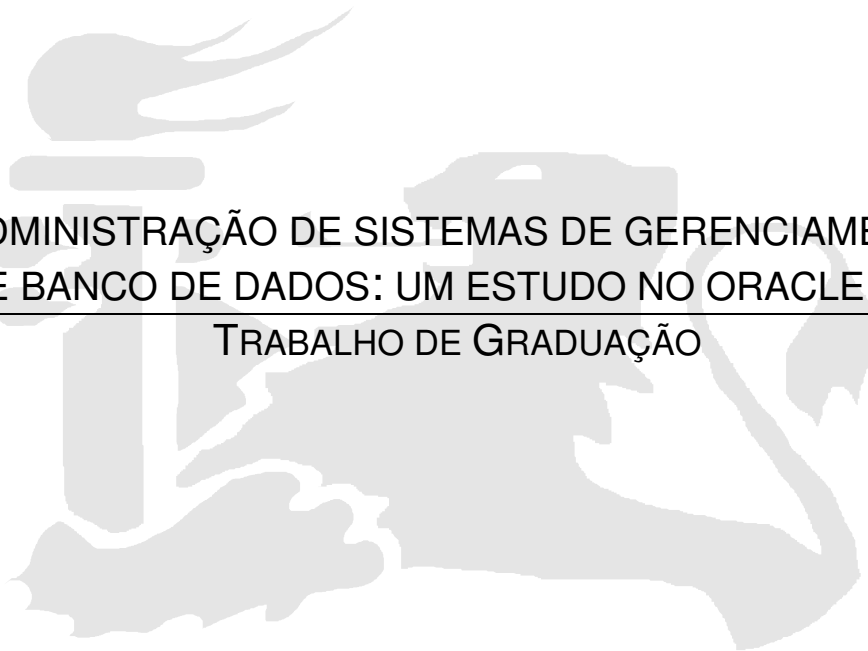
Universidade Federal de Pernambuco
Graduação em Ciência da Computação
Centro de Informática



ADMINISTRAÇÃO DE SISTEMAS DE GERENCIAMENTO DE BANCO DE DADOS: UM ESTUDO NO ORACLE 10G

TRABALHO DE GRADUAÇÃO

Aluno: Marcellus Antonius de Castro Tavares (mact@cin.ufpe.br)

Orientador: Fernando da Fonseca de Souza (fdfd@cin.ufpe.br)

Recife, 5 de outubro de 2006.

Agradecimentos

À minha família pelo apoio durante toda a minha vida e à minha namorada, por todo o carinho e pela paciência que ela teve comigo durante todo esse semestre.

Resumo

O desempenho do Sistema de Gerenciamento de Banco de Dados (SGBD) está diretamente relacionado com a eficiência das aplicações. No contexto atual, sistemas computacionais têm se tornado cada vez mais importantes nos negócios e a eficiência deles pode ser decisiva para o sucesso do empreendimento, principalmente no que se refere aos negócios on-line.

O trabalho apresentado nessa monografia aborda tópicos sobre administração de sistema de gerenciamento de banco de dados. Serão enfatizados principalmente, técnicas de administração que têm influência direta sobre a performance e a estabilidade do SGBD Oracle Database 10g [ORCL10G].

Abstract

The performance of the Database Management System (DBMS) is directly related to the efficiency of the applications. In the current context, computational systems have become very important in businesses and their efficiency can be decisive for the success of the company, especially when the business is totally on-line.

This monograph focuses on topics about the administration of Databases Management System. The project will emphasize on administration techniques that have direct influence on the performance and the stability of the DBMS Oracle Database 10g [ORCL10G].

Sumário

Lista de figuras

Lista de quadros

1 Introdução

Sistemas computacionais têm se tornado extremamente críticos para o mundo corporativo. Ocorre um aumento crescente da dependência entre negócios e os sistemas de informação. Muitas novas gerações de empreendimentos como lojas virtuais, baseiam-se e dependem inteiramente da disponibilidade dessa infra-estrutura de Tecnologia da Informação (TI).

Para sobreviver no mercado e se manterem competitivas, as organizações devem procurar cada vez mais um aumento de eficiência e produtividade. Sistemas de TI, como parte estratégica da empresa, têm um papel fundamental nessa conquista.

Dentre os sistemas computacionais utilizados pelas empresas, destacam-se os sistemas de gerenciamento de banco de dados (SGBD). Seu desempenho está em grande parte relacionado com o desempenho das aplicações.

Por estar presente na maioria das soluções corporativas, como mostra a tabela 1, o SGBD da Oracle [ORCL], mais especificamente a sua última versão, o Oracle Database 10g, foi escolhido como objeto de estudo deste trabalho.

Tabela 1 - Venda mundial de Sistemas de Gerenciamento de Banco de Dados Relacionais (Valores em milhões de dólares)

Companhia	2005	Divisão do mercado (%) 2005	2004	Divisão do mercado (%) 2004
Oracle	6,721.1	48.6	6,234.1	48.9
IBM	3,040.7	22.0	2,860.4	22.4
Microsoft	2,073.2	15.0	1,777.9	13.9
Teradata	440.7	3.2	412.1	3.2
Sybase	407.0	2.9	382.8	3.0
Outros	1,134.7	8.2	1,090.4	8.5
Total	**13,817.4**	**100.0**	**12,757.8**	**100.0**

Fonte: Gartner Dataquest (Maio 2006) [GARTNER]

1.1 Objetivo

Esse trabalho tem como objetivo auxiliar os Administradores de Banco de Dados (ABD) ou DBA (da sigla em Inglês – Database Administrator) na tarefa de administração do SGBD Oracle 10g.

Inicialmente será feito um estudo da arquitetura Oracle e posteriormente serão abordados diversos assuntos relacionados à administração do Oracle 10g como boas práticas de administração que visam aumentar a estabilidade e confiabilidade do sistema, técnicas utilizadas para se maximizar o desempenho do sistema, entre outros. Ao longo do trabalho, a ferramenta *Oracle Enterprise Manager Database Control* [OEMDBC] será utilizada para a execução de boa parte das tarefas administrativas.

1.2 Estrutura do Trabalho

Além deste capítulo introdutório, o trabalho encontra-se assim dividido:

Capítulo 2 - Aborda conceitos sobre a arquitetura do SGBD Oracle como estruturas de armazenamento e de memória.

Capítulo 3 - Trata da administração do Sistema de Gerenciamento Oracle 10g, apresenta as técnicas relacionadas à administração que têm influência na performance do sistema.

Capítulo 4 – Discute o ajuste do banco de dados, neste capítulo são identificados os elementos que podem ser ajustados de modo a melhorar o desempenho geral do sistema.

Capítulo 5 – Faz um estudo da arquitetura e das ferramentas do sistema Oracle 10g que dão suporte a arquitetura de computação em grade [TAURION04] e de que forma toda essa estrutura pode contribuir na melhoria da performance e estabilidade do sistema.

Capítulo 6 – Conclusão e sugestões de trabalhos futuros.

2 Arquitetura do Oracle 10g

O servidor de banco de dados Oracle é composto de uma ou mais base de dados e uma instância. Embora esses termos sejam erroneamente utilizados de forma equivalente, seus conceitos são bastante diferentes.

Um banco de dados é uma coleção de dados em disco. No Oracle, esses dados podem estar em um ou mais arquivos de dados no servidor de banco de dados. Além dos arquivos de dados, o banco de dados é composto de outros dois tipos de arquivos: o arquivo de controle e o arquivo de *redo log*. O arquivo de controle é essencial para o funcionamento da base de dados, ele é continuamente escrito e contém metadados sobre a base e seu status atual. Os arquivos de redo log são indispensáveis para a recuperação da base em caso de falha, pois neles são guardados os históricos das operações na base de dados.

Várias estruturas físicas e lógicas também fazem parte do banco de dados, sendo a tabela a mais comum entre essas estruturas lógicas.

Uma instância é um grande bloco de memória alocado, a SGA (System Global Area), juntamente com processos em background que interagem entre a SGA e o banco de dados.

2.1 Estruturas de armazenamento lógico

2.1.1 Tablespaces

Os arquivos de dados do banco de dados são agrupados em uma ou mais estruturas lógicas chamadas tablespace (espaço de tabela). O arquivo de dados, porém, deve pertencer a uma e somente uma tablespace.

O espaço de tabela pode ser classificado como temporário no sentido de que os segmentos salvos neste espaço são temporários. Tablespaces

temporários são utilizados para trabalho de classificação e criação de índices, por exemplo.

Tablespaces podem ser gerenciados de duas formas: por dicionário ou localmente.

Quando uma tablespace é gerenciada pelo dicionário as extensões (extents) são alocadas e desalocadas de acordo com inserções e deleções em tabelas do dicionário, o que provoca um overhead maior porque mesmo que os usuários e aplicações possuam suas tabelas no tablespace USERS, por exemplo, o tablespace administrativo SYSTEM ainda teria que ser acessado para o gerenciamento desses dados nas tabelas. Isso cria um número maior nas requisições de E/S e pode ser um problema grande para comandos que fazem muitas operações de atualização no banco.

Em uma tablespace gerenciada localmente a alocação e desalocação dos extents são realizadas através de mapas de bits no próprio tablespace.

No Oracle 10g, se o tablespace administrativo SYSTEM for administrado localmente, todos os outros também devem ser.

2.1.2 Blocos

Os blocos são as menores unidades de armazenamento de um banco de dados no Oracle 10g. O tamanho é medido em bytes e especificado no parâmetro de inicialização **DB_BLOCK_SIZE**. O tamanho do bloco deve ser um múltiplo do tamanho de bloco do sistema operacional para que as operações de E/S ocorram de modo eficiente.

2.1.3 Extensões

A extensão está um nível acima na hierarquia dos agrupamentos lógicos no banco de dados no Oracle10g. A extensão consiste de um ou mais blocos alocados a um tipo de objeto específico (tabela ou índice).

2.1.4 Segmentos

Os segmentos consistem em um conjunto de extensões. O segmento contém todos os dados de um agrupamento lógico dentro de um tablespace. Por exemplo, para cada tabela o Oracle aloca uma ou mais extensões para formar um segmento de dados da tabela, para cada índice ocorre essa mesma alocação das extensões para formar o segmento de índice.

Existem no Oracle quatro tipos de segmentos:

- Segmento de dados

  Um único segmento de dados agrupa todos os dados de uma tabela no banco, exceto nos casos de tabelas particionadas ou clustetizadas, onde cada partição é armazenada em um segmento;

- Segmento de índice

  Para todos os índices não particionados tem-se também segmentos de índice que armazenam seus dados;

- Segmento temporário

  Durante o processamento das queries do usuário o Oracle precisa frequentemente de um espaço de armazenamento temporário para a execução de uma operação SQL [SQL] (Structured Query Language). Esse espaço em disco é alocado automaticamente e é chamado de segmento temporário.

  Operações de ordenação, por exemplo, normalmente fazem uso desse espaço;

- Segmento de *rollback*

  Os segmentos de rollback ainda existem no Oracle 10g apenas no tablespace SYSTEM. Nas versões anteriores do Oracle esse segmento era criado para salvar valores anteriores de uma operação de manipulação de dados para o caso da necessidade da transação ter de ser revertida, os segmentos também eram utilizados durante a operação de recuperação do banco.

Entretanto, essa funcionalidade já é obsoleta no Oracle 10g e não estará mais disponível nas versões futuras, pelo fato do Automatic Undo Management tratar de modo automático da alocação e gerenciamento dos segmentos de rollback dentro do tablespace de undo.

2.2 Estruturas lógicas do banco de dados

O que será discutido nessa seção são algumas das mais importantes estruturas lógicas gerenciadas pelo Oracle 10g.

2.2.1 Tabelas

A tabela é a unidade mais básica de armazenamento. Independentemente de seu tipo, seus dados serão armazenados em linhas e colunas.

Abaixo são listados alguns tipos de tabelas encontrados no Oracle:

- Tabelas relacionais

  A tabela relacional é a mais comum. Suas linhas contêm uma ou mais colunas de tipos e comprimentos diversos. As linhas não são organizadas em ordem, a ordenação é feita por heap;

- Tabelas organizadas por índice

  Nas tabelas organizadas por índice o que ocorre é que as linhas da tabela são armazenadas em um índice de árvore B de forma que cada nó do índice contenha a coluna chaveada.

  A vantagem na utilização desse tipo de tabela está no fato da busca por uma linha em particular da tabela ocorrer de modo mais eficiente.

  A desvantagem na utilização nesse tipo de estrutura é que ela não é aplicável para as tabela que não possuam chave primária, como tabelas que armazenam registros de log, por exemplo. Outro inconveniente reside no fato desse tipo de tabela não poder ser membro de um cluster;

- Tabelas de objeto

  As tabelas de objeto são uma estratégia OO (*object-oriented* – orientada a objeto) para banco de dados.

  As suas linhas contêm objetos ou instanciações de tipos definidos pelo usuário. As linhas de uma tabela de objetos podem ser recuperadas pelo seu OID (*object id* – identificador do objeto);

- Tabelas clusterizadas

  As tabelas clusterizadas podem ser uma estratégia para maximizar o desempenho de consultas que fazem referência a duas ou mais tabelas que frequentemente são acessadas juntas. Tome-se como exemplo duas tabelas: empregados e departamentos que possuem a coluna id_departamento em comum, ao criar as tabelas, o Oracle armazenará seus blocos de forma separada. Clusterizando, o Oracle reúne os dados nos mesmos blocos tendo como chave de junção uma coluna em comum denominada **CLUSTER KEY**. Os benefícios são listados abaixo:

  - Redução de E/S;
  - Diminuição do tempo de acesso às tabelas clusterizadas; e
  - Menor custo e plano de execução de SQL. Na clusterização, o valor de cluster key (coluna de junção das tabelas clusterizadas) é armazenado apenas uma vez independentemente do número de linhas que contenham o mesmo valor de junção. Com isso é necessário menor armazenamento de dados relacionados de uma tabela e índice, o que não acontece em tabelas não clusterizadas.

- Tabelas particionadas

  A melhor prática a ser adotada em tabelas grandes (maiores que 2GB) é o particionamento. Uma tabela pode ser particionada ou subparticionada em n partes menores. Particionar uma tabela

traz uma melhoria no desempenho das consultas sobre essa tabela, além de tornar o seu gerenciamento mais fácil.

Isso traz um grande benefício ao DBA. Se uma partição de uma tabela estiver em um volume de disco corrompido, as outras partições ainda estarão disponíveis para consulta enquanto esse volume é reparado.

Cada linha em uma tabela particionada existe em somente uma partição. O que determina em que partição essa linha será encontrada ou adicionada é o que se chama de chave de partição.

As partições dividem-se em três tipos: partições por intervalo, nas quais ocorre uma divisão em intervalos para os possíveis valores das chaves, por lista na qual os valores das chaves são distribuídos em grupos de valores distintos. e por hash na qual as linhas são distribuídas equilibradamente entres as partições com base em uma função de hashing.

2.2.2 Índices

Os índices são utilizados na otimização de consultas. Durante a inserção de dados em uma tabela, as linhas podem ser inseridas em quaisquer blocos. Desse modo, quando ocorre um busca por uma linha específica, a tabela inteira precisa ser lida.

Assim, os índices são criados com intuito de eliminar essa varredura completa na tabela na intenção de se encontrar a linha.

O que ocorre após a criação do índice é que quando uma consulta é submetida, o índice para tabela primeiramente é lido e o identificador da linha é retornado. Por meio desse identificador, chamado de *rowid,* a linha certa é encontrada na tabela.

A criação de um índice tem impacto direto nas operações de **INSERT** e **DELETE** nas tabelas, pois cada criação ou deleção de uma linha é acompanhada de uma mesma operação na estrutura de índices. Isso acarreta

um overhead maior nessas operações e, portanto a criação dos índices deve ser feita de maneira ordenada.

Existem vários tipos de índice, cada qual adequado a um tipo particular de tabela ou aplicação:

- Índices únicos

  É a forma mais comum de índice B*Tree[1]. Freqüentemente ele é utilizado para impor restrição de chave primária de uma tabela;

- Índices não-únicos

  Utilizado para acelerar o acesso à tabela sem impor restrição de unicidade. Esse tipo de índice é criado por padrão se nenhuma outra palavra chave for utilizada em sua construção;

- Índices de chave inversa

  Nesse tipo de índice, todos os bytes em cada valor de chave da coluna são invertidos. O objetivo dessa operação visa o balanceamento da árvore de índices.

- Índice de mapa de bits

  Esse tipo de índice tem uma estrutura significantemente diferente de um B*Tree no que se refere ao nó de folha do índice. Ele armazena uma string de bits para cada valor possível da coluna que está sendo indexada. O comprimento dessa string é mesmo do número de linhas da tabela sendo indexada.

  Portanto, esse tipo de índice é recomendável para colunas de baixa cardinalidade (conjunto de valores – exemplo sexo: M/F).

2.2.3 Seqüências

Uma seqüência é utilizada para atribuir números seqüenciais, que são garantidamente únicos, caso a seqüência não seja redefinida.

Seqüências podem gerar números, com intervalos entre eles especificados, de até 38 dígitos e essa geração pode ser ascendente ou descendente.

[1] Um index B*Tree consiste de níveis de blocos. Com cada nível contendo ponteiros para o nível mais baixo e com um conjunto de blocos folhas no último nível. [BTREE]

2.2.4 Visões

As visões são uma forma de se apresentar os dados de uma tabela ou de uma junção delas. As consultas subjacentes, necessárias para a apresentação dos dados são transparentes para os usuários dessa visão.

Em uma visão regular, onde não ocorre o armazenamento de dados, essa consulta subjacente é executada toda vez que a visão é acessada. Uma extensão à visão regular é a visão materializada, na qual ocorre o armazenamento dos dados. Esse tipo de visão traz o benefício de um menor tempo de resposta para consulta sobre os seus dados, já que a consulta subjacente não é executada durante o acesso a visão.

O Oracle armazena em tabelas do sistema dados sobre a própria base de dados. Exemplos dessas informações incluem os nomes de todas as tabelas na base de dados, as colunas, os nomes, os tipos dessas tabelas, número de linhas que a tabela contém informações relacionadas à segurança, que informam, por exemplo, que usuários têm acesso a que elementos da tabela.

Esses dados relacionados à própria base de dados são chamados de *metadados.*

Com intuito de facilitar o acesso a esses metadados, o Oracle forma visões dessas tabelas. Uma base de dados do Oracle 10g contém dois tipos de visões que com esses metadados:

- Visões de dicionário de dados

  Essas visões possuem nomes que começam com **DBA_**, **ALL_** e **USER_**.

  As diferenças entre elas podem ser ilustradas utilizando a visão do dicionário de dados da **DBA_TABLES** (Quadro 2.1). Ela mostra informações de todas as tabelas da base de dados. A **ALL_TABLES** mostra apenas as tabelas que um usuário em particular do banco possui ou tem acesso. A **USER_TABLES** mostra somente os objetos que o usuário possui.

Quadro 2.1 – Visões de dicionários de dados

Visão	Descrição
DBA_TABLES	Mostra os nomes e informações de armazenamento físico sobre todas as tabelas do banco de dados
DBA_USERS	Mostra informações sobre todos os usuários do banco de dados
DBA_VIEWS	Mostra informações sobre todas as visões do banco de dados
DBA_TAB_COLUMNS	Mostra todos os nome e tipos das colunas das tabelas do banco de dados

- Visões de desempenho dinâmico

  Essas visões possuem nomes que começam com V$. Elas possuem um conteúdo mais dinâmico e fornecem dados atualizados sobre o banco de dados. Um exemplo dessas visões pode ser visto na tabela abaixo.

Quadro 2.2 – Visões de desempenho dinâmico

Visão	Descrição
V$DATABASE	Contém informações sobre o próprio banco de dados.
V$OPTION	Mostra quais componentes opcionais estão instalados no banco de dados.
V$SQL	Mostra informações sobre as instruções SQL dos usuários.

2.2.5 Usuários/Esquemas

O usuário é a pessoa que tem acesso ao SGBD Oracle. Quando um usuário é criado, ele não possui nenhum objeto. Entretanto, assim que os usuários criem os objetos eles farão parte de um esquema que terá o mesmo nome do usuário.

Um esquema pode possuir qualquer tipo de objeto no banco de dados. O proprietário do esquema ou o DBA poderá conceder acessos a esses objetos a outros usuários no banco de dados.

O acesso ao banco de dados por parte do usuário é somente garantido se esse for autenticado. Existem três métodos de autenticação disponível no Oracle 10g: autenticação de banco de dados, autenticação de sistema operacional e autenticação de rede. Essa última é baseada em um solução conhecida como *Public Key Infrastructure* e, para que esse serviço esteja disponível, o *Oracle Advanced Security* [OSECURITY] deve estar instalado.

2.3 Estruturas de armazenamento físico

O banco de dados Oracle é formado por algumas estruturas de armazenamento físico no disco.

Alguns desses arquivos armazenam dados do usuário como, por exemplo, os arquivos de log de redo e os arquivos de dados. Outros contêm meta informações sobre o próprio banco de dados, como é o caso do arquivo de controle. O relacionamento entre essas estruturas é ilustrado na figura 2.1.

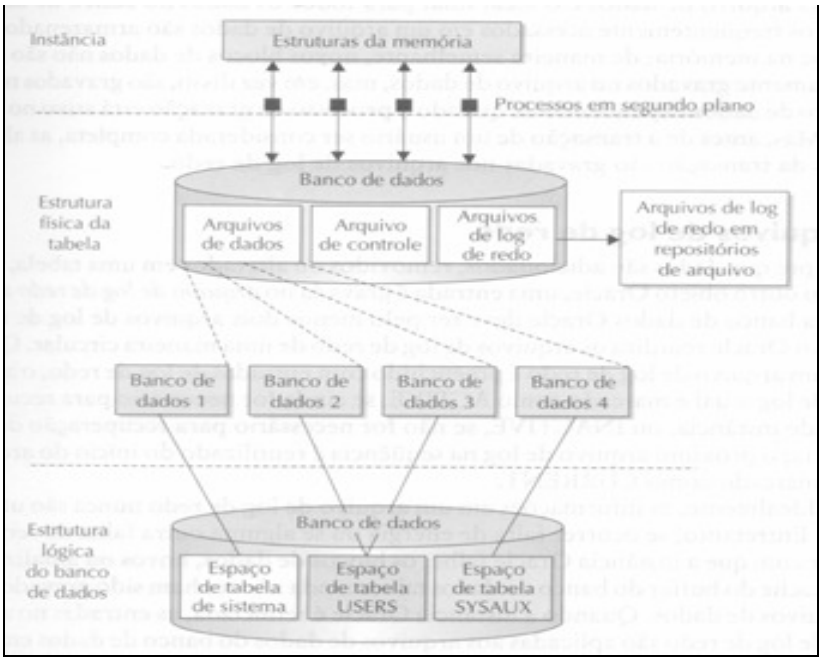


Figura 2.1 – Estruturas de armazenamento físico. Fonte: [LONEY05]

2.3.1 Arquivos de parâmetro de inicialização

Durante a inicialização da base de dados o arquivo de inicialização é lido.

Existem dois tipos de arquivos de inicialização: o PFILE, arquivo texto, conhecido pelo nome *init.ora,* e o SPFILE, arquivo de parâmetro do servidor, *spfile.ora*.

Durante a inicialização, a instância primeiramente procura pelo SPFILE, caso não encontre o PFILE é lido.

O SPFILE na verdade é uma versão binária no PFILE. Quando se utiliza o SPFILE a grande vantagem reside na auto-otimização da base. Há diversos parâmetros que tem seus valores modificados e quando essas mudanças ocorrem em nível de SPFILE, as alterações se tornam persistentes e os parâmetros modificados serão reutilizados em caso de reinicialização da base.

É possível se criar o SPFILE a partir do PFILE pelo seguinte comando:

Quadro 2.3 – Criação do SPFILE a partir do PFILE

```
create SPFILE from PFILE;
```

2.3.2 Arquivo de controle

O arquivo de controle, como dito, é onde são mantidos os metadados sobre a própria base de dados. Ele é um arquivo binário necessário para que o banco seja iniciado e opere com sucesso.

O arquivo de controle é lido para validação antes mesmo da base ser colocada no ar e seu conteúdo é modificado continuamente pelo sistema. Todas as alterações feitas à estrutura do banco de dados são imediatamente refletidas no arquivo.

Por ser tão importante para o funcionamento do sistema, o arquivo de controle pode e deve ser multiplexado.

Essa multiplexação pode ser feita de duas maneiras:

- Via comando *copy* do próprio sistema operacional; ou
- Através do comando:

Quadro 2.4 – Backup do arquivo de controle

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

Esse comando gera um arquivo texto contendo um script SQL que pode ser utilizado para recriar o arquivo de controle. Esse arquivo é criado no diretório especificado no parâmetro do arquivo de inicialização **USER_DUMP_TEST**. Esse valor poder ser obtido pela instrução:

Quadro 2.5 – Exibição do valor do parâmetro USER_DUMP_TEST

```
show parameter user_dump_test;
```

Após a multiplexação é necessário que o sistema seja informado que existem outras cópias que podem ser lidas caso o original não seja encontrado ou esteja corrompido. Para isso, o comando abaixo deve ser executado:

Quadro 2.6 – Especificação dos arquivos de controle

```
alter system set control_files =
       'caminho/ctrl01.ctl',
       'caminho/ctrl02.ctl',
       'caminho/ctrl03.ctl' scope=SPFILE;
```

Como exemplos de informações contidas no arquivo de controle podem ser citados:

- Nome do banco de dados;
- Nome e localizações dos arquivos de dados e de *redo log*;
- Histórico de *redo log*;
- Localização e estado dos logs arquivados; e

- Localização e o estado dos backups feitos pelo RMAN (Recovery Manager).

Essas informações estão disponíveis nas diversas visões dinâmicas do sistema.

2.3.3 Arquivos de *redo log*

Os arquivos de log de redo são utilizados para armazenar um histórico de transações que modificam a base de dados. Todas as modificações que ocorrem na base primeiramente são registradas no buffer de redo log e posteriormente nos arquivos de log de redo.

Cada banco de dados gerenciados pelo Oracle deve possuir pelo menos dois arquivos de log de redo pelo fato do Oracle utilizá-los de maneira circular.

Quando um arquivo de log de redo é preenchido o próximo arquivo da fila é marcado como current e passa a ser utilizado. Enquanto o arquivo de log for necessário para a recuperação da instância, ele é marcado como **ACTIVE**, caso contrário ele é marcado como **INACTIVE**.

Os arquivos de log estão organizados em grupos. Existem pelo menos dois grupos de arquivo em cada base de dados gerenciadas pelo Oracle. Os grupos por sua vez são formados por membros que contêm exatamente a mesma informação. Por razão de segurança, os membros devem ser armazenado em discos diferentes.

O número máximo de grupos é definido no parâmetro **MAXLOGFILES**. O número máximo de membros por sua vez, é definido pelo parâmetro **MAXLOGMEMBERS**. Esses parâmetros são informados na instrução de criação da base.

Quando ocorre um falha na instância Oracle, pode acontecer que os blocos no cache do buffer do banco de dados ainda não tenham sido gravados nos arquivos de dados, então, quando a instância é reiniciada, as entradas no arquivo de log de redo são aplicadas aos arquivos de dados para restaurar o banco de dados de modo a garantir a consistência.

2.3.4 Arquivos de dados

São nos arquivos de dados onde ficam armazenadas as tabelas, índices, dados temporários, dicionário de dados, entre outros.

Cada base de dados deve possuir pelo menos um arquivo de dados, cada arquivo de dados corresponde a somente um *tablespace*. Porém, um espaço de tabela pode conter mais de um arquivo de dados.

O arquivo de dado pode ser criado com a opção **AUTOEXTEND.** Isso significa que o arquivo é automaticamente expandido de tamanho caso seu limite de armazenamento seja atingido.

2.4 Estruturas de memória

2.4.1 System Global Area

Como dito anteriormente, a SGA (System Global Area), faz parte da instância do SGBD Oracle. A SGA é composta de algumas subáreas, dentre as quais podemos citar:

- Caches de Buffer

  É nessa parte da memória que ficam armazenados os blocos dos dados recém lidos do disco que foram solicitados por uma instrução de *select* ou os dados que foram recém modificados pelo usuário.

  O tamanho dessa área de memória pode ser alterado no arquivo de inicialização através do parâmetro **DB_CACHE_SIZE**;
- Shared Pool (Pool compartilhado)

  O Pool compartilhado é dimensionado pelo parâmetro de inicialização **SHARED_POOL_SIZE**. Essa área da memória é composta de duas subcaches importantes, descritas a seguir: a cache de biblioteca, área onde ficam armazenadas informações sobre instruções SQL. Essa cache é compartilhada por todos os usuários.

Portanto, os usuários podem, potencialmente, compartilhar uma mesma instrução SQL, juntamente com seu plano de execução. A outra cache é a de dicionário de dados que armazena uma coleção de dados das tabelas administrativas do sistema (**SYSTEM** e **SYS**);

- Buffer de Log de redo

  Este buffer é a área de armazenamento do histórico. Qualquer operação de modificação na base é registrada. Esses registros das operações são mantidos em memória até que eles sejam escritos nos arquivos de log;

- Stream Pool

  Contem estruturas de dados e controle que dão suporte ao recurso Oracle Streams. Esse recurso gerencia, em um ambiente distribuído, o compartilhamento de dados; e

- Program Global Area

  Esta é uma área de memória alocada para um processo. Dependendo da configuração do servidor (dedicado ou compartilhado) a configuração da PGA pode variar.

  Em um servidor compartilhado, usuários compartilham uma conexão com o banco, minimizando o uso de memória no servidor mas afetando o tempo de resposta para as solicitações do usuário. Nesse ambiente, é a SGA que contém informações sobre sessões do usuário em vez da PGA.

  Em um ambiente dedicado, cada usuário tem a sua própria conexão com o banco. A PGA nesse caso guarda informações sobre a sessão.

2.5 Processos em segundo plano

Quando uma instância é colocada no ar, vários processos que rodam em background são iniciados. São eles:

- Database Writer (DBWR)

  Escreve os blocos modificados do cache *database buffer* para os arquivos de dados físicos. O DBWR não precisa escrever os dados a cada comando **COMMIT**, pois é otimizado para minimizar o E/S. Geralmente o DBWR escreve os dados para o disco se muitos dados são lidos para o cache do database buffer na SGA e não existe espaço livre para esses novos dados. Os dados menos recentemente usados são escritos para os arquivos de dados em primeiro lugar;
- Log Writer (LGWR)

  Escreve todas as entradas de *redo log* para o disco. Os dados de redo log são armazenados em memória no redo log buffer cache, na SGA. No momento em que uma transação for efetivada com o comando **COMMIT** e o *redo log* buffer estiver preenchido, o LGWR escreve as entradas de redo log nos arquivos redo log apropriados.

  A um tempo específico, todos os dados do database buffer cache modificados são escritos em disco pelo processo DBWR; este evento é chamado de checkpoint. O processo checkpoint é responsável para informar ao processo DBWR o momento de gravar os dados em disco. O DBWR também atualiza os arquivos de controle do banco de dados para indicar o mais recente checkpoint. O processo CKPT é opcional; se ele não estiver presente, o LGWR assume sua responsabilidade;
- System Monitor

  Este processo efetua a recuperação da instância em caso de falhas, durante a sua inicialização. Em um sistema com múltiplas instâncias (como na configuração *Oracle Parallel Server* [OPS], por exemplo), o processo SMON de uma instância também pode executar a recuperação de outras instâncias que podem ter falhado. Ele também limpa os segmentos temporários que não estão sendo usados, liberando memória, e recupera qualquer transação pendente

no caso de uma falha em arquivos físicos ou mesmo no disco. O processo de recuperação dessas transações é executado pelo processo SMON quando a tablespace afetada volta a ficar disponível;

- Process Monitor

  O process monitor (PMON) executa a recuperação do processo de um usuário quando esse processo falha. Ele limpa a área de memória e libera os recursos que o processo do usuário estava usando. O PMON também verifica o processo despachante (dispatcher) e os processos servidores (server processes) e os reinicializa se tiver acontecido qualquer falha;

- ARCn

  O processo archiver (ARCH) copia os arquivos *redo log* para fita ou mesmo outro disco, no momento em que um deles torna-se completo. Esse processo geralmente está presente quando o banco de dados está sendo utilizado no modo **ARCHIVELOG**.

- RECO

  O processo recoverer (RECO) é usado para resolver transações distribuídas pendentes causadas por uma falha na rede em um sistema de bancos de dados distribuídos. A certos intervalos de tempo, o processo RECO do banco de dados local tenta conectar-se ao banco de dados remoto para automaticamente completar e efetivar a transação (**COMMIT**) ou descartar (**ROLLBACK**) a porção local de uma transação pendente em um sistema distribuído.

3 Gerenciamento do Oracle 10g

Algumas das atribuições mais importantes do administrador de banco de dados é garantir a estabilidade do SGBD e um bom desempenho do mesmo. Esse capítulo, portanto, aborda conceitos e técnicas que auxiliam o DBA na execução dessas tarefas no Oracle 10g.

3.1 Gerenciamento de Tablespaces

Um bom gerenciamento dos tablespaces tem influência direta na desempenho do Oracle 10g. Saber administrá-los, portanto, é uma tarefa fundamental na vida do administrador do banco de dados.

Existem dois tipos de tablespaces no Oracle 10g: o temporário e o permanente.

3.1.1 Tablespaces Permanentes

Os tablespaces permanentes são aqueles que contêm segmentos que persistem além da duração de uma sessão ou transação.

Como exemplo de tablespaces permanentes pode-se citar o SYSTEM e o SYSAUX. Esses dois espaços nunca devem conter segmentos de usuários por dois motivos: por questões de organização, de não se misturar dados administrativos com dados do usuário e para se evitar um gargalo de E/S devido uma alta disputa por blocos nesse espaço.

O conteúdo do espaço SYSAUX pode ser visualizado utilizando-se o *Oracle Enterprise Manager Database Control,* EM.

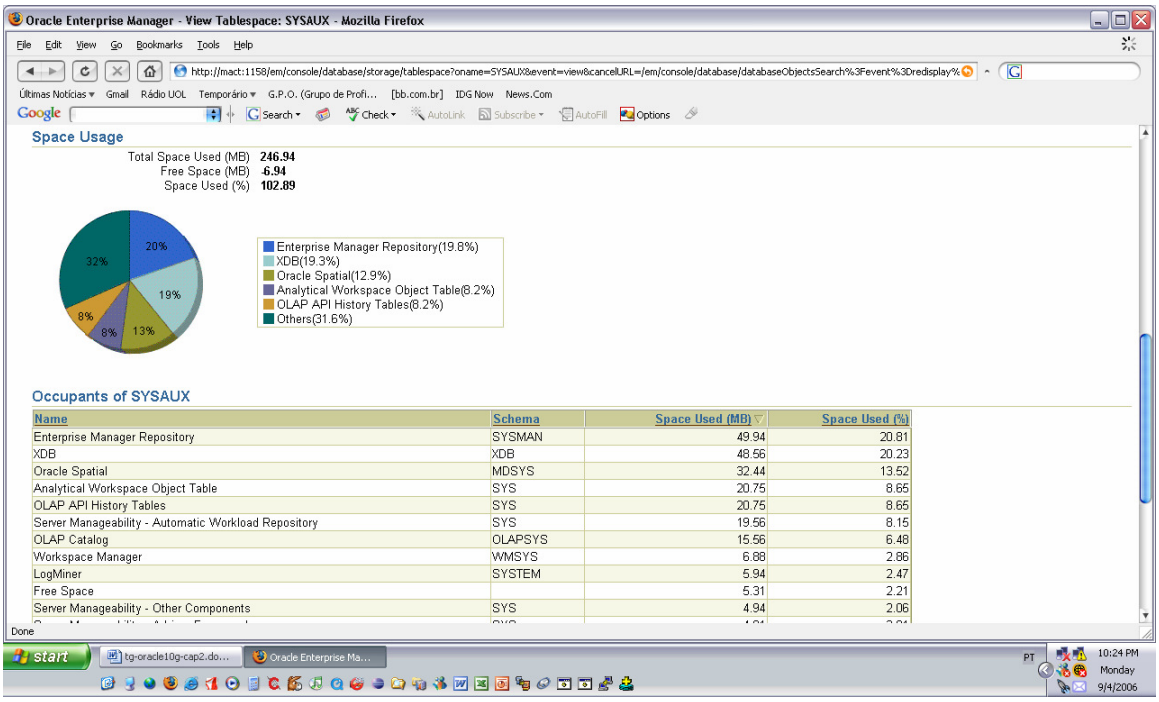


Figura 3.1 – Conteúdo do tablespace SYSAUX

É comum analisar através do EM as disputas de E/S nos tablespaces. Ao se identificar os *hotspots*, em alguns casos o DBA, na tentativa de solucionar o problema, deve mover blocos de um tablespace para outro.

Para se evitar um possível problema desse tipo, algumas sugestões devem ser seguidas. Como exemplo é dividir os segmentos em diferentes espaços de tabela com base nos seus tipos, tamanhos e freqüências de acesso. Sendo assim deve-se ter:

- Separação, em diferentes tablespaces, os segmentos grandes dos pequenos.
- Tablespaces distintos para cada aplicação.
- Uma separação dos segmentos de tabela de seus índices.
- Separação das visões materializadas das tabelas base.

3.1.2 Tablespace Temporário

No Oracle 10g é possível se definir mais de um tablespace temporário ativo no banco de dados, porém, como somente um tablespace temporário pode

ser atribuído para cada usuário, sessões desse usuário só poderão utilizar esse tablespace. Isso pode causar um problema de desempenho pela disputa de E/S como mencionado anteriormente.

Por essa razão, o Oracle dá suporte ao recurso de grupos de espaço de tabela temporário, que significam na verdade, uma lista de espaços de tabela.

A grande vantagem na utilização desse recurso vem do fato de se poder ter múltiplas sessões de um usuário utilizando diferentes tablespaces temporários em um mesmo grupo para a realização de suas operações de classificação. Isso é particularmente bastante útil quando podemos alocar discos físicos distintos para cada tablespace, reduzindo drasticamente a disputa de E/S naqueles segmentos.

O grupo deve conter pelo menos um tablespace temporário. Abaixo é ilustrada a criação do grupo temporário *grupotmp* a partir de três tablespaces (tmp1, tmp2, tmp3) previamente criados.

Quadro 3.1 – Criação de grupos temporários

```
alter tablespace tmp1 tablespace group grupotmp;
alter tablespace tmp2 tablespace group grupotmp;
alter tablespace tmp3 tablespace group grupotmp;
```

O comando para se definir a utilização desse grupo de tablespace temporário como padrão para os usuários recém criados é o mesmo para se definir um tablespace temporário aos usuários.

Quadro 3.2 – Definição de um grupo de tablespaces temporário como padrão

```
alter database default temporary tablespace grupotmp;
```

Para a remoção de um grupo de tablespace, deve-se descartar todos os seus membros. Para isso, atribuímos aos membros outro grupo ou grupo nenhum (string “”).

Quadro 3.3 – Remoção de grupos temporários

```
alter tablespace tmp1 tablespace group '';
alter tablespace tmp2 tablespace group '';
alter tablespace tmp3 tablespace group '';
```

3.1.3 Tablespace de undo

O tablespace de undo é utilizado pelo Oracle para reverter transações e fornecer consistência para as instruções de **SELECT** que são executadas sobre tabelas que estão sendo modificadas. Esse tablespace também fornece dados utilizados pelo recurso *Oracle Flashback*.

O tablespace de undo guarda dados dos registros antes que estes sejam atualizados ou excluídos de modo que se uma sessão do usuário falhar antes dele fazer o **COMMIT** ou **ROLLBACK,** essa modificações são revertidas.

Esse tablespace especificamente será estudado mais adiante nesse trabalho.

3.2 Gerenciamento de Armazenamento

No tópico anterior foi abordado o gerenciamento de tablespaces, mas não foi visto como criá-las, por exemplo. Neste tópico serão discutidos os aspectos físicos de um banco de dados e como gerenciá-los de modo que se possa maximizar o desempenho do SGBD.

3.2.1 Criação de tablespaces

Será tomada como assunto de estudo nesse tópico a criação de tablespaces permanentes. A criação dos outros tablespaces são bastante semelhantes.

O comando SQL para a criação é mostrado abaixo:

Quadro 3.4 – Comando para a criação de tablespaces

```
CREATE [BIGFILE | SMALLFILE] TABLESPACE <nome_tablespace>
DATAFILE '<caminho_e_nome_arquivo>'
SIZE <integer><K | M | G | T> [REUSE]
AUTOEXTEND <OFF | ON>
BLOCKSIZE <bytes>
[<LOGGING | NOLOGGING>][FORCE LOGGING]
[ DEFAULT <COMPRESS | NOCOMPRESS>]
EXTENT MANAGEMENT LOCAL UNIFORM SIZE <tamanho_extent>
SEGMENT SPACE MANAGEMENT AUTO
<ONLINE | OFFLINE>;
```

A criação do tablespace envolve alguns conceitos importantes. No momento de sua criação, pode-se especificar se esse é uma tablespace bigfile ou smallfile. Com o smallfile pode-se especificar mais de um arquivo de dados e assim minimizar os riscos de perda do arquivo de dados. No tipo bigfile apenas um arquivo de dados pode ser especificado, porém este pode ser muito maior que um arquivo de dados de uma tabela smallfile. Um espaço de tabela bigfile pode ter um arquivo de dados tão grande quanto 128TB.

Quanto ao gerenciamento dos extents, ele pode ser automático ou não. Quando este é automático pode-se especificar o tamanho de crescimento do arquivo de dados no parâmetro SIZE.

A opção de logging permite habilitar a geração de entradas de *redo* para segmentos contidos no tablespace, o que ocasiona um grande impacto na carga de dados principalmente. A opção de logging em um tablespace pode ser sobrescrita durante a criação de um objeto nesse tablespace (tabelas, índices, etc). Para que isso possa ser evitado, a opção force logging pode ser especificada.

A cláusula SEGMENT SPACE MANAGEMENT permite especificar de que forma os espaços livres e utilizados dentro segmento serão gerenciados. As duas possíveis opções são:

- MANUAL

  Essa opção significa usar free lists. Elas são listas de blocos de dados que contêm espaços disponíveis. Essa opção é chamada de manual porque é preciso que seja especificado manualmente alguns parâmetros para a criação dos objetos no tablespace (PCTUSED, FREELISTS e FREELISTS GROUPS) ; e

- AUTO

  O gerenciamento automático é mais eficiente. Ele elimina a necessidade de se especificar parâmetros na criação de objetos.
  O gerenciamento automático utiliza bitmaps para gerenciar os espaços livres nos segmentos. Neste caso, é utilizado um mapa que informa o status de cada bloco dentro do segmento.

3.2.2 Alteração de tablespaces

Tanto a tarefa de criação quanto a de a manutenção dos tablespaces pode ser facilitada utilizando-se a ferramenta EM.

Abaixo, na figura 3.2, 3.3 e 3.4, serão mostradas algumas telas da ferramenta EM durante a realização de algumas operações de manutenção.

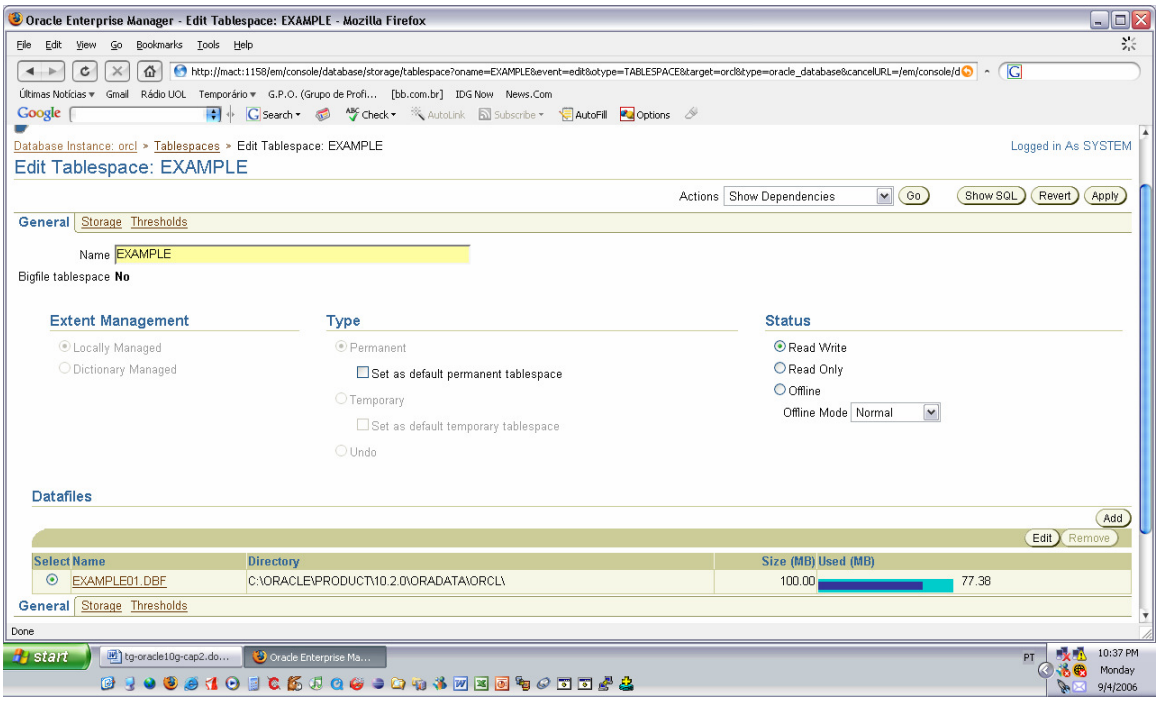


Figura 3.2 – Edição de tablespaces

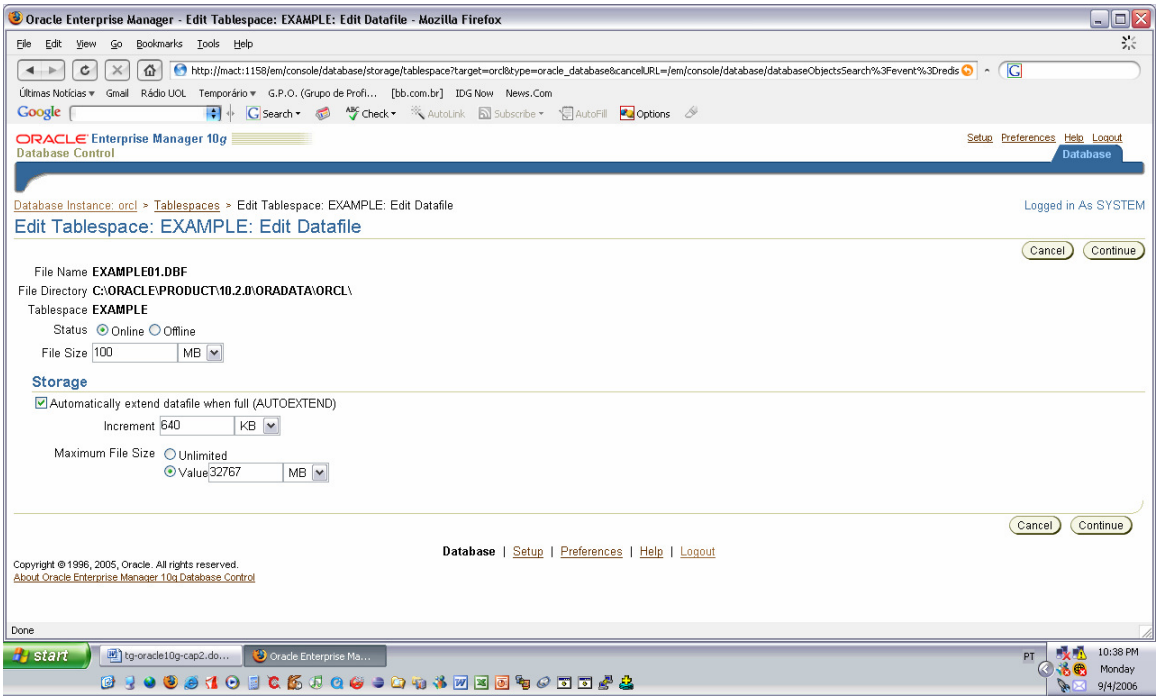


Figura 3.3 – Edição do arquivo de dados

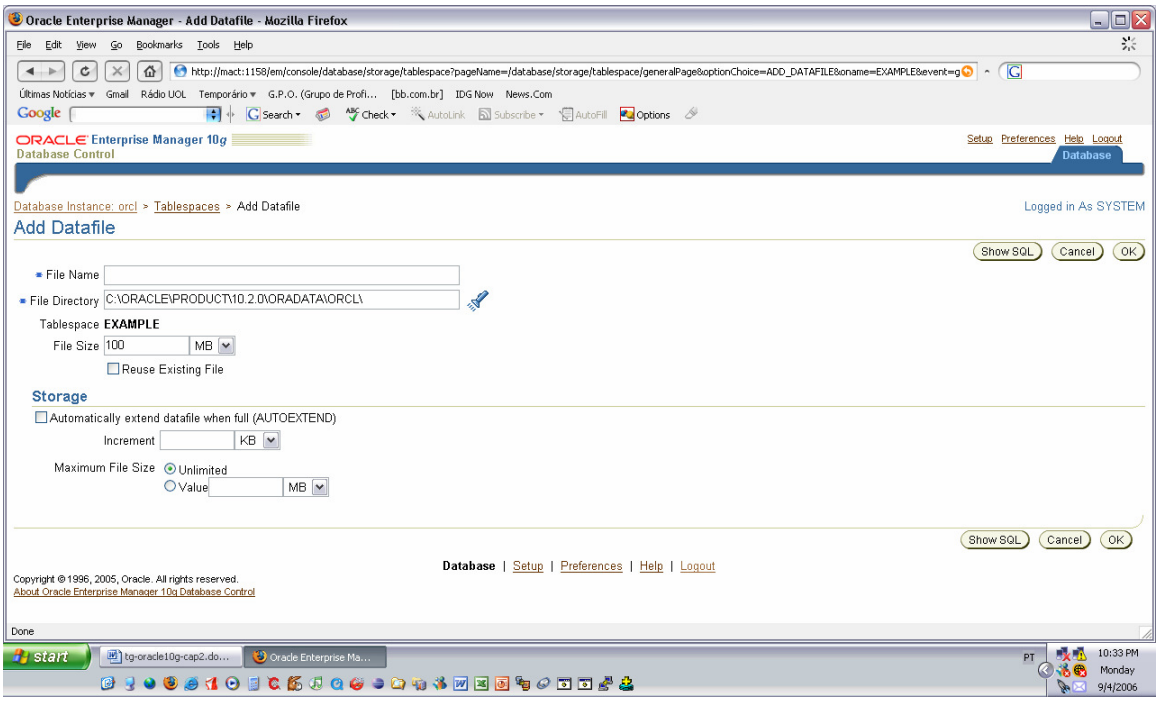


Figura 3.4 – Adicionando arquivo de dados

3.2.3 Movendo arquivos de dados

A alteração da localização dos arquivos de dados às vezes se faz necessária para se melhorar o desempenho de E/S do banco de dados. Para realizar essa tarefa, o DBA tem duas opções: através de comandos SQL como alter database ou **ALTER TABLESPACE** ou através do EM.

O comando alter tablespace não funciona para a movimentação dos arquivos de dados das tablespaces administrativas (SYSTEM e SYSAUX), dos espaços de tabelas temporários e de undo on-line. Neste caso o comando **ALTER DATABASE** deve ser utilizado.

Abaixo são mostrados os passos para a execução desses comandos.

ALTER DATABASE

1. Localizar os arquivos de dados através das visões (V$DATAFILE e V$TABLESPACE)

Quadro 3.5 – Comando para a localização dos arquivos de dados do tablespace

```
select d.name
from v$datafile d join v$tablespace t using (ts#)
where t.name = 'EXAMPLE';
```

2. Desativar a instância (o usuário deve estar logado como sysdba)
3. Utilizar os comandos do SO para movimentar os arquivos
4. Abrir o banco de dados no modo MOUNT

Quadro 3.6 – Comando para iniciar o banco de dados (modo MOUNT)

```
startup mount;
```

5. Utilizar o comando **ALTER DATABASE** para alterar as referências aos arquivos de dados

Quadro 3.7 – Comando para atualizar as referências aos arquivos de dados (ALTER DATABASE)

```
alter database rename file 'caminho_antigo/example01.dbf' to
   'novo_caminho/example01.dbf';
```

6. Abrir o banco de dados no modo OPEN

Quadro 3.8 – Comando para abrir o banco de dados

```
alter database open;
```

ALTER TABLESPACE

É sempre preferível que se utilize esse comando ao mover arquivos de dados de tablespaces diferentes dos acima descritos. A razão para isso é que o banco de dados, exceto pelo tablespace que está sendo modificado, continua disponível durante toda a operação.

1. Colocar o tablespace em modo offline.

Quadro 3.9 – Comando de alteração da tabela para modo offline

```
alter tablespace users offline;
```

2. Utilizar os comandos do SO para movimentar os arquivos
3. Utilizar o comando **ALTER TABLESPACE** para atualizar as referências ao arquivo de dados.

Quadro 3.10 - Comando para atualizar as referências aos arquivos de dados (ALTER TABLESPACE)

```
alter tablespace rename datafile
   'caminho_antigo/users01.dbf' to
   'novo_caminho/users01.dbf';
```

4. Colocar o tablespace em modo online

3.2.4 Movendo arquivos de controle

Os arquivos de controle são indispensáveis para o funcionamento da instância. Devido a sua importância, multiplexá-los, como dito no capítulo anterior, é uma boa alternativa na tentativa de se preservar esses arquivos. É importante que caso algum arquivo de controle seja corrompido, ele possa ser substituído por outro, possivelmente localizado em algum outro volume de disco.

3.3 Monitoramento do uso de espaço

Muitas vezes o DBA se depara com problemas de gerenciamento de espaço. Saber como evitá-los ou resolvê-los é, também, uma tarefa primordial na vida do DBA.

Os problemas de gerenciamento normalmente são alocados em três categorias: espaço insuficiente em um tablespace, espaço insuficiente para segmentos temporários e muito ou pouco espaço de undo alocado.

Espaço insuficiente em um tablespace

Esse problema acontece quando o tablespace não é definido com o atributo AUTOEXTEND. Neste caso, a quantidade total de dados que o tablespace pode armazenar fica limitada à quantidade total de espaço dos arquivos que a compõe.

Se o atributo AUTOEXTEND for definido, os arquivos de dados que compõem o tablespace crescerão automaticamente para atender às solicitações. Isso claramente, é limitado pela capacidade de armazenamento do disco.

Uma boa prática por parte do DBA é, portanto, monitorar o uso desse espaço para detectar tendências e assim assegurar espaço suficiente de armazenamento para as futuras solicitações.

Espaço insuficiente para segmentos temporários

O espaço de tabela temporário é utilizado nas operações de classificação ou mesclagem de dados. Quando um usuário submete uma operação de classificação, construções de índices, consultas distinct ou union ao servidor, o Oracle armazena nos segmentos temporários resultados intermediários dessas operações que não podem ser realizadas em memória.

Caso não haja espaço de armazenamento suficiente nesse espaço temporário e ele não poder ser auto-estendido, a consulta do usuário falhará.

Muito ou pouco espaço de *undo* alocado

O espaço de tabela de undo contém segmentos utilizados para reverter transações não-confirmadas e para oferecer consistência de leitura a consultas de longa duração (que começam antes de modificações ocorrerem em uma tabela).

Da mesma forma que espaços de tabela temporário, o objetivo do gerenciamento desse espaço é garantir espaço livre suficiente mas sem alocar mais que o necessário.

3.3.1 Blocos, extensões e segmentos no Oracle

No capítulo anterior esses elementos foram definidos. Neste tópico, eles serão discutidos em mais detalhes.

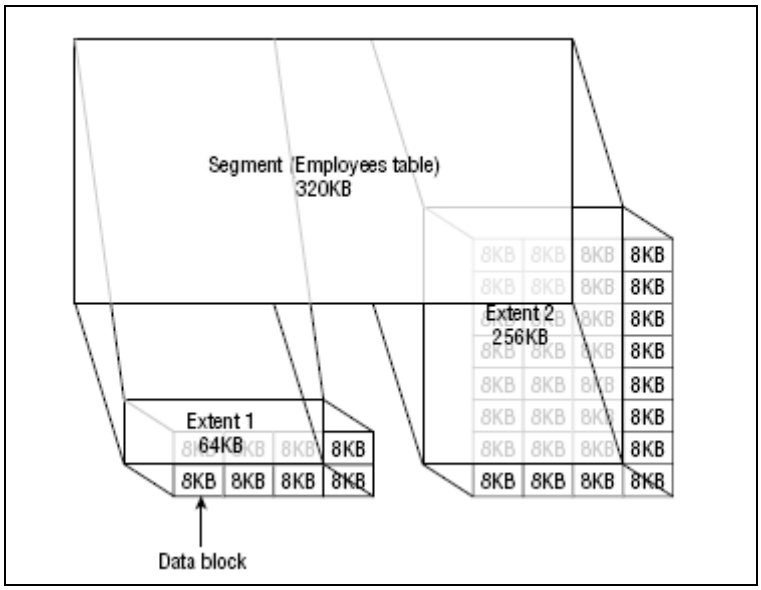


Figura 3.5 – Blocos, extensões e segmentos. Fonte: [DAWES05]

Blocos

Como dito, os blocos são as menores unidades de armazenamento do Oracle. O formato do bloco é apresentado abaixo:

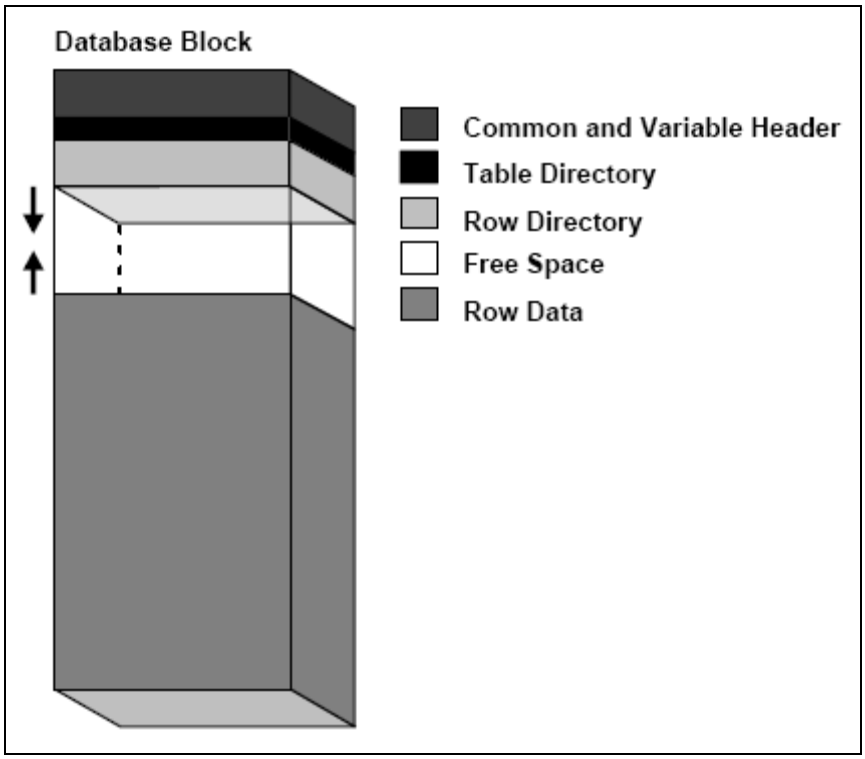


Figura 3.6 – Formato do bloco Oracle. Fonte: [ORCLCONCEPTS]

No cabeçalho encontram-se informações como tipo de dados que está armazenado no bloco – dados ou índice. A seção diretórios de tabela contém informações relativas à tabela com linhas no bloco. Um bloco somente pode conter linhas provenientes de uma única tabela ou entradas provenientes de um único índice. Uma exceção a essa regra ocorre quando a tabela é clusterizada. Nesse caso, o diretório identifica todas as tabelas com linhas nesse bloco. O diretório de linhas traz detalhes sobre linhas específicas das entradas de tabela ou de índices no bloco.

Quando um bloco é alocado, o espaço livre está disponível para novas linhas e para as atualizações das linhas já existentes. O espaço está disponível para novas inserções até o limite definido no bloco pelo parâmetro PCTFREE, especificado quando o segmento é criado. Quando este limite é atingido, nenhuma inserção é permitida.

Uma linha pode se distribuir por mais de um bloco se o tamanho da linha for maior que o tamanho do bloco ou se a linha modificada não couber mais no bloco original.

No primeiro caso, a linha será armazenada em uma cadeia de blocos. Esse caso pode ser inevitável se uma linha contiver colunas que excedam o tamanho do bloco. No Oracle 10g, o tamanho máximo do bloco é 32KB.

No segundo caso, o Oracle fará uma migração dos dados da linha inteira para um novo bloco e deixará no primeiro um ponteiro que apontará para essa nova localização. Isso pode vir a ser um problema porque um segmento com muitas linhas migradas pode causar problemas de desempenho de E/S, uma vez que o número de blocos necessários para satisfazer à consulta pode dobrar. Nesses casos, ajustar o valor do PCTFREE ou reconstruir a tabela pode vir a ser uma solução.

Extensões

A extensão é o próximo nível de alocação de espaço. A extensão consiste de um conjunto de blocos.

No momento da criação de uma tabela, uma extensão inicial é criada. Posteriormente, se necessário, extensões incrementais vão sendo alocadas.

Essas extensões incrementais podem ter ou não o mesmo tamanho da extensão inicial.

Essa diferença é definida no momento da criação do tablespace – atributo UNIFORM.

Quando essa alocação das extensões não é uniforme, ela pode ser automaticamente dimensionada pelo Oracle (AUTOALLOCATE). O Oracle utiliza um algoritmo de alocação de espaço que visa diminuir a fragmentação do tablespace.

Quando a autoalocação do espaço é utilizada, os parâmetros de armazenamento (INITIAL, NEXT, PCTINCREASE e MINEXTENTS) são utilizados apenas como parâmetros para o algoritmo interno do Oracle.

Segmentos

As extensões descritas acima são alocadas para um único segmento que por sua vez está contido dentro de um único tablespace.

O segmento representa um só tipo de objeto no banco de dados, seja ele uma tabela, uma partição, um cluster, um índice, ou um segmento temporário.

O modo de como o espaço dentro do segmento é gerenciado depende de como o tablespace que contém o segmento foi criado. Se o tablespace for gerenciado por dicionário, o Oracle utiliza freelists para o gerenciamento do espaço. Caso o tablespace seja gerenciado localmente, há duas maneiras de se fazer esse gerenciamento: freelists ou mapa de bits. É fortemente recomendado pela Oracle que esse espaço seja gerenciado por mapas de bits por questões de desempenho, pois é permitido um maior número de acessos concorrentes aos mapas de bits em comparação com as freelists.

Caso o gerenciamento do espaço seja realizado pelos mapas de bits, as instruções PCTUSED, FREELIST e FREELIS GROUP declaradas no momento da criação de uma tabela ou índice serão ignoradas.

3.3.2 Visões de Dicionário e de Desempenho Dinâmico

Algumas visões, tópico abordado no capítulo anterior, são frequentemente utilizadas pelo DBA para se ter um entendimento do uso do espaço em disco. A seguir serão citadas algumas dessas visões, suas descrições e suas funcionalidades.

DBA_TABLESPACES

Esta visão contém informações sobre os espaços de tabela. Ela possui uma linha para cada tablespace seja nativo ou conectado através de outro banco de dados, sua descrição é mostrada na figura 3.7.

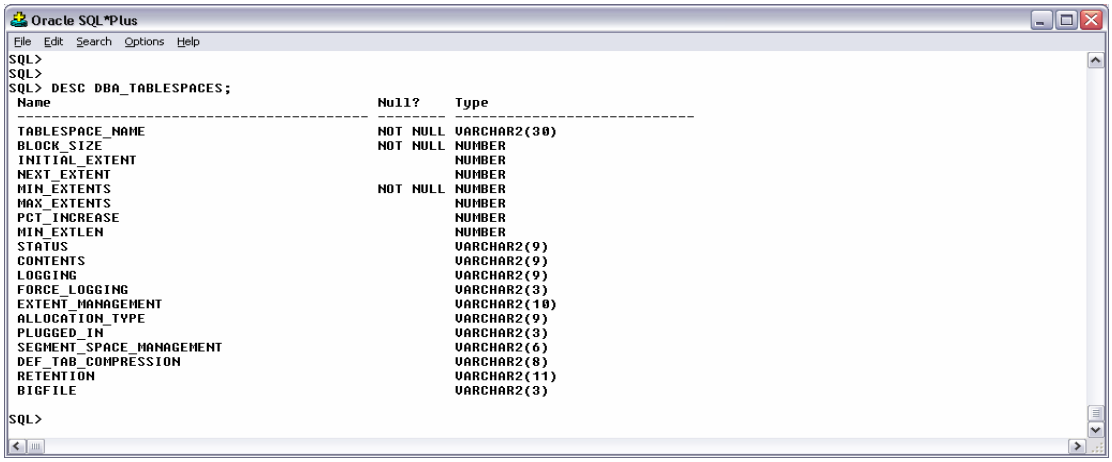


Figura 3.7 – Descrição da visão DBA_TABLESPACES

DBA_SEGMENTS

Nesta visão estão representados os segmentos de dados. De modo semelhante à visão DBA_TABLESPACES, apresenta uma linha para cada segmento de dados. Esta visão é bastante útil para se obter informações sobre o dono do objeto e o tablespace onde ele está armazenado. Sua descrição é mostrada na figura 3.8.

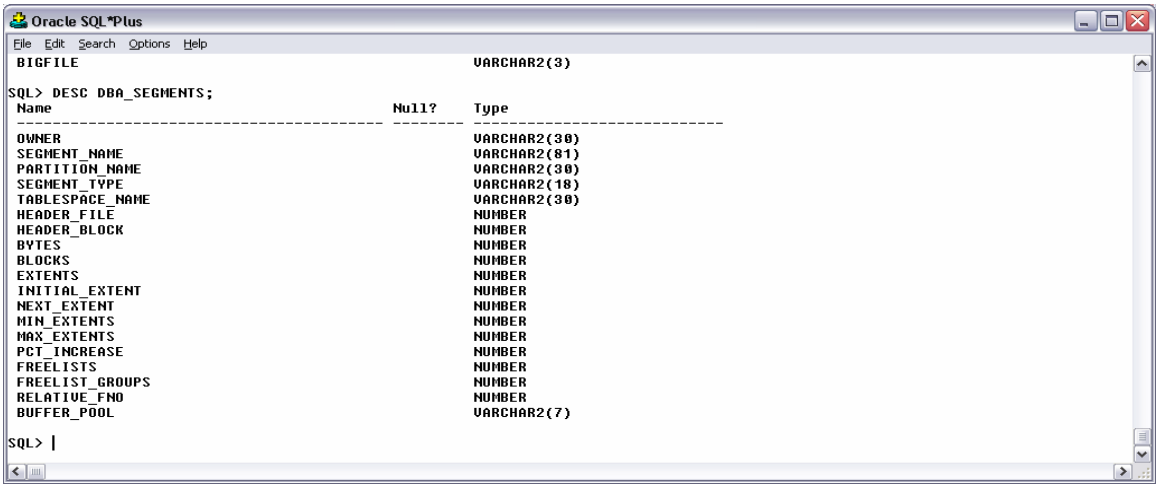


Figura 3.8 – Descrição da visão DBA_SEGMENTS

DBA_FREE_SPACE

Esta visão, descrita na figura 3.9, está dividida pelo número de arquivo de dados dentro do tablespace. Uma consulta útil sobre esta visão seria para se analisar a quantidade de bytes livres em cada tablespace.

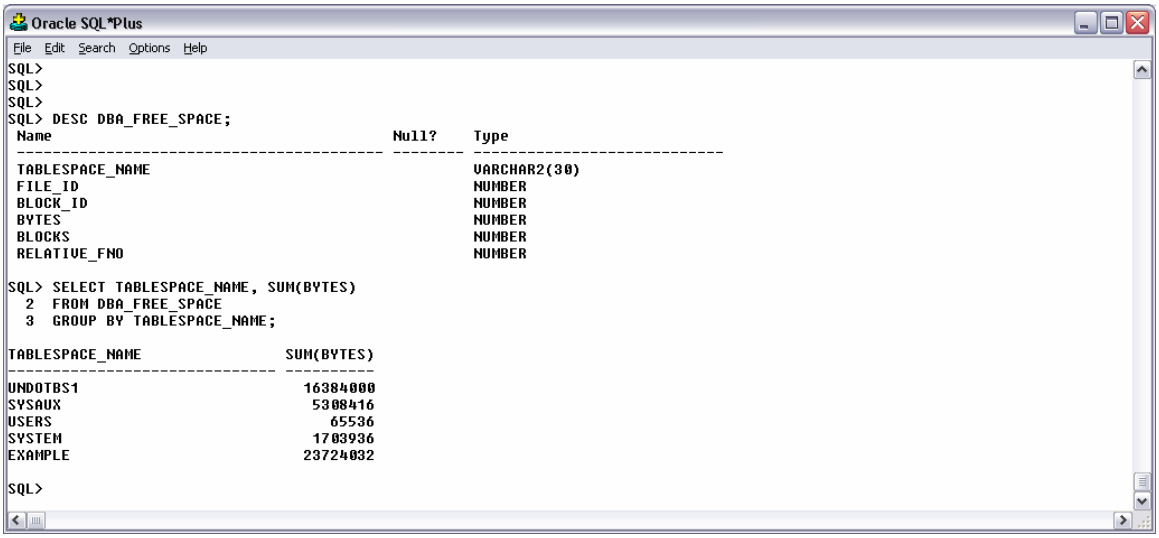


Figura 3.9 – Descrição da visão DBA_FREE_SPACE

DBA_TRESHHOLDS

Esta visão apresenta uma lista das diferentes métricas que dão uma estimativa da saúde do banco de dados e especifica uma condição sob a qual um alerta será emitido se a métrica alcançar o limiar ou exceder um valor especificado.

A manutenção dos valores dessa visão, em geral, é feita utilizando a ferramenta EM, porém o Oracle disponibiliza um conjunto de blocos PL/SQL (DBMS_SERVER_ALERT) para configurar esses valores sem o auxílio da interface. A configuração e obtenção desses valores podem ser obtidos através das funções SET_THRESHOLD e GET_THRESHOLD respectivamente.

As mensagens de alerta podem ser lidas através dos pacotes DBMS_AQ e DBMS_AQDM ou o DBA pode configurar o banco para que essas mensagens sejam automaticamente enviadas para ele por e-mail.

Na instalação padrão da base, alguns limiares já vêm pré-configurados, exemplo:

- Mais de 1200 cursores concorrentes abertos;
- Se um espaço de tabela estiver mais de 85% cheio (alerta) ou mais de 97% (crítico) cheio; e
- Se o número de processos concorrentes alcançar 80% do valor especificado no parâmetro de inicialização PROCESSES.

3.3.3 Recursos Disponíveis no Oracle 10g

O Oracle 10g dispõe de recursos que auxiliam o DBA na tarefa de gerenciamento do espaço. Eles são abordados a seguir.

Oracle Managed Files (OMF)

O OMF é um recurso que visa facilitar o controle dos arquivos físicos do sistema operacional. Sem este recurso, por exemplo, o DBA após a exclusão de um espaço de tabela se via obrigado a ter de excluir os arquivos de dados manualmente. Para a execução dessa tarefa, ele devia ter em mente, ou identificar através de consultas as visões de dicionários e as localizações desses arquivos.

Desse modo, o OMF cria e exclui automaticamente esses arquivos e também assegura que os mesmos sejam identificados unicamente. Essa última funcionalidade evita a corrupção de dados, principalmente durante a criação inadvertida de novos arquivos de dados com o mesmo nome com a cláusula REUSE.

Os arquivos não-OMF podem coexistir sem problemas com os arquivos OMF. A conversão de um arquivo não-OMF para OMF é simples. O que o DBA precisa fazer é configurar alguns parâmetros de inicialização, descritos no quadro 3.11, e assim, pode ser feita a criação dos novos arquivos como OMF.

Quadro 3.11 – Parâmetros de inicialização relacionados ao OMF

Parâmetro de Inicialização	Descrição
DB_CREATE_FILE_DEST	Diretório de arquivo do sistema operacional onde os arquivos de dados e tempfiles são criados se nenhum nome de caminho for especificado no comando **CREATE TABLESPACE.** Esse local é utilizado para arquivos de *redo log* e arquivos de controle se DB_CREATE_ONLINE_LOG_DEST_n não for especificado.
DB_CREATE_ONLINE_LOG_DEST_*n*	Especifica a localização padrão para armazenar arquivos de *redo log* e arquivos de controle quando nenhum nome de caminho é especificado para arquivos de *redo log* ou arquivos de controle em tempo de criação do banco de dados. Até cinco destinos podem ser especificados com esse parâmetro, permitindo até cinco arquivos de controle multiplexados e cinco membros de cada grupo de *redo log*.
DB_RECECOVERY_FILE_DEST	Define o nome de caminho padrão no sistema de arquivos do servidor onde backups RMAN, *redo logs* arquivados em repositórios de arquivos e logs de flashback estão localizados. Também utilizado para arquivos *redo log* e arquivos de controle se nenhum DB_CREATE_FILE_DEST nem DB_CREATE_ONLINE_LOG_DEST_*n* for especificado.

Automatic Storage Management (ASM)

O ASM também é um recurso que simplifica a administração dos arquivos. Com o ASM é permitido ao administrador referenciar um grupo de discos ao invés de apenas discos individuais.

O ASM provê funcionalidades como espelhamento e armazenamento seguro. Por exemplo: durante a criação dos arquivos de dados, estes são distribuídos automaticamente entre o grupo de discos ASM. Com isso, o desempenho das consultas aumenta, pois a E/S é balanceada entre os vários discos.

Um grupo de discos no ASM é gerenciado como uma única entidade. Os discos podem ser adicionados ou removidos de um grupo sem a necessidade de se desativa o banco de dados. Quando isto acontece, o ASM se encarrega de

fazer novamente o balanceamento dos arquivos entre os discos de modo a maximizar o desempenho de E/S.

Existem três tipos de grupos de discos no ASM: de redundância normal, de alta redundância e redundância externa.

A diferença entre os dois primeiros tipos está no número de grupos de falha. O primeiro utiliza dois grupos de falhas e o segundo utiliza pelo menos três. No terceiro tipo, a redundância é fornecida por outro mecanismo, como por exemplo, um array de armazenamento (RAID).

A criação da instância ASM pode ser feita pelo assistente de criação de base de dados. Durante a criação é necessária a especificação do parâmetro de inicialização INSTANCE_TYPE, que no caso será ASM.

Depois da criação da instância ASM, devem ser criados os grupos de discos. A sintaxe para a criação de um grupo de discos de redundância normal é exibida a seguir:

Quadro 3.12 – Comando para criação de grupos de disco de redundância normal

```
CREATE DISKGROUP disk_group_1 NORMAL REDUNDANCY
  FAILGROUP failure_group_1 DISK
    '/devices/diska1' NAME diska1,
    '/devices/diska2' NAME diska2,
  FAILGROUP failure_group_2 DISK
    '/devices/diskb1' NAME diskb1,
    '/devices/diskb2' NAME diskb2;
```

Nesse caso, ao contrário do que possa parecer, o disco b1 e o disco b2 não são espelhamentos dos discos a1 e a2. O ASM se utiliza de todos os discos para a criação de um sistema tolerante a falhas. Por exemplo: um arquivo pertencente ao grupo de disco acima pode ser criado no disco a1 com uma cópia em b2 ou criado em b1 com uma cópia em a2. Uma ocorrência de falha em um grupo não acarretará indisponibilidade, uma vez que todos os arquivos estão devidamente espelhados.

Para a criação de um tablespace no ambiente ASM, o comando **CREATE TABLESPACE** deve fazer referência ao grupo de discos, como mostrado do quadro 3.13.

Quadro 3.13 – Comando CREATE TABLESPACE (referência ao arquivo de dados / ambiente ASM)

```
   CREATE TABLESPACE tbls1 DATAFILE '+disk_group_1/user_data_1'
SIZE 1024M
```

Como o grupo de discos utiliza um sistema de arquivos virtual, esse espelhamento pode ser bastante útil quando se trata dos arquivos de *redo logs*, backups, no quadro 3.14 é ilustrada a criação de um grupo de arquivos de *redo logs* de tamanho 50MBytes:

3.3.4 Gerenciamento do tablespace SYSAUX

Como mencionado em tópicos anteriores, o tablespace SYSAUX é um espaço administrativo auxiliar ao tablespace SYSTEM. O SYSAUX armazena diversos componentes, entre eles: Enterprise Manager Repository, Oracle Intermedia, Oracle Data Mining.

Esses componentes podem ser identificados através de uma consulta a visão V$SYSAUX_OCCUPANTS, como mostrado no quadro 3.14.

Quadro 3.14 – Comando para a seleção dos ocupantes do tablespace SYSAUX

```
   SELECT occupant_name, occupant_desc, space_usage_kbytes
   FROM V$SYSAUX_OCCUPANTS;
```

O espaço de tabela SYSAUX não é essencial para o funcionamento da base. Caso esse tablespace seja colocado off-line, apenas esses componentes não estarão mais disponíveis.

Por questões de desempenho de E/S, o monitoramento e o gerenciamento desse tablespace é importante. A consulta exibida na caixa

anterior solicita informações sobre o espaço utilizado em kbytes pelo componente (space_usage_kbytes) e o resultado desse campo pode informar ao DBA se é necessário a movimentação desse componente para um espaço de tabela dedicado.

O Oracle disponibiliza alguns procedures para auxiliar o DBA na execução dessa tarefa.

A coluna MOVE_PROCEDURE da mesma visão (V$SYSAUX_OCCUPANTS) mostra ao DBA que procedure deve ser executada para a transferência do componente.

Quadro 3.15 – Consulta para seleção do procedure para movimentação dos componentes do tablespace SYSAUX

```
SELECT occupant_name, move_procedure
FROM V$SYSAUX_OCCUPANTS;
```

A movimentação do componente é então realizada dessa forma:

Quadro 3.16 – Comando para movimentação dos componentes do tablespace SYSAUX

```
EXECUTE nome_da_procedure('tablespace_destino')
```

3.3.5 Gerenciamento de arquivos de *redo log*

Uma preocupação que o DBA precisa ter é quanto ao gerenciamento de espaço de objetos que existem fora do banco de dados, como é o caso do armazenamento dos arquivos de log em repositórios de arquivos.

No modo ARCHIVELOG, o arquivo de log de redo é copiado para os destinos, especificados pelo parâmetro LOG_ARCHIVE_DEST_n.

O arquivo de log que está sendo copiado precisa que essa cópia seja realizada com sucesso para o número de destinos mínimo especificado no

parâmetro LOG_ARCHIVE_MIN_SUCCED_DEST. Caso isso não seja possível (uma causa dessa falha poderia ser falta de espaço no destino) o banco de dados é suspenso.

3.3.6 Gerenciamento de espaço – Segment Advisor

O Oracle possui um conjunto de ferramentas predefinidas que auxiliam o administrador do banco na identificação de problemas relacionados a espaço em disco no banco de dados, abordaremos a ferramenta *Segment Advisor.*

O Segment Advisor é um recurso que realiza análises sobre a fragmentação de espaços de tabela, segmentos ou objetos no banco de dados.

A fragmentação do espaço, que ocorre em virtude das operações de manipulação, poder ser desfeita através de operações de redução sobre esses objetos.

Reduzir um segmento significa dizer que o Oracle disponibiliza o espaço livre desse segmento para outro segmento no tablespace. Um efeito dessa operação é o aprimoramento das futuras operações sobre o segmento. Há uma otimização da utilização do cache, uma vez que menos blocos precisam estar no cache para satisfazer as operações sobre o segmento.

Para que o Oracle possa reduzir uma tabela, por exemplo, é necessário que se habilite o movimento das linhas dessa tabela. Isso é conseguido através do comando **ALTER TABLE**, mostrado no quadro 3.17.

Quadro 3.17 – Comando para habilitar o movimento de linhas de uma tabela

```
alter table nome_da_tabela enable row movement;
```

Como exemplo do uso desse e de outros recursos abordados em tópicos posteriores, vamos tomar a análise da tabela fictícia RH.EMPREGADO.

Para que o Segment Advisor analise a tabela, é preciso configurá-lo. Isso é feito através da execução de um bloco PL/SQL, como ilustra o quadro 3.18:

Quadro 3.18 – Configuração do Segment Advisor

```
declare
   name varchar2(100);
   descr varchar2(500);
   obj_id number;
begin
   name := '';
   descr := 'Check RH.EMPREGADO';
   dbms_advisor.create_task('Segment Advisor', :task_id, name,
                                descry, NULL);
   dbms_advisor.create_object(name,       'TABLE',       'RH',
'EMPREGADO',
                                 NULL, NULL, obj_id);
   dbms_advisor.set_task_parameter(name,       'RECOMMEND_ALL',
'TRUE');
   dbms_advisor.execute_task(name);
end;
```

A procedure DBMS_ADVISOR.CREATE_TASK especifica o tipo de advisor, que no caso é o "Segment Advisor". A procedure retorna um id, único para a tarefa e um nome gerado para o programa que chama.

Dentro da tarefa, identificada pelo nome, indicamos o objeto a ser analisado com DBMS_ADVISOR_CREATE_OBJECT. Os atributos para essa procedure variam de acordo com o tipo de objeto. Para o objeto do tipo tabela, é necessário a especificação apenas no esquema e o nome da tabela.

Os parâmetros passados para o procedimento DBMS_ADVISOR.SET_TASK_PARAMETER serve para informar a quantidade de recomendações que desejamos obter do Segment Advisor. No caso, foi solicitada a obtenção de todas as recomendações. O terceiro parâmetro é utilizado para desativar (false) ou ativar (true) as recomendações para a tarefa.

A execução da tarefa do Segment Advisor é apenas iniciada com a chamada do procedimento DBMS_ADVISOR.EXECUTE_TASK.

O resultado da operação do Segment Advisor, ou seja, suas recomendações, podem ser encontradas nas visões DBA_ADVISOR_FINDINGS e DBA_ADVISOR_RECOMENDATIONS. A identificação dessas recomendações é feita através no id da tarefa.

```
SQL> print task_id;

        TASK_ID
      -----------
              6
```

Quadro 3.19 – Consulta às recomendações do Segment Advisor

```
SQL> select owner, task_id, task_name, type, message,
more_info
     from DBA_ADVISOR_FINDINGS
     where task_id = 6;
```

A visão DBA_ADVISOR_ACTION informa o script necessário para realizar a operação de redução.

Quadro 3.20 – Seleção da ação sugerida pelo Segment Advisor

```
SQL> select owner, task_id, task_name, command, attr1
     from DBA_ADVISOR_ACTIONS
     where task_id = 6;
```

A redução das tabela não necessita espaço extra de armazenamento, nem impede a sua utilização durante a operação. Além disso, todos os índices presentes na tabela são mantidos.

3.3.7 Uso de índices

O uso de índices em tabelas diminui muito o tempo de respostas nas operações de seleção de seus elementos. Porém algum dos índices criados no banco de dados podem estar sendo subutilizados e que sua não existência faria mais sentido quando se leva em consideração o espaço ocupado pelo índice e o overhead existente durante as operações de modificações na tabela (uma inserção numa tabela leva a uma criação de uma entrada no índice).

Por esse motivo, o DBA tem a sua disposição a visão de desempenho dinâmico V$OBJECT_USAGE, a qual monitora a utilização dos objetos na base de dados.

O tópico anterior, tomamos como exemplo a tabela EMPREGADO do esquema RH. Considerando que existe um índice nessa tabela que referencia o id do trabalho do empregado, pode-se analisar a utilização desse índice alterando o índice e ativando o monitoramento de utilização, como mostrado no quadro 3.21.

Quadro 3.21 – Comando para monitorar a utilização do índice

```
SQL> alter index RH.EMPREGADO_ID_TRAB monitoring usage;
```

Após um período de operações sobre o banco, pode-se verificar na visão se o índice está realmente sendo utilizado, com o comando do quadro 3.24.

Quadro 3.23 – Consulta à visão V$OBJECT_USAGE sobre a utilização do índice

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

O índice ainda pode ser analisado com relação ao uso eficiente de espaço.

O comando a seguir pode ser utilizado periodicamente para se verificar se o uso do espaço alocado para o índice se torna ineficiente.

```
SQL> analyse index RH.EMPREGADO_ID_TRAB validate structure
     Index analysed.
SQL> select pct_used from index_stats where name =
  'EMPREGADO_ID_TRAB'

        PCT_USED
        --------
           20
```

No caso exibido, verifica-se que o índice utiliza apenas 20% do espaço a ele destinado. Em casos de desempenho insastifatório, deve-se realizar a operação de reconstrução do índice, como mostrado no quadro 3.24.

Quadro 3.24 – Comando para a reconstrução de índice

```
SQL> alter index RH.EMPREGADO_ID_TRAB rebuild;
```

3.3.8 Resumable Space Allocation

Algumas longas operações como *imports* e processos *batch* podem falhar devido à impossibilidade do servidor alocar mais espaço (extensões) para os objetos. Isto pode ocorrer devido ao fato do objeto ter atingido o limite de extensões disponível, de não haver espaço no tablespace para a expansão do objeto ou a cota de espaço do usuário ter sido excedida.

Em versões anteriores do Oracle, operações desse tipo teriam que ser reexeutadas, possivelmente com alguns rollbacks manuais sobre as operações previamente executadas que falharam.

No Oracle 10g, é possível evitar esse tipo de problema no sentido que existe um recurso, o Resumable Space Allocation, que possibilita a suspensão dessas operações de longa duração em casos de falha de alocação de espaço.

O DBA pode, portanto, tornar essas instruções retomáveis alterando o parâmtro de inicialização RESUMABLE_TIMEOUT com um valor, em segundos, maior que zero.

Um usuário pode ativar operações retomáveis, em nível de sessão, através do comando abaixo:

```
SQL> alter session enable resumable timeout 3600;
```

Isso significa dizer que a operação que não tenha espaço suficiente será suspensa por 3600 segundos (1 hora) para que a condição de falta de espaço seja reparada. Caso isso não aconteça, a transação irá falhar.

Como exemplo, será analisada a cópia de uma tabela de um tablespace para outro. No caso a seguir, as linhas da tabela EMPREGADO, que contém os dados dos funcionários da filial, estão sendo inseridas em outra tabela que contém informações sobre todos os funcionários da organização.

```
SQL> insert into RH.EMPREGADOS_ORGANIZACAO
       select * from RH.EMPREGADO;
```

Em caso de falha por falta de espaço, uma notificação do sistema alertaria o usuário quanto ao problema e a transação falharia. Para a instrução anterior, a seguinte mensagem seria exibida:

```
*
ERROR at line 1
ORA-01653: unable to extend table RH.EMPREGADOS_ORGANIZACAO by X
          in tablespace Y
```

A solução, portanto, para problemas desse tipo, seria utilizar o Resumable Space Allocation.

Com o recurso ativado, a mensagem diferiria e informaria ao usuário e ao DBA (este pode receber uma notificação via e-mail) que a transação seria suspendida.

Através da visão DBA_RESUMABLE, o DBA pode verificar que transações estão suspensas por falta de espaço:

Quadro 3.25 – Comando para listagem de transações suspensas

```
SQL> select user_id, instance_id, status, name, error_msg
        from DBA_RESUMABLE;
```

A partir disso, o DBA poderia aumentar o tamanho do tablespace para dar suporte a às operações. Essa mudança iria refletir na visão, no sentido de que status da transação passaria de SUSPENDED para NORMAL.

Como mencionado anteriormente, o DBA pode ser notificado via e-mail sobre a suspensão das transações. É necessário para isso, a definição de um trigger de sistema. Um exemplo é ilustrado abaixo:

```
CREATE OR REPLACE TRIGGER resumable_notify
      after suspend on database
DECLARE
   -- variáveis
BEGIN
   -- altera o intervalo para a solução do problema para 2h
   dbms_resumable.set_timeout(7200);
   -- envia o e-mail
   utl_mail.send ('dba@...');
END;
```

3.3.9 Automatização de tarefas com o Scheduler

Embora a execução desses e de outros recursos possam ser executados a qualquer momento, o Oracle, através do pacote DBMS_SCHEDULER, pode automatizá-los com o intuito de prevenir e capturar problemas antes que eles causem uma falha no sistema.

O DMS_SCHEDULER traz novos recursos e funcionalidades em relação ao anterior, o DBMS_JOB. Ele contém procedimentos de criação, remoção de jobs (uma tarefa a ser executada, podendo ser um bloco PL/SQL, um código binário, uma aplicação Java ou um script shell), além de facilitar a repetição automática das execuções de trabalhos como o CREATE_SCHEDULE e o particionamento desse trabalho em categorias através da procedure CREATE_JOB_CLASS.

3.3.10 Gerenciamento de Transações com o Tablespace Undo

Como dito anteriormente o tablespace de undo é utilizado para realizar *rollbacks* de transações, fornecer dados consistentes para leitura e também é utilizado nas operações de *flashback*.

Rollback

Toda transação no Oracle pode ser revertida. Para que isso ocorra quando uma operação de manipulação de dados é submetida, os valores antigos alterados por esta são armazenados no tablespace de undo.

Quando a transação precisa ser revertida, o Oracle utiliza os registros de undo correspondentes no tablespace para desfazer todas as operações contidas pertencentes à transação e, por fim, libera os blocos nas linhas afetadas.

Consistência de Leitura

Fornecer consistência de leitura significa dizer que um usuário que está lendo dados de uma tabela que está sendo modificada por outro usuário, não verá essa modificação até que o segundo confirme (**COMMIT**) sua transação.

Os segmentos de undo são, portanto, utilizados para reconstruir os blocos de dados a uma versão de leitura consistente.

A execução dessa operação pode levantar erros do tipo *snapshot too old* devido ao fato de não haver mais espaço para o armazenamento dos segmentos na tabela de undo.

Operações de Flashback

Algumas operações de flashback como: flashback table - que restaura uma tabela de acordo com um ponto do tempo no passado, flashback query - que permite a visualização de uma coluna em um período do passado e o pacote **DBMS_FLASHBACK** utilizam dados do tablespace de undo.

Gerenciamento e monitoramento do tablespace

O processo de criação e manutenção do tablespace de undo é semelhante ao de qualquer outro tablespace. Abaixo são mostrados alguns passos do processo de criação do tablespace utilizando a ferramenta Enterprise Manager, nas figuras 3.10, 3.11 e 3.12.

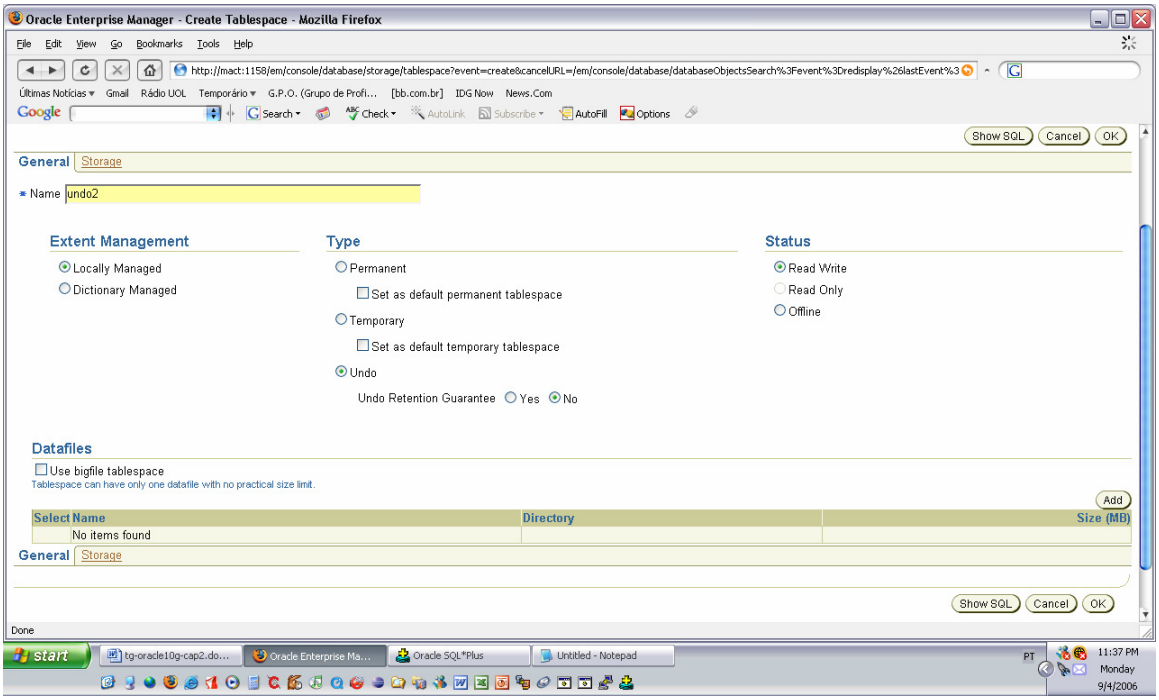


Figura 3.10 – Criação de um tablespace de undo.

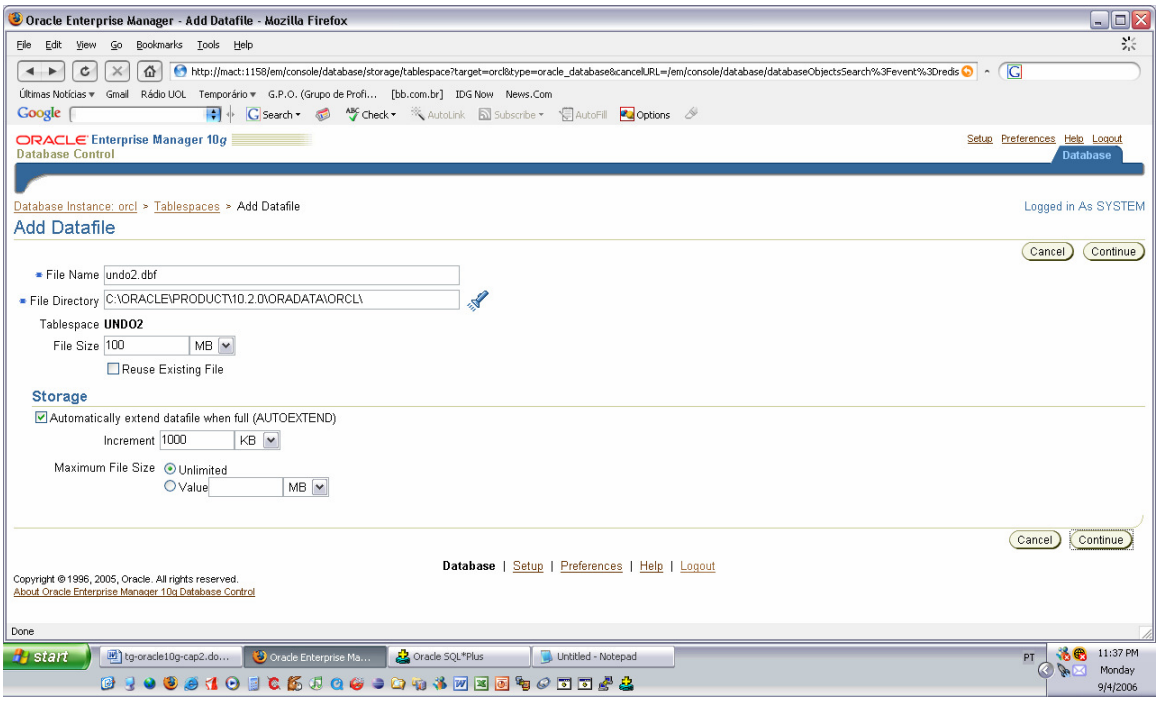


Figura 3.11 – Adicionando arquivo de dados ao tablespace de undo.

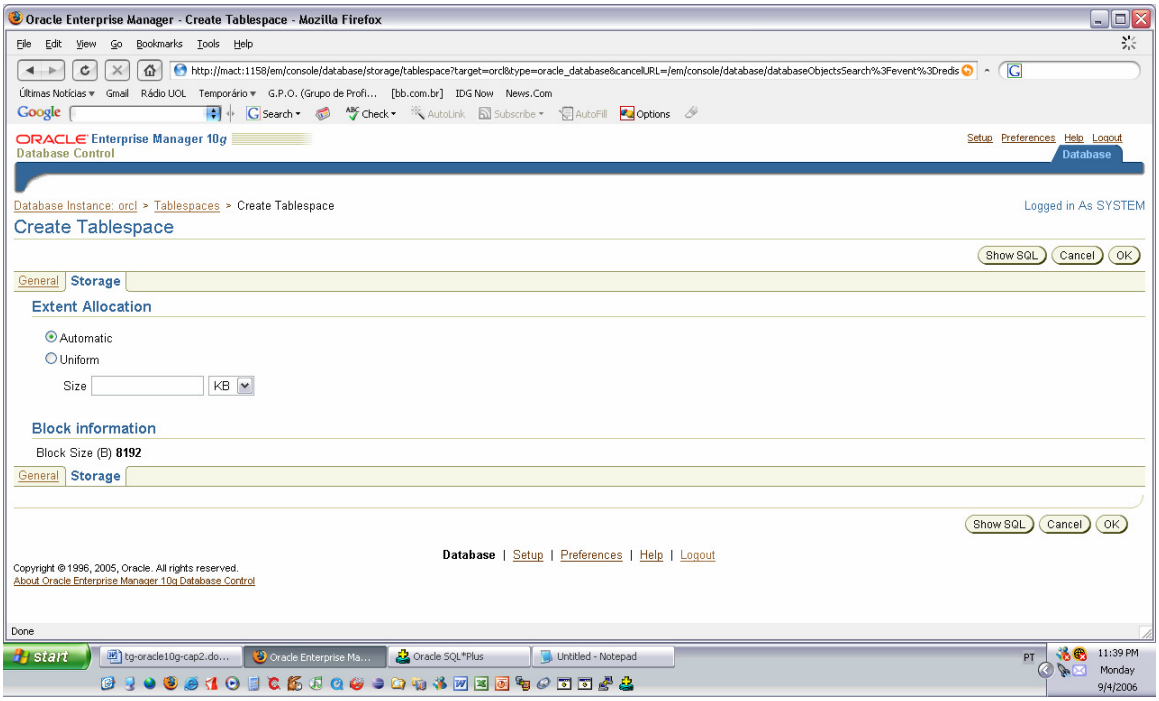


Figura 3.11- Definição da estratégia de alocação de espaço no tablespace de *undo*.

Um banco de dados pode ter mais de um tablespace de undo, porém apenas um pode estar ativo para a instância em qualquer período de tempo.

Configurações como essas algumas vezes ocorrem quando o Oracle é configurado sobre mais de um disco e com um tablespace em cada um. Porém, esse formato não é muito comum. O ideal é que se tenha um único tablespace de undo grande o suficiente para tratar das cargas das transações.

O monitoramento do tablespace de undo por sua vez, é feito através de consultas as visões de desempenho dinâmico. O quadro abaixo mostra algumas dessas visões.

Quadro 3.26 – Visões utilizadas no monitoramento do tablespace de undo.

Visão	Descrição
DBA_UNDO_EXTENTS	Todos os segmentos de undo no banco de dados incluindo seus tamanhos, extensões, tablespaces em que residem.
V$UNDO_STAT	A quantidade de uso de undo para o banco de dados em intervalos de 10 minutos.
V$ROLLSTAT	As estatísticas dos segmentos de rollback, incluindo o tamano e status.
V$TRANSACTION	Contém uma linha para cada transação ativa para a instância.

Existem três tipos de dados em um tablespace de undo: ativos, expirados e os não utilizados.

Dados ativos ou não expirados são aqueles que fornecem consistência de leitura. Quando todas as consultas que precisam dos dados de undo se completam e o período de retenção desses dados expirou, os dados ativos se tornam expirados. Os dados expirados por sua vez ainda podem ser utilizados durante a execução de outros serviços do Oracle, como o flashback.

Os dados não utilizados é o espaço não utilizado do tablespace de undo.

Para prevenir erros, o tamanho do tablespace de undo, como dito anteriormente, deve ser grande o suficiente para se armazenar todas as imagens de todos os dados provenientes das transações ativas que ainda não foram confirmadas.

O tamanho correto do tablespace de undo pode ser dimensionado de maneiras diferentes, como destacado abaixo.

Método manual

O DBA pode estimar o tamanho do espaço de tabela de undo utilizando a seguinte fórmula:

Quadro 3.27 – Cálculo para estimar o tamanho do tablespace de undo

```
undo_tablespace = UR * UPS + overhead
```

Na formula, UR, equivale em segundos à retenção dos dados (especificado no parâmetro de inicialização UNDO_RETENTION), UPS é o número máximo e blocos de undo utilizados por segundo e o overhead é um número muito pequeno em relação ao total e representa os metadados de undo.

Undo Advisor

O undo advisor é um recurso que automatiza a parte do ajuste do espaço para o tablespace de undo. Ele pode ser utilizado por meio da interface do EM ou através dos pacotes PL/SQL de forma semelhante ao Segment Advisor discutido na seção 3.3.6. Abaixo, nas figuras 3.12 e 3.13 ilustramos algumas etapas do gerenciamento do undo advisor na ferramenta EM.

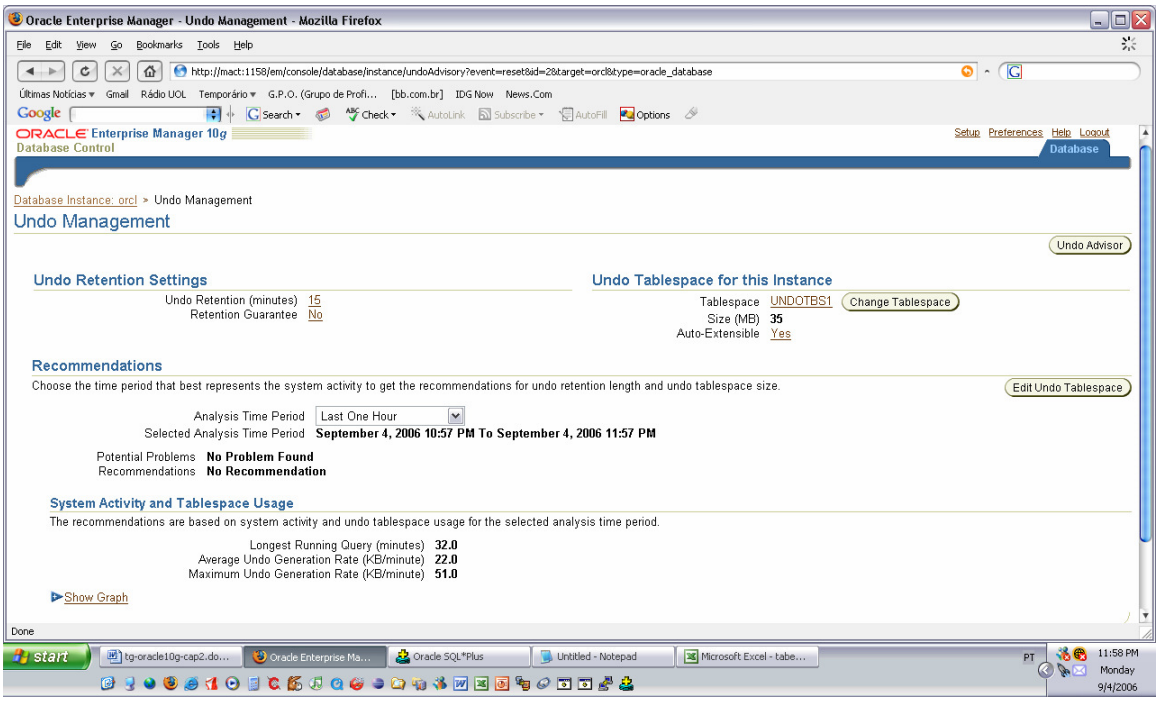


Figura 3.12 – Interface principal da ferramenta Undo Advisor no Enterprise Manager

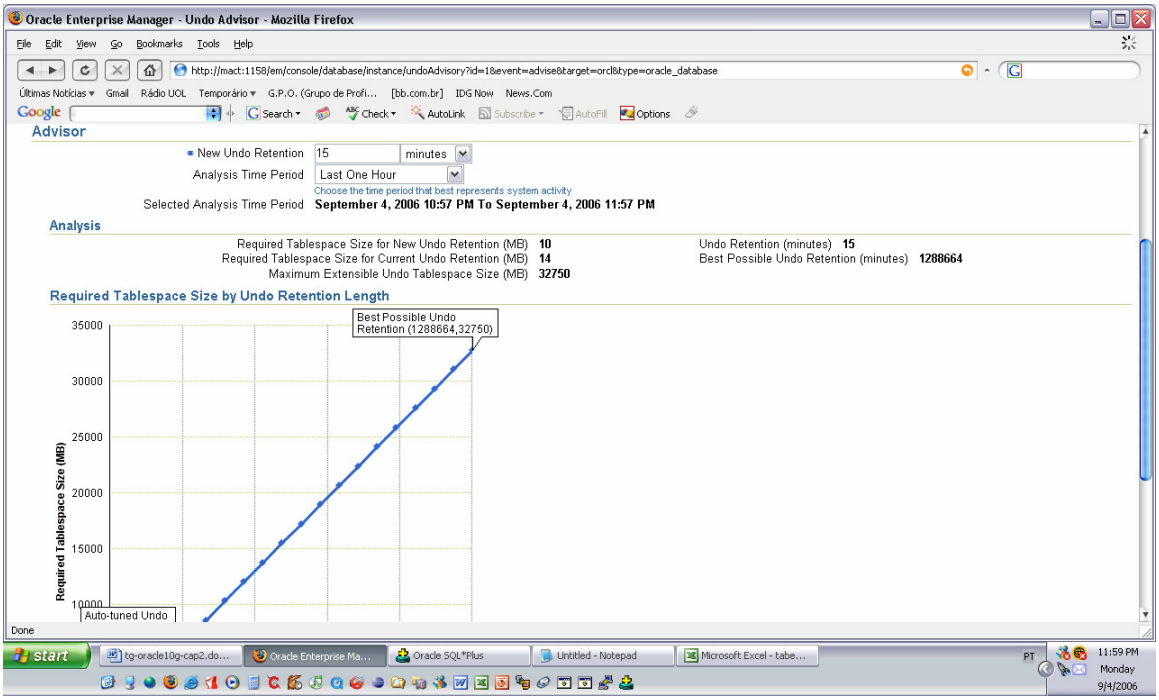


Figura 3.13 - Interface da ferramenta Undo Advisor no Enterprise Manager

4 Ajuste de banco de dados

Nenhum ambiente dispõe de uma capacidade infinita de computação. O que deve acontecer é que os desenvolvedores, juntamente com o DBA, durante o ciclo de vida da aplicação tentem minimizar essas limitações físicas das aplicações (tamanho da memória, tempo de resposta das consultas) e fazer com que elas não afetem o desempenho dos negócios.

O ajuste do desempenho do SGBD é, portanto, parte integrante desse processo. No capítulo anterior foram discutidas atividades relacionadas ao planejamento e gerenciamento dos objetos do banco de dados. Neste capítulo será abordado o ajuste de alguns tópicos, categorizados por áreas, e de que forma eles podem influenciar no desempenho das aplicações.

4.1 Ajuste de SQL

O tempo de resposta das consultas SQL está diretamente relacionado com a eficiência das aplicações. Desse modo, o DBA deve estar envolvido na revisão das consultas SQL escritas pelos desenvolvedores.

O passo mais importante para o ajuste de SQL é minimizar o caminho de pesquisa que o SGBD utiliza para realizar as consultas. A maioria das tabelas no Oracle possui um identificador de linha, o rowId. Esse identificador possui informações sobre a localização física da linha (arquivo, o bloco do arquivo e a linha do bloco). Quando uma consulta é executa sem seleções, ou seja, sem a cláusula where, o banco realiza uma varredura completa na tabela (full table scan), lendo cada bloco da tabela.

Quando se deseja selecionar linhas específicas, o SGBD pode utilizar índices que auxiliem e acelerem a recuperação dessas linhas. O que o índice faz é mapear valores lógicos em uma tabela para os seus rowId.

O índice também pode ser utilizado para se mapear várias colunas, o chamado índice concatenado.

O DBA Oracle também deve estar preocupado com a ordenação dos dados da tabela. É fundamental que essas linhas das tabelas sejam ordenadas,

caso seja freqüente a utilização de consultas por intervalo (seleção de valores em um intervalo especificado). Neste caso, a ordenação pode ter um impacto fundamental no desempenho, uma vez que menos dados são lidos para a resolução da consulta.

4.1.1 Impacto do ordenamento sobre a carga de dados

Em tabelas indexadas, a ordenação das linhas da tabela pode ter um impacto substancial (50%) sobre a melhoria de desempenho nas operações de inserção.

Quando um índice aumenta, o Oracle aloca novos blocos. Quando há um ordenamento, a nova entrada do índice é adicionada ao último bloco no índice. Caso esse bloco esteja cheio, há uma alocação de espaço para um novo bloco e o novo índice é inserido nesse bloco. Há, portanto, pouco impacto sobre o desempenho nessa operação.

O que ocorre quando a tabela está desordenada é que quando há a inserção de uma nova entrada de índice, essas novas entradas são gravadas em blocos de nós de índice já existentes. Se houver pouco espaço nesse bloco, as entradas desse bloco serão divididas em duas. Metade será mantida no bloco original e a outra metade será gravada no novo bloco. Essa operação é bastante custosa e como conseqüência, o desempenho piora durante as cargas e as consultas – devido ao fato do índice conter mais espaço não utilizado, exigindo que mais blocos seja lidos para o mesmo número de leituras de entrada.

4.1.2 Clusters

Quando duas tabelas são frequentemente consultadas ao mesmo tempo, o uso de cluster pode ser a melhor alternativa para se melhorar o desempenho. Os clusters armazenam linhas a partir de múltiplas tabelas nos mesmos blocos de dados físicos, com base na chave do cluster.

Um cluster de hash armazena uma linha em um local específico com base no seu valor na coluna da chave de cluster. Quando uma linha vai ser

inserida, o valor da sua chave é utilizado para determinar em qual bloco ele deve ser armazenado. Isso facilita a resolução das consultas, devido ao fato da análise dessa chave levar ao bloco que precisa ser recuperado.

Os clusters de hash são bastante úteis para se melhorar o desempenho das consultas por equivalência - consultas nas quais o valor de uma coluna é comparado com um valor exato.

4.1.3 Planos de explicação

Já foi discutido que a chave para o ajuste de SQL é minimizar o caminho de pesquisa que o SGBD utiliza para realizar as consultas. O DBA determina esse caminho por meio do comando EXPLAIN PLAN.

O comando EXPLAIN PLAN avalia o caminho da execução da consulta e coloca sua saída na tabela PLAN_TABLE. Como exemplo, pode-se observar a consulta abaixo:

Quadro 4.1 – Comando para geração do plano de consulta

```
explain plan
for
select contato_id from contato
where nome like '%M';
```

O comando, portanto, informa ao banco que ele deve explicar o seu plano de execução para a consulta. Opcionalmente, pode-se rotular esse plano de explicação na tabela PLAN_TABLE através da cláusula set statement_ID.

A tabela de plano deve estar disponível para o usuário que deseja executar este comando. Durante a instalação, o Oracle cria o script necessário para a criação da tabela – o arquivo é o utlxplan.sql.

A consulta à tabela PLAN_TABLE deve ser realizada através do procedimento DBMS_XPLAN.

Quadro 4.2 – Seleção do plano de execução

```
select * from table (DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

--------------------------------------------------------------------------------
| Id  | Operation                    | Name          | Rows  | Bytes | Cost  |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |               |     1 |    40 |     1 |
|   1 |  TABLE ACCESS BY INDEX ROWID| CONTATO       |     1 |    40 |     1 |
|   2 |   INDEX RANGE SCAN           | CONTATO_INDEX |     1 |       |     1 |
--------------------------------------------------------------------------------
```

O plano de execução, portanto, informa que operações o SGBD teve de realizar para resolver a consulta.

No plano mostrado anteriormente, pode-se observar que a cada passo um “custo” é atribuído. Os valores de custo devem ser analisados pelo DBA e, assim, identificar que passo contribui com o maior custo para a consulta. Desse modo, o DBA pode tentar contornar o problema através de modificações de SQL, alterações no projeto do banco de dados, entre outros.

Em alguns casos, índices são criados no SGBD e não são utilizados, o que não ocorre no exemplo mostrado nos quadros 4.1 e 4.2. Através do plano de execução, o DBA pode ter uma idéia de quais índices estão sendo subutilizados.

4.2 Ajuste do uso de memória

O DBA Oracle pode se fazer valer de um conjunto de ferramentas STATSPACK, que será discutido mais adiante, e de outras tabelas de dicionário de dados para identificar áreas problemáticas na alocação de memória do banco de dados.

Quando a memória está corretamente dimensionada, o Oracle mantém os dados acessados com mais freqüência na memória, isso reduz a quantidade de leituras físicas, maximizando, assim, o desempenho.

Através da visão V$SQL, o DBA pode observar o número de leituras lógicas e físicas para cada consulta atualmente no shared pool, bem como o número de vezes que cada consulta foi executada.

O script a seguir mostra um exemplo de consulta à visão V$SQL que exibe as consultas no shared pool que mais utilizam E/S.

Quadro 4.3 – Consultas no shared pool que mais utilizam E/S

```
select buffer_gets, disk_reads, executions,
       buffer_gets/executions b_e,
       SQL_text
from V$SQL
order by disk_reads desc;
```

A SGA, como estudada anteriormente, é composta por subáreas, dentre as quais se destaca o cache de buffer e a shared pool.

O cache de buffer, por sua vez, também é subdividido em múltiplas áreas:

- DEFAULT cache

  Área da memória destinada para os objetos que utilizam o tamanho padrão de bloco do SGBD;

- KEEP cache

  Área da memória destinada aos objetos que devem sem mantidos em memória todas as vezes;

- RECYCLE cache

  Área dedicada a objetos que devem ser removidos da memória rapidamente;

- BLOCK-SIZE-SPECIFIC caches

  Para cada tamanho não padrão de bloco do SGBD, deve haver um cache para armazenar objetos.

Por padrão, a tabela utiliza o pool DEFAULT, mas o DBA pode especificar um pool para a tabela diferente do padrão através do parâmetro buffer_pool dentro da cláusula storage da tabela.

No Oracle 10g, o DBA pode utilizar o Automatic Shared Memory Management (ASMM) [ASMM]. A ativação desse recurso é feita através da

especificação de um valor diferente de zero para o parâmetro de inicialização SGA_TARGET (todos os caches adicionados). Os outros parâmetros relacionados ao cache como DBA_CACHE_SIZE, SHARED_POOL_SIZE e LARGE_POOL_SIZE servirão apenas como os limites mais baixos para o algoritmo de ajuste automático de memória.

O monitoramento do tamanho dos caches pode ser feito através da visão V$SGASTAT.

4.2.1 Ajustando o tamanho da SGA

O ajuste da SGA pode ser automático, conforme discutido anteriormente, ou manual.

Para a configuração manual, o DBA deve configurar o parâmetro SGA_MAX_SIZE, que define o tamanho da SGA.

Na configuração manual, o DBA pode configurar os tamanhos para os caches individualmente. Essa configuração pode ser realizada dinamicamente por meio do comando alter system. No quadro 4.4 são listados os parâmetros que podem ser especificados pelo DBA.

Quadro 4.4 – Parâmetros de memória

Parâmetro	Descrição
SGA_MAX_SIZE	O tamanho máximo da SGA
SHARED_POOL_SIZE	O tamanho do pool compartilhado
DB_BLOCK_SIZE	Esse será o tamanho padrão de bloco do banco de dados
DB_CACHE_SIZE	O tamanho do cache especificado em bytes
DB_nK_BLOCK_SIZE	Se utilizar múltiplos tamanhos de bloco do banco de dados dentro de um único banco de dados, deve-se especificar um valor do parâmetro DB_CACHE_SIZE e pelo menos um valor de DB_nK_BLOCK_SIZE. Por exemplo, se o tamanho padrão de bloco do banco de dados for 4K, pode-se especificar um cache para os espaços de tabela com tamanho de bloco 8K por meio do parâmetro DB_8K_BLOCK_SIZE

Quando as áreas da SGA são corretamente dimensionadas, o desempenho das consultas e do SGBD como um todo pode ser dramaticamente melhorado.

A shared pool deve ser suficientemente grande o suficiente para armazenar as consultas mais frequentemente solicitadas.

O DBA pode fazer consultas à visão V$SHARED_POOL_ADVICE para ter uma idéia de qual seria o tamanho ideal da shared_pool. O script abaixo exemplifica uma possível consulta a essa visão.

Quadro 4.5 – Estimando o tamanho do shared pool

```
set lines  100
set pages  999

column   c1    heading 'Pool |Size(M)'
column   c2    heading 'Size|Factor'
column   c3    heading 'Est|LC(M)  '
column   c4    heading 'Est LC|Mem. Obj.'
column   c5    heading 'Est|Time|Saved|(sec)'
column   c6    heading 'Est|Parse|Saved|Factor'
column   c7    heading 'Est|Object Hits'   format 999,999,999


SELECT
   shared_pool_size_for_estimate c1,
   shared_pool_size_factor       c2,
   estd_lc_size                  c3,
   estd_lc_memory_objects        c4,
   estd_lc_time_saved                  c5,
   estd_lc_time_saved_factor           c6,
   estd_lc_memory_object_hits          c7
FROM
   v$shared_pool_advice;
```

Pool Size(M)	Size Factor	Est LC(M)	Est LC Mem. Obj.	Saved (sec)	Saved Factor	Est Object Hits
48	.5	48	20839	1459645	1	135,756,032
64	.6667	63	28140	1459645	1	135,756,101
80	.8333	78	35447	1459645	1	135,756,149
96	1	93	43028	1459645	1	135,756,253
112	1.1667	100	46755	1459646	1	135,756,842
128	1.3333	100	46755	1459646	1	135,756,842
144	1.5	100	46755	1459646	1	135,756,842
160	1.6667	100	46755	1459646	1	135,756,842
176	1.8333	100	46755	1459646	1	135,756,842
192	2	100	46755	1459646	1	135,756,842

Figura 4.1 – Estimativa dos valores da shared pool

Os resultados na figura 4.1 mostram uma variação no tamanho da shared_pool e algumas estatísticas que dão ao DBA uma boa noção do tamanho ideal do pool.

4.3 Ajuste do acesso a dados

4.3.1 Identificando linhas encadeadas

Para cada bloco de dados, o Oracle reserva dentro do bloco um espaço livre especificado pelo parâmetro pctfree. Esse espaço é utilizado quando o comprimento das linhas já armazenadas no bloco é estendido via o comando update.

Quando esse espaço é insuficiente, ou a linha desse bloco é movida para outro bloco ou essa linha é encadeada para um novo bloco de dados.

O encadeamento está diretamente ligado ao desempenho, uma vez que o Oracle precisa realizar um número maior de leituras físicas para retornar os dados pertencentes a uma mesma linha lógica.

O encadeamento pode ser evitado através da configuração correta do parâmetro *pctfree*.

O Oracle tem por padrão o valor 10 para esse parâmetro. O DBA deve então, durante a criação dos segmentos, aumentar esse valor, principalmente se a aplicação frequentemente atualizar valores nulos para valores não nulos.

O DBA pode utilizar o comando analyse para coletar informações sobre os objetos no banco. O comando analyse tem uma opção que informa as linhas encadeadas em uma tabela.

Quadro 4.6 – Linhas encadeadas em uma tabela

```
      analyse table nome_da_tabela list chained rows into
CHAINED_ROWS;
```

Para que esse comando seja realizado com sucesso, a tabela CHAINED_ROWS deve existir no esquema do usuário. O Oracle, durante a instalação, cria alguns scripts. O script para a criação dessa tabela está no arquivo utlchain.sql.

Um consulta a essa tabela mostrará as linhas da tabela que estão encadeadas. Caso esse número seja alto, é fundamental que a tabela seja reconstruída com um valor mais alto para o pctfree.

4.3.2 Tabelas organizadas por índice

Nas tabelas organizadas por índice (Index-organized tables - IOT), a linha inteira da tabela é armazenada no índice, ao contrário de valores chave especificados para uma linha, como acontece na criação de um índice normal. IOT são na verdade índices.

Nas IOT, as linhas são armazenadas e classificadas pelos valores da chave primária e, ao contrário do que ocorre com a combinação tabela/índice em que um acesso baseado no índice requer um acesso de tabela, nas IOT somente ela é acessada.

As IOT oferecem recursos adicionais que servem de suporte para a maximização do desempenho:

- Overflow area – o DBA pode configurar para que os dados da chave primária sejam armazenados separadamente dos da linha. Se os dados das linhas excederem o espaço disponível, eles serão movidos dinamicamente para uma área de estouro (overflow). O ideal é que essa área de estouro esteja em um espaço de tabela separado, o que ajuda a distribuir a E/S associada com a tabela;
- Secondary indexes – O DBA pode criar índices secundários nas IOT. Nesse caso, o Oracle utilizará os valores da chave primária como rowId lógicos para as linhas; e

- Reduced storage requirements – Na relação tabela/índice, os valores da chave são armazenados em dois lugares. Nas IOT isso não acontece, reduzindo os requisitos de armazenamento.

Para a criação de uma IOT deve-se utilizar a cláusula organization index durante a criação da tabela.

A IOT, da mesma forma que ocorre com índices, pode se tornar fragmentada. Em geral, é recomendado pela Oracle que se evite a utilização de IOT se os dados forem mais longos que 75% do tamanho do bloco. Quanto maior o comprimento das linhas e quanto mais transações são realizadas em um IOT, mais freqüentemente ela precisará ser reconstruída. Para reconstruir uma IOT deve ser utilizada a cláusula move do comando **ALTER TABLE**.

Quadro 4.7 – Reconstrução de uma IOT

```
alter table EMPREGADO_IOT
move tablespace DATA
overflow tablespace DATA_OVERFLOW
```

4.4 Ajuste do armazenamento físico

A E/S deve ser balanceada pelo maior número de dispositivos possível. Por padrão, o DBA deve seguir a solução SAME Stripe and mirror everything [SAME]. Os limites de throughput de E/S são os limites-chave a serem vencidos. Assim, distribuir E/S por vários discos permite tirar proveito dos throughputs combinados dos vários discos.

5 Oracle Grid

A idéia central da computação em grade (*grid computing*) é pensar a computação como uma utilidade, em outras palavras, para um usuário, consumidor de dados de uma aplicação, não deve interessar onde esses dados estão ou de que forma eles são recuperados ou processados, o que importa realmente é que ele possa requisitá-los quantas vezes for necessário.

Do ponto de vista computacional, o conceito de grid envolve alocação de recursos, compartilhamento de informações e alta disponibilidade. A alocação de recursos se refere à disponibilidade de recursos. Ela garante, por exemplo, que todas as requisições dos usuários estão conseguindo obter recursos necessários para o processamento. O conceito de compartilhamento de informações está relacionado com a distribuição da informação. Este compartilhamento garante que as informações que os usuários precisam estarão disponíveis quando e onde for preciso. Por fim, a alta disponibilidade garante que todo esse poder computacional está sempre disponível para requisições.

O Oracle 10g oferece um conjunto de ferramentas que dão suporte a *grid computing*. São elas: Oracle Application Server 10g – OAS [OAS], Oracle Database 10g, Oracle Collaboration Suíte 10g [OCS] e o Oracle Enterprise Manager.

A infra-estrutura que dá suporte a *grid computing* é construída baseada em dois conceitos:

- Implementação de um a partir de muitos – Construção de uma entidade lógica a partir de um *cluster* de máquinas. O objetivo dessa arquitetura é a distribuição do trabalho entre as máquinas do cluster, proporcionando ao grid uma maior disponibilidade, escalabilidade e desempenho. Nesse contexto fazem parte as ferramentas Oracle Application Server 10g e o Oracle Database 10g.

- Administrar muitos como um – A administração do grid é realizada através de uma entidade lógica. A administração do grid no Oracle é realizada através da ferramenta Oracle Enterprise Manager 10g Grid Control.

Apesar deste capítulo tratar da arquitetura do grid Oracle e das ferramentas que dão suporte ao grid no Oracle 10g, o texto focará o Oracle Database 10g. Será discutido principalmente como SGBD Oracle 10g dá suporte à arquitetura grid.

5.1 Oracle Application Server 10g

O Oracle Application Server 10g (OAS) é ferramenta desenvolvida pela Oracle para a execução de aplicações no ambiente grid.

O OAS consiste de quatro componentes principais: Oracle Container para J2EE (OC4J) [OC4J], plataforma para o desenvolvimento e publicação de aplicações corporativas escritas em Java; Oracle JDeveloper [OJD], um ambiente de desenvolvimento de aplicações Java integrado; Oracle Application Development Framework (ADF) [OADF], framework embutido no JDeveloper baseado no design pattern Model-View-Controller; e o Oracle TopLink [OTL] que provê uma camada de persistência que simplifica a forma como as aplicações J2EE acessam dados através de mapeamento objeto-relacional.

O Application Server possui algumas características que merecem destaque em relação à execução de aplicações em ambientes grid.

- *Software Provisioning*

  Em um ambiente de computação em grid típico existe uma quantidade ilimitada de servidores. A instalação e a manutenção manual de softwares nesses servidores é uma atividade praticamente inviável. O Application Server fornece meios de se automatizar esse processo.

- User Provisioning

O Application Server oferece uma ferramenta de gerenciamento de usuários (Security Management Console), utilizada na criação de usuários, papéis e privilégios, e de certificados (Certification Authority), utilizada para certificar usuários. O controle dos usuários nas aplicações é realizado através de um login único (Single Sign On).

- Gerenciamento da carga de trabalho

O Application Server possui o Workload Management vem integrado ao e isso possibilita ao DBA o gerenciamento da carga de trabalho fornecendo uma maior escalabilidade e disponibilidade.

5.2 Oracle Database 10g

O SGBD Oracle 10g dá suporte a arquitetura grid de diversas maneiras:

- Real Application Clusters (RAC)

Um SGBD RAC é altamente escalonável e disponível. O RAC distribui carga de trabalho do SGBD entre as múltiplas instâncias do grid. Desse modo, uma falha em um dos nós do *cluster* não afeta as sessões dos clientes. O que acontece durante a falha de um dos nós é apenas um aumento do tempo de resposta para as transações dos clientes.

Uma instância RAC é, de várias formas, diferente de uma instância standalone. Essas diferenças se refletem em alguns parâmetros de inicialização específicos em instâncias RAC e em visões de dicionários.

O arquivo de parâmtro do servidor (SPFILE), em um ambiente RAC, é compartilhado entre as múltiplas instâncias. Dentro do SPFILE o DBA pode atribuir valores diferentes para cada instância.

Por exemplo, se um parâmetro de inicialização for o mesmo para todas as instâncias do *cluster*, ele deve ser prefixado como '*'. Caso contrário, o parâmetro deve ser prefixado com o nome do nó.

No exemplo do quadro 5.1 é ilustrada a modificação do tamanho do *shared pool* para uma instância específica:

Quadro 5.1- Modificação to tamanho do shared pool em um ambiente RAC

```
SQL> select sid, name, value
2     from v$spparamter where name = 'shared_pool_size';

SID      NAME              VALUE
-------  ----------------  -----------
*        shared_pool_size     65614720

SQL> alter system set shared_pool_size = 48M
2     scope=spfile sid='rac2';


SQL> select sid, name, value
2     from v$spparamter where name = 'shared_pool_size';

SID      NAME              VALUE
-------  ----------------  -----------
*        shared_pool_size     65614720
rac2  shared_pool_size  50331648
```

Alguns parâmetros de inicialização são utilizados apenas em um ambiente RAC. Mesmo que esses parâmetros existam em qualquer instância, em um ambiente que não seja RAC, eles são nulos.

No quadro 5.2 são mostrados os parâmetros de inicialização relacionados ao RAC.

Quadro 5.2 – Parâmetros de inicialização do RAC

Parâmetro de Inicialização	Descrição
INSTANCE_NUMBER	Número único identificando a instância do cluster.
INSTANCE_NAME	Nome único dessa instância desntro do cluster, em geral, o nome do cluster com um sufixo numérico
CLUSTER_DATABASE	TRUE se a instância estiver participando de um ambiente RAC
CLUSTER_DATABASE_INSTANCES	Número de instâncias configurado para esse cluster
ACTIVE_INSTANCE_COUNT	Especifica a instância primária em um cluster de dois nós, do contrário, é o número de instâncias do cluster
MAX_COMMIT_PROPAGATION_DELAY	Controla a velocidade com que as transações confirmadas são propagadas para outros nós

Em um ambiente de uma única instância, todas as visões de desempenho dinâmico que iniciam com V$ têm uma visão correspondente que inicia com GV$, com a coluna adicional INST_ID sempre configurada como 1. Já em um ambiente RAC, o número de linhas da tabela é multiplicado de acordo como o número de instâncias do cluster.

- ASM

O ASM, como exposto em tópicos anteriores, possibilita um fácil gerenciamento e balanceamento da carga de E/S.

5.3 Oracle Enterprise Manager Grid Control

O *Oracle Enterprise Manager Grid Control* [OEMGC] provê um framework completo para administração e monitoramento do grid.

O *Grid Control* fornece ao DBA, por meio de uma interface web uma visão de todo o *grid*, incluindo seu status e alertas de pontos problemáticos da estrutura, por exemplo.

Ele consiste de três componentes principais:

- Oracle Management Repository (OMR)
- Um ou mais Oracle Management Service (OMS)
- Um ou mais Oracle Management Agent (OMA)

Esse componentes são ilustrados na figura 5.1

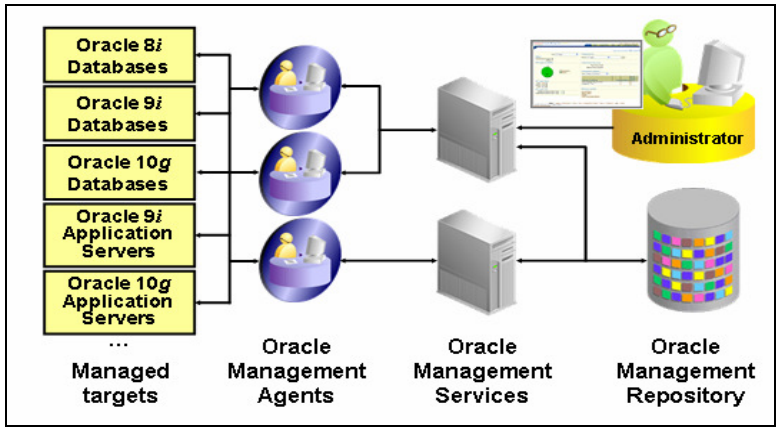


Figura 5.1 – Componentes do *Grid Control.* Fonte: [OEMSG05]

5.3.1 Oracle Management Agent - OMA

É o responsável pelo gerenciamento e comunicação entre o *Grid Control* e todos os alvos gerenciados (*managed targets*).

Alvos gerenciados são as entidades controladas pelo *Grid*. Os alvos podem ser adicionados ou removidos do *Grid Control* conforme o necessário. São exemplos de alvos:

- Oracle Databases;
- Oracle Database Listeners;
- Oracle Application Servers;
- Oracle Applications; e
- Oracle Collaboration Suíte.

O OMA é ilustrado na figura 5.2.

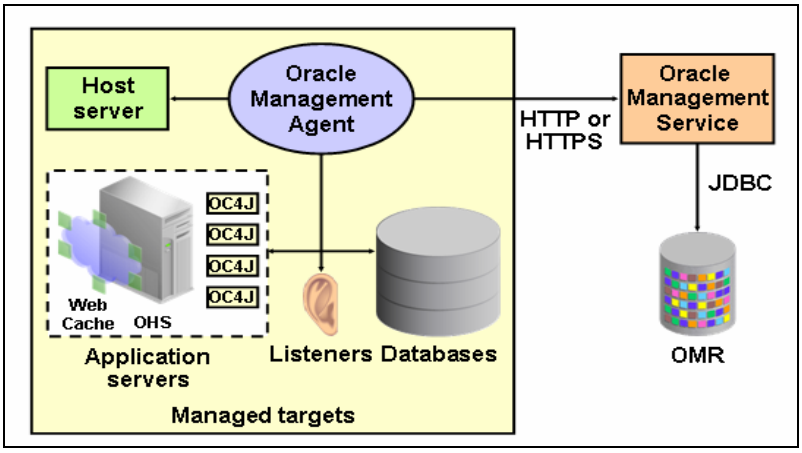


Figura 5.2 – Oracle Management Agent. Fonte: [OEMSG05]

Os alvos mais comuns são predefinidos como parte nativa do produto. Porém outros alvos podem ser personalizados para o controle através de uma API aberta fornecida pelo *Grid Control*.

Existe um OMA instalado em cada servidor e ele é responsável pelo gerenciamento de todos os seus alvos. O OMA executa tarefas como: a coleta de informações sobre a disponibilidade, configuração e performance do alvo. Essas informações são comunicadas ao OMS através do protocolo http.

5.3.2 Oracle Management Service - OMS

O OMS é uma aplicação web, escrita em Java (J2EE), que roda sobre o Oracle Application Server para o controle do grid.

O OMS é constituído de três componentes principais:

- Oracle HTTP Server (OHS) [OHS];
- Oracle Application Server Container para J2EE (OC4J); e
- OracleAS Web Cache [OASWC].

Um esquema que representa o OAS é apresentado na figura 5.3.

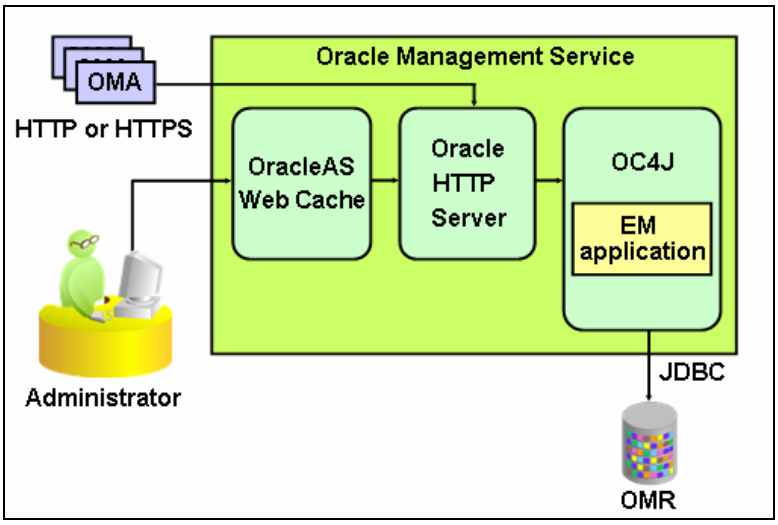


Figura 5.3 – Oracle Management Service. Fonte: [OEMSG05]

Pela figura, pode-ser perceber que o DBA se conecta ao *Grid Control* através do OHA e do OracleAS Web Cache.

O OMS periodicamente armazena informações recebidas pelos agentes no OMR.

5.3.3 Oracle Management Repository - OMR

O OMR reside no SGBD Oracle. O repositório é constituído de aproximadamente 4.000 objetos que são armazenados em dois *tablespaces* pertencentes ao esquema SYSMAN.

Esses objetos contêm informações sobre os administradores do *Grid Control*, alvos e aplicações gerenciadas por ele.

5.4 Grid Control Console

Nas figuras 5.3, 5.4, 5.5, 5.6 e 5.7 são ilustradas algumas telas do *Grid Control*.

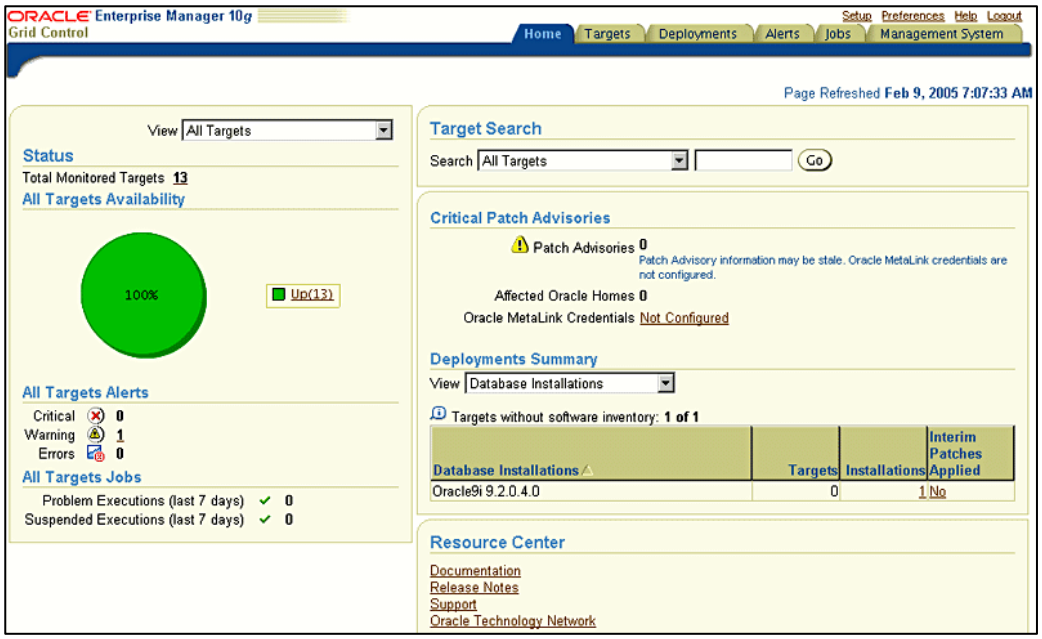


Figura 5.3 - Grid Control Console: Home

A tela inicial do Grid Control apresenta ao DBA sub-abas para o controle do grid como alvos, distribuições, alertas e jobs.

Através de um gráfico pizza o Grid Control ilustra a disponibilidade dos alvos gerenciados.

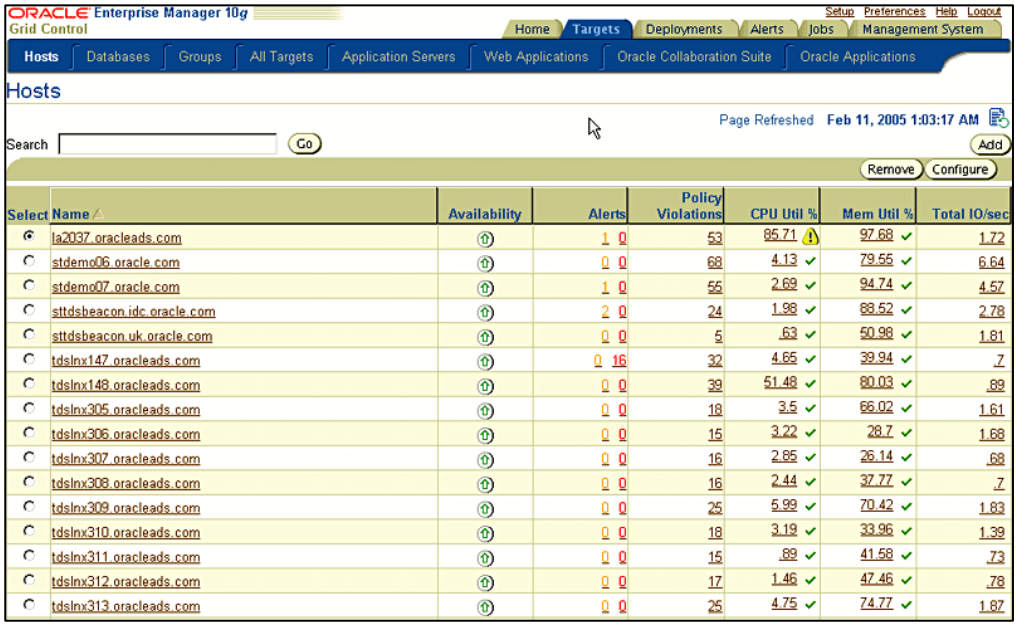


Figura 5.4 - Grid Control Console: Targets

A aba alvos mostra todos os alvos monitorados pelo *Grid Control*. A partir dessa tela o DBA pode tomar ações sobre cada alvo específico.

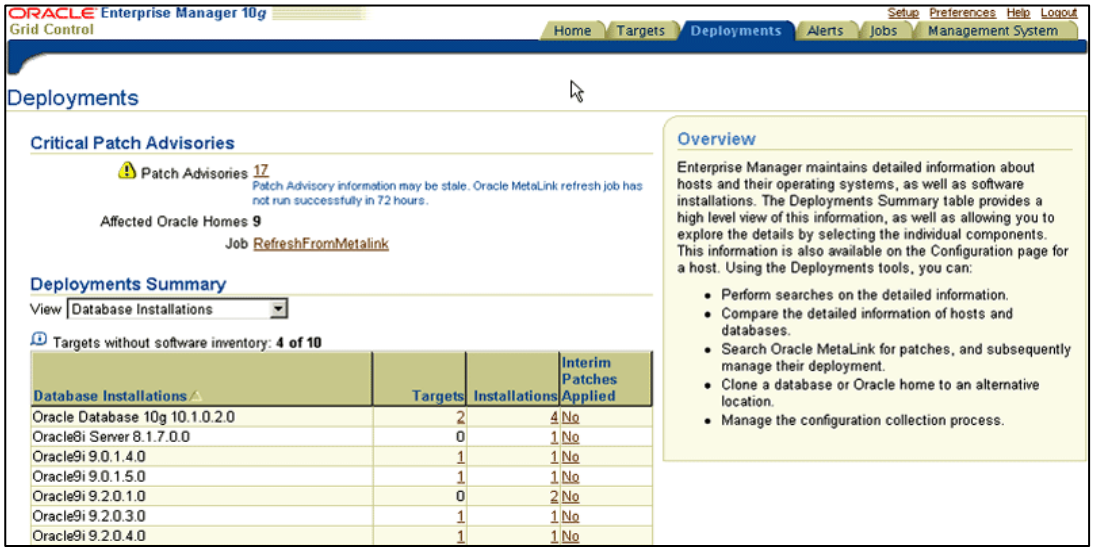


Figura 5.5 - Grid Control Console: Deployments

Em grandes grids, a configuração é uma tarefa que consome muito tempo do DBA. O *Grid Control* facilita um pouco a configuração através da sessão *deployments* do Grid Control.

Nesta sessão o DBA pode controlar as configurações dos servidores, aplicar *patches* às aplicações, clonar banco de dados, entre outros.

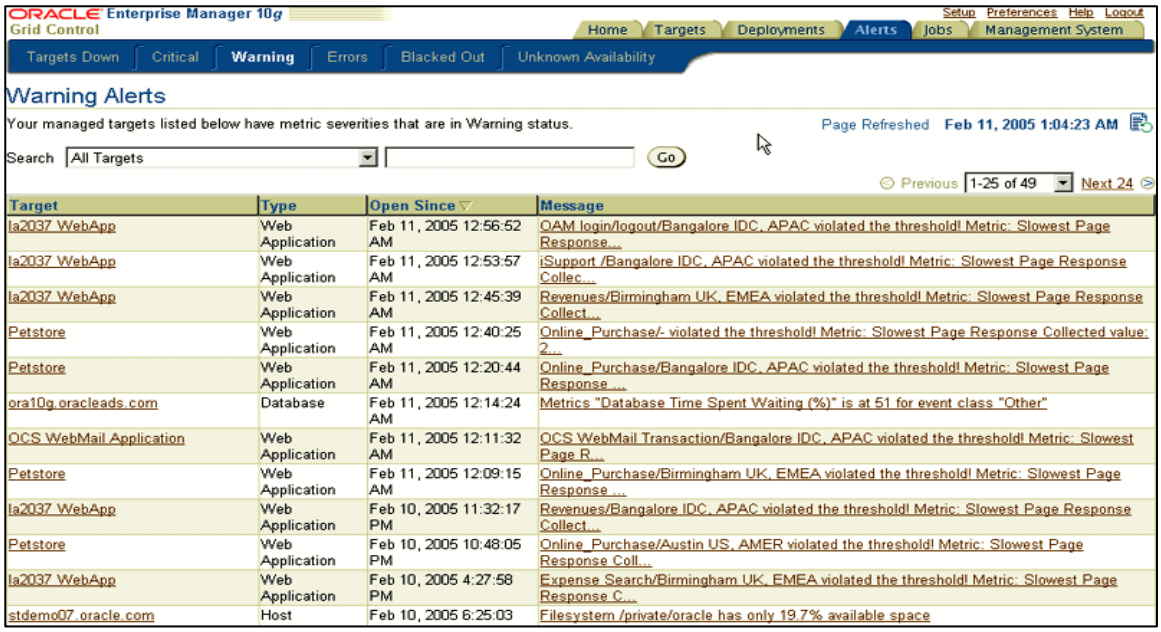


Figura 5.6 - Grid Control Console: Alerts

Pela página de alertas, o DBA é rapidamente informado sobre pendências no *grid*. Selecionando-se qualquer um alerta, o *Grid Control* leva o DBA para uma página de métricas e que métricas levaram a criação do alerta.

Ao selecionar o alvo, o *link* leva o DBA diretamente para a página de monitoramento do alvo.

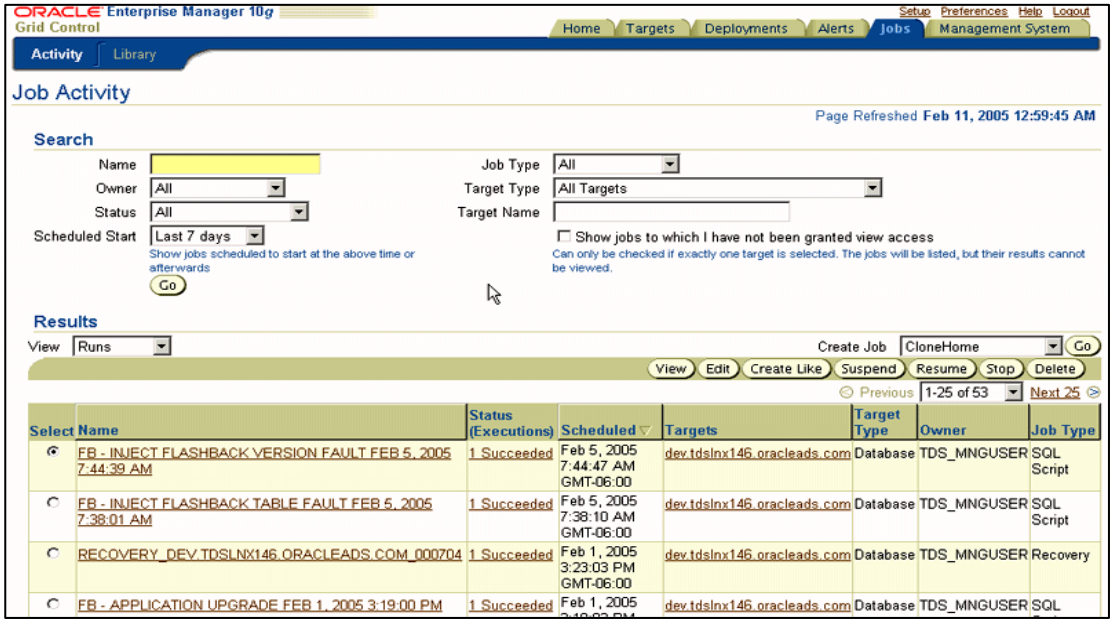


Figura 5.7 - Grid Control Console: Jobs

Jobs automatizam tarefas como backups de bancos, de estatísticas, etc. A página Jobs do Grid Control é um meio do DBA criar, remover ou editar jobs para o grid.

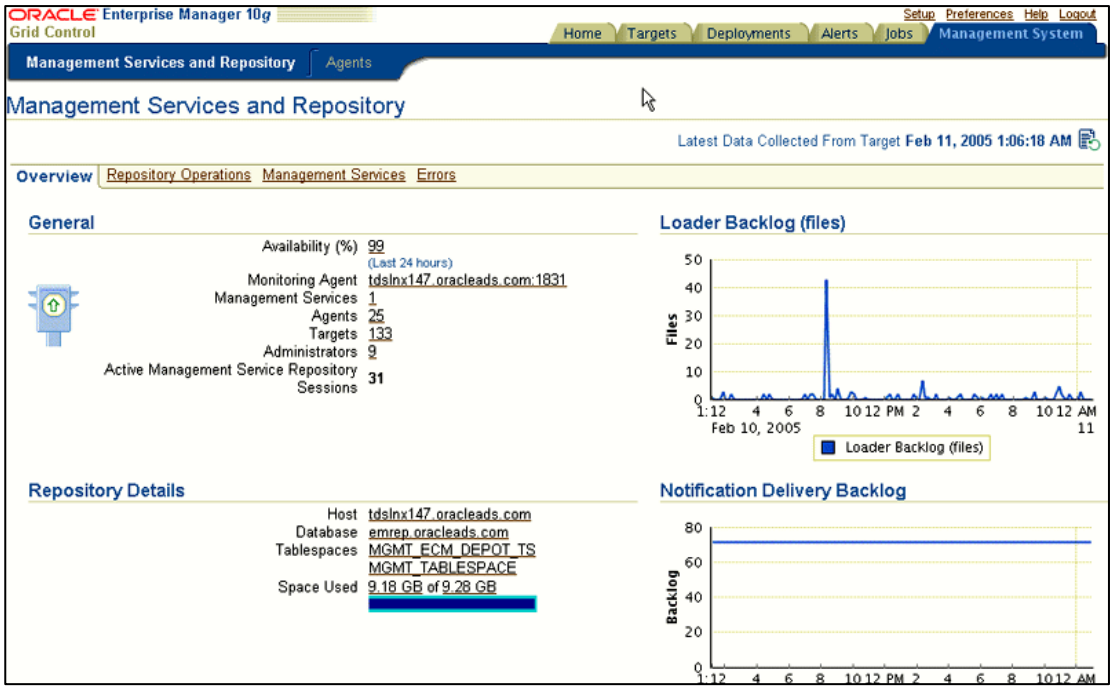


Figura 5.8 - Grid Control Console: Management System

A tela Managed System dá ao DBA uma visão da performance o status de toda a infra-estrutura grid, incluindo o *management agents*, *management services*, e o *management repository*.

6 Conclusão

Este trabalho realizou um estudo sobre os principais tópicos relacionados a administração do SGBD Oracle Database 10g. Inicialmente foram discutidos os principais conceitos relacionados à arquitetura Oracle, que serviu de base teórica durante o restante do trabalho.

Foram vistas técnicas de armazenamento e de monitoramento do uso de espaço que impactam diretamente na estabilidade e desempenho do SGBD. Para exemplificar tarefas relacionadas ao gerenciamento do SGBD foi utilizada a ferramenta de gerenciamento *Enterprise Manager*. Foram discutidas também algumas das principais soluções de ajuste do Oracle 10g de modo a maximizar a performance do mesmo.

Por fim, foi estudado o Oracle Grid, sua arquitetura e ferramentas de gerenciamento. Foram vistos tópicos como *Real Application Clusters,* estratégia que assegura a alta disponibilidade e eficiência do sistema.

Como trabalho futuro, a sugestão é a realização de testes de performance de modo demonstrar o ganho de desempenho com as técnicas apresentadas nesse trabalho e também um estudo semelhante sobre a administração do Oracle Grid.

7 Referências

[ASMM] Automatic Shared Memory Management. Endereço: http://www.oracle.com/technology/obe/obe10gdb/manage/memmgmt/memmgmt.htm

[BTREE] B*Tree. Endereço: http://www.orafaq.com/glossary/faqglosb.htm

[DAWES05] DAWES, Chip. BRYLA, Bob. OCA: Oracle 10g Administration Study Guide, 2005.

[GARTNER] Gartner Dataquest. Endereço: http://www.gartner.com/it/products/research/dataquest.jsp

[GPO] Grupo de Profissionais Oracle. Endereço: http://www.profissionaloracle.com.br/

[LONEY05] LONEY, Kevin. Oracle 10g. O manual do DBA, 2005.

[OADF] Oracle Application Development Framework. Endereço: http://www.oracle.com/technology/products/adf/index.html

[OAS] Oracle Application Server. Endereço: http://www.oracle.com/appserver/index.html

[OASWC] Oracle Application Server Web Cache 10g. Endereço: http://www.oracle.com/technology/products/ias/web_cache/index.html

[OCS] Oracle Collaboration Suite. Endereço: http://download-east.oracle.com/docs/cd/B19306_01/em.102/b16241/Collaboration_Suite_Management.htm

[OC4J] Oracle Application Server Containers for J2EE (OC4J). Endereço: http://download-east.oracle.com/docs/cd/B10467_16/tour/j2ee_oc4j.htm

[OEMDBC] Oracle Enterprise Manager Database Control Endereço: http://www.oracle.com/enterprise_manager/index.html

[OEMGC] Oracle Enterprise Manager 10g Grid Control. Endereço: http://www.oracle.com/technology/products/oem/index.html

[OEMSG] Oracle Enterprise Manager 10g Grid Control. Student Guide, 2005.

[OHS] Oracle HTTP Server. Endereço:

http://download-

east.oracle.com/docs/cd/B14117_01/server.101/b12255/overview.htm

[OJD] Oracle JDeveloper. Endereço:

http://www.oracle.com/tools/jdev_home.html

[OPS] Oracle Parallel Server. Endereço:

http://download-east.oracle.com/docs/cd/A87860_01/doc/paraserv.817/a76968/psintro.htm

[ORCL] Oracle. Endereço:

http://www.oracle.com

[ORCLAW] Oracle Database 10g: Administration Workshop, 2005.

[ORCLCONCEPTS] Oracle Database Concepts, 10g Relase 1.

[ORCL10G] Oracle Database 10g. Endereço:

http://www.oracle.com/technology/products/database/oracle10g/index.html

[ORCL2D] Oracle Database 2 Day DBA Course.

[OSECURITY] Oracle Advanced Security. Endereço:

http://www.oracle.com/technology/deploy/security/aso/index.html

[OTL] Oracle TopLink. Endereço:

http://www.oracle.com/tools/toplink_adf.html

[SAME] SAME. Endereço:

http://searchstorage.techtarget.com/tip/1,289483,sid5_gci929549,00.html

[SQL] SQL. Endereço:

http://www.w3schools.com/sql/default.asp

[TAURION04] TAURION, Cezar. Grid Computing. Um novo paradigma computacional, 20004.