

## Introduzione ad Oracle

da <http://escher07.altervista.org>

### Generalità

Oracle è database più diffuso al mondo specie nella cosiddetta fascia alta, ovvero come database o datawarehouse server in sistemi gestionali di organizzazioni complesse, con problemi di multilocalizzazione di gestione di dati geografici e così via.

Oracle nasce di per sé come strumento pensato per database grandi (almeno almeno da 1TB in su) e per essere poco *plug and play* ma molto flessibile da configurare. Questa filosofia iniziale (ben visibile nelle versioni fino alla 7) di fatto si è andata via via temperando considerando che il DB in questione ha trovato impiego, più per motivi commerciali che tecnici, anche per applicazioni di fascia media o addirittura bassa : dunque le nuove versioni (specie dalla 9 in poi) sono state dotate sempre di maggiore capacità di autoconfigurarsi (vedasi il ridotto impatto attuale nella scelta dei parametri di memoria) e di richiedere ridotte attività di amministrazione, almeno per fornire prestazioni medie.

Nonostante questa operazione di *downsizing* da un certo punto di vista Oracle, mantiene comunque tuttora la fama di uno strumento molto complesso e tutto sommato tale da scoraggiarne l'autoapprendimento. Di sicuro un qualcosa che ha contribuito a questa immagine sono anche i "sacri testi" che venivano consegnati con le prime versioni : decine di tomi che di sicuro incutevano timore.

A tuttoggi ritengo di poter dire che imparare le funzionalità base di amministrazione di un DB Oracle sia tutto sommato semplice, un po' per la grande quantità di materiale che si può trovare in rete, un po' per l'automatizzazione di cui parlavo in precedenza. E' chiaro che una installazione fatta con consapevolezza oppure un tuning fine, una gestione di repliche o di disaster recovery richiedano sempre il loro percorso : ma credo che per iniziare a lavorare in un ambiente di prova un documentino come il precedente possa bastare. O quasi.

### Come è fatto un DB Oracle

Quando in un modo o nell'altro abbiamo installato Oracle su un server, in realtà a cosa ci stiamo riferendo ? Per capirlo può essere utile tenere conto che Oracle come altri prodotti è detto DBMS (Data Base Management System) o meglio RDBMS, visto che i dati nel DB sono gestiti con il modello di DB relazionale. Intuitivamente dunque possiamo dire che:

#### **RDBMS = dati + processi di gestione**

Questa definizione è semplificativa in questo un RDBMS può gestire fra loro non necessariamente un contenitore di dati coi loro processi (istanza) ma più di uno, ciascuno con il suo identificatore univoco (ORACLE\_SID).

Tipicamente per inciso ad un ORACLE\_SID corrisponde una porta TCP (o combinazione Indirizzo IP + Porta TCP) con cui i client dialogano, ma questo lo vedremo meglio in seguito.

Ogni contenitore separato di dati+processi è detto database che è quindi un insieme di file su disco e relativi servizi di gestione.

Un concetto da tenere sempre presente è che un RDBMS può essere sempre visto tramite due ottiche : quella **fisica** e quella **logica**.

Fisicamente un'istanza, che per semplicità ipotizziamo contenente un solo database come spesso succede, è costituita da :

- file di dati ovvero datafile, raggruppati in contenitori con comuni parametri di gestione detti tablespaces;
- file di informazioni interne utilizzate dai processi ovvero control file e redo log;
- memoria RAM, in cui risiedono i processi, le cache di dati ed istruzioni SQL etc ... (SGA e PGA)

I processi principali (che possiamo vedere tramite il task manager su Windows o con un comando tipo "ps -ax" su Linux) sono i seguenti:

SMON	Avvio del DB, gestione estensioni libere, pulizia DB.
PMON	Ripulisce I processi falliti dagli utenti rendendo disponibili le risorse
DBWR	Scrive sui file di dati I blocchi modificati contenuti nella area SGA
LGWR	Gestisce la scrittura dei file di Redo LOG
CKPT	Facilita il ripristino scrivendo ad ogni checkpoint i dati relativi ai blocchi modificati
ARCH	Gestisce la scrittura ciclica dei file di log
RECO	Recupera le transazioni fallite

Logicamente invece la nostra istanza si può immaginare divisa in:

- oggetti, ad esempio tabelle, relazioni trigger, store procedures
- utenti, ovvero insieme di diritti su insiemi di utenti associati ad una coppia userid/password;

L'utente non interagisce mai direttamente con i datafiles : piuttosto lo fa coi relativi servizi di gestione che ne costituiscono l'interfaccia. Un opportuno servizio (detto Listener) è dedicato ad esporre i servizi e a costruire il canale di colloquio. L'interazione avviene tipicamente attraverso il protocollo TCP/IP e con comandi standard SQL

### **Concetti base sull'installazione di Oracle**

Il RDBMS si basa su una architettura client-server. La parte server può essere installata sotto vari sistemi operativi e contiene tutti strumenti ad interfaccia testuale. La parte client invece è riservata a sistemi windows e contiene nella sua installazione completa:

- tool di amministrazione
- runtime per l'accesso
- strumenti di sviluppo

Tipicamente l'amministrazione si fa da uno strumento grafico detto OEM (Oracle Enterprise Manager) con cui è possibile fare operazioni comuni quali:

- creare e configurare utenti
- aggiungere o modificare tablespaces o datafile

L'installazione sia su Windows che su Linux viene in genere eseguita tramite un opportuno tool grafico (OUI ovvero Oracle Universal Installer) che per la parte server prevede questi:

- installazione degli eseguibili
- creazione del database (vuoto, ovvero con il soli utenti di amministrazione)
- configurazione dell'accesso via rete

Su quest'ultimo aspetto occorre evidenziare che due siano i file fondamentali :

- tnsnames.ora
- listener.ora

Il primo dei due associa in pratica ad una coppia IP:PORT (tipicamente 1521) un ORACLE\_NAME ed un ORACLE\_SID. Questo file è presente anche in una installazione solo client e consente di dire ad esempio che il nome locale "alias" ORACLE9 è in realtà

```
ORACLE9 = ORCL su 172.16.1.1:1521
```

Dal client il principale strumento di interazione è l'interprete SQLPLUS : in questo caso con una sintassi di questo tipo:

```
sqlplus system/manager @oracle9
```

si contatta l'istanza ORCL sulla macchina in questione. Un opportuno file di configurazione farà poi sapere al server 172.16.100.1 che all'ORACLE\_SID=ORCL corrispondono gli eseguibili sotto una opportuna cartella radice.

Tale file è appunto listener.ora : qui si dichiara la presenza di un processo listener (es. ListenerORCL) per l'istanza ORCL che quando viene invocato sulla porta opportuna (1521 nell'esempio precedente) deve attivare un nuovo thread (processo child in Linux) dell'eseguibile di cui sopra.

Si diceva della "cartella radice" : i file di Oracle sono in "quasi tutti" posti sotto un'unica cartella a cui corrisponde la variabile di sistema ORACLE\_HOME che ad esempio può essere :

```
ORACLE_HOME = /oracle/product/9.2.0      (Linux)
ORACLE_HOME = C:/oracle/                 (Windows)
```

Notiamo che il database viene installato utilizzando un apposito utente detto dbowner: nel caso di Linux questo è un utente comune e tutti i processi girano coi permessi di utente standard. Nel caso di Windows è un utente più o meno standard (appartiene ad un

apposito gruppo detto Oracle DBA che prevede alcuni diritti in più su file e eseguibili dell'RDBMS) ma alcuni processi girano comunque con i diritti di SYSTEM.

## Concetti base sull'amministrazione di Oracle

Amministrare un database vuol dire ovviamente un sacco di cose. Qui ci concentriamo su solo su alcune operazioni tipiche.

Innanzitutto far partire o arrestare il DB. Questa operazione si può fare sia da interfaccia grafica che testuale. Per farlo è necessario essere un utente di livello almeno operatore (ovvero sysOPER o sysDBA). Un modo semplice è attraverso la linea di comando e l'utente SYSTEM.

Fatta la login al sistema con l'utente dbowner (in modo che siamo sicuri di avere tutti i percorsi negli eseguibili nel path e/o nel .profile) da linea di comando si procede come segue:

```
sqlplus @oracle9 /nolog
SQL> connect / as sysdba
Shutdown immediate
```

Viceversa per avviare il DB si procede come segue:

```
sqlplus @oracle9 /nolog
SQL> connect / as sysdba
startup
```

Alcune precisazioni sono d'obbligo. Intanto come si nota ci si connette al DB come amministratori senza aver dato password : ciò è possibile solo con l'utente dbowner (appunto con l'opzione /nolog) : in pratica si sfrutta l'autenticazione integrata, ovvero è possibile entrare in Oracle – coi diritti di super user! - senza autenticarsi perché ci si è già autenticati a livello di sistema operativo. Ancora : dopo l'istruzione di shutdown c'è un parametro : in questo caso è stato precisato immediate che è il metodo più drastico per effettuare la chiusura. Ce ne sono anche altri (es. transaction = chiude il DB dopo che tutte le transazioni sono state chiuse) meno "brutali" ma che non possono garantire la chiusura immediata.

Riguardo all'avvio del DB in realtà esistono vari stati in cui questo può trovarsi che variano dallo "spegnimento" totale (shutdown) al caricamento dei soli processi di base (startup nomout), dei processi e dei dati ma non delle interfacce di rete (startup mount) all'avvio completo. Quando si specifica startup senza parametri il DB percorre in realtà l'intera sequenza degli stati : se qualcosa va storto ci troveremo il DB in uno degli stati intermedi, mentre se l'avvio è stato corretto ci troveremo un messaggio di questo tipo:

```
SQL> startup
ORACLE instance started.
```

```
Total System Global Area 135340020 bytes
Fixed Size                  454644 bytes
```

```

Variable Size          109051904 bytes
Database Buffers      25165824 bytes
Redo Buffers          667648 bytes
Database mounted.
Database opened.

```

Un'altra operazione frequente è quella di vedere quante sono le sessioni aperte ed eventualmente effettuare il kill di qualcuna di queste. Per ciascun utente che si connette Oracle assegna una sessione con un proprio identificativo univoco (SESSION\_ID o brevemente SID) : la sessione è attiva se sta mandando dati nel preciso momento in cui la si osserva, mentre è inattiva viceversa. In questo caso viene misurato il tempo di inattività. Le operazioni che si possono compiere in una sessione possono essere le più varie : nei casi in cui queste comportano la modifica di dati o addirittura oggetti (es. ALTER TABLE) il RDBMS genera delle situazioni di accesso esclusivo su uno o più record o sull'intero oggetto. Questo comporta l'apertura di blocchi particolari su record o oggetti detti LOCK : è intuitivo che errori di scrittura di codice o altri fattori possono determinare dei LOCK che non vengono rilasciati che l'utente rileva come situazioni di errori in corrispondenza di normali aggiornamenti dei dati. Queste sono situazioni che l'amministratore è spesso chiamato a risolvere ed il primo (e talvolta definitivo) rimedio è spesso effettuare il Kill di sessioni o LOCK rimasti appesi.

Operativamente è molto semplice eseguire questo dall'interfaccia grafica : nel raggruppamento di voci "Istanza" c'è una funzione "Sessioni" che selezionata dà luogo ad un quadro di questo tipo:

SID	CPU	Memoria - PGA	I/O - Phys Reads	Logical Reads	Hard Parses	Stato	Nome utente	Utente del sistema operativo	ID processo del siste
9	23	623760	994	5504	149	INACTIVE	SYS	PORTATILE2\portatile	5160
10	19	445248	263	1868	38	ACTIVE	SYS	PORTATILE2\portatile	4916
14	1	1715648	93918	6340	7	ACTIVE	CED		3420
18	0	2698688	159757	485	7	ACTIVE	CED		4340
17	0	555476	229	15340	47	ACTIVE		SYSTEM	4712
15	0	1772672	92410	7319	1	ACTIVE	CED		4368
16	0	2821248	110603	477	0	ACTIVE	CED		4372
8	0	2265844	13175	943174	138	ACTIVE	CED		3140
7	0	208320	6	1520	4	ACTIVE		SYSTEM	2736
6	0	208320	1	67	1	ACTIVE		SYSTEM	5028
5	0	422472	667	4353	15	ACTIVE		SYSTEM	1380
4	0	289356	0	0	0	ACTIVE		SYSTEM	4300
3	0	6107348	20	0	0	ACTIVE		SYSTEM	1320
2	0	2564784	20	0	0	ACTIVE		SYSTEM	3624
1	0	208320	0	0	0	ACTIVE		SYSTEM	2956

In questo caso vediamo che di tutte le sessioni correnti le uniche utente sono la 9 e la 10 relative all'utente portatile del client PORTATILE2, effettuate attraverso l'utente oracle SYS. Per effettuare il Kill della 9 banalmente col tasto destro si procede così:

SID	CPU	Memoria - PGA	I/O - Phys Reads	Logical Reads	Hard Parses	Stato	Nome utente	Utente del sistema operativo	ID processo del siste
9	23	623760	004	5504	149	INACTIVE	SYS	PORTATILE2\portatile	5160
10	19	445			38	ACTIVE	SYS	PORTATILE2\portatile	4916
14	1	171			7	ACTIVE	CED		3420
18	0	269				ACTIVE	CED		4340
17	0	555				ACTIVE		SYSTEM	4712
15	0	177			1	ACTIVE	CED		4368
16	0	2821248	110603	477	0	ACTIVE	CED		4372
8	0	2285844	13175	943174	138	ACTIVE	CED		3140
7	0	208320	6	1520	4	ACTIVE		SYSTEM	2736
6	0	208320	1	67	1	ACTIVE		SYSTEM	5028
5	0	422472	667	4353	15	ACTIVE		SYSTEM	1380
4	0	289356	0	0	0	ACTIVE		SYSTEM	4300
3	0	6107348	20	0	0	ACTIVE		SYSTEM	1320
2	0	2564784	20	0	0	ACTIVE		SYSTEM	3624
1	0	208320	0	0	0	ACTIVE		SYSTEM	2956

## Concetti base sul backup di Oracle

Un elemento fondamentale da tener presente è che un DB Oracle avviato che abbia almeno una connessione attiva scrive di continuo sia sulle tabelle dati che sui file di controllo. Segue che copiando tutti i file (e le chiavi del registry se si tratta di una installazione sotto Windows) in un apposito spazio e poi ripristinandoli se il DB non era fermo non solo non avremmo ripristinato la situazione volute, ma probabilmente causeremmo un crash del sistema, perché la nostra foto relativa ad una situazione in corso avrebbe una situazione di dati e file di controllo incoerente.

Lo strumento ufficiale per il backup su Oracle si chiama RMAN ed è basato sull'archiviazione dei log. Tutto ruota intorno ai famosi "redo log files" ; questi memorizzano tutti i dati e comandi SQL che interessano il DB e di base servono all'implementazione del rollback : se vengono dati comandi di inserimento / modifica / cancellazione scorretti e la transazione non è stata chiusa è sempre possibile annullarli recuperando da questi la copia degli oggetti coinvolti precedente al comando. Normalmente tali log funzionano a rotazione quindi il loro storico è relativo solo ad un certo numero di operazioni. Esiste però una modalità, detta appunto archivio log per cui tutti i record tolti dai redo\_log vengono archiviati : in tal caso tutte le operazioni svolte sul DB sono archiviate e da queste è possibile ricostruire il DB stesso in uno stato coerente.

Un metodo tutto sommato basato sullo stesso principio è mettere a cron un opportuno script che esegua lo shutdown del DB, salvi i file fisici e poi lo riavvii.

In pratica per il backup viene spesso utilizzato uno strumento "non ufficiale" che è l'utility EXP che ha il pregio di poter funzionare anche "a caldo", ovvero che i dati da questo esportati se relativi ad un utente se ripristinati con l'opportuno strumento IMP riconducono comunque ad una situazione coerente.

Tralasciando il comando IMP, sul quale servirebbero altre precisazioni diamo un breve reference su EXP. Allora la sintassi, da linea di comando / terminale fatte queste ipotesi:

- password utente system = manager;
- utenti di DB da esportare PIPPO, PLUTO;

è questa:

```
exp system/manager@oracle9 file=utenti.dmp log= utenti.exp  
owner=(PIPP0, PLUTO) buffer=1000000 compress=N consistent=Y
```

### Parametri:

**USERID :** system/manager@oracle9  
Utente a cui viene attribuita la sessione di importazione

**FILE :** utenti.dmp  
Nome del file dove vengono salvati i dati

**LOG :** utenti.exp  
Nome file di log (utile per vedere se durante l'exp ci sono stati degli warning).

**MODALITA' :** owner=PIPP0, PLUTO  
Cosa deve essere esportato. Si può selezionare tutto il DB (full=Y) o anche solo delle tabelle (tables=TABELLA1, TABELLA2).

**BUFFER :** buffer=1000000  
frammento di memoria utilizzato per l'esportazione, in genere multipli di 1000000. Più è elevato più l'exp è veloce e più è la RAM impiegata per l'EXP.

**COMPRESS :** compress=N  
rispondendo Yes viene creata una tabella in un solo extent della dimensione pari alla somma degli extent che aveva originariamente. Si possono però avere problemi di incoerenza.

**CONSISTENT :** consistent=Y  
genera delle cross-tables coerenti.

## **Concetti base sul tuning di Oracle**

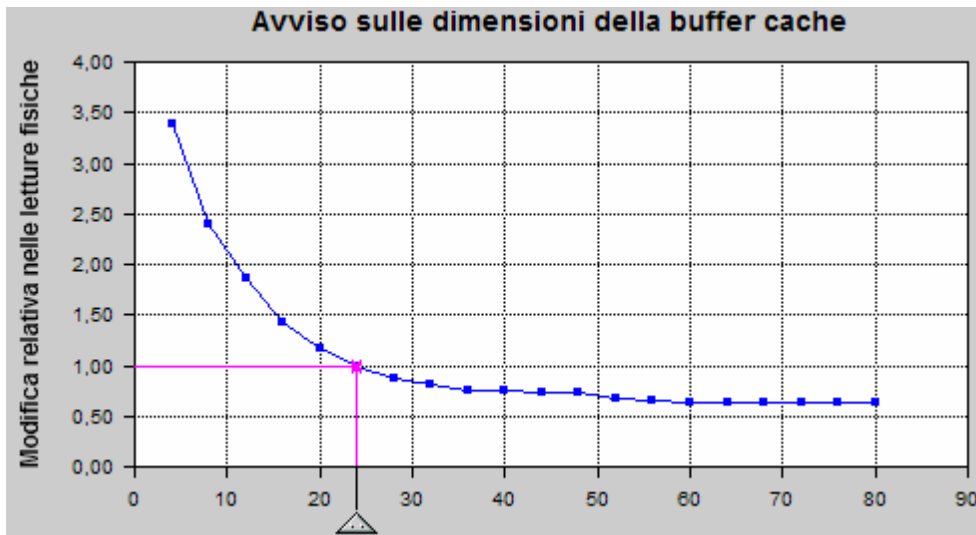
Il tuning di Oracle è una attività complessa e, come si diceva, molto meno importante con le versioni dalla 9 in poi delle precedenti. Una breve occhiata però può essere utile per capire meglio alcuni concetti.

I cosiddetti "parametri di memoria" che sono assegnati al momento dell'installazione attraverso il file initORCL.ora (ammesso che il SID sia ORCL) sono quelli che determinano vari aspetti globali ed in particolare l'utilizzo della RAM. Dall'OEM possiamo ad esempio vedere quali sono i parametri che coinvolgono l'utilizzo della memoria volatile (RAM + virtuale):



In questa piccola istanza di esempio abbiamo una SHARED\_POOL\_SIZE, che è l'area per la cache delle istruzioni SQL di 48MB, mentre la cache per i dati (BUFFER\_CACHE) è di 24MB, mentre la cache dedicata all'esecuzione dei programmi Java è di 32MB.

Al di là dei valori puntuali, anche se esistono tecniche più o meno approssimate per stabilirne i valori al momento dell'installazione, tipicamente si lasciano i default per poi eseguire le regolazioni in base allo storico di dati che viene accumulato via via (ovvero ai grafici che si ottengono dai pulsanti "Avvisi" quale il seguente).



Fra i parametri di memoria importante (un tempo basilare) è il DB\_BLOCK\_SIZE che in genere è 8K. Il blocco è l'unità minima di spazio che viene allocata : il che vuol dire che se ad esempio un datafile deve crescere (ad es. a seguito dell'inserimento di record) di 13K in questo caso saranno aggiunti 2 blocchi (se PCT\_FREE < 23% ovvero 3/13, vedi anche fra poco).

A livello di tablespaces è possibile determinare quanto vuoto iniziale deve essere lasciato e quanto deve essere almeno pieno un blocco per essere considerato "inamovibile" : questi valori sono due parametri classici, detti PCT\_FREE e PCT\_USED.

La PctFree è la parte di blocco che quando viene allocato è libera. Ad esempio se è il 20% un nuovo blocco viene scritto così:



Quando in tale blocco viene fatto un'update si riempie parte di quanto lasciato libero in precedenza e quindi più è grande la pctfree meno le update causano frammentazione (scrivere le nuove info in un blocco diverso):



La pctused ha ruolo duale : quando vengono cancellate righe in una tabella, Oracle marcherà come utilizzabili in futuro solo i blocchi con % di utilizzo minore di PCTUSED. Se questo valore è 100 Oracle riutilizzerà tutti i blocchi purchè abbiamo uno spazietto libero : è chiaro che un valore così alto provoca dispersione (ad esempio se devo allocare un 5%

della dimensione del blocco potrebbe finire anche in 5 blocchi diversi ciascuno con l'1% libero). I manuali Oracle danno la seguente regola: deve essere:

$$PCTFREE + PCTUSED = 80\%$$

Tipicamente  $PCTFREE = 20-30\%$ .

Una tablespace può avere o meno AUTOEXTEND. Se lo ha il RDBMS la fa crescere all'occorrenza mentre se AUTOEXTEND=NO la tablespace accetta nuovi dati solo finché lo spazio ad essa assegnato glielo consente.

Tipicamente vengono gestite con AUTOEXTEND le tablespace di tipo temporaneo, mentre quelle di dati hanno estensione assegnata e vengono aggiunti nuovi datafile quando la loro occupazione supera il 75%-80%.

Blocchi, percentuali libere e occupate, autoextend etc... sono parametri che in realtà servono principalmente ad una cosa : a ottimizzare la gestione dello spazio, ovvero ad avere le informazioni il meno possibile frammentate all'interno del DB. Perché gli oggetti nel DB si frammentano ? Essenzialmente perché le informazioni una volta scritte vengono anche modificate. Per capire un po' cosa succede ipotizziamo di avere un utente "pulito" su cui importiamo due tabelle come in questa immagine:



... la prima rappresentata in azzurro, la seconda in rosso. Ipotizziamo di fare una INSERT nella prima : se la quantità dei dati può essere messa nella PCT\_FREE, bene, a livello di frammentazione non succede niente. Se però lo spazio richiede nuovi blocchi abbiamo una situazione come questa:



in cui la tabella1 è costituita da 2 extents. Situazione duale se ad esempio nella tabella rossa si cancellano records per un totale che porterebbe la PCT\_USED al di sotto del valore di soglia ad esempio per 2 blocchi : in tal caso i due blocchi sarebbero marcati come disponibili ed i corrispondenti dati messi in uno o più blocchi "pieni" in una nuova estensione come in figura:



Il livello di frammentazione è in genere espresso relativamente ad oggetti logici (quali ad esempio tabelle o utenti) come numero di "pezzi" o EXTENT dai quali sono costituiti i corrispondenti fisici detti segmenti, ovvero SEGMENT.

Abbassare la frammentazione degli oggetti (specie le tabelle più utilizzate o gli indici) contribuisce in modo significativo al miglioramento delle prestazioni di un DB. Da notare che esistono degli opportuni comandi per deframmentare anche singoli oggetti ma che se si esegue un EXP, poi si elimina e si ricrea l'utente proprietario degli oggetti da deframmentare e poi si fa un IMP nel nuovo utente gli oggetti corrispondono (almeno inizialmente) a segmenti di un solo pezzo, ovvero tutti costituiti da blocchi contigui, quindi

questa può essere una rudimentale ma efficace tecnica di eliminazione della frammentazione. Da notare comunque che questa tecnica se ci sono oggetti mal gestiti (es. tabelle vuotate con `DELETE * FROM TABELLA1` invece che con `TRUNCATE TABLE TABELLA1`) non tutti i "buchi vuoti" vengono eliminati.