Buku Pegangan
Oracle Database Administrator

Edisi kedua

Oleh:
Rohmad
(http://rohmad.net)

Kata Pengantar Edisi Kedua

Di luar Dugaan, edisi pertama buku ini mendapat banyak apresiasi dari para pembaca. Kritik dan saran juga banyak yang masuk. Saya mengucapkan banyak terima kasih atas apresiasi, kritik, dan saran dari para pembaca sekalian.

Edisi pertama buku ini berjudul "Kitab Suci Oracle Database Administrator". Beberapa pembaca menyarankan agar judulnya diganti, jangan memakai kata "Kitab Suci", kesannya sombong. Pada mulanya saya memakai kata "Kitab Suci" hanya asal comot kata saja, sama sekali tidak terlintas pikiran untuk sombong ataupun bangga. Tapi sekarang sayapun membenarkan sara-saran beberapa teman pembaca tersebut, kalau dirasa-rasa memang tampak ada kesombongan yang memancar dari judul buku "Kitab Suci Oracle Database Administrator" ini, terlepas dari segala motivasi penulis (saya).

Akhirnya dengan senang hati dan ucapan terima kasih kepada teman-teman pembaca yang telah memberikan saran, di edisi kedua buku ini judulnya saya ganti menjadi "Buku Pegangan Oracle Database Administrator".

Selain mengganti judul, saya juga menambahkan 1 bahasan baru di bab 11, yaitu "Oracle Flashback Technology (Recycle bin)".

Semoga buku sederhana ini bermanfaat bagi para pembaca sekalian. Kritik dan saran tetap saya tunggu dengan senang hati.

Rohmad
Jakarta,
Oktober 2009

Kata Pengantar

Ayo, kita masyarakatkan budaya *sharing knowledge*. Mari, kita saling berbagi ilmu. Ilmu itu sangat berharga; tidak akan habis dengan diberikan pada sesama, bahkan semakin bertambah.

Pada mulanya saya mengenal blog di Multiply.com. Setelah sekian lama nge-blog hingga bosan, akhirnya blog di Mulitply saya delete. Menulis memang hoby saya. Setelah lama absen akhirnya saya nge-blog lagi, kali ini di Wordpress.com. Memakai blog gratis (dalam arti hosting dan domain gratis), rasanya kurang memuaskan karena kita tidak bisa meng-customize blog sesuka kita. Akhirnya saya putuskan membuat blog dengan hosting dan domain sendiri. Hingga akhirnya saya punya blog sendiri si http://rohmad.net

Di blog, saya menulis apa saja yang pengin saya tulis. Oleh teman-teman di kantor, saya disarankan untuk lebih banyak menulis tentang database Oracle. Lagi pula database Oracle adalah keahlian saya, sementara itu banyak sekali yang tertarik untuk belajar dan mendalami database Oracle. Hingga akhirnya artikel-artikel mengenai database Oracle mendominasi *content* blog saya.

Dengan maksud mengoptimalkan *sharing knowledge*, akhirnya saya mengkompilasi tulisan di blog ke dalam bentuk buku. Setelah melalui kategorisasi dan beberapa editing, jadilah buku yang sederhana ini. Yang saya masukkan ke dalam buku ini hanya tulisan pokoknya saja, sementara diskusi dan komentar pembaca blog tidak saya ikutkan. Bagi yang ingin mengikuti diskusi dan komentar pada tiap-tiap tulisan, silahkan baca di blog saya.

Tidak lupa, melalui kesempatan ini, saya ingin menyampaikan terima kasih yang sebesar-besarnya kepada semua pihak, atas dorongan semangat dan bantuannya, sehingga buku ini selesai saya susun. Saya sangat senang sekali menerima kritik dan saran dari pembaca sekalian, silahkan hubungi saya di email rohmadsan@yahoo.com

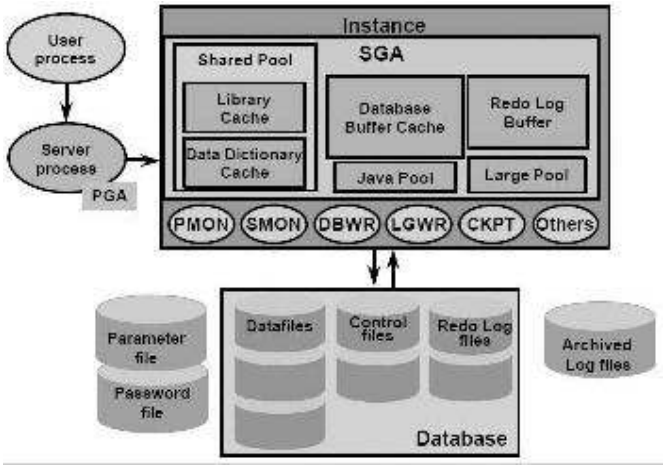
Terakhir, secara spesial buku ini saya persembahkan untuk teman-teman kerja saya di kantor. Dan secara umum buku ini saya persembahkan pada siapa saja yang bisa mengambil manfaat dari buku ini.

Rohmad
Jakarta,
Desember 2008

Daftar Isi

1. Concept:

1.1. Arsitektur Database Oracle



Kata **database** dalam frasa "arsitektur database Oracle" seharusnya adalah **Database Management System** (DBMS). Untuk penyederhanaan penyebutan, **Database Management System** memang sering cukup disebut **database** saja.

Secara umum komponen DBMS Oracle terdiri atas memory, proses, dan file-file. Lebih jauh lagi, komponen-komponen tersebut dikelompokkan sebagai berikut:

1. Instance
   - Memory yang disebut sebagai System Global Area (SGA), terdiri atas: Shared Pool (Libary Cache and Data Dictionary Cache), Database Buffer Cache, Redolog Buffer Cache, Java Pool, Large Pool.
   - Back ground process: PMON, SMON, DBWR, LGWR, CKPT, dan lain-lain
2. Database
   - Datafile
   - Control file
   - Redo log file
3. Komponen lain
   - process: Server Process, user process
   - memory: Program Global Area (PGA)
   - File: Archived log, parameter, dan password file

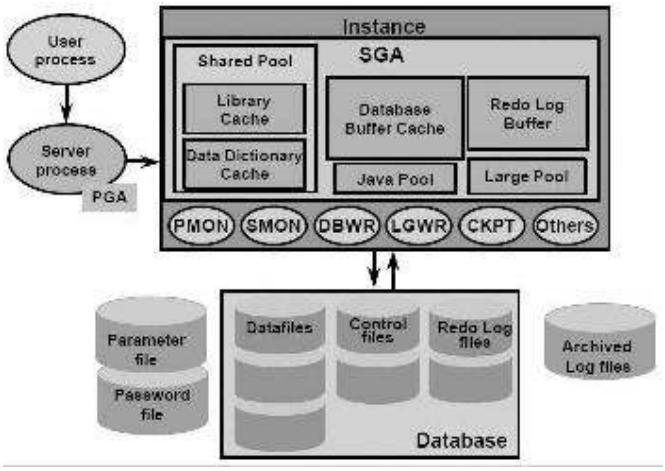
1.2. Perbedaan Instance dengan Database

Kebanyakan kita mungkin tidak tahu persis perbedaan instance dan database. Banyak yang mengira bahwa instance itu ya database. Bagi yang lebih kritis lagi lantas bertanya-tanya, kalau gitu apa gunanya Oracle memakai istilah instance dan database?

Kalau kita membuat database dengan DBCA, by default nama instance dan nama database itu memang sama. Apakah nama database yang saat ini kita maintain itu sama dengan nama instance?

- Untuk mengecek nama instance, pastinya semua DBA Oracle sudah tahu. Value dari ORACLE_SID adalah nama dari instance juga.
- Untuk mengecek nama database, gunakan command *select VALUE from v$parameter where NAME='db_name'*

Lebih jelasnya, ada baiknya kita lihat Arsitektur Database Oracle di sini
http://rohmad.net/2008/04/17/arsitektur-database-oracle/



Instance adalah struktur **proses** dan **memory** yang menjalankan sistem database (DBMS, database management system). Sedangkan database adalah sekumpulan file yang menyimpan data (yang terdiri atas datafile, controlfile, dan redo log file).

Istilah **Database Oracle** yang sering kita pakai itu merujuk pada **database management system** (DBMS) **Oracle**. Ngucapin database management system mungkin terlalu panjang. Ngucapin DBMS mungkin kok tidak enak. Jadi gampang-gampangan, sebut saja database.

1.3. Teori dan Administrasi init file (pfile dan spfile)

Konfigurasi instance disimpan di file bertipe **text** yang dikenal dengan init file (initial file) atau `pfile` (parameter file). Mulai versi 9i Oracle memperkenalkan `pfile` bertype **binary** yang disebut `spfile`.

Di Unix family, init file ada di directory `$ORACLE_HOME/dbs`. Sedangkan di Windows ada di folder `%ORACLE_HOME%\database`.

Format nama file:

1. Pfile: `initNAMAINSTANCE.ora`
   Di Unix, nama instance adalah case sensitive, instance dataku berbeda dengan DATAKU. Pfile untuk instance dataku adalah `initdataku.ora`. Dan Pfile untuk instance DATAKU adalah `initDATAKU.ora`

   Sedangkan di Windows, nama init file tidak case sensitive, instance dataku ya sama saja dengan DATAKU. Kalau kita membuat database dengan dbca, initfile yang terbentuk adalah INITdataku.ORA. Kalau file ini diganti dengan initDATAKU.ora yang tidak apa-apa, Windows gitu lho!

2. Spfile: `spfileNAMAINSTANCE.ora`
   Sama seperti pfile, case sensitive di Unix dan tidak case sensitive di Windows.

Cara membuat initfile:

1. pfile
   Dibuat secara manual pakai text editor , contoh: `notepad` di Windows dan `vi` di Unix. Bisa juga dibuat berdasarkan content spfile yang sudah ada.

   ```
   SQL> create pfile from spfile;
   ```

2. spfile
   Spfile tidak bisa dibuat dengan text editor sebagimana membuat pfile. Tentu saja, file binary tidak bisa dibuat (diedit) dengan text editor. Spfile hanya bisa dibuat dengan cara berikut ini (content-nya diambil dari pfile yang sudah ada):

   ```
   SQL> create spfile from pfile;
   ```

By default, kalau ada spfile maka ketika startup Oracle akan membaca parameter dari spfile. Kalau tidak ada spfile, Oracle membaca pfile. Kalau tidak ada kedua-duanya, instance tidak bisa di-startup.

```
SQL> startup
ORA-01078: failure in processing system parameters
LRM-00109: could not open parameter file
'/mnt01/oracle/10.2.0.3/dbs/initDATAKU.ora'
```

Mengedit Init file (mengubah parameter instance)

Parameter instance ada dua tipe yaitu `dynamic` dan `static`. Parameter dynamic bisa diubah ketika instance sedang jalan sedangkan parameter static tidak bisa, artinya perubahan parameter static harus dilakukan di initfile dan instance harus di-restart.

Contoh parameter dinamik adalah `pga_aggregate_target`. Berikut ini cara untuk mengubahnya:

1. Kalau instance sedang mati
   Naikkan instance dulu, kemudian lakukan perubahan pakai SQLPlus

   ```
   SQL> startup
   SQL> alter system set pga_aggregate_target=100m;
   ```

   Perintah alter system di atas langsung mengubah parameter di instance (`memory`) yang berjalan.

   - Kalau kita pakai spfile, perintah ini langsung mengupdate juga spfile. Sehingga ke depannya kalau kita merestart instance, Oracle membaca `pga_aggregate_target=100m` dari spfile.
   - Kalau kita pakai pfile, `pga_aggregate_target=100m` tidak di-update ke pfile sehingga ke depannya kalau kita merestart instance, pga_aggregate_target kembali ke nilai semula. Agar perubahan bersifat permanen, edit juga parameter pga_aggregate_target di pfile

2. Kalau instance sedang jalan
   Langsung lakukan perubahan di SQLPLus
   ```
   SQL> alter system set pga_aggregate_target=100m;
   ```

   Sama seperti penjelasan sebelumnya, kalau pakai spfile maka spfile juga diupdate secara otomatis. Kalau pakai pfile, agar perubahan bersifat permanen maka pfile harus diedit secara **manual** pakai text editor.

Contoh parameter statik adalah `db_writer_processes, control_files`, `sessions`, dsb. Berikut ini cara untuk mengubah parameter `db_writer_processes` yang bersifat statik itu:

1. Kalau instance sedang mati
   Kalau pakai pfile, edit pfile pakai text editor, kemudian startup instance.

   ```
   Edit pfile
   SQL> startup
   ```

   Spfile tidak bisa diedit pakai text editor, kalau tetep dipaksa edit pakai text edior maka spfile akan corrupt sehingga tidak dikenali oleh Oracle. Spfile hanya bisa diubah dengan SQLplus ketika instance naik. Jadi naikkan dulu instance, pakai nomount biar cepet toh kita tidak perlu instance mount atau open, yang penting startup dulu. Kemudian `alter system set db_writer_processes=2 scope=spfile`. Perintah ini akan mengedit spfile saja, sementara parameter di

instance sendiri masih belum berubah. Setelah itu, baru startup

```
SQL> startup nomount
SQL> alter system set db_writer_processes=2 scope=spfile;
SQL> shutdown immediate
SQL> startup
```

2. Kalau instance sedang jalan

   Kalau pakai pfile, database matiin dulu, edit pfile, kemudian restart instance

```
SQL> shutdown immediate
Edit pfile
SQL> startup
```

   Kalau pakai spfile, alter system dengan scope=spfile, kemudian restart

```
SQL> alter system set db_writer_processes=2 scope=spfile;
SQL> shutdown immediate
SQL> startup
```

Catatan

1. Pada perintah `alter system set pga_aggregate_target=100m`,

   By default kalau pakai spfile, maka
   `alter system set pga_aggregate_target=100m scope=both`
   Dan kalau pakai pfile. maka
   `alter system set pga_aggregate_target=100m scope=memory`

2. Kita tidak bisa menjalankan `alter system set db_writer_processes=2` karena ini adalah parameter static. Kalau masih dipaksa akan muncul error:

```
SQL> alter system set db_writer_processes=2;
ERROR at line 1:
ORA-02095: specified initialization parameter cannot be modified
```

   Parameter static hanya bisa diubah di scope=spfile

```
SQL> alter system set db_writer_processes=2 scope=spfile;
System altered.
```

3. Kalau dilihat pakai text editor, spfile berisi sama seperti pfile, hanya ada entry yang `"aneh-aneh"` (yang bukan merupakan parameter instance) di baris pertama.
4. Contoh kasus.

   Instance saya hanya punya spfile (`spfileDATAKU.ora`) dan tidak punya pfile (`initDATAKU.ora`). Suatu ketika saya ingin mengubah parameter static (misalnya control_files). Karena ketidak tahuan, parameter control_files saya edit di spfileDATAKU.ora pakai text editor (notepad).

   Karena spfile adalah file binary, maka spfile menjadi rusak karena diedit pakai text editor, sebagian akibatnya Oracle tidak bisa membacanya. Karena spfile tidak bisa dibaca, maka Oracle mencari pfile. Karena saya tidak punya pfile sementara spfile-

nya corrupt, ya akhirnya saya tidak bisa startup instance

```
SQL> startup
ORA-01078: failure in processing system parameters
LRM-00109: could not open parameter file
'/mnt01/oracle/10.2.0.3/dbs/initDATAKU.ora'
```

Waduh… gimana ini?!! Solusinya, buat pfile `initDATAKU.ora` yang isinya sama persis dengan `spfileDATAKU.ora`, dengan catatan: jangan ikut sertakan entry `spfileDATAKU.ora` yang `aneh-aneh` di baris pertama itu.

Untuk melihat isi spfile spfileDATAKU.ora, di unix gunakan perintah `more`. Di Windows, gunakan `wordpad`; kalau pakai `notepad` biasanya tampilannya amburadul.

1.4. Kitab Suci DBA Oracle

Subject artikel ini mungkin kedengaran keren, ya. Saya menyebutnya ‘kitab suci’ (kitab suci pakai tanda petik) karena hampir setiap hari saya buka, saya baca, saya ‘mintai’ pertolongan, dan saya jadikan ’sandaran’. Selanjutnya untuk mempermudah penulisan saya tidak pakai tanda petik, namun maknanya tetap sebagai kitab suci pakai tanda petik.

Yang saya sebut sebagai kitab suci itu adalah view DICT, dokumentasi Oracle, dan Metalink. Mari kita bahas satu per satu.

1. View DICT
DICT adalah synonym dari view DICTIONARY. DICT berisi daftar view (view bawaan Oracle) yang berisi tentang semua informasi di database Oracle. DICT berisi kolom TABLE_NAME dan COMMENTS. Lihat definisi DICT dengan comand desc (describe).

```
SQL> desc dict
```

Informasi apa saja bisa kita dapatkan di sini. Misalkan kita ingin tahu, informasi tentang partisi disimpan di VIEW apa saja kah? Caranya, select dari view DICT, masukkan kata kunci yang ingin kita cari dalam clause WHERE. Kata kunci pakai huruf besar. Formatnya adalah **select table_name from dict where table_name like ‘%KATA KUNCI%’** . Dalam contoh ini kata kunci adalah PARTITION.

```
SQL> select table_name from dict
where table_name like '%PARTITION%' order by table_name;

TABLE_NAME
------------------------------
ALL_IND_PARTITIONS
ALL_IND_SUBPARTITIONS
ALL_LOB_PARTITIONS
ALL_LOB_SUBPARTITIONS
ALL_SUBPARTITION_TEMPLATES
ALL_TAB_PARTITIONS
ALL_TAB_SUBPARTITIONS
USER_IND_PARTITIONS
USER_IND_SUBPARTITIONS
USER_LOB_PARTITIONS
USER_LOB_SUBPARTITIONS
USER_SUBPARTITION_TEMPLATES
USER_TAB_PARTITIONS
USER_TAB_SUBPARTITIONS
```

Daftar VIEW yang muncul adalah VIEW yang bisa diakses oleh user yang bersangkutan (user yang query ke DICT tersebut). Bila user tersebut mempunya ROLE DBA (misalnya SYS dan SYSTEM) maka daftar VIEW yang keluar lebih banyak lagi. Berikut ini daftar VIEW yang hanya bisa dilihat oleh user dengan role DBA:

```
DBA_IND_PARTITIONS
DBA_IND_SUBPARTITIONS
DBA_LOB_PARTITIONS
DBA_LOB_SUBPARTITIONS
DBA_SUBPARTITION_TEMPLATES
DBA_TAB_PARTITIONS
DBA_TAB_SUBPARTITIONS
```

2. Dokumentasi Database Oracle dari oracle.com
Dokumentasi yang paling sering saya pakai adalah Master Index. Istilah-istilah yang tidak saya pahami, bisa segara saya dapatkan pemahamannya melalui Master Index. Misalkan saya pengin tahu lebih banyak tentang dbms_resource_manager. Dari Master Index saya buka link yang menuju istilah-istilah yang dimulai huruf D. Kemudian saya search pakai kata kunci (keyword) dbms_resource_manager. Selanjutnya saya dapat banyak pilihan link (tema) yang mana yang saya butuhkan.

Dokumentasi berikutnya adalah Administrator Guide. Pokoknya, apa saja ada di dikumentasi Oracle. Meskipun database Oracle banyak versinya (8i, 9i, 10g, 11g), secara umum content dokumentasinya hampir sama kecuali yang version specific.

Berikut ini dokumentasi yang bisa kita dapatkan dari oracle.com. Kalau ingin akses yang lebih cepat, sebaiknya di-download dulu ke PC local kita. Documentation Index ada di sini.

- Database Oracle 11g Release 1.
  - Konsep
  - Administrator Guide
  - Master Index
  - Dan yang lain-lainnya …
- Database Oracle 10g Release 2.
  - Konsep
  - Administrator Guide
  - Master Index
  - Dan yang lain-lainnya …
- Database Oracle 9i Release 2.
  - Konsep
  - Administrator Guide
  - Master Index
  - Dan yang lain-lainnya …
- Database Oracle 8i Release 3.
  - Konsep
  - Administrator Guide
  - Master Index
  - Dan yang lain-lainnya …

3. Metalink
Linknya di sini https://metalink.oracle.com/. Ini adalah tool yang paling saya andalkan. Untuk bisa memanfaatkan Metalink, kita harus punya nomor CSI (Customer Support Identifier). Setiap company yang menggunakan database Oracle dan **membeli support Oracle (license)** pasti mendapat nomor CSI. Jadi, tidak semua orang bisa akses ke sini.

Manfaat yang bisa kita dapatkan dari Metalink:
- Minta bantuan (support) team Oracle terhadap masalah yang kita hadapi
- Minta CD software Oracle (tanpa biaya apapun alias gatis tis …)
- Download Patch
- Melihat technical documentation yang tidak di-share di “Oracle Documentation”

Apapun masalahnya, selama kita punya akses ke Metalink, pasti ada solusinya (bisa di-handle) karena kita punya senjata pamungkas yaitu “Minta bantuan (support) team Oracle

terhadap masalah yang kita hadapi". Namun ya itu tadi, kita harus punya akses ke Metalink. Atau dengan kata lain, kita mesti membeli support (license) Oracle dulu.

Bagi yang tidak punya akses (account) ke Metalink, masih ada alternatif jalan lain untuk mencari solusi (meskipun tidak seampuh Metalink). Silahkan cari di:

- google
- http://www.oracle-base.com/
- http://www.orafaq.com/
- http://asktom.oracle.com/

1.5. Mengenal Teknologi Grid

Oracle 10g, **g** adalah singkatan dari **grid**. Berikut ini pembahasan mengenai Grid yang saya kutip dari Wikipedia.

Komputasi Grid (*grid computing*) adalah penggunaan sumber daya yang melibatkan banyak komputer yang terdistribusi dan terpisah secara geografis untuk memecahkan persoalan komputasi dalam skala besar.

Latar belakang grid

Perkembangan kecepatan prosesor berkembang sesuai dengan Hukum Moore, meskipun demikian bandwidth jaringan komputer berkembang jauh lebih pesat. Semakin cepatnya jalur komunikasi ini membuka peluang untuk menggabungkan kekuatan komputasi dari sumber-sumber komputasi yang terpisah. Perkembangan ini memungkinkan skala komputasi terdistribusi ditingkatkan lebih jauh lagi secara geografis, melintasi batas-batas domain administrasi yang sudah ada.
Pesatnya perkembangan teknologi komputer di negara-negara maju, membuat para penelitinya semakin haus akan tenaga komputasi yang dapat menjawab tantangan dan permasalahan yang mereka hadapi. Walaupun sudah memiliki supercomputer dengan kapasitas yang sangat tinggi , apa yang sudah ada ini pun dirasa tetap kurang, karena mereka berusaha memecahkan permasalahan yang lebih besar lagi. Setelah semua komputer yg dimiliki seorang "peneliti haus tenaga komputasi" dipergunakan habis-habisan untuk memecahkan masalahnya, setelah berbagai cara untuk memecahkan masalah dicoba, dan dipilih yang paling efisien, tapi tetap masalahnya belum bisa dipecahkan juga, apa yang harus dia lakukan? Komputasi grid adalah salah satu jawaban dari pertanyaan ini.

Definisi grid

Menurut tulisan singkat [1] oleh Ian Foster ada check-list yang dapat digunakan untuk mengidentifikasi bahwa suatu sistem melakukan komputasi grid yaitu :

- Sistem tersebut melakukan koordinasi terhadap sumberdaya komputasi yang tidak berada dibawah suatu kendali terpusat. Seandainya sumber daya yang digunakan berada dalam satu cakupan domain administratif, maka komputasi tersebut belum dapat dikatakan komputasi grid.
- Sistem tersebut menggunakan standard dan protokol yang bersifat terbuka (tidak terpaut pada suatu implementasi atau produk tertentu). Komputasi grid disusun dari kesepakatan-kesepakatan terhadap masalah yang fundamental, dibutuhkan untuk mewujudkan komputasi bersama dalam skala besar. Kesepakatan dan standar yang dibutuhkan adalah dalam bidang autentikasi, otorisasi, pencarian sumberdaya, dan akses terhadap sumber daya.
- Sistem tersebut berusaha untuk mencapai kualitas layanan yang canggih, (nontrivial quality of service) yang jauh diatas kualitas layanan komponen individu dari komputasi grid tersebut.

Peluang grid

Dalam buku *The Grid:Blue Print for a new computing infrastructure* dijelaskan bahwa yang dimaksud dengan komputasi grid adalah infrastruktur perangkat keras dan perangkat lunak yang dapat menyediakan akses yang bisa diandalkan, konsisten, tahan lama dan tidak mahal terhadap kemampuan komputasi mutakhir yang tersedia.
"A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities."
Seandainya kelak di kemudian hari teknologi yang dibutuhkan untuk mewujudkan visi paradigma komputasi grid ini sudah mapan, peluang akan semakin terbuka bagi kerjasama lintas organisasi, lintas benua dan lintas bangsa. Akan terbuka peluang bagi peneliti di Indonesia yang ingin melakukan komputasi yang sangat rumit, dengan menggunakan supercomputer tercepat di dunia, tanpa harus melakukan investasi besar-besaran dalam bidang teknologi informasi.

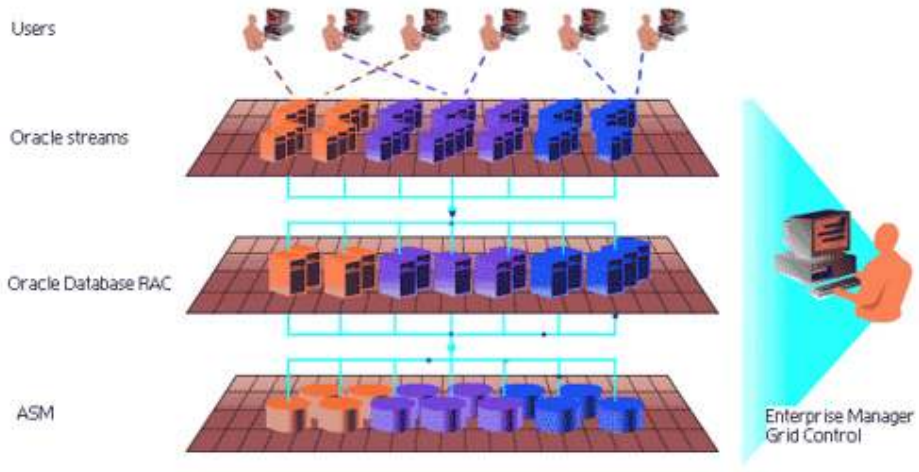
Produk Grid dari Oracle

Oracle telah membangun software infrastruktur komputasi grid yang mampu membagi dan menyeimbangkan semua beban (*workload*) di antara server-server yang berbeda, dan me-manage server-server yang berbeda itu sebagai satu kesatuan sistem.

Kemampuan komputasi grid adalah setara dengan mainframe karena semua komponennya (grid) di-cluster. Namun tidak seperti *mainframe* dan server SMP (*symmetric multiprocessing*) yang besar, grid bersifat terbuka (*open system technologies*) yang bisa mencakup system kecil-kecil dengan processor Intel dan OS Linux sehingga biaya jadi jauh lebih murah.

Adapun produk grid dari Oracle adalah:

- Storage Grid: Automatic Storage Management (ASM)
- Database Grid: Real Application Server (RAC)
- Application Grid: Oracle Streams
- Grid Control: Enterprise Manager Grid Control



2. Basic Installation, Create, and Configuration:

2.1. Install database oracle 10g di Windows XP

Secara umum, guide ini berlaku untuk semua instalasi Oracle. Di semua OS, proses instalasi itu sama, hanya sedikit berbeda di pre-installation requisite-nya.

Download software Database Oracle dulu. Free, alias gratis… tis.. Sebelumnya anda harus punya account di Oracle. Kalau belum punya, membuatnya gampang sekali, yang penting punya email. Ikuti saja proses "sign up". Kalau diminta memasukkan pin OPN, dan anda tidak punya pin OPN karena company anda bukan partner-nya Oracle, kosongkan saja.

Berikut ini panduan download "Oracle Database 10g Release 2 (10.2.0.1.0)":

1. Buka link (page) untuk download di sini
   http://www.oracle.com/technology/software/products/database/index.html
2. Pilih versi database dan tipe OS

   Dalam hal ini saya memilih "Oracle Database 10g Release 2 (10.2.0.1.0) for Microsoft Windows"

3. Kemudian muncul pilihan berikut
   - Oracle Database 10g Release 2 (10.2.0.1.0)
   - Oracle Database 10g Companion CD Release 2 (10.2.0.1.0)
   - Oracle Database 10g Client Release 2 (10.2.0.1.0)
   - Oracle Clusterware Release 2 (10.2.0.1.0)
   - Oracle Gateways 10g Release 2 (10.2.0.1.0) (10.2.0.1.0)

   Pilihlah option pertama. Kalau hanya untuk sekedar belajar database, anda cukup milih option pertama saja.

PC yang saya gunakan dalam instalasi ini adalah:
- Microsoft Windows XP Professional Version 2002 Service Pack 2
- CPU: intel 2 Ghz
- Memory: 1G
- Virtual (page) memory: 1,5G

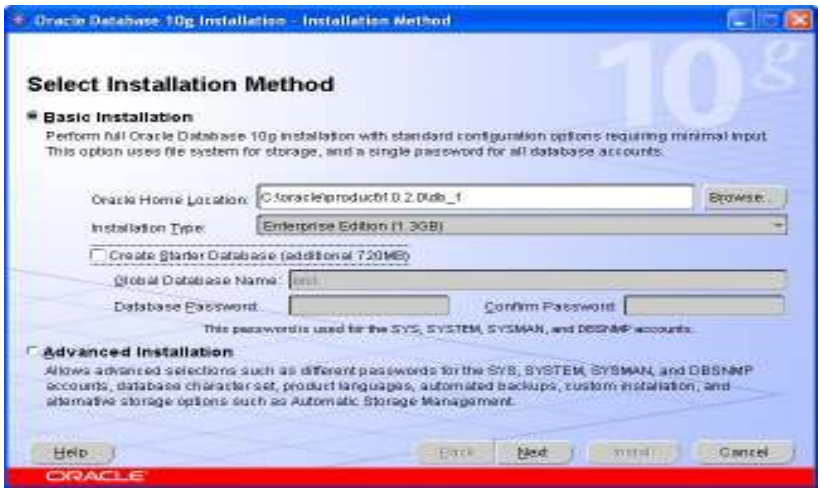
Dengan spesifikasi tersebut, jelas PC saya sangat (lebih dari cukup) memenuhi syarat. Lebih detail tentang spesifikasi komputer yang bisa diinstall, lihat dokumentasi (installation guide) yang ada di paket software yang telah di download. Secara umum, berikut ini spesifikasinya:
Hardware:
- Physical memory (RAM) : 256 MB minimum, 512 MB recommended
- Virtual memory: dua kali RAM
- Disk space: kira-kira 5 G
- Video (monitor) adapter: 256 colors
- Processor : 550 MHz

Operating system (OS)
- Windows 2000 with service pack 1 or later. All editions, including Terminal Services and

Microsoft Windows 2000 MultiLanguage Edition (MLE)
- Windows Server 2003 - all editions
- Windows XP Professional
- Windows NT is not supported.

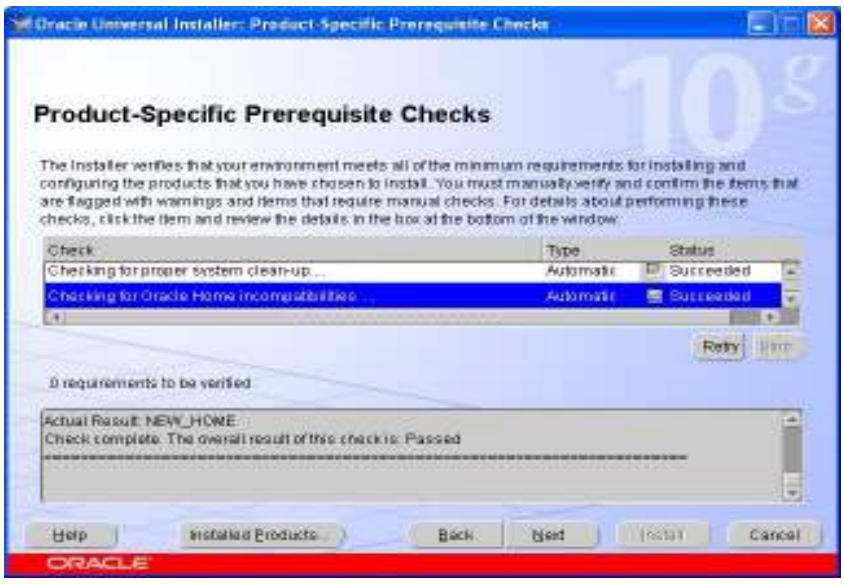
Berikut ini langkah-langkahnya:

1. Jalankan command “setup.exe” yang ada di paket software yang telah di download Kemudian muncul Install wizard (GUI). Lihat gambarnya di sini



- Pilih option “Basic Installation”
- Masukkan directory “Oracle Home Location”
- Pilih “Installation Type”
- Jangan pilih “Create Starter Database”
- Klik button “Next”

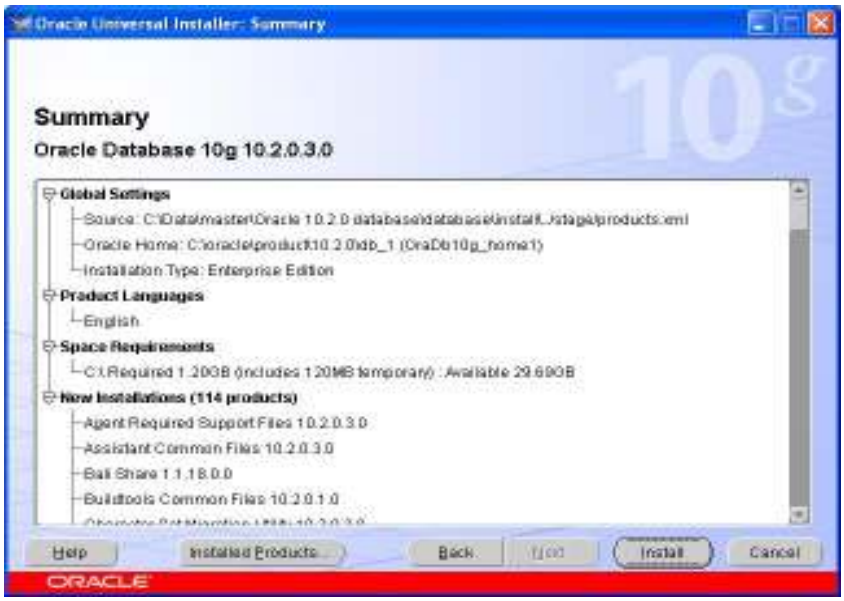
2. Oracle installer akan mengecek OS kita, apakah requirement-nya dipenuhi atau tidak. Lihat gambarnya di sini



Pastikan semua statusnya “Succeeded”. Kalau ada warning, atau statusnya bukan Succeed, bereskan dulu OS-nya. Kemudian klik button “Next”

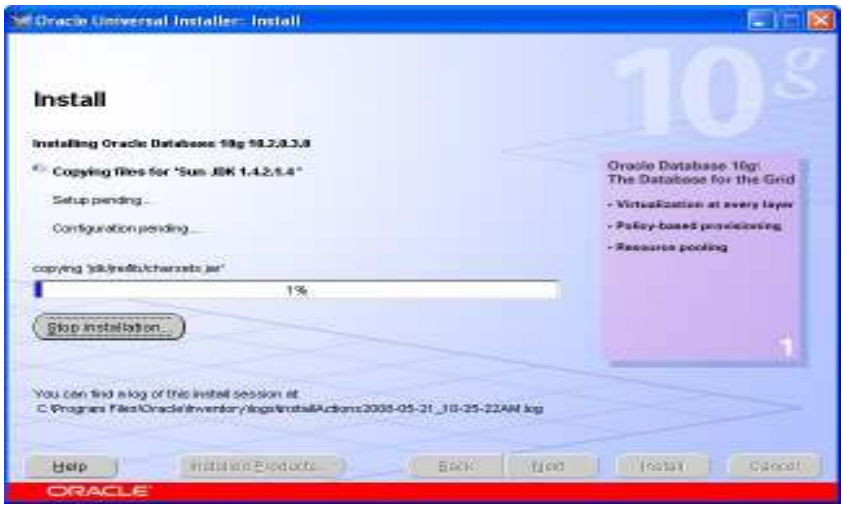
3. Dalam proses instalasi, Oracle akan menjalankan program java. Bila firewall PC anda memblock java, dan muncul alert “Windows Security Alert”, kilk tombol “Unblock”. Lihat gambarnya di sini

4. Muncul summary komponen Oracle Database 10g yang siap kita install. Lihat gambarnya di sini.



Kemudian klik tombol “Install”

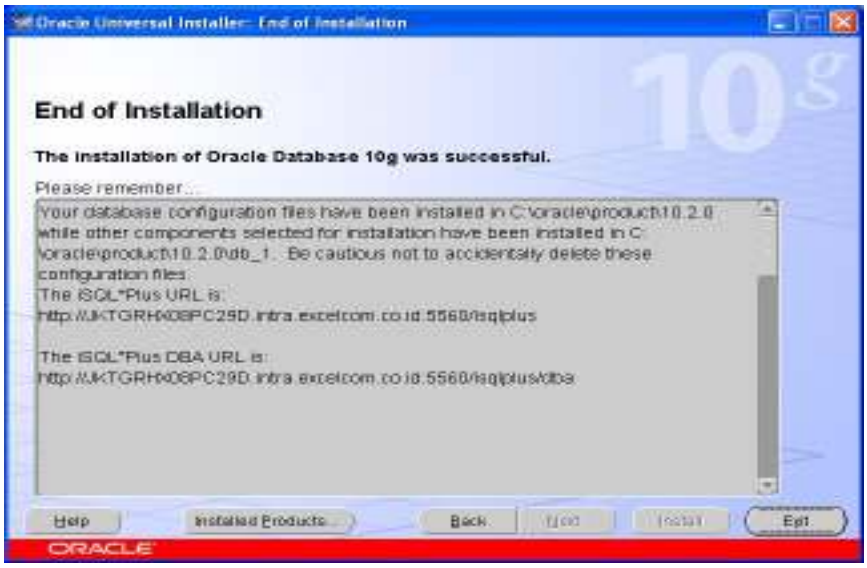
5. Installation progress ditunjukkan oleh gambar ini.



6. Setelah itu akan dilakukan konfigurasi tambahan oleh Oracle Installer. Kita cukup perhatikan saja. Lihat gambarnya di sini.

Setelah konfigurasi selesai, klik tombol “Next”. Kadang-kadang kita tidak perlu kilk tombol Next tersebut, secara otomatis wizard menuju ke berikutnya.

7. Akhirnya instalasi selesai. Lihat gambarnya di sini.



Setelah itu klik tombol “Exit”

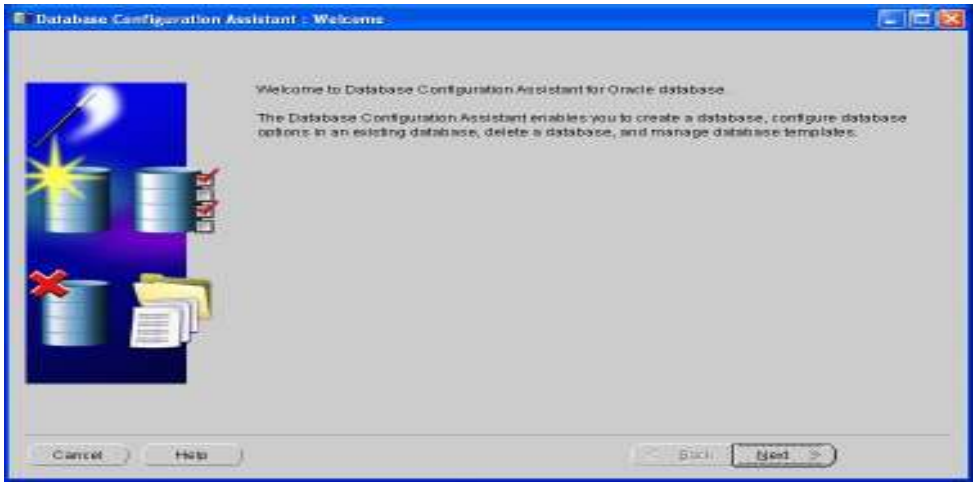
2.2. Step-step membuat database Oracle 10g

Setelah meng-install software database (RDBMS) Oracle, sekarang saatnya membuat database. Baik di Windows maupun Unix (Linux, Sun Solaris, IBM AIX, HP UX, dan lain-lain) caranya sama saja. Membuat database, bisa dengan mamakai SQL script (via SQLPlus) ataupun GUI (wizard) yang disediakan Oracle. Di versi 8i, tool GUI tersebut adalah **dbassist**; sementara versi 9i ke atas adalah **dbca**.

Untuk Windows lokasi dbca ada directory %ORACLE_HOME%/bin , sementara untuk Unix di $ORACLE_HOME/bin. Secara struktur, lokasi file-file software Oracle baik di Windows maupun Unix adalah sama saja. Yang berbeda hanya penulisan parameter. Di Windows, parameter diapit oleh tanda % (contoh %ORACLE_HOME%) sedangkan di Unix parameter didahului oleh tanda $ (contoh $ORACLE_HOME)

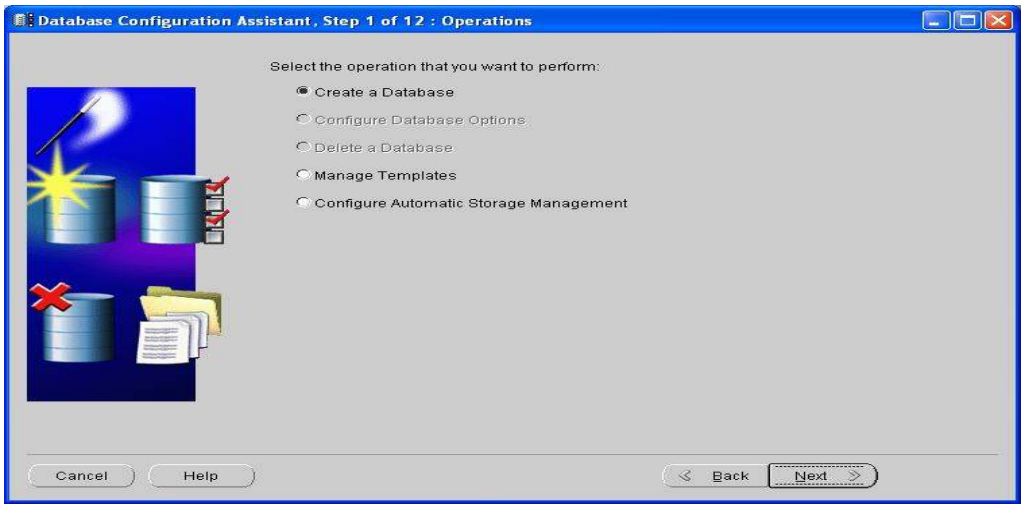
Berikut ini adalah langkah-langkah (step-step) untuk membuat database 10g:

1. Jalankan command dbca. Akan muncul form wellcome. Lihat gambarnya di sini.



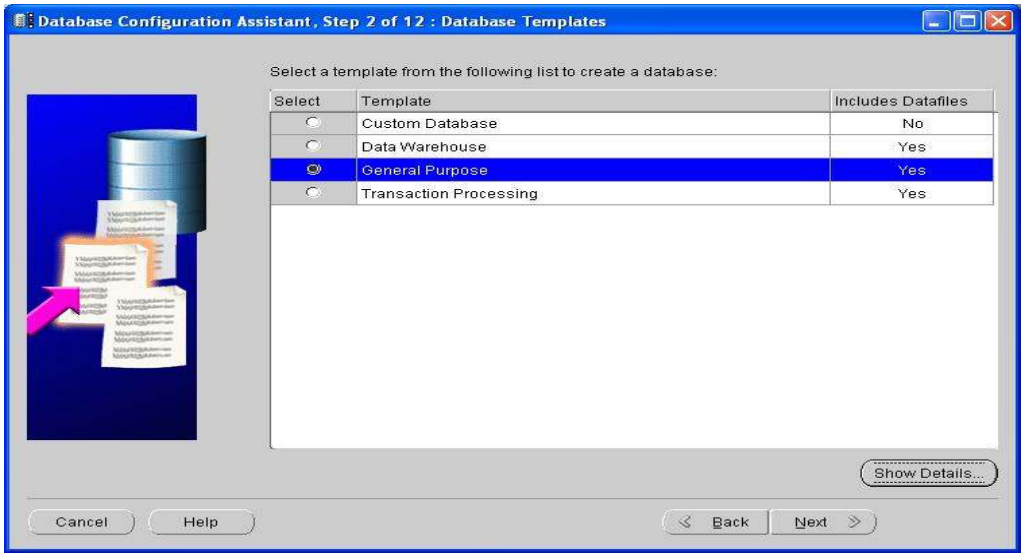
   Selanjutnya klik tombol Next.

2. Berikutnya keluar form pilihan option. Lihat gambarnya di sini.



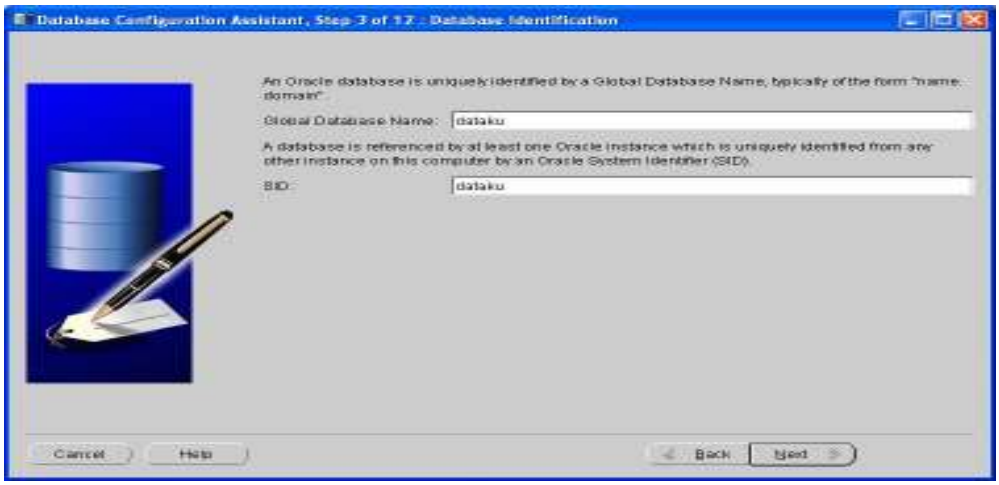
Pilih “Create a Database”. Selanjutnya klik tombol Next.

3. Berikutnya keluar form Database template. Lihat gambarnya di sini.



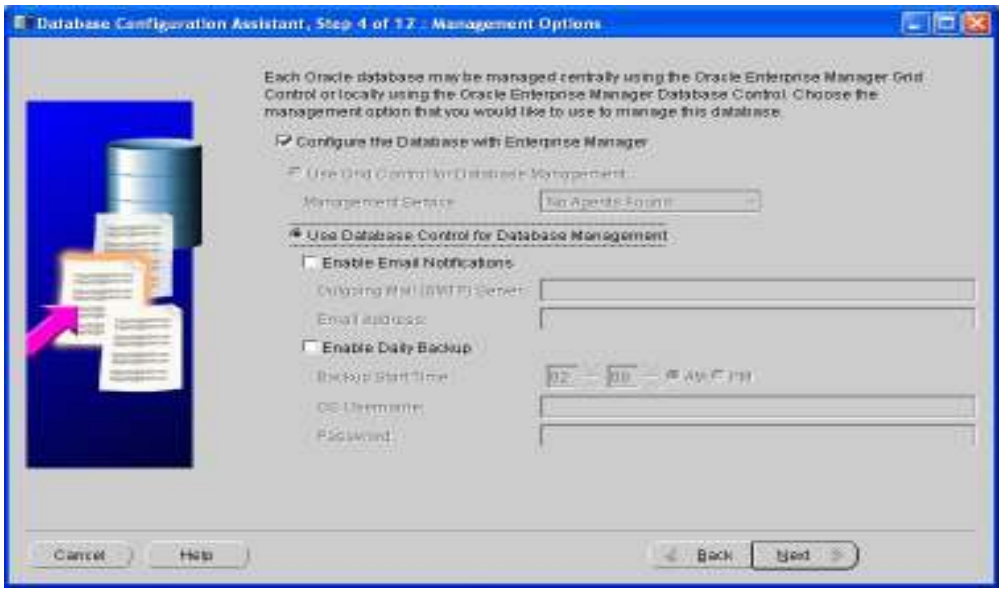
Ada 3 pilihan template. Kalau tidak mau pakai templete, pilih “Custom Database”. Dalam contoh ini saya memilih template “General Purpose”. Selanjutnya klik tombol Next.

4. Berikutnya muncul form Database Identification. Lihat gambarnya di sini.



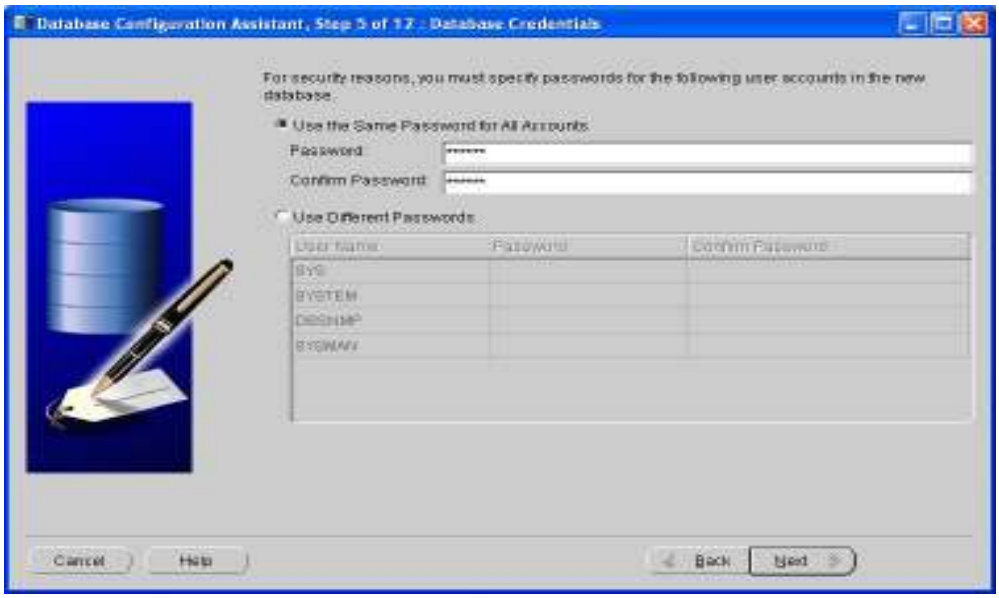
Ini nama database. Terserah mau diberi nama apa. Di sini saya namai "dataku". Selanjutnya klik tombol Next.

5. Berikutnya mucul form Management Option. Lihat gambarnya di sini.



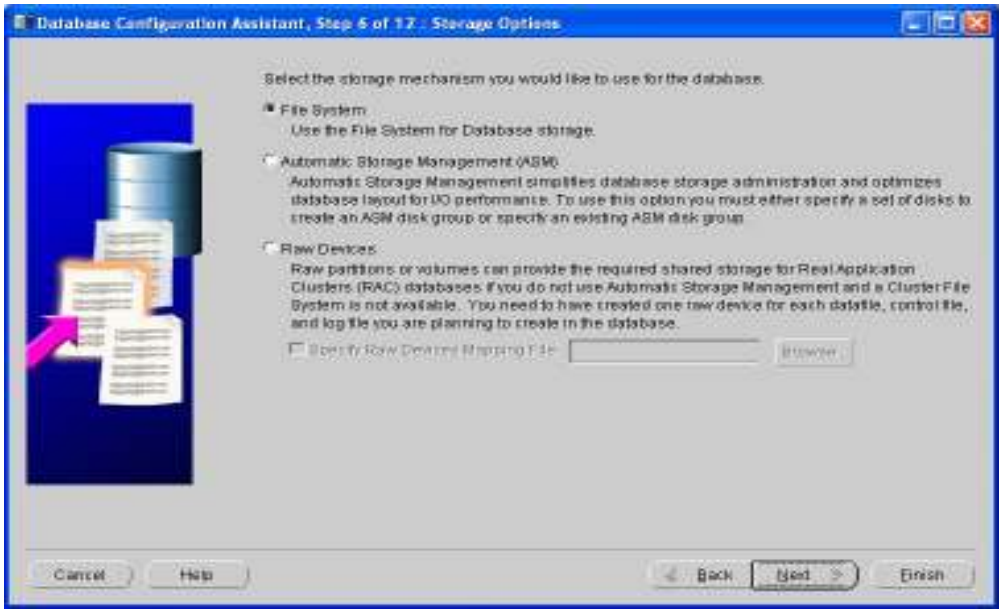
Saya memilih "Configure the Database with Enterprise Manager". Selanjutnya klik tombol Next.

6. Berikutnya mucul form Database Credential. Lihat gambarnya di sini.



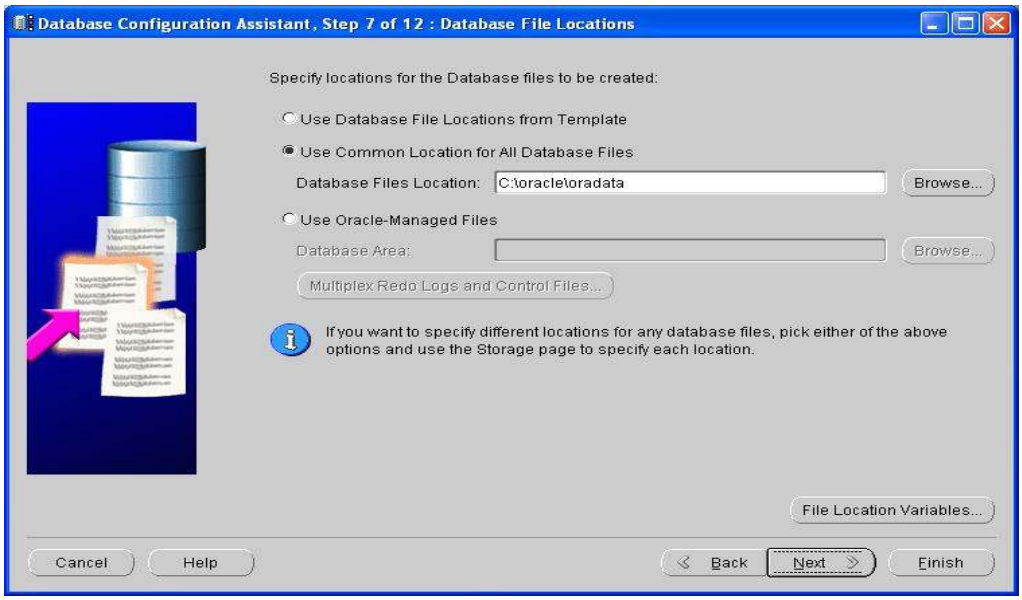
Masukkan password. Untuk mempermudah biar tidak gampang lupa, saya memilih password yang sama untuk semua account. Selanjutnya klik tombol Next.

7. Berikutnya mucul form Storage Option. Lihat gambarnya di sini.



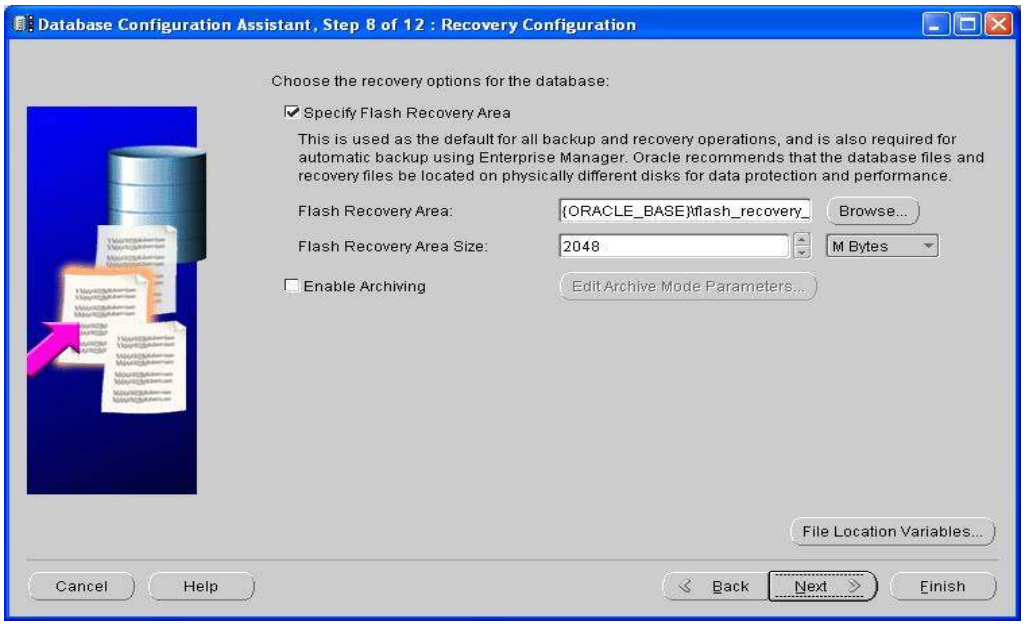
Saya memilih “File system”. Selanjutnya klik tombol Next.

8. Berikutnya mucul form Database File Location. Lihat gambarnya di sini.



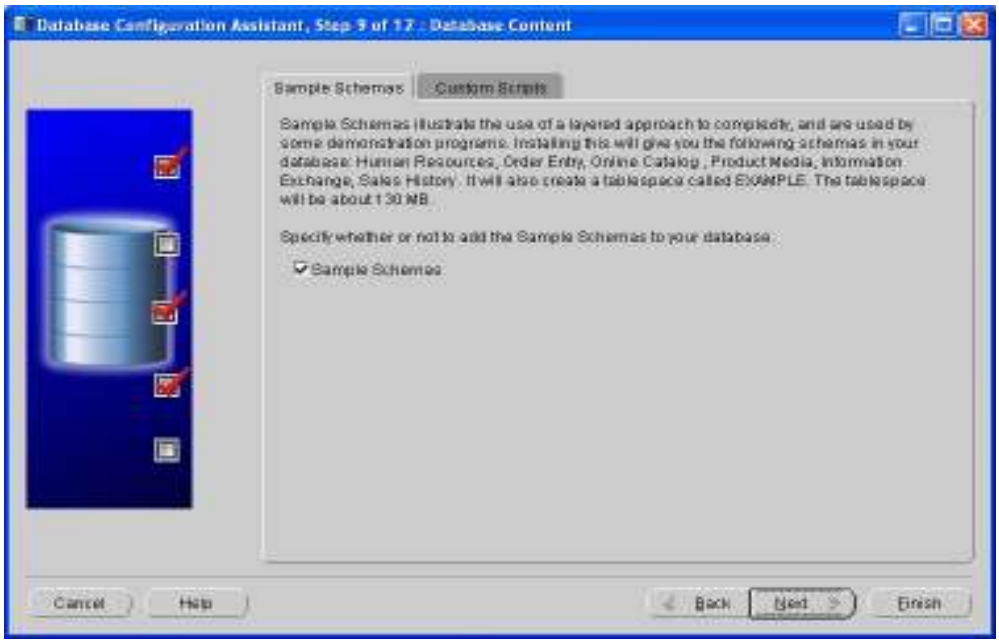
Saya memilih “C:\oracle\oradata”. Selanjutnya klik tombol Next.

9. Berikutnya mucul form Recovery Configuration. Lihat gambarnya di sini.



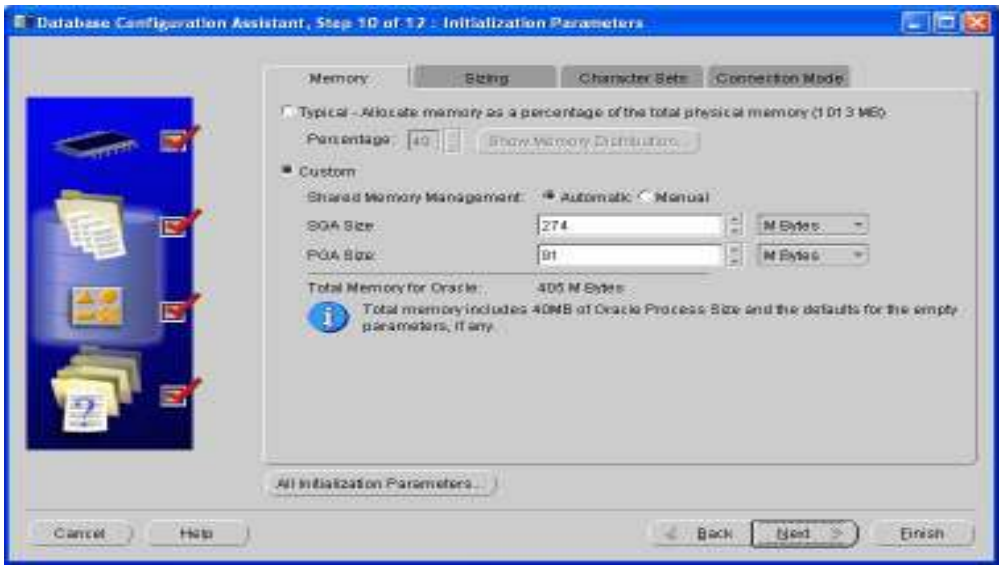
Saya memilih untuk menggunakan Flash Recovery Area. Saya memakai direktori default. Selanjutnya klik tombol Next.

10. Berikutnya mucul form Database Content. Lihat gambarnya di sini.



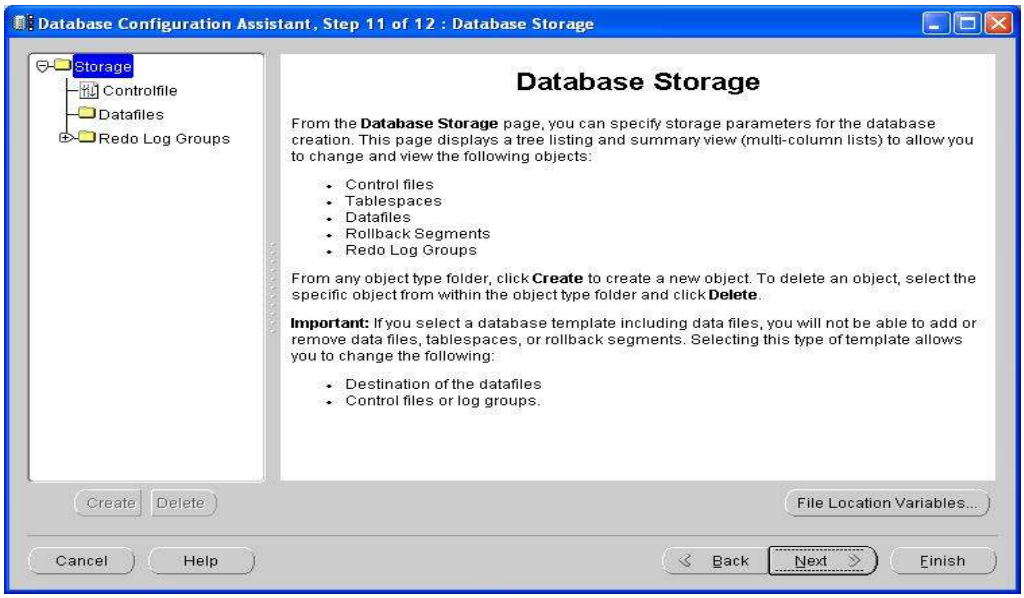
Sample schema berisi schema-schema (berserta object: table, index, view, dll) contoh dari Oracle. Ini bermanfaat bagi yang sedang belajar. Selanjutnya klik tombol Next.

11. Berikutnya mucul form Initialization Parameter. Lihat gambarnya di sini.



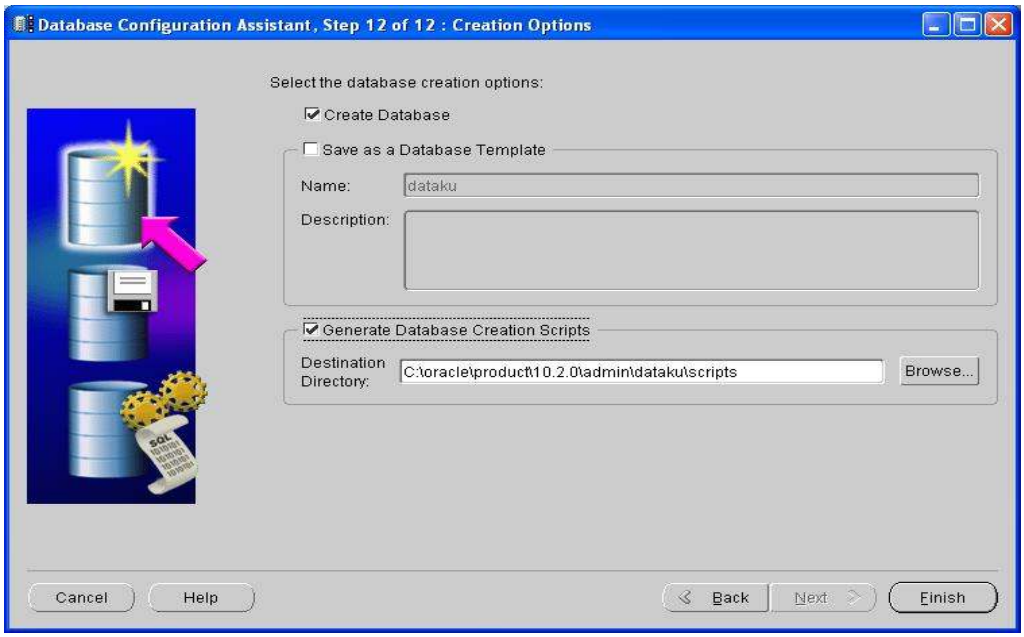
Untuk memori, saya memilih custom, dengan shared memory management: Automatic. Nilai SGA sesuaikan dengan memory komputer kita. Dengan memory PC 1G, saya masukkan SGA: 274M. Selainnya saya memakai value (nilai) default. Selanjutnya klik tombol Next.

12. Berikutnya mucul form Database storage. Lihat gambarnya di sini.



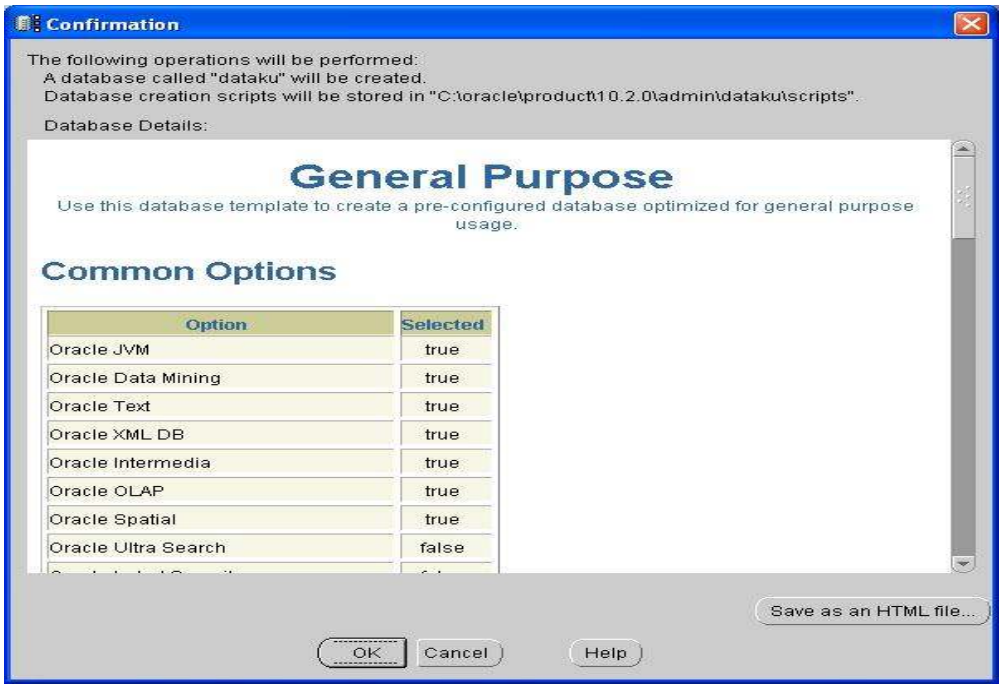
Lokasi file-file adalah sesuai dengan directory yang telah kita pilih tadi. Melalui wizard ini kita bisa mengubah ke direktori lain. Selanjutnya klik tombol Next.

13. Berikutnya mucul form Creation Option. Lihat gambarnya di sini.



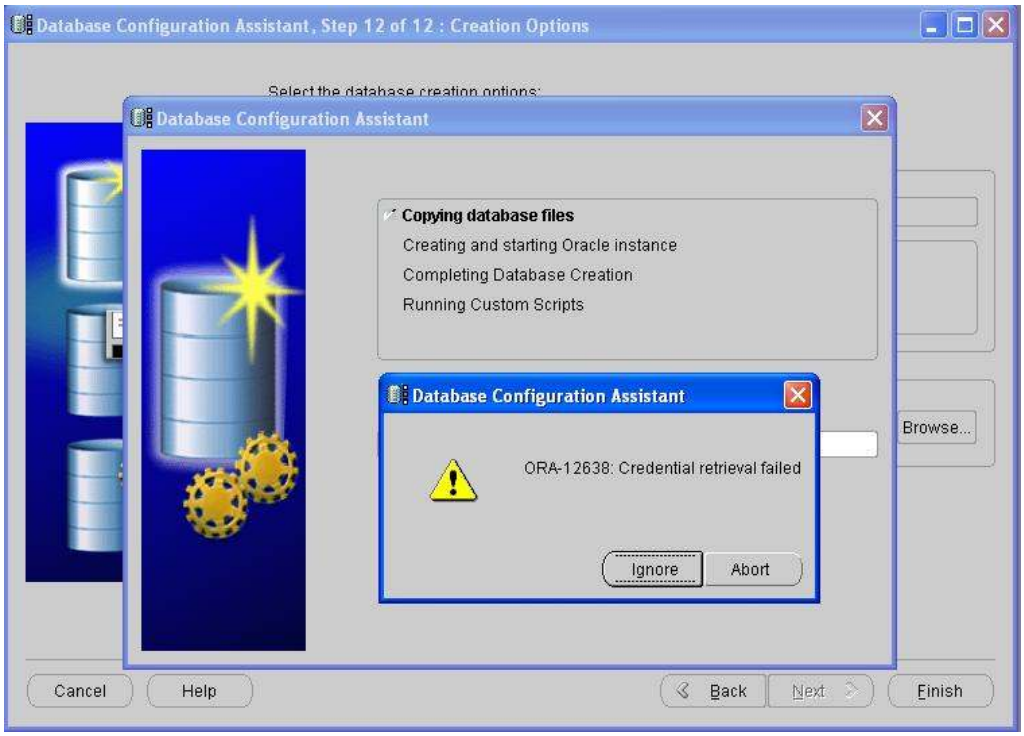
Pilih “Create Database” karena kita akan membuat database. Pilih juga “Generate database creation scripts”. Bermanfaat bagi yang belajar, untuk memahai command-command apa yang dijalankan ketika membuat database. Selanjutnya klik tombol Finish.

14. Berikutnya muncul form Confirmation. Lihat gambarnya di sini.



Berisi tentang resume database yang akan kita buat. Selanjutnya klik tombol OK.

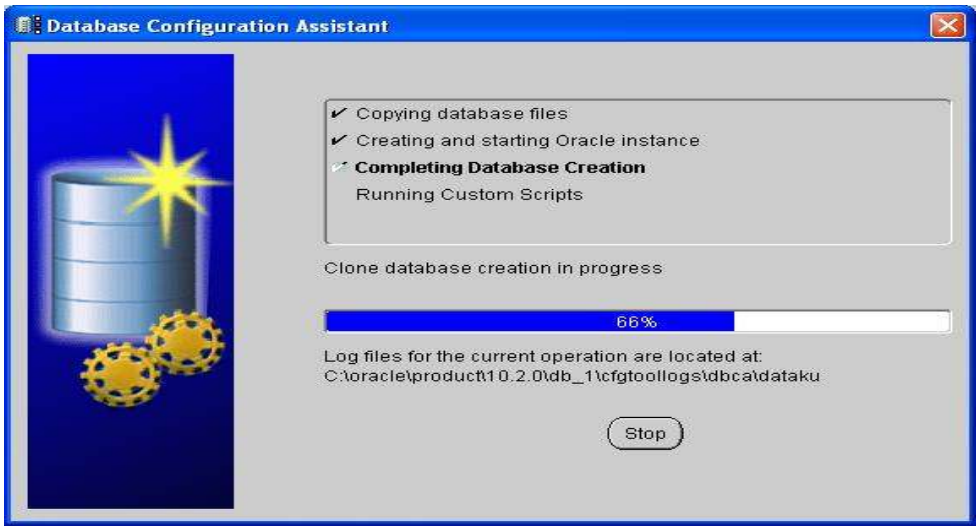
15. Berikutnya mucul Installation Progress. Kalau create di Windows mungkin muncul error, lihat gambarnya di sini.



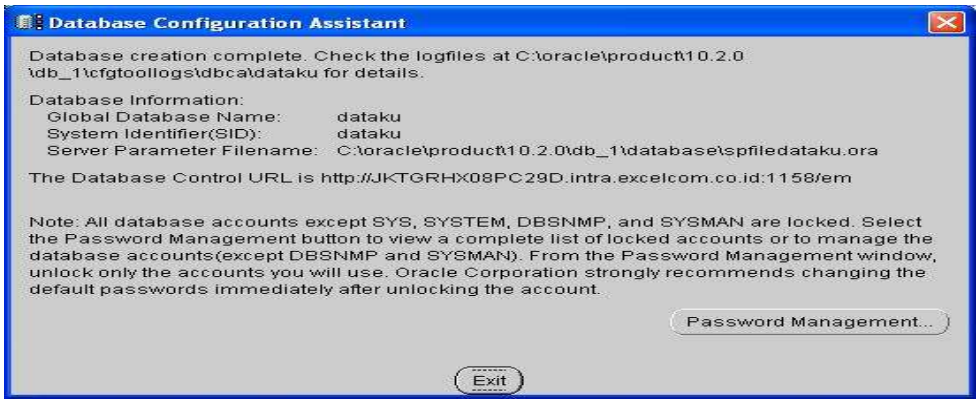
Kalau menemui error ini, tambahkan parameter berikut di file `%ORACLE_HOME%/network/admin/sqlnet.ora`

```
SQLNET.AUTHENTICATION_SERVICES = (NONE)
```

Bila file tersebut belum ada, buatlah. Jika file sudah ada dan nilai parameter tersebut bukan NONE, ubah menjadi NONE. Selanjutnya klik Ignore, nanti akan kembali ke form “Creation Option”, ulangi (lanjutkan) ke step-step berikutnya. Bila berhasil, maka “Creation progress” akan menunjukkan proses pembuatan database yang sedang berjalan. lihat gambarnya di sini



16. Akhirnya proses selesai. Lihat gambarnya di sini.



Selanjutnya klik tombol Exit. Setelah selesai, bila di WIndows, ubah nilai SQLNET.AUTHENTICATION_SERVICES dari NONE menjadi NTS`SQLNET.AUTHENTICATION_SERVICES = (NTS)` Ini penting agar bisa mudah memanage database, kapan-kapan akan saya bahas.

2.3. Membuat Listener

Setelah install software Oracle dan membuat database Oracle, kini saatnya kita untuk mengakses database lewat jaringan. Dari sisi server (Oracle database) diperlukan **listener**, sementara dari sisi client diperlukan Local Net Service Name (**TNS Names**).

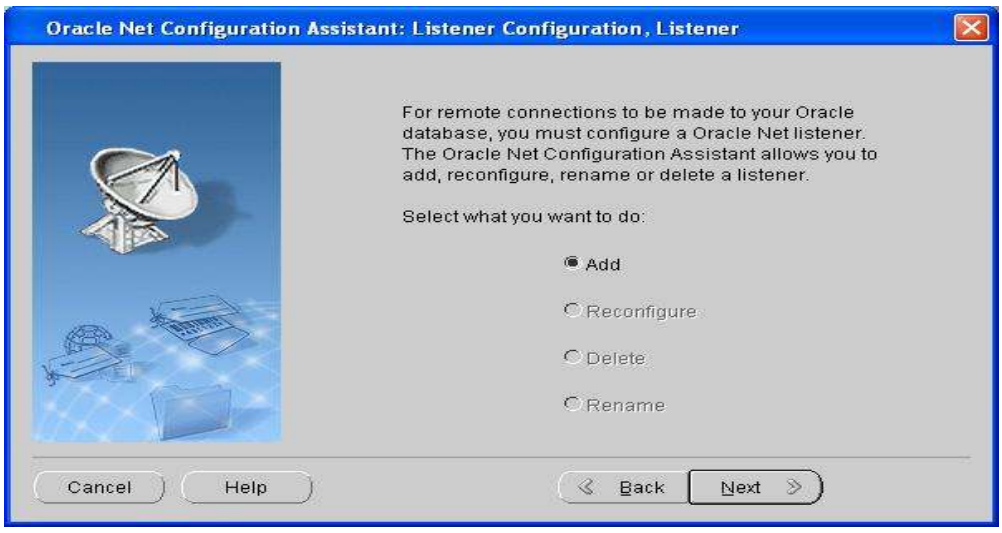
Listener bisa dibuat dengan GUI (wizard) ataupun melalui command line. Di Oracle 8i ke atas, GUI (tool) tersebut adalah **netca.** Seperti tool-tool database Oracle yang lain, lokasinya ada di $ORACLE_HOME/bin.

Berikut ini langkah-langkah membuat listener pakai netca. Sebagai contoh saya menggunakan Oracle 10g. Secara umum adalah sama untuk Oracle versi lainnya.

1. Jalankan command netca. Akan muncul form wellcome. Lihat gambarnya di sini.

   Pilih "Listener Configuration". Selanjutnya klik tombol Next.

2. Berikutnya keluar form Listener Configuration. Lihat gambarnya di sini.

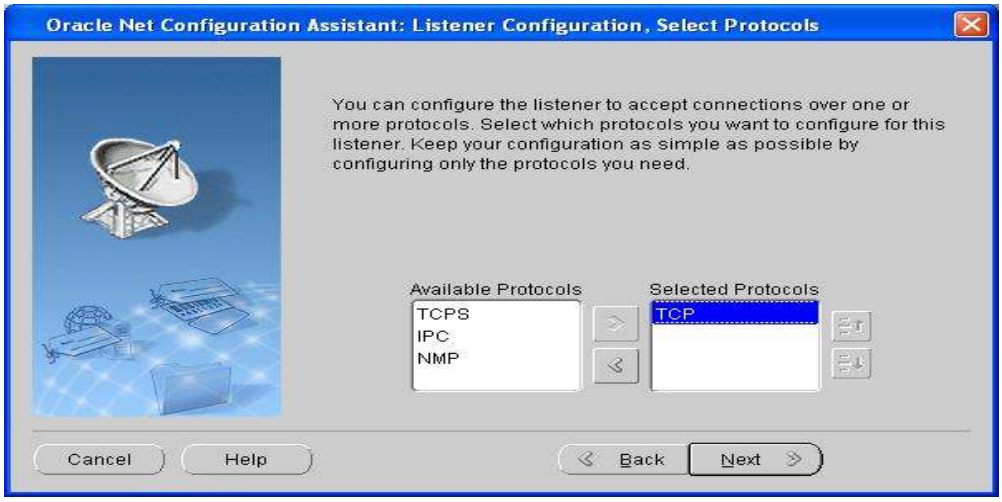


   Pilih "Add". Selanjutnya klik tombol Next.

3. Berikutnya keluar form Listener Name. Lihat gambarnya di sini.

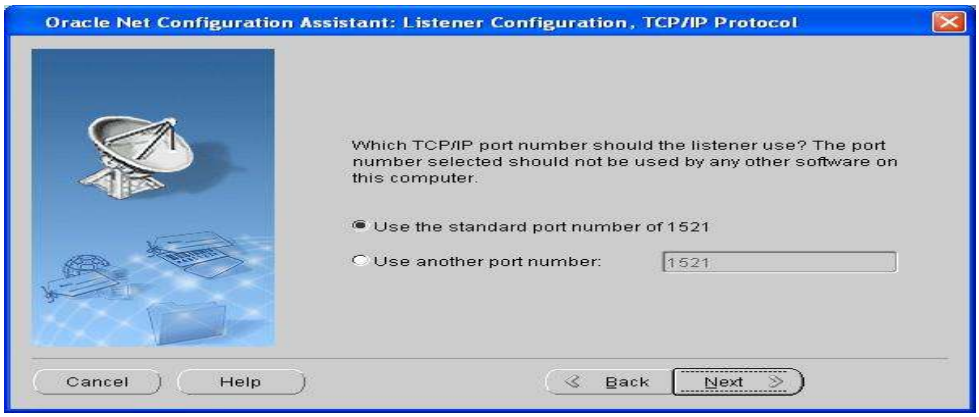
Masukkan nama listener. Kita bisa menamakan apa saja. Di sini saya biarkan pakai nama default, yaitu LISTENER. Selanjutnya klik tombol Next.

4. Berikutnya keluar form Select Protocol. Lihat gambarnya di sini.



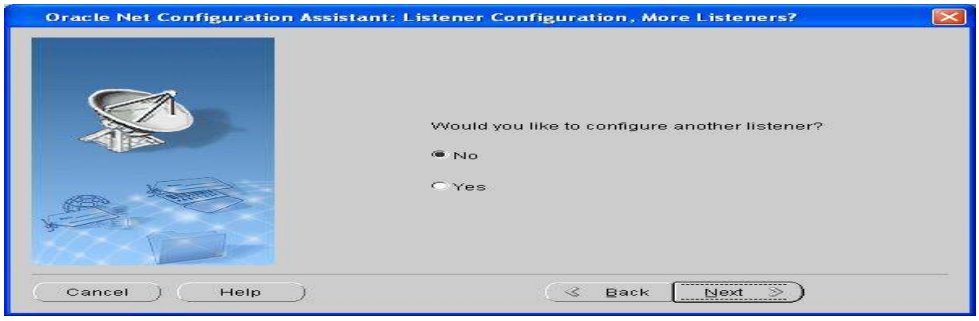
Saya memilih TCP. Selanjutnya klik tombol Next.

5. Kalau milih TCP, berikutnya akan keluar form TCP/IP Protocol. Lihat gambarnya di sini.



Kemudian pilih port yang akan digunakan. Saya menggunakan port default, 1521. Selanjutnya klik tombol Next.

6. Berikutnya akan keluar form More Listener. Lihat gambarnya di sini.



Apakah kita ingin membuat listener yang lain lagi?. Saya pilih tidak (No). Selanjutnya klik tombol Next.

7. Akhirnya selesai. Lihat gambarnya di sini.

Kalau ingin melakukan perkerjaan lainnya, klik tombol Next. Karena saya cukup membuat listener ini saja, ya sudah, klik tombol Cancel.

Pada dasarnya secara intrinsik, Oracle melakukan 2 hal berikut ini (membuat listener pakai command line ya dengan cara berikut ini):

1. Membuat file konfigurasi untuk listener di `$ORACLE_HOME/network/admin/listener.ora`.

   Adapun isi file tersebut adalah:

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(SID_NAME = PLSExtProc)
(ORACLE_HOME = C:\oracle\product\10.2.0\db_1)
(PROGRAM = extproc)
)
)LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = 10.21.108.70)(PORT = 1521))
(ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
)
)
```

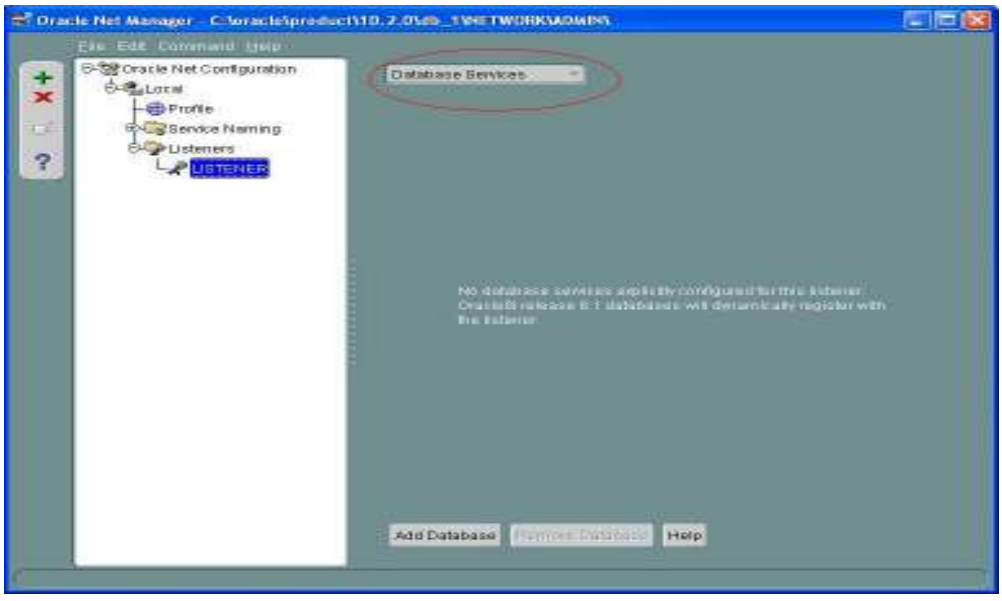
2. Menjalankan (menaikkan) proses listener

```
lsnrctl start LISTENER
```

Selanjutnya, masukkan database yang telah kita buat tadi ke dalam konfigurasi listener agar database bisa dilayani listener. Sebenarnya kalau listener dibuat (dan dinaikkan) dulu sebelum membuat database, konfigurasi ini dilakukan secara otomatis oleh dbca (bila kita membuat database menggunaka dbca).

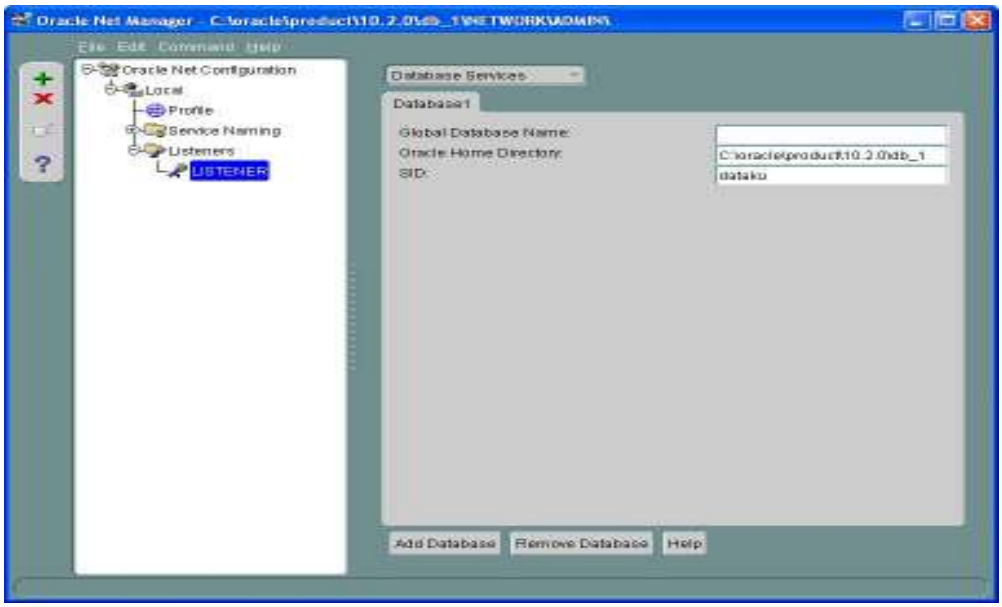
Di Oracle 9i keatas, untuk melakukannya kita bisa menggunakan tool **netmgr** yang ada di $ORACLE_HOME/bin. Di Windows sepertinya netmgr tidak dibuatkan exe-nya di %ORACLE_HOME%/bin, tapi dibuatkan sort cut-nya di menu program. Berikut ini step-step menggunakan netmgr:

1. Jalankan command netmgr. Kemudian muncul Wizard-nya. Lihat gambarnya di sini.

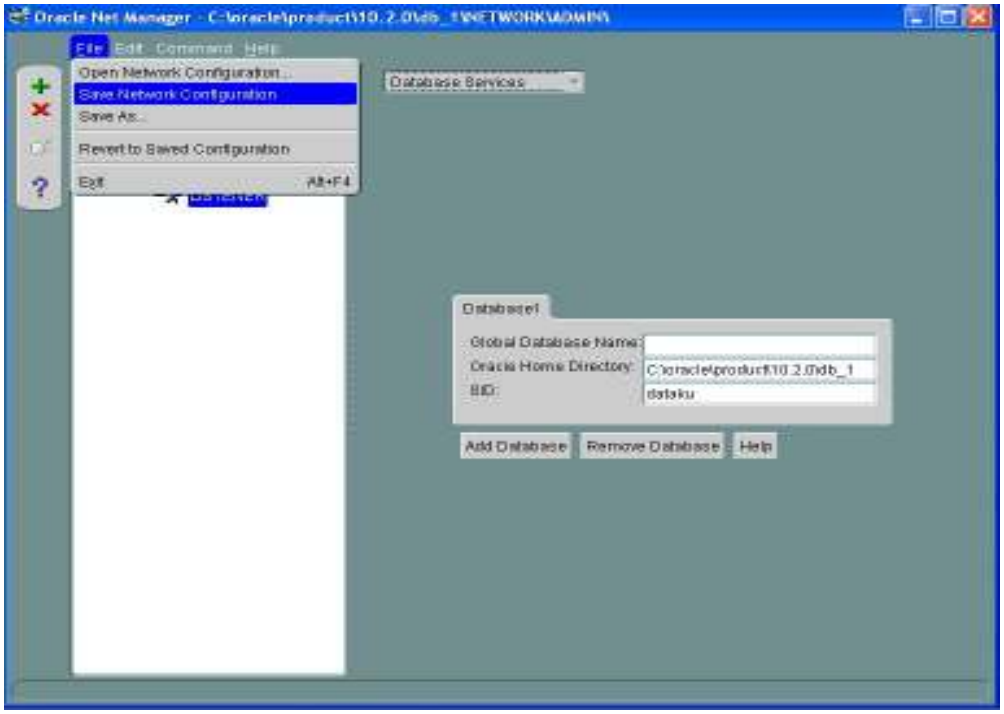


   Di tab, pilih “Database Services”. Selanjutnya klik tombol “Add Database”.

2. Kemudian masukkan ORACLE_HOME dan nama instance (SID) yang akan di-manage. Lihat gambarnya di sini



3. Setelah itu simpan konfigurasi tersebut. Lihat gambarnya di sini.



Selesai deh.

4. Agar konfigurasi bisa update, restart listener

```
lsnrctl stop LISTENER
lsnrctl start LISTENER
```

Secara intrinsik, netmgr menambahkan definisi konfigurasi di file `$ORACLE_HOME/network/admin/listener.ora` (di mana kita bisa melakukannya secara manual tanpa netmgr):

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(SID_NAME = PLSExtProc)
(ORACLE_HOME = C:\oracle\product\10.2.0\db_1)
(PROGRAM = extproc)
)
(SID_DESC =
(ORACLE_HOME = C:\oracle\product\10.2.0\db_1)
(SID_NAME = dataku)
)
)
```

2.4. Membuat (mensetting) TNS Names

Software Oracle sudah di-install, database sudah dibuat, dan listener juga sudah dibuat. Untuk bisa mengakses database di mesin server, suatu client (misalnya PC) harus di-install software Oracle Client. Setelah itu, kita perlu mensetting TNS Names di client tersebut. Tool GUI (wizard) di Oracle versi 8i ke atas adalah **netca**.

Ketika install software database (RDBMS) Oracle, secara otomatis di-install juga Oracle client. Jadi, di PC yang telah saya install RDBMS Oracle itu juga automatically sudah terinstall Oracle client. So, untuk belajar (lagipula karena keterbatasan jumlah komputer) kita bisa memakai satu PC sebagai server dan client sekaligus.

Berikut ini langkah-langkah (step-step) nya:

1. Jalankan command netca. Kemudian muncul form Welcome. Lihat gambarnya di sini.

   Pilih “Local Net Service Name configuration”. Selanjutnya klik tombol Next.

2. Berikutnya muncul form Net Service Name Configuration. Lihat gambarnya di sini.

Pilih “Add”. Selanjutnya klik tombol Next.

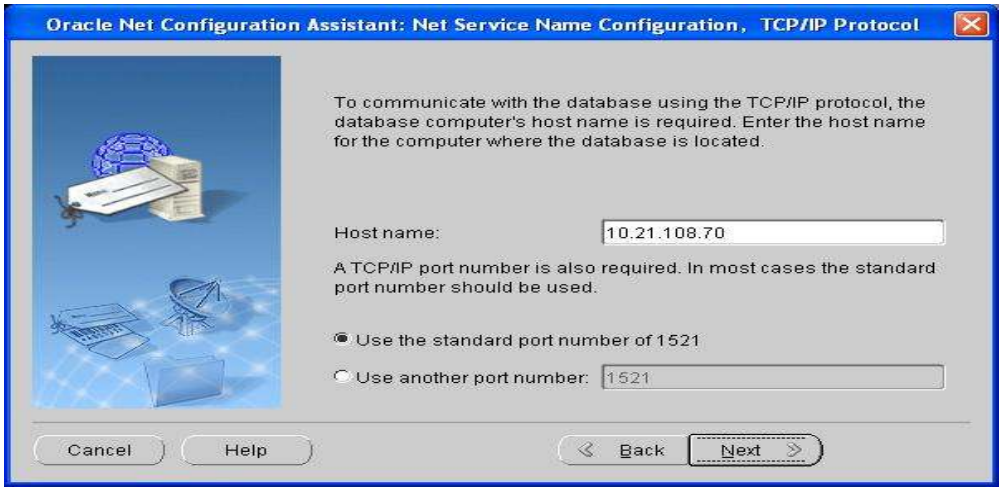
3. Berikutnya masukkan “Service Name”. Lihat gambarnya di sini.

Gampangnya saja, service name adalah sama dengan nama instance (SID). Di sini nama instance adalah dataku. Selanjutnya klik tombol Next.

4. Berikutnya pilih protocol. Lihat gambarnya di sini.

Protokol adalah sama dengan protokol yang dipakai listener. Saya pakai TCP. Selanjutnya klik tombol Next.

5. Berikutnya masukkan host name (alamat) database server. Lihat gambarnya di sini.



Bisa pakai IP dan bisa pakai nama komputer. Di sini saya memakai IP. Selanjutnya klik tombol Next.

6. Berikutnya muncul pertanyaan, apakah akan kita lakukan test?. Lihat gambarnya di sini.

Sebaiknya pilih “Yes”. Selanjutnya klik tombol Next.

7. Test koneksi memakai user system dengan password manager (default). Karena password system adalah oracle, maka connection error. Lihat gambarnya di sini.

Selanjutnya klik tombol “Change Login”.

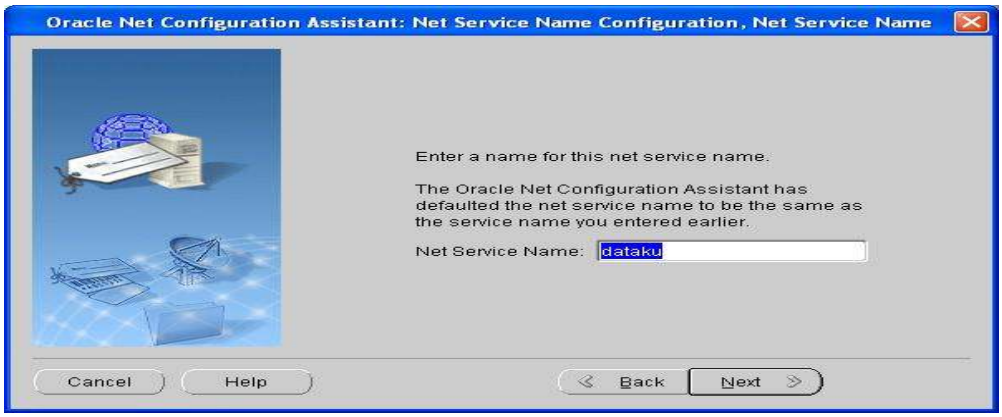
8. Masukkan user system dan password-nya yang benar. Lihat gambarnya di sini.

Selanjutnya klik tombol OK.

9. Bagus, kini koneksi success. Lihat gambarnya di sini.

Selanjutnya klik tombol Next.

10. Berikutnya masukkan nama TNS yang kita kehendaki. Lihat gambarnya di sini.



Namanya terserah kita, bebas. Di sini (by default) namanya sama dengan nama service (SID) yang tadi kita masukkan. Selanjutnya klik tombol Next.

11. Apakah akan meng-configure TNS yang lainnya?. Lihat gambarnya di sini.

Pilih "No" dan klik tombol Next.

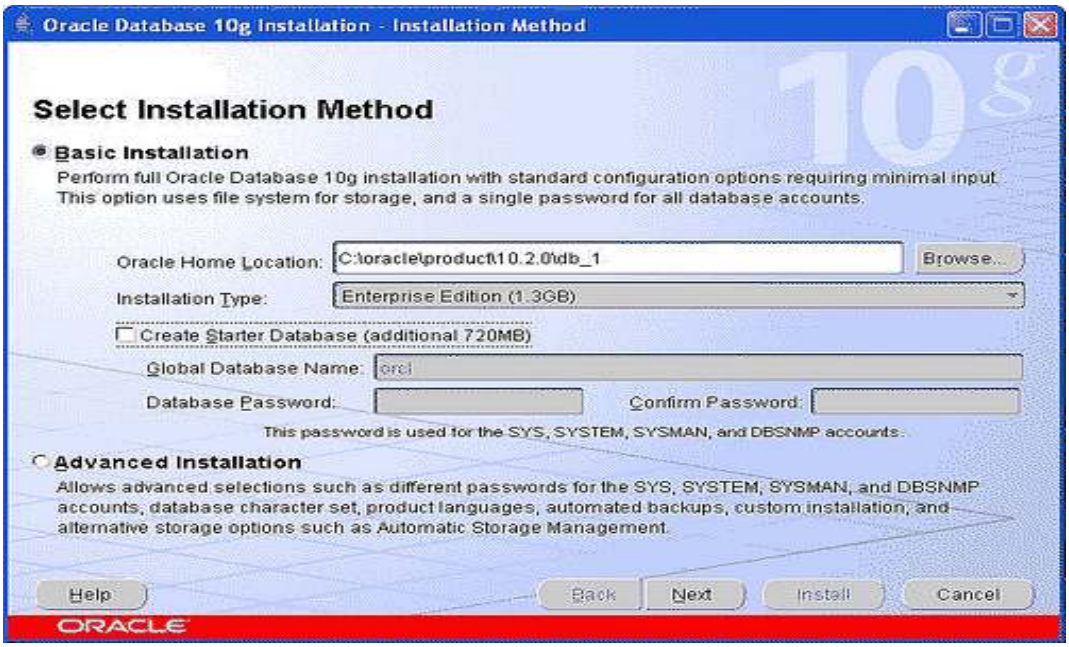
12. Selanjutnya, berhasil. Lihat gambarnya di sini.

Kalau ingin mensetting yang lainnya pakai netca, klik Next. Kalau cukup sekian ya klik Cancel.

2.5. Komputer Lambat setelah Install Database Oracle

Untuk belajar database Oracle ataupun keperluan development, kita bisa menginstall software Oracle di komputer (PC) kita. Misalkan kita menginstall database Oracle di PC kita yang punya memory 1 GB dan hard disk puluhan GB. Terus, setelah install kok performance PC jadi lambat, ya. Kenapa?

Ketika instalasi, kita ditanya apakah akan membuat database sekalian? Lihat gambar di bawah ini.



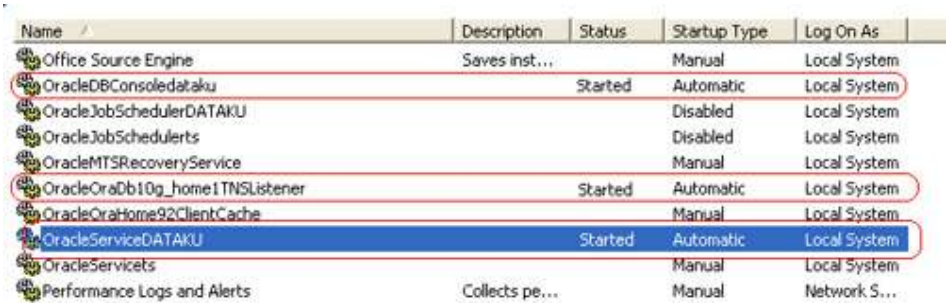
Kalau "create starter database" kita check, maka setelah installasi, installer akan membuat database. Pada contoh gambar di atas, pilihan ini tidak saya ambil.

Tentu saja database Oracle memakai memory. Kalau memory database Oracle terlalu besar untuk ukuran PC kita, jelas… PC akan jadi lambat. Untuk melihat berapa memory yang sedang dipakai, gunakan windows Task Manager.

Kalau memory yang dipakai Oracle terlalu besar untuk ukuran PC kita, untuk itu kita bisa menurunkannya. Cara menurunkan alokasi memory di database Oracle:

1. Edit init (instance) parameter.
   Misalkan saya akan menurunkan memory menjadi 200MB. Buka command prompt, jalankan SQLPLUS
   ```
   C:\>sqlplus "/ as sysdba"
   SQL> alter system set sga_target=209715200 scope=spfile;
   ```
2. Restart database
   ```
   SQL> shutdown immediate
   SQL> startup
   ```

Di Windows, database server (instance) kita dibuatkan service. Untuk melihatnya, dari **Control Panel**, buka **Administrative Tools**, kemudian buka **Services**. Lihat gambar di bawah ini (dalam contoh ini, nama database dan instance adalah DATAKU).



Startup Type services punyanya Oracle biasanya dibuat **Automatic**. Dalam contoh ini, ada 3 yaitu:
- OracleDBConsole**NAMADATABASE**
- Oracle**NAMAHOME**TNSListener
- OracleService**NAMAINSTANCE**

Agar service-service tersebut tidak segera naik waktu PC baru startup, ubah **Startup Type** menjadi **Manual**.

Terus, kalau database Oracle sedang tidak kita pakai, service-service tersebut dimatikan saja. Dan nyalakan kalau sedang dipakai saja sehingga resource (khususnya memory) tidak terbuang sia-sia.

2.6. Mengetahui konfigurasi database

Berikut ini cara mengetahui informasi seputar konfigurasi database Oracle yang kita maintain. Informasi ini sangat penting diketahui, terutama bagi DBA yang databasenya di-install/create/configure oleh orang (DBA) lain.

Informasi tentang environment Operating System (OS)

Salah satu informasi yang paling penting adalah ORACLE_HOME, yaitu directory di mana instalasi Oracle ditaruh. Lihat environment OS di sini:

1. Di Windows, lihat di registry `My Computer --> HKEY_LOCAL_MACHINE --> SOFTWARE --> ORACLE`
2. Di Unix, lihat di user profile dari database owner. Untuk shell sh/ksh/bash, user profile ada di file `.profile`. Untuk shell csh/tcsh, user profile ada di file `.login`

Informasi parameter Instance (init file)

Lokasinya ada di `$ORACLE_HOME/dbs` untuk OS Unix, atau `%ORACLE_HOME%\database` untuk Windows. Lebih detail silahkan lihat Teori dan Administrasi init file (pfile dan spfile).

Parameter instance, kalau tidak di-specify di init file, maka Oracle akan memberi nilai default. Ketika instance up, kita bisa melihat parameter-parameter instance melalui view v$parameter. Contoh:

```
SQL> select NAME,VALUE,ISDEFAULT from v$parameter order by name;
```

Kolom NAME adalah nama parameter. Kolom VALUE adalah nilai dari parameter tersebut. Kolom ISDEFAULT, kalau bernilai YES berarti nilai tersebut masih merupakan nilai default atau belum di-specify.

Feature (option) database yang sudah ter-install

Ketika meng-install software database Oracle, kita diberi 3 pilihan jenis instalasi yaitu standart, enterprise, atau custom. Feature (option) mana saja yang akan turut di-install, ya tergantung pilihan kita tersebut. Untuk mengetahui option apa saja yang include dalam software database Oracle yang telah kita install, query ke view v$option.

```
SQL> select * from v$option order by parameter;
```

Feature (option) yang dipakai oleh database

Ketika kita membuat database, tidak semua feature database Oracle (yang telah kita install) kita butuhkan. Untuk melihat feature yang dipakai oleh database, query view dba_registry

```
SQL> select COMP_NAME,VERSION,STATUS from dba_registry;
```

Jadi, perlu dipahami betul beda informasi antara yang di VIEW v$option dan di dba_registry. Feature (option) yang di-install (ada di v$option) belum tentu dipakai oleh database (ada di dba_registry). Sebaliknya, feature yang dipakai database (ada di dba_registry) sudah pasti merupakan feature yang sudah ada dalam instalasi software database Oracle (ada di view v$option).

Informasi tentang properties database

Contoh properties database adalah default tablespace (permanent dan temporary), timezone, character set, language, dll. Informasi tersebut bisa dilihat dengan query ke view database_properties.

```
SQL> select PROPERTY_NAME,PROPERTY_VALUE
from database_properties order by PROPERTY_NAME;
```

3. Basic Administration:

3.1. Memulai Koneksi ke Database

Setelah menginstall Oracle dan membuat database, untuk langkah awal administrasi adalah mulai melakukan koneksi ke database.

Administrasi dilakukan oleh user yang meng-install dan membuat database. Tool *native* dari Oracle untuk administrasi database adalah `sqlplus`, lokasi ada di `$ORACLE_HOME/bin`. Di Oracle versi 8 ke bawah, tool administrasi tersebut adalah `svrmgrl`.

Sebelum melakukan koneksi, ada OS parameter yang perlu disetting. Di Windows, parameter tersebut otomatis sudah dimasukkan ke dalam registry ketika meng-install dan membuat database pakai dbca. Di Unix, setting manual parameter berikut di user profile: ORACLE_HOME, ORACLE_SID, dan PATH.

Misalkan kita pakai shell sh atau ksh. Edit file `.profile`, tambahkan parameter berikut:

```
ORACLE_HOME=/data1/oracle/product/10.2.0; export ORACLE_HOME
ORACLE_SID=ts; export ORACLE_SID
PATH=$ORACLE_HOME/bin:$PATH; export PATH
```

Setelah mengedit file .profile, jangan lupa untuk relogin atau mengeksekusi file tersebut agar parameter yang disetting terbaca oleh current session. Berikut ini cara mengeksekusi file .profile.

```
. ./.profile
```

Koneksi pakai SQLPLus di Mesin server

Sekarang, mari kita mulai koneksi ke database. Misalkan saya akan connect pakai user system.

```
sqlplus
```

Nanti akan diminta memasukkan username dan password. Kalau belum diubah, password system adalah seperti yang ditunjukkan ketika membuat database.

Bisa juga username langsung dimasukkan ke argument-nya SQLPlus, nanti kita cuma diminta memasukkan password saja.

```
sqlplus system
```

Bisa juga langsung memasukkan username dan password. Misalkan password user system adalah oracle:

```
sqlplus system/oracle
```

Koneksi dengan langsung memasukkan username dan password sekaligus ini tidak direkomendasikan, karena password akan tampak ketika di `ps -ef`. Contoh:

```
ps -ef|grep sql
oracle  5742 25612   11:09:49 pts/1 0:00 sqlplus system/oracle
```

Cara lain juga, kita bisa masuk ke SQLPlus prompt tanpa login, kemudian jalankan perintah **connect** atau **conn** di SQL prompt. Contoh:

```
sqlplus /nolog
SQL> conn
```

Sama seperti ketika menjalankan sqlplus dari OS prompt, username dan password bisa disebutkan langsung atau tidak; kalau tidak disebutkan nanti akan ditanyakan.

```
SQL> conn system/oracle
SQL> conn system
```

Koneksi pakai user sys

User sys adalah merupakan super user, dikenal juga sebagai sysdba. Untuk koneksi pakai user sys, harus ditambahkan argument **as sysdba**. Contoh:

```
SQL> conn sys/oracle as sysdba
```

Bisa juga tanpa menyebutkan user sys, yaitu dengan memakai argument /. Contoh:

```
SQL> conn / as sysdba
```

Kalau tidak sebutkan argument as sysdba, akan muncul error berikut:

```
SQL> conn sys/oracle
ERROR:
ORA-28009: connection as SYS should be as SYSDBA or SYSOPER
Warning: You are no longer connected to ORACLE.
```

Bisa juga langsung login ketika menjalankan SQLPLUS. Contoh:

```
sqlplus "sys/oracle as sysdba"
sqlplus "/ as sysdba"
```

Koneksi dari client ke server

Untuk bisa melakukan koneksi client-server, pastikan kita sudah mensetting dan menjalakan listener di server database, dan mensetting TNSNames di client. Kalau belum punya instalasi Oracle client di mesin/komputer/PC lain, kita bisa memanfaatkan database server sebagai client sekaligus. Ketika kita install software database Oracle, by default juga diinstall Oracle client; sehingga nantinya kita bisa melakukan koneksi client-server di mesin server database kita.

Pada koneksi client-server , tambahkan argument `@namatns`. Contoh:

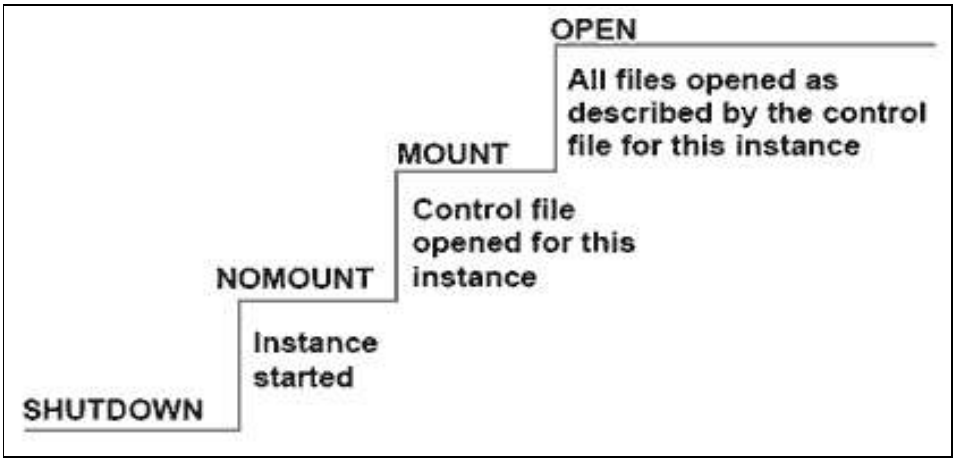
```
sqlplus system@tsprimary
sqlplus system/oracle@tsprimary
```

```
sqlplus "sys@tsprimary as sysdba"
sqlplus "sys/oracle@tsprimary as sysdba"

SQL> conn system@tsprimary
SQL> conn system/oracle@tsprimary
SQL> conn sys@tsprimary as sysdba
SQL> conn sys/oracle@tsprimary as sysdba
```

3.2. Startup dan shutdown instance

Administrasi (aktivitas) yang bisa kita lakukan pada instance adalah **startup**, **shutdown**, dan **alter**. Secara umum proses startup adalah sebagai berikut:

1. Database mati (shutdown)
   Background process belum naik. Memori belum dialokasikan
2. nomount
   Backgroung process dinaikkan. Memory dialokasikan
3. mount
   Instance membaca control file. Control file berisi konfigurasi database. Instance belum membaca data file.
4. open
   Instance sudah membaca data file (header). Database siap diakses



Command (perintah) startup :

```
startup
startup open
startup nomount
startup mount
startup force
```

Command "startup" saja tanpa argument, by default adalah "startup open"
Command "startup force" adalah sama saja dengan "shutdown abort" kemudian "startup"

Command shutdown :

```
shutdown normal
shutdown transactional
shutdown immediate
shutdown abort
```

Berikut ini perbandingan proses shutdown normal (N), transactional (T), immediate (I), dan abort (A):

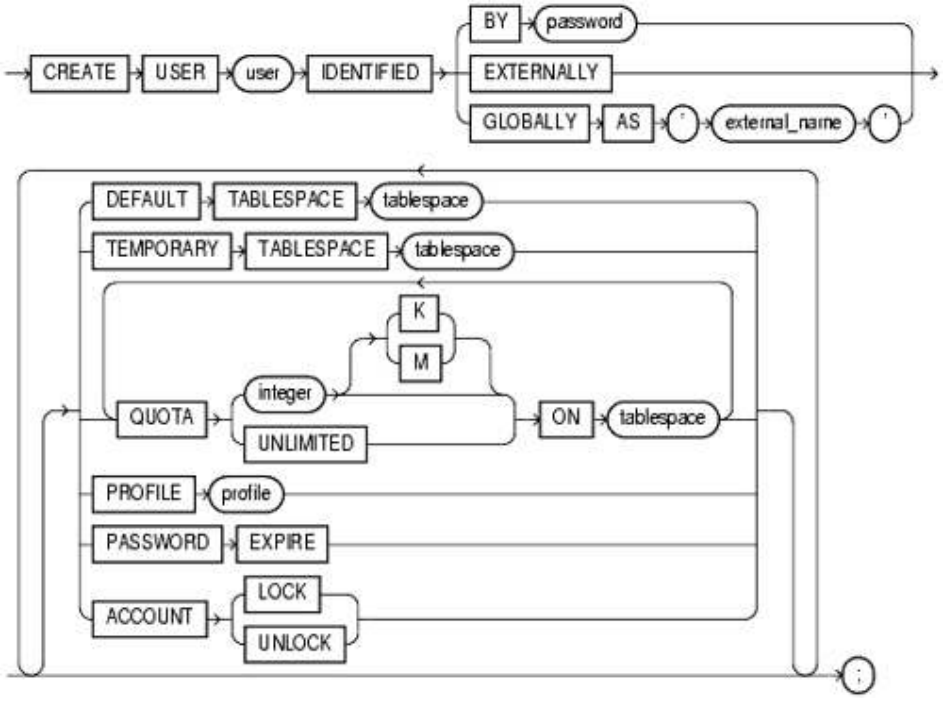
Proses	Shutdown			
	A	I	T	N
Memperbolehkan koneksi baru	No	No	No	No
Menunggu sampai semua session disconnect	No	No	No	Yes
Menunggu sampai semua transaksi selesai	No	No	Yes	Yes
Melakukan checkpoint dan close file	No	Yes	Yes	Yes

3.3. Administrasi User

Waktu kita membuat database, secara otomatis Oracle membuat user sys dan system. User sys bisa melakukan apa saja, oleh karenanya disebut sebagai super user. User system digunakan untuk administrasi database sehari-hari, misalnya membuat user, tablespace, dan lain-lain. Kalau user sys bisa melakukan semuanya, kenapa mesti ada user system? Ya tentu saja, pertimbangannya adalah masalah security. Oke, mari kita mulai membahas administrasi user di database Oracle.

Create (membuat) user

Berikut ini format perintah SQL untuk membuat user [Gambar diambil dari dokumentasi Oracle 9i]



Perintah membuat user yang paling sederhana adalah (contoh, saya akan membuat user dengan nama ROHMAD dan passwordnya PASSROH):

```
SQL> conn system
SQL> create user ROHMAD identified by PASSROH;
```

Karena tidak di-specify, maka atribut-atribut yang lainnya memakai nilai default. Untuk melihatnya lihat di view dba_users

```
SQL> select * from dba_users where username='ROHMAD';
```

Ini hasilnya:

```
USERNAME: ROHMAD
ACCOUNT_STATUS : OPEN
EXPIRY_DATE :
DEFAULT_TABLESPACE : USERS
TEMPORARY_TABLESPACE : TEMP
PROFILE : DEFAULT
INITIAL_RSRC_CONSUMER_GROUP : DEFAULT_CONSUMER_GROUP
```

Untuk melihat default tablespace dan default temporary tablespace dari suatu database, silahkan lihat di Administrasi Tablespace.

Berikut ini contoh membuat user dengan men-specify default tablespace, quota pemakaian di tablespace, dan temporary tablespace (ini contoh yang paling sering digunakan):

```
SQL> CREATE USER rohmad
IDENTIFIED BY passroh
DEFAULT TABLESPACE users
QUOTA unlimited ON users
QUOTA 500K ON data_ts
TEMPORARY TABLESPACE temp;
```

Alter (mengubah) user

Semua atribut user bisa diubah (alter) kecuali username itu sendiri. Secara umum perintah alter sama dengan create, hanya mengganti kata create menjadi alter. Contoh:

```
SQL> alter user ROHMAD identified by PASSROH2;
```

Grant User

Setelah user dibuat, user tersebut tidak bisa melakukan koneksi sebelum diberi grant (hak) untuk connect ke database.

```
SQL> conn rohmad/passroh2
ERROR:
ORA-01045: user ROHMAD lacks CREATE SESSION privilege; logon denied
```

Berikut ini perintah untuk memberi grant ke user agar bisa connect ke database

```
SQL> conn system;
SQL> grant connect to rohmad;
```

Setelah itu, dengan user rohmad tersebut, mari kita coba membuat table

```
SQL> conn rohmad/passroh2
SQL> create table tab_test (no number);
ERROR at line 1:
ORA-01031: insufficient privileges
```

OO… ternyata user rohmad belum punya priviledges untuk membuat table. Coba beri privilege ke user rohmad agar bisa membuat tabel

```
SQL> conn system;
SQL> grant create table to rohmad;
```

Sekarang buat table

```
SQL> conn rohmad/passroh2
```

```
SQL> create table tab_test (no number);
ERROR at line 1:
ORA-01950: no privileges on tablespace 'USERS'
```

OO… ternyata error. Walaupun default tablespace untuk user ROHMAD adalah tablespace USERS, ternyata ROHMAD masih belum bisa membuat table (menulis) di tablespace USERS. Beri quota ke user ROHMAD, bisa unlimited ataupun di-specify besarannya. Contoh, beri quota unlimited:

```
SQL> conn system;
SQL> alter user rohmad quota unlimited on USERS;
```

Coba lagi, dan berhasil …

```
SQL> conn rohmad/passroh2
SQL> create table tab_test (no number);
Table created.
```

Agar bisa membuat index, user rohmad harus diberi grant. Demikian juga untuk bisa membuat view. Wow… dapat anda bayangkan, berapa banyak privilege yang mesti gita beri (grant)? Untuk mengatasi itu, Oracle membuat **role**. Role berisi beberapa (banyak) privilege. Contoh, role RESOURCE berisi grant untuk membuat table, membuat index, quota unlimited di default tablespace, dan lain-lain.

Berikut ini perintah untuk memberi hak (grant) role RESOURCE ke user ROHMAD

```
SQL> conn system
SQL> grant RESOURCE to rohmad;
```

Umumnya, cukup dengan memberi grant CONNECT dan RESOURCE ke user yang baru dibuat, user tersebut sudah bisa melakukan banyak pekerjaan.

```
SQL> create user ROHMAD identified by PASSROH;
SQL> grant connect to rohmad;
SQL> grant RESOURCE to rohmad;
```

Menghapus User

```
SQL> drop user rohmad;
```

Kalau user tersebut mempunyai object (misalnya tabel), maka akan muncul error berikut

```
ORA-01922: CASCADE must be specified to drop 'ROHMAD'
```

Kalau begitu, tambahkan parameter CASCADE untuk sekaligus menghapus semua object yang dimiliki user tersebut

```
SQL> drop user rohmad CASCADE;
```

Referensi

1. Oracle9i Database Administrator's Guide Release 2 (9.2) - Managing Users and Resources
2. Oracle9i Database Administrator's Guide Release 2 (9.2) - Managing User Privileges and Roles

3.4. Administrasi Control File

Selain datafile dan log dfile, control file merupakan salah satu file utama database Oracle. Secara global ada baiknya kita lihat lagi arsitektur database Oracle. Informasi yang disimpan di contol file di antaranya adalah nama database, lokasi datafile dan logfile, nomor SCN, dan lain-lain.

View-view dictionary yang datanya diambil dari control file di antaranya adalah: v$DATABASE, v$DATAFILE, v$TEMPFILE, v$LOGFILE, V$LOG, dan lain-lain. Control file dibaca Oracle ketika instance sedang mount. Oleh karena itu, view-view tersebut bisa di-query meskipun database dalam keadaan mount (belum open).

Lokasi
Lokasi control file ditunjukkan oleh parameter control_files. Silahkan temukan parameter control_files ini di instance parameter (init file atau spfile). Atau gunakan SQL command berikut:

```
SQL> show parameter control_files
SQL> -- atau
SQL> select name from v$controlfile;
SQL> -- atau
SQL> select NAME,VALUE from  v$parameter where NAME ='control_files';
```

Melihat isi control file

Untuk melihat informasi apa saja yang disimpan di control file, gunakan SQL command berikut:

```
SQL> select * from V$CONTROLFILE_RECORD_SECTION order by type;
```

Untuk melihat definisi control file, backup control file ke file text. Gunakan perintah ini:

```
SQL> alter database backup controlfile to trace;
```

Perintah di atas akan membuat trace file yang disimpam di direktori user_dump_dest. Gunakan perintah SQL "show parameter user_dump_dest" untuk melihat lokasi direktori tersebut. Format file adalah NAMAISNTANCE_ora_OSID.trc. Di contoh saya ini, trace file yang di-generate adalah **ts_ora_22363.trc**.

Isi trace file ini adalah script untuk re-create control file. Ada dua bagian, yaitu recreate dengan opsi NORESETLOGS atau RESETLOGS. Berikut ini adalah content control file (dengan opsi NORESETLOGS)

```
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "TS" NORESETLOGS FORCE LOGGING ARCHIVELOG
MAXLOGFILES 16
MAXLOGMEMBERS 3
MAXDATAFILES 100
MAXINSTANCES 8
MAXLOGHISTORY 292
LOGFILE
```

```
GROUP 4 '/oradata/oracle/ts/redo04.log'  SIZE 5M,
GROUP 5 '/oradata/oracle/ts/redo05.log'  SIZE 5M,
GROUP 6 '/oradata/oracle/ts/redo06.log'  SIZE 5M
DATAFILE
'/oradata/oracle/ts/system01.dbf',
'/oradata/oracle/ts/undotbs01.dbf',
'/oradata/oracle/ts/sysaux01.dbf',
'/oradata/oracle/ts/users01.dbf2',
'/oradata/oracle/ts/test01.dbf',
'/oradata/oracle/ts/test02.dbf'
CHARACTER SET WE8MSWIN1252
;
RECOVER DATABASE
ALTER SYSTEM ARCHIVE LOG ALL;
ALTER DATABASE OPEN;
ALTER TABLESPACE TEMP ADD TEMPFILE '/oradata/oracle/ts/temp01.dbf2'
SIZE 20971520  REUSE AUTOEXTEND ON NEXT 655360  MAXSIZE 32767M;
```

Bagain kedua, yaitu recreate dengan opsi RESETLOGS, intinya sama saja. Hanya berbeda pada poin-poin berikut:

```
CREATE CONTROLFILE REUSE DATABASE "TS" RESETLOGS FORCE LOGGING ARCHIVELOG
RECOVER DATABASE USING BACKUP CONTROLFILE
ALTER DATABASE OPEN RESETLOGS;
```

Menambah, multiplexing (mirroring) control file

Ketika membuat database dengan dbca, by default Oracle akan membuat 3 control file. Isi ketiga control file tersebut adalah sama. Sebenarnya, 1 control file saja cukup. Kita dianjurkan untuk membuat lebih dari 1 untuk jaga-jaga (multiplex/mirror), kalau-kalau salah satu control file corrupt kita masih punya yang lainnya. Oleh karena itu, idealnya masing-masing control file ditaruh di file system (drive) yang berbeda.

Multiplexing dilakukan dengan menambah control file. Misalkan kita sudah punya 3 control file berikut:

```
/oradata/oracle/ts/control01.ctl
/oradata/oracle/ts/control02.ctl
/oradata/oracle/ts/control03.ctl
```

Saya ingin menambah 1 control file lagi dan saya taruh di directory **/data1/oracle**. Berikut ini step-stepnya:

1. Shutdown database

   ```
   SQL> shutdown immediate
   ```

2. Copy control file. Bisa yang mana saja, toh isinyanya sama semua

   ```
   cd /data1/oracle/
   cp -rp /oradata/oracle/ts/control03.ctl control04.ctl
   ```

3. Edit parameter control_files di instance parameter file (initfile atau spfile). Kalau pakai init file, edit file $ORACLE_HOME/dbs/initts.ora (dalam contoh ini, nama instance adalah **ts**). Tambahkan control file yang baru tersebut ke definisi control_files:

```
control_files='/oradata/oracle/ts/control01.ctl',
'/oradata/oracle/ts/control02.ctl',
'/oradata/oracle/ts/control03.ctl',
'/data1/oracle/control04.ctl'
```

Kalau pakai spfile, kita tidak bisa mengedit spfile pakai text edior, mau tidak mau ya harus startup instance (cukup startup nomount saja):

```
SQL> startup nomount
SQL> alter system set control_files=
'/oradata/oracle/ts/control01.ctl',
'/oradata/oracle/ts/control02.ctl',
'/oradata/oracle/ts/control03.ctl',
'/data1/oracle/control04.ctl'
scope=spfile;

SQL> shutdown immediate
```

4. Startup database. Check bahwa sekarang control file sudah bertambah

```
SQL> startup
SQL> select name from v$controlfile;

NAME
-------------------------------------
/oradata/oracle/ts/control01.ctl
/oradata/oracle/ts/control02.ctl
/oradata/oracle/ts/control03.ctl
/data1/oracle/control04.ctl
```

Me-rename (memindahkan) control file

Step-stepnya hampir sama dengan menambah control file. Misalkan kita akan memindahkan control file **/data1/oracle/control04.ctl** ke directory /data2/oracle/

1. shutdown database
2. pindahkan (rename) control file

```
cd /data2/oracle/
mv /data1/oracle/control04.ctl /data2/oracle/
```

3. Edit instance parameter file (initfile atau spfile). Ganti control file yang lama (**/data1/oracle/control04.ctl**) menjadi yang baru (**/data2/oracle/control04.ctl**):

```
control_files='/oradata/oracle/ts/control01.ctl',
'/oradata/oracle/ts/control02.ctl',
'/oradata/oracle/ts/control03.ctl',
'/data2/oracle/control04.ctl'
```

4. startup database

Mengurangi (delete/drop) control file

Step-stepnya juga hampir sama saja dengan menambah control file, intinya adalah membuang (take out) control file (yang akan dibuang) dari parameter control_files. Misalkan saya ingin menghapus control file /data2/oracle/control04.ctl.

1. shutdown database
2. Take out **/data2/oracle/control04.ctl** dari parameter control_files.

```
control_files='/oradata/oracle/ts/control01.ctl',
'/oradata/oracle/ts/control02.ctl',
'/oradata/oracle/ts/control03.ctl'
```

3. startup database

Referensi

Oracle® Database Administrator's Guide 10g Release 2 (10.2) - Managing Control Files

3.5. Administrasi Tablespace

Tablespace merupakan bagian dari arsitektur logic database Oracle [secara sekilas, struktur logik database Oracle adalah tablespace, segment, extent, dan block]. Tablespace digunakan sebagai tempat (storage) bagi segment. Segment adalah object database yang mempunyai data. Yang termasuk segment adalah table, index, cluster, rollback (undo), lobsegment, lobindex, table partition, index partition, lob partition, temporary segment, dll. Gunakan query berikut untuk melihat type-type segment yang ada di database kita

```
SQL> select distinct SEGMENT_TYPE from dba_segments;
```

Secara fisik, tablespace terdiri atas satu atau lebih datafile. Informasi tentang tablespace ada di view **v$tablespace** , **dba_tablespaces**, **dba_data_files**, **dba_temp_files**, dll.

Gunakan command berikut untuk melihat tipe-tipe tablespace

```
SQL> select distinct CONTENTS from dba_tablespaces;
```

Berdasarkan hasil query tersebut, berikut ini 3 tipe tablespace:

- UNDO. Untuk menyimpan rollback (undo) segment
- TEMPORARY. Untuk menyimpan temporary segment
- PERMANENT. Untuk menyimpan segment selain dua di atas (contoh tabel, index)

UNDO TABLESPACE

1. Contoh membuat Undo Tablespace dengan nama undotbs2, datafile /oradata/oracle/ts_bak/undotbs201.dbf, ukuran file sebesar 10M. Jangan lupa tambahkan option **undo** sesudah create.

   ```
   SQL> create undo tablespace undotbs2
   datafile '/oradata/oracle/ts_bak/undotbs201.dbf' size 10m;
   ```

2. Untuk menambah (menaikkan size/space) dapat dilakukan dengan manaikkan size dari datafile atau menambah datafile

   ```
   SQL> alter database
   datafile '/oradata/oracle/ts_bak/undotbs201.dbf' resize 20m;

   SQL> alter tablespace undotbs2 add
   datafile '/oradata/oracle/ts_bak/undotbs202.dbf' size 10m;
   ```

3. Untuk melihat datafile dan size dari tablespace UNDOTBS2

   ```
   SQL> select file_name,bytes from dba_data_files
   where tablespace_name='UNDOTBS2';
   ```

4. Untuk melihat free space tiap-tiap datafile dari tablespace UNDOTBS2

   ```
   SQL> select a.name, sum(b.bytes) from v$datafile a, dba_free_space b
   where a.file#=b.file_id and b.TABLESPACE_NAME='UNDOTBS2' group by
   a.name;
   ```

5. Untuk melihat undo tablespace yang aktif saat ini gunakan

```
SQL> show parameter undo_tablespace
```

Untuk mengubah undo_tablespace ke tablespace yang baru saja kita buat

```
SQL> alter system set undo_tablespace=UNDOTBS2;
```

TEMPORARY TABLESPACE

1. Contoh membuat temporay tablespace dengan nama TEMP2, tempfile /oradata/oracle/ts/temp21.dbf, ukuran file sebesar 10M. Jangan lupa tambahkan option temporary sesudah create, dan gunakan tempfile bukan datafile.

```
SQL> create temporary tablespace temp2
tempfile '/oradata/oracle/ts/temp21.dbf' size 10m;
```

2. Untuk menambah (menaikkan size/space) dapat dilakukan dengan manaikkan size dari tempfile atau menambah tempfile

```
SQL> alter database
tempfile '/oradata/oracle/ts/temp21.dbf' resize 20m;
```

```
SQL> alter tablespace temp2 add
tempfile '/oradata/oracle/ts/temp22.dbf' size 10m;
```

3. Untuk melihat temp file (file-file milik TEMPORARY tablespace) dan sizenya. Contoh, misalkan nama TEMPORARY tablespace tersebut adalah TEMP:

```
SQL> select file_name,bytes from dba_temp_files where
tablespace_name='TEMP';
```

4. Untuk melihat free spacenya

```
SQL> select a.name, sum(b.BYTES_FREE) from v$tempfile a,
V$TEMP_SPACE_HEADER b where a.file#=b.file_id and
b.TABLESPACE_NAME='TEMP' group by a.name;
```

5. Untuk melihat temporary tablespace yang digunakan sebagai DEFAULT di database adalah

```
SQL> select PROPERTY_VALUE from database_properties
where PROPERTY_NAME='DEFAULT_TEMP_TABLESPACE';
```

Untuk mengubah default temporary tablespace menjadi tablespace yang baru saja kita buat

```
SQL> alter database default temporary tablespace temp2;
```

PERMANENT TABLESPACE

1. Contoh membuat permanent tablespace dengan nama DATA, datafile /oradata/oracle/ts_bak/data01.dbf, ukuran file sebesar 10M.

```
SQL> create tablespace DATA
datafile '/oradata/oracle/ts_bak/data01.dbf' size 10m;
```

2. Untuk menambah (menaikkan size/space) dapat dilakukan dengan manaikkan size dari datafile atau menambah datafile. Caranya sama persis seperti pada UNDO tablespace

```
SQL> alter database
datafile '/oradata/oracle/ts_bak/data01.dbf' resize 20m;
```

```
SQL> alter tablespace DATA add
datafile '/oradata/oracle/ts_bak/data02.dbf' size 10m;
```

3. Untuk melihat datafile, size, dan free size dari PERMANENT tablespace; caranya seperti untuk UNDO tablespace, yaitu gunakan view **dba_data_files**, **v$datafile**, dan **dba_free_space**.
4. Untuk melihat permanent tablespace yang digunakan sebagai DEFAULT di database adalah

```
SQL> select PROPERTY_VALUE from database_properties where
PROPERTY_NAME='DEFAULT_PERMANENT_TABLESPACE';
```

Untuk mengubah default permanent tablespace menjadi tablespace yang baru saja kita buat

```
SQL> alter database default tablespace data;
```

MENGURANGI SIZE DARI TABLESPACE

1. Dilakukan dengan mengurangi size dari datafilenya. Perintah untuk mengurangi size adalah sama dengan perintah untuk menambah size, intinya adalah mengubah size (RESIZE). Jangan lupa, untuk temporary tablespace gunakan TEMPFILE; untuk PERMANENT dan UNDO tablespace sama, gunakan DATAFILE.

```
SQL> alter database
tempfile '/oradata/oracle/ts/temp21.dbf' resize 20m;
SQL> alter database
datafile '/oradata/oracle/ts/undotbs1.dbf' resize 20m;
```

Catatan penting
Penguranga size (resize) tidak bisa dilakukan pada block di bawah **high water mark**. High water mark adalah posisi block tertinggi yang pernah dipakai untuk extent. Nanti kapan-kapan saya bahas tentang **high water mark** ini. Eksekusi akan error kalau resize dilakukan di bawah High water mark:

```
ORA-03297: file contains used data beyond requested RESIZE value
```

Best practice-nya, kalau misalkan size datafile 4G, dan kita ingin menurunkan size-nya, lakukan secara gradual (diturunkan 100M - 100M) untuk menemukan size (high water mark) yang sesuai.

2. Dilakukan dengan menghapus temp file
   Untuk alasan keamanan, datafile tidak bisa dihapus. Ingat, yang dimaksud datafile adalah file-file milik tablespace PERMANENT dan UNDO.

```
SQL> alter database
datafile '/oradata/oracle/ts/test02.dbf' drop;
ERROR at line 1:
ORA-01916: keyword ONLINE, OFFLINE, RESIZE, AUTOEXTEND or END/DROP
expected
```

   Sedangkan temp file bisa dihapus (file milik tablespace TEMPORARY) karena file ini tidak berisi data. Dengan catatan, paling tidak sisakan 1 tempfile.

```
SQL> alter database
tempfile '/oradata/oracle/ts/temp02.dbf2' drop;
```

3. contoh kasus
   - **Pertanyaan**
     Bagaimana cara untuk resize tablepspace SYSTEM yang besar nya sudah 3G, padahal yang ke pakai cuma 500M, sudah di coba pake alter tablespace resize, tetapi tidak bisa .
   - **Jawaban**
     Resize tidak bisa dilakukan karena dulunya space 3G itu pernah kepakai. Mungkin dulu pernah sempat ada segment (table/index/temp segment) yang memakai tablespace SYSTEM, namun sekarang sudah dihapus.

     Konsep yang berkaitan dengan hal ini adalah "High Water Mark".

     Kalau size tablespace (datafile) tidak bisa dikurangi dengan "alter database datafile '…' resize" sementara itu free space-nya masih sangat banyak, satu-satunya solusi adalah recreate tablespace yang bersangkutan. Caranya:
     - export data-data yang ada di tablespace tsb
     - create tablespace baru
     - import data-data tsb ke tablespace baru
     - drop tablespace lama.

     Namun sayangnya, tablespace SYSTEM tidak bisa di-recreate. Kalau masih mau dipaksa, ya dengan recreate database:
     - export database full
     - buat database baru
     - import database
     - drop database lama

MENGHAPUS (drop) TABLESPACE

Perintahnya sama untuk ketiga jenis tablespace tersebut. Contoh

```
SQL> drop tablespace DATA;
```

Referensi dari Oracle documentation:
http://download.oracle.com/docs/cd/B19306_01/server.102/b14231/tspaces.htm
http://download.oracle.com/docs/cd/B19306_01/server.102/b14231/dfiles.htm
http://download.oracle.com/docs/cd/B19306_01/server.102/b14231/undo.htm

3.6. Memindahkan atau Me-rename Datafile

Kadang kita perlu memindahkan datafile dari satu tempat (disk/file system/directory/drive) ke tempat yang lainnya. Atau kadang juga kita perlu me-rename datafile karena ada salah ketik waktu membuatnya. Baik memindahkan file dari satu tempat ke tempat lain, maupun merename datafile di tempat yang sama, intinya adalah sama saja.

Misalkan kita ingin memindahkan/me-rename datafile dari '/oradata/oracle/ts/users01.dbf' ke '/oradata/oracle/ts/users02.dbf'

Pada database yang NOARCHIVELOG:

1. Shutdown database

```
SQL> shutdown immediate
```

2. Pindahkan/move/rename datafile. Di Windows bisa pakai Windows explorer. Di unix gunakan command ini:

```
mv /oradata/oracle/ts/users01.dbf /oradata/oracle/ts/users02.dbf
```

3. Startup mount database

```
SQL> startup mount
```

4. Rename datafile di level database

```
SQL> alter database rename file '/oradata/oracle/ts/users01.dbf' to
'/oradata/oracle/ts/users02.dbf';
```

5. Setelah itu, open database

```
SQL> alter database open;
```

Pada database yang ARCHIVELOG:

1. Tidak perlu shutdown database. Cukup offline-kan datafile yang bersangkutan

```
SQL> alter database datafile '/oradata/oracle/ts/users01.dbf'
offline;
```

2. Pindahkan/move/rename datafile. Di Windows bisa pakai Windows explorer. Di unix gunakan command ini:

```
mv /oradata/oracle/ts/users01.dbf /oradata/oracle/ts/users02.dbf
```

3. Rename datafile di level database

```
SQL> alter database rename file '/oradata/oracle/ts/users01.dbf'
to '/oradata/oracle/ts/users02.dbf';
```

4. Setelah itu, recover datafile yang telah di-rename tersebut

```
SQL> recover datafile '/oradata/oracle/ts/users02.dbf';
```

5. Terakhir, online-kan datafile yang telah di-rename tersebut

```
alter database datafile '/oradata/oracle/ts/users01.dbf' online;
```

Pada metode pertama di atas (untuk database NOARCHIVELOG), mau tidak mau database tersebut tidak bisa diakses karena harus di-shutdown dulu (ada downtime).

Sementara pada metode kedua (untuk database ARCHIVELOG) database masih bisa diakses (baik query maupun transaksi), kecuali data (bytes) yang secara intrinsik disimpan di datafile tersebut tidak bisa diakses. Misalkan datafile diatas adalah milik tablespace USERS, dan tablespace USERS punya dua datafiles; maka data yang secara intriksik ada di datafile lain (bukan yang di-offline-kan tersebut) masih bisa diakses.

3.7. Maintenance Log dan Trace File

Kalau tidak perhatian, kita bisa kaget "Lho, kok file system (drive) Oracle saya penuh! Masa sampai 40G begini? Padahal dulu waktu instalasi, cuma 5G. Lagian file-file database ditaruh di file system (drive) lain."

Yang bikin penuh itu pasti (biasanya) file-file `log` dan `trace` yang digenerate oleh Oracle. Untuk itu, kita perlu mengerti, file-file apa sih itu.

1. alert log.
   Berisi rekaman aktivitas instance (startup, shutdown, switch logfile, parameter change, error, dll). Format nama filenya `alert_NAMAINSTANCE.log`

   Contoh, database dengan nama instance (ORACLE_SID) DATAKU mempunyai alert log `alert_DATAKU.log`

   Lokasi alert log ditunjukkan oleh instance parameter `background_dump_dest`. Untuk mengetahui directory background_dump_dest, gunakan perintah berikut.
   ```
   SQL> sho parameter background_dump_dest
   ```
2. Trace file.
   Trace file ada 2 macam, yaitu system dan user trace file. Formatnya adalah `*.trc`
   Berisi rekaman aktivitas suatu proses (system atau user) yang lebih detail. Biasanya, error log secara global disebutkan di alert log, dan lebih detail ditulis di trace file.

   System trace file, lokasinya sama dengan alert log yaitu di background_dump_dest. Sementara user trace file di user_dump_dest
   ```
   SQL> sho parameter user_dump_dest
   ```

3. Core dump file
   Fungsinya seperti core dump di OS. Ketika ada masalah yang berkaitan dengan konfigurasi OS, misalnya storage accessibility, Oracle biasanya membuat core dump. Lokasinya ditunjukkan oleh parameter core_dump_dest
   ```
   SQL> sho parameter core_dump_dest
   ```

4. Audit file.
   Formatnya `ora_OSID.aud`
   Berisi history koneksi dari user sysdba (sys). Juga berisi hasil audit aktivitas user untuk AUDIT_TRAIL=OS. Lokasinya ditunjukkan oleh parameter audit_file_dest
   ```
   SQL> sho parameter audit_file_dest
   ```

5. Network log.
   Lokasinya di `$ORACLE_HOME/network/log`. Sama, di Windows ada di `%ORACLE_HOME%\network\log`. Network log ada dua, yaitu `NAMALISTENER.log` dan `sqlnet.log`. Kalau kita memakai 2 listener, maka log listener juga ada 2.

Biasanya, kelima jenis file di atas ada di satu file system (drive) yang sama. Dalam aktivitas sehari-hari, file system (tempat file-file di atas) tidak boleh penuh.

Misalkan karena suatu hal sehingga file system (drive) penuh, kita bisa menghapus file-file tersebut karena hanya berisi log (bukan konfigurasi). Memang harus dimaintenance. Kalau tidak dibutuhkan (biasanya untuk trouble shooting) ya dibuang saja. Best practisenya:

1. Audit file dan trace file di-zip (compress) dan di-keep selama beberapa bulan.
2. Setiap awal bulan, alert log di-rename (contoh menjadi alert_DATAKU_jun08.log) dan di-zip. Begitu alert log di-rename, maka Oracle akan membuat file alert log baru. Begitu juga dengan network log

Catatan:

1. Kalau file system tempat audit_file_dest penuh, kita tidak bisa login dengan user sysdba (sys). Begini errornya:
```
SQL> conn / as sysdba
ERROR:
ORA-09945: Unable to initialize the audit trail file
SVR4 Error: 28: No space left on device
ORA-09817: Write to audit file failed.
SVR4 Error: 28: No space left on device
```

2. Untuk bisa melihat parameter instance melalui SQLPlus (sho parameter NAMAPARAMETER), kita harus menggunakan user sys, system, atau user yang punya privilege DBA.Kalau kita menjumpai error di atas, maka kita tidak bisa login pakai / as sysdba. Misalkan kita lupa password system dan user DBA lainnya, tentunya kita tidak akan bisa melihat PARAMETER pakai SQlPlus. Solusinya lihat parameter tersebut di init file.

3.8. Melihat Informasi Current Session

Melihat informasi session bisa dilakukan dengan query ke VIEW V$SESSION. Namun tidak semua user bisa query ke VIEW ini. Yang bisa hanya user-user yang punya privilege berikut :

1. DBA (sys, system, dan lain-lain)
2. select on v_$session
3. select any dictionary

Meskipun untuk melihat informasi session kita sendiri, kalau tidak punya salah satu dari ketiga privilege di atas, ya tetep tidak bisa query ke VIEW v$session.

Untuk bisa melihat informasi session sendiri, selain melalui VIEW v$session, Oracle menyediakan tool/function SYS_CONTEXT.

Formatnya adalah **SYS_CONTEXT('USERENV' , 'PARAMETER')**
Untuk melihat informasi session yang kita kehendaki, kita tinggal memanggil function SYS_CONTEXT dan memasukkan parameter yang sesuai.

Contoh:
Melalui SQLPlus, melihat informasi hostname dan osuser dari mana current session berasal:

```
SQL> col "Os User" for a20
SQL> col "Hostname" for a35
SQL> select sys_context( 'USERENV','OS_USER') "Os User", sys_context(
'USERENV','HOST') "Hostname" from dual;

Os User              Hostname
-------------------- -----------------------------------
rohmadp              INTRANET\LOCALUSER_ROHMADP
```

Contoh kalau melalui PL/SQL

```
set serveroutput on
DECLARE
v_hostname varchar2(20);
v_os_user varchar2(35);
BEGIN
v_os_user:=sys_context('USERENV','OS_USER');
v_hostname:=sys_context('USERENV','HOST');
dbms_output.put_line('Os User = '||v_os_user);
dbms_output.put_line('Hostname = '||v_hostname);
END;
/

Os User  = rohmadp
Hostname = INTRANET\LOCALUSER_ROHMADP
```

***

Informasi apa saja yang bisa diambil dari SYS_CONTEXT?
Berikut ini beberapa contoh informasi yang bisa kita ambil. Lebih lengkapnya silahkan lihat referensi.

```
PARAMETER       INFORMASI SESSION
================================================================
OS_USER         User OS dari mana client melakukan koneksi
HOST            Komputer/mesin dari mana client melakukan koneksi
CURRENT_USER    User database yang dipakai untuk koneksi
DB_NAME         Nama database yang dikoneksi
Dan lain-lain   Silahkan lihat Referensi
```

Referensi

Oracle® Database SQL Reference 10g Release 2 (10.2) - SYS_CONTEXT
Oracle9i SQL Reference Release 2 (9.2) - SYS_CONTEXT

4. Advanced Administration:

4.1. Security database: Administrasi Profile

Untuk alasan keamanan, pemakaian **resource** oleh user database perlu dibatasi. Selain itu, otorisasi (**password**) user juga perlu diperketat. Di database Oracle, pembatasan itu dilakukan oleh profile.

Berikut ini informasi profile (beserta parameter-patameternya) yang ada di database:

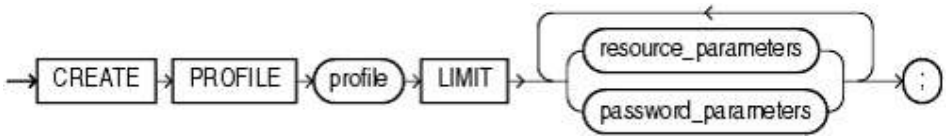
```
SQL> conn system
SQL> SELECT * FROM DBA_PROFILES ORDER BY PROFILE,RESOURCE_TYPE;

PROFILE    RESOURCE_NAME                    RESOURCE LIMIT
---------- -------------------------------- -------- ----------
DEFAULT    COMPOSITE_LIMIT                  KERNEL   UNLIMITED
DEFAULT    SESSIONS_PER_USER                KERNEL   UNLIMITED
DEFAULT    PRIVATE_SGA                      KERNEL   UNLIMITED
DEFAULT    CONNECT_TIME                     KERNEL   UNLIMITED
DEFAULT    IDLE_TIME                        KERNEL   UNLIMITED
DEFAULT    LOGICAL_READS_PER_CALL           KERNEL   UNLIMITED
DEFAULT    LOGICAL_READS_PER_SESSION        KERNEL   UNLIMITED
DEFAULT    CPU_PER_CALL                     KERNEL   UNLIMITED
DEFAULT    CPU_PER_SESSION                  KERNEL   UNLIMITED
DEFAULT    PASSWORD_VERIFY_FUNCTION         PASSWORD NULL
DEFAULT    PASSWORD_REUSE_MAX               PASSWORD UNLIMITED
DEFAULT    PASSWORD_REUSE_TIME              PASSWORD UNLIMITED
DEFAULT    PASSWORD_LIFE_TIME               PASSWORD UNLIMITED
DEFAULT    FAILED_LOGIN_ATTEMPTS            PASSWORD 10
DEFAULT    PASSWORD_LOCK_TIME               PASSWORD UNLIMITED
DEFAULT    PASSWORD_GRACE_TIME              PASSWORD UNLIMITED
```

Ketika kita membuat database, by default Oracle membuat profile dengan nama DEFAULT. Ketika kita membuat user tanpa menyebutkan profile-nya, maka user tersebut akan di-assign ke profile DEFAULT.

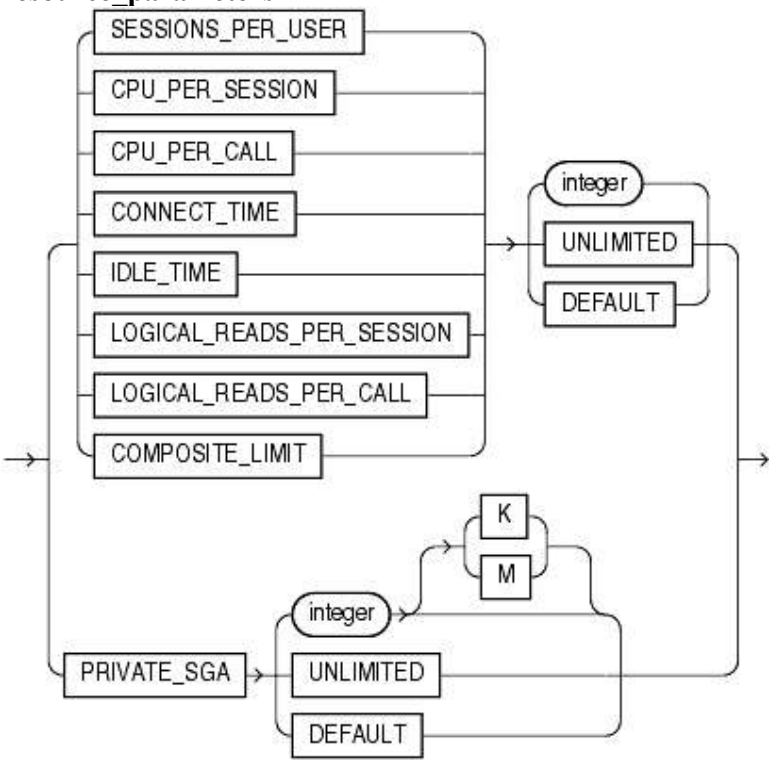
Membuat profile

Gambar diambil dari Oracle Documentation.

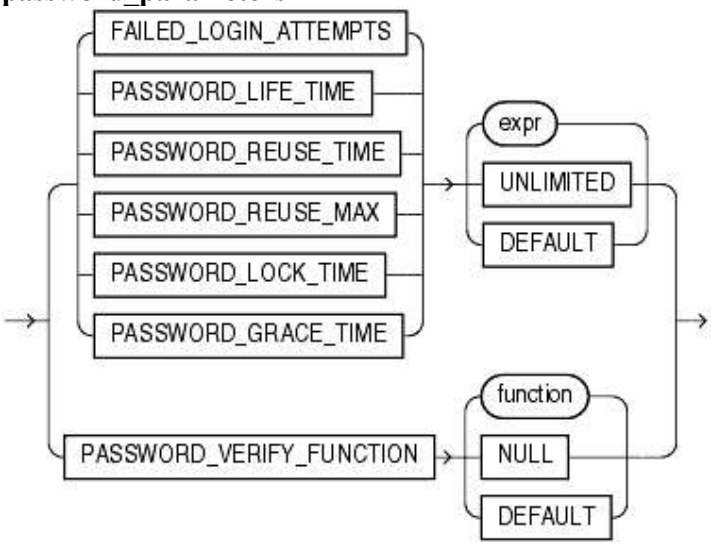
Format



resource_parameters



password_parameters



Contoh membuat profile:

```
SQL> CREATE PROFILE profileku LIMIT
   SESSIONS_PER_USER          UNLIMITED
   CPU_PER_SESSION            UNLIMITED
   CPU_PER_CALL               3000
   CONNECT_TIME               45
   LOGICAL_READS_PER_SESSION  DEFAULT
   LOGICAL_READS_PER_CALL     1000
```

```
   PRIVATE_SGA                    15K
   COMPOSITE_LIMIT                5000000;
```

Mengedit profile

Formatnya sama persis dengan CREATE profile, hanya ganti kata CREATE menjadi ALTER. Contoh:

```
SQL> ALTER PROFILE profileku LIMIT
   FAILED_LOGIN_ATTEMPTS       5
   PASSWORD_LIFE_TIME          60
   PASSWORD_REUSE_TIME         60
   PASSWORD_REUSE_MAX          5
   PASSWORD_LOCK_TIME          1/24
   PASSWORD_GRACE_TIME         10;
```

Assign profile ke user

Misalkan saya ingin melihat profile dari user TEST

```
SQL> select USERNAME,PROFILE from dba_users
where USERNAME='TEST';

USERNAME                       PROFILE
------------------------------ ----------
TEST                           DEFAULT
```

Saya akan mengubah profile user TEST dari DEFAULT ke PROFILEKU

```
SQL> alter user test profile PROFILEKU;
```

Menghapus

Untuk menghapus profile PROFILEKU, gunakan perintah berikut:

```
SQL> drop profile PROFILEKU;

ERROR at line 1:
ORA-02382: profile PROFILEKU has users assigned, cannot drop without
CASCADE
```

OO… Error. Profile PROFILEKU ada yang makai, yaitu user TEST. Kalau begitu tambahkan argument CASCADE:

```
SQL> drop profile PROFILEKU CASCADE;
```

Setelah profile dihapus, user yang di-assign ke profile tersebut akan dikembalikan ke profile DEFAULT.

Keterangan

Resource parameter:

- SESSIONS_PER_USER
  Jumlah session paling banyak yang bisa digunakan secara bersamaan (concurrent).
- CPU_PER_SESSION
  Maksimal CPU per session. Satuan: 10 ms (hundredth of seconds).
- CPU_PER_CALL
  Maksimal CPU per call (parse, execute, atau fetch). Satuan: 10 ms (hundredths of seconds).
- CONNECT_TIME
  Maksimal waktu untuk koneksi. Satuan: menit.
- IDLE_TIME
  Maksimal waktu dalam status INACTIVE (idle). Satuan: menit.
- LOGICAL_READS_PER_SESSION
  Maksimal block yang boleh dibaca per session, termasuk block dari memori dan disk.
- LOGICAL_READS_PER_CALL
  Maksimal block yang boleh dibaca per call (parse, execute, atau fetch).
- PRIVATE_SGA
  Maksimal memory untuk PRIVATE AREA. Hanya berlaku pada "shared server". Satuan: bytes.
- COMPOSITE_LIMIT
  Total resource cost per session. Satuan: **service**. Dihitung dari jumlah CPU_PER_SESSION, CONNECT_TIME, LOGICAL_READS_PER_SESSION, dan PRIVATE_SGA.

Password Parameter:

- FAILED_LOGIN_ATTEMPTS
  Jumlah maksimal berapa kali salah login (password). Setelah mencapai FAILED_LOGIN_ATTEMPTS, user yang bersangkutan akan di-lock.
- PASSWORD_LIFE_TIME
  Umur password, dalam satuan hari. Password harus diganti sebelum mencapai PASSWORD_LIFE_TIME.
- PASSWORD_GRACE_TIME (satuan: hari)
  Bila setelah mencapai PASSWORD_LIFE_TIME, namun user masih belum mengubah passwordnya, maka user akan diberi peringatan selama PASSWORD_GRACE_TIME. Bila telah mencapai PASSWORD_GRACE_TIME namun user masih belum mengubah passwordnya, maka password akan expired dan tidak bisa digunakan untuk login. Bila PASSWORD_GRACE_TIME tidak disetting, maka nilainya UNLIMITED, artinya user akan tetap bisa login meskipun telah mencapai PASSWORD_LIFE_TIME.
- PASSWORD_REUSE_TIME (satuan hari) dan PASSWORD_REUSE_MAX
  Kedua parameter ini dipakai bersamaan. PASSWORD_REUSE_TIME menyatakan berapa hari (sejak password diganti) password boleh dipakai lagi. PASSWORD_REUSE_MAX menyatakan berapa kali password harus berubah (berganti) sebelum memakai password lama yang dulu pernah dipakai itu.
- PASSWORD_LOCK_TIME (satuan hari)
  Menyatakan berapa hari user akan di-lock setelah gagal login.
- PASSWORD_VERIFY_FUNCTION

Refensi

- Oracle® Database Security Guide 10g Release 2 (10.2) Administering User Privileges, Roles, and Profiles
- Oracle® Database SQL Reference 10g Release 2 (10.2) CREATE PROFILE

4.2. Men-setting database menjadi archivelog mode

Di database Oracle, semua transaksi di-record (disimpan) di dalam log file. Dalam 1 instance, minimal ada 2 group logfile. Mekanisme kerjanya adalah sirkular. Bila logfile penuh, maka transaksi disimpan di log berikutnya. Setelah semua log terisi, maka log yang terlama akan ditulis ulang (rewrite), tentu saja dengan menghapus content (isi) sebelumnya. Tentu saja, kita akan kehilangan jejak transaksi yang ada di logfile tersebut.

Dalam database dengan mode archivelog, sebelum logfile ditulis ulang, content-nya dicopy (backup) dulu ke **archived log**. Oleh karena itu kita tidak kehilangan jejak transaksi yang disimpan di log yang ditulis ulang tersebut.

Archived log digunakan untuk recovery database. Bila kita me-restore dari hasil offline backup, maka data yang bisa diambil adalah data ketika off line backup dilakukan. Jadi, seandainya full backup dilakukan sebulan yang lalu, maka data yang bisa diselamatkan (diambil) adalah data sebulan yang lalu tersebut.

Berbeda dengan jika kita me-restore dari hasil online backup. Setelah file backup di-restore, kemudian archived log yang terbentuk setelah online backup (yang berisi rekaman transaksi itu) di-apply kembali (istilahnya recovery). Sehingga kita bisa mendapatkan data sampai archived log terakhir, atau sesaat sebelum terjadi bencana (kerusakan database).

Untuk melihat apakah database sudah dalam mode archivelog atau tidak

```
SQL> archive log list

Database log mode No Archive Mode
Automatic archival Disabled
Archive destination /oradata/oracle/ts/arc
Oldest online log sequence 56
Next log sequence to archive 58
Current log sequence 58
```

Dalam contoh di atas, mode database masih belum archivelog. Untuk mengaktifkan mode archivelog, jalankan command berikut:

```
SQL> shutdown immediate
SQL> startup mount
SQL> alter database archivelog;
SQL> alter database open;
```

Lihat, sekarang mode database sudah archivelog

```
SQL> archive log list

Database log mode Archive Mode
Automatic archival Enabled
Archive destination /oradata/oracle/ts/arc
Oldest online log sequence 56
Next log sequence to archive 58
Current log sequence 58
```

Catatan:
Perintah “alter database archivelog” adalah untuk membuat mode database menjadi ARCHIVELOG. Untuk meng-archive log file dilakukan dua cara:

1. Manual
2. otomatis

Pilihan manual adalah jarang terjadi, kecuali untuk tujuan tertentu, misalnya belajar. Semua database production selalu memilih yang otomatis.

Untuk mengotomatiskan pekerjaan archive, init parameter l**og_archive_start** harus TRUE. Jadi harus mengaktifkan parameter tersebut di file init.

Saya menggunakan database 10g release 2 (10.2.0). Parameter log_archive_start tidak perlu saya setting, alias bernilai false, tapi archive jalan otomatis. Kayaknya di versi 10g parameter ini dah gak dibutuhkan lagi, saya belum explore lebih jauh.

Yang pasti, database 10g saya archive-nya jalan otomatis tanpa mensetting parameter log_archive_start.

4.3. Step-step Mencopy Database ke Mesin yang Sama

Mengcopy database di mesin lain, itu sama saja dengan backup & recovery konvensional. Mengcopy database di mesin yang sama, ada sedikit bedanya karena di satu mesin tidak boleh ada 2 (atau lebih) database yang sama; ada step untuk mengubah nama database.

Saya punya 1 database development, nama databasenya adalah **ts2**. Untuk keperluan test, saya butuh database lagi di mesin development tersebut. Content database baru adalah sama dengan database terdahulu. Bagaimana cara membuatnya, ada dua cara:

1. Membuat database baru, export dari database lama, kemudian import ke database baru.
2. Mengcopy database lama ke yang baru, seperti kalau backup dan recovery

Cara pertama tentu saja memakan waktu yang cukup lama, apalagi kalau datanya cukup besar. Kalau begitu saya pilih cara kedua; saya backup database pertama, kemudian saya naikkan (restore & recovery) dengan nama yang berbeda (saya beri nama **tsrep**). Berikut ini langkah-langkahnya:

1. Persiapkan (copy) init file. Contoh di sini, saya menggunakan UNIX. Di Windows, lokasi init file di %ORACLE_HOME%\database

   ```
   cd $ORACLE_HOME/dbs
   cp -rp initts2.ora inittsrep.ora
   ```

   Misalkan content inittsrep.ora adalah berikut ini:

   ```
   ### Parameter ini perlu disesuaikan ###
   audit_file_dest='/data1/oracle/admin/ts2/adump'
   background_dump_dest='/data1/oracle/admin/ts2/bdump'
   core_dump_dest='/data1/oracle/admin/ts2/cdump'
   user_dump_dest='/data1/oracle/admin/ts2/udump'
   control_files='/oradata/oracle/ts2/control01.ctl',
   '/oradata/oracle/ts2/control02.ctl',
   '/oradata/oracle/ts2/control03.ctl'
   log_archive_dest_1='LOCATION=/oradata/oracle/ts2/arc'
   db_name='ts2'
   ###################################
   log_archive_dest_state_1=enable
   log_archive_format=%s_%t_%r.arc
   compatible='10.2.0.3.0'
   db_block_size=8192
   db_domain=''
   pga_aggregate_target=209715200
   sga_target=1610612736
   undo_management='AUTO'
   undo_tablespace='UNDOTBS1'
   remote_login_passwordfile='EXCLUSIVE'
   ```

   Beberapa parameter saya sesuaikan, terutama **db_name** dan yang berkaitan dengan direktori. Berikut ini parameter yang telah saya sesuaikan:

   ```
   audit_file_dest='/data1/oracle/admin/tsrep/adump'
   background_dump_dest='/data1/oracle/admin/tsrep/bdump'
   core_dump_dest='/data1/oracle/admin/tsrep/cdump'
   user_dump_dest='/data1/oracle/admin/tsrep/udump'
   ```

```
control_files='/oradata/oracle/tsrep/control01.ctl',
'/oradata/oracle/tsrep/control02.ctl',
'/oradata/oracle/tsrep/control03.ctl'
log_archive_dest_1='LOCATION=/oradata/oracle/tsrep/arc'
db_name='tsrep'
```

2. Persiapkan direktori untuk data file dan lainnya
```
mkdir /data1/oracle/admin/tsrep
mkdir /data1/oracle/admin/tsrep/adump
mkdir /data1/oracle/admin/tsrep/bdump
mkdir /data1/oracle/admin/tsrep/cdump
mkdir /data1/oracle/admin/tsrep/udump
mkdir /oradata/oracle/tsrep
mkdir /oradata/oracle/tsrep/arc
```

3. Persiapkan script untuk create database
   Di database lama (source):
```
SQL> alter database backup controlfile to trace resetlogs;
```

   Lihat trace file di direktory user_dump_dest, /data1/oracle/admin/ts2/udump
```
SQL> sho parameter user_dump_dest
```

   Copy trace file tersebut
```
cd /data1/oracle/admin/ts2/udump
cp ts2_ora_18762.trc /oradata/oracle/tsrep/crdbtsrep.sql
```

   Edit file **crdbtsrep.sql**, buang “trace file entry”, hingga jadi seperti ini
```
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "TS2" RESETLOGS ARCHIVELOG
MAXLOGFILES 16
MAXLOGMEMBERS 3
MAXDATAFILES 100
MAXINSTANCES 8
MAXLOGHISTORY 292
LOGFILE
GROUP 4 '/oradata/oracle/ts2/redo04.log' SIZE 5M,
GROUP 5 '/oradata/oracle/ts2/redo05.log' SIZE 5M,
GROUP 6 '/oradata/oracle/ts2/redo06.log' SIZE 5M
-- STANDBY LOGFILE
DATAFILE
'/oradata/oracle/ts2/system01.dbf',
'/oradata/oracle/ts2/undotbs01.dbf',
'/oradata/oracle/ts2/sysaux01.dbf',
'/oradata/oracle/ts2/users01.dbf'
CHARACTER SET WE8ISO8859P1
;
```

   Edit lagi file **crdbtsrep.sql**.
   Ganti entry ini
```
CREATE CONTROLFILE REUSE DATABASE “TS2” RESETLOGS ARCHIVELOG
```
   Menjadi
```
CREATE CONTROLFILE SET DATABASE “TS2REP” RESETLOGS ARCHIVELOG
```

   Sesuaikan directory-directory yang berkaitan.
   Dalam contoh ini saya ganti
```
/oradata/oracle/ts2/
```
   Menjadi
```
/oradata/oracle/tsrep/
```

Setelah beberapa pengeditan, file file crdbtsrep.sql menjadi

```
STARTUP NOMOUNT
CREATE CONTROLFILE SET DATABASE "TSREP" RESETLOGS ARCHIVELOG
MAXLOGFILES 16
MAXLOGMEMBERS 3
MAXDATAFILES 100
MAXINSTANCES 8
MAXLOGHISTORY 292
LOGFILE
GROUP 4 '/oradata/oracle/tsrep/redo04.log' SIZE 5M,
GROUP 5 '/oradata/oracle/tsrep/redo05.log' SIZE 5M,
GROUP 6 '/oradata/oracle/tsrep/redo06.log' SIZE 5M
-- STANDBY LOGFILE
DATAFILE
'/oradata/oracle/tsrep/system01.dbf',
'/oradata/oracle/tsrep/undotbs01.dbf',
'/oradata/oracle/tsrep/sysaux01.dbf',
'/oradata/oracle/tsrep/users01.dbf'
CHARACTER SET WE8ISO8859P1
;
```

4. Copy (backup dan restore) database source
   Kalau databasenya NOARCHIVELOG, lakukan cold (off line) backup. Karena database **ts2** ARCHIVELOG, maka saya bisa melakukan secara hot (on line) backup. Dalam contoh ini saya memakai hotbackup.

   Lihat list datafile dan tempfile yang perlu di-copy

```
SQL> select name as file_name from
(select name from v$tempfile union
select name from v$datafile);

/oradata/oracle/ts2/sysaux01.dbf
/oradata/oracle/ts2/system01.dbf
/oradata/oracle/ts2/temp01.dbf
/oradata/oracle/ts2/undotbs01.dbf
/oradata/oracle/ts2/users01.dbf
```

   Lihat tablespace yang perlu di backup

```
SQL> select distinct tablespace_name from dba_data_files;

TABLESPACE_NAME
------------------------------
SYSTEM
USERS
SYSAUX
UNDOTBS1
```

   Sebelum menjalankan BEGIN BACKUP, lihat “Current log sequence”. Informasi ini kita perlukan untuk melihat archived log mana saja yang nanti dibutuhkan untuk menaikkan database. Dalam contoh ini, “Current log sequence” adalah 17.

```
SQL> archive log list
```

   Jalankan BEGIN BACKUP pada tablespace-tablespace tersebut (Kalau cold backup, step ini diganti dengan shutdown database):

```
SQL> alter tablespace SYSTEM begin backup;
SQL> alter tablespace USERS begin backup;
```

```
SQL> alter tablespace SYSAUX begin backup;
SQL> alter tablespace UNDOTBS1 begin backup;
```

Copy datafile dan tempfile

```
cd /oradata/oracle/ts2/
cp -rp sysaux01.dbf /oradata/oracle/tsrep
cp -rp system01.dbf /oradata/oracle/tsrep
cp -rp temp01.dbf /oradata/oracle/tsrep
cp -rp undotbs01.dbf /oradata/oracle/tsrep
cp -rp users01.dbf /oradata/oracle/tsrep
```

Setelah proses copy datafile selesai, jalankan END BACKUP pada tablespace-tablespace terkait (Kalau cold backup, step ini diganti dengan startup database):

```
SQL> alter tablespace SYSTEM end backup;
SQL> alter tablespace USERS end backup;
SQL> alter tablespace SYSAUX end backup;
SQL> alter tablespace UNDOTBS1 end backup;
```

Setelah selesai END BACKUP, lihat nilai dari "Current log sequence". Dalam contoh ini nilainya adalah 18.

```
SQL> archive log list
```

Jalankan "archive log current" untuk membuat archived log dari sequence 18.

```
SQL> alter system archive log current
```

Archived log yang dibutuhkan untuk menaikkan database adalah archived log sequence sebelum BEGIN BACKUP dan seteleh END BACKUP. Jadi, sequence 17 dan 18.

Selanjutnya copy archived log yang dibutuhkan itu

```
cd /oradata/oracle/ts2/arc/
cp -rp 17_1_658171224.arc /oradata/oracle/tsrep/arc
cp -rp 18_1_658171224.arc /oradata/oracle/tsrep/arc
```

5. Persiapan sebelum menaikkan database
   Setting ORACLE_SID, di shell csh

   ```
   setenv ORACLE_SID tsrep
   ```

   Di shell ksh

   ```
   export ORACLE_SID=tsrep
   ```

   Di Windows

   ```
   set ORACLE_SID=tsrep
   ```

   Khusus di Windows, buat instance (service) pakai **oradim**. Pastikan bahwa file **inittsrep.ora** sudah dibuat.

   ```
   cd %ORACLE_HOME%\database
   oradim -NEW -SID tsrep
   ```

6. Menaikkan (create) database

   ```
   SQL> @/oradata/oracle/tsrep/crdbtsrep.sql
   ```

   Lakukan recovery.
   Dengan specify automatic, Oracle akan mencari archived log file sendiri.

   ```
   SQL> RECOVER AUTOMATIC DATABASE USING BACKUP CONTROLFILE;
   ```

   Dalam contoh ini, archived log yang dibutuhkan adalah sequence 17 sampai 18. Setelah sequence 18 di-apply, Oracle masih minta squence 19. Karena sequence 19

tidak ada, maka recovery menjadi error. Abaikan error tersebut. Selanjutnya:

```
SQL> RECOVER DATABASE until cancel;
```

Kalau database sinkron, seharusnya command di atas berhasil. Kalau Oracle masih minta archived log untuk recovery, copy saja lagi dari database source-nya. Kalau di database source archived log belum ter-create, ya jalankan "alter system archive log current", atau kalau mau, langsung saja recover lagi menggunakan logfile database source (bukan archived log-nya).Misalkan masih dibutuhkan archived log sequence 19, sementara sequence tersebut belum dicreate archived lognya. Gunakan command berikut untuk mendapat logfilenya

```
SQL> -- di database source (ts2)
SQL> select member from v$logfile where GROUP# in
(select GROUP# from v$log where SEQUENCE#=19);
```

Selanjutnya, gunakan logfile di atas untuk recovery di database **tsrep.**
Setelah proses recovery berhasil, open database:

```
SQL> ALTER DATABASE OPEN RESETLOGS;
```

Selanjutnya, tambahkan temp file:

```
SQL> ALTER TABLESPACE TEMP ADD
TEMPFILE '/oradata/oracle/tsrep/temp01.dbf'
SIZE 20971520 REUSE AUTOEXTEND OFF;
```

4.4. Memindahkan Database ke Mesin lain

Pertanyaan
Hallo Mas.., bacaan di blog nya mas sangat berguna bagi saya sebagai pemula di Oracle. Saya mau tanya sedikit masalah database Oracle. Misalnya di PC server yang saya gunakan mengalami crash.. dan tidak bisa hidup lagi. Apakah bisa hardisk yg isinya database Oracle dipindahkan ke PC yg lain dan di buka dengan oracle di PC yang baru ini? Dan diharapkan datanya utuh… atau ada solusi lain untuk masalah ini? Mohon petunjuk dari mas.

Jawaban
Kalau data-data di harddisk (terutama bagian yang memuat file-file database) masih bisa dibaca, tentu saja database anda bisa dinaikkan di mesin (komputer/PC) lain. Anggap saja data di hard disk anda sebagai backup, selanjutnya backup data ini akan anda naikkan di tempat lain. Dengan kata lain, anda akan melakukan **Restore dari off line backup**

Langkah-langkahnya adalah sebagai berikut (misalkan nama instance/database anda **DATAKU**):

1. Install database Oracle (versi harus sama dengan sebelumnya) di PC baru. Lokasi ORACLE_HOME tidak harus sama.
2. Copy initial (parameter) file dari disk lama ke direktory **%ORACLE_HOME%\database** (kalau PC anda Windows) atau **$ORACLE_HOME/dbs** (kalau PC anda Unix, misal Linux). Misalkan initial (parameter) file itu adalah **initDATAKU.ora**
3. Kalau PC anda Windows, buat instance
```
cd %ORACLE_HOME%\database
set ORACLE_SID= DATAKU
orapwd file=orapwDATAKU password=oracle entries=10 force=y
oradim -NEW -SID DATAKU
```
4. Lakukan langkah-langkah restore seperti yang saya bahas di Restore dari off line backup

Saya berharap semoga ini bisa membantu, dan database anda up kembali di PC yang baru. Good luck.

4.5. Tentang High Water Mark

Beberapa waktu yang lalu saya membahas tentang bagimana cara mengurangi size dari datafile (tablespace). Datafile tidak bisa dikurangi sampai batas High Water Mark (HWM).

Misalkan kita punya tablespace USERS yang punya datafile 3G. Namun bila kita lihat di OEM, TOAD, atau melalui view dba_free_space, ternyata datafile tersebut mempunyai free space cukup banyak yaitu 2G, itu artinya yang kepakai hanya 1G. Namun datafile tersebut tidak bisa diresize menjadi 1G, bahkan 2G saja tidak bisa.

```
SQL> alter database
datafile '/oradata/oracle/ts/users01.dbf' resize 2048M;

ORA-03297: file contains used data beyond requested RESIZE value
```

Dulunya, tablespace USERS mungkin (pasti) pernah kepakai sampai 3G. Bisa kepakai oleh table, index, ataupun temp segment. Namun sekarang sudah banyak yang dihapus sehingga yang terpakai hanya 1G. Nah ini yang penting, DULUNYA pernah dipakai. Oracle membuat aturan bahwa datafile tidak bisa di-resize menjadi ukuran maksimal yang DULUNYA pernah dipakai. Ukuran maksimal yang DULUNYA pernah dipakai ini disebut sebagai HIGH WATER MARK (HWM).

Begini analoginya. Misalkan di kampung kita ada tanggul kali yang tingginya 4 meter. Sejak adanya tanggul itu, paling tinggi air mencapai 3,5 meter (inilah yang disebut sebagai HIGH WATER MARK, batas air paling tinggi yang pernah dicapai), setengah meter lagi mencapai batas atas tanggul.

Namun sejak 10 tahun terakhir ini air paling tinggi hanya mencapai 2 meter. Karena itulah warga sekitar menyarankan agar tanggul diturunkan menjadi 2 meter saja. Tanggul yang tinggi 4 meter itu bikin pemandangan tidak sedap. Tapi… oleh pemerintah daerah tanggul tidak boleh diturunkan sampai 3,5 meter (apalagi di bawah itu). Kalau mau menurunkan ya paling tidak sampai 3,6 meter lah. Katanya, ‘pamali’ kalau menurunkan tanggul di bawah ketinggian air maksimal yang pernah dicapai.

Kira-kira begitulah pengertian HIGH WATER MARK.

Mari kita lihat lebih dalam. Datafile berisi block-block data. Bayangkan dinding yang terbuat dari tumpukan 100 batu bata. Pada awalnya block-block ini disusun secara berurutan. Pada suatu hari kita menghapus data (delete), analoginya kita mau mengambil 5 batu bata. OO… ternyata data (batu bata) yang kita cari itu ada di tumpukan paling bawah. Berikutnya kita ingin menghapus data lagi (sebanyak 4 batu bata). Dan sekarang batu bata yang ini ada di tumpukan tengah. Demikianlah seterusnya hingga akhirnya yang tersisa ada 40 batu bata.

Sekarang, tumpukan 40 batu bata itu tidak beraturan dan memamakan ruang sebesar tumpukan 100 batu bata. Namun kita tidak bisa ngapa-ngapain karena sudah begitu adanya. Kalo diutak-atik malah bisa rubuh tumpukan batu bata itu. Jadi space sebesar tumpukan 100 batu bata itulah yang disebut sebagai HIGH WATER MARK, batas tertinggi yang pernah dipakai.

Gimana cara memaksa untuk menurunkan size datafile sampai di bawah HIGH WATER MARK?

Caranya, rubuhkan dan susun ulang tumpukan batu bata itu. Di database, rubuhkan dan susun ulang block-block data itu. Begini caranya:

1. Export content (data-data) tablespace yang bersangkutan
2. Buat tablespace baru yang punya size lebih kecil
3. Import data yang telah di-export tersebut ke tablespace baru
4. Drop tablespace lama

Namun untuk tablespace SYSTEM kita tidak bisa melakukan hal di atas. Satu-satunya cara bagi tablespace SYSTEM adalah:

1. Export full database
2. Buat database baru yang ukuran tablespace SYSTEM-nya lebih kecil
3. Import full data yang telah di-export tersebut
4. Drop database lama

4.6. Men-setting Character Set

Kadang kita bingung kenapa tidak bisa melakukan insert karakter simbol (misalnya Ø) ke database Oracle.

```
SQL> create table test (a varchar2(10));
SQL> insert into test values ('Ø');
SQL> select * from test;

A
----------
?
```

Lho kok, hasilnya **?** bukannya **Ø**.
Simbol Ø adalah karakter no 216. Coba kita query ke database:

```
SQL> select chr(216) from dual;

C
-
?
```

OO.. ternyata masih tidak terbaca. Selidik punya selidik, ternyata ini berkaitan dengan CHARACTER SET di database dan di client di mana kita melakukan query. Untuk bisa melakukan hal itu, database Oracle dan client harus punya CHARACTER SET yang sesuai.

Contoh character set yang mengakomodasi simbol Ø adalah WE8MSWIN1252 dan WE8ISO8859P1, lebih detail silahkan lihat referensi. WE8MSWIN1252 adalah superset dari WE8ISO8859P1, dengan arti lain WE8MSWIN1252 berisi semua karaketer WE8ISO8859P1 dan tambahan lainnya. Kalau saat ini CHARACTER SET database tidak mencakup simbol Ø, sebaiknya gunakan WE8MSWIN1252 daripada WE8ISO8859P1.

Mari kita check apakah database kita sudah menggunakan CHARACTER SET yang sesuai

```
SQL> col PROPERTY_VALUE for a20
SQL> col PROPERTY_NAME for a20
SQL> select PROPERTY_NAME,PROPERTY_VALUE
from database_properties
where PROPERTY_NAME= 'NLS_CHARACTERSET';

PROPERTY_NAME        PROPERTY_VALUE
-------------------- --------------------
NLS_CHARACTERSET     WE8MSWIN1252
```

Database sudah menggunakan CHARACTER SET yang sesuai. Tapi tidak bisa menampilkan symbol Ø. Nah kalau gini pasti ada masalah dengan setting di client.

Di UNIX, lihat value dari environment (parameter) NLS_LANG.

```
echo $NLS_LANG
```

Yup, ternyata NLS_LANG tidak disetting. Sekarang, setting NLS_LANG

```
NLS_LANG=AMERICAN_AMERICA.WE8MSWIN1252; export NLS_LANG
```

Sekarang coba jalankan perintah SQLPLUS berikut

```
SQL> delete from test;
SQL> insert into test values('Ø');
SQL> insert into test values( chr(216) );
SQL> select * from test;

A
----------
Ø
Ø
```

OO… berhasil Coba kalau NLS_LANG di client kita ganti WE8ISO8859P1, apakah masih bisa:

```
NLS_LANG=AMERICAN_AMERICA.WE8ISO8859P1; export NLS_LANG

SQL> select * from test;

A
----------
Ø
Ø
```

OO… Ternyata masih bisa juga.

MENGUBAH CHARACTER SET DI DATABASE

Kalau CHARACTER SET di database tidak men-support karakter yang kita inginkan, berikut ini cara mengubah CHARACTER SET (misalkan kita ingin mengubahnya menjadi WE8MSWIN1252) :

```
SQL> shutdown immediate
SQL> startup mount
SQL> ALTER SYSTEM ENABLE RESTRICTED SESSION;
SQL> ALTER DATABASE CHARACTER SET WE8MSWIN1252;
```

Setelah itu, restart database

```
SQL> shutdown immediate
SQL> startup
```

Referensi:

Oracle® Database Globalization Support Guide 10g Release 2 (10.2)
Choosing a Character Set
Setting Up a Globalization Support Environment

4.7. Diskusi tentang Load balance di RAC

Ada diskusi saya dengan teman-teman di milis indo-oracle tentang RAC. Tampaknya menarik untuk saya taruh di blog ini. Ini tentang load balance.

Sharing teman I

I have rac environment: hp-ux itanium, 2 node rac, db 10.2.0.2, hp-serviceguard

Now we got problem may be bug… From the OS side it seem balance, that the currently processes running is same (throughtop/glance). The memory utilization is almost balance as well as cpu

But from inside oracle node2 has 70% from total sessions. I have configured server-side and client side load balancing altogether. clb_goal set to short, throughput in the service

From SR support told me that querying V$osstat on this platform return no rows, that may be the root cause.

Tanggapan saya

Memory dan CPU sudah balance, itu berarti "load balance" sudah tercapai. CMIIW, term "load balance" itu menitik beratkan pada load system secara keseluruhan, bukan semata-mata jumlah session di masing-masing instance.

Kadang (bahkan sering) jumlah session tidak berbanding lurus dengan load (pemakaian resource). Load untuk 1 session transaksi insert 1 row BERBEDA dengan load untuk 1 session yang melakukan deleting 1m rows.

Tanggapan teman I

Anehnya kalau dilihat dari dalam oracle gv$servicemetric, jelas2 gak balance. Kalau dari sisi OS balance hanya terlihat "process running saja" masing2 node kelahatan process running (dgn top) memang balance, tapi dari sisi utilisasi cpu, memory, disk beda banget..

so ada yg pernah mengalami? and how to solve?

Kemarin sempat balance karena saya coba re-register PMON terhadap service yg sedang running dgn merubah parameter local_listener, cuman ya itu dia kadang2 rada angot, load balance nya jadi dodol
lagi….tapi besoknya jadi balance lagi….. he…he…. seperti teka-teki silang
Tanggapan saya

He… he… kayak teka-teki silang ya. Makanya Pak, Oracle dibilang "Ora kelar-kelar"
Dulu saya pernah ngalamin yang seperti itu di versi 9i. Analisa dari Oracle Support (TAR) terlalu lama, biasanya paling-paling apply patch atau apply latest patch set. Akhirnya kita putusin gak pake RAC lagi.

Dulu juga pernah ngalamin banyak problem di RAC 10.2.0.2, terutama dengan tingginya wait-wait yang berkaitan dengan cluster. Akhirnya kita putusin pake single node saja. Viola… response time jadi cepet banget.

High availability yang ditawarkan Oracle harus kita bayar dengan turunnya response time.

Tentu saja, tidak semua solusi dari Oracle cocok dengan environment kita. Dan tentu saja juga, banyak yang terbantu dengan solusinya Oracle.

Ada situasi, kondisi, dan NASIB yang menentukan. Yang terutama, NASIB

Tanggapan teman II

Dear Pak Rohmad. Saya sangat tidak sependapat dengan statement anda dan menurut saya itu adalah pendapat dari sebuah kegagalan anda. Apa pendapat anda dari Capture database RAC saya di attament email ini…? Dan Saya sangat merasakan sekali benefit dari Oracle RAC ini, benar-benar ZERO Downtime. Sehingga Aplikasi ERP kita tidak pernah terganggu (bisnis proses tidak terganggu), sekalipun kita melakukan Restart salah satu Database Server kita maupun kita sedang melakukan maintenance server kita. Btw, saya juga tidak pernah mengalami problem seperti keluhan anda di Oracle RAC saya.

Tanggapan saya

Iya, dari sisi Oracle Corp saya mungkin orang yang gagal mengemban misi Oracle untuk menjadikan Oracle sebagai main solution. Namun dari sisi company saya, paling tidak saya bisa membantu menunjukkan product-product Oracle yang mana yang cocok dengan environment yang ada.

Saat ini saya memantain 4 RAC, masing-masing dua node. Dulunya lebih banyak lagi. Kita pernah menambah salah satu RAC menjadi 3 node; namun ya itu, keinginan tidak sesuai dengan kenyataan, akhirnya kita kembalikan lagi ke 2 node. Teori tidak selamanya memberi kenyataan yang perfect. Bug-bug sering muncul ketika resource utilization meningkat drastis. Bug, adalah bukti nyata tidak mungkin sempurnanya suatu teori (konsep).

RAC, untuk tujuan high availability, menambah layer baru di bawah database, yaitu cluster. Cluster inipun punya layer lagi, yaitu software clusternya Oracle (CRS di 10g) dan cluster di level machine.

Penambahan layer adalah penambahan possibility untuk mendapat masalah tambahan.

Ketika ada masalah, atau bug muncul, Oracle belum tentu mendapat solusi dengan cepat. Seperti yang pernah saya alami. Karena masalahnya unik, belum pernah ada, maka team Oracle support melempar masalah saya ke team development Oracle. Lhah… berapa lama saya mesti menunggu, sementara aplikasi tidak boleh lama-lama dalam keadaan bad-performance? Bad performance berarti REVENUE LOSS. Yah… tampaknya keputusan yang terbaik adalah melepaskan RAC. Setelah performance bagus; kita tetep mesti mencari solusi high availability, menunggu Oracle mendapat solusi atau memakai solusi lain selain Oracle.

Btw, syukur, RAC anda tidak mendapat masalah. Sayapun bersyukur, 4 RAC saya yang saat ini running tidak bermasalah.

5. Audit

5.1. Audit operasi di suatu table

Misalkan kita ingin tahu user-user mana saja yang melakukan perubahan (INSERT, UPDATE, DELETE) pada suatu tabel. Caranya, aktifkan parameter AUDIT_TRAIL, lakukan perintah audit, dan lihat hasil auditnya.

1. Aktifkan parameter AUDIT_TRAIL. Ada tiga pilihan value, yaitu DB,OS, dan NONE. By default nilainya adalah NONE. Pilih OS kalau ingin hasil audit disimpan di suatu file, dan pilih DB kalau ingin hasil audit disimpan di tabel (database). Dalam contoh ini saya memilih option DB. Alasannya: lebih mudah melihat (me-manage) hasil auditnya.
```
SQL> alter system set AUDIT_TRAIL=DB scope=spfile;
SQL> shutdown immediate
SQL> startup
```

2. Lakukan perintah audit. Misalkan kita ingin mengaudit operasi INSERT, UPDATE, DELETE pada tabel pegawai (di schema test)
```
SQL> AUDIT INSERT, UPDATE, DELETE ON test.pegawai
BY ACCESS WHENEVER SUCCESSFUL;
```

3. Lihat hasilnya
```
SQL> select * FROM SYS.AUD$;
```

4. Untuk meng-cancel audit
```
SQL> NOAUDIT INSERT, UPDATE, DELETE ON test.pegawai;
```

Pertanyaan
Saya mau nanya nih soal audit trail, kira2 apa pro dan kon kalau tabel aud$ dipindah ke tablepsace yang lain? Dari artikel, sebagian DBA menyarankan agar dipindah, tetapi dari note Oracle sendiri, kalau dipindah dan suatu saat ada masalah, oracle tidak mau mensupport. Jadi gimana baiknya ya?

Jawaban
By default, tabel aud$ disimpan di tablespace SYSTEM. Dengan pertimbangan maintenance, kita bisa memindahkannya ke tablespace lain. Walaupun not supported, bahkan Oracle sendiri ngasih tahu (bikin step-step) cara untuk memindahkan tabel aud$ ke tablespace lain.

Adapun maksud dari “not supported” adalah Oracle tidak mensupport kalau-kalau nanti ada masalah di database yang disebabkan oleh pemindahan tabel aud$ ini.

Lantas, gimana baiknya? Salah satu pertimbangan utama memindahkan tabel aud$ ke tablespace lain adalah untuk mencegah agar tablespace SYSTEM tidak tumbuh (growth) cepat/besar hanya gara-gara tabel aud$ ini. Namun kita bisa mensiasatinya kok, kita bisa melakukan cleanup/purge terhadap tabel aud$ tersebut:

1. Content dari tabel aud$ dicopy ke table lain, misal aud_history (yang ditaruh di tablespace lain)

2. Setelah dicopy, record di tabel aud$ bisa di-delete

Catatan
Berikut ini Quote dari Document Oracle Metalink Note:72460.1

```
Important Note:
===============

It is strongly recommended to use the
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION
provided with the DBMS_MGMT package instead of using the method described
in this document, see Note 731908.1 "New Feature DBMS_AUDIT_MGMT To Manage
And Purge Audit Information" for more information and references. Also
please understand the movement of the audit tables SYS.AUD$ (or SYSTEM.AUD$
in case of OLS) and FGA_LOG$ tables is supported by the DBMS_AUDIT_MGMT
package. But, this gives no support on adding triggers to AUD$ table.

Moreover many additional audit features such as Alerts are possible with
Audit Vault and should not need to be implemented on the source database
with custom triggers but by using a supported product feature.

The procedure described in this note is not officially supported, because
the Oracle code makes implicit assumptions about the data dictionary tables
such as SYS.AUD$, which could cause problems with upgrades and backup /
recovery scenarios. However, in most cases the procedure works as
described. If you encounter problems using a trigger on the relocated AUD$
table Oracle Support may suggest to delete it.

In case you decide to move AUD$ to a different tablespace, and the
tablespace or datafile where you relocated is unavailable, you will not be
able to use audit anymore, therefore your applications might get errors
such as ORA-9925 or ORA-9817 In such cases, please correct the problem
before calling support.

For more information on this issue, please read Oracle 8i Administration
Guide, Chapter 24 - "Auditing Database Use", under topic "Controlling the
growth and size of the audit trail":

The maximum size of the database audit trail (SYS.AUD$ table) is
predetermined during database creation. By default, up to 99 extents, each
10K in size, can be allocated for this table. You should not move SYS.AUD$
to another tablespace as a means of controlling the growth and size of the
audit trail. However, you can modify the default storage parameters in
SYS.AUD$.
```

Lebih banyak lagi tentang feature-feature audit, lihat di referensi berikut:
http://download.oracle.com/docs/cd/B10501_01/server.920/a96521/audit.htm

5.2. Audit Update Table dengan Trigger

Oracle menyediakan tool untuk mengaudit suatu tabel dengan AUDIT_TRAIL, saya pernah membahasnya di "Audit operasi di suatu table". Dari AUDIT_TRAIL kita bisa mengetahui update table ini dilakukan kapan, oleh user siapa, dari mesin mana, dan lain-lain.

Kadang AUDIT_TRAIL tidak cukup. Kita mungkin ingin menyimpan data lama sebelum di-update sehingga bisa mengetahui history perubahan data. Jenis audit yang seperti ini disebut sebagai Value-Based Auditing (VBA). Nah, untuk ini kita bisa menggunakan TRIGGER.

Misalkan kita punya tabel EMP2

```
CREATE TABLE EMP2
(
EMPNO      NUMBER(4),
ENAME      VARCHAR2(20),
JOB        VARCHAR2(10),
SAL        NUMBER
);

insert into EMP2 values (1, 'Rohmad', 'DBA', 400000);
commit;
```

Kita akan membuat TRIGGER yang mencatat waktu dan nilai lama sebelum di-update. Langkah pertama, buat tabel yang menampung hasil audit tersebut, misalkan kita beri nama EMP2_AUDIT.

```
CREATE TABLE EMP2_audit
(
Tanggal_edit date,
EMPNO      NUMBER(4),
ENAME      VARCHAR2(20),
JOB        VARCHAR2(10),
SAL        NUMBER
);
```

Selanjutnya buat trigger yang mencatat nilai sebelum di-update

```
CREATE OR REPLACE TRIGGER trg_EMP2_update
BEFORE UPDATE ON EMP2
FOR EACH ROW
DECLARE
BEGIN
insert into EMP2_audit values (
sysdate,
:old.EMPNO,
:old.ENAME,
:old.JOB,
:old.SAL
);
END;
/
```

Sekarang, mari kita coba update tabel EMP2 itu

```
update EMP2 set sal=1000000 where empno=1;
commit;
```

OK. Tabel sudah di-update. Sekarang kita lihat hasil auditnya:

```
SQL> select * from EMP2_AUDIT;

TANGGAL_E       EMPNO ENAME           JOB                 SAL
--------- ---------- --------------- ---------- ----------
06-AUG-08           1 Rohmad          DBA             400000
```

Referensi

Oracle® Database Application Developer's Guide - Fundamentals - 10g Release 2 (10.2)

5.3. Trigger: Mencatat History Startup & shutdown DB

Kita bisa mengetahui kapan database shutdown dan kapan database startup, biasanya dengan melihat di alert log. Lebih detail tentang alert log saya bahas di Maintenance Log dan Trace File. Namun kita perlu usaha ekstra untuk membaca alert log. Untuk mempermudahnya, kita bisa membuat trigger yang mencatat setiap aktivitas (startup dan shutdown) ke tabel history.

Catatan: Untuk melakukan ini, gunakanan user yang punya privilege DBA (misalnya SYS atau SYSTEM).

Langkah pertama, siapkan tabel untuk mencatat history startup dan shutdown.

```
create table tbl_database_activity
(waktu          DATE,keterangan varchar2(8))
tablespace users;
```

Selanjutnya buat Trigger yang dijalankan sebelum database shutdown. Penting: Shutdown abort tidak akan dicatat oleh trigger ini.

```
CREATE OR REPLACE TRIGGER proc_database_shutdown
BEFORE SHUTDOWN ON DATABASE
BEGIN
insert into tbl_database_activity
values(sysdate,'SHUTDOWN');
END;
/
```

Yang berikut ini trigger sesudah database startup

```
CREATE OR REPLACE TRIGGER proc_database_startup
AFTER STARTUP ON DATABASE
BEGIN
insert into tbl_database_activity
values(sysdate,'STARTUP');
END;
/
```

Untuk melihat history startup dan shutdown, tinggal query ke tabel tbl_database_activity

```
SQL> select to_char(waktu,'dd-Mon-yy hh24:mi:ss'),
keterangan from system.tbl_database_activity
order by waktu;

TO_CHAR(WAKTU,'DD- KETERANG
------------------ --------
07-Aug-08 09:43:38 SHUTDOWN
07-Aug-08 09:44:44 STARTUP
07-Aug-08 10:05:24 STARTUP
```

Dalam contoh di atas, lihat hasil di baris ketiga, kok STARTUP padahal sebelumnya sudah STARTUP. Itu artinya, pastinya sebelum ini dilakukan shutdown abort atau database mati karena accident (misal listrik mati atau hardware rusak).

5.4. Memanage History Perkembangan Data

Database Administrator (DBA) HARUS aware (tahu) berapa perkembangan data (database) yang tengah dimaintainnya. Ini penting sekali untuk:

- Dari sisi aplikasi (user), memprediksi jumlah data pada waktu mendatang. Apakah satu, dua, atau sepuluh tahun ke depan aplikasi masih mampu mensupport proses data sebesar itu?
- Dari sisi infrastruktur, memprediksi kebutuhan storage
- Bagi manajemen, tentu saja untuk memprediksi budget buat pengadaan storage

Dari view yang by default disediakan Oracle, kita bisa mengetahui pemakaian space/disk/storage. Namun view tersebut hanya untuk mengetahui kondisi saat ini, tidak mencatat history pemakaian waktu-waktu yang lalu.

Filosofinya adalah, setiap hari kita mengcapture pemakaian space saat ini. Nah dengan mencatat pemakaian space tiap hari, akhirnya kita khan bisa melihat perkembangan datanya.

Melihat pemakaian space saat ini (current space utilization)

- Untuk tablespace dengan content PERMANENT dan UNDO, view-view yang di-query adalah:
    1. DBA_TABLESPACES. Untuk mengambil informasi TABLESPACE_NAME dan CONTENTS. Kolom CONTENTS punya nilai PERMANENT dan UNDO
    2. DBA_DATA_FILES
    3. DBA_FREE_SPACE

```
select tablespace,CONTENTS,datafile,freespace
from
(select tablespace_name tablespace, CONTENTS from dba_tablespaces) a,
(select tablespace_name tablespace1, sum(bytes) datafile from
dba_data_files group by tablespace_name) b,
(select tablespace_name tablespace2, sum(bytes) freespace from
dba_free_space group by tablespace_name) c
where tablespace=tablespace1 and tablespace1=tablespace2(+);
```

- Untuk tablespace dengan content TEMPORARY, view yang di-query adalah V$TEMP_SPACE_HEADER

```
select tablespace_name tablespace,'TEMPORARY' CONTENTS,
sum(BYTES_USED+BYTES_FREE) datafile,sum(BYTES_FREE) freespace from
V$TEMP_SPACE_HEADER df group by tablespace_name;
```

- Untuk melihat pemakaian space secara keseluruhan, union (gabung) dua query di atas. Agar lebih mudah, saya buatkan VIEW saja. Penting: User yang membuat VIEW ini harus punya privilege untuk mengakses secara langsung. Kalau tidak punya privilege akses langsung, tidak akan bisa, bahkan user DBA (system) sekalipun. Dalam contoh ini, administrasi saya lakukan dengan user SYSTEM. Jadi, user SYSTEM harus di beri privilege langsung:

```
conn / as sysdba
grant select on sys.dba_data_files to system;
grant select on sys.dba_free_space to system;
```

```
grant select on sys.dba_tablespaces to system;
grant select on sys.V_$TEMP_SPACE_HEADER to system;
```

- Berikut ini script untuk membuat VIEW tersebut.

```
CREATE OR REPLACE VIEW v_current_space
(TABLESPACE,CONTENTS, DATAFILE, FREESPACE)
as
select tablespace,CONTENTS,datafile,freespace
from
(select tablespace_name tablespace, CONTENTS from dba_tablespaces) a,
(select tablespace_name tablespace1, sum(bytes) datafile from
dba_data_files group by tablespace_name) b,
(select tablespace_name tablespace2, sum(bytes) freespace from
dba_free_space group by tablespace_name) c
where tablespace=tablespace1 and tablespace1=tablespace2(+)
union
select tablespace_name tablespace,'TEMPORARY' CONTENTS,
sum(BYTES_USED+BYTES_FREE) datafile,sum(BYTES_FREE) freespace from
V$TEMP_SPACE_HEADER df group by tablespace_name;
```

  Kalau user SYSTEM (yang membuat VIEW) ini tidak diberi (grant) privilege untuk query langsung (direct query), akan muncul error berikut ketika mencoba membuat VIEW di atas:

```
ORA-01031: insufficient privileges
```

- Untuk melihat current space ulitization, cukup query ke view V_CURRENT_SPACE saja

```
col TABLESPACE for a20;
select * from V_CURRENT_SPACE order by CONTENTS,TABLESPACE;
```

```
TABLESPACE           CONTENTS    DATAFILE  FREESPACE
-------------------- --------- ---------- ----------
SYSAUX               PERMANENT  387973120    8847360
SYSTEM               PERMANENT  492830720   30867456
TEST                 PERMANENT    4194304    4063232
USERS                PERMANENT   13107200    2359296
TEMP                 TEMPORARY   20971520   16777216
UNDOTBS1             UNDO       209715200  202637312
```

Langkah Selanjutnya, Buat tabel untuk menampung data history
Tabel ini mempunyai kolom yang isinya sama persis dengan VIEW v_current_space, kecuali dengan penambahan beberapa kolom.

```
create table t_history_space
(NOMOR NUMBER default 0,
WAKTU DATE,
TABLESPACE VARCHAR2(30),
CONTENTS VARCHAR2(20),
TOTAL NUMBER,
FREE NUMBER,
USED NUMBER,
perubahan NUMBER) tablespace users;
```

Berikutnya, buat prosedur untuk mengcapture pemakaian space saat ini
Proses kerjanya adalah:

1. Lihat pemakaian space saat ini, pakai VIEW v_current_space

2. Dapatkan space yang dipakai (USED) saat ini.
   `USED = DATAFILE - FREESPACE`
3. Lihat space yang dipakai (USED) yang lalu, di tabel t_history_space
4. Dapatkan nilai perubahan data (PERUBAHAN)
   `PERUBAHAN = USED{current} - USED{yang lalu}`
5. Insert data-data dari VIEW v_current_space, USED, dan PERUBAHAN ke tabel t_history_space

Ini procedure tersebut:

```
create or replace procedure p_history_space as
v_NOMOR NUMBER;
begin
select nvl(max(nomor),0)+1 into v_NOMOR from t_history_space;
insert into t_history_space
select v_NOMOR, sysdate, A.TABLESPACE, A.CONTENTS, A.DATAFILE, A.FREESPACE,
A.USED,(A.USED-B.USED) PERUBAHAN from
(select TABLESPACE,CONTENTS, DATAFILE, FREESPACE,(DATAFILE-FREESPACE) USED
from v_current_space) A,
(select TABLESPACE,USED from t_history_space where nomor=v_NOMOR-1) B
where A.TABLESPACE=B.TABLESPACE(+);
commit;
EXCEPTION
WHEN OTHERS THEN -- handles all other errors
ROLLBACK;
end;
/
```

Berikutnya, buat job yang mengeksekusi prosedur tiap hari

```
VARIABLE jobno number;
BEGIN
DBMS_JOB.SUBMIT(:jobno,'p_history_space;',trunc(SYSDATE+1), 'TRUNC(SYSDATE + 1)');
commit;
END;
/
```

Gunakan command berikut untuk mengetahui job no yang baru kita bikin tersebut. Hanya valid bila dijalankan si SESSION yang membuat job tersebut.

```
SQL> print jobno
```

Atau gunakan command berikut. Bisa dijalankan kapan saja.

```
SQL> select job from user_jobs where upper(what) like '%P_HISTORY_SPACE%';
```

Misalkan job no adalah 4. Job ini akan dijalankan mulai besok jam 00.00, dan selanjutnya dijalankan otomatis setiap hari jam 00.00. Jadi sekarang belum dijalankan, makanya tabel t_history_space masih kosong

```
SQL> select * from t_history_space;
no rows selected
```

Biar data hari ini terisi, jalankan job tersebut secara manual.

```
SQL> EXECUTE DBMS_JOB.RUN(4);
```

Sekarang lihat, sudah ada datanya

```
SQL> set lines 120
SQL> select * from t_history_space;
```

```
     NOMOR WAKTU     TABLESPACE           CONTENTS                TOTAL       FREE       USED  PERUBAHAN
---------- --------- -------------------- -------------------- ---------- ---------- ---------- ----------
         1 09-JUL-08 TEST                 PERMANENT               4194304    4063232     131072
         1 09-JUL-08 SYSAUX               PERMANENT             387973120    7667712  380305408
         1 09-JUL-08 SYSTEM               PERMANENT             492830720   25231360  467599360
         1 09-JUL-08 USERS                PERMANENT              13107200    2359296   10747904
         1 09-JUL-08 TEMP                 TEMPORARY              20971520   16777216    4194304
         1 09-JUL-08 UNDOTBS1             UNDO                  209715200  201261056    8454144
```

Melihat perkembangan data

Misalkan setelah 3 hari, kita dapat data history untuk 3 hari. Berikut ini history pemakaian space untuk semua tablespace

```
SQL> set lines 120
SQL> select * from t_history_space;
```

```
     NOMOR WAKTU     TABLESPACE           CONTENTS                TOTAL       FREE       USED  PERUBAHAN
---------- --------- -------------------- -------------------- ---------- ---------- ---------- ----------
         1 07-JUL-08 TEST                 PERMANENT               4194304    4063232     131072
         1 07-JUL-08 SYSAUX               PERMANENT             387973120    7667712  380305408
         1 07-JUL-08 SYSTEM               PERMANENT             492830720   25231360  467599360
         1 07-JUL-08 USERS                PERMANENT              13107200    2359296   10747904
         1 07-JUL-08 TEMP                 TEMPORARY              20971520   16777216    4194304
         1 07-JUL-08 UNDOTBS1             UNDO                  209715200  201261056    8454144
         2 08-JUL-08 TEST                 PERMANENT               4194304    4063232     131072          0
         2 08-JUL-08 SYSAUX               PERMANENT             387973120    7667712  380305408          0
         2 08-JUL-08 SYSTEM               PERMANENT             492830720   25231360  467599360          0
         2 08-JUL-08 USERS                PERMANENT              24903680    1572864   23330816   12582912
         2 08-JUL-08 TEMP                 TEMPORARY              20971520   16777216    4194304          0
         2 08-JUL-08 UNDOTBS1             UNDO                  209715200  201261056    8454144          0
         3 09-JUL-08 TEST                 PERMANENT               4194304    4063232     131072          0
         3 09-JUL-08 SYSAUX               PERMANENT             387973120    7667712  380305408          0
         3 09-JUL-08 SYSTEM               PERMANENT             492830720   25231360  467599360          0
         3 09-JUL-08 USERS                PERMANENT              24903680    7864320   17039360   -6291456
         3 09-JUL-08 TEMP                 TEMPORARY              20971520   16777216    4194304          0
         3 09-JUL-08 UNDOTBS1             UNDO                  209715200  201261056    8454144          0
```

Untuk melihat perkembangan data khusus tablespace USERS

```
SQL> select * from t_history_space where TABLESPACE ='USERS' order by nomor;
```

```
     NOMOR WAKTU     TABLESPACE           CONTENTS                TOTAL       FREE       USED  PERUBAHAN
---------- --------- -------------------- -------------------- ---------- ---------- ---------- ----------
         1 07-JUL-08 USERS                PERMANENT              13107200    2359296   10747904
         2 08-JUL-08 USERS                PERMANENT              24903680    1572864   23330816   12582912
         3 09-JUL-08 USERS                PERMANENT              24903680    7864320   17039360   -6291456
```

Untuk melihat perkembangan datatabase

```
SQL> select min(WAKTU) MULAI, max(WAKTU) SAMPAI, sum(TOTAL) TOTAL,
sum(FREE) FREE, sum(USED) USED, sum(PERUBAHAN) PERUBAHAN from
t_history_space;
```

```
MULAI     SAMPAI          TOTAL       FREE       USED  PERUBAHAN
--------- --------- ---------- ---------- ---------- ----------
07-JUL-08 09-JUL-08 3409969152  776798208 2633170944    6291456
```

Secara intrinsik, data yang riel itu disimpan di tablespace PERMANENT. Tablespace UNDO dan TEMPORARY hanya berisi segment-segment yang berfungsi untuk mendukung kinerja database, bukan berisi data riel. Kita bisa menambah clause **where contents='PERMANENT'** pada query di atas.

```
SQL> select min(WAKTU) MULAI, max(WAKTU) SAMPAI, sum(TOTAL) TOTAL,
sum(FREE) FREE, sum(USED) USED, sum(PERUBAHAN) PERUBAHAN from
t_history_space where contents='PERMANENT';
```

```
MULAI     SAMPAI          TOTAL       FREE       USED  PERUBAHAN
--------- --------- ---------- ---------- ---------- ----------
07-JUL-08 09-JUL-08 2717908992  122683392 2595225600    6291456
```

6. Connectivity:

6.1. Memulai Koneksi ke Database

Setelah menginstall Oracle dan membuat database, untuk langkah awal administrasi adalah mulai melakukan koneksi ke database.

Administrasi dilakukan oleh user yang meng-install dan membuat database. Tool *native* dari Oracle untuk administrasi database adalah `sqlplus`, lokasi ada di `$ORACLE_HOME/bin`. Di Oracle versi 8 ke bawah, tool administrasi tersebut adalah `svrmgrl`.

Sebelum melakukan koneksi, ada OS parameter yang perlu disetting. Di Windows, parameter tersebut otomatis sudah dimasukkan ke dalam registry ketika meng-install dan membuat database pakai dbca. Di Unix, setting manual parameter berikut di user profile: ORACLE_HOME, ORACLE_SID, dan PATH.

Misalkan kita pakai shell sh atau ksh. Edit file `.profile`, tambahkan parameter berikut:

```
ORACLE_HOME=/data1/oracle/product/10.2.0; export ORACLE_HOME
ORACLE_SID=ts; export ORACLE_SID
PATH=$ORACLE_HOME/bin:$PATH; export PATH
```

Setelah mengedit file .profile, jangan lupa untuk relogin atau mengeksekusi file tersebut agar parameter yang disetting terbaca oleh current session. Berikut ini cara mengeksekusi file .profile.

```
. ./.profile
```

Koneksi pakai SQLPLus di Mesin server

Sekarang, mari kita mulai koneksi ke database. Misalkan saya akan connect pakai user system.

```
sqlplus
```

Nanti akan diminta memasukkan username dan password. Kalau belum diubah, password system adalah seperti yang ditunjukkan ketika membuat database.

Bisa juga username langsung dimasukkan ke argument-nya SQLPlus, nanti kita cuma diminta memasukkan password saja.

```
sqlplus system
```

Bisa juga langsung memasukkan username dan password. Misalkan password user system adalah oracle:

```
sqlplus system/oracle
```

Koneksi dengan langsung memasukkan username dan password sekaligus ini tidak direkomendasikan, karena password akan tampak ketika di `ps -ef`. Contoh:

```
ps -ef|grep sql
oracle  5742 25612   11:09:49 pts/1 0:00 sqlplus system/oracle
```

Cara lain juga, kita bisa masuk ke SQLPlus prompt tanpa login, kemudian jalankan perintah **connect** atau **conn** di SQL prompt. Contoh:

```
sqlplus /nolog
SQL> conn
```

Sama seperti ketika menjalankan sqlplus dari OS prompt, username dan password bisa disebutkan langsung atau tidak; kalau tidak disebutkan nanti akan ditanyakan.

```
SQL> conn system/oracle
SQL> conn system
```

Koneksi pakai user sys

User sys adalah merupakan super user, dikenal juga sebagai sysdba. Untuk koneksi pakai user sys, harus ditambahkan argument **as sysdba**. Contoh:

```
SQL> conn sys/oracle as sysdba
```

Bisa juga tanpa menyebutkan user sys, yaitu dengan memakai argument /. Contoh:

```
SQL> conn / as sysdba
```

Kalau tidak sebutkan argument as sysdba, akan muncul error berikut:

```
SQL> conn sys/oracle
ERROR:
ORA-28009: connection as SYS should be as SYSDBA or SYSOPER
Warning: You are no longer connected to ORACLE.
```

Bisa juga langsung login ketika menjalankan SQLPLUS. Contoh:

```
sqlplus "sys/oracle as sysdba"
sqlplus "/ as sysdba"
```

Koneksi dari client ke server

Untuk bisa melakukan koneksi client-server, pastikan kita sudah mensetting dan menjalakan listener di server database, dan mensetting TNSNames di client. Kalau belum punya instalasi Oracle client di mesin/komputer/PC lain, kita bisa memanfaatkan database server sebagai client sekaligus. Ketika kita install software database Oracle, by default juga diinstall Oracle client; sehingga nantinya kita bisa melakukan koneksi client-server di mesin server database kita.

Pada koneksi client-server , tambahkan argument **@namatns**. Contoh:

```
sqlplus system@tsprimary
```

```
sqlplus system/oracle@tsprimary
sqlplus "sys@tsprimary as sysdba"
sqlplus "sys/oracle@tsprimary as sysdba"

SQL> conn system@tsprimary
SQL> conn system/oracle@tsprimary
SQL> conn sys@tsprimary as sysdba
SQL> conn sys/oracle@tsprimary as sysdba
```

6.2. Login ke Database dengan User Lain

Sebagai DBA, kadang kita ingin login ke database dengan user lain sementara kita tidak tahu passwordnya. Contoh, kita diminta untuk mengubah password dari db link di schema TEST. Nama db link tersebut adalah TSREPLINK. Namun sayangnya, orang yang tahu password untuk user TEST sedang tidak di kantor, di telpon pun tidak diangkat. Padahal untuk mengubah db link harus dilakukan oleh user (schema) yang punya db link tersebut. Gimana kah caranya?

Misalkan definisi database link tersebut adalah berikut ini:

```
SQL> create database link TSREPLINK
connect to ROHMAD identified by ROHMADPASS using 'TSREP';
```

Sekarang password user ROHMAD di database TSREP diganti menjadi ROHMADNEW. Mau gak mau kita khan mesti me-recreate database link tersebut dengan password yang baru.

Tapi kalau tidak tahu password user TEST, kita khan tidak bisa connect pakai user TEST tersebut. Mau melakukannya pakai user SYSTEM, ya gak bisa; kalau gak percaya, coba aja:

```
SQL> conn system
SQL> drop database link TEST.TSREPLINK;
ORA-02024: database link not found
```

Menghapus tabel atau index bisa seperti itu, sebutkan nama owner dan nama objectnya (tabel atau index). Tapi menghapus db link tidak seperti itu, harus pakai user (schema) yang punya. Harusnya begini caranya:

```
SQL> connect TEST
SQL> drop database link TSREPLINK;
SQL> create database link TSREPLINK
connect to ROHMAD identified by ROHMADNEW using 'TSREP';
```

BEGINI CARANYA

Yang perlu kita lakukan adalah mengubah password user TEST. Kemudian buka session baru pakai user TEST dengan password yang baru. Kemudian kembalikan lagi password yang diubah tadi itu.

1. Lihat password user TEST (ini adalah password yang telah di-encrypt oleh Oracle; dan dengan cara apapun kita tidak bisa men-decrypt-nya). Jalankan query berikut, catat hasilnya.

   ```
   SQL> conn SYSTEM
   SQL> select PASSWORD from dba_users
   where username='TEST';

   PASSWORD
   ------------------------------
   7A0F2B316C212D67
   ```

2. Setelah itu, ubah password user TEST

```
SQL> alter user TEST identified by TEST_TEMP_PASS;
```

Catatan: walaupun password diganti, session (dari user TEST) tetep *establish*, tidak terpengaruh apa-apa. Hanya saja kita tidak bisa login pakai user TEST dengan password yang lama. Oleh karena itu, segera buru-buru ke langkah selanjutnya.

3. Buka sesssion baru dengan dengan user TEST dan password yang baru

```
SQL> conn TEST/TEST_TEMP_PASS
```

4. Kembali ke session SYSTEM. Kembalikan password user TEST ke password yang lama.

```
SQL> alter user TEST
identified by values '7A0F2B316C212D67';
```

Walaupun password diubah lagi (dikembalikan ke semula), session yang connect dengan password baru tersebut masih *establish* dan bisa melakukan aktifitas biasa.

Sekarang dengan session user TEST yang masih *establish* itu, tinggal kira recreate database link tersebut:

```
SQL> drop database link TSREPLINK;
SQL> create database link TSREPLINK
connect to ROHMAD identified by ROHMADNEW using 'TSREP';
```

6.3. Contoh koneksi dari PHP ke Oracle

Dalam artikel ini saya akan membahas bagaimana membuat koneksi dari PHP ke database Oracle. Dalam contoh, saya akan menampilkan view DBA_USERS ke halaman PHP.

```
SID         = EMREP
Username    = system
Password    = systempasswd
ORACLE_HOME = /oracle/9.2.0
Query       = select username,default_tablespace,
               temporary_tablespace,account_status,
               profile from dba_users order by username;
```

Hasilnya nanti akan seperti ini

Current Tablespace Utilization

Query: 24-July-2008 04:35 PM

User Name	Default Tablespace	Temporary Tablespace	Account Status	Profile
AGUS	MGMT_TABLESPACE	TEMP	OPEN	DEFAULT
DBMON	USERS	TEMP	OPEN	DEFAULT
DBSNMP	SYSTEM	TEMP	OPEN	DEFAULT
MGMT_VIEW	MGMT_TABLESPACE	TEMP	OPEN	DEFAULT
NCCFAULTDC	MGMT_TABLESPACE	TEMP	OPEN	DEFAULT
NCCITVAS	MGMT_TABLESPACE	TEMP	OPEN	DEFAULT
OUTLN	SYSTEM	TEMP	OPEN	DEFAULT
PERFSTAT	TOOLS	TEMP	OPEN	DEFAULT
PRODXL	USERS	TEMP	OPEN	DEFAULT
ROHMAD	MGMT_TABLESPACE	TEMP	OPEN	DEFAULT
SYS	SYSTEM	TEMP	OPEN	DEFAULT
SYSMAN	MGMT_TABLESPACE	TEMP	OPEN	DEFAULT
SYSTEM	SYSTEM	TEMP	OPEN	DEFAULT

Berikut ini source code-nya:

```
<HTML>
<HEAD>
<TITLE>Tablespace</TITLE>
</HEAD>
<BODY TEXT="#000080">
<H1>Current Tablespace Utilization</H1>
<HR>
Query: <? echo (date("d-F-Y h:i A ")); ?>
<BR>
<BR>
<?
$ORACLE_SID = getenv("ORACLE_SID");
$ORACLE_HOME = getenv("ORACLE_HOME");
$user="system";
$pass="systempasswd";
$sid="EMREP";
```

```
$tns_name="/oracle/9.2.0/network/admin/tnsnames.ora";
$c1 = ociplogon($user, $pass, $sid, $tns_name);
if ($c1 == false){
echo OCIError($c1)."<BR>";
exit;
}
echo ("<TABLE BORDER=1>");
echo ("<TH BGCOLOR=#99CCCC>User Name</TH>");
echo ("<TH BGCOLOR=#99CCCC>Default Tablespace</TH>");
echo ("<TH BGCOLOR=#99CCCC>Temporary Tablespace</TH>");
echo ("<TH BGCOLOR=#99CCCC>Account Status</TH>");
echo ("<TH BGCOLOR=#99CCCC>Profile</TH>");
$query = "select username,default_tablespace,";
$query .= "temporary_tablespace,account_status,";
$query .= "profile from dba_users order by username";
$stmt = OCIparse($c1,$query);
OCIexecute($stmt,OCI_DEFAULT);
while (OCIfetchInto($stmt,&$userinfo)) {
?>
<TR>
<TD ALIGN="left"><? echo $userinfo[0]; ?></TD>
<TD ALIGN="right"><? echo $userinfo[1]; ?></TD>
<TD ALIGN="right"><? echo $userinfo[2]; ?></TD>
<TD ALIGN="right"><? echo $userinfo[3]; ?></TD>
<TD ALIGN="right"><? echo $userinfo[4]; ?></TD>
</TR>
<?} // endwhile
OCIFreeStatement($stmt);
OCILogoff($c1);
//}
?>
</BODY>
</HTML>
```

7. SQL

7.1. Reserved Word di database Oracle

Reserved word adalah kata yang sudah di reserved (dikapling) oleh database Oracle. Contoh reserved word adalah select, delete, update, session, uid, key, rowid, dll. Daftar reserved word ada di view V$RESERVED_WORDS.

```
SQL> select * from V$RESERVED_WORDS;
```

Semua kata yang tercakup dalam reserved word TIDAK bisa digunakan untuk memberi nama object database, nama kolom pada tabel, dll. Contoh object database adalah: tabel, index, view, synonym, database link, dll. Gunakan query berikut untuk melihat tipe-tipe object database:

```
SQL> select distinct OBJECT_TYPE from DBA_OBJECTS order by OBJECT_TYPE;
```

Contoh masalah yang sering muncul berkaitan dengan reserved word ini adalah ketika kita migrasi database dari non Oracle (misal MySQL) ke database Oracle. Misalkan di MySQL kita punya tabel SESSION dengan kolom NO dan UID. Ketika kita migrasi ke Oracle, katakanlah kita membuat tabel dengan definisi yang sama:

```
SQL> CREATE TABLE SESSION (NO NUMBER(2),UID VARCHAR2(20));

ORA-00903: invalid table name
```

SESSION dibilang sebagai “invalid table name”. Setelah kita check di V$RESERVED_WORDS, ternyata kata SESSION termasuk dalam daftar reserved word. Demikian juga kata UID yang kita pakai untuk nama kolom tersebut, juga termasuk reserved word.

Lantas, apa solusinya? Mau tidak mau kita harus mengubah nama table dan nama column tersebut.

```
SQL> CREATE TABLE SESSION_NEW (NO NUMBER(2),UID_NEW VARCHAR2(20));
```

Dengan mengubah nama tabel dan kolom tersebut, mau tidak mau kita harus mengedit script aplikasi kita. Wah… berat sekali ya, kalau aplikasi kita banyak dan complicated.

Kalau masih memaksa pengin memakai kata yang termasuk dalam reserved word tersebut, gunakan kutip dua (double quote) pada kata tersebut. Contoh

```
SQL> CREATE TABLE "SESSION" (NO NUMBER(2),"UID" VARCHAR2(20));
```

Namun, kita masih harus tetep mengubah coding di aplikasi.

7.2. Menampilkan rownum ganjil dan genap

Misalkan suatu aplikasi ingin menampilkan rownum ganjil dan genap, bagaimana caranya? [Dalam contoh ini saya menggunakan view dba_users sebagai object yang diquery].

Secara logika, untuk menampilkan rownum genap adalah berikut ini :

```
SQL> select rownum from dba_users where rownum/2=trunc(rownum/2);
```

Dan berikut ini untuk menampilkan rownum ganjil:

```
SQL> select rownum from dba_users where rownum/2<>trunc(rownum/2);
```

OO… ternyata hasilnya tidak muncul, padahal view tersebut mempunyai isi. Setelah dicoba-coba, ternyata operasi rownum yang diperbolehkan di clause where hanya operasi lebih dari, kurang dari, dan between.

Kalau begitu harus menggunakan sedikit trik. Lakukan query dalam query:

```
SQL> select a.numrow nogenap
from (select rownum as numrow from dba_users) a
where (a.numrow/2)=trunc(a.numrow/2);
```

OO… ternyata berhasil  Untuk menampilkan rownum ganjil di kolom 1 dan rownum genap di kolom 2, berikut ini query-nya:

```
SQL> select noganjil, nogenap
from
(select a.numrow nogenap from
(select rownum as numrow from dba_users) a
where (a.numrow/2)=trunc(a.numrow/2)) genap,
(select a.numrow noganjil from
(select rownum as numrow from dba_users) a
where (a.numrow/2)<>trunc(a.numrow/2)) ganjil
where nogenap (+)= (noganjil+1);
```

7.3. Pivot Query: konversi row ke column

Inti pivot query adalah menampilkan data row menjadi column. Contoh praktisnya, saya punya data berikut ini

```
create table trx(PART varchar2(1), TRX_DATE date,
DOC_NO varchar2(10), TRX_CODE varchar2(1), AWAL number,
TRX_QTY number,AKHIR number);
```

```
insert into trx values('A',to_date('01-01-08','dd-mm-yy'),'DOC 1','I',10, 5 , 15);
insert into trx values('A',to_date('01-01-08','dd-mm-yy'),'DOC 2','I',15 , 5 , 20);
insert into trx values('A',to_date('01-01-08','dd-mm-yy'),'DOC 3','O',20, 10 , 10 );
insert into trx values('B',to_date('01-01-08','dd-mm-yy'),'DOC 1','I',100 , 25 , 125 );
insert into trx values('B',to_date('01-01-08','dd-mm-yy'),'DOC 2','I', 125 , 25 , 150 );
insert into trx values('B',to_date('01-01-08','dd-mm-yy'),'DOC 3','O', 150 , 75 , 75 );
insert into trx values('A',to_date('02-01-08','dd-mm-yy'),'DOC 4','O',10 , 5 , 5);
insert into trx values('A',to_date('02-01-08','dd-mm-yy'),'DOC 5','I', 5 , 5 , 10);
insert into trx values('A',to_date('02-01-08','dd-mm-yy'),'DOC 6','O',10 , 10 , 0);
insert into trx values('B',to_date('02-01-08','dd-mm-yy'),'DOC 4','I',75 , 25 , 100 );
insert into trx values('B',to_date('02-01-08','dd-mm-yy'),'DOC 5','I',100 , 25 , 125);
insert into trx values('B',to_date('02-01-08','dd-mm-yy'),'DOC 6','O',125 , 100 , 25 );
```

```
SQL> select * from trx order by TRX_DATE,part;
```

P	TRX_DATE	DOC_NO	T	AWAL	TRX_QTY	AKHIR
A	01-JAN-08	DOC 1	I	10	5	15
A	01-JAN-08	DOC 2	I	15	5	20
A	01-JAN-08	DOC 3	O	20	10	10
B	01-JAN-08	DOC 1	I	100	25	125
B	01-JAN-08	DOC 2	I	125	25	150
B	01-JAN-08	DOC 3	O	150	75	75
A	02-JAN-08	DOC 4	O	10	5	5
A	02-JAN-08	DOC 5	I	5	5	10
A	02-JAN-08	DOC 6	O	10	10	0
B	02-JAN-08	DOC 4	I	75	25	100
B	02-JAN-08	DOC 5	I	100	25	125
B	02-JAN-08	DOC 6	O	125	100	25

Lihat kolom TRX_CODE dan TRX_QTY. Nilai kolom TRX_CODE adalah I dan O. Saya ingin menampilkan total transaksi (TRX_QTY) untuk TRX_IN (di mana TRX_CODE = I) dan TRX_OUT (di mana TRX_CODE = O)

```
TRX_IN TRX_OUT
------ ------
   115    200
```

Untuk menampilkan hasil query di atas, perintah SQL nya adalah sebagai berikut

```
SQL> SELECT
MAX(DECODE(TRX_CODE,'I',total,null)) TRX_IN,
MAX(DECODE(TRX_CODE,'O',total,null)) TRX_OUT
FROM
(
```

```
SELECT TRX_CODE,sum(TRX_QTY) total
FROM trx
GROUP BY TRX_CODE
)
;
```

Sekarang, saya ingin menampilkan total transaksi (TRX_QTY) untuk TRX_IN dan TRX_OUT masing-masing PART tiap harinya:

```
PART TRX_DATE      TRX_IN    TRX_OUT
---- --------- ---------- ----------
A    01-JAN-08         10         10
B    01-JAN-08         50         75
A    02-JAN-08          5         15
B    02-JAN-08         50        100
```

Command SQL untuk hasil query di atas adalah

```
SELECT PART,TRX_DATE,
MAX(DECODE(TRX_CODE,'I',total,null)) TRX_IN,
MAX(DECODE(TRX_CODE,'O',total,null)) TRX_OUT
FROM
(
SELECT PART,trunc(TRX_DATE) TRX_DATE,TRX_CODE,sum(TRX_QTY) total
FROM trx
GROUP BY TRX_CODE, PART,trunc(TRX_DATE)
)
group by PART,TRX_DATE
order by TRX_DATE,PART
;
```

Bisa juga gunakan perintah SQL berikut ini:

```
SELECT DISTINCT part, date_trx, trx_in, trx_out FROM (
SELECT part, trx_date date_trx,
SUM(DECODE(trx_code,'I',trx_qty,0)) OVER (PARTITION BY part, trx_date)
trx_in,
SUM(DECODE(trx_code,'O',trx_qty,0)) OVER (PARTITION BY part, trx_date)
trx_out
FROM trx)
ORDER BY date_trx, part;
```

Sekarang saya ingin menambahkan tampilan data-data berikut ini
- AWAL : Diambil dari nilai awal dari transaksi pertama part tsb.
- AKHIR : Diambil dari nilai akhir dari transaksi terakhir part tsb.

```
PART DATE_TRX        AWAL     TRX_IN    TRX_OUT      AKHIR
---- --------- ---------- ---------- ---------- ----------
A    01-JAN-08         10         10         10         10
B    01-JAN-08        100         50         75         75
A    02-JAN-08         10          5         15          0
B    02-JAN-08         75         50        100         25
```

Perintah SQL untuk hasil query di atas adalah (hint: gunakan analytic function) :

```
SELECT DISTINCT part, date_trx, awal, trx_in, trx_out, akhir FROM (
SELECT part, trx_date date_trx,
FIRST_VALUE(awal) OVER (PARTITION BY part, trx_date ORDER BY doc_no) awal,
SUM(DECODE(trx_code,'I',trx_qty,0)) OVER (PARTITION BY part, trx_date)
trx_in,SUM(DECODE(trx_code,'O',trx_qty,0)) OVER (PARTITION BY part,
trx_date)
trx_out,LAST_VALUE(akhir) OVER (PARTITION BY part, trx_date) akhir
FROM trx)
ORDER BY date_trx, part;
```

8. PL/SQL

8.1. Mengenal Oracle PL/SQL (1): Contoh Kasus

PL/SQL (Procedural Language/Structured Query Language) merupakan pengembagan SQL oleh Oracle. Prasyarat mempelajari PL/SQL adalah paling tidak mengetahui dasar-dasar SQL. Sebagai awalan belajar PLSQL, mari kita lihat contoh kasus dan contoh blok PL/SQL berikut ini.

Misalkan saya punya tabel MYTAB. Tabel ini berisi data transaksi. Kolom rcg_id (menjadi PRIMARY KEY) berisi transaksi ID yang digenerate oleh SEQUENCE, jadi nilainya urut (karena digenerate oleh sequence) dan unique (karena primary key).

Saya ingin menghapus data transaksi di bawah tanggal 17-JUN-08. Kita bisa saja men-delete dengan perintah SQL berikut:

```
delete from MYTAB where
SYS_CREATION_DATE < to_date('17-JUN-08','DD-MON-YY');
```

Karena datanya sangat banyak, maka akan diperlukan undo (rollback) segment yang besar karena COMMIT dilakukan setelah proses delete selesai. Saya tidak ingin ada konsumsi rollback segment yang besar, karena akan mempengaruhi performa database dan tentu saja perlu UNDO space yang besar. Sebagai alternatif lainnya, saya ingin mendelete (dan commit) data secara per record. Nah, sekarang saatnya saya memakai PL/SQL.

Saya akan mendelete per record (baris). Acuan yang saya gunakan adalah kolom rcg_id, karena nilainya unique (primary key) dan urut (digenerate oleh sequence). Ini langkah-langkahnya

1. Saya perlu mendapat rcg_id minimal dan maximal untuk data transaksi di bawah tanggal 17-JUN-08
   ```
   SQL> select min (rcg_id),max(rcg_ID) from MYTAB where
   SYS_CREATION_DATE < to_date('17-JUN-08','DD-MON-YY');
   ```
2. Setelah mendapat rcg_id minimal dan maximal, selanjutnya saya akan buat PROSES-nya. Proses delete dimulai dari rcg_id minimal, kemudian rcg_id minimal + 1, kemudian rcg_id minimal + 2, dan seterusnya hingga mencapai rcg_id maximal
3. Selanjutnya saya akan buat program PL/SQL nya

Berikut ini block PL/SQL yang telah saya buat

```
DECLARE
V_rcg_min NUMBER;
v_rcg_max number;
V_iterasi NUMBER;
BEGIN
V_rcg_min:= &1;
v_rcg_max:= &2;
V_iterasi:=V_rcg_min;
WHILE V_iterasi <= v_rcg_max LOOP
```

```
delete from MYTAB where rcg_id=V_iterasi;
commit;
V_iterasi:=V_iterasi+1;
END LOOP;
dbms_output.put_line('Deleting sucess');
dbms_output.put_line('Min RCG_ID '||V_rcg_min);
dbms_output.put_line('Max RCG_ID '||V_rcg_max);
EXCEPTION
WHEN OTHERS THEN dbms_output.put_line('error here');
END;
/
```

Penjelasan
Block PL/SQL di atas bisa kita jalankan langsung di SQLPlus. Bisa juga kita taruh di file dan kemudian dari SQLPlus kita panggil file tersebut. Contoh, block PL/SQL ini saya taruh di file roh.sql. Berikut ini cara manggil dari SQLPlus: (Ups, jangan lupa untuk menjalankan perintah "set serveroutput on" agar hasil dari "dbms_output.put_line" bisa tampak di monitor)

```
SQL> set serveroutput on
SQL> @roh.sql
```

Begitu script roh.sql kita jalankan, maka kita akan diminta memasukkan nilai untuk parameter &1 dan &2. Seperti ini tampilannya

```
SQL> @roh.sql
Enter value for 1: 305206565
old 6: V_rcg_min:= &1;
new 6: V_rcg_min:= 305206565 ;
Enter value for 2: 305209524
old 7: v_rcg_max:= &2;
new 7: v_rcg_max:= 305209524;
```

Anda bisa juga langsung menyertakan nilai &1 (305206565) dan &2 (305209524) ketika memanggil **roh.sql**

```
SQL> @roh.sql 305206565 305209524
```

Hasilnya akan nampak di monitor seperti berikut ini

```
==========================
Deleting sucess
Min RCG_ID 305206565
Max RCG_ID 305209524
PL/SQL procedure successfully completed.
```

Bila kita tidak menjalankan "set serveroutput on" sebelumnya, maka yang nampak di monitor hanya

```
PL/SQL procedure successfully completed.
```

Bila tidak ingin muncul pesan "PL/SQL procedure successfully completed", jalankan command "set feed off" di SQLPlus

```
SQL> set serveroutput off
```

Bersambung ke Mengenal Oracle PL/SQL (2): Struktur

Referensi
Oracle® Database PL/SQL User's Guide and Reference 10g Release 2 (10.2)

8.2. Mengenal Oracle PL/SQL (2): Struktur

Setelah melihat contoh penggunaannya, sekarang mari kita bahas dasar-dasar PL/SQL. Silahkan lihat block PL/SQL yang telah kita bahas tersebut:

```
DECLARE
     V_rcg_min NUMBER;
     v_rcg_max number;
     V_iterasi NUMBER;
BEGIN
     V_rcg_min:= &1;
     v_rcg_max:= &2;
     V_iterasi:=V_rcg_min;
WHILE V_iterasi <= v_rcg_max LOOP
     delete from MYTAB where rcg_id=V_iterasi;
     commit;
     V_iterasi:=V_iterasi+1;
END LOOP;
     dbms_output.put_line('Deleting sucess');
     dbms_output.put_line('Min RCG_ID '||V_rcg_min);
     dbms_output.put_line('Max RCG_ID '||V_rcg_max);
EXCEPTION
     WHEN OTHERS THEN dbms_output.put_line('error here');
END;
/
```

Struktur pokok PL/SQL adalah sebagai berikut

```
DECLARE
BEGIN
END;
/
```

DECLARE

Berisi deklarasi variabel. Adapun isi dari deklarasi variabel adalah nama variable, tipe data, constraint, dan default value. Setiap satu deklarasi diakhiri dengan tanda **;** (titik koma). Contoh:

1. Yang wajib ada : nama variabel dan tipe data
   ```
   v_rcg_min NUMBER;
   ```
2. Dengan menambahkan nilai default
   ```
   v_jam_kerja INTEGER DEFAULT 40;
   v_jam_kerja INTEGER := 0;
   ```
3. Dengan menambahkan constraint "not null" dan nilai default
   ```
   v_acc_id INTEGER(4) NOT NULL := 9999;
   ```
4. Deklarasi konstanta
   ```
   v_jumlah_hari_pertahun CONSTANT INTEGER := 366;
   v_wni CONSTANT BOOLEAN := FALSE;
   ```
5. Memakai tipe data dari suatu kolom di tabel (contoh, tabel: tbl emp, kolom: empid)
   ```
   v_empid tbl_emp.empid%TYPE;
   ```
6. Contoh lain, silahkan lihat referensi

Bagian Utama

Bagian Utama ada di antara **BEGIN** dan **END**. Setelah END, tambahkan tanda **;** (titik koma).

Agar block PL/SQL bisa dieksekusi, tambahkan baris baru di bawah END dan beri tanda / (slash atau garis miring).

Bagian utama berisi operasi (pekerjaan) yang kita lakukan. Contoh block PL/SQL di atas berisi:

- Memberi nilai variable
- Operasi SQL (delete dan commit)
- Operasi aritmatika (penjumlahan)
- Control struktur (LOOP dan WHILE)
- EXCEPTION (error handler)

Memberi nilai variabel

Cara memberi nilai pada variabel adalah memakai **:=** (titik dua dan sama dengan). Contoh:

```
v_rcg_min:= &1;
v_rcg_max:= 100;
v_iterasi:=v_rcg_min;
```

Kita juga bisa memberi nilai ke variabel melalui SQL command. Lihat contoh berikut ini, nilai untuk variabel **v_job** adalah hasil dari "select job from emp where EMPID=10″

```
DECLARE
     v_job VARCHAR2(9);
BEGIN
     select job into v_job from emp where EMPID=10;
     dbms_output.put_line(v_job);
END;
/
```

Operasi SQL

Hampir semua perintah SQL bisa dijalankan di sini. Iya dong, khan sesuai dengan namanya, di mana PL/SQL adalah pengembangan dari SQL.

Control Struktur

Macam-macam control structure adalah:

- Testing Conditions: IF dan CASE
- Controlling Loop Iterations: LOOP dan EXIT
- Sequential Control: GOTO dan NULL

Menggunakan LOOP

Contoh yang telah saya pakai di atas adalah WHILE …LOOP

```
WHILE V_iterasi <= v_rcg_max LOOP
     delete from MYTAB where rcg_id=V_iterasi;
     commit;
     V_iterasi:=V_iterasi+1;
END LOOP;
```

Dengan hasil yang sama, kita bisa menggunakan LOOP … EXIT WHEN

```
LOOP
    delete from MYTAB where rcg_id=V_iterasi;
    commit;
    V_iterasi:=V_iterasi+1;
EXIT WHEN V_iterasi > v_rcg_max;
END LOOP;
```

Bisa juga dengan memakai LOOP dan di dalamnya ada IF … THEN

```
LOOP
    delete from MYTAB where rcg_id=V_iterasi;
    commit;
    V_iterasi:=V_iterasi+1;
IF (V_iterasi > v_rcg_max) THEN
    exit;
END IF;
END LOOP;
```

Contoh Menggunakan IF … THEN

```
IF (v_gaji > v_umr) THEN
    v_bonus:=v_gaji*2;
END IF;

IF (v_gaji < v_umr) THEN
    v_bonus:=v_gaji*4;
ELSIF (v_gaji = v_umr) THEN
    v_bonus:=v_gaji*3;
ELSE
    v_bonus:=v_gaji*2;
END IF;
```

Contoh Mengguakan CASE

```
v_nilai := 'B';
CASE v_nilai
WHEN 'A' THEN v_predikat := 'Excellent';
WHEN 'B' THEN v_predikat := 'Very Good';
WHEN 'C' THEN v_predikat := 'Good';
WHEN 'D' THEN v_predikat := 'Fair';
WHEN 'F' THEN v_predikat := 'Poor';
ELSE v_predikat := 'Nothing';
END CASE;
```

Referensi

Oracle® Database PL/SQL User's Guide and Reference 10g Release 2 (10.2)

8.3. PL/SQL: Membuat Prosedur

Tulisan ini merupakan lanjutan dari dasar-dasar pengenalan PL/SQL yang telah saya tulis sebelumnya, yaitu contoh penggunaan PL/SQL dan Struktur PL/SQL.

Prosedur merupakan subprogram PL/SQL yang berdiri sendiri. Kalau kita punya pekerjaan rutin dan command-commandnya pun itu-itu saja, kita bisa menyimpan comand-command tersebut dan memanggilnya kapan saja kita mau. Itulah filosofi dari prosedur.

Cara Membuat Prosedur

Caranya sama persis dengan membuat blok PL/SQL biasa, cuma ganti :

```
DECLARE
```

Menjadi :

```
create or replace procedure NAMA_PROSEDUR as
```

User yang membuat prosedur harus punya privilege "create procedure". Contoh, memberi privilege kepada user ROHMAD agar bisa membuat prosedur

```
SQL> conn SYSTEM
SQL> grant create procedure to roh;
```

User yang tidak punya privilege "create procedure" , kalau membuat prosedur akan mendapat error berikut:

```
ORA-01031: insufficient privileges
```

Contoh

Sebagai contoh, saya punya tabel MYTAB

```
create table MYTAB (SYS_CREATION_DATE date, RCG_ID number);
```

Berikut ini Block PL/SQL untuk mengosongkan dan mengisi ulang table MYTAB. Prosesnya dalah sbb:

1. Truncate table mytab
2. Insert ke tabel mytab, mulai dari rcg_id minimal (10) sampai rcg_id maksimal (100)

```
DECLARE
     V_rcg_min NUMBER;
     v_rcg_max number;
     V_iterasi NUMBER;
     v_date      DATE;
BEGIN
     V_rcg_min := 10;
     v_rcg_max := 100;
     V_iterasi := V_rcg_min;
     V_DATE    := sysdate;
     EXECUTE IMMEDIATE 'truncate table MYTAB';
WHILE V_iterasi <= v_rcg_max LOOP
     insert into MYTAB values (v_date, V_iterasi);
     commit;
     v_date   := v_date+1;
```

```
     V_iterasi:= V_iterasi+1;
END LOOP;
END;
/
```

Selanjutnya, mari kita coba membuat prosedur berdasarkan block PL/SQL di atas. Ingat kuncinya, ganti kata "DECLARE" menjadi "create or replace procedure NAMA_PROSEDUR as". Di contoh ini prosedurnya saya beri nama PROC_REFRESH_MYTAB:

```
create or replace procedure PROC_REFRESH_MYTAB as
     V_rcg_min NUMBER;
     v_rcg_max number;
     V_iterasi NUMBER;
     v_date      DATE;
BEGIN
     V_rcg_min := 10;
     v_rcg_max := 100;
     V_iterasi := V_rcg_min;
     V_DATE    := sysdate;
     EXECUTE IMMEDIATE 'truncate table MYTAB';
WHILE V_iterasi <= v_rcg_max LOOP
     insert into MYTAB values (v_date, V_iterasi);
     commit;
     v_date   := v_date+1;
     V_iterasi:= V_iterasi+1;
END LOOP;
END;
/
```

Untuk menjalankan prosedur, jalankan:

1. Di SQLPlus
   ```
   SQL> exec PROC_REFRESH_MYTAB;
   ```
   atau
   ```
   SQL> execute PROC_REFRESH_MYTAB;
   ```

2. Di block PL/SQL, tulis saja nama prosedur tersebut
   ```
   DECLARE
   BEGIN
   PROC_REFRESH_MYTAB;
   END;
   /
   ```

8.4. PL/SQL: Memasukkan Variabel dalam Prosedur

Sebagaimana prosedur dalam bahasa pemrograman lain, kitapun bisa memasukkan variabel ke dalam prosedur. Lihat contoh prosedur PROC_REFRESH_MYTAB yang lalu

```
create or replace procedure PROC_REFRESH_MYTAB as
        V_rcg_min NUMBER;
        v_rcg_max number;
        V_iterasi NUMBER;
        v_date    DATE;
BEGIN
        V_rcg_min := 10;
        v_rcg_max := 100;
        V_iterasi := V_rcg_min;
        V_DATE    := sysdate;
        EXECUTE IMMEDIATE 'truncate table MYTAB';
WHILE V_iterasi <= v_rcg_max LOOP
        insert into MYTAB values (v_date, V_iterasi);
        commit;
        v_date   := v_date+1;
        V_iterasi:= V_iterasi+1;
END LOOP;
END;
/
```

Dalam contoh di atas, nilai v_rcg_min dan v_rcg_max dimasukkan dalam hard code. Kalau kita mau mengubah nilai rcg_id minimum dan maksimum, ya mesti mengubah code lagi. Cappe deh … 😃
Caranya, devinisikan variabel-variabel tersebut di belakang NAMA_PROSEDUR. Nilainya nanti dimasukkan ketika memanggil prosedur. Berikut ini prosedur tersebut:

```
create or replace procedure PROC_REFRESH_MYTAB2
(V_rcg_min NUMBER, v_rcg_max number) as
        V_iterasi NUMBER;
        v_date    DATE;
BEGIN
        V_iterasi := V_rcg_min;
        V_DATE    := sysdate;
        EXECUTE IMMEDIATE 'truncate table MYTAB';
WHILE V_iterasi <= v_rcg_max LOOP
        insert into MYTAB values (v_date, V_iterasi);
        commit;
        v_date   := v_date+1;
        V_iterasi:= V_iterasi+1;
END LOOP;
END;
/
```

Berikut ini cara memanggil prosedur sekaligus memasukkan nilai untuk variabelnya (misalkan v_rcg_min saya beri nilai 20 dan v_rcg_max saya beri nilai 30):

1. Di SQLPlus
   ```
   SQL> exec PROC_REFRESH_MYTAB2 (20,30);
   ```
2. Di block PL/SQL, tulis saja nama prosedur tersebut
   ```
   DECLARE
   ```

```
BEGIN
PROC_REFRESH_MYTAB2 (20,30);
END;
/
```

Kalau nilai variabel tidak dimasukkan, terlalu banyak, atau terlalu sedikit; maka akan error:

```
SQL> exec PROC_REFRESH_MYTAB2 (20);
SQL> exec PROC_REFRESH_MYTAB2 (20, 30, 40);
SQL> exec PROC_REFRESH_MYTAB2;

ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00306: wrong number or types of arguments in call to
'PROC_REFRESH_MYTAB2'
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored
```

8.5. Menjalankan OS Command atau Shell Script dari PL/SQL

Menjalankan OS Command atau shell script dari SQLPlus sudah biasa kita lakukan. Biasanya kita menggunakan **!** atau **host** (catatan: di Windows hanya bisa pakai **host**, tidak bisa pakai **!**), contoh:

```
SQL> ! ls -la
SQL> ! /data1/oracle/Users/rohmad/test.sh
SQL> host ls -al
SQL> host /data1/oracle/Users/rohmad/test.sh
```

Namun ini hanya berlaku:

- OS command ini eksekusi di mana kita menjalankan SQL Plus.
- OS command yang dijalankan adalah OS command di mana kita menjalankan SQL Plus. Misalnya kita menjalankan SQLPlus di PC kita, maka OS command yang dijalankan adalah OS command yang ada di PC kita, bukan OS command di mesin/server database

Tantangan berikutnya:

- bagaimana menjalankan OS command (shell script) yang ada di mesin/server database sementara kita memanggilnya lewat SQLPlus yang ada di PC kita?
- Bagimana caranya menjalankan OS command (shell script) dari PL/SQL atau prosedur?

Nah, artikel ini akan menjawab tantangan tersebut. Kita akan memanfaatkan DBMS_SCHEDULER, feature ini mulai dikenalkan sejak Database Oracle versi 10g. Untuk versi 9i ke bawah, bisa mencari referensi di asktom.oracle.com

* * *

Misalkan saya punya script:

```
/data1/oracle/Users/rohmad/test.sh
```

Content script ini adalah:

```
#!/usr/bin/ksh
cd /data1/oracle/Users/rohmad/
/usr/bin/echo "Hello `date` " >>  test.log
```

Persiapan

Saya ingin menjalankan script tersebut melalui PL/SQL. Oleh Oracle, shell script tersebut dipandang sebagai external script (bukan scriptnya Oracle).

Khusus di Solaris dan Linux (saya tidak tahu untuk system Unix lainnya): Untuk bisa menjalankan external script, pastikan bahwa Oracle bisa menjalankan **external job**; berikut ini step-step untuk meng-enable-kan external job tersebut:

1. Lihat file $ORACLE_HOME/rdbms/admin/externaljob.ora

   ```
   cd $ORACLE_HOME/rdbms/admin
   ls -la externaljob.ora

   -rw-r-----   1 oracle   dba   externaljob.ora
   ```

   Kemudian lihat isinya:

   ```
   more externaljob.ora

   run_user = nobody
   run_group = nobody
   ```

2. Dengan user root, ubah owner dan mode dari file externaljob.ora tersebut

   ```
   chown root externaljob.ora
   chmod 640 externaljob.ora
   ```

   Edit file externaljob.ora, ganti *run_user* dan *run_group* menjadi user dan group dari Oracle Installation. Dalam contoh ini, Oracle Installation adalah milik user *oracle* dengan group *dba*

   ```
   run_user = oracle
   run_group = dba
   ```

3. Lihat file $ORACLE_HOME/bin/extjob
   Dengan user root, ubah owner dan mode

   ```
   cd $ORACLE_HOME/bin
   chown root extjob
   chmod 4750 extjob
   ```

Dalam contoh ini, shell script tersebut akan dijalankan oleh user TEST. Beri privilege ke user TEST

```
SQL> conn system
SQL> grant create job to test;
SQL> grant create EXTERNAL job to test;
```

Langkah Utamanya

Caranya mudah sekali. Jalankan dengan block PL/SQL berikut ini:

```
connect TEST

BEGIN
DBMS_SCHEDULER.CREATE_JOB(
job_name=>'testjob',
job_type=>'EXECUTABLE',
```

```
job_action=>'/data1/oracle/Users/rohmad/test.sh',
enabled=>true,
auto_drop=>true);
end;
/
```

Intinya adalah:

1. Buat job dengan DBMS_SCHEDULER
2. Begitu di-create, jalankan job tersebut.
   Ditunjukkan oleh parameter *enabled=>true*
3. Setelah job dijalankan, drop (hapus) job tersebut
   Ditunjukkan oleh parameter *auto_drop=>true*

Kalau cara di atas terasa kepanjangan, kita bisa membuat prosedur-nya. Misalnya prosedur itu kita beri nama **jalankan**

```
Create or replace procedure jalankan (cmd in varchar2) as
Begin
        DBMS_SCHEDULER.CREATE_JOB(
                job_name=>'testjob',
                job_type=>'EXECUTABLE',
                job_action=> cmd,
                enabled=>true,
                auto_drop=>true);
end;
/
```

Sekarang, tinggal kita panggil prosedur **jalankan** tersebut

1. Bisa melalui command SQL berikut

   ```
   exec jalankan ('/data1/oracle/Users/rohmad/test.sh');
   ```

2. Ataupun memanggilnya lewat block PL/SQL

   ```
   begin
   jalankan ('/data1/oracle/Users/rohmad/test.sh');
   end;
   /
   ```

Catatan :

- Baik dipanggil dari SQLPlus yang ada di server database maupun di client (PC kita), prosedur **jalankan** ini tetep menjalankan shell script (OS command) yang ada di server database.
- Gunakan SQL command berikut untuk melihat, apakah job yang kita buat itu berhasil apa tidak

  ```
  select log_id, log_date, job_name, status, error#,
  additional_info from user_scheduler_job_run_details
  where job_name like 'TEST%';
  ```

- Gunakan SQL command berikut untuk memastikan bahwa job yang kita buat itu telah di-drop begitu selesai dijalankan

```
select job_name,job_type,job_action from
user_SCHEDULER_JOBS;
```

- Penting: Untuk bisa dijalanakan oleh DBMS_SCHEDULER, paling tidak script tersebut harus bisa dijalanakan lewat crontab. Tambahkan shell definition pada script, contoh kalau menggunakan shell KSH:

```
#!/usr/bin/ksh
```

Kalau memanggil perintah Oracle misalnya sqlplus atau sqlldr, tambahkan parameter ORACLE_HOME dan PATH (misalkan ORACLE_HOME ada di /dirOracle/10.2.3)

```
export ORACLE_HOME=/dirOracle/10.2.3
export PATH=$ORACLE_HOME/bin:$PATH
```

Referensi:

1. Oracle® Database Administrator's Guide 10g Release 2 (10.2) — Using the Scheduler
2. Oracle® Database PL/SQL Packages and Types Reference 10g Release 2 (10.2) — DBMS_SCHEDULER

9. Application development

9.1. Partitioning Table: Definisi dan Contoh

Filosofi partisi adalah memecah tabel ke dalam beberapa segment (partisi atau subpartisi), di mana tabel konvensional hanya mempunyai satu segment.

Misalkan kita punya tabel PENJUALAN dengan 8 juta records, kita ingin query data untuk quartal pertama tahun ini. Pada tabel konvensional (non partition), query akan men-scan keseluruhan 8 juta records data tersebut karena berada dalam 1 segment. Nah, kalau tabel itu dipartisi (by range untuk kolom tanggal penjualan) maka query akan men-scan khusus segment di mana data itu berada; tidak semua 8 juta records data itu di-scan, sehingga proses query lebih cepat.

Manfaat lain dari partitioning adalah tiap-tiap segment (partisi atau subpartisi) bisa ditaruh di tablespace yang berbeda, sehingga kita mendapat manfaat dari *spreading* (menyebar) tablespace, yaitu penyebaran I/O dan mengurangi resiko loss data karena tablespace corrupt.

Ada 3 metode utama partisi, dan ada 2 macam composite (gabungan):

1. Range partitioning
2. List partitioning
3. Hash partitioning
4. Composite range-list partitioning
5. Composite range-hash partitioning

Misalkan saya punya tabel penjualan yang punya kolom no_invoice, tgl_jual, dan area.

```
CREATE TABLE penjualan
    ( no_invoice       NUMBER,
      tgl_jual         DATE,
      area             varchar2(10));
```

Dalam artikel ini saya juga akan memberi contoh macam-macam partisi yang bisa dilakukan pada tabel penjualan tersebut.

Range Partition

Pada range partition, data dikelompokkan berdasarkan range (rentang) nilai yang kita tentukan. Range partition ini cocok digunakan pada kolom yang nilainya terdistribusi secara merata. Contoh yang paling sering adalah kolom tanggal.

Berikut ini contoh membuat table PENJUALAN dengan partisi range pada kolom tgl_jual (untuk menegaskan bahwa ini adalah contoh range partition, tabel saya beri nama PENJUALAN_RANGE):

```
CREATE TABLE penjualan_range
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
```

```
area varchar2(10))
PARTITION BY RANGE (tgl_jual)
(
PARTITION jual_kw1 VALUES LESS THAN (TO_DATE('01-APR-2008','DD-MON-YYYY')) TABLESPACE users,
PARTITION jual_kw2 VALUES LESS THAN (TO_DATE('01-JUL-2008','DD-MON-YYYY')) TABLESPACE users,
PARTITION jual_kw3 VALUES LESS THAN (TO_DATE('01-OCT-2008','DD-MON-YYYY')) TABLESPACE users,
PARTITION jual_kw4 VALUES LESS THAN (TO_DATE('01-JAN-2009','DD-MON-YYYY')) TABLESPACE users
);
```

Dalam contoh di atas, spesifikasi tablespace ada pada tiap segment (partisi). Yang seperti ini (spesifikasi tablespace pada tiap segment) biasanya dilakukan kalau tiap segment (partisi) ditempatkan pada tablespace yang berbeda. Karena keterbatasan tablespace, dalam contoh ini saya taruh dalam satu tablespace yang sama, yaitu tablespace USERS.

List Partition

Pada list partition, data dikelompokkan berdasarkan nilainya. Cocok untuk kolom yang variasi nilainya tidak banyak. Saya masih menggunakan contoh table penjualan. Yang cocok dengan list partition adalah kolom area.

Berikut ini contoh membuat table PENJUALAN dengan partisi list pada kolom area (untuk membedakan dengan contoh range partition, tabel saya beri nama PENJUALAN_LIST):

```
CREATE TABLE penjualan_list
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
area varchar2(10))
PARTITION BY LIST (area)
(
PARTITION daerah_barat VALUES ('JAKARTA','MEDAN','BANDUNG') ,
PARTITION daerah_timur VALUES ('SEMARANG','SURABAYA','MAKASAR')
) TABLESPACE users;
```

Definisi tablespace bisa didefinisikan di level partisi ataupun tabel. Dalam contoh di atas, karena semua partisi ditaruh di tablespace yang sama, maka definisi tablespace cukup ditaruh di definisi tabel (tidak perlu di tiap partisi).

Hash Partition

Jika kita ingin melakukan partisisi namun tidak cocok dengan RANGE ataupun LIST, maka kita bisa menggunakan HASH partition. Penentuan "nilai mana di taruh di partisi mana" itu diatur secara internal oleh Oracle (berdasarkan *hash value*).

Kenapa kita memaksakan memakai partisi sementara tidak cocok dengan RANGE ataupun LIST? Lha, ya itu tadi, kita ingin mendapat manfaat dari filosofi PARTITIONING di mana data disebar ke segment-segment yang berbeda.

Untuk tabel penjualan, kolom yang cocok dengan HASH partitioning adalah kolom no_invoice. Berikut ini contohnya (tabel saya beri nama PENJUALAN_HASH):

```
CREATE TABLE penjualan_hash
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
area varchar2(10))
```

```
PARTITION BY HASH (no_invoice)
PARTITIONS 4 tablespace users;
```

Secara otomatis Oracle akan membuat 4 partisi (sesuai dengan nilai dari parameter PARTITIONS). Dalam contoh di atas, definisi tablespace ditaruh di definisi tabel, yang berarti semua partisi (segment) di taruh di tablespace yang sama (tablespace USERS).

Kita bisa menaruh definisi tablespace di tiap partisi bila kita ingin tiap partisi (segment) disimpan di tablespace yang berbeda. Karena keterbatasan tablespace, dalam contoh ini semua partisi ditaruh di tablespace Users:

```
CREATE TABLE penjualan_hash
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
area varchar2(10))
PARTITION BY HASH (no_invoice)
PARTITIONS 4 STORE IN (users, users, users, users);
```

Composite range-list partition

Dengan karakteristik yang seperti itu, tabel PENJUALAN bisa kita partisi pada kolom TGL_JUAL dan AREA. Berikut ini contohnya (untuk membedakan dengan contoh-contoh sebelumnya, tabel saya beri nama PENJUALAN_RANGE_LIST):

```
CREATE TABLE PENJUALAN_RANGE_LIST
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
area varchar2(10))
PARTITION BY RANGE (tgl_jual)
SUBPARTITION BY LIST (area)
(
PARTITION jual_kw1 VALUES LESS THAN (TO_DATE('01-APR-2008','DD-MON-YYYY')) TABLESPACE users
(SUBPARTITION kw1_barat VALUES ('JAKARTA','MEDAN','BANDUNG'),
SUBPARTITION kw1_timur VALUES ('SEMARANG','SURABAYA','MAKASAR')
),
PARTITION jual_kw2 VALUES LESS THAN (TO_DATE('01-JUL-2008','DD-MON-YYYY')) TABLESPACE users
(SUBPARTITION kw2_barat VALUES ('JAKARTA','MEDAN','BANDUNG'),
SUBPARTITION kw2_timur VALUES ('SEMARANG','SURABAYA','MAKASAR')
),
PARTITION jual_kw3 VALUES LESS THAN (TO_DATE('01-OCT-2008','DD-MON-YYYY')) TABLESPACE users
(SUBPARTITION kw3_barat VALUES ('JAKARTA','MEDAN','BANDUNG'),
SUBPARTITION kw3_timur VALUES ('SEMARANG','SURABAYA','MAKASAR')
),
PARTITION jual_kw4 VALUES LESS THAN (TO_DATE('01-JAN-2009','DD-MON-YYYY'))
(SUBPARTITION kw4_barat VALUES ('JAKARTA','MEDAN','BANDUNG') TABLESPACE users ,
SUBPARTITION kw4_timur VALUES ('SEMARANG','SURABAYA','MAKASAR')  TABLESPACE users
)
);
```

Dalam contoh di atas, definisi tablespace ada yang di level PARTISI dan ada yang level di SUBPARTISI.

Composite range-hash partition

Kitapun bisa mempartisi tabel PENJUALAN pada kolom TGL_JUAL dan NO_INVOICE. Berikut ini contohnya (untuk membedakan dengan contoh-contoh sebelumnya, tabel saya beri nama PENJUALAN_RANGE_HASH):

```
CREATE TABLE penjualan_range_hash
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
area varchar2(10))
PARTITION BY RANGE (tgl_jual)
SUBPARTITION BY HASH (no_invoice)
SUBPARTITIONS 4 STORE IN (users, users, users, users)
(
PARTITION jual_kw1 VALUES LESS THAN (TO_DATE('01-APR-2008','DD-MON-YYYY')),
PARTITION jual_kw2 VALUES LESS THAN (TO_DATE('01-JUL-2008','DD-MON-YYYY')),
PARTITION jual_kw3 VALUES LESS THAN (TO_DATE('01-OCT-2008','DD-MON-YYYY')),
PARTITION jual_kw4 VALUES LESS THAN (TO_DATE('01-JAN-2009','DD-MON-YYYY'))
);
```

Urutan tablespace

Berikut ini hirarki untuk alokasi tablespace:

1. Kalau di definisi SUBPARTITION tidak disebutkan tablespacenya, maka SUBPARTITION akan ditaruh di tablespace yang didefinisikan di PARTITION.
2. Kalau di definisi PARTITION tidak disebutkan tablespacenya, maka PARTITION akan ditaruh di tablespace yang didefinisikan di TABLE
3. Kalau tablespacenya tidak disebutkan (baik di level SUPPARTITION, PARTITION, maupun TABLE) maka, semua segment (baik SUPPARTITION maupun PARTITION) ditaruh di default tablespace dari user (schema) yang bersangkutan.

VIEW (data dictionary)

Beberapa contoh VIEW yang sering dipakai untuk melihat informasi PARTISI dan SUBPARTISI adalah DBA_TABLES, DBA_TAB_PARTITIONS, DBA_TAB_SUBPARTITIONS, DBA_INDEXES, DBA_IND_PARTITIONS, DBA_IND_SUBPARTITIONS.

Untuk melihat VIEW-VIEW yang lain, bisa query ke DICT:

```
SQL> select table_name from dict
where table_name like '%PARTITION%' order by table_name;
```

Referensi

Oracle® Database Administrator's Guide 10g Release 2 (10.2)
Managing Partitioned Tables and Indexes

9.2. Partitioning Table: Informasi Segment & Tablespace

Kadang kita bingung, ketika query TABLESPACE_NAME di view DBA_TABLES, kok nilai TABLESPACE_NAME ada yang NULL (blank/kosong).

```
SQL> select distinct TABLESPACE_NAME from dba_tables;

TABLESPACE_NAME
------------------------------
SYSTEM
                                                <--- Kosong/blank/NULL
USERS
SYSAUX
```

Tidak perlu bingung, itu tandanya ada tabel berpartisi di database tersebut. Pada tabel yang berpartisi, data disimpan di dalam partisinya. Contoh, lihat tabel berpartisi PENJUALAN_RANGE yang pernah dibahas di Partitioning Table: Definisi dan Contoh.

```
CREATE TABLE penjualan_range
( no_invoice  NUMBER,
tgl_jual    DATE NOT NULL,
area varchar2(10))
PARTITION BY RANGE (tgl_jual)
(
PARTITION jual_kw1 VALUES LESS THAN (TO_DATE('01-APR-2008','DD-MON-YYYY')) TABLESPACE users,
PARTITION jual_kw2 VALUES LESS THAN (TO_DATE('01-JUL-2008','DD-MON-YYYY')) TABLESPACE users,
PARTITION jual_kw3 VALUES LESS THAN (TO_DATE('01-OCT-2008','DD-MON-YYYY')) TABLESPACE users,
PARTITION jual_kw4 VALUES LESS THAN (TO_DATE('01-JAN-2009','DD-MON-YYYY')) TABLESPACE users
);
```

Coba lihat, view DBA_TABLES. Hasil query berikut ini adalah 1 record, di mana nilai kolom TABLESPACE_NAME adalah NULL.

```
SQL> select tablespace_name from dba_tables where
table_name='PENJUALAN_RANGE';

TABLESPACE_NAME
------------------------------

SQL> select count(*) from dba_tables where table_name='PENJUALAN_RANGE';

  COUNT(*)
----------
         1
```

Sekarang, query view DBA_TAB_PARTITIONS. Tampak bahwa data disimpan di partisi JUAL_KW1, JUAL_KW2, JUAL_KW3, dan JUAL_KW4.

```
SQL> select TABLESPACE_NAME,PARTITION_NAME from DBA_TAB_PARTITIONS where
table_name='PENJUALAN_RANGE';

TABLESPACE_NAME                PARTITION_NAME
------------------------------ ------------------------------
USERS                          JUAL_KW1
USERS                          JUAL_KW2
```

```
USERS                          JUAL_KW3
USERS                          JUAL_KW4
```

Lihat juga informasi di view DBA_SEGMENTS

```
SQL> select SEGMENT_NAME,PARTITION_NAME,TABLESPACE_NAME
from dba_segments where SEGMENT_NAME='PENJUALAN_RANGE';

SEGMENT_NAME       PARTITION_NAME  TABLESPACE
------------------ --------------- ----------
PENJUALAN_RANGE    JUAL_KW1        USERS
PENJUALAN_RANGE    JUAL_KW2        USERS
PENJUALAN_RANGE    JUAL_KW3        USERS
PENJUALAN_RANGE    JUAL_KW4        USERS
```

9.3. Menggunakan SQL*Loader

SQL Loader adalah tool Oracle untuk me-load data dari flat (text) file ke dalam tabel di database Oracle. Program (executable) SQL Loader adalah sqlldr, lokasi ada di $ORACLE_HOME/bin.

SQL Loader sering digunakan untuk mengatasi kendala jaringan. Misalkan perusahaan punya kantor cabang yang tidak tersambung online dengan pusat; data transaksi biasanya dikirim ke pusat dalam bentuk text file; di pusat, data tersebut kemudian di-load ke database pakai SQL Loader.

SQL Loader juga sering digunakan untuk me-load data dari system yang berbeda. Misalkan transaksi di-handle oleh database non Oracle, sementara datawarehouse menggunakan Oracle. Nah, data transaksi dari non Oracle ini kemudian di-export ke dalam text file, dan kemudian di-import ke Oracle pakai SQL*Loader.

Misalkan saya punya file penjualan.dat yang berisi:

```
3286;23-DEC-08;SEMARANG
3287;24-DEC-08;SURABAYA
3288;25-DEC-08;MAKASAR
3289;26-DEC-08;MEDAN
3290;26-DEC-08;MAGELANG TENGAH
```

File penjualan.dat tersebut akan saya load ke tabel penjualan:

```
CREATE TABLE penjualan
    ( no_invoice       NUMBER,
      tgl_jual         DATE,
      area             varchar2(10));
```

Langkap pertama, buat control file yang berisi parameter-parameter untuk SQL Loader. Misalkan saya beri nama penjualan.ctl.

```
load data
APPEND
into table PENJUALAN
fields terminated by ";"
TRAILING NULLCOLS
(no_invoice,tgl_jual,area)
```

Langkah selanjutnya, load data di penjualan.dat ke tabel PENJUALAN. Gunakan perintah berikut:

```
sqlldr USERID=test/test CONTROL=penjualan.ctl, DATA=penjualan.dat,
LOG=penjualan.log, BAD=penjualan.bad
```

Parameter untuk sqlldr adalah:

1. USERID: user dan password ke database Oracle
2. CONTROL: control file yang akan digunakan

3. DATA: file (data) yang akan di-load
4. LOG: file untuk menyimpan log dari proses loader
5. BAD: file untuk menyimpan data yang tidak diprose

Setelah menjalankan sqlldr di atas, lihat lognya di file penjualan.log. Oo… ternyata ada error:

```
Record 5: Rejected - Error on table PENJUALAN, column AREA.
ORA-12899: value too large for column "TEST"."PENJUALAN"."AREA" (actual:
15, maximum: 10)
```

Record (baris) kelima tidak diproses karena lebar kolom AREA hanya 10 karakter, sementara nilai yang dimasukkan panjangnya 15 karakter (MAGELANG TENGAH). Record yang tidak diposes ini ditaruh di file penjualan.bad. Sekarang coba query ke tabel PENJUALAN, data yang masuk hanya 4 record.

```
SQL> select * from penjualan;

NO_INVOICE TGL_JUAL  AREA
---------- --------- ----------
      3286 23-DEC-08 SEMARANG
      3287 24-DEC-08 SURABAYA
      3288 25-DEC-08 MAKASAR
      3289 26-DEC-08 MEDAN
```

Referensi

Oracle® Database Utilities 10g Release 2 (10.2) - SQL*Loader

9.4. Menggunakan External Table

External table adalah tabel yang datanya ada di luar database, biasanya berupa text file. External table sering digunakan untuk :

1. membaca file dari database Oracle.
2. Me-load (import) data dari text file ke database. Sebagai alternatif lain dari SQL*Loader.

Misalkan saya punya file `penjualan.dat` di direktori `/data1/oracle/Users/rohmad/external`.

```
$ cd /data1/oracle/Users/rohmad/external
$ more penjualan.dat

3286;23-DEC-08;SEMARANG
3287;24-DEC-08;SURABAYA
3288;25-DEC-08;MAKASAR
3289;26-DEC-08;MEDAN
3290;26-DEC-08;MAGELANG TENGAH
```

Berikut ini langkah-langkah untuk membuat external table berdasarkan file tersebut.

Persiapan
Buat directory di database yang mengarah ke directory file tersebut.

```
SQL> conn / as sysdba
SQL> CREATE OR REPLACE DIRECTORY external_dir
AS '/data1/oracle/Users/rohmad/external';
```

Beri privilege ke user agar bisa membaca dan menulis ke directory tersebut.

```
SQL> GRANT READ ON DIRECTORY external_dir TO test;
SQL> GRANT WRITE ON DIRECTORY external_dir TO test;
```

Membuat External Table

```
SQL> conn test
SQL> CREATE TABLE ext_penjualan
    ( no_invoice       NUMBER,
      tgl_jual         DATE,
      area             varchar2(10))
ORGANIZATION EXTERNAL
     (
       TYPE ORACLE_LOADER
       DEFAULT DIRECTORY external_dir
       ACCESS PARAMETERS
       (
         records delimited by newline
         badfile external_dir:'penjualan.bad'
         logfile external_dir:'penjualan.log'
         fields terminated by ';'
         missing field values are null
```

```
         ( no_invoice, tgl_jual, area
         )
       )
       LOCATION ('penjualan.dat')
     )
     REJECT LIMIT UNLIMITED;
```

Query External Table

```
SQL> select * from ext_penjualan;

NO_INVOICE TGL_JUAL  AREA
---------- --------- ----------
      3286 23-DEC-08 SEMARANG
      3287 24-DEC-08 SURABAYA
      3288 25-DEC-08 MAKASAR
      3289 26-DEC-08 MEDAN
```

Dari kelima records yang ada di file penjualan.dat, ada 4 records yang terbaca oleh Oracle. Sekarang kita lihat log-nya, pasti ada error ketika membaca salah satu record tersebut.

```
$ cd /data1/oracle/Users/rohmad/external
$ more penjualan.log

error processing column AREA in row 5 for datafile
/data1/oracle/Users/rohmad/external/penjualan.dat
ORA-12899: value too large for column AREA (actual: 15, maximum: 10)
```

Errornya sama persis dengan yang pernah saya bahas di Menggunakan SQL*Loader. Sebagaimana SQL*Loader, record yang tidak diproses ditaruh di BADFILE (penjualan.bad).

Kalau yang salah adalah datanya, misalnya kolomnya kepanjangan, ya datanya yang mesti diedit. Kalau datanya tidak masalah, itu artinya kolom di tabel yang kurang panjang. Kita bisa mengedit external tabel tersebut:

```
SQL> alter table ext_penjualan modify (area varchar2(15));
SQL> select * from ext_penjualan;

NO_INVOICE TGL_JUAL  AREA
---------- --------- ---------------
      3286 23-DEC-08 SEMARANG
      3287 24-DEC-08 SURABAYA
      3288 25-DEC-08 MAKASAR
      3289 26-DEC-08 MEDAN
      3290 26-DEC-08 MAGELANG TENGAH
```

Load data dengan external table

Nah, sekarang kita bisa me-load (import) data dari file penjualan.dat kedatabase dengan memanfaatkan external table.

Misalkan data akan diload ke tabel history_penjualan:

```
CREATE TABLE history_penjualan
    ( no_invoice       NUMBER,
      tgl_jual         DATE,
      area             varchar2(15));
```

Ya, tinggal insert saja:

```
SQL> insert into history_penjualan select * from ext_penjualan;
SQL> select * from history_penjualan;

NO_INVOICE TGL_JUAL  AREA
---------- --------- ---------------
      3286 23-DEC-08 SEMARANG
      3287 24-DEC-08 SURABAYA
      3288 25-DEC-08 MAKASAR
      3289 26-DEC-08 MEDAN
      3290 26-DEC-08 MAGELANG TENGAH
```

Referensi

Oracle® Database Administrator's Guide 10g Release 2 (10.2) - Managing External Table

10. Replication and Distributed System:

10.1. Administrasi database link

Database link, atau dikenal dengan db link, adalah sarana komunikasi antar dua database. Dengan db link, kita bisa melakukan query dari satu database ke database lain. Sering dipakai dalam sistem database terdistribusi (distributed system) dan replikasi (replication).

Pada dasarnya tidak ada batasan apakah kedua database Oracle yang berkomunikasi melalui db link itu versinya sama atau tidak, kecuali secara eksplisit disebutkan Oracle. Yang pasti, database yang sama versinya (walaupun beda OS-nya) komunikasi db link terjadi secara perfect. Kalau kedua database beda versi, menurut pengalaman saya:

- Dari 8i ke 9i (atau sebaliknya) gak masalah
- Dari 8i ke 10g (atau sebaliknya) ada masalah, misalnya data bisa diquery tapi hasilnya "no rows selected" padahal seharusnya ada rows-nya.
- Dari 9i ke 10g (atau sebaliknya) gak malasalah

Ada pertanyaan: masalahnya penyebabnya apa pak dan bagaimana pemecahannnya?

Penjelasan saya: Saya dah lupa, itu sekitar 2 tahun yang lalu. Pernah bikin SR ke Oracle, kayaknya sih emang ada incompatibility db link antara 8i dan 10g. CMIIW. Akhirnya yang saya lakukan:

1. Untuk sementara waktu, saya bikin view-view di 9i (yang datanya merupakan query pakai db link ke database yang 10g). Trus … dari 8i saya query pakai db link ke 9i. Jadi, saya pakai database 9i untuk 'jembatan'.
2. Karena tabel di 10g yang mesti diquery dari 8i ada banyak, pakai solusi sementara di atas adalah melelahkan. Akhirnya database yang 8i saya upgrade ke 9i.

Langkah-langkah membuat database link

Misalkan saya punya database tsrep (IP 10.21.75.200). Di database ini saya akan membuat user TSREP_USER yang mempunyai tabel TSREP_TAB. Guide untuk membuat user ada di Administrasi User

```
SQL> conn SYSTEM
SQL> create user TSREP_USER identified by TSREP_PASS;
SQL> grant connect, resource to TSREP_USER;
SQL> conn TSREP_USER
SQL> create table TSREP_TAB (NOMOR number);
SQL> insert into TSREP_TAB values (45);
SQL> commit;
```

Saya ingin melakukan query di tabel TSREP_TAB dari database lain (misalnya database TSPRIM). Di database TSPRIM lakukan langkah-langkah berikut:

1. Buat tnsnames
   Kita bisa melakukannya pakai netca atau secara manual menambahkan entry berikut ke file $ORACLE_HOME/network/admin/tnsnames.ora

```
tsrep=
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = 10.21.75.200)(PORT = 1521))
)
(CONNECT_DATA =
(SID = tsrep)
)
)
```

2. Persiapkan user yang akan melakukan query
```
SQL> conn SYSTEM
SQL> create user TS_USER identified by TS_PASS;
SQL> grant connect, resource to TS_USER;
SQL> conn TS_USER
```

3. Beri grant "create database link" ke user TS_USER agar bisa membuat db link
```
SQL> conn SYSTEM
SQL> grant create database link to TS_USER;
```

4. Buat database link
   Format perintahnya adalah:
```
create database link NAMA_DATABASE_LINK
connect to NAMA_USER_DI_DATABASE_REMOTE
identified by PASSWORD_USER_DI_DATABASE_REMOTE
using TNSNAMES
```

   Berikut ini contohnya:
```
SQL> conn TS_USER
SQL> create database link GET_TSREP connect to TSREP_USER identified
by TSREP_PASS using 'tsrep';
```

   Kalau user TS_USER tidak diberi grant "create database link" akan muncul error ini
```
ORA-01031: insufficient privileges
```

5. coba query ke tabel di schema TSREP_USER di database TSREP. Tambahkan **@NAMADATABASE_LINK** di belakang tabel/view yang akan di-query. Contoh:
```
SQL> select * from TSREP_TAB@GET_TSREP;
```

   Bisa juga query ke view-view yang bisa diakses oleh user TSREP_USER di database TSREP
```
SQL> select * from tab@GET_TSREP;
SQL> select * from dual@GET_TSREP;
SQL> select * from user_tables@GET_TSREP;
```

DATABASE LINK PUBLIC

Db link yang baru dibuat di atas hanya bisa diakses oleh user yang membuatnya (TS_USER). Agar db link bisa diakses oleh user lain, maka:

- User lain tersebut membuat db link sendiri

- Perlu dibuat PUBLIC database link. Db link PUBLIC bisa diakses oleh semua user database.

Yang bisa membuat db link public adalah user yang punya privilege DBA (misalnya SYS dan SYSTEM) atau user biasa yang mempunyai privilege "create public database link". Contoh, beri privilege ke user TS_USER:

```
SQL> conn SYSTEM
SQL> grant create public database link to TS_USER;
```

Cara membuat db link public sama seperti db link biasa, hanya tambahkan kata PUBLIC sebelum nama db link. Contoh:

```
SQL> conn TS_USER
SQL> create PUBLIC database link GET_TSREP_PUB connect to TSREP_USER
identified by TSREP_PASS using 'tsrep';
```

Sekarang coba query db link tersebut dari user lain. Misalkan saya sudah punya user TEST

```
SQL> conn TEST
SQL> select * from dual@GET_TSREP;
ORA-02019: connection description for remote database not found
```

User TEST tidak bisa query ke db link GET_TSREP karena db link ini secara privat milik user lain (TS_USER). Tetapi bisa query ke db link GET_TSREP_PUB yang public itu:

```
SQL> conn TEST
SQL> select * from dual@GET_TSREP_PUB;
```

MENGUBAH DATABASE LINK

Misalkan kita ingin mengubah definisi db link, contoh: mengubah nama user untuk connect atapun password. Tidak ada command alter buat db link. Kalau begitu, ya mesti

- Drop (hapus) db link yang dimaksud
- Buat db link lagi dengan dengan spesifikasi (definisi) yang dikendaki

MENGAPUS DATABASE LINK

```
SQL> conn TS_USER
SQL> drop database link GET_TSREP;
```

Untuk db link public, tidak bisa dihapus dengan cara di atas. Akan ada error berikut:

```
SQL> conn TS_USER
SQL> drop database link GET_TSREP_PUB;
ORA-02024: database link not found
```

Untuk itu harus disebutkan juga clause PUBLIC

```
SQL> drop public database link GET_TSREP_PUB;
```

Yang bisa menghapus db link public hanya user dengan privilege DBA (misalnya SYS dan SYSTEM) dan user biasa yang diberi privilege "drop public database link". Walaupun user TS_USER punya privilege untuk membuat db link public, dia tetep tidak bisa mengapus db link public sebelum diberi privilege "drop public database link" (meski db link public itu dia

sendiri yang membuatnya)

```
SQL> conn TS_USER
SQL> drop public database link GET_TSREP_PUB;
ORA-01031: insufficient privileges
```

Beri privilege ke user TS_USER

```
SQL> conn SYSTEM
SQL> grant drop public database link to TS_USER;
```

10.2. Replikasi: Membuat Materialized View (Snapshot)

Solusi replikasi dari Oracle di antaranya adalah Oracle Stream dan Advanced Replication . Advanced Replication meliputi Multimaster, Materialized View, dan hybrid (antara Multimaster replication dan materialized view).

Istilah Materialized View (MV) dipakai Oracle sejak versi 9i. Di versi 8i ke bawah di sebut sebagai snapshot. MV merupakan View yang dimaterialisasi. View konvensional tidak menyimpan data, dia hanya menyimpan definisi (nama kolom, table) sementara data secara fisik masih ada di tabel source-nya. Saya punya contoh 2 view:

1. Query ke suatu tabel di database lain melalui database link (db link)
2. Query yang join ke beberapa tabel

Kalau kedua View itu sering diakses sementara datanya sangat besar, dapat kita bayangkan betapa beratnya pekerjaan itu. Seandainya hasil view itu ditaruh ke dalam tabel dummy, dan akses ke view selanjutnya diarahkan ke tabel dummy tersebut, sungguh akan sangat mempercepat proses (tanpa query melalui database link yang dibatasi oleh bandwidth network, dan tanpa perlu melakukan join query yang berulang-ulang). Inilah yang mendasari Oracle untuk mematerialisasi view (semacam membuat tabel dummy untuk view tersebut).

MV dengan database link biasanya digunakan untuk replikasi (**replication**) dan distribusi data (**distributed Database**). Sementara MV dengan multi join (ke banyak tabel) digunakan untuk **data warehouse**.

Berikut ini langkah-langkah membuat MV dengan database link. Dalam script ini kata MATERIALIZED VIEW saya ganti SNAPSHOT, di mana dua terminologi ini mempunyai arti dan fungsi yang sama.

ENVIRONMENT

Database source
- IP: 10.21.106.81
- Nama Instance: DBSOURCE
- Nama Table: TBLMV
- Nama Owner: OWNSOURCE

Di Database yang akan kita buat Snapshot:
- Nama Owner: SNAPSHOT_USER
- Nama snapshot: TBLMV. Refresh (sinkronisasi) dilakukan tiap malam
- TNS Name untuk koneksi ke database DBSOURCE: DBSOURCE
- Nama db link: LINKDBSOURCE

Persiapan di Database Source

Buat MV log (snapshot log)

```
SQL> conn OWNSOURCE
SQL> create snapshot log on TBLMV tablespace USERS;
```

Snapshot log menyimpan delta (perubahan) data di tabel source. Dengan adanya log ini, ketika snapshot di-refresh, maka snapshot hanya mengambil delta yang ada di source tersebut. Inilah yang disebut dengan **refresh fast**. Kalau tidak menggunakan snapshot log, kita tidak bisa melakukan **refresh fast**, dengan kata lain ketika refresh maka yang dilakukan adalah melakukan query ulang ke tabel source.

Ini step-step di database tempat Snapshot

1. Siapkan tablespace untuk menyimpan data milik schema SNAPSHOT_USER
```
Create tablespace snapshot_tbs
datafile '/data/snapshot_tbs01.dbf' size 8192M;
alter tablespace snapshot_tbs add
datafile '/data/snapshot_tbs02.dbf' size 8192M;'
```

2. Buat user
```
SQL> conn / as sysdba
SQL> create user snapshot_user identified by napshot_passwd default
tablespace snapshot_tbs temporary tablespace temp;
SQL> grant resource to snapshot_user;
SQL> grant connect to snapshot_user;
SQL> grant create snapshot to snapshot_user;
SQL> grant create database link to snapshot_user;
SQL> grant create public synonym to snapshot_user;
```

3. Buat TNS Names. Bisa pakai netca atau manual dengam menambahkan entry berikut di file $**ORACLE_HOME/network/admin/tnsnames.ora**
```
DBSOURCE =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS=(PROTOCOL=TCP)(HOST=10.21.106.81)(PORT=1521))
)
(CONNECT_DATA =
(service_name = DBSOURCE)
)
)
```

4. Buat database link
```
SQL> conn SNAPSHOT_USER
SQL> create database link LINKDBSOURCE connect to OWNSOURCE
identified by OWNSOURCE_PASSWD using 'DBSOURCE';
```

5. Buat Snapshot
```
SQL> conn SNAPSHOT_USER
SQL> create snapshot TBLMV
refresh fast
start with sysdate next trunc(sysdate + 1)
as select * from TBLMV@LINKDBSOURCE;
```

   Snapshot di atas akan direfresh mulai nanti malam jam 12, **trunc(sysdate + 1)**. Pekerjaan refresh ini dilakukan oleh job Oracle, lihat jobnya di view USER_JOBS atau DBA_JOBS. Dengan DBMS_JOB kita bisa merubah schedule refresh.Secara fisik, snapshot TBLMV mempunyai tabel dengan name TBLMV (disimpan di default tablespace: snapshot_tbs).

6. Kalau diperlukan, kitapun bisa membuat index di tabel TBLMV sesuai kehendak kita (tidak ada aturan untuk menyamakan index di sini dengan yang di SOURCE). Contoh, kita akan membuat index untuk kolom nilai:
```
SQL> create index TBLMV_01_IDX on TBLMV (nilai)
```

Referensi

1. Oracle® Database Advanced Replication - 10g Release 2 (10.2) Materialized View Concepts and Architecture
2. Oracle® Database Data Warehousing Guide - 10g Release 2 (10.2) Basic Materialized Views
3. Oracle® Database Data Warehousing Guide - 10g Release 2 (10.2) Advanced Materialized Views

11. Backup and Recovery

11.1. Off line (cold) backup database Oracle

Ada dua metode untuk membackup database Oracle, yaitu off line dan online. Off line backup dilakukan dengan mematikan database terlebih dahulu, baru kemudian membackup datafile. Online backup dilakukan tanpa mematikan database, jadi database masih bisa diakses selama proses backup.

Online backup mensyaratkan database berada dalam mode archive log. Nanti akan saya bahas di artikel lain. Kali ini saya akan membahas tentang off line backup.

Berikut ini step-step untuk off line backup:

1. lihat daftar file oracle (datafile, logfile, dan control file)
Gunakan query berikut ini untuk melihat semua file

```
SQL> select name as file_name from
(select name from v$tempfile union
select name from v$datafile union
select name from v$controlfile union
select member as name from v$logfile)
order by file_name;
```

2. Matikan database

```
SQL> shutdown immediate;
```

3. Backup file-file database (datafile, logfile, dan control file)

```
$ cp /oradata/ts/control01.ctl /backup/ts/control01.ctl
$ cp /oradata/ts/control02.ctl /backup/ts/control02.ctl
$ cp /oradata/ts/control03.ctl /backup/ts/control03.ctl
$ cp /oradata/ts/redo04.log /backup/ts/redo04.log
$ cp /oradata/ts/redo05.log /backup/ts/redo05.log
$ cp /oradata/ts/redo06.log /backup/ts/redo06.log
$ cp /oradata/ts/sysaux01.dbf /backup/ts/sysaux01.dbf
$ cp /oradata/ts/system01.dbf /backup/ts/system01.dbf
$ cp /oradata/ts/temp01.dbf /backup/ts/temp01.dbf
$ cp /oradata/ts/undotbs01.dbf /backup/ts/undotbs01.dbf
$ cp /oradata/ts/users01.dbf /backup/ts/users01.dbf
```

4. Nyalakan database

```
SQL> startup;
```

Untuk me-restore dari offline backup ini, silahkan lihat di
Restore dari off line backup

11.2. Online (hot) backup database Oracle

Beberapa waktu yang lalu saya telah menulis tentang off line backup database Oracle di Off line (cold) backup database Oracle. Kali ini saya akan membahas tentang online backup. Beberapa keuntungan metode online backup adalah :

1. Waktu backup, database tidak perlu dimatikan sehingga tidak ada downtime
2. Bisa melakukan backup per tablespace, bahkan per datafile
3. Bisa merestore data sampai terakhir sebelum masalah. Bahkan bisa merestore data sampai waktu yang ditentukan. Berbeda dengan restore hasil offline backup di mana yang bisa direstore hanya data terakhir melakukan backup saja.

Seperti yang sudah pernah dibahas, syarat online backup adalah database harus dalam mode **archivelog**. Untuk melihat atau men-setting mode database apakah archivelog atau noarchivelog, lihat artikel di Men-setting database menjadi archivelog mode

Berikut ini step-step online backup.

1. Backup control file
```
SQL> alter database backup controlfile to
'/backupdir/backupcontrol_22042008.bak';
```

2. Lihat Tablespace yang berisi data
```
SQL> select TABLESPACE_NAME from dba_tablespaces where CONTENTS
<>'TEMPORARY';

TABLESPACE_NAME
------------------------------
SYSTEM
UNDOTBS1
SYSAUX
USERS
```

   Temporary tablespace tidak perlu dibackup, karena bukan berisi data

3. Ubah mode tablespace menjadi **backup mode**
```
SQL> alter tablespace SYSTEM begin backup;
SQL> alter tablespace UNDOTBS1 begin backup;
SQL> alter tablespace SYSAUX begin backup;
SQL> alter tablespace USERS begin backup;
```

4. Lihat semua datafile yang perlu dibackup (selain file dari temporary tablespace)
```
SQL> select name from v$datafile;

NAME
------------------------------------
/oradata/oracle/ts/system01.dbf
/oradata/oracle/ts/undotbs01.dbf
/oradata/oracle/ts/sysaux01.dbf
/oradata/oracle/ts/users01.dbf
```

5. Backup datafile
```
$ cp /oradata/oracle/ts/system01.dbf /backupdir/...
$ cp /oradata/oracle/ts/undotbs01.dbf /backupdir/..
$ cp /oradata/oracle/ts/sysaux01.dbf /backupdir/...
$ cp /oradata/oracle/ts/users01.dbf /backupdir/..
```

6. Ubah mode database kembali ke normal
```
SQL> alter tablespace SYSTEM end backup;
SQL> alter tablespace UNDOTBS1 end backup;
SQL> alter tablespace SYSAUX end backup;
SQL> alter tablespace USERS end backup;
```

7. Archive current log
```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
```

8. Jangan lupa, backup juga archived log-nya. Archived log ini nanti kita butuhkan untuk recovery. Archive log yang masih dipakai adalah archived log sejak terakhir online (hot) backup. Archive log sebelum hot backup sudah tidak dipakai lagi.

Kesimpulannya, yang dibackup adalah:

1. Control file
2. Datafile
3. Archived log file

11.3. Restore dari off line backup

Restore dari hasil offline backup adalah sangat sederhana, sesederhana backup-nya. Lihat offline backup di sini http://rohmad.net/…/off-line-backup-database-oracle/

Berikut ini step-step restore dengan memakai instance yang sama.

1. Siapkan file-file yang akan direstore
   File-file tersebut adalah control, log, data, dan temp file

2. Pastikan instance sudah mati
```
SQL> shutdown immediate;
```

3. Restore file-file backup ke directory asalnya
```
$ cp /backup/ts/control01.ctl /oradata/ts/control01.ctl
$ cp /backup/ts/control02.ctl /oradata/ts/control02.ctl
$ cp /backup/ts/control03.ctl /oradata/ts/control03.ctl
$ cp /backup/ts/redo04.log /oradata/ts/redo04.log
$ cp /backup/ts/redo05.log /oradata/ts/redo05.log
$ cp /backup/ts/redo06.log /oradata/ts/redo06.log
$ cp /backup/ts/sysaux01.dbf /oradata/ts/sysaux01.dbf
$ cp /backup/ts/system01.dbf /oradata/ts/system01.dbf
$ cp /backup/ts/temp01.dbf /oradata/ts/temp01.dbf
$ cp /backup/ts/undotbs01.dbf /oradata/ts/undotbs01.dbf
$ cp /backup/ts/users01.dbf /oradata/ts/users01.dbf
```

4. Nyalakan database
```
SQL> startup;
```

Jika karena suatu hal, kita tidak bisa merestore ke direktori asalnya, maka kita bisa merestore ke tempat (direktori) lain. Step 1 dan 2 masih seperti yang di atas. Step 3 dan seterusnya adalah berikut ini:

3. Restore ke directory baru
   Control file
```
$ cp /backup/ts/control01.ctl /newdir/ts/control01.ctl
$ cp /backup/ts/control02.ctl /newdir/ts/control02.ctl
$ cp /backup/ts/control03.ctl /newdir/ts/control03.ctl
```

   Log file
```
$ cp /backup/ts/redo04.log /newdir/ts/redo04.log
$ cp /backup/ts/redo05.log /newdir/ts/redo05.log
$ cp /backup/ts/redo06.log /newdir/ts/redo06.log
```

   Data file
```
$ cp /backup/ts/sysaux01.dbf /newdir/ts/sysaux01.dbf
$ cp /backup/ts/system01.dbf /newdir/ts/system01.dbf
$ cp /backup/ts/undotbs01.dbf /newdir/ts/undotbs01.dbf
$ cp /backup/ts/users01.dbf /newdir/ts/users01.dbfTemp file
$ cp /backup/ts/temp01.dbf /newdir/ts/temp01.dbf
```

4. Ubah konfigurasi control file. Edit init (instance parameter) file.
   Filenya di $ORACLE_HOME/dbs/init[NAMAINSTANCE].ora
   Ganti lokasi control file dari yang lama ke yang baru.

Value yang lama:

```
control_files='/oradata/ts/control01.ctl',
'/oradata/ts/control02.ctl','/oradata/ts/control03.ctl'
```

Value yang baru:

```
control_files='/newdir/ts/control01.ctl',
'/newdir/ts/control02.ctl','/newdir/ts/control03.ctl'
```

5. Ubah konfigurasi file yang lainnya (log, data, dan temp file)

```
SQL> startup mount

SQL> alter database rename file '/oradata/ts/redo04.log' to
'/newdir/ts/redo04.log';

SQL> alter database rename file '/oradata/ts/sysaux01.dbf' to
'/newdir/ts/sysaux01.dbf';

SQL> alter database rename file '/oradata/ts/temp01.dbf' to
'/newdir/ts/temp01.dbf';

dan seterusnya ...
```

6. Open database

```
SQL> alter database open;
```

Error yang terkait

1. Jika control file tidak ada, atau ada tapi direktorinya berubah dan init file belum diedit

```
SQL> startup

ORACLE instance started.Total System Global Area 1610612736 bytes
Fixed Size 2177912 bytes
Variable Size 396149896 bytes
Database Buffers 1207959552 bytes
Redo Buffers 4325376 bytes

ORA-00205: error in identifying control file, check alert log for
more info
```

2. Jika ada data file yang kelewatan, atau ada tapi direktorinya berubah dan belum di-alter/rename

```
SQL> startup

ORACLE instance started.Total System Global Area 1610612736 bytes
Fixed Size 2177912 bytes
Variable Size 396149896 bytes
Database Buffers 1207959552 bytes
Redo Buffers 4325376 bytes
Database mounted.

ORA-01157: cannot identify/lock data file 4 - see DBWR trace file
ORA-01110: data file 4: '/oradata/ts/users01.dbf'
```

11.4. Restore dan Recovery dari online backup

Sebaiknya silahkan dibaca dulu online backup di Online (hot) backup database Oracle dan tentang archived log di Men-setting database menjadi archivelog mode. Secara sederhana, kedua artikel tersebut membahas landasan teorinya.

Restore dari online backup adalah sama persis dengan restore dari offline backup, kecuali ada 1 tambahan step setelah melakukan restore dari online backup, yaitu kita WAJIB melakukan **recovery**. Lihat secara detail tentang restore dari offline backup di Restore dari off line backup

Ringkasan restore dari offline backup:

1. Shutdown database
   ```
   SQL> shutdown immediate
   ```
2. Lakukan restore semua file-file yang bersangkutan
3. Startup database
   ```
   SQL> startup
   ```

Sedangkan untuk restore dan recovery dari online backup adalah sbb:

1. Shutdown database
   ```
   SQL> shutdown immediate
   ```
2. Lakukan restore semua file-file yang bersangkutan
   Restore juga archived lognya
3. Startup mount database
   ```
   SQL> startup mount
   ```
4. Recover database
   ```
   SQL> recover database using BACKUP CONTROLFILE;
   ```
   Nanti akan diminta memasukkan archive log. Bila archived log sudah di-restore ke lokasinya, pilih AUTO. Setelah semua archived log di-apply, dan database masih minta archived log lagi, pilih CANCEL.
5. Open database resetlogs.
   ```
   SQL> alter database open resetlogs;
   ```

Catatan-catatan:

1. Bila file-file (data, control, dan log file) direstore ke tempat yang berbeda dari aslinya, lakukan step-step untuk mengubah konfigurasi file-file tersebut seperti yang sudah di bahas di "Restore dari offline backup".
2. Ingat, control file yang digunakan untuk menaikkan *restored* database adalah control file hasil dari "alter database backup controlfile to '/backupdir/backupcontrol_22042008.bak';"

11.5. Oracle Flashback Technology (Recycle bin)

Salah satu feature Oracle 10g (ke atas) yang jadi andalan saya adalah Oracle Flashback Technology. Dengan Oracle Flashback, kita bisa me-restore (recovery) data dengan sangat mudah.

Sebagai contoh, ketika kita secara tidak sengaja menghapus sebagian record di file Excel, untuk mengembalikannya, kita cukup klik tombol UNDO. Kita klik UNDO sekian kali untuk kembali ke kondisi data yang kita inginkan. Ketika kita tidak sengaka menghapus (drop/delete) file, untuk mengembalikan file tersebut kita cukup me-restore-nya dari recycle bin.

Nah, sederhanya, Oracle Flashback Technology menyediakan feature seperi UNDO dan RECYCLE BIN di Windows. Enak, bukan?

Macam-macam Oracle flashback:

1. Oracle Flashback Query.
Berguna untuk melihat (query) isi tabel di masa lalu. Misalkan siang ini kita baru saja men-delete atau update record. Karena terkanjur commit, kita tidak bisa melakukan rollback. Dengan Oracle Flashback Query, kita bisa melihat record (isi tabel) tadi pagi sebelum kita delete atau update. Berikut ini contoh command-nya (Catatan: nama tabel yang saya pakai untuk contoh di artikel ini adalah TB):

```
SQL> SELECT * FROM TB AS OF TIMESTAMP
     TO_TIMESTAMP('2009-06-03 06:08:03', 'yyyy-mm-dd hh24:mi:ss');
SQL> SELECT * FROM TB AS OF TIMESTAMP
     TO_TIMESTAMP('2009-06-03 06:08:03', 'yyyy-mm-dd hh24:mi:ss')
     where nomor=3;
```

Keterbatasan:

1. Karena feature ini memanfaatkan UNDO segment di UNDO tablespace, maka size dari UNDO tablespace sangat mempengaruhi sampai berapa lama masa lalu yang bisa dikembalikan lagi. Semakin besar UNDO tablespace maka semakin besar (lama) masa lalu yang bisa dikembalikan.Bila data yang diquery sudah tidak ada lagi di UNDO tablespace maka akan muncul error berikut:

   ```
   ORA-08180: no snapshot found based on specified time
   ```

2. Batas yang bisa query adalah setelah operasi DDL (data definition language) terakhir. Contoh DDL adalah mengubah definisi tabel (alter table, add column, alter column, truncate, dll). Bila data yang di-query ada pada waktu sebelum DDL terakhir (last DDL) maka akan muncul error berikut:

   ```
   ORA-01466: unable to read data - table definition has changed
   ```

2. Oracle Flashback Table.
Berguna untuk mengembalikan kondisi (isi) tabel seperti kondisi di masa lalu. Seperti contoh

di atas, kita bisa mengembalikan tabel TB sebagimana tadi pagi sebelum kita melakukan delete atau update record. Berikut ini perintahnya:

```
SQL> FLASHBACK TABLE TB TO TIMESTAMP
     TO_TIMESTAMP('2009-06-03 06:08:03', 'yyyy-mm-dd hh24:mi:ss');
```

Syaratnya: “ROW MOVEMENT” harus di-enable. Berikut ini command-nya:

```
SQL> ALTER TABLE TB ENABLE ROW MOVEMENT;
```

Kalau tidak di-enable, bila menjalankan command FLASHBACK TABLE akan muncul error message berikut:

```
ORA-08189: cannot flashback the table because row movement is not enabled
```

Keterbatasan: sama dengan Oracle Flashback Query.

User yang tidak punya ROLE DBA, agar bisa melakukan flashback harus mempunyai privilege FLASHBACK ANY TABLE. Contoh memberi privilege ke user EMP:

```
SQL> grant FLASHBACK ANY TABLE to EMP;
```

3. Oracle Flashback Drop.
Berguna untuk mengembalikan tabel yang telah di-drop. Kalau di Windows adalah restore file dari recycle bin. Command untuk melihat isi dari recycle bin:

```
SQL> select * from dba_recyclebin;
SQL> select * from user_recyclebin;
SQL> select * from recyclebin;
```

Restore tabel:

```
SQL> FLASHBACK TABLE TB TO BEFORE DROP;
```

Kita juga bisa me-restore dan mengubah nama tabel tersebut:

```
SQL> FLASHBACK TABLE TB TO BEFORE DROP RENAME TO TB_OLD;
```

Keterbatasan:
Setelah (misalkan) tabel TB di-drop, secara physic data masih ada di tablespace, tidak dihapus, hanya diberi tanda (flag) bahwa space yang dipakai oleh tabel tersebut sewaktu-waktu bisa dihapus dan dipakai untuk yang lain. Ketika space kosong di tablespace sudah habis, sementara dibutuhkan space lagi untuk data yang baru masuk, maka space dari tabel TB tersebut akan dibersikan dan siap dipakai untuk data baru. Jadi, sampai berapa lama tabel akan disimpan di recycle bin? Ya tergantung ketersediaan free space di tablespace yang bersangkutan.

Catatan:
Berkaitan dengan truncate, dalam beberapa hal saya menghindari truncate dan lebih memilih drop table. Karena truncate adalah DDL maka kita tidak bisa mengembalikan data (table) pada kondisi sebelum truncate. Sementara itu drop table bisa di-restore kembali oleh Oracle Flashback Drop.

4. Oracle Flashback Database.

Merupakan alternatif lain dari database Point-In-Time Recovery.
Kalau sempat akan saya bahas di lain kesempatan.

Referensi

Oracle® Database Backup and Recovery Basics 10g Release 2 (10.2)

12. Data guard:

12.1. Dataguard 10g: Membuat Physical Standby DB (1)

Dataguard adalah solusi HIGH AVAILABILITY dari Oracle. Tujuannya adalah untuk membuat database standby (secondary/tambahan) agar bila sewaktu-waktu database production (primary) mati maka database standby itu bisa menggantikan posisi menjadi production.

Setiap ada perubahan di primary, maka standby segera diupdate, sebisa mungkin diusahakan agar delta (perbedaan data antara primary dan standby) kecil. Mekanisme update standby adalah dengan apply archived log. Archived log dikirim dari primary ke standby, kemudian di apply di standby.

Berdasarkan mekanisme apply archived log, database standby ada dua tipe:

1. Physical standby database.
   Archived log diapply secara konvensional (by block per block), sebagaimana kalau kita melakukan recovery database dengan archived log.
2. Logical standby database
   Dari Archived log diextract SQL statement-nya, kemudian SQL statement itu di apply

Artikel ini berisi contoh membuat Physical standby database. Dalam contoh ini, struktur directory (untuk semua file Oracle) adalah sama persis antara database primary dan standby. Contoh ini menggunakan environment sebagai berikut:

1. Nama instance dan database: ts
2. Versi database: Oracle Database 10g Enterprise Edition Release 2
3. OS: SunOS 5.10

Secara umum langkah-langkah ini sama untuk semua database 10 Release 2 di Operating System manapun baik Windows maupun Unix (Sun Solaris, IBM AIX, HP UX, Linux, dan lain-lain). Untuk instalasi versi lainnya (8i, 9i, dan 10g) silahkan lihat masing-masing dokumentasinya, caranya tidak berbeda jauh dengan ini.

PERSIAPAN DI PRIMARY DATABASE

Sebelum membuat standby database, persiapkan dulu segala persyaratan (environment) di database primary.

1. Apply FORCE LOGGING
   ```
   SQL> ALTER DATABASE FORCE LOGGING;
   ```
2. Database harus sudah archived log
3. Membuat password file
   Cek apakah password file sudah ada. Di Windows lokasinya di **%ORACLE_HOME%\database**, biasanya berformat **PWD[namainstance].ora**; contoh instance dengan nama DATAKU mempunyai password file **PWDdataku.ora.** Di UNIX lokasinya di **$ORACLE_HOME/dbs**, biasanya berformat **orapw[namainstance]**; contoh instance dengan nama ts mempunyai password file **orapwts**. Kalau belum ada password file (atau anda ingin membuat ulang), buatlah

dengan command **orapwd**. Ini adalah perintah bawaan Oracle, lokasinya standart yaitu di **$ORACLE_HOME/bin**.

Berikut ini contoh membuat password file:

```
cd $ORACLE_HOME/dbs
orapwd file=orapwts password=oracle entries=10 force=y
```

4. Persiapkan initial parameter (cukup disebut init). Di Windows, init file ada di **%ORACLE_HOME%\database\init[namainstance].ora**. Di Unix ada di **$ORACLE_HOME/dbs/init[namainstance].ora.** Berikut ini init file yang berkaitan dengan dataguard (standby database)
   - `log_file_name_convert = '/oradata/oracle/ts/'`
   - `remote_login_passwordfile='EXCLUSIVE'`
   - `log_archive_config='DG_CONFIG=(tsprimary,tsstandby)'`
   - `log_archive_dest_1='LOCATION=/oradata/oracle/ts/arc'`
   - `log_archive_dest_state_1=enable`
   - `log_archive_dest_state_2=enable`
   - `log_archive_format=%s_%t_%r.arc`
   - `fal_client='tsstandby'`
   - `fal_server='tsprimary'`
   - `log_archive_dest_2='service=tsstandby optional LGWR ASYNC NOAFFIRM valid_for=(online_logfiles,primary_role) db_unique_name=ts'`

   Parameter lainnya biarkan sebagaimana adanya. Sebagai contoh, berikut ini adalah parameter-parameter dari instance yang saya pakai. Tampak settingannya sangat minimalis (banyak memakai nilai default), gak pa-pa, yang penting bisa digunakan untuk contoh.
   - `audit_file_dest='/data1/oracle/admin/ts/adump'`
   - `background_dump_dest='/data1/oracle/admin/ts/bdump'`
   - `core_dump_dest='/data1/oracle/admin/ts/cdump'`
   - `user_dump_dest='/data1/oracle/admin/ts/udump'`
   - `control_files='/oradata/oracle/ts/control01.ctl', '/oradata/oracle/ts/control02.ctl', '/oradata/oracle/ts/control03.ctl'`
   - `compatible='10.2.0.3.0'`
   - `db_block_size=8192`
   - `db_domain=''`
   - `db_name='ts'`
   - `pga_aggregate_target=209715200`
   - `sga_target=1610612736`
   - `undo_management='AUTO'`
   - `undo_tablespace='UNDOTBS1'`

5. Buat Oracle Net Service Name (TNS Names), masing-masing untuk primary dan standby database. Bisa dengan **netca** ,atau langsung menambah entry berikut file **$ORACLE_HOME/network/admin/tnsnames.ora**:

```
tsprimary =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = 10.21.106.161)(PORT = 1521))
)
(CONNECT_DATA =
(SID = ts)
)
)
```

```
tsstandby =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = 10.21.75.200)(PORT = 1521))
)
(CONNECT_DATA =
(SID = ts)
)
)
```

PERSIAPAN Di STANDBY DATABASE

1. Persiapkan init file di %ORACLE_HOME%\database\initts.ora (untuk Windows) atau $ORACLE_HOME/db/initts.ora (di Unix). Isinya sama persis dengan yang di primary, kecuali untuk entry berikut
   ```
   log_archive_dest_2='service=tsprimary optional LGWR ASYNC NOAFFIRM
   valid_for=(online_logfiles,primary_role) db_unique_name=ts'
   ```

2. Buat password file. Password harus sama dengan database primary. Jadi, command-nya samakan dengan command untuk membuat password file primary database. Atau cara lain, copy (ftp) password file dari primary.
3. Buat service (instance). Ini khusus di Windows, karena di Unix tidak perlu. Gunakan oradim, ini tool standar Oracle.
   ```
   cd %ORACLE_HOME%\database
   oradim -NEW -SID ts
   ```

   Perintah di atas akan membaca file initts.ora yang telah di buat. Untuk melihat bahwa instance ts sudah ter-create, lihat di tool "Services"-nya Windows, atau coba login dengan SQL Plus
   ```
   sqlplus '/ as sysdba'
   SQL*Plus: Release 10.2.0.3.0 - Production on Mon Jun 9 13:13:06 2008
   Copyright (c) 1982, 2006, Oracle. All Rights Reserved.
   Enter password:
   Connected to an idle instance.
   SQL>
   ```

4. Buat Oracle Net Service Name (TNS Names), sama persis seperti di primary database, yaitu **tsprimary** dan **tsstandby**
5. Buat direktory untuk file-file dump dan database
   ```
   $ mkdir /data1/oracle/admin/ts/adump
   $ mkdir /data1/oracle/admin/ts/bdump
   $ mkdir /data1/oracle/admin/ts/cdump
   $ mkdir /data1/oracle/admin/ts/udump
   $ mkdir /oradata/oracle/ts/
   ```

Referensi:
Oracle® Data Guard Concepts and Administration, 10g Release 2 (10.2)

12.2. Dataguard 10g: Membuat Physical Standby DB (2)

Setelah persiapan di mesin primary dan standby selesai, selanjutnya tinggal membuat standby database. Berikut ini langkah-langkah (step-step) nya:

1. Di primary database, buat standby control file
```
SQL> alter database create standby controlfile
as '/oradata/oracle/ts/controltsstandby.ctl';
```

2. Di primary database, lakukan backup full database.
   Bisa secara online ataupun offline. Dalam contoh ini saya menggunakan metode online (hot) backup biar database production tidak perlu mati.List daftar tablespace yang bukan TEMPORARY
```
SQL> select TABLESPACE_NAME from dba_tablespaces
      where CONTENTS <>'TEMPORARY';

NAME
------------------------------
SYSTEM
UNDOTBS1
SYSAUX
USERS
```

   Jalankan perintah BEGIN BACKUP
```
SQL> alter tablespace SYSTEM begin backup;
SQL> alter tablespace UNDOTBS1 begin backup;
SQL> alter tablespace SYSAUX begin backup;
SQL> alter tablespace USERS begin backup;
```

   Lihat semua file yang perlu dibackup (datafile, tempfile, dan logfile):
```
SQL> select name as file_name from
(select name from v$tempfile union
select name from v$datafile union
select member as name from v$logfile)
order by file_name;

FILE_NAME
--------------------------------------------------
/oradata/oracle/ts/redo04.log
/oradata/oracle/ts/redo05.log
/oradata/oracle/ts/redo06.log
/oradata/oracle/ts/sysaux01.dbf
/oradata/oracle/ts/system01.dbf
/oradata/oracle/ts/temp01.dbf2
/oradata/oracle/ts/undotbs01.dbf
/oradata/oracle/ts/users01.dbf2
```

   File-file tersebut bisa dibackup di TAPE, directory temporary, ataupun langsung ditaruh (ftp) di mesin standby.
   Setelah file-file dibackup, Jalankan perintah END BACKUP
```
SQL> alter tablespace SYSTEM end backup;
SQL> alter tablespace UNDOTBS1 end backup;
SQL> alter tablespace SYSAUX end backup;
SQL> alter tablespace USERS end backup;
```

3. Restore semua file ke mesin standby, taruh di directory yang sama
   Dalam contoh ini lokasi semua file adalah sama, yaitu /oradata/oracle/ts/
   ```
   ls -la /oradata/oracle/ts/

   controltsstandby.ctl
   redo04.log
   redo05.log
   redo06.log
   sysaux01.dbf
   system01.dbf
   temp01.dbf2
   undotbs01.dbf
   users01.dbf2
   ```

4. Di mesian standby, siapkan control file. Copy control file hasil dari "alter database create standby controlfile" ke directory control file (sebagimana yang ditunjuk dalam file init)
   ```
   cd /oradata/oracle/ts/
   cp -rp controltsstandby.ctl control01.ctl
   cp -rp controltsstandby.ctl control02.ctl
   cp -rp controltsstandby.ctl control03.ctl
   ```

5. Naikkan database standby
   ```
   SQL> startup mount;
   ```

6. Jalankan recovery di standby database untuk meng-apply arhived log
   ```
   SQL> alter database recover managed standby database disconnect;
   ```

Akhirnya standby database selesai di-create. Untuk melihat archived log yang telah di-apply di standby database gunakan command ini

```
SQL> set pages 100
SQL> col name for a50
SQL> select name,to_char(FIRST_TIME,'dd-mon-yy hh24:mi:ss') TIME
,SEQUENCE#,APPLIED from v$archived_log;
```

Pastikan colomn APPLIED bernilai YES.

Referensi:
Oracle® Data Guard Concepts and Administration, 10g Release 2 (10.2)

12.3. Dataguard 10g: Administrasi Physical Standby DB

Setelah semua persiapan beres dan standby database selesai dibuat, sekarang kita coba administrasinya.

MONITOR ARCHIVED LOG DI STANDBY DATABASE

Di standby database, lihat archived log yang sudah dan belum diapply:

```
SQL> set lines 120
SQL> col name for a50
SQL> select name,to_char(FIRST_TIME,'dd-mon-yy hh24:mi:ss') TIME
,SEQUENCE#,APPLIED from v$archived_log order by sequence#;
```

Misalkan hasilnya berikut ini (kolom TIME tidak ditampilkan, untuk menghemat space)

```
NAME---------------------- SEQUENCE#------ APPLIED
================================================
/oradata/oracle/ts/arc106_1_655805448.arc 106 YES
/oradata/oracle/ts/arc107_1_655805448.arc 107 YES
/oradata/oracle/ts/arc108_1_655805448.arc 108 YES
/oradata/oracle/ts/arc110_1_655805448.arc 110 NO
/oradata/oracle/ts/arc111_1_655805448.arc 111 NO
/oradata/oracle/ts/arc112_1_655805448.arc 112 NO
/oradata/oracle/ts/arc113_1_655805448.arc 113 NO
/oradata/oracle/ts/arc114_1_655805448.arc 114 NO
```

Berdasarkan hasil query di atas, pastikan bahwa:

1. **Sequence harus urut**.
   Kalau ada sequence yang tidak urut (atau ada gap), itu berarti ada archived log yang tidak terkirim, alasannya bisa bermacam-macam, misalnya karena network error atau standby database sempat mati. Pada contoh di atas, ada gap pada sequence 109. Kalau seperti itu:
   a. Archived log harus ada (dicopy manual dari database primary, atau di-restore dari backup)
   b. Register archived yang baru saja di-restore itu
      ```
      SQL> ALTER DATABASE REGISTER
      LOGFILE '/oradata/oracle/ts/arc109_1_655805448.arc';
      ```

      Setelah diregister, archived log sequence 109 harusnya sudah masuk kedalam list di view v$archived_log

2. **Kolom APPLIED harus bernilai YES**, artinya archived log sudah di-apply
   Archived log belum di-apply itu ada 2 kemungkinan:
   a. Ada GAP. Contoh di atas SEQUENCE# 110 belum di-apply karena ada gap di sequence atasnya (SEQUENCE# 109). Solusinya sudah disebut di atas
   b. Proses recovery berhenti. Solusinya, jalankan lagi
      ```
      SQL> alter database recover managed standby database
      disconnect;
      ```

STARTUP DAN SHUTDOWN

1. startup
   ```
   SQL> STARTUP MOUNT;
   ```
   Setelah itu, apply archived log secara background
   ```
   SQL> alter database recover managed standby database disconnect;
   ```

2. Shutdown. Matikan dulu proses recovery-nya
   ```
   SQL> alter database recover managed standby database cancel;
   ```

   Setelah itu, baru shutdown
   ```
   SQL> SHUTDOWN IMMEDIATE;
   ```

Mode Recovery dan Open Read Only

1. standby database bisa dibuat Open (hanya READ ONLY, tidak bisa READ WRITE) sehingga kita bisa melakukan query. Biasanya ini dilakukan untuk tujuan reporting. Untuk menjadilkan mode open read only, matikan dulu proses recoverynya
   ```
   SQL> alter database recover managed standby database cancel;
   ```

   Setelah itu, alter database open
   ```
   SQL> alter database open read only;
   ```

2. setelah selesai dengan query reporting, kita bisa mengembalikan database ke mode recovery lagi.
   ```
   SQL> alter database close;
   SQL> alter database recover managed standby database disconnect;
   ```

Referensi:
Oracle® Data Guard Concepts and Administration, 10g Release 2 (10.2) - Managing a Physical Standby Database

12.4. Dataguard 10g: Switchover Physical Standby DB

Tulisan ini merupakan lanjutan dari serial Dataguard 10g. Tulisan terdahulu telah membahas tentang persiapannya, cara membuatnya, me-manage-nya, dan ada juga contoh kasus kalau kita kehilangan archived log. Sekarang saya akan membahas switchover (role transition) physical standby database.

Role transition, perubahan fungsi dari PRIMARY ke STANDBY dan sebaliknya, ada dua macam yaitu:

1. **Switchover**
   Sesuai dengan istilahnya, ini adalah men-switch (mengubah) peran dari database STANDBY menjadi PRIMARY dan database PRIMARY menjadi STANDBY. Biasanya dilakukan kalau kita ingin melakukan maintenance pada mesin PRIMARY, misalnya menambah memory atau CPU yang memerlukan downtime.
2. **Failover**
   Bila karena suatu hal tiba-tiba database PRIMARY mati, maka kita tidak bisa melakukan switchover karena database PRIMARY keburu mati duluan. Yang bisa kita lakukan adalah 'memaksa' database STANDBY untuk menjadi PRIMARY. Istilah 'memaksa' inilah yang kira-kira lebih pas untuk menjelaskan fail over.

Langkah-langkah melakukan Switch Over pada Physical Standby DB

Dalam contoh ini, database PRIMARY ada di MESIN A dan database STANDBY ada di MESIN B. Berikut ini persiapan yang mesti dilakukan:

1. Di database PRIMARY, close semua transaksi dan user session
2. Database PRIMARY harus dalam keadaan OPEN

```
SQL> select DATABASE_ROLE from v$database;

DATABASE_ROLE
----------------
PRIMARY

SQL> select open_mode from v$database;

OPEN_MODE
----------
READ WRITE
```

Database STANDBY harus dalam keadaan MOUNT

```
SQL> select DATABASE_ROLE from v$database;

DATABASE_ROLE
----------------
PHYSICAL STANDBY

SQL> select open_mode from v$database;

OPEN_MODE
----------
MOUNTED
```

3. Verify bahwa switch over siap dilakukan. Jalankan query ini di kedua database PRIMARY dan STANDBY
```
SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE;
```

   Kondisi paling bagus (paling siap dilakukan switchover) adalah kalau kolom SWITCHOVER_STATUS di database PRIMARY nilainya "TO STANDBY" dan di database STANDBY nilainya "TO PRIMARY"

   Namun lebih sering kolom SWITCHOVER_STATUS nilainya "SESSIONS ACTIVE". Inipun gak masalah, switchover bisa dilanjutkan.

   Lebih detail tentang kolom SWITCHOVER_STATUS di view V$DATABASE, silahkan lihat di sini Oracle® Database Reference - 10g Release 2 (10.2) - V$DATABASE

Sekarang kita menuju step utamanya

1. **Di database PRIMARY (MESIN A)**
   Lakukan switchover dengan perintah berikut
```
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PHYSICAL STANDBY;
```

   Kalau masih ada user yang aktif, akan muncul error berikut
```
ORA-01093: ALTER DATABASE CLOSE only permitted
with no sessions connected
```

   Solusinya adalah: close (kill) semua user session atau tambahkan parameter WITH SESSION SHUTDOWN sehingga command-nya menjadi:
```
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PHYSICAL STANDBY WITH
SESSION SHUTDOWN;
```

   Setelah itu shutdown dan startup mount.
```
SQL> SHUTDOWN IMMEDIATE;
SQL> STARTUP MOUNT;
```

   Sekarang verifikasi bahwa database PRIMARY telah beralih menjadi STANDBY
```
SQL> select DATABASE_ROLE from v$database;

DATABASE_ROLE
----------------
PHYSICAL STANDBY
```

2. **Di database STANDBY (MESIN B)**
   Sekarang, database STANDBY kita switchover menjadi PRIMARY. Jalankan command berikut:
```
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
```

   Mungkin kita akan menjumpai error seperti ini:
```
ORA-16139: media recovery required
```

   Kalau menjumpai error tersebut, jalankan "recover managed standby database" kemudian jalankan switchover:
```
SQL> recover managed standby database;
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
```

   Setelah switchover berhasil, kemudian open database:
```
SQL> alter database open;
```

```
ORA-01507: database not mounted
```

Oo… Error. Ternyata tadinya STANDBY database (sejak last startup) pernah saya "OPEN READ ONLY", jadinya error kayak gitu. Kalau tidak pernah di-"OPEN READ ONLY" harusnya tidak error. OK, kalo gitu, solusinya: restart database.

```
SQL> SHUTDOWN IMMEDIATE;
SQL> startup
```

Dah berhasil. Sekarang verify bahwa STANDBY database telah menjadi PRIMARY

```
SQL> select DATABASE_ROLE from v$database;

DATABASE_ROLE
----------------
PRIMARY
```

Referensi

Oracle® Data Guard Concepts and Administration - 10g Release 2 (10.2)
Role Transitions

12.5. Dataguard 10g: Solusi Archived Log Hilang

Karena masalah jaringan, beberapa waktu yang lalu archived log di primary database tidak terkirim ke standby database (hal seperti ini sering saya alami). Kadang-kadang karena kesibukan, saya tahunya setelah beberapa hari, jadi mesti mengejar untuk apply archived log yang tertinggal itu.

Policy saya, archived log di production saya **compress (zip), backup ke tape, dan hapus** setiap hari. Archived log di-keep di backup sehingga tidak banyak memakai banyak space di production (primary).

Jadi, mau gak mau saya mesti me-restore backup archived log untuk di-apply di mesin standby:

- Restore archived log ke mesin standby
- Uncompress archived log (yang di-compress sebelum backup)
- Register: **ALTER DATABASE REGISTER LOGFILE '…';**
- Recover : **alter database recover managed standby database;**

Kadang-kadang recover gagal karena archived log corrupt.

```
ORA-00308: cannot open archived log '/.../_30383.arc'
ORA-27046: file size is not a multiple of logical block size
Recovery interrupted!
```

Kalau begitu biasanya archived log (sequence 30383) tersebut saya coba restore dan recovery ulang. Sering berhasil. Kadang juga masih tidak bisa. Gimana dong? Ada dua solusi:

- Recreate standby database lagi
- Recover dengan RMAN

Recreate standby database lagi, wah… perlu effort yang besar. Akhirnya saya pilih option kedua, recover dengan RMAN. Berikut ini step-step Recovery dengan RMAN

1. Di standby database, lihat SCN (system change number) terakhir
   ```
   SQL> col CURRENT_SCN for 9999999999999999999
   SQL> select current_scn from v$database;
   ```
   Misalkan saya dapatkan current SCN 2254678976792

2. Di primary, lakukan incremental backup sejak sequence 2254678976792. Backup bisa dilakukan di disk ataupun tape. Dalam dontoh ini, backup ke disk.
   ```
   mkdir /tmp/rman_bkup
   rman target /
   ```

   ```
   RMAN> BACKUP DEVICE TYPE DISK INCREMENTAL FROM SCN
   2254678976792 DATABASE FORMAT '/tmp/rman_bkup/bkup_%U';
   ```

3. Restore hasil backup ke mesin standby. Dalam contoh ini saya restore ke /tmp/restore_rman/. Kemudian jalankan RMAN di standby
   ```
   rman target /
   RMAN> CATALOG START WITH '/tmp/restore_rman/';
   ```

4. Kemudian jalankan recover di standby database
   ```
   RMAN> RECOVER DATABASE NOREDO;
   ```

Kadang error seperti ini

```
ORA-01219: database not open: queries allowed on fixed tables/views only
```

Solusinya, restart database, dan jalankan RMAN lagi

```
SQL> shutdown immediate
SQL> startup mount
RMAN> exit
rman target /
RMAN> RECOVER DATABASE NOREDO;
```

5. Sampai di sini, database standby sudah sinkron dengan primary, jadi archived log sequence 30383 sudah tidak dibutuhkan lagi. Namun, tampaknya standby database masih butuh archived log sequence 30383 tersebut ketika saya menjalan recover standby database seperti biasa:

```
SQL> alter database recover managed standby database;
ORA-00308: cannot open archived log '/.../_30383.arc'
ORA-27046: file size is not a multiple of logical block size
Recovery interrupted!
```

6. Ya… ternyata standby control file harus dicreate ulang
   Di database primary, buat standby control file

```
SQL> alter database create
standby controlfile as '/tmp/standby.ctl';
```

   Copy contol file **/tmp/standby.ctl** ke mesin standby.
   Shutdown standby database.
   Delete (rename) control file yang lama, dan ganti dengan standby control file yang baru dibuat tersebut:

```
mv control01.ctl control01.ctl.bak
mv control02.ctl control02.ctl.bak
mv control03.ctl control03.ctl.bak
cp -rp standby.ctl control01.ctl
cp -rp standby.ctl control02.ctl
cp -rp standby.ctl control03.ctl
```

   Kemudian startup dan kembali recover

```
SQL> startup mount
SQL> alter database recover managed standby database;
```

7. OO… Berhasil

Referensi:

Oracle® Database Backup and Recovery Advanced User's Guide
10g Release 2 (10.2)

13. Performance Tuning:

13.1. Dasar-dasar Tuning

Berikut ini pertanyaan-pertanyaan seputar tuning:

1. Bagian mana yang perlu dituning. Siapa yang melakukan?
2. Dari sisi aplikasi, apanya yang perlu dituning?
3. Bagian database yang mana yang perlu dituning?
4. Darimana kita bisa belajar banyak tentang tuning?

Bagian mana yang perlu dituning? Oleh siapa?

1. Aplikasi, oleh application designer dan developer
2. Database, oleh database administrator (DBA)
3. System Operasi (OS), oleh system administrator (sysadmin)

Tuning Aplikasi

Tuning aplikasi memberikan kontribusi tersebar terhadap performa sistem secara keseluruhan. Termasuk bagian-bagian tuning aplikasi adalah:

1. Normalisasi yang tepat
2. Pemakaian index
3. Pemakaian hint dalam query SQL atau PL/SQL
4. Pemanfaatan sharing cursor
5. dan lain-lain

Bagian database yang mana yang perlu dituning?

1. Memory -> PGA dan SGA SGA (shared pool, large pool, buffer cache, redo log buffer, sort area size)
2. I/O -> distributing I/O, striping, multiple DBWn processes, DBWn I/O slaves.
3. CPU -> CPU utilization.
4. Network
5. Space management -> extent allocation dan Oracle block efficiency.
6. Redo log dan checkpoint -> redo log file configuration, redo entries, dan checkpoint.
7. Rollback segment -> sizing rollback segments. Mulai versi 9i, banyakbagian yang dibuat automatic
8. dan lain-lain

Darimana kita bisa belajar banyak tentang tuning?

1. Oracle8i Designing and Tuning for Performance Release 2 (8.1.6)
2. Oracle9i Database Performance Planning Release 2 (9.2)
3. Oracle9i Database Performance Tuning Guide and Reference Release 2 (9.2)
4. Oracle® Database 2 Day + Performance Tuning Guide 10g Release 2 (10.2)
5. Oracle® Database Performance Tuning Guide 10g Release 2 (10.2)
6. Oracle® Database 2 Day + Performance Tuning Guide 11g Release 1 (11.1)
7. Oracle® Database Performance Tuning Guide 11g Release 1 (11.1)

13.2. Tuning Query dengan Explain Plan

Suatu proses (query) sebelum dijalankan, database Oracle menentukan dulu mana langkah-langkah yang paling optimal (efektif dan efesien) yang akan dipilih. Contoh query yang melibatkan 5 tabel, paling tidak ada 1×2x3×4x5 pilihan langkah (execution plan) tabel-table mana yang akan di-joint terlebih dahulu. Urutan join tentu saja menentukan resource (cost) yang akan dipakai.

Untuk database dengan query yang kecil, tuning query dengan explain plan mungkin tidak begitu kelihatan manfaatnya. Namun untuk query yang melibatkan data besar-besaran, wow… benar-benar terasa.

Sebelum menjalankan query, kita bisa melihat "execution plan" mana yang akan dipilih oleh Oracle. Caranya adalah dengan menjalankan "explain plan". Untuk dapat memanfaatkan feature explain plan ini, berikut langkah-langkahnya:

1. Pastikan bahwa instance parameter OPTIMIZER_MODE tidak sama dengan RULE. (Pilihan value untuk OPTIMIZER_MODE adalah rule, choose, all_rows, first_rows , first_rows_n). Kalau nilainya RULE, maka Oracle tidak akan menentukan execution plan berdasarkan cost-nya, tapi berdasarkan aturan (rule) default-nya Oracle.
2. Jalankan script utlxplan.sql untuk membuat **table plan**. Ini dijalankan satu kali saja oleh user yang akan melakukan Explain Plan.
   ```
   SQL> @?/rdbms/admin/utlxplan.sql
   ```

3. Berikut ini contoh command untuk membuat plan dari suatu query
   ```
   SQL> explain plan for
   select * from b where owner='ROHMAD'
   union select * from c where owner='ROHMAD';
   ```

4. Setelah itu, lihat execution plan-nya
   Di Oracle 8i
   ```
   SQL> @?/rdbms/admin/utlxpls
   ```

   Di Oracle 9i ke atas
   ```
   SQL> select * from table(dbms_xplan.display);
   ```

Contoh kasus. Saya punya tabel A, B, dan C di schema TEST yang strukturnya sama persis

- Tabel A yang berisi data TABEL dan INDEX dari database.
- Tabel B yang berisi data TABEL dari database.
- Tabel C yang berisi data INDEX dari database.

Jadi content (isi) tabel A adalah sama dengan content table B ditambah content tabel C. Masing-masing tabel punya index untuk kolom OWNER. Kalau saya ingin query data dengan OWNER='ROHMAD', mana yang lebih cepat?

1. query di A saja?
   ```
   SQL> select * from a where owner='ROHMAD';
   ```
2. atau query di tabel B kemudian di UNION (gabung) dengan query di tabel C?
   ```
   SQL> select * from b where owner='ROHMAD' union select * from c where
   owner='ROHMAD';
   ```

Untuk mengetahuinya, kita perlu membuat explain plan untuk kedua pilihan query di atas.

1. Buat ketiga tabel dan index contoh tersebut

```
SQL> create table a as select * from dba_objects
where OBJECT_TYPE in ('TABLE','INDEX');
SQL> create table b as select * from dba_objects
where OBJECT_TYPE in ('TABLE');
SQL> create table c as select * from dba_objects
where OBJECT_TYPE in ('INDEX');
SQL> create index a_owner on a (owner);
SQL> create index b_owner on b (owner);
SQL> create index c_owner on c (owner);
```

   Kemudian buat statistiknya (gather statistic)

```
SQL> exec DBMS_STATS.GATHER_TABLE_STATS(OWNNAME=> 'TEST', TABNAME =>
'A', CASCADE => TRUE, ESTIMATE_PERCENT => 5 , METHOD_OPT => 'FOR ALL
COLUMNS SIZE 1', DEGREE => 4, GRANULARITY=> 'DEFAULT');

SQL> exec DBMS_STATS.GATHER_TABLE_STATS(OWNNAME=> 'TEST', TABNAME =>
'B', CASCADE => TRUE, ESTIMATE_PERCENT => 5 , METHOD_OPT => 'FOR ALL
COLUMNS SIZE 1', DEGREE => 4, GRANULARITY=> 'DEFAULT');

SQL> exec DBMS_STATS.GATHER_TABLE_STATS(OWNNAME=> 'TEST', TABNAME =>
'C', CASCADE => TRUE, ESTIMATE_PERCENT => 5 , METHOD_OPT => 'FOR ALL
COLUMNS SIZE 1', DEGREE => 4, GRANULARITY=> 'DEFAULT');
```

2. Jalankan explain plan untuk query pertama

```
SQL> explain plan for
select * from a where owner='ROHMAD';
```

   Lihat execution plan-nya

```
SQL> set lines 120
SQL> select * from table(dbms_xplan.display);
```

   Jalankan explain plan untuk query kedua

```
SQL> explain plan for
select * from b where owner='ROHMAD' union select * from c
where owner='ROHMAD';
```

   Lihat execution plan-nya

```
SQL> set lines 120
SQL> select * from table(dbms_xplan.display);
```

3. Bandingkan kedua execution plan tersebut. OO… ternyata query kedua lebih besar cost-nya dibandingkan query pertama. Perintah **union** ternyata menambah pekerjaan tambahan yaitu **SORT UNIQUE**.

Kesimpulannya, pilih query pertama yang cost-nya lebih kecil

Referensi

Oracle9i Database Performance Tuning Guide and Reference

13.3. Tuning Query dengan SQL Trace dan tkprof

Beberapa waktu yang lalu saya membahas tuning query dengan explain plan. Dengan explain plan kita bisa tahu (meng-estimate) **nantinya** query kita itu memakai "execution plan" yang mana. Sementara dengan SQL trace kita bisa mengetahui query yang **sedang** berjalan ini menggunakan "execution plan" yang mana. Jadi "explain plan" adalah untuk meramalkan, sedangkan "sql trace" untuk melihat kejadian yang sesungguhnya.

Kelebihan SQL trace adalah SQL trace menampilkan informasi yang lebih banyak. Lebih detail tentang informasi yang bisa digali dari sql trace, silahkan lihat referensi di akhir tulisan ini. Berikut ini langkah-langkah (step-step) untuk mengaktifkan SQL trace:

1. Pastikan bahwa instance parameter TIMED_STATISTICS=true.
2. Aktifkan instance parameter sql_trace=true. Kita cukup lakukan di level session saja.
   ```
   SQL> alter session set sql_trace=true;
   ```
3. Jalankan query yang akan dianalisa.
   ```
   SQL> select * from b where owner='ROHMAD'
   union select * from c where owner='ROHMAD';
   ```
4. setelah selesai, disable sql_trace.
   ```
   SQL> alter session set sql_trace=false;
   ```
5. Hasil trace ditaruh di directory udump. Untuk melihat lokasi udump, gunakan command ini (pakai user yang punya role dba)
   ```
   SQL> sho parameter user_dump_dest
   ```

   File trace yang dihasilkan berformat **namainstance_ora_OSID.trc**. Dalam contoh ini nama instance adalah ts. File yang dihasilkan adalah **ts_ora_14662.trc**. OSID bisa diperoleh dengan command berikut:
   ```
   SQL> select a.SPID from v$process a, v$session b
   where a.addr=b.paddr and
   b.username='nama_user_yang_menjalankan_sql_trace';
   ```

   Mungin kita tidak perlu report-report mencari OSID, kalau file-file di derektori udump tidak banyak, kita mungkin bisa langsung menemukan file trace tersebut.
6. Untuk membaca file trace dengan format yang user fiendly, gunakan tool **tkprof**. Berikut ini contohnya
   ```
   cd lokasi_direkroty_user_dump_dest
   tkprof ts_ora_14662.trc output=ts_ora_14662.trc.log
   ```

   Hasil yang diformat ditaruh di file yang ditunjukkan oleh parameter output, yaitu ts_ora_14662.trc.log.

Setelah dapat trace file (hasil tkprof), lihat bagian "execution plan". Biasanya yang paling penting adalah kalau ada "full table scan", nah kita bisa mencoba-coba gimana sih kalau pakai index.

Pengalaman saya, sebagian besar porsi tuning query adalah:
- Menemukan bagian mana yang melakukan full table scan

- Memakai (membuat) index yang berkaitan dengan query
- OO… ternyata dengan memakai index, query jadi jauh lebih cepat

Referensi:
http://download.oracle.com/docs/cd/B10501_01/server.920/a96533/sqltrace.htm

13.4. Gather statistic untuk Performance

Cost based optimizer (CBO) menggunakan statistic untuk menentukan execution plan yang paling optimal. Saya pernah membahas sekilas tentang CBO ini di artikel Tuning Query dengan Explain Plan.

Di versi 9i ke bawah, gather statistic dilakukan secara manual dengan package DBMS_STATS. By default di database Oracle 10g, gather statistic ini dilakukan oleh Oracle secara otomatis; selanjutnya kitapun bisa memilih cara melakukan gather statistic ini, apakah secara otomatis atau manual.

Agar Database melakukan gather statistic secara manual, jalankan command berikut di SQL:

```
conn system
BEGIN
  DBMS_SCHEDULER.DISABLE('GATHER_STATS_JOB');
END;
/
```

Untuk mengembalikan gather statistic berjalan otomatic, jalankan command berikut di SQL:

```
BEGIN
  DBMS_SCHEDULER.ENABLE('GATHER_STATS_JOB');
END;
/
```

Untuk mempermudah pekerjaan, saya menyarankan untuk menjalankan gather statistic secara otomatis. Untuk database versi 9i, mau tidak mau kita harus melakukan secara manual karena di 9i belum ada *feature* gather statistic otomatis. Walaupun manual, kita bisa mensiasatinya dengan menjalankan DBMS_STATS lewat DBMS_JOB.

Menggunakan DBMS_STATS

Berikut ini procedure yang ada di dalam package DBMS_STATS yang sering digunakan:

```
Procedure                      Collects
=============================================================================
GATHER_INDEX_STATS             Index statistics
GATHER_TABLE_STATS             Table, column, and index statistics
GATHER_SCHEMA_STATS            Statistics for all objects in a schema
GATHER_DATABASE_STATS          Statistics for all objects in a database
GATHER_SYSTEM_STATS            CPU and I/O statistics for the system
```

Untuk menjalankan GATHER_INDEX_STATS, GATHER_TABLE_STATS, dan GATHER_SCHEMA_STATS bisa dilakukan oleh user (schema) yang bersangkutan. Kalau saya lebih suka memakai user SYSTEM, untuk mempermudah saja.

Contoh:

1. Gather satistic untuk semua object milik schema DBMON,sample yang diambil statistic 20%.
   ```
   DBMS_STATS.GATHER_SCHEMA_STATS('DBMON',20);
   ```

Kalau ingin statistic benar-benar valid, kita bisa mengambil sampel 100%. Namun bila tabelnya sangat besar, dan panjang data tiap kolomnya seragam, sebaiknya sampel bisa diperkecil. Sampel yang besar akan memakan waktu gather statistic lebih lama. Mulai versi 9i, berapa besarnya sampel ini bisa kita serahkan ke Oracle; gunakan DBMS_STATS.AUTO_SAMPLE_SIZE.

```
DBMS_STATS.GATHER_SCHEMA_STATS('DBMON',DBMS_STATS.AUTO_SAMPLE_SIZE);
```

2. Gather statistic tabel TAB_DATAHIST_XLDM milik schema DBMON, partisi NULL karena tidak mempunyai partisi, sample yang diambil statistic adalah 20%, index juga diikutsertakan

```
DBMS_STATS.GATHER_TABLE_STATS('DBMON','TAB_DATAHIST_XLDM',null,20,cascade=>TRUE);
```

3. Contoh lebih lanjut dan pembahasan lebih detail, silahkan lihat di referensi

Beberapa informasi statistic tersebut bisa dilihat di view DBA_TABLES dan DBA_INDEXES.

Referensi

- Oracle® Database Performance Tuning Guide 10g Release 2 (10.2) - Managing Optimizer Statistics
- Oracle® Database PL/SQL Packages and Types Reference 10g Release 2 (10.2) - DBMS_STATS

14. Troubleshooting

14.1. Mendeteksi lock

Kadang transaksi (update atau delete) berjalan lama banget,padahal biasanya dua detik juga selesai. Atau juga mau nglakuin DDL (misalnya aler table) tidak bisa dengan error “ORA-00054: resource busy and acquire with NOWAIT specified”. Nah, kalau begini pasti ada session yang me-lock object (table, index).

Gunakan SQL command berikut untuk melihat locking di database Oracle kita

```
set linesize    150
set pages       100

col name                for a21         head "Locked Object"
col session_id          for 99999       head SID
col serial#             for 99999       head SER#
col oracle_username     for a12         head "Locking User"
col lock_type           for a12         head "Lock Type"
col mode_held           for a12         head "Mode Held"
col event               for a30

SELECT a.session_id, b.serial#, a.oracle_username, c.name,
decode(d.type,
'MR', 'Media Recovery',
'RT', 'Redo Thread',
'UN', 'User Name',
'TX', 'Transaction',
'TM', 'DML',
'UL', 'PL/SQL User Lock',
'DX', 'Distrib Xaction',
'CF', 'Control File',
'IS', 'Instance State',
'FS', 'File Set',
'IR', 'Instance Recovery',
'ST', 'Disk Space Transaction',
'TS', 'Temp Segment',
'IV', 'Library Cache Invalidation',
'LS', 'Log Start or Switch',
'RW', 'Row Wait',
'SQ', 'Sequence Number',
'TE', 'Extend Table',
'TT', 'Temp Table',
d.type) lock_type,
decode(d.lmode,
0, 'None',           /* Mon Lock equivalent */
1, 'Null',           /* N */
2, 'Row-S (SS)',     /* L */
3, 'Row-X (SX)',     /* R */
4, 'Share',          /* S */
5, 'S/Row-X (SSX)',  /* C */
6, 'Exclusive',      /* X */
to_char(d.lmode)) mode_held,
e.event, e.SECONDS_IN_WAIT "Wait(Seconds)"
FROM    sys.obj$ c, v$session b, v$locked_object a,
sys.v_$lock d, v$session_wait e
WHERE   a.session_id=b.sid
```

```
AND     b.sid=e.sid
AND     c.obj#=a.object_id
AND     a.object_id=d.id1
AND     b.sid=d.sid
order   by e.SECONDS_IN_WAIT desc
;
```

Misalkan hasilnya berikut ini:

SID	SER#	Locking User	Locked Object	Lock Type	Mode Held	EVENT	Wait(Seconds)
1558	45822	APPUSR1	CST_CM_TRXLOG_TRACK	DML	Row-X (SX)	SQL*Net message from client	231
2151	449	APPUSR1	RM_RESOURCE_STK_HIST	DML	Row-X (SX)	direct path read temp	34
2151	449	APPUSR1	CUSTOMER	DML	Row-X (SX)	SQL*Net message from client	9
3453	26576	APPUSR1	RM_RESOURCE_STK_HIST	DML	Row-X (SX)	SQL*Net message from client	1
4185	40934	APPUSR1	VM1_TENT_VCH	DML	Row-X (SX)	db file sequential read	1
3453	26576	APPUSR1	RM_RESOURCE_STOCK	DML	Row-X (SX)	SQL*Net message from client	0
4185	40934	APPUSR3	VM1_TENT_VCH_HISTORY	DML	Row-X (SX)	db file sequential read	0
4185	40934	APPUSR2	VM1_TENT_VCH_HISTORY	DML	Row-X (SX)	db file sequential read	0
4185	40934	APPUSR2	VM1_TENT_VCH	DML	Row-X (SX)	db file sequential read	0
3453	26576	APPUSR2	MLOG$_RM_RESOURCE_STO	DML	Row-X (SX)	SQL*Net message from client	0

Penjelasan:

1. EVENT: Aktivitas session (yang melakukan lock) saat ini.
   Wait(Seconds): Lamanya aktivitas saat ini
2. User APPUSR1 (SID 1558) tengah me-lock tabel CST_CM_TRXLOG_TRACK. Dari keterangan kolom EVENT adalah “SQL*Net message from client”, ini berarti user sedang idle (INACTIVE). Kalau begitu, tadi user APPUSR1 tersebut pernah melakukan transaksi (insert, update, atau delete) pada tabel CST_CM_TRXLOG_TRACK namun sampai sekarang belum COMMIT atau ROLLBACK

Aktivitas lock di Oracle adalah hal yang biasa. Sebagai DBA kita harus bisa mengetahui mana-mana session yang sedang menunggu (wait) karena tabel yang di-update sedang di-lock oleh session lain. Berikut ini perintah SQL untuk melihat session yang melakukan lock dan yang menunggu:

```
SELECT a.sid "Locking Sid (yang nge-lock)",
b.sid "Locked SID (Sedang Menunggu)"
FROM v$lock a , v$lock b
WHERE a.id1=b.id1
AND   a.id2=b.id2
AND   a.request=0
AND   b.lmode=0
;
```

Msalkan hasilnya berikut ini:

```
Locking Sid (yang nge-lock) Locked SID (Sedang Menunggu)
--------------------------- ----------------------------
1558                         2223
```

Gara-gara user APPUS1 (SID 1558) belum melakukan commit atau rollback, session dengan SID 2223 terpaksa harus menunggu. Kalau begitu apa solusinya? User APPUS1 tersebut harus segera melakukan COMMIT atau ROLLBACK, kalau terpaksa ya mungkin bisa di-kill.

14.2. Startup Inconsistent Database

Sebagai DBA kita kadang dipusingkan dengan masalah yang disebabkan oleh file corrupt. Masalah ini bisa disebabkan oleh database yang mati tiba-tiba karena listrik mati atau masalah di storage (disk). Misalnya database tidak bisa startup dengan error berikut

```
SQL> startup

ORACLE instance started.Total System Global Area 1610612736 bytes
Fixed Size 2177912 bytes
Variable Size 396149896 bytes
Database Buffers 1207959552 bytes
Redo Buffers 4325376 bytes
Database mounted.

ORA-01113: file 2 needs media recovery
ORA-01110: data file 2: '/oradata/oracle/ts/undotbs01.dbf'
```

Biasanya database tidak bisa startup kalau yang perlu di-recovery adalah file UNDO atau SYSTEM. [Dalam contoh di atas adalah file UNDO]. Dalam kasus ini baik yang bermasalah file SYSTEM atau UNDO, perlakuan (work around atau solusi sementara) nya adalah sama.

Lakukan recover database

```
SQL> recover database;

ORA-00279: change 7516226638 generated at 05/05/2008 12:32:11
needed for thread 1
ORA-00289: suggestion : /oradata/oracle/ts/arc/1_42_653916655.dbf
ORA-00280: change 7516226638 for thread 1 is in sequence #42

Specify log: {ret=suggested | filename | AUTO | CANCEL}
AUTO

ORA-00308: cannot open archived log
'/oradata/oracle/ts/arc/1_42_653916655.dbf'
ORA-27037: unable to obtain file status
SVR4 Error: 2: No such file or directory
Additional information: 3
```

Open database

```
SQL> alter database open;

alter database open
*
ERROR at line 1:
ORA-01113: file 2 needs media recovery
ORA-01110: data file 2: '/oradata/oracle/ts/undotbs01.dbf'
```

Mau tidak mau kita butuh archived log dan/atau logfile yang berisi sequence yang dibutuhkan. Bila archived log atau logfile (yang dibutuhkan) tidak tersedia karena corrupt (atau hilang) maka sampai kapanpun database tidak akan bisa dinaikkan.

Berikut ini adalah work around (solusi sementara) untuk menaikkan database. Database nantinya akan up dengan kondisi inconsistent (tidak konsisten). Kalau database berisi data yang sangat urgent, ini mungkin terasa tidak masalah, yang penting database bisa up dan data bisa diselamatkan.

Work Around

- Edit Init file
  Remark parameter
  `undo_management`
  dan
  `undo_tablespace`

  Tambahkan parameter berikut:
  ```
  UNDO_MANAGEMENT=MANUAL
  _ALLOW_RESETLOGS_CORRUPTION = TRUE
  _ALLOW_ERROR_SIMULATION = TRUE
  _CORRUPTED_ROLLBACK_SEGMENTS =(_SYSSMU1,_SYSSMU2,_SYSSMU3, ...)
  ```

  Dapatkan value _CORRUPTED_ROLLBACK_SEGMENTS dari [OS] command berikut
  ```
  cd system_datafile_directory
  strings system01.dbf | grep _SYSSMU | cut -d $ -f 1 | sort -u
  ```

- Matikan database, dan nyalakan dengan initfile yang telah diedit tersebut. Pfile HARUS disebutkan secara eksplisit di command startup.
  ```
  SQL> startup mount
  pfile='/data1/oracle/product/10.2.0/dbs/initts.ora';

  ORACLE instance started.
  Total System Global Area 1610612736 bytes
  Fixed Size 2177912 bytes
  Variable Size 396149896 bytes
  Database Buffers 1207959552 bytes
  Redo Buffers 4325376 bytes
  Database mounted.
  ```

- Recover database until cancel
  ```
  SQL> recover database until cancel;

  ORA-00279: change 7516226638 generated at 05/05/2008 12:32:11
  needed for thread 1
  ORA-00289: suggestion :
  /oradata/oracle/ts/arc/1_42_653916655.dbf
  ORA-00280: change 7516226638 for thread 1 is in sequence #42

  Specify log: {ret=suggested | filename | AUTO | CANCEL}
  CANCEL

  ORA-01547: warning: RECOVER succeeded but OPEN RESETLOGS would get
  error below
  ORA-01194: file 1 needs more recovery to be consistent
  ORA-01110: data file 1: '/oradata/oracle/ts/system01.dbf'
  ORA-01112: media recovery not started
  ```

- Open database
  ```
  SQL> ALTER DATABASE OPEN RESETLOGS;
  ```

15. Membangun Karir sebagai DBA:

15.1. Cara Belajar Database Oracle secara Otodidak

Oracle itu sangat royal, terlalu gampang membagi knowledge. Kita bisa nge-download software Oracle RDBMS dan dokumentasinya (yang berisi bejibun e-book) secara gratis. Tentunya itu bukan semata-mata untuk meng-educate user (dan calon user), yang pasti intinya adalah bagaimana Oracle bisa dikenal banyak orang. Setelah itu orang memakai Oracle. Dan kemudian orang membayar lisence Oracle 😀

Berikut ini kiat-kiat belajar Database Oracle secara Otodidak. Saya mengambil contoh untuk yang versi 10g,versi-versi yang lainnya intinya sih sama saja.

I. Membaca dokumentasi Oracle.
Anda bisa membaca secara online di website Oracle. Anda pun bisa nge-download-nya dulu, kemudian membacanya secara offline di PC anda.

Berikut ini daftar dokumentasi database Oracle:

1. Untuk semua versi (8, 8i, 9i, 10g, dan 11g)
   http://www.oracle.com/technology/documentation/index.html
2. Source dokumentasi 10g yang bisa di download
   http://download.oracle.com/docs/cds/B19306_01.zip
3. Dokumentasi yang 10g yang bisa dibaca online
   http://www.oracle.com/pls/db102/homepage

Dari mana mulai membacanya? Saya merekomendasikan untuk membaca tema-tema berikut, sebaiknya dibaca secara berurutan:

1. *Oracle® Database Concepts*
2. *Oacle® Database 2 Day DBA*
3. *Oracle® Database Administrator's Guide*
4. *Oracle® Database 2 Day + Performance Tuning Guide*
5. *Oracle® Database Performance Tuning Guide*
6. *Oracle® Database Backup and Recovery Basics*
7. *Oracle® Database Backup and Recovery Quick Start Guide*
8. *Oracle® Database Backup and Recovery Advanced User's Guide*

II. Install Software, dan Create Database Oracle di PC anda

Dapatkan source database Oracle di sini
http://www.oracle.com/technology/software/products/database/index.html

Selanjutnya, buat database dan langsung praktekkan apa yang sedang (dan telah) anda pelajari. Gimana, mudah sekali, bukan?

15.2. Kiat Mempersiapkan Diri Jadi DBA Oracle

Banyak yang bertanya pada saya, terutama teman-teman yang mau lulus kuliah, "Gimana kiat-kiat mempersiapkan diri untuk jadi DBA Oracle. Dengan kata lain, Gimana sih caranya menggapai cita-cita jadi DBA Oracle?"

Saran saya:

1. Belajar sendiri Database Oracle
2. Ambil Training Database Oracle
3. Ambil sertifikat Database Oracle
4. Ikut milis dan forum Oracle di internet

Belajar sendiri Database Oracle
Belajar sendiri juga tidak terlalu susah, bagi yang mau. Untuk guide awalnya, saya pernah menulis Cara Belajar Database Oracle secara Otodidak. Saya juga pernah menulis Petunjuk Memahami Database Oracle berdasarkan tulisan-tulisan di blog ini. Daftar Artikel di blog ini juga saya susun berdasarkan kategori-kategori pokok database Oracle.

Ambil Training Database Oracle
Ada beberapa alasan kenapa kita mengambil training:

1. Mempercepat menguasai materi database Oracle
2. Males belajar dan explorasi sendiri. Emang paling enak dikasih tahu daripada nyari tahu sendiri
3. Pernah mengambil training (apalagi yang bersertifikat) akan semakin mempercantik CV. Tetep, di mata penerima kerja, ada point lebih bagi yang pernah ngambil training database Oracle. Paling tidak ada bukti bahwa kita pernah belajar database Oracle.
4. Tentu saja, karena punya dana yang cukup

Lebih lanjut tentang training Database Oracle saya bahas di Strategi Mengambil Training Database Oracle

Ambil sertifikat Database Oracle

Mempunyai Sertifikat adalah sebuah keutamaan (bukan keharusan), apalagi bekerja sebagai DBA di perusahaan konsultan (vendor). Sertifikat menunjukkan bahwa pemegangnya mempunyai pengetahuan yang mendalam terhadap database Oracle. Dengan pengetahuan dan pengalaman yang sama, pemegang sertifikat mempunyai lebih banyak peluang untuk mendapatkan pekerjaan dan gaji yang lebih tinggi.

Sekilas tentang sertifikasi database Oracle saya bahas di Sertifikasi Database Oracle. Juga lebih detail tentang Oracle Certified Associate (OCA) dan Oracle Certified Professional (OCP) saya bahas di Ujian OCA dan OCP Database Oracle 10g

Ikut milis dan forum Oracle di internet
Ikutlah milis dan forum di internet. Banyak ilmu yang bertebaran di sana. Contoh milis yang bagus adalah indo-oracle@yahoogroups.com. Boleh juga join di forum indo-oracletech.com.

15.3. Strategi Mengambil Training Database Oracle

Banyak yang bertanya pada saya, "Pak, apa saja materi training yang ada? Materi apa aja yang perlu saya ambil duluan? Trus, bagusnya saya ambil training database versi berapa ya? Lantas, di mana tempat trainingnya? …"

Database yang paling banyak digunakan saat ini adalah 9i dan 10g. Versi 8i sudah obsolete (Oracle sudah tidak mensupport lagi); masih banyak dipakai karena application specific (aplikasi yang dipakai hanya certified untuk database 8i), juga masih dipakai karena sudah ada dari dulu dan tidak ada urgensi untuk di-upgrade. Sementara versi 11g untuk saat ini (Jul 2008) masih relatif baru.

Walaupun versi Oracle yang masih dipakai ada banyak, secara fundamental esensinya sama saja. Secara umum, garis-garis besar materi database Oracle Adalah:

1. Oracle SQL dan PL/SQL
2. Administration
3. Backup and Recovery
4. Oracle Network
5. Performance Tuning
6. Product specific (Dataguard, RAC, Datawarehouse, dll)

Untuk mengetahui feature-feature yang baru dan yang obsolete tiap versi, silahkankan lihat di sini: 10g Release 2, 9i Release 2, dan 11g.

Materi (modul) training yang disarankan

Sebaiknya mengambil training database versi berapa? Bagi yang sudah bekerja, mungkin menyesuaikan dengan versi (environment) database yang saat ini dimaintain. Bagi yang sedang mempersiapkan diri menjadi DBA, saya sarankan yang versi 10g.

Untuk materi training, saya menyarankan berdasarkan garis-garis besar materi Oracle dan materi ujian sertifikasi OCA (Oracle Certified Associate) dan OCP (Oracle Certified Professional).

Untuk versi 9i, berikut ini modul training yang saya sarankan:

1. Introduction to Oracle9i: SQL
2. Oracle9i Database Administration Fundamentals I
   Materi: Administrasi dasar
3. Oracle9i Database Administration Fundamentals II
   Materi: Oracle Network dan Backup & Recovery
4. Oracle9i Database Performance Tuning

Berikut ini saran modul training untuk versi 10g:

1. Introduction to Oracle10g: SQL
2. Oracle Database 10g: Administration Workshop I Release 2
   Materi: Administrasi, Backup & Recovery, dan Tuning dasar. Dibahas juga Oracle Network
3. Oracle Database 10g: Administration Workshop II Release 2
   Materi: Administrasi , Backup & Recovery, dan Tuning lanjutan

Untuk Oracle 11g, parallel dengan Oracle 10g, hanya beda versinya saja

1. Oracle Database 11g: SQL Fundamentals I
2. Oracle Database 11g: Administration I
   Materi: Administrasi, Backup & Recovery, dan Tuning dasar. Dibahas juga Oracle Network
3. Oracle Database 11g: Administration II
   Materi: Administrasi , Backup & Recovery, dan Tuning lanjutan

Untuk tiap materi (modul), lama training 5 hari kerja. Di Oracle Indonesia, harga training tiap modul di atas adalah US$ 1050 (hampir RP 10 juta). Kalau kita mengambil training dengan peserta banyak (misal 3 orang atau lebih), biasanya kita bisa minta diskon. Kadang-kadang kalau lagi ada promo, bisa murah banget. Apalagi kalo promo buat mahasiswa, bisa super murah. Ya, murah-murahnya sih tetep juta-jutaan juga.

Kalau punya anggaran banyak, bisa mengambil training yang lainnya, misalnya SQL Advanced, PL/SQL, ataupun Oracle Product specific.

Mahal sekali ya. Tentu saja kita tidak harus mengambil semua modul. Kita-kita yang 'cekak' dokunya, mungkin cukup training **Database Administration I** saja. Yang lainnya mungkin bisa belajar sendiri.

Nah, yang enak kalau training dibiayai perusahaan. apalagi kalo kerja di perusahaan konsultan, ujian sertifikasipun dibiayai perusahaan; karena perusahaan konsultan butuh untuk jualan jasanya.

Untuk melihat daftar lengkap training database Oracle, silahkan lihat di education.oracle.com, pilih **Database and Grids**.

Ujian Sertifikasi

Setelah mengambil training, kita bisa sekalian Ujian sertifikasi. Harga ujian tiap modul adalah sekitar US$ 125 (sekitar Rp 1 jutaan). Kalau ujian gagal dan mau ngulang ya mesti mbayar lagi.

1. Untuk versi 9i
   Materi ujian OCA: modul 1 dan 2; materi ujian OCP: modul 3 dan 4
2. Untuk versi 10g
   Materi OCA: modul 2; materi ujian OCP: modul 3. Modul 1 (SQL) tidak menjadi syarat OCA. Namun per 1 Desember 2008 nanti Ujian SQL akan menjadi syarat OCA 10g.
3. Untuk versi 11g
   Materi OCA: modul 1 dan 2; materi OCP: modul 3

Tempat Training

Tempat training yang saya tahu hanya yang di daerah Jakarta. Yang di luar Jakarta saya tidak begitu tahu. Menurut pengalaman saya, harga training di Oracle Indonesia relatif lebih mahal.

Mungkin karena menyandang nama “Oracle” sehingga berani menjual mahal, tentu saja ya fasilitasnya memang bagus. Tentang kualitas training, saya kira sih tergantung pengajarnya.

Berikut ini daftar tempat training Database Oracle di Jakarta:

1. **Oracle Indonesia**
   Sentral Senayan I, Lt. 9. Jl. Asia Afrika No. 8, Jakarta 10270
   Contanct person:
   Anna Cahyawati - 021 5723222 ext 1077, anna.cahyawati@oracle.com
   Yudi Adicawarman - 0215723222 ext 1022, yudi.adicawarman@oracle.com
2. **Mitra Integrasi Informatika (MII)**
   Wisma Metropolitan I lt. 9
   Komp. Metropolitan
   Jl. Jend. Sudirman Kav 29 - 31 Jakarta 12920
   Tel. 021 2511360, Fax. 021 5705988
   Contact Person:
   Agung Sujatmiko ext. 2460 agung.sujatmiko@metrodata.co.id
   Reni Heryani ext. 2435 reni.heryani@metrodata.co.id
   Susilowati ext. 2437 Susilowati@metrodata.co.id
3. **Sisindokom**
   Graha Sisindokom
   Jl. Panataran No. 2, Pegangsaan
   Jakarta Pusat 10320
   Tel. 021 31900711, Fax. 021 3140358
   Contact Email: inquiry@sisindokom.com , ccare@sisindokom.com
4. **Brainmatics**
   Menara Bidakara, Lantai 2, Suite 0205
   Jl. Jend. Gatot Subroto Kav. 71-73, Jakarta 12870
   Telp. +62-21-83793383; Fax. +62-21-83793384
5. **Asaba**
   Jl. Setiabudi Selatan No.1
   Jakarta 12920
   Phone. :(+62-21) 5799 4700 Fax.: (+62-21) 5790 4551
   E-Mail : information: info@asaba.co.id
6. **Inixindo**
   Permata Senayan Blok E2-E5
   Jl. Tentara Pelajar No.5, Jakarta 12210
   Phone : (021) 579 40868 Fax : (021) 579 40869
   Email : sales@inixindo.co.id
7. **Dan lain-lain**

15.4. Sertifikasi Database Oracle

Database Management System (DBMS), atau cukup disebut database, yang saat ini paling banyak digunakan oleh enterprise (perusahaan besar) adalah Oracle. Indikator paling gampang adalah lowongan pekerjaan Database Administrator (DBA) Oracle adalah yang paling banyak diantara lowongan DBA yang ada. Coba saja *search* vacancy di jobsdb.com dengan keyword "database" atau "DBA", yang paling banyak muncul [hampir] pasti DBA Oracle.

Dengan banyaknya vacancy DBA Oracle itu, tentu menambah peluang bagi para pencari kerja. Berikut ini adalah hal-hal yang perlu dimiliki oleh para pencari kerja tsb:

- Pemahaman konsep database Oracle (wajib)
- Pengalaman sebagai DBA (biasanya wajib, tidak wajib bagi entry level)
- Training database Oracle (keutamaan)
- Sertifikat DBA Oracle (keutamaan)

Mempunyai Sertifikat adalah sebuah keutamaan, apalagi bekerja sebagai DBA di perusahaan konsultan (vendor). Sertifikat menunjukkan bahwa pemegangnya mempunyai pengetahuan yang mendalam terhadap database Oracle. Dengan pengetahuan dan pengalaman yang sama, pemegang sertifikat mempunyai lebih banyak peluang untuk mendapatkan pekerjaan dan gaji yang lebih tinggi.

- Oracle mengeluarkan beberapa jenis sertifikat DBA
- Oracle Certified Associate (OCA)
- Oracle Certified Professional (OCP)
- Oracle Certified Master (OCM)

Saat ini, Oracle mengeluarkan sertifikat untuk database versi 9i, 10g, dan 11g. Sertifikat untuk database versi 8i ke bawah sudah tidak berlaku lagi.

Penjelasan tentang OCA dan OCP Database 10g ada di Ujian OCA dan OCP Database Oracle 10g

15.5. Ujian OCA dan OCP Database Oracle 10g

Beberapa waktu yang lalu saya telah menulis artikel tentang sertifikasi database Oracle di Sertifikasi Database Oracle. Sekarang saya akan menulis lebih detail tentang sertifikasi OCA dan OCP 10g.

Ujian-ujian yang terkait adalah:
1. 1Z0-042 (Oracle Database 10g: Administration I)
2. 1Z0-043 (Oracle Database 10g: Administration II)
3. 1Z0-040 (Oracle Database 10g: New Features for Administrators)

Syarat memperoleh OCA 10G adalah lulus ujian 1Z0-042.
Adapun syarat memperoleh OCP 10G adalah:
1. Sudah lulus ujian untuk OCA 10G (1Z0-042)
2. Pernah mengambil training resmi Oracle
3. Lulus ujian 1Z0-043

Bagi yang sudah memegang certifikat OCP 9i, kita bisa melakukan upgrade ke OCP 10g dengan ujian 1Z0-040.

Cara termudah untuk persiapan adalah mengikuti training Oracle. Semua materi ujian tak lepas dari materi yang ada di buku training. Bagi yang tidak sempat training, saya sarankan untuk meminjam buku temannya :). Bagi yang tidak pernah training dan tidak punya teman yang punya bukunya, ya mesti belajar sendiri. Namun tidak perlu terlalu berkecil hati, Oracle menyediakan kisi-kisinya kok. Kisi-kisi tersebut hampir sama dengan daftar isi buku trainingnya Oracle.

Kisi-kisi ujian 1Z0-042 ada di daftar isi modul training “Oracle Database 10g: Administration I”, demikian seterusnya.

***

Detail 1Z0-042
==========
Biaya: $125.00 USD
Lama ujian: 120 menit
Jumlah pertanyaan: 84
Passing grade: 68%
Materi Ujian (kisi-kisi):
- Architecture
- Installing the Oracle Database Software
- Creating an Oracle Database
- Managing the Oracle Instance
- Managing Database Storage Structures
- Administering User Security
- Managing Schema Objects
- Managing Data and Concurrency
- Managing Undo Data
- Implementing Oracle Database Security

- Configuring the Oracle Network Environment
- Proactive Maintenance
- Performance Management
- Backup and Recovery Concepts
- Performing Database Backup
- Performing Database Recovery
- Performing Flashback
- Moving Data

Detail 1Z0-043
==========
Biaya: $125.00 USD
Lama ujian: 90 menit
Jumlah pertanyaan: 70
Passing grade: 70%
Materi ujian (kisi-kisi):
- Using Globalization Support Objectives
- Configuring Recovery Manager
- Recovering from User Errors
- Dealing with Database Corruption
- DBVERIFY
- Automatic Database Management
- Using Recovery Manager
- Recovering from Non-Critical Losses
- Monitoring and Managing Storage
- Automatic Storage Management
- Monitoring and Managing Memory
- Database Recovery
- Flashback Database
- Managing Resources
- Automating Tasks with the Scheduler

Detail 1Z0-040
==========
Biaya: $125.00 USD
Lama ujian: 105 menit
Jumlah pertanyaan: 60
Passing grade: 73%
Materi ujian (kisi-kisi):
- Installation
- Server Configuration
- Load and Unload Data
- Automatic Management
- Manageability Infrastructure
- Application Tuning
- System Resource Management
- Automating Tasks with the Scheduler
- Space Management
- Improved VLDB Support
- Backup and Recovery Enhancements

- Flashback Any Error
- General Storage Enhancement
- Automatic Storage Management
- Maintain Software
- Security
- Miscellaneous New Features

Referensi:

- Daftar training resmi Oracle: http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=112
- Detail 1Z0-042 (kisi-kisi): http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=41&p_exam_id=1Z0_042
- Detail 1Z0-043 (kisi-kisi): http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=41&p_exam_id=1Z0_043
- Detail 1Z0-040 (kisi-kisi): http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=41&p_org_id=15942?=US&p_exam_id=1Z0_040

15.6. Gaji DBA Oracle di Indonesia

Beberapa waktu yang lalu saya ditelpon temen yang kerja di IT consultan. Dia nanya apakah saya ada temen DBA yang masih fresh? DBA fresh maksud dia kira-kira: mengetahui konsep dasar dan administrasi dasar database Oracle. Orang kayak gitu biasanya baru 1 tahun kerja jadi DBA Oracle, dan belum punya sertifikat OCA atau OCP. Nantinya ditempatkan di customer (telco). Gajinya di atas 5 juta perbulan. Kalau perform, gak lama lagi bisa naik gaji.

Tiga tahun yang lalu di perusahaan lama saya (juga IT consultan), orang baru lulus kuliah namun punya sertifikat OCP 9i digaji 3 juta perbulan. Setahun kemudian dia pindah ke perusahaan kompetitor digaji 7 juta perbulan. Satu tahun kemudian dia pindah lagi ke perusahaan kompetitor lain, saya gak tahu dia digaji berapa, pastinya naik.

Beberapa DBA Oracle teman saya yang betah kerja di suatu tempat, gajinya kalau naik tidaklah seberapa. Malahan sering gajinya lebih rendah dibandingkan DBA-DBA yang baru masuk. Ternyata hukum itu tetap berlaku, kalau mau naik gaji tinggi, ya mesti pindah company dulu.

Cerita saya di atas menunjukkan kisaran gaji DBA Oracle di Indonesia (khususnya Jakarta). Silahkan ditarsirkan sendiri-sendiri, itu termasuk gaji yang rendah apa tinggi.

Berdasarkan "Indonesia Salary Guide 2006″ yang dikeluarkan oleh Kelly Services, Gaji DBA (database administrator) sedikit di bawah IT Administrator, Network Administrator, Systems Engineer; dan lebih tinggi dari Analyst Programmer. DBA yang disebut mungkin terlalu general, tidak spesifik ke product tertentu (misalnya Oracle).

Saya tidak tahu keakuratan guide yang dibuat oleh Kelly Services tersebut. Namun yang jelas, berdasarkan sharing pengamalan dengan temen-temen yang kerja di dunia IT, gaji DBA Oracle tidak kalah (dan bersaing) dengan profesi-profesi yang lainnya seperti Network Administrator, Systems Engineer, dan lain-lain.

So? Buat temen-temen yang baru jadi DBA Oracle, atau yang bercita-cita ingin menjadi DBA Oracle someday nanti; ayo bersemangat. Dunia DBA Oracle menanti anda dengan memberi banyak harapan cerah 

16. Artikel Populer tentang Oracle Corporation:

16.1. Oracle Serakah, apa saja diakuisisi

Tanggal 29 April 2008 lalu secara resmi Oracle mengakuisi BEA senilai US$ 8,5 Milyar. Ini semakin mengokohkan Oracle corporation sebagai enterprise software company terbesar di dunia. Oracle yang dulunya hanya menggarap database saja, kini telah melebarkan sayapnya ke semua layer, misalnya Oracle's Fusion middleware software suite (Application Server) , Oracle Applicationd (ERP), dan lain-lain. Tampaknya hanya tinggal OS saja yang belum dimiliki Oracle.

Selama ini BEA dikenal dengan produk andalannya, WebLogic dan Tuxedo. WebLogic merupakan application server untuk aplikasi Java, sementara Tuxedo untuk applikasi non java (C dan lai-lain). Salah satu tujuan akuisisi BEA oleh Oracle adalah, adanya WebLogic akan semakin memperkuat produk Oracle's Fusion middleware software suite.

Oracle memang sangat agresif ('serakah'). Sepanjang tahun 2008 ini (sampai saat ini) sudah ada 3 company yang diakuisisi, yaitu Empirix, Captovation, dan BEA. Tahun 2007 ada 11 akuisisi. Tahun 2006 ada 13 akuisisi. Dan tahun 2005 ada 13 akuisisi.

Berikut ini daftar Company yang telah diakuisis oleh Oracle:

- Tahun 2008
  - BEA
  - Captovation
  - Empirix
- Tahun 2007
  - Agile
  - AppForge
  - Bharosa
  - Bridgestream
  - Hyperion
  - Interlace Systems
  - LODESTAR
  - LogicalApps
  - Moniforce
  - Netsure Telecom Limited
  - Tangosol
- Tahun 2006
  - 360Commerce
  - Demantra
  - HotSip
  - MetaSolv Software
  - Net4Call
  - Portal Software
  - Siebel
  - Sigma Dynamics
  - Sleepycat
  - SPL WorldGroup
  - Stellent

- Sunopsis
- Telephony@Work

- Tahun 2005
  - Context Media
  - G-Log
  - i-flex
  - Innobase
  - Oblix
  - OctetString
  - PeopleSoft
  - ProfitLogic
  - Retek
  - TempoSoft
  - Thor Technologies
  - TimesTen
  - TripleHop

16.2. Oracle juara 1 sebagai implementator aplikasi

Saya dapat kiriman Press Release dari Oracle yang menerangkan bahwa

Firma Analisis Independen Menempatkan Oracle pada Posisi #1 dalam Pasar Software Pengimplementasi Aplikasi di Asia Pasifik (tidak termasuk Jepang)"

Oracle Juga Menunjukkan Peranan BEA yang Diakuisisi Oracle Dalam Strategi Produk Untuk Middleware Generasi Lanjutan

JAKARTA – 21 Juli 2008

- Oracle mengumumkan bahwa IDC, sebuah firma riset dan penasehat pasar TI, telah menempatkat Oracle sebagai pemimpin pasar di Asia Pasifik (kecuali Jepang) dengan pangsa pasar sebesar 30,5 persen dalam pasar software pengimplementasi aplikasi sesuai dengan penghasilan untuk tahun 2007.
- Menurut laporan IDC, Oracle merupakan vendor software pengimplementasi aplikasi dengan pertumbuhan tercepat dari lima vendor terkemuka lainnya di Asia Pasifik (kecuali Jepang) di tahun 2007, mencatat pertumbuhan pendapatan sebesar 38,1 persen dari 2006 sampai 2007.
- Oracle tumbuh lebih cepat dari pertumbuhan keseluruhan pasar.
- Menurut IDC, pasar software pengimplementasi aplikasi tumbuh 23,7 persen di tahun 2007 untuk Asia Pasifik (kecuali Jepang) sebesar $665.5 juta.
- Pada 1 Juli 2008, dalam webcast langsung yang dihadiri oleh mitra dan pelanggan dari seluruh dunia, Oracle President Charles Phillips dan Oracle Fusion Middleware Senior Vice President Thomas Kurian menguraikan secara singkat strategi middleware Oracle, yang menyatukan produk terbaik dari BEA dan Oracle® Fusion Middleware dan menghadirkan kesinambungan dan perlindungan modal untuk pelanggan dua produk ini.
- Oracle berencana menghadirkan fungsi dan kemampuan yang meningkat di setiap portfolio produk mereka yang semakin meluas.
- Oracle menunjukkan rencana untuk prduk strategis BEA dan Oracle Fusion Middleware – termasuk rencana untuk Application Server (www.oracle.com/appserver), Tuxedo (www.oracle.com/tuxedo), BPM (www.oracle.com/bpm), WebCenter (www.oracle.com/webcenter), SOA (www.oracle.com/soa), dan Development Tools (www.oracle.com/tools). (Untuk informasi lebih lanjut dan daftar produk lengkap kunjungi: www.oracle.com/goto/july1)
- Oracle juga berencana mempertahankan dan memperluas ekosistem mitra – termasuk System Integrators, Independent Software Vendors, Value-Added Distributors dan Value-Added Resellers – untuk membantu mempermudah pengimplementasian Oracle Fusion Middleware.
- Oracle Indonesia juga mengadakan Oracle Developer Day pada 11 Juni 2008 untuk menerangkan teknologi dari Oracle seperti Oracle Fusion Middleware dan Oracle Database 11g kepada para mitra dan pelanggan. Tampak Blair Layton, Senior Manager, Developer Program, Oracle APAC, berbicara di depan para partisipan Oracle Developer Day 2008.

Tentang Oracle Fusion Middleware

Oracle Fusion Middleware adalah jajaran produk yang komprehensif, terintegrasi awal, dan hot-pluggable yang dipakai oleh lebih dari 77.000 pelanggan di seluruh dunia di setiap industri. Jajaran produk infrastruktur middleware yang komprehensif dan berbasis standard – dari server aplikasi Java #1 sampai produk manajemen konten dan portal Enterprise 2.0 terkemuka – memberikan solusi teknologi yang diperlukan oleh perusahaan dan sektor publik pada waktu mereka memerlukannya.

Produk Oracle Fusion Middleware terintegrasi awal dengan produknya yang lain dan dengan produk Oracle Database dan Applications memungkinkan biaya kepemilikan yang lebih rendah. Kemampuan hot-pluggable yang unik memungkinkan pelanggan memperluas biaya modal yang telah terpakai dalam lingkungan TI yang heterogen.

Oracle Fusion Middleware didukung oleh sebuah ekosistem yang berjumlah lebih dari 11.000 vendor software independen, value added resellers dan integrator sistem. Untuk informasi lebih lanjut kunjungi http://www.oracle.com/products/middleware/index.html