Usuario X Squema no Oracle

Análise de Sistemas de Engenharia
Universidade Veiga de Almeida (UVA)
46 pag.

USUÁRIO X SCHEMA

Para estabelecermos acesso ao banco de dados ORACLE precisamos de um USER. Associado a cada usuário existe um SCHEMA.

DIFERENÇAS ENTRE USUÁRIO E SCHEMA

Usuário e Schema são duas características de um banco de dados ORACLE extremamente relacionadas e indissolúveis.

USUÁRIO

Podemos controlar o acesso a um banco de dados ORACLE criando, alterando, removendo e monitorando usuários.

Um USUÁRIO é:

- ◊ requerido para acesso ao banco de dados ;
- ◊ necessário para cada aplicação do banco de dados;
- ◊ definido no banco de dados;

O usuário está associado a acesso e segurança. Para que o ORACLE possa identificar e controlar o acesso aos objetos do banco, cada um de nós deve ter uma identificação que nos garanta privilégios para ler e/ou alterar informações no banco de dados. Esta identificação é o USER.

SCHEMA

Quando o DBA cria um novo usuário, o ORACLE, simultaneamente, cria um SCHEMA (de mesmo nome) para este usuário.

Um SCHEMA corresponde a uma coleção de objetos pertencentes ao usuário que possui o mesmo nome do SCHEMA.

O SCHEMA está associado a armazenamento. Para que o ORACLE possa identificar a quem pertence determinado objeto e, por conseguinte, controlar o uso dos dados deve haver uma identificação, para cada objeto. Esta identificação física é o SCHEMA.

OBJETOS DENTRO E FORA DE SCHEMA

Um banco de dados possui um número variado de objetos. Alguns destes objetos pertencem a usuários e outros não.

Neste tópico conheceremos os objetos subordinados a usuários e os objetos administrativos, ou seja, aqueles que podem ser utilizados pelos usuários sem estarem subordinados a eles.

OBJETOS ADMINISTRATIVOS

No grupo de objetos administrativos encontramos: TABLESPACES, USERS, PROFILES, ROLES, UNDO SEGMENTS e DIRECTORIES.

Estes objetos podem ser utilizados pelos diversos usuários do banco de dados e pertencem ao banco de dados como um todo. Nos próximos capítulos estaremos estudando a funcionalidade de cada um deles.

OBJETOS DO USUÁRIO

No grupo de objetos do usuário encontramos: TABELAS, TYPES, VIEWS, CLUSTERS, INDEXES, SNAPSHOTS, SEQUENCES, FUNCTIONS, PROCEDURES e PACKAGES.

Quando criamos um objeto de um dos tipos acima, obrigatoriamente, os subordinamos a usuários. Eles são criados nos SCHEMAS dos usuários e pertencem ao usuário dono do SCHEMA. Este usuário tem todos os privilégios sobre os objetos em seu SCHEMA.

CONTROLANDO O ACESSO E USO DO BANCO DE DADOS

A segurança do banco de dados gira em torno de permissão das ações do usuário no banco de dados e sobre os objetos dentro dele.

O modelo de segurança do ORACLE usa SCHEMA e privilégios para garantir o controle de acesso aos dados e para restringir o uso dos diversos recursos do banco de dados.

Os direitos de acesso de um usuário são controlados por diferentes características. Quando um DBA cria um novo usuário ele deve definir estes diversos aspectos, que envolvem:

◊Criação de username e password, com identificação de tablespace default e temporário.

◊Uma lista de tablespaces aos quais o usuário poderá ter acesso com as respectivas quotas de espaço para cada tablespace.

◊Os limites referentes aos recursos do sistema que o usuário poderá utilizar (tempo IDLE, tempo de processados, quantidade de I/Os, etc).

◊Identificação dos privilégios relativos a operações sobre o banco de dados e sobre outros objetos.

ALTERANDO O SCHEMA DA SESSÃO

Uma característica implementada na versão 9i, no comando Alter Session é a possibilidade de alterarmos o schema corrente (sem alteração do usuário corrente).

```
ALTER SESSION SET CURRENT_SCHEMA = HR;
```

A alteração de schema faz com que as referências feitas sem qualificação sejam direcionadas para o schema especificado no comando. Isto é uma forma prática de acesso a dados que estão em outro usuário.

Esta característica, no entanto, não dá privilégios especiais ao usuário. Para que o acesso seja possível, ele deve ter os privilégios necessários sobre o objeto desejado.

AUTENTICAÇÃO

A autenticação corresponde ao momento que o usuário solicita conexão ao banco de dados e este reconhece este como sendo um usuário válido para acesso ao banco de dados.

A autenticação pode ser feita :

◊pelo sistema operacional
◊por um serviço de rede
◊pelo banco de dados Oracle

Os três métodos podem ser usados para uma mesma instância do banco de dados.

Para garantir a segurança o Oracle criptografa as passwords durante a transmissão. Os DBAs necessitam de procedimentos de autenticação especiais uma vez que executarão operações especiais sobre o banco de dados.

Conheceremos a seguir cada um destes métodos de autenticação.

AUTENTICAÇÃO PELO SISTEMA OPERACIONAL

Alguns sistemas operacionais permitem que o Oracle use informações mantidas por ele para autenticação dos usuários.

Como benefícios deste método, temos:

◊O usuário pode estabelecer conxão ao Oracle sem especificar um USERNAME e PASSWORD. Por exemplo o usuário pode acionar o SQL*Plus e informar apenas /. O UNIX é um sistema operacional que suporta esta característica. Já o Windows não.

◊O controle sobre autorização do usuário está centralizado no sistema operacional. O Oracle não precisa gerenciar e armazenar PASSWORDS. Os USERS são mantidos no banco de dados.

◊As entradas na trilha de audit têm correspondência entre o usuário do sistema operacional e o do ORACLE.

AUTENTICAÇÃO PELA REDE

O ORACLE suporta diversos métodos de autenticação pela REDE, como veremos a seguir:

◊*THIRD PARTY-BASED AUTHENTICATION TECHNOLOGIES* – se houverem serviços de autenticação de rede tais como DCE, Kerberos ou SESAME, o ORACLE aceita a autenticação feita por este serviço.

Para que possamos utilizar serviços de autenticação de rede com o ORACLE, precisamos do Oracle Enterprise Edition com Oracle Advanced Security option.

◊*PUBLIC KEY INFRASTRUCTURE-BASED AUTHENTICATION* - autenticação baseada em uma sistema de criptografia com chave pública usam um certificado digital para os usuários (clientes), que usam-nos para autenticar diretamente para os servidores sem envolvimento de um servidor de autenticação.

O Oracle fornece uma infraestrutura de chave pública (PKI) para deste método, que consiste dos seguintes componentes:

- Gerenciamento da autenticação e de chave segura usando Secure Sockets Layer (SSL).
- Oracle Call Interface (OCI) e funções em PL/SQL para sinalizar os dados definidos pelo usuário utilizando chave privada e certificado e para verificar a assinatura dos ddados usando um certificado e validação.
- Oracle wallets que são estruturas de dados que contém uma chave privativa do usuário, um certificado do usuário e um conjunto de pontos de validação.
- Oracle Wallet Manager que proteje as chaves do usuário e gerencia certificados X.509v3 em clientes e servidores Oracle.
- Certificados X.509v3 que devemos obter de uma autoridade em certificação (não Oracle). Os certificados são carregados nos Oracle wallets para habilitar a autenticação.
- Directory-enabled Oracle Security Manager que fornece privilégios centralizados de gerenciamento para que a administração seja simplificada e o nível de segurança incrementado.

  Permite que armazenemos e recuperemos roles do Oracle Internet Directory ou de qualquer diretório compatível com o Lightweight Directory Access Protocol (LDAP).

- Oracle Internet Directory que é um diretório LDAP v3-compliant construído no Oracle. Ele permite que gerenciemos o usuário e o ambiente de configuração do sistema incluindo atributos de segurança e privilégios para usuários autenticados usando certificados X.509.

  O Oracle Internet Directory assegura controle de acesso a nível de atributo, sendo possível restringir os privilégios de leitura, gravação ou atualizado em atributos específicos para usuários específicos. Ele também suporta proteção e autenticação de directory queries através de criptografia SSL.

- Para usarmos este método descrito acima precisamos do Oracle Enterprise Edition com Oracle Advanced Security option.

◊*REMOTE AUTHENTICATION* – o Oracle suporta este tipo de autenticação através do Remote Dial-In User service (RADIUS), um protocolo padrão usado para autenticação, autorização de contabilização de usuários.

Para usarmos este método descrito acima precisamos do Oracle Enterprise Edition com Oracle Advanced Security option

AUTENTICAÇÃO PELO BANCO DE DADOS ORACLE

O ORACLE autentica os usuários que tentarem estabelecer conexão com o banco de dados utilizando informações armazenadas no próprio banco de dados.

A tempo de criação do usuário especificamos uma PASSWORD que deve ser fornecida a tempo de conexão e validada pelo Oracle. A PASSWORD está armazenada no banco de dados em um formato criptografado para maior segurança.

Desta forma, a informação mais preciosa e que deve ter maiores cuidados tanto no armazenamento quanto no fornecimento (por parte do usuário) é a PASSWORD.

Opções visando o aumento do nível de confidencialidade.

◊*PASSWORD ENCRYPTION WHILE CONNECTING* – O Oracle permite que criptografemos as passwords durante as conexões pela rede (cliente-servidor e servidor-servidor). Se habilitarmos esta funcionalidade em máquinas cliente e máquinas servidoras, o Oracle criptografa as paswords usando um algoritmo DES (Data Encryption Standards) modificado antes de enviá-la pela rede. Isto pode ser bastante útil em ambientes distribuídos.

◊*ACCOUNT LOCKING* – O Oracle poderá bloquear um usuário se houver falha na tentativa de conexão após um determinado número de tentativas. Podemos determinar que o usuário seja desbloqueado automaticamente após um determinado intervalo de termpo ou somente pelo DBA. Estas especificações são feitas com o objeto PROFILE (será visto no curso).

◊*PASSWORD LIFETIME AND EXPIRATION* – com esta característica podemos especificar um tempo de vida para as passwords. Após este período ela torna-se inválida e deve ser modificada antes da conexão ser autorizada. Na primeira tentativa de conexão após a password ter expirado o usuário entra em um período de "graça" e uma mensagem de aviso é apresentada a cada tentativa de conexão durante este período. Quando o tempo determinado tiver terminado sem que o usuário tenha trocado a password, o usuário é bloqueado e, somente, será liberado pelo DBA.

◊*PASSWORD HISTORY* – esta opção verifica as password especificadas ultimamente para garantir que o usuário não repita uma password durante um determinado tempo ou quantidade de vezes. O objeto PROFILE deve ser usado para estabelecer as regras deste histórico.

◊*PASSWORD COMPLEXITY VERIFICATION* – esta opção garante que a password seja complicada o suficiente para que "intrusos" tenham dificuldade em descobri-la. A rotina de verificação de complexidade da Oracle exige que a password:

- Tenha um mínimo de 4 caracteres de comprimento.
- Não seja igual ao USERID.
- Inclua pelo menos um caracter alfanumérico, um caracter numérico e um caracter de pontuação.
- Não seja igual a uma lista interna de palavras simples como welcome, account, database, user, etc.
- Seja diferente da password anterior em pelo menos 3 caracteres.

AUTENTICAÇÃO DO DBA

Uma vez que o DBA executa operações especiais que não devem ser realizadas por usuários comuns, o Oracle cria um esquema de autenticação mais seguro para DBAs.

Podemos escolher entre autenticação pelo sistema operacional ou através de um arquivo de passwords.

Em muitos sistemas operacionais a autenticação pelo sistema operacional requer que o usuário do DBA (que é o mesmo no SO e no ORACLE) seja colocado em um grupo especial (grupo **dba** no UNIX).

O banco de dados usa arquivos de passwords para garantir o acesso de usuários que tenham recebido um dos privilégios: SYSDBA ou SYSOPER. Com estes privilégios o DBA poderá realizar as seguintes ações:

◊**SYSOPER** – permite que executemos STARTUP, SHUTDOWN, ALTER DATABASE OPEN / MOUNT, ALTER DATABASE BACKUP, ARCHIVE LOG, RECOVER e inclui o privilégio RESTRICTED SESSION.

◊**SYSDBA** – contém todos os privilégios de sistema com a opção WITH ADMIN OPTION, o privilégio de sistema SYSOPER, permite a execução do comando CREATE DATABASE e recuperação baseada em tempo.

AUTENTICAÇÃO DE UM PROGRAMA

Em ambientes multi-camadas, o Oracle controla a segurança das aplicações que estiverem no servidor de aplicações (middle-tier), limitando seus privilégios, preservando a identidade do cliente am todas as camadas e auditando ações feitas em nome dos clientes.

Neste tipo de arquitetura um servidor de aplicações fornece dados para clientes e servidores como um elemento intermediário entre os clientes e os diversos servidores de banco de dados.

Nesta arquitetura podemos usar um servidor de aplicações para validar as credenciais de um cliente (por exemplo um web browser). O servidor de banco de dados pode, também, auditar operações realizadas pelo próprio servidor de aplicação e as que ele fizer em nome dos clientes.

A autenciação em ambientes multi-camadas é baseado em “trust regions”, da seguinte forma:

◊O cliente fornece uma prova de autenticação para o servidor de aplicação: usando uma password ou um certificado X.509.

◊O servidor de aplicação verifica a autenticação do cliente e, então, autentica a si mesmo para o servidor de banco de dados.

◊O servidor de banco de dados verifica a autenticação do servidor de aplicação, verifica se o cliente existe e, verifica se o servidor de aplicação tem o privilégio de estabelecer conexão para este cliente.

Se forem usadas ROLES para autenticação (observar o comando ALTER USER a seguir) o servidor de banco de dados também verifica:

◊Se o cliente tem estas roles.

◊Se o servidor de aplicação tem o privilégio de estabelecer conexão em nome deste usuário usando estas roles.

Outras características deste ambiente não serão detalhadas neste curso. O curso que aborda e estuda a montagem de um ambiente de rede é Oracle Administração de Redes.

PASSOS PARA CRIAÇÃO DE UM USUÁRIO

A criação de um novo usuário requer a especificação de algumas características (já comentadas superficialmente anteriormente), como veremos a seguir:

- ◊ **Tablespace Default** – especifica onde os objetos serão criados, se nenhum Tablespace for definido durante os comandos CREATE TABLE, CREATE INDEX ou CREATE CLUSTER.

- ◊ **Tablespace Temporário** – especifica uma área para armazenamento de dados temporários. Este tablespace é, particularmente, utilizado em operações que necessitem de ordenação.

- ◊ **Cotas em Tablespace** – determina o máximo de espaço que o usuário pode consumir para cada tablespace (uma cota 0 torna o tablespace inacessível). Por default cada usuário não tem direito de armazenamento (QUOTA) em nenhum Tablespace. Desta forma se o usuário precisar criar qualquer objeto deve ser atribuído a este usuário uma quantidade de área em um dos Tablespaces da instalação ou privilégio para criação do objeto no SCHEMA de um outro usuário que tenha espaço disponível.

- ◊ **Limites de recursos do sistema** - Inclui o tempo de CPU, o número de leituras lógicas, o número de sessões concorrentes por usuários, e o tempo inativo por sessão, especificado através de PROFILES.

- ◊ **Características de controle para PASSWORD** – este assunto, apesar de bastante comentado, requer uma especificação. Podemos determinar um padrão de controle para a instalação que se aplique a todos os usuários que viermos a criar. Definiremos, então a freqüência de troca da password, seu layout, validação de texto, etc.

O GRUPAMENTO PUBLIC

Cada banco de dados contém um grupamento chamado PUBLIC. Poderíamos dizer que se trata de um usuário com características especiais, porém a Oracle considera um grupamento e não (exatamente) um usuário.

O grupamento PUBLIC fornece acesso público a objetos do banco de dados subordinados a um SCHEMA e estabelece privilégios comuns a todos os usuários. Todos os usuários do banco de dados pertencem ao grupamento PUBLIC.

Podemos atribuir privilégios de sistema ou sobre objetos específicos para o grupamento PUBLIC. Quando fazemos isto, todos os usuários, adquirem os privilégios autorizados.

OBS: A autorização ou revogação de alguns privilégios de sistema ou sobre objetos para um grupamento PUBLIC pode fazer com que views, procedures, functions, packages e triggers no banco de dados seja recompilados.

Como restrição, temos:

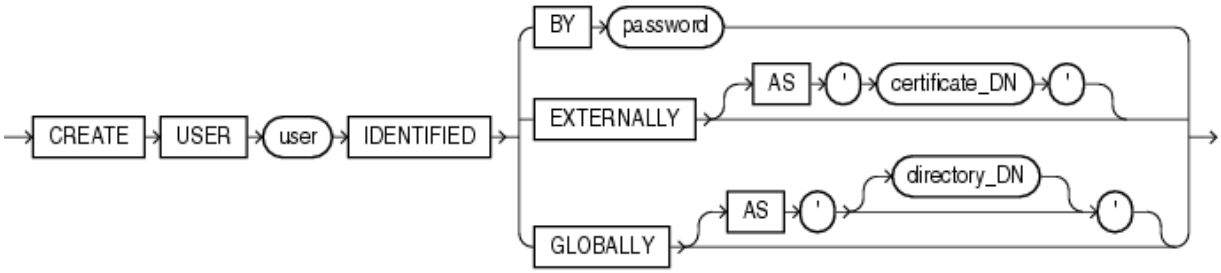
◊Não podemos associar cotas de TABLESPACES para PUBLIC, porém podemos atribuir o privilégio UNLIMITED TABLESPACE para PUBLIC.

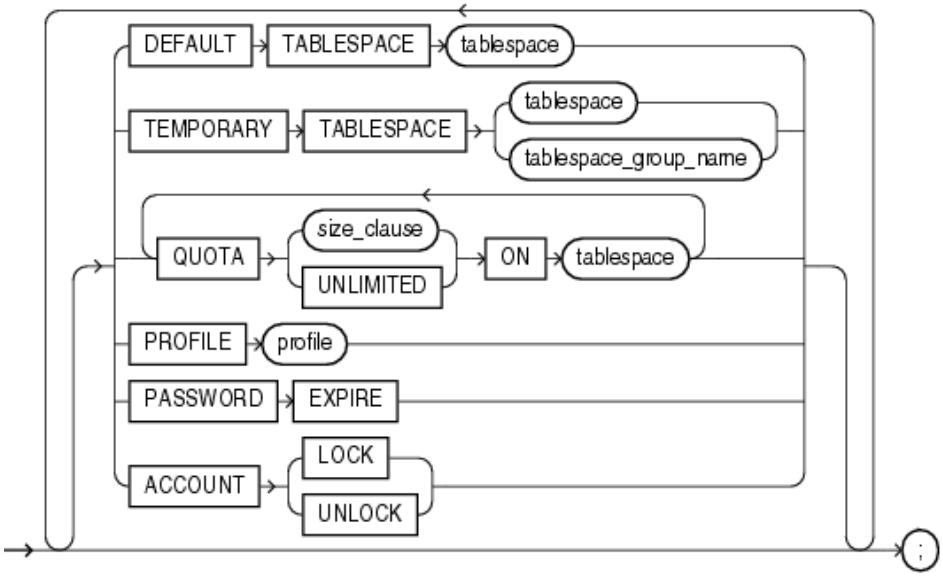
◊Podemos criar database links e sinônimos como PUBLIC, porém nenhum outro objeto pode ser subordinado diretamente a PUBLIC. Por exemplo: create table public.xxx é inválido.

◊Os segmentos de undo podem ser criados com a palavra chave PUBLIC mas, na verdade, não ficarão subordinados ao grupamento PUBLIC (todos os segmentos de undo são subordinados ao usuário SYS).

SINTAXE CREATE USER

Podemos criar um usuário com a caixa de diálogo CREATE USER do Database Manager ou com o comando de DDL CREATE USER, o qual veremos a seguir.





OBS: Quando criamos um usuário usando o Server Manager, a role CONNECT é associada a ele, automaticamente.

Onde:

- ◊ *user* - identifica o nome do usuário que vai ser criado.
- ◊ BY *password* – especifica a *password* para o logon.
- ◊ EXTERNALLY – indica que a autenticação será feita através do Sistema Operacional.
- ◊ GLOBALLY AS '*external_name'* – cria um usuário global e indica que o usuário deve ser autenticado pelo Enterprise Directory Service. '*external_name'* é o nome do X.509 no EDS (Enterprise Directory Service) que identifica o usuário. Deve ter o formato: 'CN=*user*, *demais atributos'*, onde "*demais atributos*" corresponde ao restante do Distinguisehd Name (DN) do usuário no diretório.
- ◊ DEFAULT TABLESPACE – identifica a tablespace *default* para objetos do usuário.
- ◊ TEMPORARY TABLESPACE – identifica um tablespace temporário que servirá como área de trabalho temporária para o usuário.
- ◊ QUOTA – permite a especificação de um limite de área para o usuário em determinado tablespace ou a especificação de uso indiscriminado de espação (UNLIMITED) em um tablespace.
- ◊ PROFILE *profile* – especifica o profile nomeado para o usuário, o qual estabelecerá regras para utilização de determinados recursos do sistema.
- ◊ DEFAULT ROLE – especifica a role (ou conjunto de roles) a serem adquiridas pelo usuário a tempo de conexão. Caso esta opção não seja utilizada, por default, todas as roles que o usuário tem direito de usar serão, automaticamente, atribuídas a ele a tempo de conexão.
- ◊ PASSWORD EXPIRE – indica que quando o usuário tentar estabelecer a primeira conexão, sua password já deverá ser trocada.
- ◊ ACCOUNT LOCK / UNLOCK – indica que para este usuário deve ser estabelecido o bloqueio (ou não) do usuário.

EXEMPLO

Criar o usuário GABRIEL com a password CURSO. A tablespace default é USERS, com cota 15M. A tablespace temporária é TEMP. Liberar uma quota de 10M no tablespace SYSTEM. Indicar que o usuário deverá trocar a password antes de efetuar uma conexão com este USER pela primeira vez.

```
SQL> CREATE USER GABRIEL
  2  IDENTIFIED BY curso
  3  DEFAULT TABLESPACE users
  4  QUOTA 15M ON users
  5  TEMPORARY TABLESPACE TEMP
  6  QUOTA 10M ON system
  7  PASSWORD EXPIRE;

Usuário criado.

SQL> GRANT create session TO GABRIEL;

Operação de Grant bem-sucedida.

SQL> connect GABRIEL/curso@ORCL
ERRO:
ORA-28001: a senha expirou


Alterando senha para GABRIEL
Senha antiga: *****
Ocorreu um erro no Sistema Operacional
Senha não-alterada
Aviso: Você não está mais conectado ao ORACLE.
SQL> connect aluno/aluno@ORCL
Conectado.
```

OBS:

- ◊ Dois usuários são *reservados* para o banco de dados ORACLE: ***sys*** e ***system***. Para estes usuários não devemos fazer qualquer tipo de alteração nas características do usuário (exceto PASSWORD, que podemos trocar).
- ◊ Por default, o usuário não tem acesso a nenhum tablespace do banco de dados .
- ◊ Não é suficiente a criação de um usuário, devemos atribuir a ele pelo menos o privilégio capaz de estabelecer conexão.

ALTERANDO UM USUÁRIO

Modificar as características de segurança para um usuário do banco de dados existente para mudar as opções associadas com aquele usuário.

As seguintes modificações podem ser feitas quando alteramos um usuário:

- ◊ Password
- ◊ Autenticação do Sistema Operacional
- ◊ Tablespace default
- ◊ Tablespace temporária
- ◊ Cotas por tablespace
- ◊ Profile
- ◊ Roles defaults

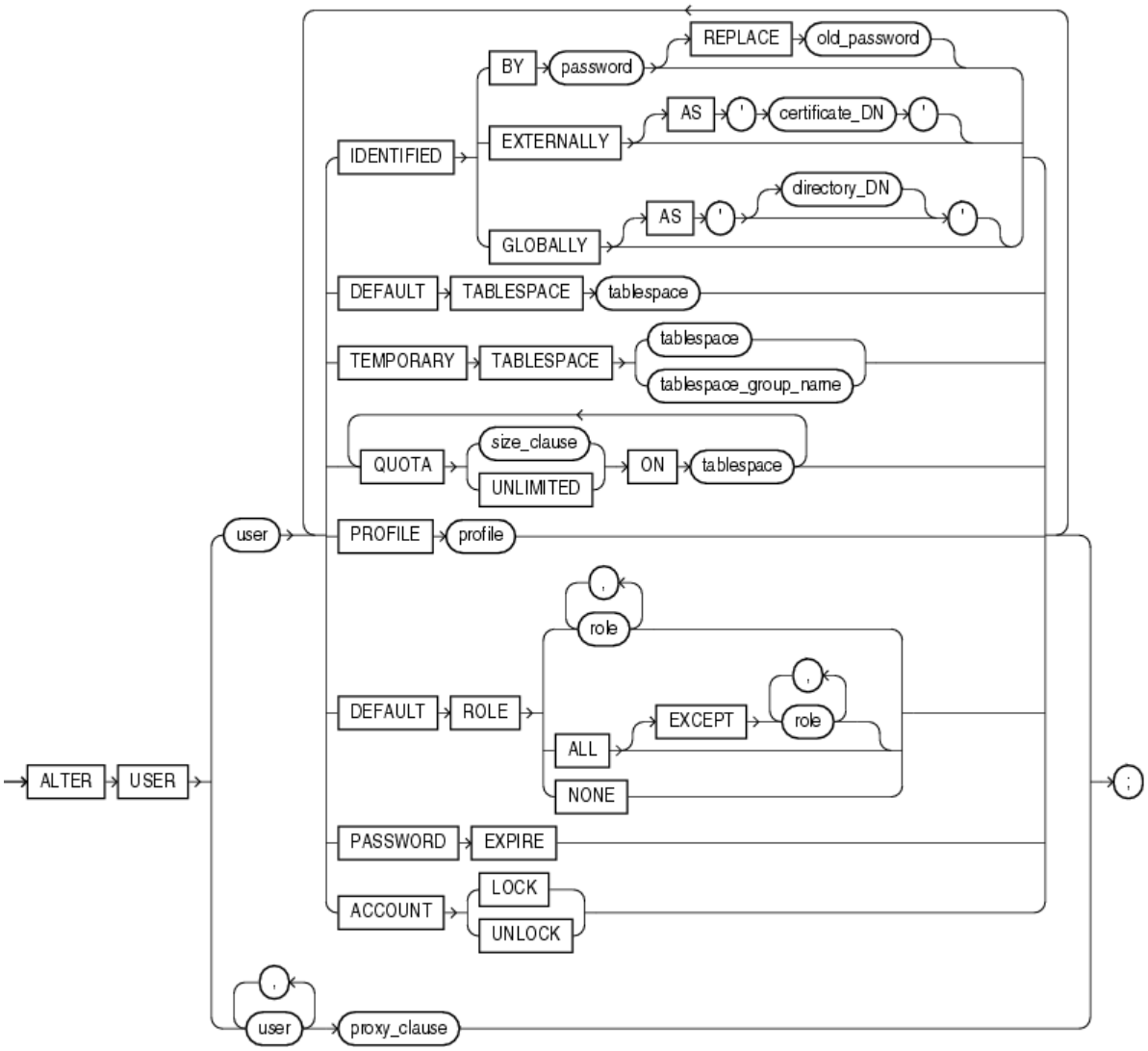
OBS:

- ◊ Modificações nas características de segurança para um usuário não afetam as sessões correntes, apenas as sessões subseqüentes.
- ◊ Usuários podem mudar suas próprias *passwords*.
- ◊ A modificação da quota de área para um tablespace tanto pode ser uma redução quanto uma ampliação. Ao modificarmos o valor da quota para 0, estaremos revogando totalmente o acesso àquele tablespace.
- ◊ Os *profiles* serão alterados se o DBA decidir modificar os limites sobre os recursos do sistema.
- ◊ As roles devem ser re-associadas se um usuário mudar de departamento (ou de sistema, ou de cargo, ou...) e necessitar de diferentes privilégios para as novas tabelas do novo departamento (dependendo do critério usado para a criação e associação das *roles*).

SINTAXE ALTER USER

A alteração das características de um usuário pode ser feita com a caixa de diálogo ALTER USER ou através do comando ALTER USER do banco de dados.

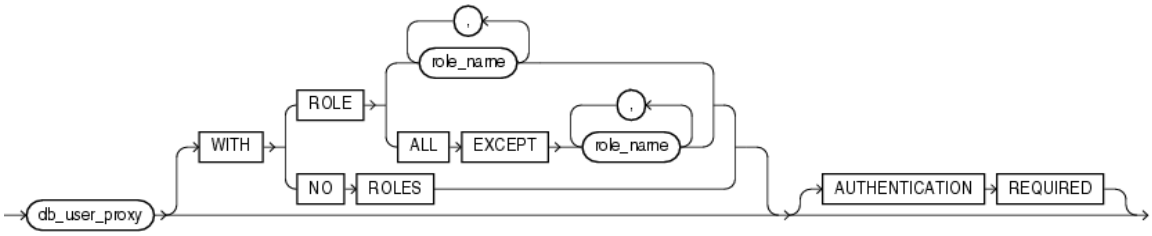
alter_user::=



(size_clause::=)

proxy_clause ::=

proxy



Onde:

- ◊ *user* - identifica o nome do usuário que vai ser alterado.
- ◊ BY *password* – especifica a *password* para o logon.
- ◊ EXTERNALLY – indica que a autenticação será feita através do Sistema Operacional.
- ◊ GLOBALLY AS '*external_name'* – indica que o usuário deve ser autenticado pelo Enterprise Directory Service. '*external_name'* é o nome do X.509 no EDS (Enterprise Directory Service) que identifica o usuário. Deve ter o formato: 'CN=*user*, *demais atributos'*, onde "*demais atributos*" corresponde ao restante do Distinguisehd Name (DN) do usuário no diretório.
- ◊ DEFAULT TABLESPACE – identifica a tablespace *default* para objetos do usuário.
- ◊ TEMPORARY TABLESPACE – identifica um tablespace temporário que servirá como área de trabalho temporária para o usuário.
- ◊ QUOTA – permite a especificação de um limite de área para o usuário em determinado tablespace ou a especificação de uso indiscriminado de espação (UNLIMITED) em um tablespace.
- ◊ PROFILE *profile* – especifica o profile nomeado para o usuário, o qual estabelecerá regras para utilização de determinados recursos do sistema.
- ◊ DEFAULT ROLE – especifica a role (ou conjunto de roles) a serem adquiridas pelo usuário a tempo de conexão. Caso esta opção não seja utilizada, por default, todas as roles que o usuário tem direito de usar serão, automaticamente, atribuídas a ele a tempo de conexão.
- ◊ PASSWORD EXPIRE – indica que quando o usuário tentar estabelecer a primeira conexão, sua password já deverá ser trocada.
- ◊ ACCOUNT LOCK / UNLOCK – indica que para este usuário deve ser estabelecido o bloqueio (ou não) do usuário.
- ◊ PROXY CLAUSE – esta cláusula controla a possibilidade de um *proxy* (que pode ser apenas uma aplicação ou um servidor de aplicações) de estabelecer conexão como um determinado usuário e ativar uma (ou conjunto de ou nenhuma) role específica.
- ◊ *proxy* – identifica o programa a receber esta permissão.

EXEMPLO

O privilégio ALTER USER é requerido para usar tanto a caixa de diálogo Server Manager Alter User quanto para o comando ALTER USER de SQL.

Alterar o usuário GABRIEL. Mudar a password para *teste*. Mudar a tablespace default para DES60_DATA, com uma quota de 10M. Associar a este usuário o profile default.

```
SQL> ALTER USER GABRIEL
  2  IDENTIFIED BY teste
  3  DEFAULT TABLESPACE des60_data
  4  QUOTA 10M ON des60_data
  5  PROFILE DEFAULT;

Usuário alterado.
```

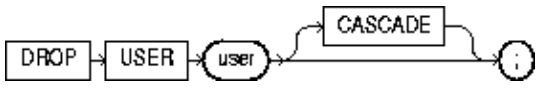
OBS:

- ◊ Apenas as opções especificadas no comando ALTER USER são alteradas, todos os recursos previamente liberados são mantidos.

- ◊ Quando uma cota de 0 é determinada, os objetos pertencentes ao usuário permanecem no tablespace revogado, porém seu crescimento está bloqueado.

SINTAXE DROP USER

Remover um usuário do banco de dados com a caixa de diálogo DROP USER ou com o comando de SQL DROP USER.



Onde:

- ◊ *user* - identifica o nome do usuário que vai ser removido.
- ◊ CASCADE – indica que todos os objetos do usuário devem ser removidos também. Esta opção é obrigatória quando no SCHEMA do usuário existirem objetos.

OBS:

- ◊ A opção CASCADE só deve ser usada em caso extremo. Quando um usuário é removido com a opção CASCADE, o *username* e o Schema associado são removidos do dicionário de dados e todos os objetos contidos no Schema do usuário são imediatamente removidos.
- ◊ Um usuário que está conectado com o banco de dados não pode ser removido.
- ◊ O privilégio DROP USER é necessário para usarmos tanto a caixa de diálogo DROP USER quanto para o comando de SQL DROP USER.

EXEMPLO

Remover o usuário GABRIEL e selecionar a opção que remove todos os objetos deste usuário.

```
SQL> DROP USER GABRIEL CASCADE;

Usuário eliminado.
```

MONITORANDO USUÁRIOS

Podemos consultar o DICIONÁRIO DE DADOS para obter informações sobre cada usuário do banco de dados.

CONSULTANDO INFORMAÇÕES DO USUÁRIO

O dicionário de dados contém as seguintes informações :

- ◊ Todos os usuários do banco de dados;
- ◊ Tablespace default para as tabelas, clusters e índices de cada usuário;
- ◊ Tablespace usada para segmentos temporários;
- ◊ Cotas;

Views úteis do DICIONÁRIO DE DADOS. Durante o curso trabalharemos sempre com as views de prefixo DBA.

- ◊ USER_USERS
- ◊ ALL_USERS
- ◊ DBA_USERS
- ◊ USER_TS_QUOTAS
- ◊ DBA_TS_QUOTAS

Podemos monitorar o banco de dados através das views do dicionário de dados para visualizarmos que informações estão registradas.

As views do dicionário de dados nos fornecem informações atualizadas sobre os usuários e objetos. Um DBA deve estar familiar com as views apresentadas acima para monitorar usuários e cotas.

EXEMPLO

Verificar o layout da view do dicionário de dados DBA_USERS.

```
SQL> DESC DBA_USERS
 Name                            Null?    Type
 ------------------------------- -------- ----
 USERNAME                        NOT NULL VARCHAR2(30)
 USER_ID                         NOT NULL NUMBER
 PASSWORD                                 VARCHAR2(30)
 ACCOUNT_STATUS                  NOT NULL VARCHAR2(32)
 LOCK_DATE                                DATE
 EXPIRY_DATE                              DATE
 DEFAULT_TABLESPACE              NOT NULL VARCHAR2(30)
 TEMPORARY_TABLESPACE            NOT NULL VARCHAR2(30)
 CREATED                         NOT NULL DATE
 PROFILE                         NOT NULL VARCHAR2(30)
 INITIAL_RSRC_CONSUMER_GROUP              VARCHAR2(30)
 EXTERNAL_NAME                            VARCHAR2(4000)
```

Onde:

- ◊ USERNAME - identifica o nome do usuário.
- ◊ USER_ID – identifica o usuário internamente para o ORACLE. É um código numérico atribuído a cada usuário.
- ◊ PASSWORD – corresponde à password criptografada.
- ◊ ACCOUNT_STATUS – indica se o USER está LOCKED, EXPIRED ou UNLOCKED.
- ◊ LOCK_DATE – apresenta a data que o USER foi LOCKED, se a coluna ACCOUNT_STATUS estiver com o valor LOCKED.
- ◊ EXPIRY_DATE – data de expiração da password do usuário.
- ◊ DEFAULT TABLESPACE – identifica a tablespace *default* para objetos do usuário.
- ◊ TEMPORARY TABLESPACE – identifica um tablespace temporário que servirá como área de trabalho temporária para o usuário.
- ◊ CREATED – data que o usuário foi criado.
- ◊ PROFILE– indica o profile nomeado para o usuário.
- ◊ INITIAL_RSRC_CONSUMER_GROUP – indica o RESOURCE CONSUMER GROUP inicial para o usuário.
- ◊ EXTERNAL_NAME – indica o nome do X.509 no EDS (Enterprise Directory Service) que identifica o usuário.

Mostrar informações sobre dois usuários(SCOTT, ALUNO) do banco de dados com a view do dicionário de dados DBA_USERS.

```
SQL> SET LINESIZE 100
SQL> SELECT * FROM DBA_USERS
  2  WHERE USERNAME IN ('HR', 'ALUNO');

USERNAME              USER_ID   PASSWORD
--------------------- --------- ------------------------------
ACCOUNT_STATUS        LOCK_DAT EXPIRY_D DEFAULT_TABLESPACE
--------------------- -------- -------- -------------------------
TEMPORARY_TABLESPACE  CREATED  PROFILE
--------------------- -------- ------------------------------
INITIAL_RSRC_CONSUMER_GROUP    EXTERNAL_NAME
------------------------------ ------------------------
ALUNO                 37       539DF812E6FD77F6
OPEN                                  USERS
TEMP                  20/05/06 DEFAULT
DEFAULT_CONSUMER_GROUP

HR                    26       F894844C34402B67
OPEN                                  USERS
TEMP                  01/03/05 DEFAULT
DEFAULT_CONSUMER_GROUP
```

Obter a descrição da view DBA_TS_QUOTAS

```
SQL> DESC DBA_TS_QUOTAS
 Name                           Null?    Type
 ------------------------------ -------- ----
 TABLESPACE_NAME                NOT NULL VARCHAR2(30)
 USERNAME                       NOT NULL VARCHAR2(30)
 BYTES                                   NUMBER
 MAX_BYTES                               NUMBER
 BLOCKS                         NOT NULL NUMBER
 MAX_BLOCKS                              NUMBER
```

Mostrar as cotas por tablespace para o usuário GABRIEL com a view do dicionário de dados DBA_TS_QUOTAS.

```
SQL> set linesize 200
SQL> SELECT * FROM DBA_TS_QUOTAS
  2   WHERE USERNAME = 'GABRIEL';

TABLESPACE_NAME USERNAME    BYTES MAX_BYTES    BLOCKS MAX_BLOCKS
--------------- ----------- ----- --------- --------- ----------
SYSTEM          GABRIEL         0  10485760         0       5120
USERS           GABRIEL         0  15728640         0       7680
DES60 DATA      GABRIEL         0  10485760         0       5120
```

Onde:

- ◊ TABLESPACE_NAME - Nome do tablespace.
- ◊ BYTES - Número de bytes carregados para o usuário.
- ◊ MAX_BYTES - Cota do usuário em bytes, ou UNLIMITED (-1).
- ◊ BLOCKS - Número de blocos ORACLE carregados para o usuário.
- ◊ MAX_BLOCKS - Cota do usuário em blocos ORACLE, ou UNLIMITED

OBS: O valor **-1** indica que a cota é **ilimitada**.

CONCEITUANDO SESSÃO

Quando um usuário executa uma aplicação ou uma ferramenta da Oracle, sua primeira ação é fornecer um nome de usuário para que seja estabelecida uma conexão com o banco de dados.

Os termos “conexão” e “sessão” estão intimamente relacionados ao termo “processo do usuário”, porém possuem significados bastante diferenciados.

Uma ***conexão*** é uma comunicação entre um processo do usuário e uma instância ORACLE. Corresponde a um caminho para troca de mensagens e garantido por mecanismos de comunicação ou softwares de rede.

Já uma ***sessão*** é o período de tempo contínuo em que um processo do usuário estabelece conexão com uma instância Oracle utilizando para tal um *username* e *password* informados pelo usuário.

Podemos estabelecer múltiplas sessões utilizando o mesmo *username* e *password* e, às vezes, até mesmo com o mesmo processo usuário.

FINALIZANDO UMA SESSÃO DO USUÁRIO

Quando necessário, podemos terminar (forçar o término) uma sessão enquanto o usuário estiver conectado ao banco de dados .

Isto pode ser útil para:

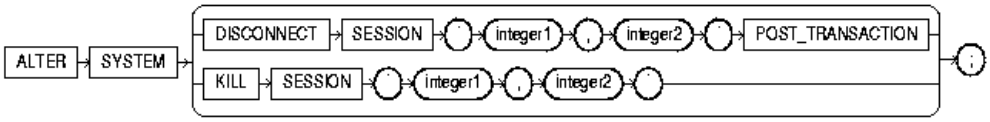
- ◊ impedir que o usuário emita futuras chamadas ao banco de dados (o DBA precisa fechar o banco de dados);
- ◊ liberar recursos bloqueados (um usuário pode estar prendendo recursos que precisam ser usados por outros usuários);

Esta ação:

- ◊ exibe uma mensagem para o usuário;
- ◊ requer o privilégio ALTER SYSTEM.

A opção IMMEDIATE durante o comando SHUTDOWN pode ser uma alternativa para finalizarmos a sessão de todos os usuários.

SINTAXE ALTER SYSTEM (PARCIAL)



onde:

◊DISCONNECT SESSION – desconecta a sessão especificada destruindo o Processo Servidor (ou o circuito virtual se a conexão tiver sido feita por meio de um processo Multi-Threaded). Esta cláusula permite a conclusão da transação antes da sessão ser desconectada.

Para identificarmos a sessão do usuário, devemos consultar a view V$SESSION, onde "integer1" corresponde ao valor da coluna SID e "integer2" corresponde ao valor da coluna SERIAL#.

◊KILL SESSION – termina a sessão, desmanchando a transação em uso e liberando todos os locks e recursos associados àquela sessão.

Para identificarmos a sessão do usuário, devemos consultar a view V$SESSION, onde "integer1" corresponde ao valor da coluna SID e "integer2" corresponde ao valor da coluna SERIAL#.

Se a sessão estiver executando alguma atividade que deva ser completada (por exemplo aguardando resposta de um banco de dados remoto ou desmanchando uma transação), o Oracle aguarda que a atividade seja completada, "matando" a sessão em seguida e retornando o controle para o DBA. Se o tempo de espera for superior a um minuto, o Oracle marca a sessão para ser "killed" e retorna o controle para o DBA com uma mensagem indicando que a sessão está marcada desta forma. O Oracle, então, "mata" a sessão quando a atividade estiver completada.

EXEMPLO

Abrir 2 cópias do SQL*PLUS com os usuários HR e GABRIEL. Executar um comando de atualização em cada um desses usuários (sem COMMIT). Consultar a view V$SESSION. Usar a opção DISCONNECT para o usuário HR e a opção KILL para o usuário GABRIEL.

HR

```
SQL> show user
USER é "HR"
SQL> INSERT INTO emp (empno, ename) VALUES (1, 'HR');
1 linha criada.
```

GABRIEL

```
SQL> show user
USER é "GABRIELI"
SQL>  INSERT  INTO  scott.emp  (empno,  ename)  VALUES  (2,
'GABRIEL');
1 linha criada.
```

SYS

```
SQL> SELECT SID, SERIAL#, USERNAME FROM V$SESSION;
      SID   SERIAL# USERNAME
--------- --------- ------------------------------
        1         1
...
        8      4579
       11       263 SYS
       12      2386 HR
       13      2033 GABRIEL

SQL> ALTER SYSTEM DISCONNECT SESSION '12, 2386'
  2  POST_TRANSACTION;
Sistema alterado.

SQL> ALTER SYSTEM KILL SESSION '13, 2033';
Sistema alterado.
```

HR

```
SQL> COMMIT;
Validação completa.

SQL> SELECT * FROM EMP WHERE EMPNO = 1;
SELECT * FROM EMP WHERE EMPNO = 1
*
ERRO na linha 1:
ORA-00028: your session has been killed
```

GABRIEL

```
SQL> COMMIT;
COMMIT
*
ERRO na linha 1:
ORA-00028: a sessão foi cancelada
```

PROFILES

Na criação de usuários observamos que podemos estabelecer limites em relação às passwords e limites em relação a uso recursos do sistema.

A tempo de criação do usuário não informamos todas estas políticas diretamente, o fazemos especificando um PROFILE e associando-o ao usuário.

Um PROFILE corresponde a um conjunto de limites sobre diversos recursos do sistema e que podemos associar a um usuário em um banco de dados Oracle.

Como vantagens na utilização de limites, temos:

◊permite o grupamento de restrições;

◊podem ser associados aos usuários;

◊podem ser habilitados ou desabilitados para todo o sistema;

◊simplifica o gerenciamento dos recursos;

◊pode ser usado em grandes sistemas multi-usuários ou quando existem normas rígidas de segurança;

◊permite a adoção de uma política de segurança em relação ao uso de PASSWORDs.

◊restringe os usuários na execução algumas operações que requeiram uso de muitos recursos do sistema.

◊garante a retirada dos usuários do banco de dados quando tenham excedido o limite estabelecido.

◊grupa limites para usuários similares.

Na versão 9i a utilidade do profile cresceu na medida em que podemos estabelecer, com ele, um controle sobre o uso de PASSWORD.

CARACTERÍSTICAS

Os profiles podem ser criados, alterados ou removidos.

Os limites podem ser especificados individualmente ou podemos estabelecer um número composto que dará maior flexibilidade aos limites individuais.

Podemos dividir o controle em dois grupos: limites associados à PASSWORD e limites associados aos recursos.

Os limites associados aos recursos podem ser assegurados a nível de sessão (a cada conexão), chamada (CALL) ou ambos.

CONSEQÜÊNCIA DO USO DE LIMITES

Os limites estabelecidos a nível de sessão se aplicam a cada conexão. Quando um limite estabelecido a nível de sessão é excedido :

◊A instrução atual é desmanchada.

◊Todas as instruções anteriores permanecerão intactas.

◊Somente uma operação de COMMIT ou ROLLBACK (ou a desconexão) é permitida.

◊Nenhuma outra tarefa pode ser realizada por aquela sessão.

Os limites estabelecidos a nível de chamada (CALL) são assegurados para cada chamada feita enquanto estivermos executando um comando de SQL. Quando um limite estabelecido a nível de chamada (CALL) é excedido :

◊O processamento da instrução é interrompido.

◊A instrução é desmanchada.

◊Todas as instruções anteriores permanecerão intactas.

◊A sessão do usuário permanece conectada.

Os limites associados à PASSWORD estão ligados à conexão. Quando um limite associado à PASSWORD é excedido:

◊A sessão atual do usuário não é afetada.

◊Na próxima conexão o usuário perceberá que o limite foi excedido.

LIMITANDO OS RECURSOS

Para limitarmos os recursosdevemos, inicialmente, indicar ao Oracle que desejamos efetuar este controle.

RESOURCE_LIMIT

O parâmetro RESOURCE_LIMIT indica ao Oracle se desejamos ou não controlar o uso de limites. Esta indicação pode ser feita no arquivo de inicialização *init.ora* ou através do comando ALTER SYSTEM.

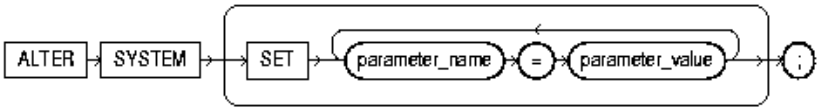
OBS: Este comando não afeta os recursos associados a PASSWORDS. Estes estão sempre habilitados.

SINTAXE ALTER SYSTEM (PARCIAL)

Com o uso deste comando estaremos alterando a situação do uso de profiles enquanto o banco de dados estiver em atividade.

Este comando tem validade até que o banco saia de atividade ou até que outro comando seja fornecido.

Podemos usar este comando quando não pudermos retirar o banco de atividade, porém se desejarmos que esta opção se mantenha, devemos, também, modificar o arquivo de parâmetros de inicialização.



Onde :

◊SET *parameter_name* – refere-se a um dos parâmetros de inicialização que poderemos modificar. No caso atual desejamos modificar o parâmetro de sistema RESOURCE_LIMIT.

◊*parameter_value* – refere-se ao valor a ser associado ao parâmetro dinamicamente. No caso podemos especificar TRUE ou FALSE.

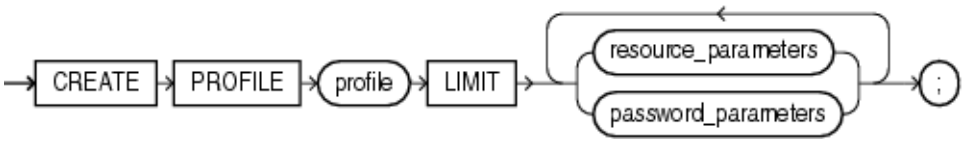
OBS:

◊Quando modificamos o valor do parâmetro RESOURCE_LIMIT dinamicamente, os recursos associados aos usuários, somente tem efeito a partir do momento que o usuário estabelecer conexão. As sessões em uso não são afetadas.

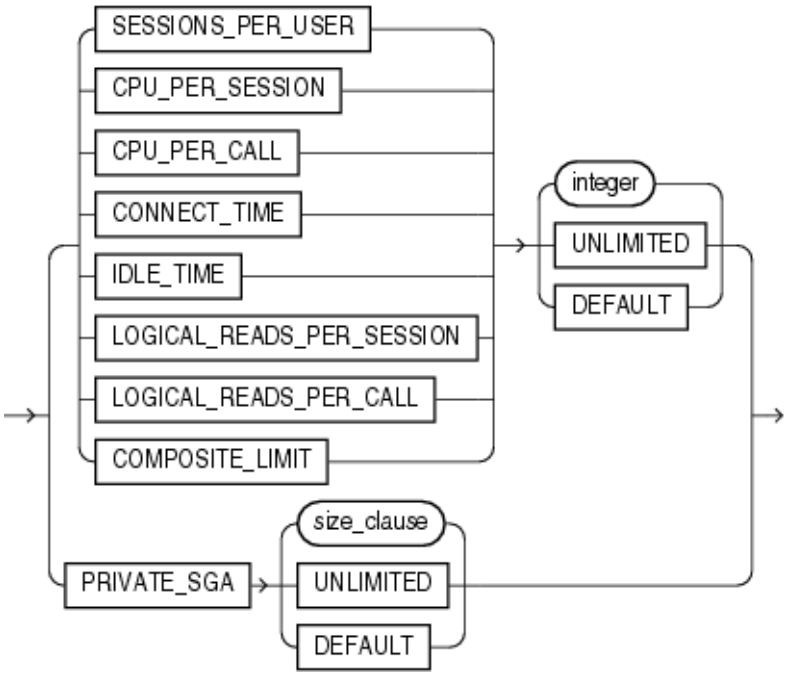
◊Este parâmetro só tem efeito se houvermos atribuído anteriormente um profile com limites para os usuários.

SINTAXE CREATE PROFILE

create_profile::=

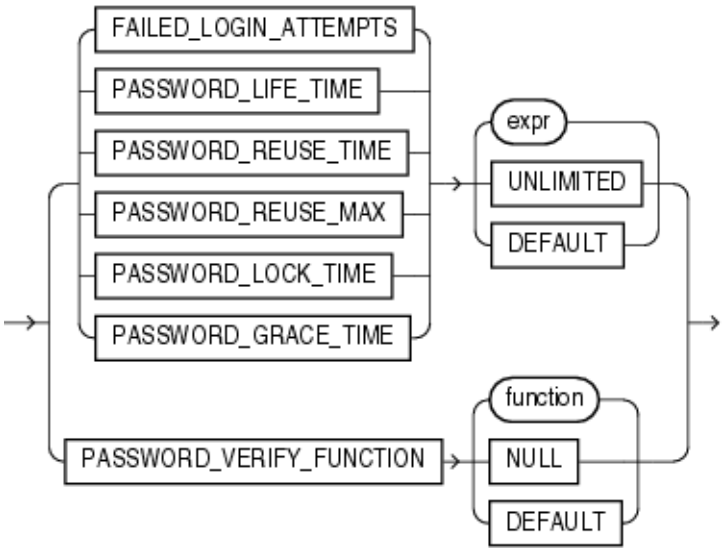


resource_parameters::=



(size_clause::=

password_parameters ::=



Onde:

◊UNLIMITED - indica que o usuário associado a este profile pode usar uma quantidade de recursos ilimitada ou que não existe limite definido para este parâmetro de PASSWORD.

◊DEFAULT - indica que este profile está sujeito aos limites de recurso estabelecidos no profile DEFAULT.

resource_parameters - session

◊SESSIONS_PER_USER - número de sessões concorrentes por usuário.

◊CPU_PER_SESSION - tempo total de CPU em centésimos de segundo.

◊CONNECT_TIME - tempo de conexão em minutos.

◊IDLE_TIME - tempo de inatividade em minutos. É calculado somente para processos servidores. Não afeta o tempo de inatividade de uma aplicação. Este tempo também não é afetado pela execução de longas queries e outras operações.

◊LOGICAL_READS_PER_SESSION - número de blocos de dados lidos por sessão. É uma limitação do número total de leituras tanto de disco quanto de memória.

◊PRIVATE_SGA – especifica a quantidade de espaço na SGA (Session Information) que uma sessão pode alocar (aplicável apenas para arquitetura Multi-Thread e pode ser especificado em M ou K).

◊COMPOSITE_LIMIT - limita o total de recursos para uma sessão expresso em unidades de serviço. Corresponde à soma dos valores de CPU_PER_SESSION, CONNECT_TIME, LOGICAL_READS_PER_SESSION e PRIVATE_SGA multiplicados por seu peso específico na composição. Este peso é dado pelo comando RESOURCE_COST.

---

resource_parameters - call

◊CPU_PER_CALL - tempo de CPU por chamada em centésimos de segundo.

◊LOGICAL_READS_PER_CALL - número de blocos de dados lidos por chamada. É uma limitação do número total de leituras tanto de disco quanto de memória

password_parameters

◊FAILED_LOGIN_ATTEMPTS – especifica o número limite de tentativas falhas que o usuário pode fazer antes do usuário ficar bloqueado.

◊PASSWORD_LIFE_TIME – determina o tempo de vida de uma password(fornecido em dias). Após este períodoa password expirará obrigando o usuário a fazer uma modificação.

◊PASSWORD_REUSE_TIME – determina o número de dias antes do qual uma password não pode ser reutilizada. Quando preenchermos este parâmetro com um número, o parâmetro PASSWORD_REUSE_MAX deve ser preenchido com UNLIMITED.

◊PASSWORD_REUSE_MAX – determina o número de trocas de password necessários para que a password corrente possa reutilizada. Se preenchermos o parâmetro PASSWORD_REUSE_MAX com um número, o parâmetro PASSWORD_REUSE_TIME deve ser preenchido com UNLIMITED.

◊PASSWORD_LOCK_TIME – especifica o número de dias que o usuário ficará bloqueado após o número especificado de tentativas consecutivas falhas de login ter acontecido.

◊PASSWORD_GRACE_TIME – especifica o número de dias, após o tempo da password ter expirado, durante o qual o usuário receberá um aviso a cada tentativa de conexão. Se a password não for modificada durante este período, a password expira.

◊PASSWORD_VERIFY_FUNCTION – permite que indiquemos uma rotina que faça a validação da password informada. O valor DEFAULT para este parâmetro utiliza a rotina padrão do oracle.

restrições

◊Se ambos os parâmetros PASSWORD_REUSE_TIME e PASSWORD_REUSE_MAX receberem o valor UNLIMITED, o Oracle não utilizará estes critérios.

◊A função de validação deve ser criada subordinada ao usuário SYS e deve ter o formato:

```
routine_name (userid_parameter IN VARCHAR(30),
              password_parameter IN VARCHAR (30),
              old_password_parameter IN VARCHAR (30)
) RETURN BOOLEAN
```

A rotina DEFAULT fornecida pela ORACLE pode ser vista no manual Oracle Administrator's Guide Release (capítulo 22, tópico Password Complexity Verification).

EXEMPLO

Criar um profile chamado TESTE que estabeleça as seguintes regras:

◊O usuário poderá estabelecer qualquer número de sessões concorrentes.
◊Em uma única sessão o usuário poderá consumir uma quantidade de CPU ilimitada.
◊Em uma única chamada o usuário não poderá consumir mais de 30 segundos de tempo de CPU.
◊Uma sessão não pode ficar conectada por mais de 60 minutos.
◊Em uma sessão o número de blocos de dados lidos do disco ou da memória ficará sujeito ao limite especificado pelo profile DEFAULT.
◊O tamanho máximo de área da SGA que uma sessão poderá alocar será de 20K.
◊O restante dos limites será garantido pelo profile DEFAULT.

```
SQL> CREATE PROFILE teste
  2  LIMIT SESSIONS_PER_USER    UNLIMITED
  3  CPU_PER_SESSION            UNLIMITED
  4  CPU_PER_CALL               3000
  5  CONNECT_TIME               60
  6  LOGICAL_READS_PER_SESSION  DEFAULT
  7  PRIVATE_SGA                20k;

Perfil criado.
```

Criar um profile chamado pass_control que estabeleça as seguintes regras:

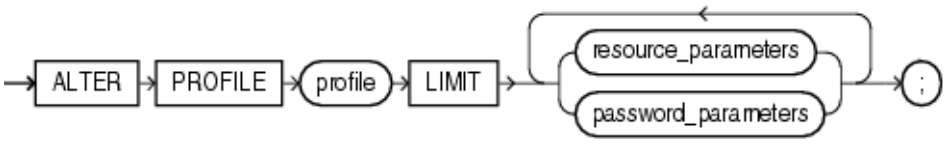
◊O número de tentativas de conexão não pode ser superior a 5.
◊O tempo de vida útil de uma password deve ser de 60 dias.
◊O tempo mínimo em que uma password não deve ser reutilizada é de 60 dias.
◊Usar a função VALIDA_SENHA para estabelecer as regras da password.
◊O usuário deverá ficar bloqueado por pelo menos 1 hora se houver falha de conexão.
◊O período durante o qual o usuário poderá trocar sua password é de 10 dias.

```
SQL> CREATE PROFILE pass_control LIMIT
  2  FAILED_LOGIN_ATTEMPTS 5
  3  PASSWORD_LIFE_TIME 60
  4  PASSWORD_REUSE_TIME 60
  5  PASSWORD_REUSE_MAX UNLIMITED
  6  PASSWORD_VERIFY_FUNCTION valida_senha
  7  PASSWORD_LOCK_TIME 1/24
  8  PASSWORD_GRACE_TIME 10;

Perfil criado.
```

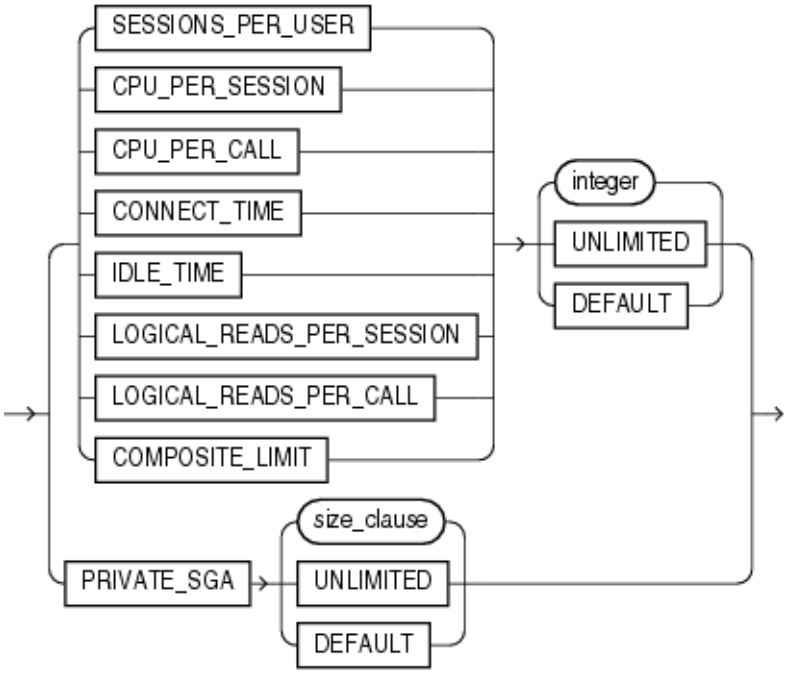
SINTAXE ALTER PROFILE

alter_profile::=



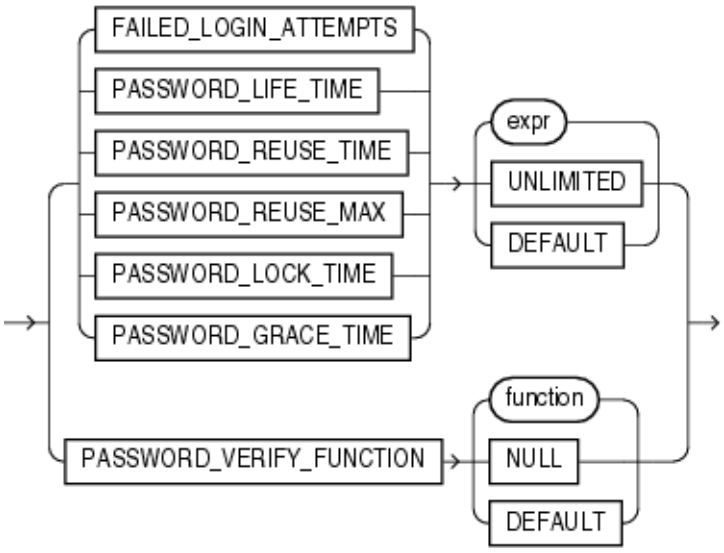
(resource_parameters::=, password_parameters::=)

resource_parameters::=



(size_clause::=

password_parameters::=



Onde:

◊UNLIMITED - indica que o usuário associado a este profile pode usar uma quantidade de recursos ilimitada ou que não existe limite definido para este parâmetro de PASSWORD.

◊DEFAULT - indica que este profile está sujeito aos limites de recurso estabelecidos no profile DEFAULT.

resource_parameters - session

◊SESSIONS_PER_USER - número de sessões concorrentes por usuário.

◊CPU_PER_SESSION - tempo total de CPU em centésimos de segundo.

◊CONNECT_TIME - tempo de conexão em minutos.

◊IDLE_TIME - tempo de inatividade em minutos. É calculado somente para processos servidores. Não afeta o tempo de inatividade de uma aplicação. Este tempo também não é afetado pela execução de longas queries e outras operações.

◊LOGICAL_READS_PER_SESSION - número de blocos de dados lidos por sessão. É uma limitação do número total de leituras tanto de disco quanto de memória.

◊PRIVATE_SGA – especifica a quantidade de espaço na SGA (Session Information) que uma sessão pode alocar (aplicável apenas para arquitetura Multi-Thread e pode ser especificado em M ou K).

◊COMPOSITE_LIMIT - limita o total de recursos para uma sessão expresso em unidades de serviço. Corresponde à soma dos valores de CPU_PER_SESSION, CONNECT_TIME, LOGICAL_READS_PER_SESSION e PRIVATE_SGA multiplicados por seu peso específico na composição. Este peso é dado pelo comando RESOURCE_COST.

---

resource_parameters - call

◊CPU_PER_CALL - tempo de CPU por chamada em centésimos de segundo.

◊LOGICAL_READS_PER_CALL - número de blocos de dados lidos por chamada. É uma limitação do número total de leituras tanto de disco quanto de memória

password_parameters

◊FAILED_LOGIN_ATTEMPTS – especifica o número limite de tentativas falhas que o usuário pode fazer antes do usuário ficar bloqueado.

◊PASSWORD_LIFE_TIME – determina o tempo de vida de uma password(fornecido em dias). Após este períodoa password expirará obrigando o usuário a fazer uma modificação.

◊PASSWORD_REUSE_TIME – determina o número de dias antes do qual uma password não pode ser reutilizada. Quando preenchermos este parâmetro com um número, o parâmetro PASSWORD_REUSE_MAX deve ser preenchido com UNLIMITED.

◊PASSWORD_REUSE_MAX – determina o número de trocas de password necessários para que a password corrente possa reutilizada. Se preenchermos o parâmetro PASSWORD_REUSE_MAX com um número, o parâmetro PASSWORD_REUSE_TIME deve ser preenchido com UNLIMITED.

◊PASSWORD_LOCK_TIME – especifica o número de dias que o usuário ficará bloqueado após o número especificado de tentativas consecutivas falhas de login ter acontecido.

◊PASSWORD_GRACE_TIME – especifica o número de dias, após o tempo da password ter expirado, durante o qual o usuário receberá um aviso a cada tentativa de conexão. Se a password não for modificada durante este período, a password expira.

◊PASSWORD_VERIFY_FUNCTION – permite que indiquemos uma rotina que faça a validação da password informada. O valor DEFAULT para este parâmetro utiliza a rotina padrão do oracle.

restrições

◊Se ambos os parâmetros PASSWORD_REUSE_TIME e PASSWORD_REUSE_MAX receberem o valor UNLIMITED, o Oracle não utilizará estes critérios.

◊A função de validação deve ser criada subordinada ao usuário SYS e deve ter o formato:

```
routine_name (userid_parameter IN VARCHAR(30),
              password_parameter IN VARCHAR (30),
              old_password_parameter IN VARCHAR (30)
) RETURN BOOLEAN
```

A rotina DEFAULT fornecida pela ORACLE pode ser vista no manual Oracle Administrator's Guide (capítulo 22, tópico Password Complexity Verification).

REGRAS PARA ALTERAÇÃO

A modificação de um profile não afeta às sessões em execução. As modificações atingem às sessões subseqüentes.

Para que tenhamos condições de avaliar valores razoáveis para limite, devemos passar um período (típico para a instalação) coletando dados. Para tal podemos:

◊usar a opção AUDIT SESSION para obter informações históricas sobre CONNECT_TIME, LOGICAL_READS_PER_SESSION e LOGICAL_READS_PER_CALL.

◊monitorar informações estatísticas dadas pelo Enterprise Manager para determinar os outros valores.

◊Para as PASSWORDS devemos estabelecer um padrão para todas as passwords da instalação antes de alterarmos os limites no banco de dados.

EXEMPLO

Alterar o profile TESTE para :

◊limitar o número de sessões para 2;

◊limitar a quantidade de CPU para 2 minutos (120 segundos);

◊limitar o tempo de IDLE para 30 minutos;

◊limitar em 1000 o número de leituras lógicas por chamada;

```
SQL> ALTER PROFILE teste LIMIT
  2  SESSIONS_PER_USER 2
  3  CPU_PER_SESSION 12000
  4  IDLE_TIME 30
  5  LOGICAL_READS_PER_CALL 1000;

Perfil alterado.
```

SINTAXE DROP PROFILE

***drop_profile*::=**

Onde:

◊*profile* - determina o nome do profile a ser removido.

◊CASCADE - desassocia o profile de qualquer usuário para o qual o profile tenha sido associado. O Oracle Server, automaticamente, associa o profile DEFAULT para estes usuários. Podemos especificar esta opção para remover um profile que esteja sendo, atualmente, associado a usuários.

O profile DEFAULT não pode ser removido.

Quando um profile é removido, isto afeta apenas às sessões subsequentes, não às atuais.

EXEMPLO

Remover o profile teste e desassociá-lo de qualquer usuário que o esteja usando.

```
SQL> DROP PROFILE teste CASCADE;

Perfil eliminado.
```

O PROFILE DEFAULT

Cada banco de dados possui um profile chamado DEFAULT.

CARACTERÍSTICAS

Quando criamos novos usuários e não determinamos um profile específico, estes ficam implicitamente ao profile DEFAULT.

Todos os limites não especificados de qualquer profile utilizam a especificação do profile DEFAULT.

O profile DEFAULT, inicialmente tem todos os valores ilimitados (UNLIMITED).

O profile DEFAULT pode ser alterado para que nenhum usuário tenha uso ilimitado para os recursos do sistema.

EXEMPLO

Alterar o profile DEFAULT, determinando o limite máximo de 5 sessões por usuário, 36 segundos de tempo de CPU por chamada e 30 minutos de tempo inativo.

```
SQL> ALTER PROFILE default LIMIT
  2  SESSIONS_PER_USER 5
  3  CPU_PER_CALL 3600
  4  IDLE_TIME 30;

Perfil alterado.
```

USANDO LIMITES COMPOSTOS

Podemos limitar de forma combinada alguns limites do sistema com a opção COMPOSITE_LIMIT.

CARACTERÍSTICAS

O limite composto corresponde à soma dos valores de quatro dos limites de recursos: CPU_PER_SESSION, LOGICAL_READS_PER_SESSION, CONNECT_TIME e PRIVATE_SGA. Pode ser combinado com os limites explícitos dos recursos no profile.

Tem a vantagem de permitir flexibilidade adicional quando limitando recursos.

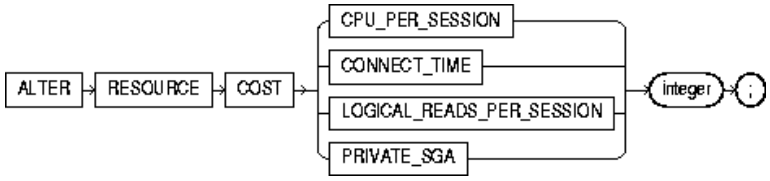
Uma vez que os valores individuais dos limites possuem ordem de grandeza diferente (por exemplo o tempo de CPU sendo expresso de centésimos de segundo obriga-nos a informarmos um valor numérico grande, enquanto que o número de leituras lógicas possui ordem de grandeza muito menor). Desta forma para que pudéssemos estabelecer um peso para estes valores a fim de que tenham um influência de nosso interesse na limitação, usamos RESOURCE_COST.

O uso de custo serve para alterarmos a proporção (o peso) dos limites CPU_PER_SESSION, CONNECT_TIME, LOGICAL_READS_PER_SESSION e PRIVATE_SGA no cálculo do limite composto.

É aplicável somente aos limites compostos de um profile.Não tem valor para, explicitamente, atribuirmos limites individualmente.

O limite (composto ou explícito) que for atingido primeiro irá interromper a atividade da sessão.

SINTAXE ALTER RESOURCE COST



EXEMPLO

```
SQL> ALTER RESOURCE COST
  2  CPU_PER_SESSION 10
  3  LOGICAL_READS_PER_SESSION 5
  4  CONNECT_TIME 20
  5  PRIVATE_SGA 1;

Custo do recurso alterado.
```

OBTENDO INFORMAÇÕES SOBRE OS PROFILES

Podemos consultar o dicionário de dados para listar informações sobre os profiles.

As views associadas a profiles são : DBA_USERS (já vista), USER_RESOURCE_LIMITS, DBA_PROFILES, RESOURCE_COST.

Obter a descrição da view DBA_PROFILES

```
SQL> DESC DBA_PROFILES
 Name                           Null?    Type
 ------------------------------ -------- ----
 PROFILE                        NOT NULL VARCHAR2(30)
 RESOURCE_NAME                  NOT NULL VARCHAR2(32)
 RESOURCE_TYPE                           VARCHAR2(8)
 LIMIT                                   VARCHAR2(40)
```

Onde:

◊PROFILE - determina o nome do profile.

◊RESOURCE_NAME – indica o nome do recurso que está sendo limitado. Por exemplo: CONNECT_TIME, CPU_PER_SESSION, etc.

◊RESOURCE_TYPE – indica o tipo do recurso. Os valores válidos são: PASSWORD ou KERNEL.

◊LIMIT – indica o limite.

Obter a descrição da view RESOURCE_COST.

```
SQL> DESC RESOURCE_COST
 Name                           Null?    Type
 ------------------------------ -------- ----
 RESOURCE_NAME                  NOT NULL VARCHAR2(32)
 UNIT_COST                      NOT NULL NUMBER
```

Onde:

◊RESOURCE_NAME – indica o nome do recurso que está sendo limitado. Por exemplo: CONNECT_TIME, CPU_PER_SESSION, etc.

◊UNIT_COST – indica o peso do recurso. Para utilização com limite composto.

```
SQL> DESC USER_RESOURCE_LIMITS
 Name                           Null?    Type
 ------------------------------ -------- ----
 RESOURCE_NAME                  NOT NULL VARCHAR2(32)
 LIMIT                                   VARCHAR2(40)
```

Onde:

◊RESOURCE_NAME – indica o nome do recurso que está sendo limitado. Por exemplo: CONNECT_TIME, CPU_PER_SESSION, etc.

◊LIMIT – indica o limite.

Obter os limites sobre os recursos válidos atualmente para o usuário ALUNO.

```
SQL> SELECT * FROM USER_RESOURCE_LIMITS;

RESOURCE_NAME                  LIMIT
------------------------------ ------------
COMPOSITE_LIMIT                UNLIMITED
SESSIONS_PER_USER              5
CPU_PER_SESSION                UNLIMITED
CPU_PER_CALL                   3600
LOGICAL_READS_PER_SESSION      UNLIMITED
LOGICAL_READS_PER_CALL         UNLIMITED
IDLE_TIME                      30
CONNECT_TIME                   UNLIMITED
PRIVATE_SGA                    UNLIMITED
```