Bases de Dados

José Machado
Departamento de Informática
Escola de Engenharia
Universidade do Minho

jmac@di.uminho.pt

2017

O modelo relacional

Uma base de dados relaciona um conjunto de tabelas na forma $R(A_1,A2,...,Ar)$, com grau r.

A cardinalidade é o número de elementos de R.

O conjunto dos possíveis valores de um atributo é o domínio $dom(A_i) = D_i$ (e.g., o domínio dum código é o conjunto dos números positivos e o domínio da idade de um trabalhador activo é o conjunto de números inteiros {18,..,65}).

Regra 1: não pode haver 2 tuplos idênticos na mesma relação.
Regra 2: não existe uma ordem pré-definida numa relação.

Álgebra Relacional

UNIÃO

A	B	C
a1	b2	c1
a5	b1	c2
a2	b4	c4
a3	b3	c3

∪

A	B	C
a2	b3	c2
a1	b2	c1
a2	b4	c4

=

A	B	C
a1	b2	c1
a5	b1	c2
a2	b4	c4
a3	b3	c3
a2	b3	c2

INTERSECÇÃO

A	B	C
a1	b2	c1
a5	b1	c2
a2	b4	c4

∩

A	B	C
a2	b3	c2
a1	b2	c1
a2	b4	c4

=

A	B	C
a1	b2	c1
a2	b4	c4

DIFERENÇA

A	B	C
a1	b2	c1
a5	b1	c2
a2	b4	c4

−

A	B	C
a2	b3	c2
a1	b2	c1
a2	b4	c4

=

A	B	C
a5	b1	c2

Álgebra Relacional

PRODUTO CARTESIANO

A	B
a1	b2
a5	b1
a2	b4

X

C	D
c2	d3
c1	d2

=

A	B	C	D
a1	b2	c2	d3
a1	b2	c1	d2
a5	b1	c2	d3
a5	b1	c1	d2
a2	b4	c2	d3
a2	b4	c1	d2

PROJECÇÃO

$\pi_{(A,C)}$

A	B	C
a2	b3	c4
a1	b2	c1
a2	b4	c4

=

A	C
a2	c4
a1	c1

SELECÇÃO

$\sigma_{(A = a2)}$

A	B	C
a2	b3	c2
a1	b2	c1
a2	b4	c4

=

A	B	C
a2	b3	c2
a2	b4	c4

JUNÇÃO

A	B	C
a1	b2	c1
a5	b1	c2
a2	b4	c4

∞

C	D
c2	d3
c1	d2
c2	d1

=

A	B	C	D
a1	b2	c1	d2
a5	b1	c2	d3
a5	b1	c2	d1

Álgebra Relacional

DIVISÃO

A	B	C	D
a1	b2	c2	d3
a5	b1	c1	d2
a2	b4	c1	d2
a3	b5	c4	d4
a2	b4	c2	d3
a1	b2	c1	d2

÷

C	D
c2	d3
c1	d2

=

A	B
a1	b2
a2	b4

Álgebra Relacional

Operações básicas :

$R \cup S$

$R - S$

$R \times S$

$\Pi_{Ai,...,Aj}(R) = \Pi_{i,...,j}(R)$

$\sigma_{Cond}(R)$

Operações adicionais:

$R \cap S$

$R \div S$

$R \bowtie_{i\Theta j} S$

(Θ-junção e.g.,<-junção)

$R \bowtie S$

$R \ltimes S$

Equivalências

a) $R \cap S = R - (R - S)$

b) $R(A1,...,Ar)\ \ S(Ar\text{-}s+1,..,As)\ \ r > s$ e $S \neq \Phi$

$$R \div S = \Pi_{1,...,r\text{-}s}(R) - \Pi_{1,...,r\text{-}s}((\Pi_{1,...,r\text{-}s}(R) \times S) - R)$$

c) $R \bowtie_{i\Theta j} S = \sigma_{\$i\ \Theta\ \$r+j}(RXS)$

d) $R \bowtie S = \Pi_{i1,...,im}(\sigma_{R.Ai=S.Ai \text{ e } ... \text{ e } R.Ak=S.Ak}(R \times S))$

e) $R \ltimes S = \Pi_R(R \bowtie S)$

Bases de Dados Dedutivas

R(A1,...,Ar) S(A1,...,Ar) T(A1,...,Ar)
R ∪ S = T
t(A1,...,Ar) ← r(A1,...,Ar). ou t(A1,...,Ar) ← r(A1,...,Ar) ∨ s(A1,...,Ar).
t(A1,...,Ar) ← s(A1,...,Ar).

R – S = T
t(A1,...,Ar) ←r(A1,...,Ar) ∧ ¬ s(A1,...,Ar).

R(A1,...,Ar) S(Ar+1,...,Ar+s) T(A1,...,Ar+s) T = R X S
t(A1,...,Ar+s) ← r(A1,...,Ar) ∧ s(Ar+1,...,Ar+s).

R(A1,..,Ar) T(Ai,...,Aj) i<= r, j<= r $\Pi_{Ai,...,Aj}(R) = T$
t(Ai,...,Aj) ← r(A1,...,Ar).

R(A1,...,Ar) $\sigma_{Cond}(R) = T$
t(A1,...,Ar) ← r(A1,...,Ar) ∧ Cond.

Bases de Dados Dedutivas

R(A1,...,Ar) S(A1,...,Ar) T(A1,...,Ar)
$R \cap S = T$ $t(A1,...,Ar) \leftarrow r(A1,...,Ar) \wedge s(A1,...,Ar)$.

$R - (R - S) = T$ $t(A1,...,Ar) \leftarrow r(A1,...,Ar) \wedge \neg (r(A1,...,Ar) \wedge \neg s(A1,...,Ar))$.
$R - (R - S) = R \cap S$ $A \wedge \neg (A \wedge \neg B) = A \wedge (\neg A \vee B) = A \wedge B$

$R \div S = T$ R(A1,...,Ar) S(Ar-s+1,...,Ar) T(A1,...,Ar-s) $R \div S = \Pi_{1,...,r-s}(R) - \Pi_{1,...,r-s}((\Pi_{1,...,r-s}(R) \times S) - R)$

$r1(A1,...,Ar-s) \leftarrow r(A1,...,Ar)$. $R1 = \Pi_{1,...}(R)$
$r2(A1,...,Ar) \leftarrow r1(A1,...,Ar-s) \wedge s(Ar-s+1,...,Ar)$. $R2 = R1 \times S$
$r3(A1,...,Ar) \leftarrow r2(A1,...,Ar) \wedge \neg r(A1,...,Ar)$. $R3 = R2 - R$
$r4(A1,...,Ar-s) \leftarrow r3(A1,...,Ar)$. $R4 = \Pi_{1,...}(R3)$
$t(A1,...,Ar-s) \leftarrow r1(A1,...,Ar-s) \wedge \neg r4(A1,...,Ar-s)$. $T = R1 - R4$

$R \bowtie_{i\Theta j} S = T$ $t(A1,...,Ar+s) \leftarrow r(A1,...,Ar) \wedge s(Ar+1,...,Ar+s) \wedge Ai \ \Theta \ Ar+j$.
$R \bowtie S = T$ e k é o número de atributos comuns
$t(A1,...,Ar+s-k) \leftarrow r(A1,...,Ar) \wedge s(Ar+1,...,Ar+s) \wedge Ai1=Aj1 \wedge ... \wedge Aik = Ajk$.

SQL

União : (select * from R) union (select * from S)

Diferença : (select * from R) minus(select * from S)

select * from R where R.* not in (select * from S)

Produto : select R.*,S.* from R,S

Projecção : select Ai,..,Aj from R

Selecção : select * from R where Cond

Intersecção : (select * from R) intersect (select * from S)

select * from R where R.* in (select * from S)

Θ-junção : select R.*,S.* from R,S where Ai Θ Ar+j

Junção : select R.*,S.Ar+1,S.Ar+s-k from R,S where R.Ai1 = S.Ai1 and ...
and R.Aik = S.Aik

Divisão : (ver exemplo e aplicar $\forall x\ P(x) = \neg \exists x\ \neg P(x)$)

Optimização de queries

Avaliação do desempenho de uma interrogação (query)

Os procedimentos comuns para avaliar o desempenho de uma interrogação são:

- reescrever a interrogação em álgebra relacional;
- encontrar algoritmos para calcular resultados intermédios da forma mais simples;
- executar os algoritmos e obter os resultados.

Avaliação do desempenho de uma query

Estes procedimentos podem esbarrar em dificuldades na medida em que na álgebra relacional podem-se obter expressões equivalentes, há problemas em lidar com sub-interrogações e os tamanhos intermédios obtidos são muito importantes. Os resultados da avaliação dependem da extensão das relações, o que nem sempre é possível saber dada a dinâmica de crescimento das tabelas.

Avaliação do desempenho de uma query

Medicos(M,NM,CE,Anos) 50 bytes por tuplo, 100 tuplos

Consultas(M,P,D,Preço) 20 bytes por tuplo, 5000 tuplos

paciente P = 200 só tem 1 consulta!

Interrogação:

SELECT m.NM FROM Medicos m,Consultas c where
c.M = m.M and P = 200 and Anos > 5

Tradução para a Àlgebra relacional:

$\Pi_{NM}(\sigma_{P=200 \wedge Anos>5}(Medicos \bowtie Consultas))$

Avaliação do desempenho de uma query

A selecção e a projecção são realizadas após a junção!
Custo da solução: a volta de 100 * 5000 * 2 = 1 000 000 tuplos processados!

Nova expressão em Álgebra Relacional:
$\Pi_{NM}(\sigma_{P=200}$ Consultas $\bowtie \sigma_{Anos>5}$Medicos)

É feito um *full table scan* às 2 tabelas, obtendo duas tabelas mais pequenas (cardinalidade respectivamente C1 e 1) e depois é feita a junção entre as 2 tabelas resultantes.

Custo da solução: a volta de 100 + 5000 + C1 tuplos processados!

Por exemplo admitindo uma redução de cerca de 50% na selecção sobre Medicos (C1 = 50), o resultado é 100 + 5000 + 50 = 5150

Optimização de queries

As expressões seguintes são expressões equivalentes mas com custos muito diferentes:

$\sigma_C(E_1 \cup E_2) = \sigma_C(E_1) \cup \sigma_C(E_2)$

$\sigma_C(E_1 \times E_2) = E_1 \bowtie_C E_2$

$\sigma_C(E_1 - E_2) = \sigma_C(E_1) - \sigma_C(E_2)$

$\sigma_C(E_1 \times E_2) = \sigma_C(E_1) \times E_2$ se o conjunto de atributos de E_1 pertence a C.

$\sigma_C(E_1 \cap E_2) = \sigma_C(E_1) \cap \sigma_C(E_2)$

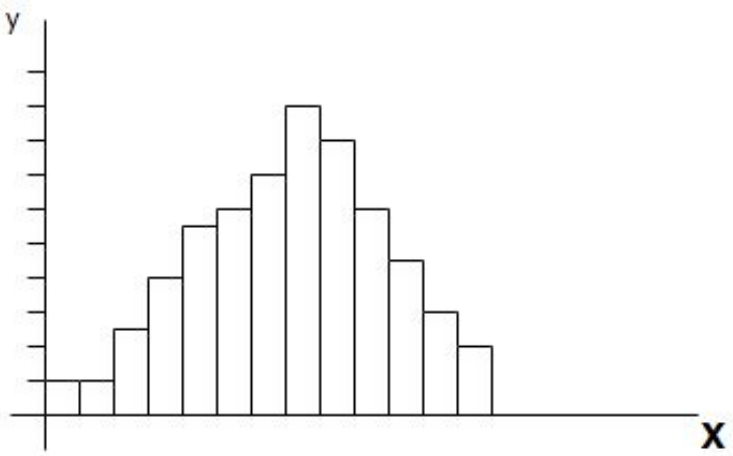
$\Pi_L(E_1 \bowtie E_2) = \Pi_L(\Pi_{L \cup AE2) \cap AE1}(E_1) \bowtie \Pi_{L \cup AE1) \cap AE2}(E_2))$

$\Pi_L(\sigma_C(E_1)) = \Pi_L(\sigma_C(\Pi_A(E_1)))$ onde A é o conjunto de atributos mencionados em C.

Histogramas

Os histogramas podem ser usados para avaliar o custo da seleção.

Permitem analisar cardinalidades por intervalos.

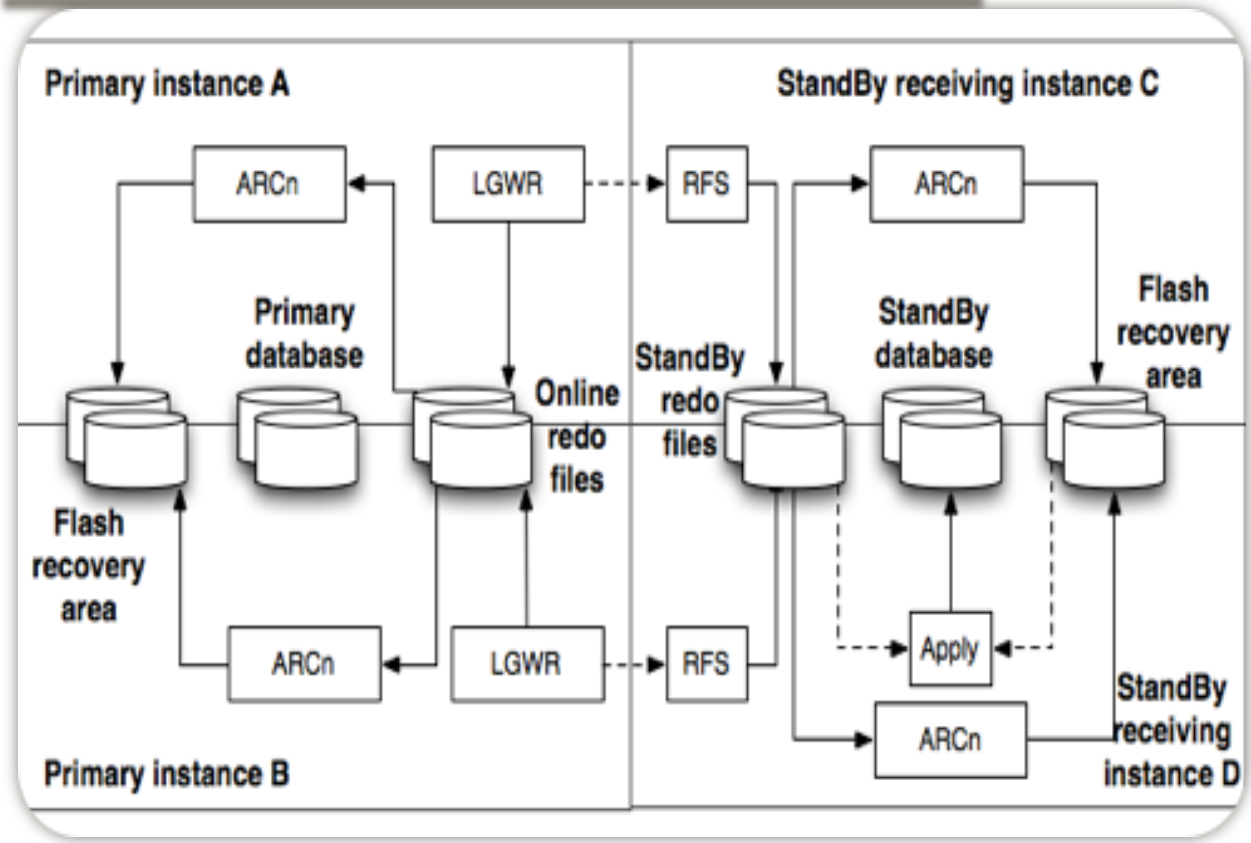


Tuning

O Tuning é um processo de acompanhamento da performance das interrogações de forma a escolher bons caminhos.

O tuning pode sugerir a reescrita das interrogações / código, a indexação de algumas tabelas e a eliminação de *full table scans*.

RAC e DataGuard



Administração de bases de dados Oracle

- Tipos de utilizadores
- Tarefas de um administrador
- Identificação da versão de software
- Segurança e privilégios
- Autenticação
- Criação e manutenção de “Password Files”
- Utilitários de administração

Tipos de utilizadores

- Administradores de bases de dados
- Responsáveis pela segurança
- Administradores de rede
- Programadores de aplicações
- Administradores de aplicações
- Utilizadores de bases de dados

Administradores de bases de dados

Cada base de dados tem pelo menos um administrador (DBA). Considerando que uma base de dados Oracle pode ser um sistema de elevada complexidade e ter muitos utilizadores o administrador pode não ser uma única pessoa.Nestes casos existe um grupo de DBAs que partilham as seguintes responsabilidades:

- Instalar e actualizar o servidor Oracle e as ferramentas aplicacionais
- Alocar espaço de armazenamento e gerir as necessidades futuras em termos de capacidade de armazenamento
- Criar estruturas para o armazenamento (tablespaces) para serem usadas pelas aplicações desenvolvidas pelos programadores
- Criar objectos ou entidades principais (tabelas, views, índices) para executar eficientemente as aplicações
- Modificar a estrutura da base de dados a partir da informação enviada pelos programadores
- Gerir utilizadores e segurança do sistema
- Verificar licenças de software Oracle
- Controlar e monitorizar o acesso dos utilizadores à base de dados
- Monitorizar e optimizar a performance da base de dados
- Gerir cópias de segurança e recuperação de informação perdida
- Manter os dados arquivados em suportes auxiliares
- Contactar a Oracle Corporation para pedir suporte técnico.

Responsáveis pela segurança

Em alguns casos, são atribuídos um ou mais responsáveis pela segurança a uma base de dados. O responsável pela segurança gere utilizadores, controla e monitoriza o acesso à base de dados e mantem a segurança do sistema, libertando o administrador da base de dados dessas responsabilidades.

Administradores de rede

Pode ser contratados um ou mais administradores de rede para administrar os produtos de rede da Oracle, tal como Oracle Net Services. Em ambientes distribuídos e na gestão de bases de dados distribuídas estas responsabilidades são mais visíveis.

Programadores de aplicações

Os programadores desenvolvem, algumas destas tarefas em colaboração com os DBAs:

- Desenhar e desenvolver aplicações
- Desenhar a estrutura da base de dados para correr uma aplicação
- Estimar as necessidades em termos de capacidade de armazenamento
- Especificar alterações à estrutura da base de dados para correr uma aplicação
- Relatar necessidades estruturais aos administradores
- Ajustar as aplicações à base de dados na fase de desenvolvimento
- Estabelecer medidas de segurança aplicacionais durante o desenvolvimento

Administradores de aplicações

Podem ser contratados um ou mais administradores de aplicações para uma aplicação em particular, libertando os DBAs desta responsabilidade, Cada aplicação pode ser o seu próprio administrador.

Utilizadores de bases de dados

Os utilizadores de bases de dados podem ter as seguintes permissões:

- Aceder a, modificar ou apagar dados quando permitido;
- Gerar relatórios a partir dos dados.

Tarefas do administrador de bases de dados

- Tarefa 1: Avaliar o hardware do servidor da base de dados
- Tarefa 2: Instalar software
- Tarefa 3: Projectar a base de dados
- Tarefa 4: Criar e abrir a base de dados
- Tarefa 5: Copiar os dados da base de dados
- Tarefa 6: Gerir os utilizadores do sistema
- Tarefa 7: Implementar o desenho da base de dados
- Tarefa 8: Copiar as funcionalidades da base de dados
- Tarefa 9: Ajustar a performance da base de dados

Administrador da base de dados – Segurança e privilégios

A conta do sistema operativo para o DBA

O administrador deve ter acesso a comandos do sistema operativo para executar operações fndamentais para a administração da base de dados. A conta do sistema operativo do Administrador deve ter maiores privilégios de que a conta de um simples utilizador Oracle. Os ficheiros Oracle não devem ser armazenados em sub-directorias da conta do Administrador no sistema operativo mas este deve ter acesso a esses ficheiros.

Nomes de utilizador do Administrador da base de dados

Duas contas são criadas automaticamente na base de dados:

- SYS (password inicial: CHANGE_ON_INSTALL)
- SYSTEM (password inicial: MANAGER)

Comentários adicionais sobre segurança

- Durante a instalação do software de base de dados usando o *Database Configuration Assistant (DCBA)*, todos os utilisadores estão bloqueados (locked) e desactivados (expired), excepto SYS, SYSTEM, SCOTT, DBSNMP, OUTLN, AURORAJISUTILITY$, AURORA$ORB$UNAUTHENTICATED e OSE$HTTP$ADMIN. Para reactivar contas bloqueadas, o DBA deve desbloquear a conta e reatribuir-lhe uma nova palavra de passe.
- O DBCA pede uma palavra de passe para os utilizadores SYS e SYSTEM durante a fase de instalação da base de dados. O comando CREATE DATABASE submetido manualmente também permite atribuir estas duas palavras de passé fundamentais.

Nomes de utilizadores para administrar a base de dados

SYS

- **O utilizador SYS é criado automaticamente sempre que a base de dados é ciado com o papel de DBA.**
- **Todas as tabelas e views de base para o dicionário de dados da base de dados são armazenados no esquema SYS. Estas tabelas e views são fundamentais para o bom funcionamento do sistema e são apenas manipuladas pelo motor Oracle para manter a integridade dos dados. Não é aconselhado criar novas tabela no esquema SYS, podendo apenas alterar-se os parametros de armazenamento do tablespace associado. Os utilizadores são devem usar a conta SYS para realizar exclusivamente operações de administração.**

SYSTEM

- **O utilizador SYSTEM é criado automaticamente sempre que a base de dados é ciada com o papel de DBA.**
- **O utilizador SYSTEM serve para criar tabelas e views adicionais para mostrar informação de caracter administrativo e tabelas ou views internas usadas em ferramentas ou opções do Oracle. Não se deve usar esta valência para o interesse individual de uitilizadores.**

O papel de DBA (DBA role)

- **Um papel pré-definido, DBA, é automaticamente criado para cada base de dados. Este papel contem muitos privilégios do sistema, e deve ser atribuído apenas a utilizadores que necessitam de realizar todas as operações de administração.**

O papel de DBA não incluí os provilégios SYSDBA e SYSOPER. Estes privilégios são especiais e além de permtir fazer tarefas básicas de administração também permitem criar a base de dados e executar a instâncias de *startup* e *shutdown* de base de dados.

Autenticação do Administrador

O DBA executa operações especiais (e.g. ligar (*start*) e desligar (*shutdown*) a base de dados). Por isso, os utilizadores com privilégios de administração obdecem a um esquema de autenticação com preocupações especiais em matéria de segurança.

Privilégios de administração

Os dois privilégios de administração para operações básicas de administração são o SYSDBA e o SYSOPER. Dependendo do nível de autorização exigido, um desses dois privilégios podem ser atribuídos a um administrador.

SYSDBA e SYSOPER

Privilégio	Operações
SYSDBA Este privilégio oferece ao utilizador as funcionalidades do utilizador SYS	STARTUP e SHUTDOWN ALTER DATABASE: open, mount, backup, e change character set CREATE DATABASE CREATE SPFILE ARCHIVELOG e RECOVERY Incluí o privlégio RESTRICTED SESSION
SYSOPER Este privilégio permite ao utilizador desenvolver tarefas operacionais básicas mas sem a possibilidade de aceder aos dados individuais dos outros utilizadores.	STARTUP e SHUTDOWN CREATE SPFILE ALTER DATABASE OPEN/MOUNT/BACKUP ARCHIVELOG e RECOVERY Incluí o privlégio RESTRICTED SESSION

Ligação com privilégio de Administração

Assuma que o utilizador *scott* executa:

```
CONNECT scott/password
CREATE TABLE admin_test(name VARCHAR2(20));
```

Mais tarde executa o comando seguinte:

```
CONNECT scott/password AS SYSDBA
SELECT * FROM admin_test;
```

scott recebe a seguinte mensagem de erro

```
ORA-00942: table or view does not exist
```

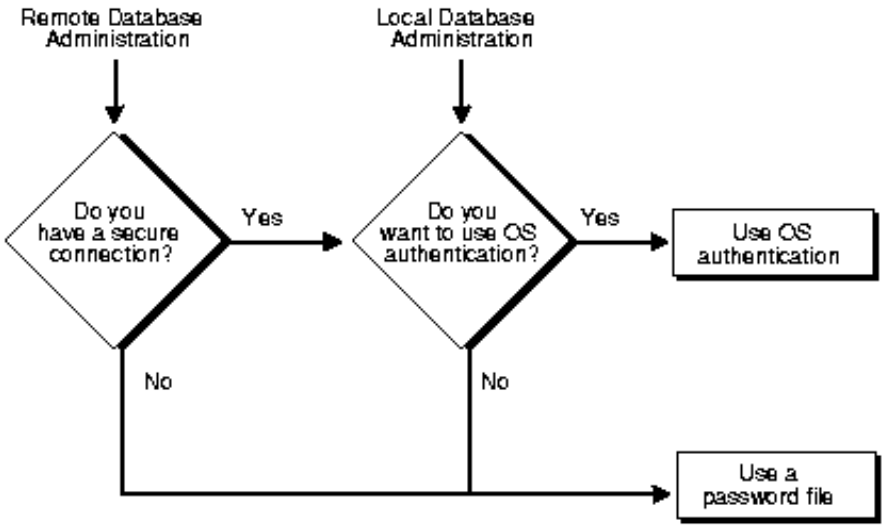
A tabela foi criada no esquema *scott* e não no esquema *SYS*. O comando correcto seria:

```
SELECT * FROM scott.admin_test;
```


Métodos de autenticação

Existem dois métodos de autenticação:

- Autenticação via Sistema Operativo (OS authentication)
- Password files



Autenticação OS

Para permitir a autenticação de um administrador usando o sistema operativo deve-se:

- **Criar uma conta no sistema operativo**
- **Adicione o utilizador aos grupos OSDBA ou OSOPER do sistema operativo**

Verifique se o parâmetro REMOTE_LOGIN_PASSWORDFILE é NONE (valor por omissão).

Ligação usando autenticação OS

Por exemplo, no caso de ligação local:
CONNECT / AS SYSDBA
CONNECT / AS SYSOPER

No caso de uma ligação remota:

CONNECT /@net_service_name AS SYSDBA
CONNECT /@net_service_name AS SYSOPER

OSDBA e OSOPER

Dois grupos especiais do sistema operativo controlam as ligações dos administradores da base de dados em situação de autenticação via sistema operativo. Estes grupos são designados por OSDBA e OSOPER e dependem do sistema operativo:

Grupo no Sistema OPerativo	UNIX	Windows
OSDBA	dba	ORA_DBA
OSOPER	oper	ORA_OPER

Autenticação por Password File

Para autenticação por password file, um administrador deve:

- Criar uma conta de sistema operativo;
- Criar a password file (caso não exista) usando o utilitário ORAPWD:
  ORAPWD FILE=filename PASSWORD=password ENTRIES=max_users
- Inicializar o parâmetro REMOTE_LOGIN_PASSWORDFILE com o valor EXCLUSIVE.
- Ligar-se à base de dados com o utilizador SYS (ou outro com privilégio de administrador).
- Criar o utilizador se não existir. Atribuir o privilégio SYSDBA ou SYSOPER ao utilizador:
  GRANT SYSDBA to scott;

Este comando adiciona o utilizador à password file, permitindo a ligação como SYSDBA.

Ligação usando autenticação por Password File

CONNECT scott/tiger AS SYSDBA

Utilitários de Administração

SQL*Loader

O SQL*Loader é usado por administradores ou utilizadores normais. Permite carregar dados a partir de ficheiros do sistema (e.g. ficheiros em formato texto) para tabelas de bases de dados Oracle.

Export e Import

Os utilitários *export* e *import* permitem mover data em formato proprietário entre bases de dados Oracle. Por exemplo, ficheiros criados através do *export* podem ser depois carregados através do *import* noutras bases de dadis que corram em máquinas com o mesmo sistema operativo ou não.

Exemplos

Utilização de uma máscara e da função TO_DATE :

```
INSERT INTO empregado (n_emp, nome, categoria, entrada, salario)
  VALUES (1234, JOSE', 'PROFESSOR',
  TO_DATE ('22-02-2002', 'DD-MONTH-YYYY'), 3000);
```

Utilização de restrições ou *constraints*:

```
CREATE TABLE socio
    ( nsocio     NUMBER(9)          PRIMARY KEY,
    nome         VARCHAR2(30)       NOT NULL,
    num_fiscal   NUMBER(9)          UNIQUE,
    a_quota      NUMBER(9)          CHECK (a_quota <5000)
    );
```


Exemplos

Criação de tabela com partições usando o campo data_da_venda:

```
CREATE TABLE  venda
    (cod_venda        NUMBER(9)        NOT NULL,
     vendedor              VARCHAR2(30)     NOT NULL,
     produto               VARCHAR2(30)     NOT NULL,
     distrito      VARCHAR2(30)     NOT NULL,
     data_da_venda     DATE                          NOT NULL,
     valor                 NUMBER(12)                    NOT NULL)
   PARTITION BY RANGE (data_da_venda)
    (PARTITION vendas_jan   VALUES LESS THAN
      (TO_DATE ('1-Fev-2002','DD-MON-YYYY'))       TABLESPACE users,
    PARTITION vendas_fev    VALUES LESS THAN
      (TO_DATE ('1-Mar-2002','DD-MON-YYYY'))       TABLESPACE users,
    PARTITION vendas_mar    VALUES LESS THAN
      (TO_DATE ('1-Abr-2002','DD-MON-YYYY'))       TABLESPACE users,
    PARTITION vendas_q2 VALUES LESS THAN
      (TO_DATE ('1-Jul-2002','DD-MON-YYYY'))       TABLESPACE users,
    PARTITION vendas_q3 VALUES LESS THAN
      (TO_DATE ('1-Out-2002','DD-MON-YYYY'))       TABLESPACE users,
    PARTITION vendas_q4 VALUES LESS THAN
      (TO_DATE ('1-Jan-2003','DD-MON-YYYY'))       TABLESPACE users);
```


Exemplos

Criação de sequências:

```
CREATE SEQUENCE cod_seq
        MINVALUE 1
        MAXVALUE 9999999
        START WITH 1
        INCREMENT BY 1
        CACHE 20;
```

Utilização de sequências:

```
INSERT INTO venda
    (cod_venda, vendedor, produto, distrito, data_da_venda,
     valor )
  VALUES
    (cod_seq.nextval, 'José Ferreira', 'Artigo B', 'Braga',
    TO_DATE('5-4-2005','DD-MM-YYYY'), 2000);
```


Exemplos

Criação de procedimentos:

```
CREATE OR REPLACE  PROCEDURE total_vendas_distrito
    (
    aux  IN       varchar2,
    total OUT     number
    )
 AS  BEGIN
   SELECT SUM(valor) INTO total FROM venda2 WHERE distrito =
   aux;
 END;
```


Exemplos

Invocação de *packages*:

```
SQL> var numero_aleatorio  number
SQL> execute dbms_random.seed(12345678)
SQL> execute :numero_aleatorio  :=dbms_random.random
SQL> print numero_aleatorio
NUMERO_ALEATORIO
---------------
722700502
```


Exemplos

Utilização de *triggers*:

```
CREATE OR REPLACE TRIGGER regista_del
    AFTER DELETE ON vendas FOR EACH ROW
      BEGIN
        INSERT INTO registo_del (operacao, data, utilizador)
            VALUES ('DELETE', SYSDATE, USER);
      END;

CREATE OR REPLACE TRIGGER reg_logon
  AFTER LOGON ON DATABASE
    BEGIN
      INSERT INTO registo_del(operacao,data,utilizador)
          VALUES('LOGON',sysdate,user);
     END;
```


Exemplos

Criação de sinónimos públicos:

```
CREATE PUBLIC SYNONYM empregados FOR scott.emp;
```

Criação de Ligações (*database links*):

```
CREATE DATABASE LINK vendas.com USING 'sales_us';
CREATE PUBLIC DATABASE LINK vendas
    CONNECT TO scott IDENTIFIED BY tiger
    USING 'sales_us';
```

Utilizar *database links*:

```
SELECT * FROM employee@vendas;
```

Atribuição de privilégios sobre a forma de *roles*:

```
CREATE ROLE estagiario  NOT IDENTIFIED;
GRANT INSERT ON  tomanix.vendas TO "estagiario";
GRANT SELECT ON  tomanix.vendas TO "estagiario";
GRANT UPDATE ON  tomanix.vendas TO "estagiario";
GRANT estagiario TO scott;
```


Exemplos

Consultar a informação sumária sobre a SGA (System Global Area):
SELECT * FROM v$sga;

NAME	VALUE
Fixed Size	282576
Variable Size	83886080
Database Buffers	33554432
Redo Buffers	532480

Exemplos

Consultar a informação sobre os parâmetros dinâmicos da SGA:

```
SELECT * FROM v$sga_dynamic_components;
COMPONENT               CURRENT_SIZE   MIN_SIZE   MAX_SIZE
---------               ------------ --------  ---------
shared pool             50331648    50331648     50331648
large pool              8388608     8388608      8388608
buffer cache            25165824    25165824     25165824
```

Alterar o parâmetro SHARED_POOL_SIZE:

```
ALTER SYSTEM SET shared_pool_size = 50331648;
```

Forçar um *checkpoint*:

```
ALTER SYSTEM CHECKPOINT;
```

Consultar a tabela de *jobs*:

```
SELECT * FROM all_jobs;
```

Consultar os *jobs* em execução:

```
SELECT * FROM all_jobs;
```

Consultar a lista de *tablespaces* existentes na base de dados:

```
SELECT * dba_tablespaces;
```

Consultar a lista dos *free extents*, ou seja, espaço livre nos vários *tablespaces*:

```
SELECT * FROM dba_free_space;
```


Exemplos

Atribuir um determinado *tablespace* a um utilizador:

```
SELECT * FROM dba_free_space; CREATE  USER manuela IDENTIFIED  BY
      password_da_manuela
   DEFAULT  TABLESPACE  USERS;
ALTER  USER manuela QUOTA 0 ON SYSTEM;
```

Alterar o *tablespace* por omissão para um utilizador já existente:

```
ALTER  USER fernando DEFAULT  TABLESPACE  USERS;
```

Criação de um novo *undo tablespace*:

```
CREATE UNDO  TABLESPACE  UNDOTBS2
 DATAFILE  'D:\ORACLE\ORADATA\ORA920\UNDOTBS2.DBF'   SIZE  50M;
```

Criação de um *temporary tablespace*:

```
CREATE TEMPORARY  TABLESPACE  TEMP2
   TEMPFILE  'D:\ORACLE\ORADATA\ORA920\TEMP2.ora'   SIZE  20M;
```

Criação de utilizadores, indicando qual o seu *temporary tablespace* por omissão:

```
ALTER  TABLESPACE  "USERS_LOCAL_MANAGED"   READ ONLY;
```


Exemplos

CREATE USER manuela IDENTIFIED BY password_da_manuela
DEFAULT *TABLESPACE* USERS
TEMPORARY *TABLESPACE* TEMP2;

Atribuir um *temporary* *TABLESPACE* TEMP2;
***tablespace* a um utilizador já existente:**
ALTER USER fernando TEMPORARY

Criação de um *tablespace locally managed*:
CREATE *TABLESPACE* "USERS_LOCAL_MANAGED"
DATAFILE 'D:*ORACLE*\ORADATA\ORA920\USERS.DBF'
SIZE 5M **EXTENT MANAGEMENT LOCAL**;

Criação de um *tablespace dictionary managed*
CREATE *TABLESPACE* "USERS_DICT_MANAGED"
DATAFILE 'D:*ORACLE*\ORADATA\ORA920\USERS.DBF'
SIZE 5M **EXTENT MANAGEMENT DICTIONARY**;

Alterar um *tablespace* para *read-write*:
ALTER *TABLESPACE* "USERS_LOCAL_MANAGED" READ WRITE ;

Exemplos

Recriar índices noutro *tablespace*:
ALTER INDEX livro_idx REBUILD *TABLESPACE* INDICES;

Mover uma tabela para outro *tablespace*:
ALTER TABLE livro MOVE *TABLESPACE* DATA1;

Definir uma quota num *tablespace*:
ALTER USER fernando QUOTA 5M ON USERS;

Colocar um *tablespace offline*:
ALTER *TABLESPACE* DATA1 OFFLINE NORMAL;

Exemplos

Consultar a lista dos nomes, tamanhos e *tablespaces* associados aos *datafiles*:

```
SELECT file_name, blocks, tablespace_name
   FROM dba_data_files;
```

Consultar a lista dos *control files* e a sua localização:

```
SELECT * FROM V$CONTROLFILE;
```

Fazer um *backup* do *control file*:

```
ALTER DATABASE BACKUP CONTROLFILE TO 'd:\backups\control.bkp';
```

Criação de um *script SQL* que permita recriar o *control file*:

```
ALTER DATABASE BACKUP CONTROLFILE TRACE;
```


Exemplos

Consultar a informação sobre os *redo log groups* e os seus membros:
SELECT * FROM V$LOGFILE;

Criação de um novo grupo de *redo logs*:
ALTER DATABASE ADD LOGFILE GROUP 5
('c:\oradata\log5a.rdo',d:*Oracle*\oradata\ora920\log5b.rdo')
SIZE 200K;

Adicionar um membro (*redo* logs) a um grupo:
ALTER DATABASE ADD LOGFILE MEMBER
'c:\oradata\log1b.rdo' TO GROUP 1;

Remover membros (*redo* logs) do grupo:
ALTER DATABASE DROP LOGFILE MEMBER 'c:\oradata\log1b.rdo';

Exemplos

Forçar uma mudança de *redo log group*:
ALTER SYSTEM SWITCH LOGFILE;

Criação de um *tablespace* usando *Oracle Managed Files* (OMF):
ALTER SYSTEM SET DB_CREATE_FILE_DEST = 'C:\ORADATA';
CREATE *TABLESPACE* dados3 *DATAFILE* AUTOEXTEND ON
MAXSIZE 500M;

Criação de um *tablespace* usando OMFs com dois *datafiles*:
ALTER SYSTEM SET DB_CREATE_FILE_DEST = 'C:\ORADATA';
CREATE *TABLESPACE* indices2 *DATAFILE* SIZE 5M, SIZE 5M;

Exemplos

Adicionar um *datafile* a um *tablespace* usando OMFs:

```
ALTER SYSTEM SET DB_CREATE_FILE_DEST = 'C:\ORADATA';
ALTER TABLESPACE dados3 ADD DATAFILE AUTOEXTEND ON
   MAXSIZE 100M;
```

Consultar a vista USER_OBJECTS que mostra todos os objectos pertencentes ao utilizador que está ligado à base de dados:

```
SELECT * FROM USER_OBJECTS;
```

Consultar a vista DBA_OBJECTS que dá uma vista geral da base de dados mostrando os objectos pertencentes ao utilizador SYS:

```
SELECT * FROM DBA_OBJECTS;
```

Consultar a vista ALL_OBJECTS que dá uma perspectiva de todos os objectos da base de dados que o utilizador corrente tem acesso:

```
SELECT * FROM ALL_OBJECTS;
```

Adicionar um *datafile* a um *tablespace* usando OMFs:

```
ALTER SYSTEM SET DB_CREATE_FILE_DEST = 'C:\ORADATA';
ALTER TABLESPACE dados3 ADD DATAFILE AUTOEXTEND ON
   MAXSIZE 100M;
```


Exemplos

Consultar a vista USER_OBJECTS que mostra todos os objectos pertencentes ao utilizador que está ligado à base de dados:

```
SELECT * FROM USER_OBJECTS;
```

Consultar a vista DBA_OBJECTS que dá uma vista geral da base de dados mostrando os objectos pertencentes ao utilizador SYS:

```
SELECT * FROM DBA_OBJECTS;
```

Consultar a vista ALL_OBJECTS que dá uma perspectiva de todos os objectos da base de dados que o utilizador corrente tem acesso:

```
SELECT * FROM ALL_OBJECTS;
```

Consultar a vista V$SGA que mostra informação sobre a *System Global Area* (SGA):

```
SELECT * FROM V$SGA;
NAME                 VALUE
-------------------- ----------
Fixed Size             453512
Variable Size        113246208
Database Buffers      25165824
Redo Buffers            667648
```


Exemplos

Consultar a vista V$TABLESPACE que mostra os vários *tablespaces* na base de dados:

SELECT * FROM V$*TABLESPACE*;

TS#	NAME	INC
0	SYSTEM	YES
1	UNDOTBS1	YES
9	USERS	YES
2	TEMP	YES

Exemplos

Consultar o espaço livre em cada *tablespace* usando a *view* DBA_FREE_SPACE:

```
SELECT tablespace_name, sum(bytes) FROM dba_free_space
    GROUP BY tablespace_name;
TABLESPACE_NAME                SUM(BYTES)
---------------                ----------
CWMLITE                          11141120
DRSYS                            10813440
EXAMPLE                            131072
INDX                             26148864
ODM                              11206656
SYSTEM                            4325376
TOOLS                             4128768
UNDOTBS1                        204865536
USERS                            26148864
XDB                               196608
10 linhas seleccionadas.
```


Exemplos

Criação um novo *datafile* no *tablespace* USERS com a dimensão inicial de 5 M que cresce dinamicamente em pedaços de 1280 K e sem limite de tamanho:

```
ALTER TABLESPACE "USERS" ADD
  DATAFILE 'E:\ORACLE\ORADATA\ORA9I\USERS02.DBF'
  SIZE 5M REUSE AUTOEXTEND
  ON NEXT  1280K MAXSIZE UNLIMITED;
```

Parar o crescimento automático do *datafile*:

```
ALTER DATABASE
  DATAFILE 'D:\ORACLE\ORADATA\ORA920\USERS1.DBF'
  AUTOEXTEND OFF;
```

Diminuir o tamanho de um *datafile*:

```
ALTER DATABASE
  DATAFILE 'E:\ORACLE\ORADATA\ORA9I\USERS05.DBF'
  RESIZE 40M;
```


Exemplos

Mover *datafiles* entre *tablespaces*:

Antes de mais, fazer backup da base de dados.

Partido do princípio que se pretende mover um dos datafiles do *tablespace* DATA1, ver a lista dos *datafile*s que fazem parte desse *tablespace*.

```
SELECT file_name, bytes FROM dba_data_files
 WHERE tablespace_name  = 'DATA1';

 FILE_NAME                                  BYTES
------------------------------              ----------
E:\ORACLE\ORADATA\ORA9I\DATA1.DBF        5242880
E:\ORACLE\ORADATA\ORA9I\DATA2.DBF        5242880
```


Exemplos

Pôr o *tablespace* DATA1 *offline.*

```
ALTER TABLESPACE "DATA1" OFFLINE NORMAL;
```

Copiar o *datafile* pretendido para o novo local, usando os comandos do sistema operativo.
Indicar ao SGBD que o *datafile* está noutro local.

```
ALTER TABLESPACE data1
  RENAME DATAFILE 'E:\ORACLE\ORADATA\ORA9I\DATA2.DBF'
  TO 'C:\ORADATA\DATA2.DBF';
```

Trazer o *tablespace* DATA1 novamente para *online.*

```
ALTER TABLESPACE "DATA1" ONLINE;
```

Fazer novo backup da base de dados.

Remover *datafiles*:

Exemplos

Criar um novo *tablespace*.

```
 CREATE TABLESPACE  "LIXO"
   DATAFILE  'E:\ORACLE\ORADATA\ORA9I\LIXO.DBF'
SIZE 5M
```

Mover o *datafile* para o novo *tablespace*, ver exemplo anterior.

Remover o *tablespace*.

Embora não obrigatório, é preferível pôr o *tablespace offline* antes de o remover.

Exemplos

ALTER TABLESPACE "LIXO" OFFLINE NORMAL;
É possível indicar ao *Oracle* que para além de apagar o *tablespace* deve remover fisicamente os *datafile*s que lhe pertencem com a cláusula INCLUDING CONTENTS AND *DATAFILE*S.

```
DROP TABLESPACE "LIXO" INCLUDING CONTENTS AND DATAFILES;
```

Mover tabelas entre *tablespaces*:

```
ALTER TABLE MEUS_CDS MOVE TABLESPACE DADOS1;
```

Mover índices entre *tablespaces*:

```
ALTER INDEX SCOTT.NOME REBUILD TABLESPACE DADOS1;
```

Consultar a *view* DBA_TABLES de forma a saber que objectos lhe pertencem:

```
SELECT table_name, tablespace_name FROM dba_tables
  WHERE table_name = 'MEUS_CDS';

TABLE_NAME                     TABLESPACE_NAME
----------                     ------------------------------
MEUS_CDS                       DADOS1
```


Exemplos

Consultar a *view* DBA_INDEXES de forma a saber que objectos lhe pertencem:

```
SELECT index_name,tablespace_name FROM dba_indexes
    WHERE index_name='NOME';

INDEX_NAME  TABLESPACE_NAME
----------  ------------------------------
 NOME                DADOS1
```

Renomear um *tablespace*:

```
ALTER TABLESPACE “USERS” RENAME TO “USERS_OLD”;
```


Exemplos

Verificar se diferentes sistemas operativos têm o mesmo *byte order* podemos usar uma *view* da base de dados, V$TRANSPORTABLE _PLATFORM:
SELECT * FROM V$TRANSPORTABLE_PLATFORM;

PLATFORM_ID	PLATFORM_NAME	ENDIAN_FORMAT
1	Solaris[tm] OE (32-bit)	Big
2	Solaris[tm] OE (64-bit)	Big
7	Microsoft Windows IA (32-bit)	Little
10	Linux IA (32-bit)	Little
6	AIX-Based Systems (64-bit)	Big
3	HP-UX (64-bit)	Big
5	HP Tru64 UNIX	Little
4	HP-UX IA (64-bit)	Big
11	Linux IA (64-bit)	Little
15	HP Open VMS	Little
8	Microsoft Windows IA (64-bit)	Little
9	IBM zSeries Based Linux	Big
13	Linux 64-bit for AMD	Little
16	Apple Mac OS	Big
12	Microsoft Windows 64-bit for AMD	Little

Exemplos

Transportar um *tablespace* entre duas bases de dados *Oracle10g* instaladas respectivamente em *MS Windows* e em *Linux*:

Colocar o *tablespace* a transportar em modo só leitura, *read only*.
SQL > ALTER TABLESPACE DADOS5 READ ONLY;

Exportar a metainformação sobre o *tablespace*. A partir de uma janela MS-DOS, executar o comando exp, de notar que o utilizador é o sys com o privilégio de sysdba com a respectiva *password*, fazer:

```
D:\ exp tablespaces=dados5 transport_tablespace=y file=tbs_dados5.dmp
Export: Release 10.1.0.2.0 - Production on Qua Jul 14 11:05:15 2004
Copyright (c) 1982, 2004, Oracle. All rights reserved.
Username: sys as sysdba
Password:
Connected to: Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - Production
With the Partitioning, OLAP and Data Mining options
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
Note: table data (rows) will not be exported
About to export transportable tablespace metadata...
For tablespace DADOS5 ...
. exporting cluster definitions
. exporting table definitions
. . exporting table                          TAB1
. . exporting table                          TAB2
. . exporting table                          TAB3
. . exporting table                          TAB4
. exporting referential integrity constraints
. exporting triggers
. end transportable tablespace metadata export
Export terminated successfully without warnings.
```


Exemplos

Copiar o *tablespace* e o ficheiro de *export* com a metainformação para o novo sistema, neste caso copiar os ficheiros "DADOS5.DBF" e o "tbs_dados5.dmp".

Esta operação pode ser feita usando, por exemplo, o **FTP**; neste caso não esquecer de copiar em modo binário.

Importar o *tablespace* com a metainformação, fazer:

```
$ imp tablespaces=dados5 transport_tablespace=y file=tbs_dados5.dmp
datafiles='DADOS5.DBF'
Import: Release 10.1.0.2.0 - Production on Qua Jul 14 11:58:07 2004
Copyright (c) 1982, 2004, Oracle. All rights reserved.
Username: sys as sysdba
Password:
Connected to: Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - Production
With the Partitioning, OLAP and Data Mining options
Export file created by EXPORT:V10.01.00 via conventional path
About to import transportable tablespace(s) metadata...
import done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
. importing SYS's objects into SYS
. importing SCOTT's objects into SCOTT
. . importing table                    "TAB1"
. . importing table                    "TAB2"
. . importing table                    "TAB3"
. . importing table                    "TAB4"
. importing SYS's objects into SYS
Import terminated successfully without warnings.
```


Exemplos

Finalmente não esquecer de voltar a colocar o *tablespace on line*, para permitir operações de escrita.

```
SQL> ALTER TABLESPACE DADOS5 READ WRITE;
```

Para visualizar as operações de *sort* numa determinada instância, recorrendo à vista **V$SORT_USAGE** em conjunto com a **V$SESSION**, é possível recolher informação sobre que utilizadores estão a executar operações de *sort*:

```
SELECT 'vs.username', vs.sid, vs.serial#, vsu.contents
      FROM v$session vs, v$sort_usage vsu
      WHERE vs.saddr = vsu.session_addr;

USERNAME                SID   SERIAL#     CONTENTS
--------------------      ---   -------    --------
SCOTT                     23       75    PERMANENT
```


Exemplos

Identificar utilizadores a executar operações de *sort*:

```
SELECT      substr(vs.username,1,20) "Utilizador BD",
            substr(vs.osuser,1,20)   "Utilizador SO",
            substr(vsn.name,1,20)    "Tipo de Sort",
            vss.value
FROM        v$session vs,
            v$sesstat vss,
            v$statname vsn
WHERE       (vss.statistic#=vsn.statistic#)
  AND     (vs.sid = vss.sid)
  AND     (vsn.name like '%sort%')
ORDER BY 2,3;
```

Utilizador BD	Utilizador so	Tipo de Sort	VALUE
SYSTEM	tomanix	sorts (disk)	0
SYS	tomanix	sorts (disk)	0
SYSTEM	tomanix	sorts (disk)	0
SYSTEM	tomanix	sorts (memory)	17
SYSTEM	tomanix	sorts (memory)	67
SYS	tomanix	sorts (memory)	18
SYSTEM	tomanix	sorts (rows)	107
SYSTEM	tomanix	sorts (rows)	1241
SYS	tomanix	sorts (rows)	64

Exemplos

Adicionar um *datafile* temporário a um *tablespace* temporário:

```
ALTER TABLESPACE "TEMP"
 ADD TEMPFILE 'D:\ORACLE\ORADATA\ORA92\TEMP02.DBFb'
 SIZE  5M;
```

Utilizar vários *temporary tablespaces*:

```
CREATE TEMPORARY TABLESPACE TBSTEMP1
        TEMPFILE 'e:\oracle\oradata\ora10g\tbstemp1.dbf'
        SIZE 5M
        TABLESPACE GROUP TBSTEMPGRP1;

CREATE TEMPORARY TABLESPACE TBSTEMP2
        TEMPFILE 'e:\oracle\oradata\ora10g\tbstemp2.dbf'
        SIZE 15M
        TABLESPACE GROUP TBSTEMPGRP1;
```


Exemplos

Alterar o *tablespace* temporário por omissão na base de dados:

```
ALTER DATABASE ora10g DEFAULT TEMPORARY TABLESPACE TBSTEMPGRP1;
```

Consultar a lista de grupos de *tablespaces*, temporários ou não:

```
SQL> SELECT * FROM DBA_TABLESPACE_GROUPS;
GROUP_NAME              TABLESPACE_NAME
TBSTEMPGRP1             TBSTEMP1
TBSTEMPGRP1             TBSTEMP2
```

Criar um *undo tablespace*:

```
CREATE UNDO TABLESPACE "NOVOUNDOTBS" DATAFILE
    '/home/ora10g/oradata/orcl/novoundotbs.dbf' SIZE 5M;
```

Controlar o tempo que a instância retém os dados no *undo tablespace* através do parâmetro de arranque *UNDO_RETENTION*:

```
ALTER SYSTEM SET UNDO_RETENTION = 3600;
```


Exemplos

Monitorar e recolher estatísticas do *undo tablespace*, podemos usar a vista **V$UNDOSTAT**:

```
SQL> SELECT TO_CHAR(BEGIN_TIME, 'DD/MM/YYYY HH24:MI:SS')
     BEGIN_TIME,
     TO_CHAR (END_TIME, 'DD/MM/YYYY HH24:MI:SS') END_TIME,
     UNDOTSN,UNDOBLKS, TXNCOUNT, MAXCONCURRENCY AS "MAXCON"
     FROM V$UNDOSTAT;

BEGIN_TIM      END_TIME    UNDOTSN  UNDO  BLKS  TXN    COUNT  MAXCON
-----------------  ------------  -------  --------  -----  -----  -----------  --------
     ----------
15/07/2004      11:19:08      15/07/2004 11:28:36   1     3          46      1
15/07/2004      11:09:08      15/07/2004 11:19:08   1     4          71      1
15/07/2004      10:59:08      15/07/2004 11:09:08   1    38         109      1
15/07/2004      10:49:08      15/07/2004 10:59:08   1     6          52      2
15/07/2004      10:39:08      15/07/2004 10:49:08   1   253         386      3
15/07/2004      10:29:08      15/07/2004 10:39:08   1   566        1095      3
6 rows selected.
```


Exemplos

Criar um novo segmento de *rollback*:

```
CREATE ROLLBACK SEGMENT "RBS01"
    TABLESPACE "RBS01";

ALTER ROLLBACK SEGMENT "RBS01" ONLINE;
```

Consulta de informação sobre os vários segmentos de *rollback*:

```
SELECT SEGMENT_NAME, TABLESPACE_NAME, BYTES
FROM DBA_SEGMENTS
WHERE SEGMENT_TYPE = 'ROLLBACK';
```


Exemplos

Verificar a lista de utilizadores na base de dados e o seu estado:

```
SQL> SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS;
USERNAME                       ACCOUNT_STATUS
------------------------------ --------------------
SYS                            OPEN
SYSTEM                         OPEN
OUTLN                          EXPIRED & LOCKED
DBSNMP                         OPEN
SYSMAN                         OPEN
MGMT_VIEW                      OPEN
MDSYS                          EXPIRED & LOCKED
ORDSYS                         EXPIRED & LOCKED
EXFSYS                         EXPIRED & LOCKED
DMSYS                          EXPIRED & LOCKED
WMSYS                          EXPIRED & LOCKED
WKSYS                          EXPIRED & LOCKED
WK_TEST                        EXPIRED & LOCKED
CTXSYS                         EXPIRED & LOCKED
ANONYMOUS                      EXPIRED & LOCKED
XDB                            EXPIRED & LOCKED
WKPROXY                        EXPIRED & LOCKED
ORDPLUGINS                     EXPIRED & LOCKED
SI_INFORMTN_SCHEMA             EXPIRED & LOCKED
OLAPSYS                        EXPIRED & LOCKED
SCOTT                          OPEN
BI                             EXPIRED & LOCKED
PM                             EXPIRED & LOCKED
MDDATA                         EXPIRED & LOCKED
IX                             EXPIRED & LOCKED
SH                             EXPIRED & LOCKED
DIP                            EXPIRED & LOCKED
OE                             EXPIRED & LOCKED
HR                             EXPIRED & LOCKED
29 rows selected.
```


Exemplos

Procedimento que cria uma vista com o nome de **DBA_USER_PRIVS** que lista todos os utilizadores e respectivos privilégios:

```
SET ECHO off
REM NAME:    TFSPRVVW.SQL
REM USAGE:"@path/tfsprvvw"
REM ----------------------------------------------
REM REQUIREMENTS:
REM    Should be run as SYS
REM ----------------------------------------------
REM AUTHOR:
REM    Anonymous
REM    Copyright 1995, Oracle Corporation
REM ----------------------------------------------
REM PURPOSE:
REM    The following script will generate a view
       DBA_USER_PRIVS
REM    which may be queried to obtain information
       about which
REM    privileges are available to a given user, or which
       users
REM    enjoy a particular privilege.
REM ----------------------------------------------
REM EXAMPLE:
REM    USERNAME       ROLENAME          PRIVILEGE
REM    --------       ----------------- ---------
REM    SCOTT          CONNECT           ALTER SESSION
REM    SCOTT          CONNECT           CREATE CLUSTER
REM    SCOTT          CONNECT           CREATE DATABASE LINK
REM    SCOTT          CONNECT           CREATE TABLE
REM    SCOTT          CONNECT           CREATE VIEW
REM    SCOTT          DBA               ALTER ANY CLUSTER
REM    SCOTT          DBA               ALTER ANY INDEX
REM    SCOTT          DBA               ALTER ANY PROCEDURE
REM    SCOTT          DBA               ALTER ANY ROLE
REM    SCOTT          DBA               ALTER ANY SEQUENCE
REM    SCOTT          DBA               ALTER PROFILE
REM    SCOTT          DBA               UPDATE ANY TABLE
REM    SCOTT          RESOURCE          CREATE CLUSTER
REM    SCOTT          RESOURCE          CREATE PROCEDURE
REM    SCOTT          RESOURCE          CREATE SEQUENCE
REM    SCOTT          RESOURCE          CREATE TABLE
REM    SCOTT          RESOURCE          CREATE TRIGGER
REM    SCOTT                            UNLIMITED TABLESPACE
REM
REM ----------------------------------------------
REM DISCLAIMER:
REM    This script is provided for educational purposes only. It
REM    is NOT supported by Oracle World Wide Technical Support.
REM    The script has been tested and appears to work as
REM    intended. You should always run new scripts on a test
REM    instance initially.
REM ----------------------------------------------
REM Main text of script follows:
```


Exemplos

```
CREATE OR REPLACE VIEW DBA_USER_PRIVS (USERNAME, ROLENAME,
    PRIVILEGE) AS
  SELECT DECODE(SA1.GRANTEE#, 1, 'PUBLIC', U1.NAME),
    SUBSTR(U2.NAME,1,20),
   SUBSTR(SPM.NAME,1,27)
FROM SYS.SYSAUTH$ SA1, SYS.SYSAUTH$ SA2, SYS.USER$ U1,
    SYS.USER$ U2, SYS.SYSTEM_PRIVILEGE_MAP SPM
WHERE SA1.GRANTEE# = U1.USER#
   AND SA1.PRIVILEGE# = U2.USER#
   AND U2.USER# = SA2.GRANTEE#
   AND SA2.PRIVILEGE# = SPM.PRIVILEGE
UNION
  SELECT U.NAME, NULL, SUBSTR(SPM.NAME,1,27)
FROM SYS.SYSTEM_PRIVILEGE_MAP SPM, SYS.SYSAUTH$ SA,
  SYS.USER$ U
WHERE SA.GRANTEE#=U.USER#
  AND SA.PRIVILEGE#=SPM.PRIVILEGE
```


Exemplos

Criação de utilizadores:

```
CREATE USER "RUI"
  PROFILE "CONTABILIDADE"
  IDENTIFIED BY "password" PASSWORD EXPIRE
  DEFAULT TABLESPACE "USERS"
  TEMPORARY TABLESPACE "TEMP"
  QUOTA 5 M ON "DADOS2"
  QUOTA 1 M ON "INDICES"
  QUOTA 1 M ON "USERS"
  ACCOUNT UNLOCK;
GRANT "CONNECT" TO "RUI";
```

Alterar a quota do utilizador:

```
ALTER USER "RUI" QUOTA 50 M ON "DADOS2";
```

Alterar a *password* do utilizador:

```
ALTER USER "RUI" IDENTIFIED BY "xpto";
```

Remover utilizadores:

```
DROP USER "RUI" CASCADE;
```

Bloquear o acesso a uma conta:

```
ALTER USER "RUI" ACCOUNT LOCK;
```

Consultar a informação sobre todos os utilizadores da base de dados:

```
SELECT * FROM DBA_USERS;
```

Limitar utilização recursos da base de dados:

```
ALTER SYSTEM SET RESOURCE_LIMIT = TRUE;
```


Exemplos

Criação de perfis:

```
CREATE PROFILE "CONTABILIDADE" LIMIT
      CPU_PER_SESSION 3600
      CPU_PER_CALL DEFAULT
      CONNECT_TIME UNLIMITED
      IDLE_TIME 15
      SESSIONS_PER_USER 1;
```

Alteração de perfis:

```
ALTER PROFILE "CONTABILIDADE" LIMIT
  LOGICAL_READS_PER_SESSION 1000;
```

Remover um *profile* existente:

```
DROP PROFILE "CONTABILIDADE" CASCADE;
```

Atribuir um perfil a um utilizador:

```
ALTER USER "MIGUEL"  PROFILE "CONTABILIDADE";
```


Exemplos

Visualizar informação sobre perfis e sua utilização:

```
SELECT * FROM DBA_PROFILES;
PROFILE              RESOURCE_NAME                  RESOURCE     LIMIT
---------------      ---------------                --------     ------
DEFAULT              COMPOSITE_LIMIT                KERNEL       UNLIMITED
CONTABILIDADE        COMPOSITE_LIMIT                KERNEL       DEFAULT
DEFAULT              FAILED_LOGIN_ATTEMPTS          PASSWORD     UNLIMITED
CONTABILIDADE        FAILED_LOGIN_ATTEMPTS          PASSWORD     DEFAULT
DEFAULT              SESSIONS_PER_USER              KERNEL       UNLIMITED
CONTABILIDADE        SESSIONS_PER_USER               KERNEL       1
DEFAULT              PASSWORD_LIFE_TIME             PASSWORD     UNLIMITED
CONTABILIDADE        PASSWORD_LIFE_TIME             PASSWORD     DEFAULT
DEFAULT              CPU_PER_SESSION                KERNEL       UNLIMITED
.....
```

Impor um limite de 60 dias antes que a mesma *password* possa ser reutilizada:

```
ALTER PROFILE "CONTABILIDADE"
  LIMIT PASSWORD_REUSE_MAX 60;
```


Exemplos

Procedimento para impor complexidade às palavras-chave:

```
Rem
Rem $Header: utlpwdmg.sql 31-aug-2000.11:00:47 nireland Exp $
Rem
Rem utlpwdmg.sql
Rem
Rem  Copyright (c) Oracle Corporation 1996, 2000. All Rights
Rem
Rem    NAME
Rem      utlpwdmg.sql - script for Default Password Resource
Rem
Rem    DESCRIPTION
Rem    This is a script for enabling the password management
Rem    features by setting the default password resource limits.
Rem
Rem    NOTES
Rem    This file contains a function for minimum checking of
Rem    password complexity.  This is more of a sample function
Rem    that the customer    can use to develop the function for
Rem    actual complexity checks that the customer wants to make Rem   on the new password.
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    nireland    08/31/00 - Improve check for
Rem                          username=password. #1390553
Rem    nireland    06/28/00 - Fix null old password test.
Rem                                                          #1341892
Rem    asurpur     04/17/97 - Fix for bug479763
Rem    asurpur     12/12/96 - Changing the name of
Rem                                                          password_verify_function
Rem    asurpur     05/30/96 - New script for default password
Rem                                                          management
Rem    asurpur     05/30/96 - Created
Rem
```


Exemplos

```
-- This script sets the default password resource parameters
-- This script needs to be run to enable the password features.
-- However the default resource parameters can be changed based
-- on the need.
-- A default password complexity function is also provided.
-- This function makes the minimum complexity checks like
-- the minimum length of the password, password not same as the
-- username, etc. The user may enhance this function according --- to the need.
-- This function must be created in SYS schema.
-- connect sys/<password> as sysdba before running the script
CREATE OR REPLACE FUNCTION verify_function
(username varchar2,
  password varchar2,
  old_password varchar2)
  RETURN boolean IS
   n boolean;
   m integer;
   differ integer;
   isdigit boolean;
   ischar  boolean;
   ispunct boolean;
   digitarray varchar2(20);
   punctarray varchar2(25);
   chararray varchar2(52);
```

Exemplos

```
BEGIN
   digitarray:= '0123456789';
   chararray:=
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
   punctarray:='!"#$%&()``*+,-/:;<=>?_';
   --
   -- Esta condição verifica se a password é igual ao username,
   -- uma das regras básicas para impedir a utilização de
   -- passwords óbvias
   --
   -- Check if the password is same as the username
   IF NLS_LOWER(password)  = NLS_LOWER(username)  THEN
     raise_application_error(-20001,  'Password same as or simlar
to user');
   END IF;
   --
   -- Verificação do tamanho mínimo da password, para uma
   -- política
   -- mais rigorosa podemos impor um tamanho de 8 caracteres
   -- substituindo o 4 pelo 8
   --
   -- Check for the minimum length of the password
   IF length(password)  < 4 THEN
     raise_application_error(-20002,  'Password length less than 4');
   END IF;
```


Exemplos

```
--
  -- Verifica se uma password é demasiado simples; é possível
  -- manter um
  -- dicionário de palavras consideradas óbvias, no nosso caso
  -- poderíamos
  -- acrescentar 'benfica', 'sporting', 'porto', 'boavista',
  -- etc
  --
  -- Check if the password is too simple. A dictionary of words
  -- may be
  -- maintained and a check may be made so as not to allow the
  -- words
  -- that are too simple for the password.
  IF NLS_LOWER(password) IN ('welcome', 'database', 'account',
  'user',
    'password', 'Oracle', 'computer', 'abcd') THEN
    raise_application_error(-20002, 'Password too simple');
  END IF;
  --
  -- Verifica se a password contém pelo menos um algarismo e um
  -- carácter de pontuação de forma a tornar mais difícil
  -- adivinhar
  --
  -- Check if the password contains at least one letter, one
```


Exemplos

```
-- digit and one
  -- punctuation mark.
  -- 1. Check for the digit
     isdigit:=FALSE;
     m := length(password);
     FOR i IN 1..10 LOOP
        FOR j IN 1..m LOOP
           IF substr(password,j,1) = substr(digitarray,i,1) THEN
              isdigit:=TRUE;
               GOTO findchar;
           END IF;
        END LOOP;
     END LOOP;
     IF isdigit = FALSE THEN
        raise_application_error(-20003, 'Password should \ contain  at least one digit,
     one character and one punctuation');
     END IF;
     -- 2. Check for the character
     <<findchar>>
     ischar:=FALSE;
```


Exemplos

```
FOR i IN 1..length(chararray) LOOP
        FOR j IN 1..m LOOP
          IF substr(password,j,1) = substr(chararray,i,1) THEN
            ischar:=TRUE;
             GOTO findpunct;
          END IF;
        END LOOP;
      END LOOP;
      IF ischar = FALSE THEN
        raise_application_error(-20003, 'Password should contain
\ at least one digit, one character and one punctuation');
      END IF;
      -- 3. Check for the punctuation
      <<findpunct>>
      ispunct:=FALSE;
      FOR i IN 1..length(punctarray) LOOP
        FOR j IN 1..m LOOP
          IF substr(password,j,1) = substr(punctarray,i,1) THEN
            ispunct:=TRUE;
             GOTO endsearch;
          END IF;
        END LOOP;
      END LOOP;
      IF ispunct = FALSE THEN
        raise_application_error(-20003, 'Password should contain at least one \
              digit, one character and one punctuation');
      END IF;
      <<endsearch>>
```


Exemplos

-- Verifica se a *password* é diferente da anterior em pelo menos 3 caracteres

```
  --
  -- Check if the password differs from the previous password by at least 3 letters
  IF old_password IS NOT NULL THEN
    differ := length(old_password) - length(password);
    IF abs(differ) < 3 THEN
      IF length(password) < length(old_password) THEN
        m := length(password);
      ELSE
        m := length(old_password);
      END IF;
      differ := abs(differ);
      FOR i IN 1..m LOOP
      IF substr(password,i,1) != substr(old_password,i,1)
      THEN
        differ := differ + 1;
      END IF;
    END LOOP;
    IF differ < 3 THEN
      raise_application_error(-20004, 'Password should \
      differ by at least 3 characters');
    END IF;
  END IF;
 END IF;
 -- Everything is fine; return TRUE ;
 RETURN(TRUE);
END;
/
```


Exemplos

```
-- This script alters the default parameters for Password
-- Management
-- This means that all the users on the system have Password
-- Management
-- enabled and set to the following values unless another
-- profile is
-- created with parameter values set to different value or
-- UNLIMITED is created and assigned to the user.
 ALTER PROFILE DEFAULT LIMIT
 PASSWORD_LIFE_TIME 60
 PASSWORD_GRACE_TIME 10
 PASSWORD_REUSE_TIME 1800
 PASSWORD_REUSE_MAX UNLIMITED
 FAILED_LOGIN_ATTEMPTS 3
 PASSWORD_LOCK_TIME 1/1440
 PASSWORD_VERIFY_FUNCTION verify_function;
```

Alterar a própria *password* usando o comando **PASSWORD**:

```
SQL> password
 A alterar a senha para MIGUEL
 Senha antiga:
  Senha nova:
  Reintroduza a senha nova:
  Senha foi alterada
  SQL>
```


Exemplos

Alterar a *password* de outro utilizador usando o comando PASSWORD:

```
SQL> password miguel
  A alterar a senha para MIGUEL
  Senha nova:
  Reintroduza a senha nova:
  Senha foi alterada
  SQL>
```

Criação de um utilizador que será validado pelo sistema operativo:

```
SQL> CREATE USER "OPS$RUI"  PROFILE "DEFAULT"
     IDENTIFIED EXTERNALLY
     DEFAULT TABLESPACE "USERS"
     ACCOUNT UNLOCK;

SQL> GRANT "CONNECT" TO "OPS$RUI";
```

Criação de um ficheiro de *passwords*:

Criar o ficheiro de *passwords* usando o utilitário **ORAPWD**, numa janela *Command Prompt* no *Windows* ou no utilizador *Oracle* no *Linux*:

```
$ orapwd file=ficheiro password=plim entries=10
```

Verificar o parâmetro de inicialização da instância:
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE

Exemplos

Entrar na base de dados com privilégios AS SYSDBA usando a validação do sistema operativo:
$ sqlplus /nolog
SQL > connect / as sysdba
Entrar na base de dados com privilégios AS SYSDBA usando o ficheiro de *passwords*:
$ sqlplus /nolog
SQL > connect user/password as sysdba
Dar a outros utilizadores da base de dados a possibilidade de ligação como SYSDBA, usando o ficheiro de *passwords*:
GRANT SYSDBA TO RUI;
Revogar o privilégio de ligação AS SYSDBA:
REVOKE SYSDBA FROM RUI;
Consultar a lista de utilizadores com privilégio SYSDBA:
SQL> SELECT * FROM V$PWFILE_USERS;

USERNAME	SYSDB	SYSOP
SYS	TRUE	TRUE

Entrar na base de dados com privilégios AS SYSOPER usando a validação do sistema operativo:
$ sqlplus /nolog
SQL > connect / as sysoper
Entrar na base de dados com privilégios AS SYSOPER usando o ficheiro de *passwords*:
$ sqlplus /nolog
SQL > connect user/password as sysoper
Dar a outros utilizadores da base de dados a possibilidade de ligação como SYSOPER, usando o ficheiro de *passwords*:
GRANT SYSOPER TO RUI;
Revogar o privilégio de ligação AS SYSOPER:
REVOKE SYSOPER FROM RUI;

Exemplos

Consultar a lista de utilizadores com privilégio SYSOPER:

```
SQL> SELECT * FROM V$PWFILE_USERS;
USERNAME                       SYSDB      SYSOP
------------------------------ ---------- --------
SYS                            TRUE       TRUE
OPERADOR                       FALSE      TRUE
```

Entrar na base de dados com privilégios normais de utilizador usando a validação do sistema operativo:

```
$ sqlplus /nolog
SQL > connect /
$ sqlplus /nolog
SQL > connect rui/"password_do_rui"
$ sqlplus rui/"password_do_rui"@ora920
```

Atribuir privilégios de sistema:

```
GRANT ALTER ANY TABLE TO "MIGUEL";
```

Revogar privilégios de sistema:

```
REVOKE ALTER ANY TABLE FROM "MIGUEL";
```

Atribuir um privilégio sobre um determinado objecto a um utilizador:

```
GRANT SELECT  ON  "SCOTT"."EMP" TO "MIGUEL";
```

Dar permissão de cedência de permissões:

```
GRANT UPDATE (SAL) ON "SCOTT"."EMP" TO "MIGUEL"
    WITH GRANT OPTION;
```


Exemplos

Criação um *role*:

```
CREATE ROLE "RECURSOS_HUMANOS";
```

Atribuir privilégios ao *role* sobre um ou mais objectos:

```
GRANT SELECT, INSERT, UPDATE ON "HR"."COUNTRIES"
    TO "RECURSOS_HUMANOS";
GRANT SELECT, INSERT, UPDATE ON "HR"."DEPARTMENTS"
    TO "RECURSOS_HUMANOS";
GRANT SELECT, INSERT, UPDATE ON "HR"."EMPLOYES"
    TO "RECURSOS_HUMANOS";
GRANT SELECT, INSERT, UPDATE ON "HR"."JOBS"
    TO "RECURSOS_HUMANOS";
```

Atribuir privilégios de sistema a *roles*:

```
GRANT CREATE SESSION TO "RECURSOS_HUMANOS";
```

Atribuir outros *roles* a *roles*:

```
GRANT "RESOURCE" TO "RECURSOS_HUMANOS";
```

Atribuir *passwords* a *roles*:

```
CREATE ROLE "GESTOR_DE_RH" IDENTIFIED BY password;
```

Alterar os *roles* activos numa sessão:

```
SET ROLE GESTOR_DE_RH IDENTIFIED BY password;
```


Exemplos

Consultar quais os *roles* que tem disponíveis na sessão:

```
SQL> SELECT * FROM SESSION_ROLES;
ROLE
------------------------------
GESTOR_DE_RH
```

Consultar a lista de privilégios na sessão:

```
SQL> SELECT * FROM SESSION_PRIVS;
 PRIVILEGE
 ----------------------------------------
 CREATE SESSION
 ALTER SESSION
 CREATE TABLE
 ALTER ANY TABLE
 CREATE CLUSTER
 CREATE SYNONYM
 CREATE VIEW
 CREATE SEQUENCE
 CREATE DATABASE LINK
```

Alterar a lista de *roles* por omissão para um determinado utilizador:

```
ALTER USER MIGUEL
      DEFAULT ROLE CONNECT,GESTOR_DE_RH;
```

Atribuir *roles* a utilizadores juntamente com o privilégio de poderem, por sua vez, atribuí-los a outros utilizadores:

```
GRANT "GESTOR_DE_RH" TO "MIGUEL"
      WITH ADMIN OPTION;
```

Remover *roles*:

```
SQL> DROP ROLE GESTOR_DE_RH;
```


Exemplos

Consultar a lista de utilizadores com privilégios na tabela "EMPLOYEES":

```
SQL> SELECT GRANTEE, OWNER, PRIVILEGE
        FROM DBA_TAB_PRIVS
        WHERE TABLE_NAME = 'EMPLOYEES';
```

GRANTEE	OWNER	PRIVILEGE
OE	HR	SELECT
OE	HR	REFERENCES
RECURSOS_HUMANOS	HR	INSERT
RECURSOS_HUMANOS	HR	SELECT
RECURSOS_HUMANOS	HR	UPDATE
RECURSOS_HUMANOS	HR	ALTER
RECURSOS_HUMANOS	HR	DELETE

Consultar a lista de privilégios de sistema do *role* "DBA":

```
SELECT * FROM ROLE_SYS_PRIVS WHERE ROLE = 'DBA';
```

ROLE	PRIVILEGE	ADM
DBA	AUDIT ANY	YES
DBA	DROP USER	YES
DBA	RESUMABLE	YES
DBA	ALTER USER	YES
DBA	ANALYZE ANY	YES
DBA	BECOME USER	YES
DBA	CREATE ROLE	YES
DBA	CREATE RULE	YES
DBA	CREATE TYPE	YES
DBA	CREATE USER	YES

Exemplos

Activar o *audit* na base de dados:

```
ALTER SYSTEM SET AUDIT_TRAIL = DB SCOPE=SPFILE;
```

Activar a auditoria às operações de *login*:

```
AUDIT SESSION;
```

Consultar os registos de auditoria:

```
SELECT USERNAME, ACTION_NAME, RETURNCODE, TIMESTAMP
      FROM DBA_AUDIT_SESSION  WHERE USERNAME = 'SCOTT';
USERNAME ACTION_NAME          RETURNCODE    TIMESTAMP
---------      --------------        -----------  ---------------
SCOTT       LOGON               1017        02.09.25
~
SELECT USERNAME, ACTION_NAME, RETURNCODE, TIMESTAMP
    FROM DBA_AUDIT_SESSION  WHERE USERNAME = 'PEDRO';
USERNAME ACTION_NAME        RETURNCODE       TIMESTAMP
----------  --------------                          ------------     ---------------
PEDRO       LOGOFF                    0             02.09.25
```

Registar só as tentativas com êxito ou só as tentativas falhadas:

```
SQL> AUDIT SESSION WHENEVER SUCCESSFUL;
SQL> AUDIT SESSION WHENEVER NOT SUCCESSFUL;
```

Registar todas as operações que afectem tabelas:

```
AUDIT TABLE;
```


Exemplos

Consultar a lista de códigos e seus significados das várias operações sujeitas a auditoria:

```
SQL> SELECT * FROM AUDIT_ACTIONS;
ACTION     NAME
----------     ----------------------------
0                    UNKNOWN
1                    CREATE TABLE
2                    INSERT
3                    SELECT
4                    CREATE CLUSTER
5                    ALTER CLUSTER
.....
```

Activar o registo de ocorrências especificando o utilizador e a operação a registar:

```
AUDIT DELETE TABLE BY MIGUEL;
```

Desactivar uma determinada auditoria:

```
NOAUDIT TABLE;
```


Exemplos

Registo de acções sobre objectos da base de dados (*object audits*):

```
AUDIT DELETE ON PEDRO.CLIENTE BY ACCESS;
```

Consultar a informação sobre o resultado do *audit*:

```
SELECT USERNAME, TIMESTAMP, OBJ_NAME,
       ACTION_NAME, RETURNCODE
       FROM   DBA_AUDIT_OBJECT WHERE OBJ_NAME = 'CLIENTE';
USERNAME   TIMESTAM  OBJ_NAME    ACTION_NAME      RETURNCODE
---------  --------  ----------  -----------      ----------- SYSTEM     02.09.28     CLIENTE
DELETE                 0
SYSTEM      02.09.28     CLIENTE       DELETE                 0
SCOTT       02.09.28     CLIENTE       DELETE              2004
```

Registar somente as tentativas sem sucesso de acessos ou alterações aos dados da tabela:

```
AUDIT SELECT, INSERT, UPDATE, DELETE ON PEDRO.CLIENTE
  BY ACCESS
  WHENEVER NOT SUCCESSFUL
```


INSERT INTO ... SELECT

```
create table klinhas
(ki number(3),
ano varchar2(4),
mes varchar2(2),
cod_servico_unidade  varchar2(1),
cod_servico_unidade_prev  varchar2(1),
cod_unidade_inter number(3),
cod_unidade_inter_prev number(3),
cod_especialidade number(5),
cod_especialidade_prev number(5),
tint number(8),
tnom number(8),
tdenom number(8)
);
```

```
insert into klinhas(ki,ano,mes,cod_servico_unidade,
cod_servico_unidade_prev,cod_unidade_inter,
cod_unidade_inter_prev,
cod_especialidade,cod_especialidade_prev,tint)
select
1,to_char(dta_saida,'yyyy'),to_char(dta_saida,'mm'),cod_se
rvico_unidade,
cod_servico_unidade_prev,cod_unidade_inter,
cod_unidade_inter_prev,cod_especialidade,
cod_especialidade_prev, count(distinct int_episodio)
from transferencias
where to_char(dta_saida,'yyyy') = '2009'
group by 1,to_char(dta_saida,'yyyy'),
to_char(dta_saida,'mm'),
cod_servico_unidade,cod_servico_unidade_prev,
cod_unidade_inter,cod_unidade_inter_prev,
cod_especialidade,cod_especialidade_prev;
```


CREATE PROCEDURE

```
CREATE OR REPLACE PROCEDURE CRIA_KI (iniki_ number, finiki_ number) is
Begin
declare ki_ number(3); ano_ varchar2(4);
mes_ varchar2(2); cod_servico_unidade_ varchar2(1);
cod_servico_unidade_prev_ varchar2(1); cod_unidade_inter_ number(5);
 cod_unidade_inter_prev_ number(5); cod_especialidade_ number(5);
cod_especialidade_prev_ number(5);
tnom_ number(8); tdenom_ number(8);
exp1_ varchar2(200); exp2_ varchar2(200); exp3_ varchar2(200);
tipo_ number(2); interid_ number(8); interid1_ number(8); interid2_ number(8);
Begin
....
End;
End;
```


Begin ... End

```
Begin
ki_ := iniki_ - 1;
while (ki_ < finiki_) loop
ki_ := ki_ + 1;
if (kpi_ = 1) then
  exp1_ := '.....';   tipo_ := 1;
end if;
....
if (kpi_ = 12) then
 exp1_ := ' ...';
 tipo_ := 5;
 end if;
 begin ... end;
end;
```


Cursores

begin

declare cursor c1 is select ano, mes, cod_servico_unidade, cod_servico_unidade_prev, nvl(cod_unidade_inter,0), nvl(cod_unidade_inter_prev,0), cod_especialidade, cod_especialidade_prev from kilinhas where ki = ki_; begin

open c1;

fetch c1 into ano_,mes_,cod_servico_unidade_,cod_servico_unidade_prev_, cod_unidade_inter_, cod_unidade_inter_prev_, cod_especialidade_, cod_especialidade_prev_;

while c1%FOUND **loop**

SELECT ...

UPDATE ...

COMMIT

fetch c1 into ano_, mes_, cod_servico_unidade_, cod_servico_unidade_prev_, cod_unidade_inter_, cod_unidade_inter_prev_, cod_especialidade_, cod_especialidade_prev_;

end loop;

close c1;

end;

RAC e DataGuard