetlogs).
- Il peut compresser les sauvegardes.
- Il peut faire des rapports sur ce qui est obsolète, les sauvegardes nécessaires...
- Il s'interface (via un agent) aux outils de sauvegarde classiques du marché.

5. Recommandations et outils

Recommandations

la première recommandation est : **Sauvegardez** !

- sauvegardez la sauvegarde physique Oracle
- les sauvegardes ne doivent pas rester:
    - sur le même espace de stockage que la base
    - sur le même serveur
    - dans la même salle machine (PRA)

La protection des données n'est pas que la sauvegarde !
- l'accès aux données doit être contrôlé
- l'accès au système d'exploitation doit être contrôlé
- l'accès à la (salle) machine doit aussi être contrôlé

cela s'effectue à différents niveaux :
- réseau :
    - vlan / acl
    - firewall
- des services type ssh
- listener (contrôle distant possible)
- droits des users
- moteurs du sgbd: CPU et patchs/updates

listener: sécurisation

Il faut utiliser *lsnrctl* pour définir le mot de passe puis éditer le fichier $ORACLE_HOME/network/admin/listener.ora

```
LOGGING_LISTENER = ON
LOG_FILE_LISTENER = LISTENER.log
LOG_DIRECTORY_LISTENER =/...../network/log
ADMIN_RESTRICTIONS_LISTENER = ON
LOCAL_OS_AUTHENTICATION_LISTENER = ON
PASSWORDS_LISTENER = ***CRYPTED***
```

users:

- les passwords :
  - insensibles à la casse (sauf 11g)
  - ceux par défaut sont à changer
  - cryptés avec DES sans vrai sel
    donc des attaques du type brut force sur le hash sont possibles (7 char ~ 1j / 8 char 42j / 9 char 1600j)
- par défaut les comptes sont verrouillés après 10 tentatives successives de connexions infructueuses.
- les grants à éviter:
  - *select / execute any_*XXXXX
  - *create public synonym / database link*
  - *create any / external job*
- attention aux rôles: il peuvent évoluer
  grant create session != *grant connect*
- vérouiller et expirer les comptes non utilisés
  alter user XXXX [account lock | password expire]

CPU: critical patch update

- corrige des bugs: tous, donc de l'opérationnel (avant la version 10.2.0.3)
- corrige uniquement les problèmes de sécurité (à partir de la version 10.2.0.3)
  (a noter l'existence des Patch Set Updates: PSU qui corrigent les bugs)

Application d'un CPU:

- *optach apply*
- exécution du scripts catcpu.sql sur les bases actuelles et les futures !

Pour connaître ce qui est installé:

- de manière générale
  opatch lsinventory -detail
- au niveau des scripts catcpu.sql
  *select * from SYS.REGISTRY$HISTORY*

Les outils

ils sont d'abord Oracle :

- Metalink contient beaucoup d'informations ! (http://metalink.oracle.com)
  Par exemple la note 131704.1 est une liste de "scripts to do X" et de "how to do X"
- Dictionnaire de données: *select * from dictionary;*
  - vues statiques:
    dba_users, dba_segments, dba_temps_files, dba_roles, dba_profiles, dba_objects, dba_synonyms, ...
    user_* (identique aux vues dba_* mais portent sur le schéma courant)
    all_* (identique aux vues dba_* mais portent sur les objets pour lequels le user courant à les droits)
  - vues dynamiques:
    v$database, v$instance, vsga, vsession, v$process, v$lock, v$latch, v$locked_object, v$session_wait, v$sqlarea, v$option, v$archive_log, vlog, vobjet_usage...

ce qui donne par exemple pour obtenir:

le buffer cache hit ratio

```
SELECT  (1-(pr.value/(dbg.value+cg.value)))*100
FROM    v$sysstat pr, v$sysstat dbg, v$sysstat cg
WHERE   pr.name = 'physical reads'
   AND     dbg.name = 'db block gets'
   AND     cg.name = 'consistent gets';
```

(plus le nombre est proche de 100 mieux c'est car cela signifie que les données sont presque toutes en cache)

les 10 ordes sql les plus exécutés

```
SELECT * FROM
   (SELECT sql_text,executions, rows_processed, rows_processed/executions
   "Rows/Exec", hash_value,address
   FROM V$SQLAREA
   WHERE executions > 100
   ORDER BY executions DESC)
WHERE rownum <= 10
```

utilisation du tablespace temporaire par session

```
SELECT b.tablespace, b.segfile#, b.segblk#,
round(((b.blocks*p.value)/1024/1024),2) size_mb,
a.sid, a.serial#, a.username, a.osuser, a.program, a.status
FROM v$session a, v$sort_usage b, v$process c, v$parameter p
WHERE p.name='db_block_size'
   AND a.saddr = b.session_addr
   AND a.paddr=c.addr
ORDER BY b.tablespace,b.segfile#,b.segblk#,b.blocks;
```

la session utilisateur en fonction du PID système

```
select s.sid, s.serial#, p.spid, s.osuser, s.program, s.logon_time
from   v$process p, v$session s
where  p.addr = s.paddr and p.spid=PID_SYSTEM;
```

la liste des connexions

```
select s.username, s.program, s.logon_time
from v$session s, v$process p, sys.v_$sess_io si
where s.paddr = p.addr(+)
and si.sid(+) = s.sid
and  s.type='USER' ;
```

- les packages:
  - *dbms_stats.gather_shema_stats*
    dbms_stats.gather_database_stats_job_proc (entre 20h / 6h par defaut)
  - *sys.dbms_space_admin (migration de tablespace)*
  - *statspack (stats$sysstat) statspack.snap*
    @ORACLE_HOME/rdbms/admin/statsrep.sql
  - *dbms_xplan: select * from TABLE(dbms_xplan.display);*
  - logminer dbms_logmnr
  - flashback query
  - dbms_job
  - ...

- les événements:

  par exemple lorsque l'on a "ORA-00942 object not found" on peut faire

```
alter system events '942 trace name errorstack level 1';
```

ainsi dans le répertoire de log on obtient un fichier trace avec l'erreur et la requête qui pose problème ce qui permet d'obtenir l'objet manquant ou pour lequel il manque des droits

on peut aussi tracer une session (.trc dans udump):

```
oradebug setospid PIDSYSTEM;
oradebug event 10046 trace name context forever, level 12 ;

-- ..... (effectuer au niveau clients ce qui pose problème)

oradebug event 10046 trace name context off;
```

(level 1:basic, 4:bind, 8:wait events>cpu, 12= 4 + 8)

L'utilisation de SYS.DBMS_SUPPORT.START_TRACE_IN_SESSION est aussi possible pour tracer une session.

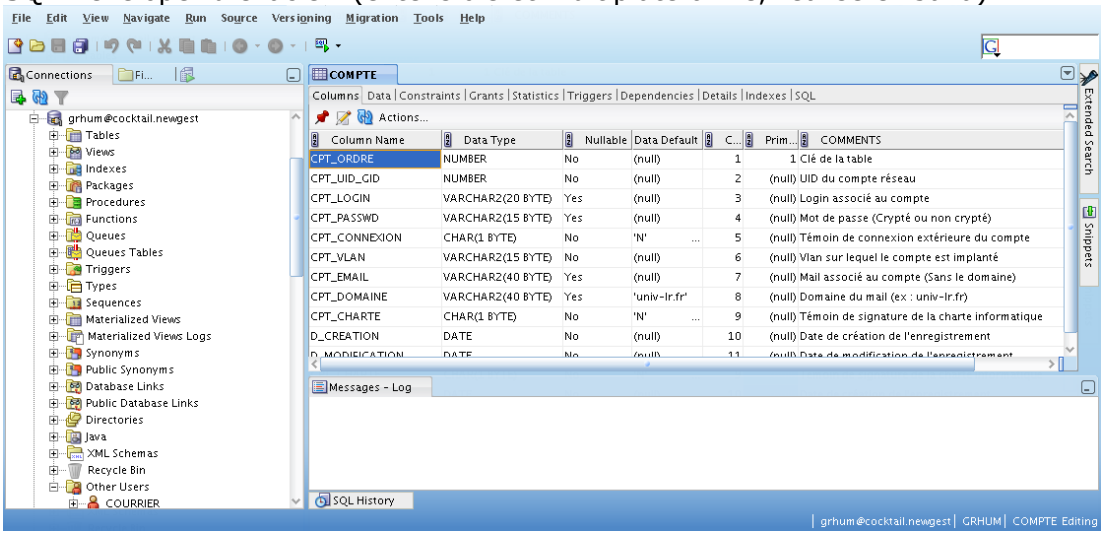
- outils d'exploitations des fichiers de trace (.trc)
    - édition manuelle: verbeux donc parfois difficile de trouver ce que l'on cherche
    - tkprof: plus synthétique
      tkprof trace.trc output.log sys=no sort=execpu,fchcpu
    - trace analyzer: trcanlzr (à télécharger sur metalink, voir note 224270.1),
      exec trca$i.trace_analyzer('....trc') ;
    - OraSRP (http://www.oracledba.ru/orasrp/ donc non Oracle)
      orasrp --sort=execpu,fchcpu --sys=no trace.trc report.html
      rapport en html et n'a pas besoin d'Oracle (peut donc être fait sur une autre machine)

- sqlplus
  configuration dans *~/login.sql* ou *$ORACLE_HOME/sqlplus/admin/glogin.sql*
  commandes spécifiques:

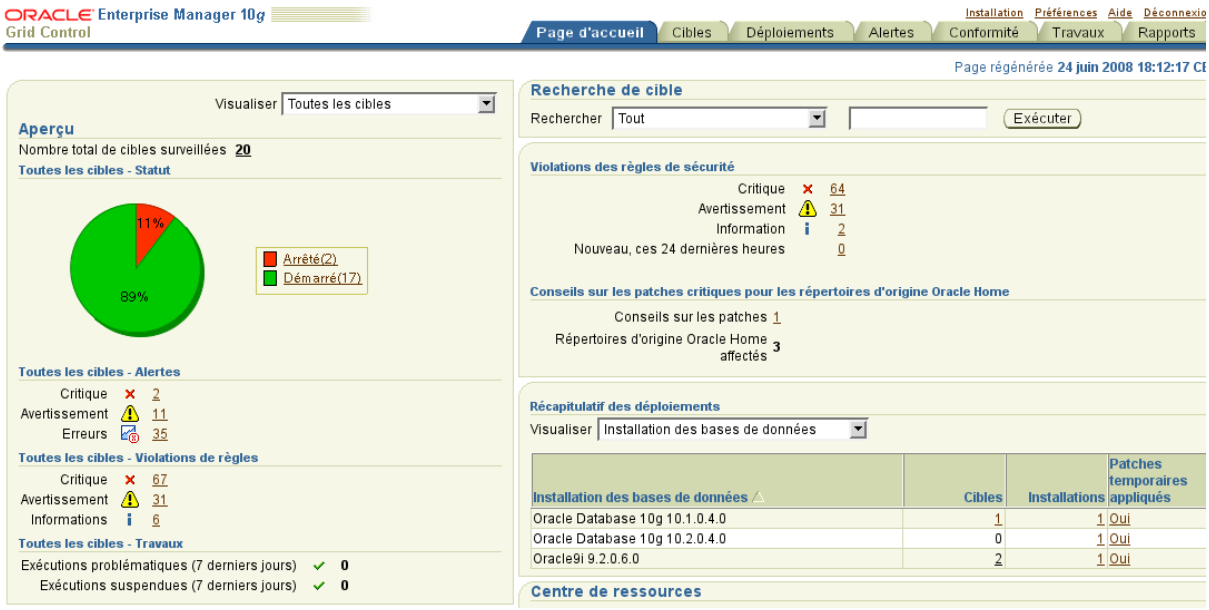
```
set autotrace on;

set timing on;

set sqlprompt "_user'@'_Connect_identifier>"

spool   ..... spool off

@/....sql

@@xxxx.sql

PROMPT specify a password for sys as parameter 1;
DEFINE sysPassword = &1
host orapwd file=.... password=&&sysPassword force=y
host fait la substitution de variable & ou && alors que ! non.

LIST

help index
```

- outils graphiques: il y a SQL Developer (gratuit), Toad (payant) ... qui sont du type client-server ou des outils centralisés comme OEM dans sa version Grid-control.

  SQL Developer d'Oracle: (extensible et multiplateforme, réalisé en Java)



Oracle Enterprise Manager Grid Control qui est une console centralisée (interface http) qui peux surveiller ou administrer des produits autres que ceux d'Oracle (ils y a des packs pour Oracle Enterprise, ils sont soumis à licences, il y a notamment des packs qui donne le droit d'exploiter AWR : une l'interface de tuning)



En résumer l'exploitation de bases Oracle c'est:

- vérifier les sauvegardes avec par exemple:
    - création, à partir de la sauvegarde à froid, d'une base de formation
    - création d'une base avec tablespaces et imp full (puis drop)
- monitoring des ressources du serveur (cpu disque mémoire io)
- monitoring des temps de réponse SQL, des temps d'attente (wait times), de l'espace libre dans les tablespaces
- monitoring du fichier d'alerte
- tracer, analyser, corriger
- scripter (la machine doit travailler pour vous, non ?)

Critical Patch Update Advisory
Critical Patch Update Implementation (PDF)
Oracle Database Security Checklist (PDF)

Conclusion

L'exploitation d'une base Oracle n'est pas forcement simple au premier abord, il faut:

- comprendre le fonctionnement
- savoir faire les divers opérations de maintenance

après ce n'est principalement que du contrôle et de la surveillance.