
Banco de Dados Oracle 10g: Fundamentos de SQL I

Volume II • Guia do Aluno

D17108BP10

Produção 1.0

Junho 2004

D39573

ORACLE®

Autor

Nancy Greenberg

Revisores e Colaboradores Técnicos

Wayne Abbott
Christian Bauwens
Perry Benson
Brian Boxx
Zarko Cesljas
Dairy Chan
Laszlo Czinkoczki
Marjolein Dekkers
Matthew Gregory
Stefan Grenstad
Joel Goodman
Rosita Hanoman
Sushma Jagannath
Angelika Krupp
Christopher Lawless
Marcelo Manzano
Isabelle Marchand
Malika Marghadi
Valli Pataballa
Elspeth Payne
Ligia Jasmin Robayo
Bryan Roberts
Helen Robertson
Lata Shivaprasad
John Soltani
Priya Vennapusa
Ken Woolfe

Editor

Nita K. Brozowski

Copyright © 2004, Oracle. Todos os direitos reservados.

Esta documentação contém informações de propriedade da Oracle Corporation. Ela é fornecida sob um contrato de licença que contém restrições quanto ao uso e à divulgação, além de ser protegida pela legislação de direitos autorais. É proibida a engenharia reversa do software. Se esta documentação for distribuída a uma Agência Governamental subordinada ao Departamento de Defesa dos EUA, ela terá direitos restritos e o seguinte aviso deverá ser aplicado:

Aviso de Direitos Restritos

A utilização, a duplicação ou a divulgação pelo governo estará sujeita às restrições impostas a um software comercial e deverão ser aplicadas as leis federais relativas a um software com direitos restritos, como definidos no subparágrafo (c)(1)(ii) de DFARS 252.227-7013, Rights in Technical Data and Computer Software (Direitos sobre Dados Técnicos e Software de Computadores) (outubro de 1988).

Este material, ou parte dele, não poderá ser copiado de qualquer forma ou por qualquer meio sem a prévia permissão expressa por escrito da Oracle Corporation. Qualquer outra cópia constituirá uma violação da legislação de direitos autorais e poderá resultar em indenizações civis e/ou criminais.

Se esta documentação for distribuída a uma Agência Governamental que não pertença ao Departamento de Defesa dos EUA, ela terá "direitos restritos", conforme definido no FAR 52.227-14, Rights in Data-General (Direitos Gerais sobre Dados), incluindo Alternate III (Alternativa III) (junho de 1987).

As informações contidas neste documento estão sujeitas a alterações sem aviso prévio. Se você encontrar algum problema na documentação, envie ao departamento Worldwide Education Services uma descrição de tal problema por escrito. Oracle Corporation, 500 Oracle Parkway, Box SB-6, Redwood Shores, CA 94065 - USA. Distribuidor no Brasil: Oracle do Brasil Sistemas Ltda. Rua José Guerra, 127, São Paulo, SP - 04719-030 - Brasil - CNPJ: 59.456.277/0001-76. A Oracle Corporation não garante que esta documentação esteja isenta de erros.

Oracle e todas as referências a produtos da Oracle são marcas comerciais ou registradas da Oracle Corporation.

Todos os outros nomes de empresas e produtos são usados com o único propósito de identificação e podem ser marcas comerciais dos respectivos proprietários.

Conteúdo

Prefácio

Introdução

Objetivos da Lição	I-2
Metas do Curso	I-3
Oracle10g	I-4
Banco de Dados Oracle 10g	I-6
Oracle Application Server 10g	I-7
Oracle Enterprise Manager 10g Grid Control	I-8
Sistemas de Gerenciamento de Banco de Dados Relacional e Banco de Dados Relacional de Objeto	I-9
Plataforma Oracle para a Internet	I-10
Ciclo de Vida de Desenvolvimento do Sistema	I-11
Armazenamento de Dados em Diferentes Mídias	I-13
Conceito de Banco de Dados Relacional	I-14
Definição de um Banco de Dados Relacional	I-15
Modelos de Dados	I-16
Modelo de Relacionamento entre Entidades	I-17
Convenções de Modelagem de Relacionamento entre Entidades	I-19
Relacionando Várias Tabelas	I-21
Terminologia do Banco de Dados Relacional	I-23
Propriedades do Banco de Dados Relacional	I-25
Comunicando-se com um RDBMS por Meio de SQL	I-26
Sistema de Gerenciamento de Banco de Dados Relacional da Oracle	I-27
Instruções SQL	I-28
Tabelas Usadas no Curso	I-29
Sumário	I-30

1 Recuperando Dados com a Instrução SQL **SELECT**

Objetivos	1-2
Recursos de Instruções SQL SELECT	1-3
Instrução SELECT Básica	1-4
Selecionando Todas as Colunas	1-5
Selecionando Colunas Específicas	1-6
Criando Instruções SQL	1-7
Defaults de Cabeçalhos de Colunas	1-8
Expressões Aritméticas	1-9
Usando Operadores Aritméticos	1-10
Precedência de Operadores	1-11
Definindo um Valor Nulo	1-12
Valores Nulos em Expressões Aritméticas	1-13
Definindo um Apelido de Coluna	1-14
Usando Apelidos de Colunas	1-15
Operador de Concatenação	1-16
Strings de Caracteres Literais	1-17
Usando Strings de Caracteres Literais	1-18

Operador de Aspas (q) Alternativo	1-19
Linhas Duplicadas	1-20
Interação entre SQL e <i>iSQL*Plus</i>	1-21
Instruções SQL e Comandos <i>iSQL*Plus</i>	1-22
Visão Geral do <i>iSQL*Plus</i>	1-23
Efetuando Login no <i>iSQL*Plus</i>	1-24
Ambiente <i>iSQL*Plus</i>	1-25
Exibindo a Estrutura de Tabelas	1-26
Interagindo com Arquivos de Script	1-28
Página History do <i>iSQL*Plus</i>	1-32
Definindo Preferências do <i>iSQL*Plus</i>	1-34
Definindo a Preferência de Localização da Saída	1-35
Sumário	1-36
Exercício 1: Visão Geral	1-37

2 Restringindo e Classificando Dados

Objetivos	2-2
Limitando Linhas por Seleção	2-3
Limitando as Linhas Seleccionadas	2-4
Usando a Cláusula <code>WHERE</code>	2-5
Strings de Caracteres e Datas	2-6
Condições de Comparação	2-7
Usando Condições de Comparação	2-8
Usando a Condição <code>BETWEEN</code>	2-9
Usando a Condição <code>IN</code>	2-10
Usando a Condição <code>LIKE</code>	2-11
Usando as Condições <code>NULL</code>	2-13
Condições Lógicas	2-14
Usando o Operador <code>AND</code>	2-15
Usando o Operador <code>OR</code>	2-16
Usando o Operador <code>NOT</code>	2-17
Regras de Precedência	2-18
Usando a Cláusula <code>ORDER BY</code>	2-20
Classificação	2-21
Variáveis de Substituição	2-22
Usando a Variável de Substituição <code>&</code>	2-24
Valores de Caractere e Data com Variáveis de Substituição	2-26
Especificando Nomes de Colunas, Expressões e Texto	2-27
Usando a Variável de Substituição <code>&&</code>	2-28
Usando o Comando <code>DEFINE</code> do <i>iSQL*Plus</i>	2-29
Usando o Comando <code>VERIFY</code>	2-30
Sumário	2-31
Exercício 2: Visão Geral	2-32

3 Usando Functions de uma Única Linha para Personalizar a Saída

Objetivos 3-2

Functions SQL 3-3

Dois Tipos de Functions SQL 3-4

Functions de uma Única Linha 3-5

Functions de Caractere 3-7

Functions de Manipulação de Maiúsculas e Minúsculas 3-9

Usando Functions de Manipulação de Maiúsculas e Minúsculas 3-10

Functions de Manipulação de Caracteres 3-11

Usando as Functions de Manipulação de Caracteres 3-12

Functions de Número 3-13

Usando a Function ROUND 3-14

Usando a Function TRUNC 3-15

Usando a Function MOD 3-16

Trabalhando com Datas 3-17

Aritmética com Datas 3-20

Usando Operadores Aritméticos com Datas 3-21

Functions de Data 3-22

Usando Functions de Data 3-23

Exercício 3: Visão Geral da Parte 1 3-25

Functions de Conversão 3-26

Conversão Implícita de Tipos de Dados 3-27

Conversão Explícita de Tipos de Dados 3-29

Usando a Function TO_CHAR com Datas 3-32

Elementos do Modelo de Formato de Data 3-33

Usando a Function TO_CHAR com Datas 3-37

Usando a Function TO_CHAR com Números 3-38

Usando as Functions TO_NUMBER e TO_DATE 3-41

Formato de Data RR 3-43

Exemplo do Formato de Data RR 3-44

Aninhando Functions 3-45

Functions Gerais 3-47

Function NVL 3-48

Usando a Function NVL 3-49

Usando a Function NVL2 3-50

Usando a Function NULLIF 3-51

Usando a Function COALESCE 3-52

Expressões Condicionais 3-54

Expressão CASE 3-55

Usando a Expressão CASE 3-56

Function DECODE 3-57

Usando a Function DECODE 3-58

Sumário 3-60

Exercício 3: Visão Geral da Parte 2 3-61

4 Gerando Relatórios de Dados Agregados com as Functions de Grupo

- Objetivos 4-2
- O Que São Functions de Grupo? 4-3
- Tipos de Functions de Grupo 4-4
- Functions de Grupo: Sintaxe 4-5
- Usando as Functions AVG e SUM 4-6
- Usando as Functions MIN e MAX 4-7
- Usando a Function COUNT 4-8
- Usando a Palavra-Chave DISTINCT 4-9
- Functions de Grupo e Valores Nulos 4-10
- Criando Grupos de Dados 4-11
- Criando Grupos de Dados: Sintaxe da Cláusula GROUP BY 4-12
- Usando a Cláusula GROUP BY 4-13
- Agrupando por Mais de Uma Coluna 4-15
- Usando a Cláusula GROUP BY em Várias Colunas 4-16
- Consultas Inválidas Usando Functions de Grupo 4-17
- Restringindo Resultados de Grupos 4-19
- Restringindo Resultados de Grupos com a Cláusula HAVING 4-20
- Usando a Cláusula HAVING 4-21
- Aninhando Functions de Grupo 4-23
- Sumário 4-24
- Exercício 4: Visão Geral 4-25

5 Exibindo Dados de Várias Tabelas

- Objetivos 5-2
- Obtendo Dados de Várias Tabelas 5-3
- Tipos de Joins 5-4
- Unindo Tabelas com a Sintaxe SQL:1999 5-5
- Criando Joins Naturais 5-6
- Recuperando Registros com Joins Naturais 5-7
- Criando Joins com a Cláusula USING 5-8
- Unindo Nomes de Colunas 5-9
- Recuperando Registros com a Cláusula USING 5-10
- Qualificando Nomes de Colunas Ambíguos 5-11
- Usando Apelidos de Tabelas 5-12
- Criando Joins com a Cláusula ON 5-13
- Recuperando Registros com a Cláusula ON 5-14
- Auto-Joins Usando a Cláusula ON 5-15
- Aplicando Outras Condições a uma Join 5-17
- Criando Joins Tridimensionais com a Cláusula ON 5-18
- Não-Equijoins 5-19
- Recuperando Registros com Não-Equijoins 5-20
- Joins Externas 5-21
- Joins Internas e Externas 5-22
- LEFT OUTER JOIN 5-23
- RIGHT OUTER JOIN 5-24

FULL OUTER JOIN	5-25
Produtos Cartesianos	5-26
Gerando um Produto Cartesiano	5-27
Criando Joins Cruzadas	5-28
Sumário	5-29
Exercício 5: Visão Geral	5-30

6 Usando Subconsultas para Solucionar Consultas

Objetivos	6-2
Usando uma Subconsulta para Solucionar um Problema	6-3
Sintaxe da Subconsulta	6-4
Usando uma Subconsulta	6-5
Diretrizes de Uso de Subconsultas	6-6
Tipos de Subconsultas	6-7
Subconsultas de uma Única Linha	6-8
Executando Subconsultas de uma Única Linha	6-9
Usando Functions de Grupo em uma Subconsulta	6-10
A Cláusula HAVING com Subconsultas	6-11
O Que Está Errado Nesta Instrução?	6-12
Esta Instrução Retornará Linhas?	6-13
Subconsultas de Várias Linhas	6-14
Usando o Operador ANY em Subconsultas de Várias Linhas	6-15
Usando o Operador ALL em Subconsultas de Várias Linhas	6-16
Valores Nulos em uma Subconsulta	6-17
Sumário	6-19
Exercício 6: Visão Geral	6-20

7 Usando os Operadores de Conjunto

Objetivos	7-2
Operadores de Conjunto	7-3
Tabelas Usadas Nesta Lição	7-4
Operador UNION	7-8
Usando o Operador UNION	7-9
Operador UNION ALL	7-11
Usando o Operador UNION ALL	7-12
Operador INTERSECT	7-13
Usando o Operador INTERSECT	7-14
Operador MINUS	7-15
Diretrizes de Operadores de Conjunto	7-17
O Servidor Oracle e os Operadores de Conjunto	7-18
Correspondência entre Instruções SELECT	7-19
Correspondência entre Instruções SELECT: Exemplo	7-20
Controlando a Ordem das Linhas	7-21
Sumário	7-23
Exercício 7: Visão Geral	7-24

8 Manipulando Dados

- Objetivos 8-2
- Data Manipulation Language 8-3
- Adicionando uma Nova Linha a uma Tabela 8-4
- Sintaxe da Instrução `INSERT` 8-5
- Inserindo Novas Linhas 8-6
- Inserindo Linhas com Valores Nulos 8-7
- Inserindo Valores Especiais 8-8
- Inserindo Valores de Data Específicos 8-9
- Criando um Script 8-10
- Copiando Linhas de Outra Tabela 8-11
- Alterando Dados de uma Tabela 8-12
- Sintaxe da Instrução `UPDATE` 8-13
- Atualizando Linhas de uma Tabela 8-14
- Atualizando Duas Colunas com uma Subconsulta 8-15
- Atualizando Linhas com Base em Outra Tabela 8-16
- Removendo uma Linha de uma Tabela 8-17
- Instrução `DELETE` 8-18
- Deletando Linhas de uma Tabela 8-19
- Deletando Linhas com Base em Outra Tabela 8-20
- Instrução `TRUNCATE` 8-21
- Usando uma Subconsulta em uma Instrução `INSERT` 8-22
- Transações de Banco de Dados 8-24
- Vantagens das Instruções `COMMIT` e `ROLLBACK` 8-26
- Controlando Transações 8-27
- Fazendo Rollback de Alterações até um Marcador 8-28
- Processamento de Transação Implícita 8-29
- Estado dos Dados antes de `COMMIT` ou `ROLLBACK` 8-31
- Estado dos Dados após `COMMIT` 8-32
- Submetendo Dados a Commit 8-33
- Estado dos Dados após `ROLLBACK` 8-34
- Rollback no Nível de Instrução 8-36
- Consistência de Leitura 8-37
- Implementação da Consistência de Leitura 8-38
- Sumário 8-39
- Exercício 8: Visão Geral 8-40

9 Usando Instruções DDL para Criar e Gerenciar Tabelas

- Objetivos 9-2
- Objetos de Banco de Dados 9-3
- Regras de Nomeação 9-4
- Instrução `CREATE TABLE` 9-5
- Fazendo Referência a Tabelas de Outro Usuário 9-6
- Opção `DEFAULT` 9-7
- Criando Tabelas 9-8
- Tipos de Dados 9-9
- Tipos de Dados de Data/Horário 9-11

Incluindo Constraints	9-17
Diretrizes de Constraints	9-18
Definindo Constraints	9-19
Constraint NOT NULL	9-21
Constraint UNIQUE	9-22
Constraint PRIMARY KEY	9-24
Constraint FOREIGN KEY	9-25
Constraint FOREIGN KEY: Palavras-chave	9-27
Constraint CHECK	9-28
CREATE TABLE: Exemplo	9-29
Violando Constraints	9-30
Criando uma Tabela com uma Subconsulta	9-32
Instrução ALTER TABLE	9-34
Eliminando uma Tabela	9-35
Sumário	9-36
Exercício 9: Visão Geral	9-37

10 Criando Outros Objetos de Esquema

Objetivos	10-2
Objetos de Banco de Dados	10-3
O Que É uma View?	10-4
Vantagens das Views	10-5
Views Simples e Complexas	10-6
Criando uma View	10-7
Recuperando Dados de uma View	10-10
Modificando uma View	10-11
Criando uma View Complexa	10-12
Regras para Executar Operações DML em uma View	10-13
Usando a Cláusula WITH CHECK OPTION	10-16
Negando Operações DML	10-17
Removendo uma View	10-19
Exercício 10: Visão Geral da Parte 1	10-20
Seqüências	10-21
Instrução CREATE SEQUENCE: Sintaxe	10-23
Criando uma Seqüência	10-24
Pseudocolunas NEXTVAL e CURRVAL	10-25
Usando uma Seqüência	10-27
Armazenando Valores de Seqüência em Cache	10-28
Modificando uma Seqüência	10-29
Diretrizes para Modificar uma Seqüência	10-30
Índices	10-31
Como Criar Índices?	10-33
Criando um Índice	10-34
Diretrizes para Criar Índices	10-35
Removendo um Índice	10-36

Sinônimos 10-37
 Criando e Removendo Sinônimos 10-39
 Sumário 10-40
 Exercício 10: Visão Geral da Parte 2 10-41

11 Gerenciando Objetos com Views de Dicionário de Dados

Objetivos 11-2
 O Dicionário de Dados 11-3
 Estrutura do Dicionário de Dados 11-4
 Como Usar as Views de Dicionário 11-6
 View USER_OBJECTS 11-7
 Informações sobre Tabelas 11-9
 Informações sobre Colunas 11-10
 Informações sobre Constraints 11-12
 Informações sobre Views 11-15
 Informações sobre Seqüências 11-16
 Informações sobre Sinônimos 11-18
 Adicionando Comentários a uma Tabela 11-19
 Sumário 11-20
 Exercício 11: Visão Geral 11-21

A Soluções dos Exercícios

B Dados e Descrições de Tabelas

C Sintaxe de Join Oracle

D Usando o SQL*Plus

Índice

Exercícios Adicionais

Exercícios Adicionais: Dados e Descrições de Tabelas

Exercícios Adicionais: Soluções

Usando Instruções DDL para Criar e Gerenciar Tabelas

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Ao concluir esta lição, você será capaz de:

- **Categorizar os principais objetos de banco de dados**
- **Examinar a estrutura de tabelas**
- **Listar os tipos de dados disponíveis para colunas**
- **Criar uma tabela simples**
- **Compreender como as constraints são criadas quando uma tabela é criada**
- **Descrever o funcionamento de objetos de esquema**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Esta lição apresenta as instruções DDL (Data Definition Language). Você conhecerá os princípios básicos de como criar, alterar e remover tabelas simples. São mostrados os tipos de dados disponíveis em DDL e são apresentados conceitos de esquema. Esta lição também aborda constraints. As mensagens de exceção geradas pela violação de constraints durante as operações DML são mostradas e explicadas.

Objetos de Banco de Dados

Objeto	Descrição
Tabela	Unidade básica de armazenamento; composta de linhas
View	Representa logicamente subconjuntos de dados de uma ou mais tabelas
Seqüência	Gera valores numéricos
Índice	Melhora o desempenho de algumas consultas
Sinônimo	Fornece nomes alternativos a objetos

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Objetos de Banco de Dados

Um banco de dados Oracle pode conter várias estruturas de dados. É necessário descrever cada estrutura no projeto do banco de dados para que seja possível criá-la durante o estágio de construção do desenvolvimento do banco de dados.

- **Tabela:** Armazena dados
- **View:** Subconjunto de dados de uma ou mais tabelas
- **Seqüência:** Gera valores numéricos
- **Índice:** Melhora o desempenho de algumas consultas
- **Sinônimo:** Fornece nomes alternativos a objetos

Estruturas de Tabelas do Oracle

- É possível criar tabelas a qualquer momento, mesmo enquanto os usuários usam o banco de dados.
- Não é necessário especificar o tamanho de uma tabela. O tamanho é definido, em última instância, pelo espaço alocado para o banco de dados como um todo. No entanto, é importante estimar o espaço que a tabela usará no decorrer do tempo.
- É possível modificar a estrutura de uma tabela on-line.

Observação: Existem outros objetos de banco de dados, mas eles não são abordados neste curso.

Regras de Nomeação

Os nomes de tabelas e colunas:

- Devem começar com uma letra
- Devem ter de 1 a 30 caracteres
- Devem conter apenas os caracteres A a Z, a a z, 0 a 9, _, \$ e #
- Não devem duplicar o nome de outro objeto pertencente ao mesmo usuário
- Não devem ser palavras reservadas do servidor Oracle

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Regras de Nomeação

Nomeie as colunas e as tabelas do banco de dados de acordo com as regras padrão de nomeação de objetos do banco de dados Oracle:

- Os nomes de tabelas e colunas devem começar com uma letra e ter de 1 a 30 caracteres.
- Os nomes devem conter apenas os caracteres de A a Z, a a z, 0 a 9, _ (sublinhado), \$ e # (caracteres válidos cujo uso não é incentivado).
- Os nomes não devem duplicar o nome de outro objeto pertencente ao mesmo usuário do servidor Oracle.
- Os nomes não devem ser palavras reservadas do servidor Oracle.

Diretrizes de Nomeação

Use nomes descritivos para tabelas e outros objetos de banco de dados.

Observação: Os nomes não fazem distinção entre maiúsculas e minúsculas. Por exemplo, EMPLOYEES é tratado como o mesmo nome que eMPLOYEES ou eMpLOYEES.

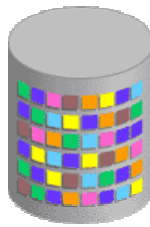
Para obter mais informações, consulte "Object Names and Qualifiers" no manual *Oracle Database SQL Reference*.

Instrução CREATE TABLE

- **Você deve ter:**
 - O privilégio CREATE TABLE
 - Uma área de armazenamento

```
CREATE TABLE [schema.] table  
    (column datatype [DEFAULT expr] [, ...]);
```

- **Especifique:**
 - O nome da tabela
 - O nome, o tipo de dados e o tamanho da coluna



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Instrução CREATE TABLE

Crie tabelas para armazenar dados executando a instrução SQL CREATE TABLE. Essa instrução é uma das instruções DDL, as quais constituem um subconjunto de instruções SQL usado para criar, modificar e remover estruturas do banco de dados Oracle. Essas instruções têm efeito imediato sobre o banco de dados e registram informações no dicionário de dados.

Para criar uma tabela, um usuário deve ter o privilégio CREATE TABLE e uma área de armazenamento na qual criará objetos. O administrador de banco de dados usa instruções DCL (Data Control Language) para conceder privilégios a usuários (as instruções DCL são abordadas em uma lição posterior).

Na sintaxe:

<i>schema</i>	é igual ao nome do proprietário
<i>table</i>	é o nome da tabela
DEFAULT <i>expr</i>	especifica um valor default quando um valor é omitido na instrução INSERT
<i>column</i>	é o nome da coluna
<i>datatype</i>	é o tipo de dados e o tamanho da coluna

Fazendo Referência a Tabelas de Outro Usuário

Um *esquema* é um conjunto de objetos. Os objetos de um esquema são as estruturas lógicas que fazem referência direta aos dados de um banco de dados. Esses objetos incluem tabelas, views, sinônimos, seqüências, stored procedures, índices, clusters e vínculos de banco de dados.

Se uma tabela não pertencer ao usuário, o nome do proprietário deverá ser incluído como prefixo da tabela. Por exemplo, se houver esquemas denominados USERA e USERB e ambos tiverem uma tabela EMPLOYEES, quando USERA quiser acessar a tabela EMPLOYEES pertencente a USERB, ele deverá adicionar o nome do esquema como prefixo ao nome da tabela:

```
SELECT *  
FROM   userb.employees;
```

Se USERB quiser acessar a tabela EMPLOYEES pertencente a USERA, ele deverá adicionar o nome do esquema como prefixo ao nome da tabela:

```
SELECT *  
FROM   usera.employees;
```


Opção DEFAULT

- Especifique um valor default para uma coluna durante uma inserção.

```
... hire_date DATE DEFAULT SYSDATE, ...
```

- Os valores literais, as expressões e as functions SQL são valores válidos.
- O nome de outra coluna ou uma pseudocoluna são valores inválidos.
- O tipo de dados default deve corresponder ao tipo de dados da coluna.

```
CREATE TABLE hire_dates  
  (id          NUMBER(8),  
   hire_date DATE DEFAULT SYSDATE);  
Table created.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Opção DEFAULT

Quando define uma tabela, você pode especificar que uma coluna receba um valor default usando a opção DEFAULT. Essa opção impedirá que sejam inseridos valores nulos em uma coluna se uma linha sem um valor for inserida nessa coluna. O valor default pode ser um literal, uma expressão ou uma function SQL (como SYSDATE ou USER), mas não pode ser o nome de outra coluna ou de uma pseudocoluna (como NEXTVAL ou CURRVAL). A expressão default deve corresponder ao tipo de dados da coluna.

Observação: CURRVAL e NEXTVAL são explicadas posteriormente nesta lição.

Criando Tabelas

- Crie a tabela.

```
CREATE TABLE dept
  (deptno      NUMBER(2),
   dname       VARCHAR2(14),
   loc         VARCHAR2(13),
   create_date DATE DEFAULT SYSDATE);
```

Table created.

- Confirme a criação da tabela.

```
DESCRIBE dept
```

Name	Null?	Type
DEPTNO		NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)
CREATE_DATE		DATE

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Criando Tabelas

O exemplo do slide cria a tabela DEPT com quatro colunas: DEPTNO, DNAME, LOC e CREATE_DATE. A coluna CREATE_DATE tem um valor default. Se nenhum valor for fornecido para uma instrução INSERT, a data do sistema será inserida automaticamente.

A confirmação da criação da tabela é obtida pela execução do comando DESCRIBE.

Como a criação de uma tabela é uma instrução DDL, ocorre um commit automático quando essa instrução é executada.

Tipos de Dados

Tipo de Dados	Descrição
<code>VARCHAR2 (size)</code>	Dados de caractere de tamanho variável
<code>CHAR (size)</code>	Dados de caractere de tamanho fixo
<code>NUMBER (p, s)</code>	Dados numéricos de tamanho variável
<code>DATE</code>	Valores de datas e horários
<code>LONG</code>	Dados de caractere de tamanho variável (até 2 GB)
<code>CLOB</code>	Dados de caractere (até 4 GB)
<code>RAW</code> e <code>LONG RAW</code>	Dados binários brutos
<code>BLOB</code>	Dados binários (até 4 GB)
<code>BFILE</code>	Dados binários armazenados em um arquivo externo (até 4 GB)
<code>ROWID</code>	Um sistema numérico de base 64 que representa o endereço exclusivo de uma linha na tabela correspondente

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Tipos de Dados

Quando identifica uma coluna de uma tabela, você precisa fornecer um tipo de dados para essa coluna. Há vários tipos de dados disponíveis:

Tipo de Dados	Descrição
<code>VARCHAR2 (size)</code>	Dados de caractere de tamanho variável (É necessário especificar um <i>tamanho</i> máximo: o <i>tamanho</i> mínimo é 1; o <i>tamanho</i> máximo é 4.000.)
<code>CHAR [(size)]</code>	Dados de caractere com tamanho fixo em bytes (O <i>tamanho</i> default e mínimo é 1; o <i>tamanho</i> máximo é 2.000.)
<code>NUMBER [(p, s)]</code>	Número com precisão <i>p</i> e escala <i>s</i> (A precisão é o número total de dígitos decimais, e a escala é o número de dígitos à direita da vírgula decimal; a precisão pode variar de 1 a 38, e a escala pode variar de -84 a 127.)
<code>DATE</code>	Valores de data e horário até o segundo mais próximo entre 1º de janeiro de 4712 A.C. e 31 de dezembro de 9999 D.C.
<code>LONG</code>	Dados de caractere de tamanho variável (até 2 GB)
<code>CLOB</code>	Dados de caractere (até 4 GB)

Tipos de Dados (continuação)

Tipo de Dados	Descrição
RAW (<i>size</i>)	Tamanho dos dados binários brutos (É necessário especificar um <i>tamanho</i> máximo: o <i>tamanho</i> máximo é 2.000.)
LONG RAW	Dados binários brutos de tamanho variável (até 2 GB)
BLOB	Dados binários (até 4 GB)
BFILE	Dados binários armazenados em um arquivo externo (até 4 GB)
ROWID	Um sistema numérico de base 64 que representa o endereço exclusivo de uma linha na tabela correspondente

Diretrizes

- Uma coluna LONG não é copiada quando uma tabela é criada com uma subconsulta.
- Não é possível incluir uma coluna LONG em uma cláusula GROUP BY ou ORDER BY.
- É possível usar somente uma coluna LONG por tabela.
- Não é possível definir constraints em uma coluna LONG.
- Convém usar uma coluna CLOB em vez de uma coluna LONG.

Tipos de Dados de Data/Horário

É possível usar vários tipos de dados de data/horário:

Tipo de Dados	Descrição
TIMESTAMP	Data com segundos fracionais
INTERVAL YEAR TO MONTH	Armazenado como um intervalo de anos e meses
INTERVAL DAY TO SECOND	Armazenado como um intervalo de dias, horas, minutos e segundos



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Outros Tipos de Dados de Data/Horário

Tipo de Dados	Descrição
TIMESTAMP	Permite que o horário seja armazenado como uma data com segundos fracionais. Há várias variações deste tipo de dados.
INTERVAL YEAR TO MONTH	Permite que o horário seja armazenado como um intervalo de anos e meses. Usado para representar a diferença entre dois valores de data/horário em que apenas o ano e o mês são significativos.
INTERVAL DAY TO SECOND	Permite que o horário seja armazenado como um intervalo de dias, horas, minutos e segundos. Usado para representar a diferença precisa entre dois valores de data/horário.

Observação: Estes tipos de dados de data/horário estão disponíveis no Oracle9i e em releases mais recentes. Para obter informações detalhadas sobre os tipos de dados de data/horário, consulte os tópicos "TIMESTAMP Datatype", "INTERVAL YEAR TO MONTH Datatype" e "INTERVAL DAY TO SECOND Datatype" no manual *Oracle SQL Reference*.

Tipos de Dados de Data/Horário

- O tipo de dados **TIMESTAMP** é uma extensão do tipo de dados **DATE**.
- Ele armazena o ano, o mês e o dia do tipo de dados **DATE**, além dos valores de hora, minuto, segundo e segundo fracional.
- Você também pode especificar o fuso horário.

```
TIMESTAMP[(fractional_seconds_precision)]
```

```
TIMESTAMP[(fractional_seconds_precision)]  
WITH TIME ZONE
```

```
TIMESTAMP[(fractional_seconds_precision)]  
WITH LOCAL TIME ZONE
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Tipo de Dados **TIMESTAMP**

O tipo de dados **TIMESTAMP** é uma extensão do tipo de dados **DATE**. Ele armazena o ano, o mês e o dia do tipo de dados **DATE**, além dos valores de hora, minuto e segundo. Esse tipo de dados é usado para armazenar valores de hora precisos.

O valor de `fractional_seconds_precision` especifica, como alternativa, o número de dígitos da parte fracional do campo de data/horário **SECOND** e pode ser um número na faixa entre 0 e 9. O default é 6.

Exemplo

Neste exemplo, a tabela **NEW_EMPLOYEES** é criada com uma coluna **START_DATE** cujo tipo de dados é **TIMESTAMP**:

```
CREATE TABLE new_employees  
(employee_id NUMBER,  
 first_name VARCHAR2(15),  
 last_name VARCHAR2(15),  
 ...  
 start_date TIMESTAMP(7),  
 ...);
```

Suponha que duas linhas sejam inseridas na tabela **NEW_EMPLOYEES**. A saída exibida mostra as diferenças. (Um tipo de dados **DATE** assume o default para exibir o formato **DD-MON-RR.**):

Tipo de Dados **TIMESTAMP** (continuação)

```
SELECT start_date
FROM   new_employees;
```

```
17-JUN-03 12.00.00.000000 AM
21-SEP-03 12.00.00.000000 AM
```

Tipo de Dados **TIMESTAMP WITH TIME ZONE**

O tipo de dados **TIMESTAMP WITH TIME ZONE** é uma variante de **TIMESTAMP** cujo valor inclui um deslocamento de fuso horário. O deslocamento de fuso horário é a diferença (em horas e minutos) entre a hora local e o UTC (Universal Time Coordinate, previamente conhecido como Horário de Greenwich). Esse tipo de dados é usado para coletar e avaliar as informações de data nas regiões geográficas.

Por exemplo:

```
TIMESTAMP '2003-04-15 8:00:00 -8:00'
```

é o mesmo que

```
TIMESTAMP '2003-04-15 11:00:00 -5:00'
```

Isto é, 8:00 a.m. Pacific Standard Time é o mesmo que 11:00 a.m. Eastern Standard Time.

Também é possível especificar esse valor desta forma:

```
TIMESTAMP '2003-04-15 8:00:00 US/Pacific'
```

Tipo de Dados **TIMESTAMP WITH LOCAL TIME ZONE**

O tipo de dados **TIMESTAMP WITH LOCAL TIME ZONE** é outra variante de **TIMESTAMP** cujo valor inclui um deslocamento de fuso horário. Ele difere do tipo de dados **TIMESTAMP WITH TIME ZONE**, pois os dados armazenados no banco de dados são normalizados para o fuso horário do banco de dados, e o deslocamento de fuso horário não é armazenado como parte dos dados da coluna. Quando os usuários recuperam os dados, eles são retornados no fuso horário da sessão local desses usuários. O deslocamento de fuso horário representa a diferença (em horas e minutos) entre o horário local e o UTC. Diferentemente de **TIMESTAMP WITH TIME ZONE**, você pode especificar colunas do tipo **TIMESTAMP WITH LOCAL TIME ZONE** como parte de uma chave primária ou exclusiva, como no exemplo a seguir:

```
CREATE TABLE time_example
  (order_date TIMESTAMP WITH LOCAL TIME ZONE);

INSERT INTO time_example VALUES('15-JAN-04 09:34:28 AM');

SELECT *
FROM   time_example;

ORDER_DATE
-----
15-JAN-04 09.34.28.000000 AM
```

O tipo **TIMESTAMP WITH LOCAL TIME ZONE** é apropriado às aplicações de duas camadas nas quais você deseja exibir datas e horários usando o fuso horário do sistema cliente.

Tipos de Dados de Data/Horário

- O tipo de dados **INTERVAL YEAR TO MONTH** armazena um período usando os campos de data/horário **YEAR** e **MONTH**:

```
INTERVAL YEAR [(year_precision)] TO MONTH
```

- O tipo de dados **INTERVAL DAY TO SECOND** armazena um período em termos de dias, horas, minutos e segundos:

```
INTERVAL DAY [(day_precision)]  
TO SECOND [(fractional_seconds_precision)]
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Tipo de Dados **INTERVAL YEAR TO MONTH**

O tipo de dados **INTERVAL YEAR TO MONTH** armazena um período usando os campos de data/horário **YEAR** e **MONTH**.

Use esse tipo de dados para representar a diferença entre dois valores de data/horário em que apenas o ano e o mês são significativos. Por exemplo, você pode usar esse valor para definir um lembrete relativo a uma data daqui a 120 meses ou verificar se já se passaram 6 meses desde uma data específica.

Na sintaxe:

year_precision é o número de dígitos no campo de data/horário **YEAR**.
O valor default de **year_precision** é 2.

Exemplos

- **INTERVAL '123-2' YEAR(3) TO MONTH**
Indica um intervalo de 123 anos e 2 meses
- **INTERVAL '123' YEAR(3)**
Indica um intervalo de 123 anos e 0 meses
- **INTERVAL '300' MONTH(3)**
Indica um intervalo de 300 meses
- **INTERVAL '123' YEAR**
Retorna um erro, pois a precisão default é 2 e 123 tem 3 dígitos

Tipo de Dados INTERVAL YEAR TO MONTH (continuação)

```
CREATE TABLE time_example2
(loan_duration INTERVAL YEAR (3) TO MONTH);

INSERT INTO time_example2 (loan_duration)
VALUES (INTERVAL '120' MONTH(3));

SELECT TO_CHAR( sysdate+loan_duration, 'dd-mon-yyyy')
FROM time_example2;           --today's date is 26-Sep-2001
```

TO_CHAR(SYS
26-sep-2011

Tipo de Dados INTERVAL DAY TO SECOND

O tipo de dados INTERVAL DAY TO SECOND armazena um período em termos de dias, horas, minutos e segundos.

Utilize esse tipo de dados para representar a diferença precisa entre dois valores de data/horário. Por exemplo, você pode utilizar esse valor para definir um lembrete relativo a um horário daqui a 36 horas ou registrar o horário entre o início e o término de uma corrida. Para representar com alta precisão longos intervalos de tempo, que incluem vários anos, utilize um valor alto para a parte relativa a dias.

Na sintaxe:

day_precision	é o número de dígitos no campo de data/horário DAY. Os valores aceitos vão de 0 a 9. O default é 2.
fractional_seconds_precision	é o número de dígitos da parte fracional do campo de data/horário SECOND. Os valores aceitos vão de 0 a 9. O default é 6.

Exemplos

- INTERVAL '4 5:12:10.222' DAY TO SECOND(3)
Indica 4 dias, 5 horas, 12 minutos, 10 segundos e 222 miliares de segundo.
- INTERVAL '180' DAY(3)
Indica 180 dias.
- INTERVAL '4 5:12:10.222' DAY TO SECOND(3)
Indica 4 dias, 5 horas, 12 minutos, 10 segundos e 222 miliares de segundo
- INTERVAL '4 5:12' DAY TO MINUTE
Indica 4 dias, 5 horas e 12 minutos
- INTERVAL '400 5' DAY(3) TO HOUR
Indica 400 dias e 5 horas.
- INTERVAL '11:12:10.2222222' HOUR TO SECOND(7)
Indica 11 horas, 12 minutos e 10,2222222 segundos.

Tipo de Dados INTERVAL DAY TO SECOND (continuação)

Exemplo

```
CREATE TABLE time_example3
(day_duration INTERVAL DAY (3) TO SECOND);

INSERT INTO time_example3 (day_duration)
VALUES (INTERVAL '180' DAY(3));

SELECT sysdate + day_duration "Half Year"
FROM   time_example3;           --today's date is 26-Sep-2001
```

Half Year
25-MAR-02

Incluindo Constraints

- As constraints impõem regras no nível da tabela.
- As constraints impedem a deleção de uma tabela quando existem dependências.
- Os seguintes tipos de constraints são válidos:
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraints

O servidor Oracle usa constraints para impedir a entrada de dados inválidos em tabelas.

Você pode usar constraints para:

- Impor regras aos dados de uma tabela sempre que uma linha for inserida, atualizada ou deletada dessa tabela. Para que a operação seja bem-sucedida, é necessário atender à constraint.
- Impedir a deleção de uma tabela quando existem dependências de outras tabelas
- Fornecer regras para ferramentas Oracle, como o Oracle Developer

Constraints de Integridade de Dados

Constraint	Descrição
NOT NULL	Especifica a coluna que não pode conter um valor nulo
UNIQUE	Especifica uma coluna ou uma combinação de colunas cujo valores devem ser exclusivos para todas as linhas na tabela
PRIMARY KEY	Identifica exclusivamente cada linha na tabela
FOREIGN KEY	Estabelece e impõe um relacionamento de chave estrangeira entre a coluna e uma coluna da tabela referenciada
CHECK	Especifica uma condição que deve ser verdadeira

Diretrizes de Constraints

- Nomeie uma constraint, ou o servidor Oracle gerará um nome usando o formato `SYS_Cn`.
- Crie uma constraint em um destes momentos:
 - Durante a criação da tabela
 - Após a criação da tabela
- Defina uma constraint no nível da coluna ou da tabela.
- Exiba uma constraint no dicionário de dados.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Diretrizes de Constraints

Todas as constraints são armazenadas no dicionário de dados. Se você fornecer às constraints um nome significativo, poderá facilmente fazer referência a elas. Os nomes das constraints deverão seguir as regras padrão de nomeação de objetos. Se você não nomear a constraint, o servidor Oracle gerará um nome com o formato `SYS_Cn`, em que *n* é um inteiro, para que o nome da constraint seja exclusivo.

É possível definir constraints no momento da criação da tabela ou depois que a tabela é criada.

Para obter mais informações, consulte "Constraints" no manual *Oracle Database SQL Reference*.

Definindo Constraints

- **Sintaxe:**

```
CREATE TABLE [schema.] table
  (column datatype [DEFAULT expr]
   [column_constraint],
   ...
   [table_constraint] [, ...]);
```

- **Constraint no nível da coluna:**

```
column [CONSTRAINT constraint_name] constraint_type,
```

- **Constraint no nível da tabela:**

```
column, ...
[CONSTRAINT constraint_name] constraint_type
(column, ...),
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Definindo Constraints

O slide mostra a sintaxe de definição de constraints durante a criação de uma tabela. É possível criar as constraints no nível da coluna ou da tabela. As constraints definidas no nível da coluna são incluídas quando a coluna é definida. As constraints no nível da tabela são especificadas no final da definição da tabela e devem fazer referência a uma ou mais colunas relativas à constraint em um conjunto de parênteses.

É necessário definir as constraints NOT NULL no nível da coluna.

É necessário definir as constraints que se aplicam a mais de uma coluna no nível da tabela.

Na sintaxe:

schema	é igual ao nome do proprietário
table	é o nome da tabela
DEFAULT expr	especifica um valor default a ser usado se for omitido um valor na instrução INSERT
column	é o nome da coluna
datatype	é o tipo de dados e o tamanho da coluna
column_constraint	é uma constraint de integridade que faz parte da definição da coluna
table_constraint	é uma constraint de integridade que faz parte da definição da tabela

Definindo Constraints

- **Constraint no nível da coluna:**

```
CREATE TABLE employees(  
  employee_id NUMBER(6)  
    CONSTRAINT emp_emp_id_pk PRIMARY KEY,  
  first_name VARCHAR2(20),  
  ...);
```

1

- **Constraint no nível da tabela:**

```
CREATE TABLE employees(  
  employee_id NUMBER(6),  
  first_name VARCHAR2(20),  
  ...  
  job_id VARCHAR2(10) NOT NULL,  
  CONSTRAINT emp_emp_id_pk  
    PRIMARY KEY (EMPLOYEE_ID));
```

2

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Definindo Constraints (continuação)

Normalmente, as constraints são criadas junto com a tabela. É possível adicionar constraints a uma tabela após a sua criação e desativá-las temporariamente.

Os dois exemplos do slide criam uma constraint de chave primária na coluna EMPLOYEE_ID da tabela EMPLOYEES.

1. O primeiro exemplo usa a sintaxe no nível da coluna para definir a constraint.
2. O segundo exemplo usa a sintaxe no nível da tabela para definir a constraint.

Informações mais detalhadas sobre a constraint de chave primária são fornecidas posteriormente nesta lição.

Constraint NOT NULL

Garante que a coluna não aceite valores nulos:

EMPLOYEE_ID	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	DEPARTMENT_ID
100	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000	90
101	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000	90
102	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000	90
103	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000	60
104	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	6000	60
178	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	7000	
200	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	4400	10

...
20 rows selected.

↑
Constraint NOT NULL
(Nenhuma linha desta
coluna pode conter
um valor nulo.)

↑
**Constraint
NOT NULL**

↑
**Ausência da
constraint NOT NULL**
(Qualquer linha desta
coluna pode conter
um valor nulo.)

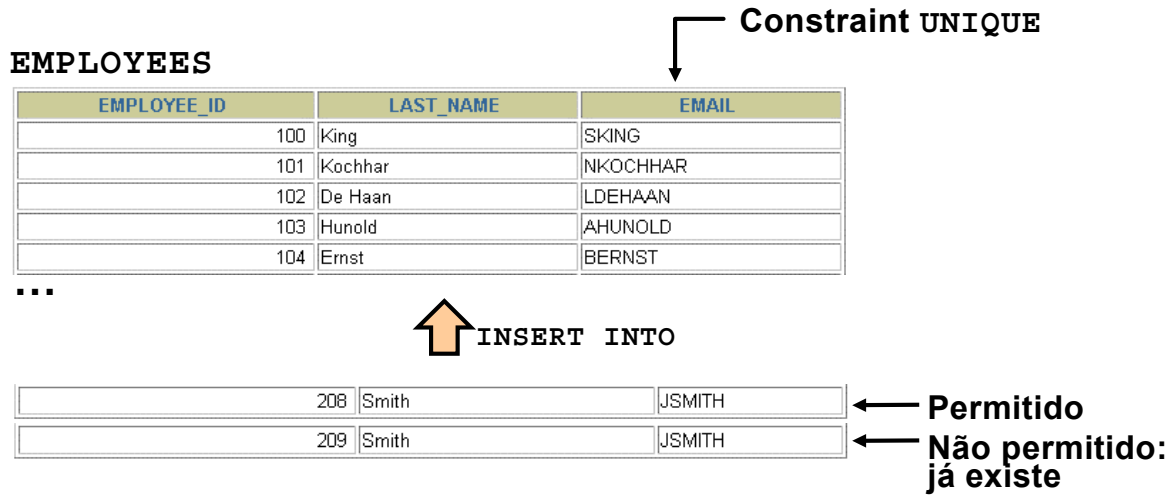
ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint NOT NULL

A constraint NOT NULL garante que a coluna não contenha valores nulos. As colunas sem essa constraint podem conter valores nulos por default. É necessário definir as constraints NOT NULL no nível da coluna.

Constraint UNIQUE



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint UNIQUE

A constraint de integridade de chave UNIQUE exige que todos os valores de uma coluna ou conjunto de colunas (chave) sejam exclusivos – isto é, duas linhas de uma tabela não podem ter valores duplicados em uma coluna ou conjunto de colunas especificado. A coluna (ou o conjunto de colunas) incluída na definição da constraint de chave UNIQUE é denominada *chave exclusiva*. Se a constraint UNIQUE incluir mais de uma coluna, esse grupo de colunas será denominado *chave exclusiva composta*.

As constraints UNIQUE permitirão a entrada de valores nulos, a menos que você também defina constraints NOT NULL para as mesmas colunas. Na verdade, qualquer linha de coluna sem a constraint NOT NULL pode incluir valores nulos, já que os valores nulos não são considerados iguais a nenhum valor. Um valor nulo em uma coluna (ou em todas as colunas da chave UNIQUE composta) sempre atende a uma constraint UNIQUE.

Observação: Em função do mecanismo de pesquisa de constraints UNIQUE em mais de uma coluna, você não pode atribuir valores idênticos nas colunas não nulas de uma constraint de chave UNIQUE composta parcialmente nula.

Constraint UNIQUE

Definida no nível da tabela ou da coluna:

```
CREATE TABLE employees(  
    employee_id      NUMBER(6),  
    last_name        VARCHAR2(25) NOT NULL,  
    email            VARCHAR2(25),  
    salary            NUMBER(8,2),  
    commission_pct   NUMBER(2,2),  
    hire_date        DATE NOT NULL,  
    ...  
    CONSTRAINT emp_email_uk UNIQUE(email));
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint UNIQUE (continuação)

É possível definir constraints UNIQUE no nível da coluna ou da tabela. Uma chave exclusiva composta é criada por meio da definição no nível da tabela.

O exemplo do slide aplica a constraint UNIQUE à coluna EMAIL da tabela EMPLOYEES. O nome da constraint é EMP_EMAIL_UK.

Observação: O servidor Oracle impõe a constraint UNIQUE criando implicitamente um índice exclusivo na(s) coluna(s) de chave exclusiva.

Constraint PRIMARY KEY

DEPARTMENTS

PRIMARY KEY

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

...

Não permitido
(valor nulo)



INSERT INTO

	Public Accounting		1400
50	Finance	124	1500

Não permitido
(já existe o valor 50)

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

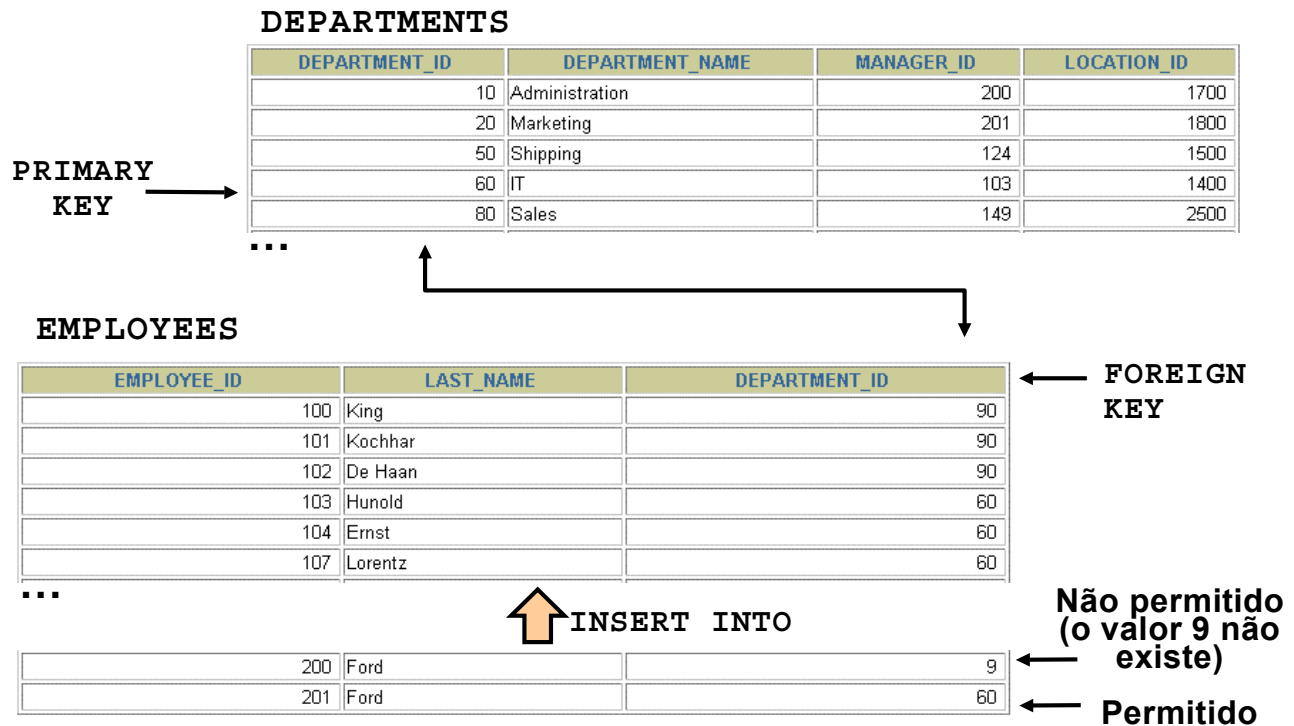
Oracle University and Impacta Tecnologia use only.

Constraint PRIMARY KEY

A constraint PRIMARY KEY cria uma chave primária para a tabela. Só é possível criar uma chave primária para cada tabela. A constraint PRIMARY KEY é uma coluna ou um conjunto de colunas que identifica com exclusividade cada linha de uma tabela. Essa constraint impõe a exclusividade da coluna ou da combinação de colunas e garante que nenhuma coluna que faça parte da chave primária possa conter um valor nulo.

Observação: Como a exclusividade faz parte da definição de constraint de chave primária, o servidor Oracle impõe a exclusividade criando implicitamente um índice exclusivo na(s) coluna(s) de chave primária.

Constraint FOREIGN KEY



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint FOREIGN KEY

A constraint FOREIGN KEY (ou constraint de integridade referencial) designa uma coluna ou uma combinação de colunas como chave estrangeira e estabelece um relacionamento entre uma chave primária ou uma chave exclusiva na mesma tabela ou em uma tabela diferente.

No exemplo do slide, a coluna DEPARTMENT_ID foi definida como a chave estrangeira da tabela EMPLOYEES (tabela filha ou dependente); ela faz referência à coluna DEPARTMENT_ID da tabela DEPARTMENTS (tabela mãe ou referenciada).

Diretrizes

- Um valor de chave estrangeira deve corresponder a um valor existente na tabela mãe ou deve ser NULL.
- As chaves estrangeiras baseiam-se em valores de dados e são ponteiros lógicos, e não físicos.

Constraint FOREIGN KEY

Definida no nível da tabela ou da coluna:

```
CREATE TABLE employees(  
    employee_id      NUMBER(6),  
    last_name        VARCHAR2(25) NOT NULL,  
    email            VARCHAR2(25),  
    salary            NUMBER(8,2),  
    commission_pct   NUMBER(2,2),  
    hire_date        DATE NOT NULL,  
    ...  
    department_id    NUMBER(4),  
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)  
        REFERENCES departments(department_id),  
    CONSTRAINT emp_email_uk UNIQUE(email));
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint FOREIGN KEY (continuação)

É possível definir constraints FOREIGN KEY no nível da coluna ou da tabela. Uma chave estrangeira composta deve ser criada por meio da definição no nível da tabela.

O exemplo do slide define uma constraint FOREIGN KEY na coluna DEPARTMENT_ID da tabela EMPLOYEES usando a sintaxe no nível da tabela. O nome da constraint é EMP_DEPTID_FK.

Também é possível definir a chave estrangeira no nível da coluna, desde que a constraint seja baseada em uma única coluna. A diferença na sintaxe é que as palavras-chave FOREIGN KEY não aparecem. Por exemplo:

```
CREATE TABLE employees  
(  
    ...  
    department_id NUMBER(4) CONSTRAINT emp_deptid_fk  
        REFERENCES departments(department_id),  
    ...  
)
```

Constraint FOREIGN KEY: Palavras-chave

- **FOREIGN KEY:** Define a coluna da tabela filha no nível de constraint da tabela
- **REFERENCES:** Identifica a tabela e a coluna da tabela mãe
- **ON DELETE CASCADE:** Deleta as linhas dependentes da tabela filha quando uma linha da tabela mãe é deletada
- **ON DELETE SET NULL:** Converte os valores da chave estrangeira dependente em nulos

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint FOREIGN KEY: Palavras-chave

A chave estrangeira é definida na tabela filha e a tabela que contém a coluna referenciada é a tabela mãe. Essa chave é definida por meio de uma combinação das seguintes palavras-chave:

- **FOREIGN KEY** é usada para definir a coluna da tabela filha no nível de constraint de tabela.
- **REFERENCES** identifica a tabela e a coluna da tabela mãe.
- **ON DELETE CASCADE** indica que, quando a linha da tabela mãe é deletada, as linhas dependentes da tabela filha também são deletadas.
- **ON DELETE SET NULL** converte os valores da chave estrangeira em nulos quando o valor pai é removido.

O comportamento default é chamado de *regra restrita*, que não permite a atualização ou a deleção de dados referenciados.

Sem a opção **ON DELETE CASCADE** ou **ON DELETE SET NULL**, não será possível deletar a linha da tabela mãe se for feita referência a ela na tabela filha.

Constraint CHECK

- Define uma condição que cada linha deve atender
- As seguintes expressões não são permitidas:
 - Referências às pseudocolunas CURRVAL, NEXTVAL, LEVEL e ROWNUM
 - Chamadas de functions SYSDATE, UID, USER e USERENV
 - Consultas que fazem referência a outros valores em outras linhas

```
..., salary    NUMBER(2)  
      CONSTRAINT emp_salary_min  
      CHECK (salary > 0),...
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Constraint CHECK

A constraint CHECK define uma condição que cada linha deve atender. A condição pode usar a mesma construção que as condições de consulta, com as seguintes exceções:

- Referências às pseudocolunas CURRVAL, NEXTVAL, LEVEL e ROWNUM
- Chamadas de functions SYSDATE, UID, USER e USERENV
- Consultas que fazem referência a outros valores em outras linhas

Uma única coluna pode ter várias constraints CHECK que fazem referência à coluna na respectiva definição. Não há limite ao número de constraints CHECK que você pode definir em uma coluna.

É possível definir constraints CHECK no nível da coluna ou da tabela.

```
CREATE TABLE employees  
(...  
  salary NUMBER(8,2) CONSTRAINT emp_salary_min  
  CHECK (salary > 0),  
  ...
```

CREATE TABLE: Exemplo

```
CREATE TABLE employees
( employee_id      NUMBER(6)
  CONSTRAINT emp_employee_id PRIMARY KEY
, first_name       VARCHAR2(20)
, last_name        VARCHAR2(25)
  CONSTRAINT emp_last_name_nn NOT NULL
, email            VARCHAR2(25)
  CONSTRAINT emp_email_nn    NOT NULL
  CONSTRAINT emp_email_uk    UNIQUE
, phone_number     VARCHAR2(20)
, hire_date        DATE
  CONSTRAINT emp_hire_date_nn NOT NULL
, job_id           VARCHAR2(10)
  CONSTRAINT emp_job_nn      NOT NULL
, salary           NUMBER(8,2)
  CONSTRAINT emp_salary_ck   CHECK (salary>0)
, commission_pct   NUMBER(2,2)
, manager_id       NUMBER(6)
, department_id    NUMBER(4)
  CONSTRAINT emp_dept_fk    REFERENCES
    departments (department_id));
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Exemplo de CREATE TABLE

O exemplo mostra a instrução usada para criar a tabela EMPLOYEES no esquema HR.

Violando Constraints

```
UPDATE employees
SET department_id = 55
WHERE      department_id = 110;
```

```
UPDATE employees
      *
ERROR at line 1:
ORA-02291: integrity constraint (HR.EMP_DEPT_FK)
violated - parent key not found
```

O departamento 55 não existe.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Erro de Constraint de Integridade

Quando há constraints em colunas, um erro será exibido se você tentar violar a regra da constraint.

Por exemplo, se você tentar atualizar um registro com um valor vinculado a uma constraint de integridade, um erro será exibido.

No exemplo do slide, como o departamento 55 não existe na tabela mãe DEPARTMENTS, é exibido o erro ORA-02291 de violação de *chave mãe*.

Violando Constraints

Não é possível deletar uma linha que contém uma chave primária usada como chave estrangeira em outra tabela.

```
DELETE FROM departments
WHERE      department_id = 60;
```

```
DELETE FROM departments
          *
ERROR at line 1:
ORA-02292: integrity constraint (HR.EMP_DEPT_FK)
violated - child record found
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Erro de Constraint de Integridade (continuação)

Se você tentar deletar um registro com um valor vinculado a uma constraint de integridade, um erro será exibido.

O exemplo do slide tenta deletar o departamento 60 da tabela DEPARTMENTS, mas ocorre um erro porque o número do departamento é usado como chave estrangeira na tabela EMPLOYEES. Se o registro pai que você tenta deletar tiver registros filhos, será exibido o erro ORA-02292 de violação de *registro filho encontrado*.

Esta instrução funciona porque não há funcionários no departamento 70:

```
DELETE FROM departments
WHERE      department_id = 70;
```

1 row deleted.

Criando uma Tabela com uma Subconsulta

- Crie uma tabela e insira linhas combinando a instrução `CREATE TABLE` e a opção de *subconsulta* `AS`.

```
CREATE TABLE table  
      [(column, column...)]  
AS subquery;
```

- Estabeleça uma correspondência entre as colunas especificadas e o número de colunas da subconsulta.
- Defina colunas com nomes e valores default.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Criando uma Tabela com Linhas de Outra Tabela

Um segundo método para criar uma tabela é aplicar a cláusula de *subconsulta* `AS`, que cria a tabela e insere linhas retornadas da subconsulta.

Na sintaxe:

<i>table</i>	é o nome da tabela
<i>column</i>	é o nome da coluna, o valor default e a constraint de integridade
<i>subquery</i>	é a instrução <code>SELECT</code> que define o conjunto de linhas a ser inserido na nova tabela

Diretrizes

- A tabela é criada com os nomes de colunas especificados, e as linhas recuperadas pela instrução `SELECT` são inseridas nessa tabela.
- A definição da coluna só poderá conter o nome da coluna e o valor default.
- Se forem fornecidas especificações de colunas, o número de colunas deverá ser igual ao número de colunas da lista da subconsulta `SELECT`.
- Se não forem fornecidas especificações de colunas, os nomes das colunas da tabela serão iguais aos nomes das colunas incluídos na subconsulta.
- As regras de integridade não serão passadas para a nova tabela, somente as definições de tipos de dados das colunas.

Criando uma Tabela com uma Subconsulta

```
CREATE TABLE dept80
AS
SELECT  employee_id, last_name,
        salary*12 ANNSAL,
        hire_date
FROM    employees
WHERE   department id = 80;
```

Table created.

```
DESCRIBE dept80
```

Name	Null?	Type
EMPLOYEE_ID		NUMBER(6)
LAST_NAME	NOT NULL	VARCHAR2(25)
ANNSAL		NUMBER
HIRE_DATE	NOT NULL	DATE

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Criando uma Tabela com Linhas de Outra Tabela (continuação)

O exemplo do slide cria uma tabela denominada DEPT80, que contém detalhes de todos os funcionários que trabalham no departamento 80. Observe que os dados da tabela DEPT80 originam-se da tabela EMPLOYEES.

Você pode verificar a existência de uma tabela de banco de dados e as definições das colunas com o comando *iSQL*Plus* DESCRIBE.

Forneça um apelido de coluna quando selecionar uma expressão. A expressão SALARY*12 recebe o apelido ANNSAL. Sem o apelido, o seguinte erro é gerado:

ERROR at line 3:

ORA-00998: must name this expression with a column alias

Instrução ALTER TABLE

Use a instrução ALTER TABLE para:

- Adicionar uma nova coluna
- Modificar uma coluna existente
- Definir um valor default para a nova coluna
- Eliminar uma coluna

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Instrução ALTER TABLE

Após criar uma tabela, você talvez precise alterar sua estrutura por uma destas razões:

- Para acrescentar uma coluna omitida.
- Para alterar a definição de uma coluna.
- Para remover colunas.

Para fazer isso, use a instrução ALTER TABLE. Para obter informações sobre a instrução ALTER TABLE, consulte o curso *Banco de Dados Oracle 10g: Fundamentos de SQL II*.

Eliminando uma Tabela

- A estrutura e os dados da tabela são deletados.
- As transações pendentes são submetidas a commit.
- Todos os índices são eliminados.
- Todas as constraints são eliminadas.
- **Não é possível fazer rollback da instrução DROP TABLE.**

```
DROP TABLE dept80;  
Table dropped.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Eliminando uma Tabela

A instrução DROP TABLE remove a definição de uma tabela Oracle. Quando você elimina uma tabela, o banco de dados perde todos os dados dessa tabela e todos os índices associados a ela.

Sintaxe

```
DROP TABLE table
```

Na sintaxe, *table* é o nome da tabela.

Diretrizes

- Todos os dados são deletados da tabela.
- As views e os sinônimos permanecem, mas são inválidos.
- As transações pendentes são submetidas a commit.
- Somente o autor da tabela ou um usuário com o privilégio DROP ANY TABLE pode remover uma tabela.

Observação: A instrução DROP TABLE, depois de executada, é irreversível. O servidor Oracle não questiona a ação quando você executa a instrução DROP TABLE. Se você for proprietário ou tiver um privilégio de alto nível na tabela, essa tabela será removida imediatamente. Como ocorre com todas as instruções DDL, o commit de DROP TABLE é efetuado automaticamente.

Sumário

Nesta lição, você aprendeu a usar a instrução `CREATE TABLE` para criar uma tabela e incluir constraints.

- **Categorize os principais objetos de banco de dados**
- **Examine a estrutura de tabelas**
- **Liste os tipos de dados disponíveis para colunas**
- **Crie uma tabela simples**
- **Compreenda como as constraints são criadas quando uma tabela é criada**
- **Descreva o funcionamento dos objetos de esquema**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Sumário

Nesta lição, você aprendeu a:

CREATE TABLE

- Usar a instrução `CREATE TABLE` para criar uma tabela e incluir constraints.
- Criar uma tabela baseada em outra usando uma subconsulta.

DROP TABLE

- Remover linhas e a estrutura de uma tabela.
- Depois que esta instrução for executada, seu rollback não será permitido.

Exercício 9: Visão Geral

Este exercício aborda os seguintes tópicos:

- **Criando novas tabelas**
- **Criando uma nova tabela com a sintaxe `CREATE TABLE AS`**
- **Verificando a existência de tabelas**
- **Eliminando tabelas**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Exercício 9: Visão Geral

Crie novas tabelas com a instrução `CREATE TABLE`. Verifique se a nova tabela foi adicionada ao banco de dados. Crie a sintaxe no arquivo de comandos e execute esse arquivo para criar a tabela.

Exercício 9

1. Crie a tabela DEPT com base no quadro de instâncias de tabela a seguir. Inclua a sintaxe no script lab_09_01.sql e execute a instrução no script para criar a tabela. Verifique se a tabela foi criada.

Nome da Coluna	ID	NAME
Tipo de Chave	Primary key	
Valores Nulos/Exclusivos		
Tabela FK		
Coluna FK		
Tipo de dados	NUMBER	VARCHAR2
Tamanho	7	25

Name	Null?	Type
ID		NUMBER(7)
NAME		VARCHAR2(25)

2. Preencha a tabela DEPT com dados da tabela DEPARTMENTS. Inclua somente as colunas necessárias.
3. Crie a tabela EMP com base no quadro de instâncias de tabela a seguir. Inclua a sintaxe no script lab_09_03.sql e execute a instrução nesse script para criar a tabela. Verifique se a tabela foi criada.

Nome da Coluna	ID	LAST_NAME	FIRST_NAME	DEPT_ID
Tipo de Chave				
Valores Nulos/Exclusivos				
Tabela FK				DEPT
Coluna FK				ID
Tipo de dados	NUMBER	VARCHAR2	VARCHAR2	NUMBER
Tamanho	7	25	25	7

Name	Null?	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)

Exercício 9 (continuação)

4. Crie a tabela EMPLOYEES2 com base na estrutura da tabela EMPLOYEES. Inclua somente as colunas EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY e DEPARTMENT_ID. Nomeie as colunas da nova tabela como ID, FIRST_NAME, LAST_NAME, SALARY e DEPT_ID, respectivamente.
5. Elimine a tabela EMP.

10

Criando Outros Objetos de Esquema

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Ao concluir esta lição, você será capaz de:

- **Criar views simples e complexas**
- **Recuperar dados de views**
- **Criar, manter e usar seqüências**
- **Criar e manter índices**
- **Criar sinônimos privados e públicos**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Esta lição apresenta os objetos view, seqüência, sinônimo e índice. Você conhecerá os princípios básicos de como criar e usar views, seqüências e índices.

Objetos de Banco de Dados

Objeto	Descrição
Tabela	Unidade básica de armazenamento; composta de linhas
View	Representa logicamente subconjuntos de dados de uma ou mais tabelas
Seqüência	Gera valores numéricos
Índice	Melhora o desempenho de algumas consultas
Sinônimo	Fornece nomes alternativos a objetos

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetos de Banco de Dados

Além das tabelas, existem vários outros objetos em um banco de dados. Nesta lição, você conhecerá objetos como views, seqüências, índices e sinônimos.

As views permitem exibir e ocultar dados de tabelas.

Muitas aplicações exigem o uso de números exclusivos como valores de chave primária. Você pode criar um código na aplicação para atender a esse requisito ou usar uma seqüência para gerar números exclusivos.

Para melhorar o desempenho de algumas consultas, considere a criação de um índice. Use índices também para impor exclusividade em uma coluna ou conjunto de colunas.

Você pode fornecer nomes alternativos a objetos usando sinônimos.

O Que É uma View?

Tabela EMPLOYEES

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALA
100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_FRES	2401
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	1701
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	1701
103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	901
104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	601
107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-93	IT_PROG	421
124	Kevin	Mourgos	KMOURGOS	650.123.5234	18-NOV-99	ST_MAN	581
141	Trenna	Rae	TRAE	650.121.3009	17-OCT-95	ST_CLERK	351
142	Curtis	Davis	CDAVIES	650.121.2994	29-JAN-97	ST_CLERK	311
143	Randall	Mates	RMATES	650.121.2074	15-MAR-90	ST_CLERK	261
149	Zlotkey			10500	JUL-96	ST_CLERK	251
174	Abel			11000	JAN-00	SA_MAN	1051
175	Taylor			0600	MAY-96	SA_REP	1101
176	Rudenberg	Grant	GRANT	011.44.1844.429200	MAR-98	SA_REP	861
177	Rudenberg	Grant	GRANT	011.44.1844.429200	24-MAY-99	SA_REP	701
200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	441
201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-96	MK_MAN	1301
202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MK_REP	601
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	1201
206	William	Gietz	WGIEZT	515.123.8181	07-JUN-94	AC_ACCOUNT	831

20 rows selected.

ORACLE

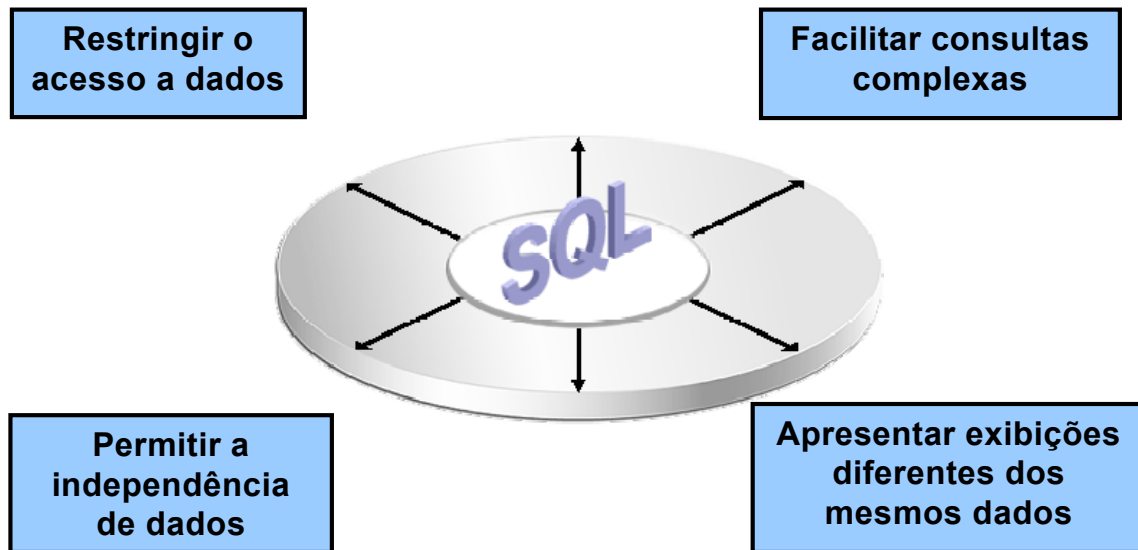
Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

O Que É uma View?

Você pode apresentar combinações ou subconjuntos lógicos de dados criando views de tabelas. Uma view é uma tabela lógica baseada em uma tabela ou em outra view. Uma view em si não contém dados, mas é semelhante a uma janela por meio da qual é possível exibir ou alterar dados de tabelas. As tabelas nas quais uma view é baseada são denominadas *tabelas-base*. A view é armazenada como uma instrução SELECT no dicionário de dados.

Vantagens de Views



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Vantagens de Views

- As views restringem o acesso aos dados, pois podem exibir colunas selecionadas da tabela.
- As views permitem fazer consultas simples para recuperar os resultados de consultas complicadas. Por exemplo, as views permitem aos usuários consultar informações de várias tabelas mesmo sem saber criar uma instrução de join.
- As views permitem a independência de dados a usuários ad hoc e programas aplicativos. É possível usar uma view para recuperar dados de várias tabelas.
- As views permitem o acesso de grupos de usuários a dados de acordo com critérios específicos.

Para obter mais informações, consulte "CREATE VIEW" no manual *Oracle SQL Reference*.

Views Simples e Complexas

Recurso	Views Simples	Views Complexas
Número de tabelas	Uma	Uma ou mais
Contêm functions	Não	Sim
Contêm grupos de	Não	Sim
Permitem a execução de operações DML	Sim	Nem sempre

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Views Simples e Complexas

Existem duas classificações para views: simples e complexas. A diferença básica está relacionada às operações DML (INSERT, UPDATE e DELETE).

- Uma view simples é aquela que:
 - É derivada de dados de uma única tabela
 - Não contém functions ou grupos de dados
 - Permite a execução de operações DML
- Uma view complexa é aquela que:
 - É derivada de dados de várias tabelas
 - Contém functions ou grupos de dados
 - Nem sempre permite a execução de operações DML

Criando uma View

- Incorpore uma subconsulta à instrução

CREATE VIEW:

```
CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW view
  [(alias[, alias]...)]
  AS subquery
[WITH CHECK OPTION [CONSTRAINT constraint]]
[WITH READ ONLY [CONSTRAINT constraint]];
```

- A subconsulta pode conter uma sintaxe SELECT complexa.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Criando uma View

Você pode criar uma view incorporando uma subconsulta à instrução CREATE VIEW.

Na sintaxe:

OR REPLACE	recria a view quando ela já existe
FORCE	cria a view independentemente da existência das tabelas-base
NOFORCE	só cria a view quando as tabelas-base existem (Este é o default.)
<i>view</i>	é o nome da view
<i>alias</i>	especifica nomes para as expressões selecionadas pela consulta da view (O número de apelidos deve corresponder ao número de expressões selecionadas pela view.)
<i>subquery</i>	é uma instrução SELECT completa (Você pode usar apelidos para as colunas na lista SELECT.)
WITH CHECK OPTION	especifica que apenas as linhas acessíveis à view podem ser inseridas ou atualizadas
<i>constraint</i>	é o nome designado à constraint CHECK OPTION
WITH READ ONLY	garante que não seja possível executar operações DML na view

Criando uma View

- Crie a view EMPVU80 com detalhes de funcionários do departamento 80:

```
CREATE VIEW empvu80
AS SELECT employee_id, last_name, salary
FROM employees
WHERE department_id = 80;
View created.
```

- Descreva a estrutura da view usando o comando iSQL*Plus DESCRIBE:

```
DESCRIBE empvu80
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Criando uma View (continuação)

O exemplo do slide cria uma view com o número, o sobrenome e o salário de cada funcionário do departamento 80.

Você pode exibir a estrutura da view usando o comando iSQL*Plus DESCRIBE.

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
LAST_NAME	NOT NULL	VARCHAR2(25)
SALARY		NUMBER(8,2)

Diretrizes para Criar uma View:

- A subconsulta que define uma view pode conter a sintaxe SELECT complexa, inclusive joins, grupos e subconsultas.
- Se você não especificar um nome de constraint para a view criada com WITH CHECK OPTION, o sistema designará um nome default no formato SYS_Cn.
- Você pode usar a opção OR REPLACE para alterar a definição da view sem eliminá-la e recriá-la, ou sem conceder novamente com grant privilégios de objeto concedidos anteriormente.

Criando uma View

- **Crie uma view usando apelidos de colunas na subconsulta:**

```
CREATE VIEW    salvu50
AS SELECT    employee_id ID_NUMBER, last_name NAME,
            salary*12 ANN_SALARY
FROM        employees
WHERE       department_id = 50;
View created.
```

- **Selecione as colunas dessa view pelos apelidos definidos:**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Criando uma View (continuação)

Você pode controlar os nomes das colunas incluindo apelidos de colunas na subconsulta.

O exemplo do slide cria uma view que contém o número (EMPLOYEE_ID) com o apelido ID_NUMBER, o nome (LAST_NAME) com o apelido NAME e o salário anual (SALARY) com o apelido ANN_SALARY de todos os funcionários do departamento 50.

Como alternativa, você pode usar um apelido depois da instrução CREATE e antes da subconsulta SELECT. O número de apelidos listados deve corresponder ao número de expressões selecionadas na subconsulta.

```
CREATE OR REPLACE VIEW    salvu50 (ID_NUMBER, NAME, ANN_SALARY)
AS SELECT    employee_id, last_name, salary*12
FROM        employees
WHERE       department_id = 50;
View created.
```

Recuperando Dados de uma View

```
SELECT *  
FROM salvu50;
```

ID_NUMBER	NAME	ANN_SALARY
124	Mourgos	69600
141	Rajs	42000
142	Davies	37200
143	Matos	31200
144	Vargas	30000

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Recuperando Dados de uma View

Você pode recuperar dados de uma view como faria com qualquer tabela. É possível exibir todo o conteúdo da view ou apenas linhas e colunas específicas.

Modificando uma View

- **Modifique a view EMPVU80 usando a cláusula CREATE OR REPLACE VIEW. Adicione um apelido para cada nome de coluna:**

```
CREATE OR REPLACE VIEW empvu80
(id_number, name, sal, department_id)
AS SELECT  employee_id, first_name || ' '
           || last_name, salary, department_id
FROM      employees
WHERE     department_id = 80;
View created.
```

- **Os apelidos de colunas na cláusula CREATE OR REPLACE VIEW são listados na mesma ordem que as colunas na subconsulta.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Modificando uma View

A opção OR REPLACE permite a criação de uma view mesmo que já exista outra com o mesmo nome, substituindo a versão antiga pela versão de seu respectivo proprietário. Isso significa que é possível alterar a view sem eliminar, recriar e conceder novamente privilégios de objeto com grant.

Observação: Ao designar apelidos de colunas na cláusula CREATE OR REPLACE VIEW, lembre-se de que os apelidos são listados na mesma ordem que as colunas na subconsulta.

Criando uma View Complexa

Crie uma view complexa que contenha functions de grupo para exibir valores de duas tabelas:

```
CREATE VIEW dept_sum_vu
(name, minsal, maxsal, avgsal)
AS SELECT      d.department_name, MIN(e.salary),
              MAX(e.salary), AVG(e.salary)
FROM      employees e, departments d
WHERE      e.department_id = d.department_id
GROUP BY    d.department_name;
View created.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Criando uma View Complexa

O exemplo do slide cria uma view complexa de nomes de departamentos, salários mínimos, salários máximos e salários médios por departamento. Observe que foram especificados nomes alternativos para a view. Esse é um requisito quando alguma coluna da view é derivada de uma function ou de uma expressão.



Você pode exibir a estrutura da view usando o comando *iSQL*Plus* DESCRIBE. Para exibir o conteúdo da view, execute a instrução SELECT.

```
SELECT *
FROM    dept_sum_vu;
```

NAME	MINSAL	MAXSAL	AVGSAL
Accounting	8300	12000	10150
Administration	4400	4400	4400
Executive	17000	24000	19333.3333
IT	4200	9000	6400
Marketing	6000	13000	9500
Sales	8600	11000	10033.3333
Shipping	2500	5800	3500

7 rows selected.

Regras para Executar Operações DML em uma View

- Em geral, é possível executar operações DML em views simples. 
- Você não poderá remover uma linha se a view contiver:
 - Functions de grupo
 - Uma cláusula GROUP BY
 - A palavra-chave DISTINCT
 - A palavra-chave da pseudocoluna ROWNUM

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Executando Operações DML em uma View

Você poderá executar operações DML em dados por meio de uma view se essas operações seguirem certas regras.

É possível remover uma linha de uma view, a menos que ela contenha:

- Functions de grupo
- Uma cláusula GROUP BY
- A palavra-chave DISTINCT
- A palavra-chave da pseudocoluna ROWNUM

Regras para Executar Operações DML em uma View

Você não poderá modificar dados de uma view se ela contiver:

- **Functions de grupo**
- **Uma cláusula GROUP BY**
- **A palavra-chave DISTINCT**
- **A palavra-chave da pseudocoluna ROWNUM**
- **Colunas definidas por expressões**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Executando Operações DML em uma View (continuação)

Você poderá modificar dados por meio de uma view, a menos que ela contenha uma das condições mencionadas no slide anterior ou inclua colunas definidas por expressões (por exemplo, `SALARY * 12`).

Regras para Executar Operações DML em uma View

Você não poderá adicionar dados por meio de uma view se ela contiver:

- **Functions de grupo**
- **Uma cláusula GROUP BY**
- **A palavra-chave DISTINCT**
- **A palavra-chave da pseudocoluna ROWNUM**
- **Colunas definidas por expressões**
- **Colunas NOT NULL nas tabelas-base que não estejam selecionadas pela view**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Executando Operações DML em uma View (continuação)

Você poderá adicionar dados por meio de uma view, a menos que ela contenha um dos itens listados no slide. Não será possível adicionar dados a uma view se ela contiver colunas NOT NULL sem valores default na tabela-base. Todos os valores necessários devem estar na view. Lembre-se de que você está adicionando valores diretamente à tabela subjacente *por meio* da view.

Para obter mais informações, consulte "CREATE VIEW" no manual *Oracle SQL Reference*.

Usando a Cláusula WITH CHECK OPTION

- Você pode garantir que as operações DML executadas na view se restrinjam ao domínio dessa view usando a cláusula WITH CHECK OPTION:

```
CREATE OR REPLACE VIEW empvu20
AS SELECT      *
   FROM        employees
   WHERE       department_id = 20
   WITH CHECK OPTION CONSTRAINT empvu20_ck ;
View created.
```

- Qualquer tentativa de alterar o número do departamento relativo a uma linha da view não terá êxito porque violará a constraint WITH CHECK OPTION.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Usando a Cláusula WITH CHECK OPTION

Você pode executar verificações de integridade referencial por meio de views. Pode também impor constraints no nível do banco de dados. É possível usar a view para proteger a integridade dos dados, mas o uso é muito limitado.

A cláusula WITH CHECK OPTION especifica que as instruções INSERT e UPDATE executadas por meio da view não podem criar linhas que a view não pode selecionar. Assim, ela permite a imposição de constraints de integridade e verificações de validação nos dados que estão sendo inseridos ou atualizados. Se houver uma tentativa de execução de operações DML em linhas não selecionadas pela view, será exibido um erro com o nome da constraint, caso ele tenha sido especificado.

```
UPDATE empvu20
SET department_id = 10
WHERE employee_id = 201;
```

causa:

```
ERROR at line 1:
ORA-01402: view WITH CHECK OPTION where-clause violation
```

Observação: Nenhuma linha será atualizada porque, se for necessário alterar o número do departamento para 10, a view não poderá mais ver esse funcionário. Portanto, com a cláusula WITH CHECK OPTION, a view só poderá ver os funcionários do departamento 20 e não permitirá que o número de departamento desses funcionários seja alterado.

Negando Operações DML

- Para garantir que não ocorram operações DML, adicione a opção `WITH READ ONLY` à definição da view.
- Qualquer tentativa de executar uma operação DML nas linhas da view resultará em erro do servidor Oracle.



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Negando Operações DML

Para garantir que não ocorram operações DML em uma view, crie essa view com a opção `WITH READ ONLY`. O exemplo do próximo slide modifica a view `EMPVU10` para impedir a execução de operações DML nessa view.

Negando Operações DML

```
CREATE OR REPLACE VIEW empvu10
  (employee_number, employee_name, job_title)
AS SELECT      employee_id, last_name, job_id
  FROM      employees
  WHERE      department_id = 10
  WITH READ ONLY ;
View created.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Negando Operações DML (continuação)

Qualquer tentativa de remover uma linha de uma view com uma constraint somente leitura resultará em um erro:

```
DELETE FROM empvu10
WHERE employee_number = 200;
DELETE FROM empvu10
      *
ERROR at line 1:
ORA-01752: cannot delete from view without exactly one
key-preserved table
```

Qualquer tentativa de inserir ou modificar uma linha usando a view com uma constraint somente leitura resultará em um erro do servidor Oracle:

```
01733: virtual column not allowed here.
```

Removendo uma View

Você pode remover uma view sem perder dados, pois uma view é baseada em tabelas subjacentes do banco de dados.

```
DROP VIEW view;
```

```
DROP VIEW empvu80;  
View dropped.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Removendo uma View

Use a instrução `DROP VIEW` para remover uma view. A instrução remove a definição da view do banco de dados. A eliminação de uma view não tem efeito sobre as tabelas nas quais a view foi baseada. As views ou outras aplicações baseadas em views deletadas tornam-se inválidas. Apenas o autor ou um usuário com o privilégio `DROP ANY VIEW` pode remover uma view.

Na sintaxe:

view é o nome da view

Exercício 10: Visão Geral da Parte 1

Este exercício aborda os seguintes tópicos:

- Criando uma view simples
- Criando uma view complexa
- Criando uma view com uma constraint de verificação
- Tentando modificar dados da view
- Removendo views

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Exercício 10: Visão Geral da Parte 1

A Parte 1 do exercício desta lição contém várias atividades que permitem criar, utilizar e remover views.

Faça as questões de 1 a 6 no final desta lição.

Seqüências

Objeto	Descrição
Tabela	Unidade básica de armazenamento; composta de linhas
View	Representa logicamente subconjuntos de dados de uma ou mais tabelas
Seqüência	Gera valores numéricos
Índice	Melhora o desempenho de algumas consultas
Sinônimo	Fornece nomes alternativos a objetos

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

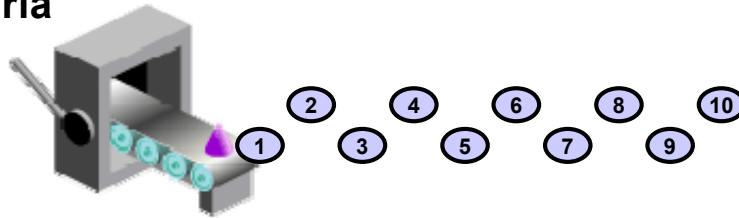
Seqüências

Uma seqüência é um objeto de banco de dados que cria valores inteiros. Você pode criar seqüências e utilizá-las para gerar números.

Seqüências

Uma seqüência:

- Pode gerar números exclusivos automaticamente
- É um objeto compartilhável
- Pode ser usada para criar um valor de chave primária
- Substitui o código da aplicação
- Aumenta a eficiência do acesso a valores de seqüência quando armazenados no cache da memória



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Seqüências

Uma seqüência é um objeto de banco de dados criado por um usuário que pode ser compartilhado por vários usuários para gerar inteiros.

Você pode definir uma seqüência para gerar valores exclusivos ou para reciclar e reutilizar os mesmos números.

As seqüências são muito usadas para criar um valor de chave primária, que precisa ser exclusivo para cada linha. A seqüência é gerada e incrementada (ou decrementada) por uma rotina interna do Oracle. Esse objeto pode poupar tempo reduzindo o tamanho do código da aplicação necessário para criar uma rotina de geração de seqüência.

Os números de seqüência são armazenados e gerados independentemente de tabelas. Portanto, é possível usar a mesma seqüência para várias tabelas.

Instrução CREATE SEQUENCE:

Sintaxe

Defina uma seqüência para gerar números seqüenciais automaticamente:

```
CREATE SEQUENCE sequence
  [INCREMENT BY n]
  [START WITH n]
  [{MAXVALUE n | NOMAXVALUE}]
  [{MINVALUE n | NOMINVALUE}]
  [{CYCLE | NOCYCLE}]
  [{CACHE n | NOCACHE}];
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Criando uma Seqüência

Gere automaticamente números seqüenciais usando a instrução CREATE SEQUENCE.

Na sintaxe:

<i>sequence</i>	é o nome do gerador de seqüência
INCREMENT BY <i>n</i>	especifica o intervalo entre os números da seqüência, em que <i>n</i> é um inteiro (Se esta cláusula for omitida, a seqüência será incrementada em 1.)
START WITH <i>n</i>	especifica o primeiro número da seqüência a ser gerado (Se esta cláusula for omitida, a seqüência começará com 1.)
MAXVALUE <i>n</i>	especifica o valor máximo que a seqüência pode gerar
NOMAXVALUE	especifica o valor máximo 10^{27} para uma seqüência em ordem crescente e -1 para uma seqüência em ordem decrescente (Esta é a opção default.)
MINVALUE <i>n</i>	especifica o valor mínimo da seqüência
NOMINVALUE	especifica o valor mínimo 1 para uma seqüência em ordem crescente e $-(10^{26})$ para uma seqüência em ordem decrescente (Esta é a opção default.)

Criando uma Seqüência

- Crie uma seqüência denominada DEPT_DEPTID_SEQ a ser usada para a chave primária da tabela DEPARTMENTS.
- Não use a opção CYCLE.

```
CREATE SEQUENCE dept_deptid_seq  
        INCREMENT BY 10  
        START WITH 120  
        MAXVALUE 9999  
        NOCACHE  
        NOCYCLE;
```

Sequence created.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Criando uma Seqüência (continuação)

CYCLE | NOCYCLE

especifica se a seqüência continuará a gerar valores depois de atingir seu valor máximo ou mínimo (NOCYCLE é a opção default.)

CACHE *n* | NOCACHE

especifica quantos valores o servidor Oracle pré-aloca e mantém na memória (Por default, o servidor Oracle armazena 20 valores no cache.)

O exemplo do slide cria uma seqüência denominada DEPT_DEPTID_SEQ a ser usada para a coluna DEPARTMENT_ID da tabela DEPARTMENTS. A seqüência começa em 120, não permite armazenamento em cache e não percorre um ciclo.

Não use a opção CYCLE se a seqüência for usada para gerar valores de chave primária, a menos que você tenha um mecanismo confiável que expurgue as linhas antigas mais rápido que os ciclos da seqüência.

Para obter mais informações, consulte "CREATE SEQUENCE" no manual *Oracle SQL Reference*.

Observação: A seqüência não é vinculada a uma tabela. Em geral, nomeie a seqüência de acordo com o uso pretendido. No entanto, é possível usar a seqüência em qualquer local, independentemente de seu nome.

Pseudocolunas NEXTVAL e CURRVAL

- **NEXTVAL** retorna o próximo valor disponível da seqüência. Quando referenciada, mesmo que por usuários distintos, ela retorna um valor exclusivo.
- **CURRVAL** obtém o valor atual da seqüência.
- **É necessário executar NEXTVAL para a seqüência antes que CURRVAL contenha um valor.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Pseudocolunas NEXTVAL e CURRVAL

Depois que você cria uma seqüência, ela gera números seqüenciais para uso nas suas tabelas. Faça referência aos valores da seqüência usando as pseudocolunas NEXTVAL e CURRVAL.

A pseudocoluna NEXTVAL é usada para extrair números sucessivos de uma seqüência especificada. Qualifique NEXTVAL com o nome da seqüência. Quando você fizer referência a *seqüência*.NEXTVAL, um novo número da seqüência será gerado e o número atual da seqüência será inserido em CURRVAL.

A pseudocoluna CURRVAL é usada para fazer referência a um número da seqüência que o usuário atual acabou de gerar. É preciso usar NEXTVAL para gerar um número de seqüência na sessão do usuário atual antes que seja possível fazer referência a CURRVAL. Qualifique CURRVAL com o nome da seqüência. Quando você fizer referência a *seqüência*.CURRVAL, o último valor retornado para o processo desse usuário será exibido.

Pseudocolunas NEXTVAL e CURRVAL (continuação)

Regras de Uso de NEXTVAL e CURRVAL

Você pode usar NEXTVAL e CURRVAL nos seguintes contextos:

- Lista SELECT de uma instrução SELECT que não faz parte de uma subconsulta
- Lista SELECT de uma subconsulta em uma instrução INSERT
- Cláusula VALUES de uma instrução INSERT
- Cláusula SET de uma instrução UPDATE

Você não pode usar NEXTVAL e CURRVAL nos seguintes contextos:

- Lista SELECT de uma view
- Instrução SELECT com a palavra-chave DISTINCT
- Instrução SELECT com as cláusulas GROUP BY, HAVING ou ORDER BY
- Subconsulta em uma instrução SELECT, DELETE ou UPDATE
- Expressão DEFAULT em uma instrução CREATE TABLE ou ALTER TABLE

Para obter mais informações, consulte "Pseudocolumns" e "CREATE SEQUENCE" no manual *Oracle SQL Reference*.

Usando uma Seqüência

- Insira um novo departamento denominado "Support" no ID de local 2500:

```
INSERT INTO departments(department_id,  
                        department_name, location_id)  
VALUES                (dept_deptid_seq.NEXTVAL,  
                      'Support', 2500);  
1 row created.
```

- Exiba o valor atual da seqüência DEPT_DEPTID_SEQ:

```
SELECT dept_deptid_seq.CURRVAL  
FROM    dual;
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Usando uma Seqüência

O exemplo do slide insere um novo departamento na tabela DEPARTMENTS. Ele usa a seqüência DEPT_DEPTID_SEQ para gerar um novo número de departamento como mostrado a seguir.

Você pode exibir o valor atual da seqüência:

```
SELECT dept_deptid_seq.CURRVAL  
FROM    dual;
```

CURRVAL	
	120

Agora, suponha que você queira admitir funcionários para a equipe do novo departamento. A instrução INSERT a ser executada para todos os novos funcionários pode incluir o seguinte código:

```
INSERT INTO employees (employee_id, department_id, ...)  
VALUES (employees_seq.NEXTVAL, dept_deptid_seq.CURRVAL, ...);
```

Observação: O exemplo anterior supõe que uma seqüência denominada EMPLOYEE_SEQ já tenha sido criada para gerar novos números de funcionários.

Armazenando Valores de Seqüência no Cache

- O armazenamento de valores da seqüência no cache da memória permite um acesso mais rápido a esses valores.
- Os intervalos em valores de seqüência ocorrem quando:
 - É efetuado um rollback
 - Ocorre uma falha do sistema
 - Uma seqüência é usada em outra tabela

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Armazenando Valores de Seqüência no Cache

É possível armazenar seqüências no cache da memória para permitir o acesso mais rápido aos valores da seqüência. O cache é preenchido na primeira vez em que você faz referência à seqüência. Cada solicitação do próximo valor da seqüência é recuperada da seqüência armazenada no cache. Depois que o último valor da seqüência é usado, a próxima solicitação dessa seqüência armazena outro cache de seqüências na memória.

Intervalos da Seqüência

Embora os geradores de seqüência gerem números seqüenciais sem intervalos, essa ação ocorre independentemente de um commit ou rollback. Portanto, se você executar rollback de uma instrução que contenha uma seqüência, o número será perdido.

Outro evento que pode causar intervalos na seqüência é uma falha do sistema. Se a seqüência armazenar valores no cache da memória, esses valores serão perdidos caso ocorra uma falha do sistema.

Como as seqüências não estão diretamente vinculadas a tabelas, é possível usar a mesma seqüência para várias tabelas. Se você fizer isso, cada tabela poderá conter intervalos nos números seqüenciais.

Modificando uma Seqüência

Altere o valor de incremento, o valor máximo, o valor mínimo, a opção de ciclo ou a opção de cache:

```
ALTER SEQUENCE dept_deptid_seq  
          INCREMENT BY 20  
          MAXVALUE 999999  
          NOCACHE  
          NOCYCLE;  
Sequence altered.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Modificando uma Seqüência

Se o limite MAXVALUE definido para a seqüência for atingido, não serão alocados outros valores da seqüência e você receberá um erro indicando que essa seqüência excede MAXVALUE. Para continuar a usar a seqüência, você poderá modificá-la usando a instrução ALTER SEQUENCE.

Sintaxe

```
ALTER SEQUENCE sequence  
    [INCREMENT BY n]  
    [{MAXVALUE n | NOMAXVALUE}]  
    [{MINVALUE n | NOMINVALUE}]  
    [{CYCLE | NOCYCLE}]  
    [{CACHE n | NOCACHE}];
```

Na sintaxe, *sequence* é o nome do gerador de seqüência.

Para obter mais informações, consulte "ALTER SEQUENCE" no manual *Oracle SQL Reference*.

Diretrizes para Modificar uma Seqüência

- Você precisa ser o proprietário da seqüência ou ter o privilégio ALTER.
- Somente os futuros números da seqüência são afetados.
- É necessário eliminar e recriar a seqüência para reiniciá-la em um número diferente.
- É executada uma validação.
- Para remover uma seqüência, use a instrução DROP:

```
DROP SEQUENCE dept_deptid_seq;  
Sequence dropped.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Diretrizes para Modificar uma Seqüência

- Você precisa ser o proprietário da seqüência ou ter o privilégio ALTER para modificá-la. Você precisa ser o proprietário da seqüência ou ter o privilégio DROP ANY SEQUENCE para removê-la.
- Somente os futuros números da seqüência serão afetados pela instrução ALTER SEQUENCE.
- Não é possível alterar a opção START WITH usando ALTER SEQUENCE. É necessário eliminar e recriar a seqüência para reiniciá-la em um número diferente.
- É executada uma validação. Por exemplo, não é possível impor um novo MAXVALUE menor que o número atual da seqüência.

```
ALTER SEQUENCE dept_deptid_seq  
    INCREMENT BY 20  
    MAXVALUE 90  
    NOCACHE  
    NOCYCLE;
```

```
ALTER SEQUENCE dept_deptid_seq  
*
```

ERROR at line 1:

ORA-04009: MAXVALUE cannot be made to be less than the
current value

Índices

Objeto	Descrição
Tabela	Unidade básica de armazenamento; composta de linhas
View	Representa logicamente subconjuntos de dados de uma ou mais tabelas
Seqüência	Gera valores numéricos
Índice	Melhora o desempenho de algumas consultas
Sinônimo	Fornece nomes alternativos a objetos

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

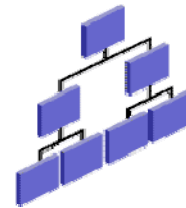
Índices

Os índices são objetos de banco de dados que você pode criar para melhorar o desempenho de algumas consultas. Eles também podem ser criados automaticamente pelo servidor quando você cria uma constraint exclusiva ou de chave primária.

Índices

Um índice:

- É um objeto de esquema
- É usado pelo servidor Oracle para acelerar a recuperação de linhas usando um ponteiro
- Pode reduzir a entrada/saída de disco por um método de acesso de caminho rápido para localizar dados rapidamente
- É independente da tabela que indexa
- É usado e mantido automaticamente pelo servidor Oracle



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Índices (continuação)

Um índice do servidor Oracle é um objeto de esquema que pode acelerar a recuperação de linhas usando um ponteiro. É possível criar índices de forma explícita ou automática. Se você não tiver um índice na coluna, ocorrerá uma varredura integral da tabela.

Um índice permite acesso direto e rápido a linhas de uma tabela. Seu objetivo é reduzir a necessidade de entrada/saída de disco usando um caminho indexado para localizar os dados com rapidez. O servidor Oracle usa e mantém automaticamente o índice. Depois que um índice é criado, não é exigida qualquer atividade direta do usuário.

Os índices são independentes do ponto de vista lógico e físico da tabela que indexam. Isso significa que é possível criá-los ou eliminá-los a qualquer momento sem causar efeitos nas tabelas-base ou em outros índices.

Observação: Quando você elimina uma tabela, os índices correspondentes também são eliminados.

Para obter mais informações, consulte "Schema Objects: Indexes" em *Database Concepts*.

Como Criar Índices?

- **Automaticamente:** Um índice exclusivo é criado automaticamente quando você define uma constraint de chave PRIMARY ou UNIQUE em uma definição de tabela.



- **Manualmente:** Os usuários podem criar índices não exclusivos em colunas para acelerar o acesso às linhas.



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Tipos de Índice

É possível criar dois tipos de índice.

Índice exclusivo: O servidor Oracle cria automaticamente este índice quando você define uma constraint de chave PRIMARY KEY ou UNIQUE para a coluna de uma tabela. O nome do índice é aquele fornecido à constraint.

Índice não exclusivo: É um índice que um usuário pode criar. Por exemplo, você pode criar um índice de coluna FOREIGN KEY para uma join em uma consulta a fim de aumentar a velocidade de recuperação.

Observação: Você pode criar um índice exclusivo de forma manual, mas é recomendável criar uma constraint exclusiva, que gera implicitamente um índice exclusivo.

Criando um Índice

- Crie um índice em uma ou mais colunas:

```
CREATE INDEX index
ON table (column[, column]...);
```

- Aumente a velocidade de acesso da consulta à coluna `LAST_NAME` da tabela `EMPLOYEES`:

```
CREATE INDEX emp_last_name_idx
ON employees(last_name);
Index created.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Criando um Índice

Crie um índice em uma ou mais colunas executando a instrução `CREATE INDEX`.

Na sintaxe:

<i>index</i>	é o nome do índice
<i>table</i>	é o nome da tabela
<i>column</i>	é o nome da coluna da tabela a ser indexada

Para obter mais informações, consulte "CREATE INDEX" no manual *Oracle SQL Reference*.

Diretrizes para a Criação de um Índice

Crie um índice quando:	
<input checked="" type="checkbox"/>	Uma coluna contiver uma grande faixa de valores
<input checked="" type="checkbox"/>	Uma coluna contiver um grande número de valores nulos
<input checked="" type="checkbox"/>	Uma ou mais colunas forem usadas em conjunto com frequência em uma cláusula WHERE ou em uma condição de join
<input checked="" type="checkbox"/>	A tabela for grande e for esperado que a maioria das consultas recupere menos de 2% a 4% das linhas da tabela
Não crie um índice quando:	
<input checked="" type="checkbox"/>	As colunas não forem usadas com frequência como uma condição na consulta
<input checked="" type="checkbox"/>	A tabela for pequena ou for esperado que a maioria das consultas recupere mais de 2% a 4% das linhas da tabela
<input checked="" type="checkbox"/>	A tabela for atualizada com frequência
<input checked="" type="checkbox"/>	As colunas indexadas forem referenciadas como parte de uma expressão

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Mais Não Significa Melhor

Um número maior de índices em uma tabela não resulta em consultas mais rápidas. Para cada operação DML submetida a commit em uma tabela com índices, é necessário atualizar os índices. Quanto mais índices estiverem associados a uma tabela, maior será o esforço do servidor Oracle para atualizar todos os índices após uma operação DML.

Quando Criar um Índice

Portanto, você só deverá criar índices se:

- A coluna contiver uma grande faixa de valores
- A coluna contiver um grande número de valores nulos
- Uma ou mais colunas forem usadas em conjunto com frequência em uma cláusula WHERE ou em uma condição de join
- A tabela for grande e for esperado que a maioria das consultas recupere menos de 2% a 4% das linhas

Lembre-se de que, para impor a exclusividade, você deverá definir uma constraint exclusiva na definição da tabela. Depois, um índice exclusivo será criado automaticamente.

Removendo um Índice

- Para remover um índice do dicionário de dados, use o comando **DROP INDEX**:

```
DROP INDEX index;
```

- Remova o índice **UPPER_LAST_NAME_IDX** do dicionário de dados:

```
DROP INDEX emp_last_name_idx;  
Index dropped.
```

- Para eliminar um índice, você precisa ser o proprietário dele ou ter o privilégio **DROP ANY INDEX**.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Removendo um Índice

Não é possível modificar índices. Para alterar um índice, elimine-o e, depois, recrie-o.

Remova uma definição de índice do dicionário de dados executando a instrução **DROP INDEX**. Para eliminar um índice, você precisa ser o proprietário dele ou ter o privilégio **DROP ANY INDEX**.

Na sintaxe, *index* é o nome do índice.

Observação: Se você eliminar uma tabela, os índices e as constraints serão eliminados automaticamente, mas as views e as seqüências permanecerão.

Sinônimos

Objeto	Descrição
Tabela	Unidade básica de armazenamento; composta de linhas
View	Representa logicamente subconjuntos de dados de uma ou mais tabelas
Seqüência	Gera valores numéricos
Índice	Melhora o desempenho de algumas consultas
Sinônimo	Fornece nomes alternativos a objetos

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Sinônimos

Os sinônimos são objetos de banco de dados que permitem chamar uma tabela usando outro nome. É possível criar sinônimos para atribuir um nome alternativo a uma tabela.

Sinônimos

Simplifique o acesso a objetos criando um sinônimo (outro nome para um objeto). Com sinônimos, você pode:

- Criar uma referência mais fácil a uma tabela pertencente a outro usuário
- Reduzir nomes longos de objetos

```
CREATE [PUBLIC] SYNONYM synonym  
FOR object;
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Criando um Sinônimo para um Objeto

Para fazer referência a uma tabela pertencente a outro usuário, é preciso inserir o nome do autor dessa tabela seguido de um ponto como prefixo do nome da tabela. A criação de um sinônimo elimina a necessidade de qualificar o nome do objeto com o esquema e fornece um nome alternativo para uma tabela, uma view, uma sequência, um procedure ou outros objetos. Esse método pode ser útil especialmente com nomes longos de objetos, como views.

Na sintaxe:

<code>PUBLIC</code>	cria um sinônimo acessível a todos os usuários
<code><i>synonym</i></code>	é o nome do sinônimo a ser criado
<code><i>object</i></code>	identifica o objeto para o qual o sinônimo é criado

Diretrizes

- O objeto não pode estar contido em um package.
- O nome de um sinônimo privado deve ser diferente do de outros objetos pertencentes ao mesmo usuário.

Para obter mais informações, consulte "CREATE SYNONYM" no manual *Oracle SQL Reference*.

Criando e Removendo Sinônimos

- Crie um nome abreviado para a view DEPT_SUM_VU:

```
CREATE SYNONYM d_sum  
FOR dept_sum_vu;  
Synonym Created.
```

- Elimine um sinônimo:

```
DROP SYNONYM d_sum;  
Synonym dropped.
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Criando um Sinônimo

O exemplo do slide cria um sinônimo relativo à view DEPT_SUM_VU para agilizar a referência.

O administrador do banco de dados pode criar um sinônimo público acessível a todos os usuários. Este exemplo cria um sinônimo público denominado DEPT para a tabela DEPARTMENTS da usuária Alice:

```
CREATE PUBLIC SYNONYM dept  
FOR alice.departments;  
Synonym created.
```

Removendo um Sinônimo

Para remover um sinônimo, use a instrução DROP SYNONYM. Somente o administrador do banco de dados pode eliminar um sinônimo público.

```
DROP PUBLIC SYNONYM dept;  
Synonym dropped.
```

Para obter mais informações, consulte "DROP SYNONYM" no manual *Oracle SQL Reference*.

Sumário

Nesta lição, você aprendeu a:

- **Criar, usar e remover views**
- **Gerar automaticamente números seqüenciais usando um gerador de seqüência**
- **Criar índices para aumentar a velocidade de recuperação de consultas**
- **Usar sinônimos para fornecer nomes alternativos a objetos**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Sumário

Nesta lição, você conheceu objetos de banco de dados como views, seqüências, índices e sinônimos.

Exercício 10: Visão Geral da Parte 2

Este exercício aborda os seguintes tópicos:

- **Criando seqüências**
- **Usando seqüências**
- **Criando índices não exclusivos**
- **Criando sinônimos**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Exercício 10: Visão Geral da Parte 2

A Parte 2 do exercício desta lição contém várias atividades que permitem criar e usar uma seqüência, um índice e um sinônimo.

Faça as questões de 7 a 10 no final desta lição.

Exercício 10

Parte 1

1. A equipe do departamento de recursos humanos deseja ocultar alguns dados da tabela EMPLOYEES. Ela deseja uma view denominada EMPLOYEES_VU com base nos números e nomes de funcionário, bem como nos números de departamento da tabela EMPLOYEES. Também deseja atribuir EMPLOYEE como o cabeçalho do nome do funcionário.
2. Verifique se a view funciona. Exiba o conteúdo da view EMPLOYEES_VU.

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
107	Lorentz	60
...		
206	Gietz	110

20 rows selected.

3. Usando a view EMPLOYEES_VU, crie uma consulta para o departamento de recursos humanos a fim de exibir todos os nomes de funcionário e números de departamento.

EMPLOYEE	DEPARTMENT_ID
King	90
Kochhar	90
...	
Gietz	110

20 rows selected.

Exercício 10

- O departamento 50 precisa de acesso aos dados de seus funcionários. Crie uma view denominada DEPT50 que contenha os números e os sobrenomes, bem como os números de departamento de todos os funcionários do departamento 50. Foi solicitada a atribuição dos labels EMPNO, EMPLOYEE e DEPTNO às colunas da view. Para fins de segurança, não permita que um funcionário seja redesignado a outro departamento por meio da view.
- Exiba a estrutura e o conteúdo da view DEPT50.

Name	Null?	Type
EMPNO	NOT NULL	NUMBER(6)
EMPLOYEE	NOT NULL	VARCHAR2(25)
DEPTNO		NUMBER(4)

EMPNO	EMPLOYEE	DEPTNO
124	Mourgos	50
141	Rajs	50
142	Davies	50
143	Matos	50
144	Vargas	50

- Teste a view. Tente redesignar Matos ao departamento 80.

Exercício 10

Parte 2

7. Você precisa de uma seqüência que possa ser usada com a coluna de chave primária da tabela DEPT. A seqüência deve começar com o valor 200 e ter o valor máximo 1.000. Incremente a seqüência em 10. Nomeie-a como DEPT_ID_SEQ.
8. Para testar a seqüência, crie um script para inserir duas linhas na tabela DEPT. Nomeie o script como lab_10_08.sql. Certifique-se de utilizar a seqüência criada para a coluna ID. Adicione dois departamentos: Education e Administration. Confirme as adições. Execute os comandos no script.
9. Crie um índice não exclusivo na coluna DEPT_ID da tabela DEPT.
10. Crie um sinônimo para a tabela EMPLOYEES. Nomeie o sinônimo como EMP.

11

Gerenciando Objetos com Views de Dicionário de Dados

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Ao concluir esta lição, você será capaz de:

- **Usar as views de dicionário de dados para pesquisar dados sobre objetos**
- **Consultar diversas views de dicionário de dados**

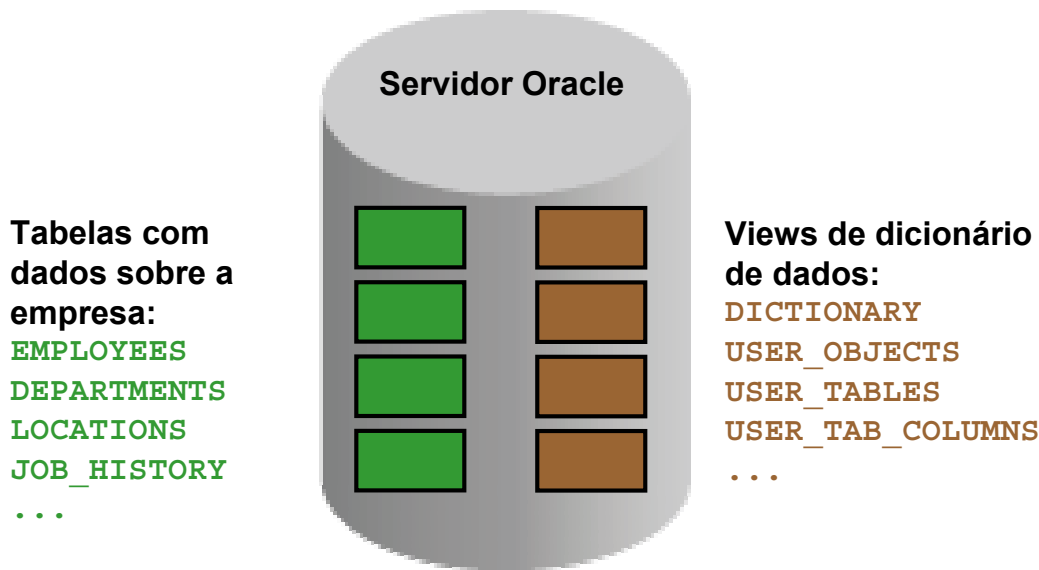
ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Esta lição apresenta as views de dicionário de dados. Você aprenderá que é possível usar as views de dicionário para recuperar metadados e criar relatórios sobre objetos de esquema.

O Dicionário de Dados



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

O Dicionário de Dados

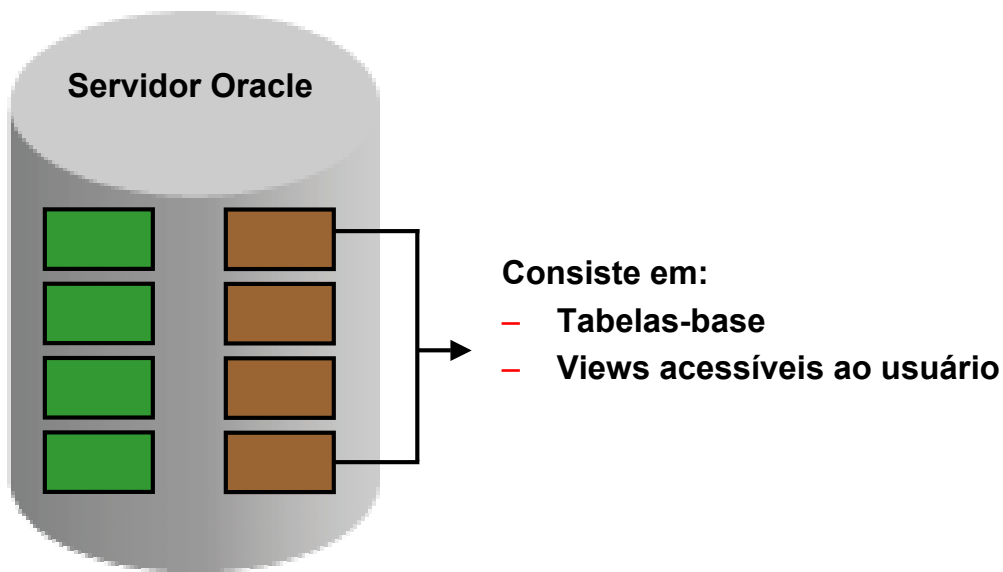
As tabelas de usuário são criadas pelo usuário e contêm dados sobre a empresa, como EMPLOYEES. Existe um outro conjunto de tabelas e views no banco de dados Oracle conhecido como *dicionário de dados*. Esse conjunto é criado e mantido pelo servidor Oracle e contém informações sobre o banco de dados. O dicionário de dados é estruturado em tabelas e views, da mesma forma que outros dados do banco de dados. Além de ser essencial para todos os bancos de dados Oracle, o dicionário de dados é uma importante ferramenta para todos os usuários, desde usuários finais a designers de aplicações e administradores de bancos de dados.

Para acessar o dicionário de dados, use instruções SQL. Como o dicionário de dados é somente para leitura, é possível executar apenas consultas às respectivas tabelas e views.

Você pode consultar as views de dicionário baseadas nas tabelas de dicionário para localizar informações como:

- Definições de todos os objetos de esquema do banco de dados (tabelas, views, índices, sinônimos, seqüências, procedures, functions, packages, triggers e outros)
- Valores default de colunas
- Informações sobre constraints de integridade
- Nomes dos usuários Oracle
- Privilégios e atribuições concedidos a cada usuário
- Outras informações gerais sobre o banco de dados

Estrutura do Dicionário de Dados



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Estrutura do Dicionário de Dados

As tabelas-base subjacentes armazenam informações sobre o banco de dados associado. Apenas o servidor Oracle pode executar operações de gravação e leitura nessas tabelas. Você raramente terá acesso direto a elas.

Várias views resumam e exibem as informações armazenadas nas tabelas-base do dicionário de dados. Essas views decodificam os dados das tabelas-base e os apresentam como informações úteis (como nomes de tabelas ou usuários) usando joins e cláusulas WHERE para simplificar as informações. A maioria dos usuários obtém acesso às views, e não às tabelas-base.

O usuário Oracle SYS é proprietário de todas as tabelas-base e views acessíveis ao usuário do dicionário de dados. Nenhum usuário Oracle poderá alterar (UPDATE, DELETE ou INSERT) linhas ou objetos de esquema contidos no esquema SYS, pois essa atividade poderá comprometer a integridade dos dados.

Estrutura do Dicionário de Dados

Convenção de nomeação de views:

Prefixo da View	Objetivo
USER	View do usuário (o conteúdo do seu esquema e pertencente a você)
ALL	View expandida do usuário (o que você pode acessar)
DBA	View do administrador do banco de dados (o conteúdo dos esquemas de
V\$	Dados relacionados ao desempenho

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Estrutura do Dicionário de Dados (continuação)

O dicionário de dados consiste em conjuntos de views. Em vários casos, um conjunto é formado por três views com informações semelhantes e diferenciadas por prefixos. Por exemplo, há uma view denominada USER_OBJECTS, outra denominada ALL_OBJECTS e uma terceira denominada DBA_OBJECTS.

Essas três views contêm informações semelhantes sobre objetos do banco de dados, com exceção do escopo, que é diferente. USER_OBJECTS contém informações sobre objetos de sua propriedade ou que você criou. ALL_OBJECTS contém informações sobre todos os objetos aos quais você tem acesso. DBA_OBJECTS contém informações sobre todos os objetos pertencentes a todos os usuários. Em geral, as views com o prefixo ALL ou DBA contêm uma coluna adicional, denominada OWNER, que identifica o proprietário do objeto.

Há também um conjunto de views com o prefixo V\$. Essas views são dinâmicas por natureza e contêm informações sobre desempenho. As tabelas dinâmicas de desempenho não são tabelas de verdade e não podem ser acessadas pela maioria dos usuários. No entanto, os administradores de banco de dados podem consultar e criar views nas tabelas e, com o comando grant, conceder acesso a essas views a outros usuários. Este curso não aborda essas views detalhadamente.

Como Usar as Views de Dicionário

Comece com DICTONARY. Ela contém os nomes e as descrições das tabelas e views de dicionário.

```
DESCRIBE DICTONARY
```

Name	Null?	Type
TABLE_NAME		VARCHAR2(30)
COMMENTS		VARCHAR2(4000)

```
SELECT *  
FROM dictionary  
WHERE table_name = 'USER_OBJECTS';
```

TABLE_NAME	COMMENTS
USER_OBJECTS	Objects owned by the user

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Como Usar as Views de Dicionário

Para se familiarizar com as views de dicionário, você pode usar a view de dicionário DICTONARY. Ela contém o nome e a descrição resumida de cada view de dicionário à qual você tem acesso.

Você pode criar consultas para procurar informações sobre um nome de view específico ou pesquisar uma palavra ou expressão na coluna COMMENTS. No exemplo mostrado, a view DICTONARY é descrita. Ela tem duas colunas. A instrução SELECT recupera informações sobre a view de dicionário USER_OBJECTS. Essa view contém informações sobre todos os objetos pertencentes a você.

É possível criar consultas para pesquisar uma palavra ou expressão na coluna de comentários. Por exemplo, a consulta a seguir retorna os nomes de todas as views às quais você tem acesso que contêm a palavra *columns* na coluna COMMENT:

```
SELECT table_name  
FROM dictionary  
WHERE LOWER(comments) LIKE '%columns';
```

Observação: Os nomes no dicionário de dados são exibidos em letras maiúsculas.

View USER_OBJECTS

- Descreve todos os objetos pertencentes a você
- É uma forma prática de obter uma listagem de todos os nomes e tipos de objetos do seu esquema, além das seguintes informações:
 - Data de criação
 - Data da última modificação
 - Status (válido ou inválido)
- Consultas **ALL_OBJECTS** para exibir todos os objetos aos quais você tem acesso

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

View USER_OBJECTS

Você pode consultar a view **USER_OBJECTS** para ver os nomes e os tipos de todos os objetos do seu esquema. Essa view contém várias colunas:

- **OBJECT_NAME**: Nome do objeto
- **OBJECT_ID**: Número do objeto de dicionário
- **OBJECT_TYPE**: Tipo de objeto (como TABLE, VIEW, INDEX, SEQUENCE)
- **CREATED**: Timestamp da criação do objeto
- **LAST_DDL_TIME**: Timestamp da última modificação do objeto resultante de um comando DDL
- **STATUS**: Status do objeto (VALID, INVALID ou N/A)
- **GENERATED**: O nome deste objeto foi gerado pelo sistema? (Y | N)

Observação: Nem todas as colunas foram relacionadas nessa listagem. Para obter uma listagem completa, consulte "USER_OBJECTS" no manual *Oracle Database Reference*.

Você também pode consultar a view **ALL_OBJECTS** para obter uma listagem de todos os objetos aos quais tem acesso.

View USER_OBJECTS

```
SELECT object_name, object_type, created, status
FROM   user_objects
ORDER BY object_type;
```

OBJECT_NAME	OBJECT_TYPE	CREATED	STATUS
REG_ID_PK	INDEX	10-DEC-03	VALID
...			
DEPARTMENTS_SEQ	SEQUENCE	10-DEC-03	VALID
REGIONS	TABLE	10-DEC-03	VALID
LOCATIONS	TABLE	10-DEC-03	VALID
DEPARTMENTS	TABLE	10-DEC-03	VALID
JOB_HISTORY	TABLE	10-DEC-03	VALID
JOB_GRADES	TABLE	10-DEC-03	VALID
EMPLOYEES	TABLE	10-DEC-03	VALID
JOBS	TABLE	10-DEC-03	VALID
COUNTRIES	TABLE	10-DEC-03	VALID
EMP_DETAILS_VIEW	VIEW	10-DEC-03	VALID

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

View USER_OBJECTS (continuação)

O exemplo mostra os nomes, os tipos, as datas de criação e o status de todos os objetos pertencentes ao usuário.

A coluna OBJECT_TYPE contém o valor TABLE, VIEW, SEQUENCE, INDEX, PROCEDURE, FUNCTION, PACKAGE ou TRIGGER.

A coluna STATUS contém o valor VALID, INVALID ou N/A. Embora as tabelas sejam sempre válidas, as views, os procedures, as functions, os packages e os triggers podem ser inválidos.

A View CAT

Para consultas e informações de saída simplificadas, é possível consultar a view CAT. Essa view contém apenas duas colunas: TABLE_NAME e TABLE_TYPE. Ela fornece os nomes de todos os objetos INDEX, TABLE, CLUSTER, VIEW, SYNONYM, SEQUENCE ou UNDEFINED.

Informações sobre Tabelas

USER_TABLES:

```
DESCRIBE user_tables
```

Name	Null?	Type
TABLE_NAME	NOT NULL	VARCHAR2(30)
TABLESPACE_NAME		VARCHAR2(30)
CLUSTER_NAME		VARCHAR2(30)
IOT_NAME		VARCHAR2(30)

```
SELECT table_name  
FROM user_tables;
```

TABLE_NAME
JOB_GRADES
REGIONS
COUNTRIES
LOCATIONS
DEPARTMENTS
...

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

View USER_TABLES

Você pode usar a view USER_TABLES para obter os nomes de todas as suas tabelas. Essa view contém informações sobre as suas tabelas. Além do nome da tabela, ela contém informações detalhadas sobre o local de armazenamento.

Observação: Para obter uma listagem completa das colunas na view USER_TABLES, consulte "USER_TABLES" no manual *Oracle Database Reference*.

Você também pode consultar as views ALL_TABLES e TABS para exibir uma listagem de todas as tabelas às quais tem acesso:

```
SELECT table_name  
FROM tabs;
```

Informações sobre Colunas

USER_TAB_COLUMNS:

```
DESCRIBE user_tab_columns
```

Name	Null?	Type
TABLE_NAME	NOT NULL	VARCHAR2(30)
COLUMN_NAME	NOT NULL	VARCHAR2(30)
DATA_TYPE		VARCHAR2(106)
DATA_TYPE_MOD		VARCHAR2(3)
DATA_TYPE_OWNER		VARCHAR2(30)
DATA_LENGTH	NOT NULL	NUMBER
DATA_PRECISION		NUMBER
DATA_SCALE		NUMBER
NULLABLE		VARCHAR2(1)
COLUMN_ID		NUMBER
DEFAULT_LENGTH		NUMBER
DATA_DEFAULT		LONG

...

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Informações sobre Colunas

Você pode consultar a view `USER_TAB_COLUMNS` para obter informações detalhadas sobre as colunas das suas tabelas. Enquanto a view `USER_TABLES` fornece informações sobre o nome e o local de armazenamento das tabelas, a view `USER_TAB_COLUMNS` contém informações detalhadas sobre colunas.

A view contém as seguintes informações:

- Nomes de colunas
- Tipos de dados da coluna
- Tamanho dos tipos de dados
- Precisão e escala das colunas `NUMBER`
- Se são permitidos valores nulos (Há uma constraint `NOT NULL` na coluna?)
- Valor default

Observação: Para obter uma listagem completa e a descrição das colunas na view `USER_TAB_COLUMNS`, consulte "`USER_TAB_COLUMNS`" no manual *Oracle Database Reference*.

Informações sobre Colunas

```
SELECT column_name, data_type, data_length,  
       data_precision, data_scale, nullable  
FROM   user_tab_columns  
WHERE  table_name = 'EMPLOYEES';
```

COLUMN_NAME	DATA_TYPE	DATA_LENGTH	DATA_PRECISION	DATA_SCALE	NUL
EMPLOYEE_ID	NUMBER	22	6	0	N
FIRST_NAME	VARCHAR2	20			Y
LAST_NAME	VARCHAR2	25			N
EMAIL	VARCHAR2	25			N
PHONE_NUMBER	VARCHAR2	20			Y
HIRE_DATE	DATE	7			N
JOB_ID	VARCHAR2	10			N
SALARY	NUMBER	22	8	2	Y
COMMISSION_PCT	NUMBER	22	2	2	Y
MANAGER_ID	NUMBER	22	6	0	Y
DEPARTMENT_ID	NUMBER	22	4	0	Y

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Informações sobre Colunas (continuação)

A consulta à tabela USER_TAB_COLUMNS permite localizar detalhes sobre colunas, como nomes, tipos de dados, tamanhos dos tipos de dados, constraints nulas e valores default.

O exemplo mostra as colunas, os tipos de dados, os tamanhos dos dados e as constraints nulas da tabela EMPLOYEES. Observe que essas informações são semelhantes à saída do comando *iSQL*Plus* DESCRIBE.

Informações sobre Constraints

- **USER_CONSTRAINTS** descreve as definições de constraints nas tabelas.
- **USER_CONS_COLUMNS** descreve as colunas pertencentes a você e especificadas nas constraints.

```
DESCRIBE user_constraints
```

Name	Null?	Type
OWNER	NOT NULL	VARCHAR2(30)
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)
CONSTRAINT_TYPE		VARCHAR2(1)
TABLE_NAME	NOT NULL	VARCHAR2(30)
SEARCH_CONDITION		LONG
R_OWNER		VARCHAR2(30)
R_CONSTRAINT_NAME		VARCHAR2(30)
DELETE_RULE		VARCHAR2(9)
STATUS		VARCHAR2(8)

...

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Informações sobre Constraints

Você pode obter os nomes das constraints, o tipo de constraint, o nome da tabela à qual se aplica a constraint, a condição das constraints de verificação, informações sobre constraints de chave estrangeira, a regra de deleção das constraints de chave estrangeira, o status e vários outros tipos de informações sobre constraints.

Observação: Para obter uma listagem completa e a descrição das colunas na view **USER_CONSTRAINTS**, consulte "USER_CONSTRAINTS" no manual *Oracle Database Reference*.

Informações sobre Constraints

```
SELECT constraint_name, constraint_type,
       search_condition, r_constraint_name,
       delete_rule, status
FROM   user_constraints
WHERE  table_name = 'EMPLOYEES';
```

CONSTRAINT_NAME	CON	SEARCH_CONDITION	R_CONSTRAINT_NAME	DELETE_RULE	STATUS
EMP_LAST_NAME_NN	C	"LAST_NAME" IS NOT NULL			ENABLED
EMP_EMAIL_NN	C	"EMAIL" IS NOT NULL			ENABLED
EMP_HIRE_DATE_NN	C	"HIRE_DATE" IS NOT NULL			ENABLED
EMP_JOB_NN	C	"JOB_ID" IS NOT NULL			ENABLED
EMP_SALARY_MIN	C	salary > 0			ENABLED
EMP_EMAIL_UK	U				ENABLED
EMP_EMP_ID_PK	P				ENABLED
EMP_DEPT_FK	R		DEPT_ID_PK	NO ACTION	ENABLED
EMP_JOB_FK	R		JOB_ID_PK	NO ACTION	ENABLED
EMP_MANAGER_FK	R		EMP_EMP_ID_PK	NO ACTION	ENABLED

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

USER_CONSTRAINTS: Exemplo

No exemplo mostrado, a view USER_CONSTRAINTS é consultada para localizar os nomes, os tipos, as condições de verificação, o nome da constraint exclusiva à qual a chave estrangeira faz referência, a regra de deleção de uma chave estrangeira e o status das constraints na tabela EMPLOYEES.

CONSTRAINT_TYPE pode ser:

- C (constraint de verificação em uma tabela)
- P (chave primária)
- U (chave exclusiva)
- R (integridade referencial)
- V (com opção de verificação, em uma view)
- O (somente para leitura, em uma view)

DELETE_RULE pode ser:

- CASCADE: Se o registro pai for deletado, os registros filhos também o serão.
- NO ACTION: Um registro pai só poderá ser deletado se não houver registros filhos.

STATUS pode ser:

- ENABLED: A constraint está ativa.
- DISABLED: A constraint é desativada.

Informações sobre Constraints

```
DESCRIBE user_cons_columns
```

Name	Null?	Type
OWNER	NOT NULL	VARCHAR2(30)
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)
TABLE_NAME	NOT NULL	VARCHAR2(30)
COLUMN_NAME		VARCHAR2(4000)
POSITION		NUMBER

```
SELECT constraint_name, column_name
FROM   user_cons_columns
WHERE  table_name = 'EMPLOYEES';
```

CONSTRAINT_NAME	COLUMN_NAME
EMP_EMAIL_UK	EMAIL
EMP_SALARY_MIN	SALARY
EMP_JOB_NN	JOB_ID
EMP_HIRE_DATE_NN	HIRE_DATE

...

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Consultando USER_CONS_COLUMNS

Consulte a view de dicionário USER_CONS_COLUMNS para obter os nomes das colunas às quais se aplica uma constraint. Essa view informa o nome do proprietário de uma constraint, o nome da constraint, a tabela que contém a constraint, o nome da coluna da constraint e a posição original da coluna ou do atributo na definição do objeto.

Você também pode criar uma join entre USER_CONSTRAINTS e USER_CONS_COLUMNS para gerar uma saída personalizada das duas tabelas.

Informações sobre Views

1

```
DESCRIBE user_views
```

Name	Null?	Type
VIEW_NAME	NOT NULL	VARCHAR2(30)
TEXT_LENGTH		NUMBER
TEXT		LONG

2

```
SELECT DISTINCT view_name FROM user_views;
```

VIEW_NAME
EMP_DETAILS_VIEW

3

```
SELECT text FROM user_views  
WHERE view_name = 'EMP_DETAILS_VIEW';
```

TEXT
SELECT e.employee_id, e.job_id, e.manager_id, e.department_id, d.location_id, l.country_id, e.first_name, e.last_name, e.salary, e.commission_pct, d.department_name, j.job_title, l.city, l.state_province, c.country_name, r.region_name FROM employees e, departments d, jobs j, locations l, countries c, regions r WHERE e.department_id = d.department_id AND d.location_id = l.location_id AND l.country_id = c.country_id AND c.region_id = r.region_id AND j.job_id = e.job_id WITH READ ONLY

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Views do Dicionário de Dados

Depois que a view for criada, você poderá consultar a view de dicionário de dados USER_VIEWS para verificar o nome e a definição da view. O texto da instrução SELECT que constitui a view é armazenado em uma coluna de tipo de dados LONG. A coluna LENGTH representa o número de caracteres na instrução SELECT. Por default, quando você seleciona um valor em uma coluna LONG, são exibidos apenas os 80 primeiros caracteres do valor da coluna. Para exibir mais do que 80 caracteres, use o comando *iSQL*Plus* SET LONG:

```
SET LONG 1000
```

Nos exemplos do slide:

1. As colunas USER_VIEWS são exibidas. Observe que essa é uma listagem parcial.
2. Os nomes das views são recuperados.
3. A instrução SELECT referente a EMP_DETAILS_VIEW é exibida a partir do dicionário.

Acesso a Dados Usando Views

Quando você acessa dados usando uma view, o servidor Oracle executa as seguintes operações:

- Recupera a definição da view da tabela do dicionário de dados USER_VIEWS.
- Verifica os privilégios de acesso da tabela-base da view.
- Converte a consulta da view em uma operação equivalente na(s) tabela(s) base subjacente(s). Em outras palavras, os dados são recuperados das tabelas-base ou ocorre uma atualização nessas tabelas.

Informações sobre Seqüências

```
DESCRIBE user_sequences
```

Name	Null?	Type
SEQUENCE_NAME	NOT NULL	VARCHAR2(30)
MIN_VALUE		NUMBER
MAX_VALUE		NUMBER
INCREMENT_BY	NOT NULL	NUMBER
CYCLE_FLAG		VARCHAR2(1)
ORDER_FLAG		VARCHAR2(1)
CACHE_SIZE	NOT NULL	NUMBER
LAST_NUMBER	NOT NULL	NUMBER

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

View USER_SEQUENCES

A view USER_SEQUENCES descreve todas as seqüências que pertencem a você. Quando cria a seqüência, você especifica critérios que são armazenados na view USER_SEQUENCES. As colunas dessa view são:

- SEQUENCE_NAME: Nome da seqüência
- MIN_VALUE: Valor mínimo da seqüência
- MAX_VALUE: Valor máximo da seqüência
- INCREMENT_BY: Valor de incremento da seqüência
- CYCLE_FLAG: A seqüência é reiniciada quando o limite é atingido?
- ORDER_FLAG: Os números da seqüência são gerados em ordem?
- CACHE_SIZE: Quantidade de números de seqüência armazenados no cache
- LAST_NUMBER: Último número de seqüência gravado no disco. Se uma seqüência usar armazenamento em cache, o número gravado no disco será o último número incluído no cache de seqüência. Esse número provavelmente será maior que o último número de seqüência usado.

Informações sobre Seqüências

- **Verifique os valores de uma seqüência na tabela USER_SEQUENCES do dicionário de dados.**

```
SELECT  sequence_name, min_value, max_value,
        increment_by, last_number
FROM    user_sequences;
```

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
LOCATIONS_SEQ	1	9900	100	3300
DEPARTMENTS_SEQ	1	9990	10	280
EMPLOYEES_SEQ	1	1.0000E+27	1	207

- **A coluna LAST_NUMBER exibirá o próximo número disponível da seqüência se NOCACHE for especificado.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Confirmando Seqüências

Após a criação da seqüência, ela é documentada no dicionário de dados. Como uma seqüência é um objeto de banco de dados, você pode identificá-la na tabela USER_OBJECTS do dicionário de dados.

Para confirmar as definições da seqüência, selecione-a na view USER_SEQUENCES do dicionário de dados.

Exibindo o Próximo Valor Disponível da Seqüência sem Incrementá-lo

Se a seqüência tiver sido criada com NOCACHE, você poderá exibir o próximo valor disponível dessa seqüência sem incrementá-lo consultando a tabela USER_SEQUENCES.

Informações sobre Sinônimos

```
DESCRIBE user_synonyms
```

Name	Null?	Type
SYNONYM_NAME	NOT NULL	VARCHAR2(30)
TABLE_OWNER		VARCHAR2(30)
TABLE_NAME	NOT NULL	VARCHAR2(30)
DB_LINK		VARCHAR2(128)

```
SELECT *  
FROM user_synonyms;
```

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
EMP	ORA1	EMPLOYEES	

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

View USER_SYNONYMS

A view de dicionário USER_SYNONYMS descreve sinônimos privados (sinônimos que pertencem a você).

Você pode consultar essa view para localizar seus sinônimos. Também pode consultar ALL_SYNONYMS para obter o nome de todos os sinônimos disponíveis e os objetos aos quais esses sinônimos se aplicam.

As colunas dessa view são:

- SYNONYM_NAME: Nome do sinônimo
- TABLE_OWNER: Proprietário do objeto referenciado pelo sinônimo
- TABLE_NAME: Nome da tabela ou view referenciada pelo sinônimo
- DB_LINK: Nome da referência do vínculo do banco de dados (se houver)

Adicionando Comentários a uma Tabela

- Você pode adicionar comentários a uma tabela ou coluna usando a instrução **COMMENT**:

```
COMMENT ON TABLE employees  
IS 'Employee Information';  
Comment created.
```

- É possível exibir comentários nas views de dicionário de dados:
 - ALL_COL_COMMENTS
 - USER_COL_COMMENTS
 - ALL_TAB_COMMENTS
 - USER_TAB_COMMENTS

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Adicionando Comentários a uma Tabela

Você pode adicionar um comentário de até 2.000 bytes sobre uma coluna, uma tabela, uma view ou um snapshot usando a instrução **COMMENT**. O comentário é armazenado no dicionário de dados e é possível exibi-lo em uma das seguintes views do dicionário de dados na coluna **COMMENTS**:

- ALL_COL_COMMENTS
- USER_COL_COMMENTS
- ALL_TAB_COMMENTS
- USER_TAB_COMMENTS

Sintaxe

```
COMMENT ON TABLE table | COLUMN table.column  
IS 'text';
```

Na sintaxe:

table é o nome da tabela
column é o nome da coluna da tabela
text é o texto do comentário

Para eliminar um comentário do banco de dados, defina-o como uma string vazia (' '):

```
COMMENT ON TABLE employees IS ' ';
```

Sumário

Nesta lição, você aprendeu a obter informações sobre objetos por meio das seguintes views de dicionário:

- **DICTIONARY**
- **USER_OBJECTS**
- **USER_TABLES**
- **USER_TAB_COLUMNS**
- **USER_CONSTRAINTS**
- **USER_CONS_COLUMNS**
- **USER_VIEWS**
- **USER_SEQUENCES**
- **USER_TAB_SYNONYMS**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Sumário

Nesta lição, você conheceu algumas views de dicionário disponíveis. É possível usar essas views de dicionário para obter informações sobre tabelas, constraints, views, seqüências e sinônimos.

Exercício 11: Visão Geral

Este exercício aborda os seguintes tópicos:

- Consultando as views de dicionário para obter informações sobre tabelas e colunas
- Consultando as views de dicionário para obter informações sobre constraints
- Consultando as views de dicionário para obter informações sobre views
- Consultando as views de dicionário para obter informações sobre seqüências
- Consultando as views de dicionário para obter informações sobre sinônimos
- Adicionando um comentário a uma tabela e consultando as views de dicionário para verificar comentários

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Exercício 11: Visão Geral

Neste exercício, você consultará as views de dicionário para obter informações sobre os objetos do seu esquema.

Exercício 11

1. Para uma tabela especificada, crie um script que informe os nomes de colunas, os tipos de dados, os tamanhos dos tipos de dados e a permissão ou não de valores nulos. Solicite o nome da tabela ao usuário. Forneça apelidos apropriados às colunas DATA_PRECISION e DATA_SCALE. Salve esse script no arquivo lab_11_01.sql.

Por exemplo, se o usuário informar DEPARTMENTS, este será o resultado:

COLUMN_NAME	DATA_TYPE	DATA_LENGTH	PRECISION	SCALE	NUL
DEPARTMENT_ID	NUMBER	22	4	0	N
DEPARTMENT_NAME	VARCHAR2	30			N
MANAGER_ID	NUMBER	22	6	0	Y
LOCATION_ID	NUMBER	22	4	0	Y

2. Crie um script que informe o nome da coluna, o nome da constraint, o tipo de constraint, a condição de pesquisa e o status de uma tabela especificada. Junte as tabelas USER_CONSTRAINTS e USER_CONS_COLUMNS para obter todas essas informações. Solicite o nome da tabela ao usuário. Salve o script no arquivo lab_11_02.sql.

Por exemplo, se o usuário informar DEPARTMENTS, este será o resultado:

COLUMN_NAME	CONSTRAINT_NAME	CON	SEARCH_CONDITION	STATUS
DEPARTMENT_NAME	DEPT_NAME_NN	C	"DEPARTMENT_NAME" IS NOT NULL	ENABLED
DEPARTMENT_ID	DEPT_ID_PK	P		ENABLED
LOCATION_ID	DEPT_LOC_FK	R		ENABLED
MANAGER_ID	DEPT_MGR_FK	R		ENABLED

3. Adicione um comentário à tabela DEPARTMENTS. Em seguida, consulte a view USER_TAB_COMMENTS para verificar se o comentário está presente.

COMMENTS
Company department information including name, code, and location.

4. Localize os nomes de todos os sinônimos existentes no esquema.

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
EMP	ORA1	EMPLOYEES	

Exercício 11

5. Você precisa determinar os nomes e as definições de todas as views do seu esquema. Crie um relatório que recupere informações sobre views: o nome e o texto das views da view de dicionário de dados USER_VIEWS.

Observação: Já existe outra view. A view EMP_DETAILS_VIEW foi criada como parte do seu esquema.

Observação: Para exibir mais conteúdo de uma coluna LONG, use o comando SET LONG *n* do iSQL*Plus, em que *n* é o valor do número de caracteres da coluna LONG que você deseja exibir.

VIEW_NAME	TEXT
EMPLOYEES_VU	SELECT employee_id, last_name employee, department_id FROM employees
EMP_DETAILS_VIEW	SELECT e.employee_id, e.job_id, e.manager_id, e.department_id, d.location_id, l.country_id, e.first_name, e.last_name, e.salary, e.commission_pct, d.department_name, j.job_title, l.city, l.state_province, c.country_name, r.region_name FROM employees e, departments d, jobs j, locations l, countries c, regions r WHERE e.department_id = d.department_id AND d.location_id = l.location_id AND l.country_id = c.country_id AND c.region_id = r.region_id AND j.job_id = e.job_id WITH READ ONLY

6. Obtenha os nomes das suas seqüências. Crie uma consulta em um script para exibir as seguintes informações sobre suas seqüências: nome, valor máximo, tamanho do incremento e último número. Nomeie o script como lab_11_06.sql. Execute a instrução no script.

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPARTMENTS_SEQ	9990	10	280
DEPT_ID_SEQ	1000	10	200
EMPLOYEES_SEQ	1.0000E+27	1	207
LOCATIONS_SEQ	9900	100	3300

A

Soluções dos Exercícios

Exercício 1: Soluções

Parte 1

Teste seu conhecimento:

1. Inicie uma sessão do *iSQL*Plus* com o ID de usuário e a senha fornecidos pelo instrutor.
2. Os comandos do *iSQL*Plus* acessam o banco de dados.
Verdadeiro/Falso
3. Esta instrução SELECT é executada com êxito:

```
SELECT last_name, job_id, salary AS Sal
FROM employees;
```

Verdadeiro/Falso

4. Esta instrução SELECT é executada com êxito:

```
SELECT *
FROM job_grades;
```

Verdadeiro/Falso

5. Há quatro erros de codificação nesta instrução. Você consegue identificá-los?

```
SELECT employee_id, last_name
sal x 12 ANNUAL SALARY
FROM employees;
```

- A tabela **EMPLOYEES** não contém uma coluna denominada **sal**. O nome da coluna é **SALARY**.
- O operador de multiplicação é *****, e não **x**, como mostrado na linha 2.
- O apelido **ANNUAL SALARY** não pode conter espaços. É necessário informá-lo como **ANNUAL_SALARY** ou especificá-lo entre aspas duplas.
- Está faltando uma vírgula após a coluna **LAST_NAME**.

Parte 2

Você foi admitido como programador SQL da Acme Corporation. Sua primeira tarefa é criar alguns relatórios com base nos dados das tabelas de recursos humanos.

6. Sua primeira tarefa é determinar a estrutura e o conteúdo da tabela **DEPARTMENTS**.

```
DESCRIBE departments

SELECT *
FROM departments;
```


Exercício 1: Soluções (continuação)

7. Você precisa determinar a estrutura da tabela EMPLOYEES.

```
DESCRIBE employees
```

O departamento de recursos humanos deseja executar uma consulta para exibir o sobrenome, o código do cargo, a data de admissão e o telefone de cada funcionário, com o número do funcionário exibido primeiro. Forneça o apelido STARTDATE para a coluna HIRE_DATE. Salve a instrução SQL no arquivo lab_01_07.sql para encaminhar esse arquivo ao departamento de recursos humanos.

```
SELECT employee_id, last_name, job_id, hire_date StartDate
FROM employees;
```

8. Teste a consulta no arquivo lab_01_07.sql para verificar se é executada corretamente.

```
SELECT employee_id, last_name, job_id, hire_date StartDate
FROM employees;
```

9. O departamento de recursos humanos precisa de uma consulta para exibir todos os códigos de cargo exclusivos da tabela EMPLOYEES.

```
SELECT DISTINCT job_id
FROM employees;
```

Parte 3

Se tiver tempo, faça os seguintes exercícios:

10. O departamento de recursos humanos deseja cabeçalhos de coluna mais descritivos em seu relatório sobre funcionários. Copie a instrução de lab_01_07.sql para a janela Edit do iSQL*Plus. Nomeie os cabeçalhos de coluna como Emp #, Employee, Job e Hire Date, respectivamente. Execute a consulta novamente.

```
SELECT employee_id "Emp #", last_name "Employee",
       job_id "Job", hire_date "Hire Date"
FROM employees;
```

11. O departamento de recursos humanos solicitou um relatório de todos os funcionários e os respectivos IDs de cargo. Exiba o sobrenome concatenado com o ID do cargo (separado por uma vírgula e um espaço) e nomeie a coluna como Employee and Title.

```
SELECT last_name||', '||job_id "Employee and Title"
FROM employees;
```

Exercício 1: Soluções (continuação)

Se deseja mais um desafio, faça este exercício:

12. Para se familiarizar com os dados da tabela EMPLOYEES, crie uma consulta para exibir todos os dados dessa tabela. Separe cada saída de coluna com uma vírgula. Nomeie o título da coluna como THE_OUTPUT.

```
SELECT employee_id || ',' || first_name || ',' || last_name  
       || ',' || email || ',' || phone_number || ',' || job_id  
       || ',' || manager_id || ',' || hire_date || ',' ||  
       || salary || ',' || commission_pct || ',' || department_id  
       THE_OUTPUT  
FROM   employees;
```

Exercício 2: Soluções

O departamento de recursos humanos precisa da sua ajuda para criar algumas consultas.

1. Em função de questões orçamentárias, o departamento precisa de um relatório com o sobrenome e o salário dos funcionários que ganham mais de US\$ 12.000. Inclua a instrução SQL no arquivo de texto lab_02_01.sql. Execute a consulta.

```
SELECT last_name, salary
FROM employees
WHERE salary > 12000;
```

2. Crie um relatório que exiba o sobrenome e o número do departamento do funcionário 176.

```
SELECT last_name, department_id
FROM employees
WHERE employee_id = 176;
```

3. O departamento de recursos humanos precisa localizar funcionários com altos e baixos salários. Modifique lab2_1.sql para exibir o sobrenome e o salário de todos os funcionários cuja faixa salarial não esteja entre US\$ 5.000 e US\$ 12.000. Inclua a instrução SQL no arquivo de texto lab_02_03.sql.

```
SELECT last_name, salary
FROM employees
WHERE salary NOT BETWEEN 5000 AND 12000;
```

4. Crie um relatório para exibir o sobrenome, o ID do cargo e a data de admissão dos funcionários cujos sobrenomes sejam Matos e Taylor. Organize a consulta em ordem crescente por data de admissão.

```
SELECT last_name, job_id, hire_date
FROM employees
WHERE last_name IN ('Matos', 'Taylor')
ORDER BY hire_date;
```

5. Exiba o sobrenome e o número do departamento de todos os funcionários nos departamentos 20 e 50 em ordem alfabética crescente por nome.

```
SELECT last_name, department_id
FROM employees
WHERE department_id IN (20, 50)
ORDER BY last_name ASC;
```

Exercício 2: Soluções (continuação)

6. Modifique lab_02_03.sql para listar o sobrenome e o salário dos funcionários que ganham entre US\$ 5.000 e US\$ 12.000 e estão no departamento 20 ou 50. Atribua às colunas os labels Employee e Monthly Salary, respectivamente. Salve novamente lab_02_03.sql como lab_02_06.sql. Execute a instrução em lab_02_06.sql.

```
SELECT    last_name "Employee", salary "Monthly Salary"
FROM      employees
WHERE     salary BETWEEN 5000 AND 12000
AND       department_id IN (20, 50);
```

7. O departamento de recursos humanos precisa de um relatório que exiba o sobrenome e a data de admissão de todos os funcionários admitidos em 1994.

```
SELECT    last_name, hire_date
FROM      employees
WHERE     hire_date LIKE '%94';
```

8. Crie um relatório que exiba o sobrenome e o cargo de todos os funcionários não subordinados a um gerente.

```
SELECT    last_name, job_id
FROM      employees
WHERE     manager_id IS NULL;
```

9. Exiba o sobrenome, o salário e a comissão de todos os funcionários que ganham comissão. Classifique os dados em ordem decrescente de salário e comissões.

```
SELECT    last_name, salary, commission_pct
FROM      employees
WHERE     commission_pct IS NOT NULL
ORDER BY salary DESC, commission_pct DESC;
```

10. Os membros do departamento de recursos humanos desejam ter mais flexibilidade em relação às consultas criadas. Eles desejam um relatório que exiba o sobrenome e o salário dos funcionários que ganham mais do que uma quantia especificada pelo usuário após o prompt. (Você pode usar a consulta criada no exercício 1 e modificá-la.) Salve essa consulta no arquivo lab_02_10.sql. Se você informar 12000 quando a quantia for solicitada, o relatório exibirá estes resultados:

```
SELECT    last_name, salary
FROM      employees
WHERE     salary > &sal_amt;
```

Exercício 2: Soluções (continuação)

11. O departamento de recursos humanos deseja executar relatórios baseados em um gerente. Crie uma consulta que solicite um ID de gerente ao usuário e gere o ID de funcionário, o sobrenome, o salário e o departamento dos funcionários desse gerente. O departamento de recursos humanos deseja ter permissão para classificar o relatório em uma coluna selecionada. Você pode testar os dados com os seguintes valores:

ID do gerente = 103, classificado pelo sobrenome do funcionário

ID do gerente = 201, classificado pelo salário

ID do gerente = 124, classificado pelo ID do funcionário

```
SELECT employee_id, last_name, salary, department_id
FROM employees
WHERE manager_id = &mgr_num
ORDER BY &order_col;
```

Se tiver tempo, faça os seguintes exercícios:

12. Exiba todos os sobrenomes dos funcionários cuja terceira letra do nome seja *a*.

```
SELECT last_name
FROM employees
WHERE last_name LIKE '__a%';
```

13. Exiba o sobrenome de todos os funcionários que contenha *a* e *e*.

```
SELECT last_name
FROM employees
WHERE last_name LIKE '%a%'
AND last_name LIKE '%e%';
```

Se deseja mais um desafio, faça estes exercícios:

14. Exiba o sobrenome, o cargo e o salário de todos os funcionários cujo cargo seja representante de vendas ou estoquista e cujo salário seja diferente de US\$ 2.500, US\$ 3.500 ou US\$ 7.000.

```
SELECT last_name, job_id, salary
FROM employees
WHERE job_id IN ('SA_REP', 'ST_CLERK')
AND salary NOT IN (2500, 3500, 7000);
```

15. Modifique lab_02_06.sql para exibir o sobrenome, o salário e a comissão de todos os funcionários cuja comissão seja de 20%. Salve novamente lab_02_06.sql como lab_02_15.sql. Reexecute a instrução em lab_02_15.sql.

```
SELECT last_name "Employee", salary "Monthly Salary",
commission_pct
FROM employees
WHERE commission_pct = .20;
```

Exercício 3: Soluções

1. Crie uma consulta para exibir a data atual. Atribua à coluna o label Date.

```
SELECT sysdate "Date"
FROM dual;
```

2. O departamento de recursos humanos precisa de um relatório para exibir o número do funcionário, o sobrenome, o salário e o salário com 15,5% de aumento (especificado como um número inteiro) de cada funcionário. Atribua à coluna o label New Salary. Inclua a instrução SQL no arquivo de texto lab_03_02.sql.

```
SELECT employee_id, last_name, salary,
       ROUND(salary * 1.155, 0) "New Salary"
FROM employees;
```

3. Execute a consulta no arquivo lab_03_02.sql.

```
SELECT employee_id, last_name, salary,
       ROUND(salary * 1.155, 0) "New Salary"
FROM employees;
```

4. Modifique a consulta lab_03_02.sql para adicionar uma coluna que subtraia o salário antigo do novo salário. Atribua à coluna o label Increase. Salve o conteúdo do arquivo como lab_03_04.sql. Execute a consulta modificada.

```
SELECT employee_id, last_name, salary,
       ROUND(salary * 1.155, 0) "New Salary",
       ROUND(salary * 1.155, 0) - salary "Increase"
FROM employees;
```

5. Crie uma consulta que exiba o sobrenome (com a primeira letra maiúscula e todas as outras minúsculas) e o tamanho do sobrenome de todos os funcionários cujos nomes comecem com a letra J, A ou M. Atribua um label apropriado a cada coluna. Classifique os resultados pelos sobrenomes dos funcionários.

```
SELECT INITCAP(last_name) "Name",
       LENGTH(last_name) "Length"
FROM employees
WHERE last_name LIKE 'J%'
OR last_name LIKE 'M%'
OR last_name LIKE 'A%'
ORDER BY last_name ;
```

Exercício 3: Soluções (continuação)

Recrie a consulta de modo que a primeira letra do sobrenome seja solicitada ao usuário. Por exemplo, se o usuário informar H quando uma letra for solicitada, a saída deverá mostrar todos os funcionários cujos sobrenomes começam com a letra H.

```
SELECT  INITCAP(last_name) "Name",
        LENGTH(last_name) "Length"
FROM    employees
WHERE   last_name LIKE '&start_letter%'
ORDER BY last_name;
```

6. O departamento de recursos humanos deseja saber qual é o tempo de emprego de cada funcionário. Para cada funcionário, exiba o sobrenome e calcule o número de meses entre hoje e a data de admissão do funcionário. Atribua o label MONTHS_WORKED à coluna. Ordene os resultados pelo número de meses em que o funcionário está empregado. Arredonde o número de meses para o número inteiro mais próximo.

Observação: Os resultados serão diferentes.

```
SELECT last_name, ROUND(MONTHS_BETWEEN(
        SYSDATE, hire_date)) MONTHS_WORKED
FROM    employees
ORDER BY MONTHS_BETWEEN(SYSDATE, hire_date);
```

7. Crie um relatório que produza estas informações para cada funcionário:
<employee last name> earns <salary> monthly but wants <3 times salary>.

Atribua o label Dream Salaries à coluna.

```
SELECT  last_name || ' earns '
        || TO_CHAR(salary, 'fm$99,999.00')
        || ' monthly but wants '
        || TO_CHAR(salary * 3, 'fm$99,999.00')
        || '.' "Dream Salaries"
FROM    employees;
```

Se tiver tempo, faça os seguintes exercícios:

8. Crie uma consulta para exibir o sobrenome e o salário de todos os funcionários. Formate o salário para defini-lo com um tamanho de 15 caracteres e preenchê-lo à esquerda com o símbolo \$. Atribua o label SALARY à coluna.

```
SELECT last_name,
        LPAD(salary, 15, '$') SALARY
FROM    employees;
```

Exercício 3: Soluções (continuação)

9. Exiba o sobrenome, a data de admissão e a data de revisão de salário de cada funcionário, que é a primeira segunda-feira após seis meses de serviço. Atribua o label REVIEW à coluna. Formate as datas para que sejam exibidas no formato semelhante a “Monday, the Thirty-First of July, 2000”.

```
SELECT last_name, hire_date,  
       TO_CHAR(NEXT_DAY(ADD_MONTHS(hire_date, 6), 'MONDAY'),  
               'fmDay, "the" Ddspth "of" Month, YYYY') REVIEW  
FROM   employees;
```

10. Exiba o sobrenome, a data de admissão e o dia da semana em que o funcionário começou a trabalhar. Atribua o label DAY à coluna. Ordene os resultados pelo dia da semana, começando por segunda-feira.

```
SELECT last_name, hire_date,  
       TO_CHAR(hire_date, 'DAY') DAY  
FROM   employees  
ORDER BY TO_CHAR(hire_date - 1, 'd');
```

Se deseja mais um desafio, faça estes exercícios:

11. Crie uma consulta que exiba os sobrenomes e as comissões dos funcionários. Se um funcionário não ganhar comissão, a informação “No Commission” deverá ser exibida. Atribua o label COMM à coluna.

```
SELECT last_name,  
       NVL(TO_CHAR(commission_pct), 'No Commission') COMM  
FROM   employees;
```

12. Crie uma consulta que exiba os oito primeiros caracteres dos sobrenomes dos funcionários e indique os valores dos salários com asteriscos. Cada asterisco representa mil dólares. Classifique os dados em ordem decrescente de salário. Atribua o label EMPLOYEES_AND_THEIR_SALARIES à coluna.

```
SELECT rpad(last_name, 8) || ' ' ||  
       rpad(' ', salary/1000+1, '*')  
       EMPLOYEES_AND_THEIR_SALARIES  
FROM   employees  
ORDER BY salary DESC;
```


Exercício 3: Soluções (continuação)

13. Com a function DECODE, crie uma consulta que exiba o nível de todos os funcionários com base no valor da coluna JOB_ID. Use estes dados:

<i>Cargo</i>	<i>Nível</i>
AD_PRES	A
ST_MAN	B
IT_PROG	C
SA_REP	D
ST_CLERK	E
Nenhuma das opções anteriores	0

```
SELECT job_id, decode (job_id,  
                        'ST_CLERK', 'E',  
                        'SA_REP', 'D',  
                        'IT_PROG', 'C',  
                        'ST_MAN', 'B',  
                        'AD_PRES', 'A',  
                        '0')GRADE  
FROM employees;
```

14. Recrie a instrução no exercício anterior usando a sintaxe CASE.

```
SELECT job_id, CASE job_id  
                WHEN 'ST_CLERK' THEN 'E'  
                WHEN 'SA_REP'   THEN 'D'  
                WHEN 'IT_PROG'  THEN 'C'  
                WHEN 'ST_MAN'   THEN 'B'  
                WHEN 'AD_PRES'  THEN 'A'  
                ELSE '0' END GRADE  
FROM employees;
```

Exercício 4: Soluções

Determine a validade das três afirmações a seguir. Circule Verdadeiro ou Falso.

1. As functions de grupo atuam em várias linhas para produzir um resultado por grupo.

Verdadeiro/Falso

2. As functions de grupo incluem valores nulos em cálculos.

Verdadeiro/**Falso**

3. A cláusula WHERE restringe linhas antes da inclusão em um cálculo de grupo.

Verdadeiro/Falso

O departamento de recursos humanos necessita dos seguintes relatórios:

4. Obtenha o maior salário, o menor salário, a soma dos salários e o salário médio de todos os funcionários. Atribua às colunas os labels

Maximum, Minimum, Sum e Average, respectivamente. Arredonde os resultados para o número inteiro mais próximo. Inclua a instrução SQL no arquivo de texto

lab_04_04.sql.

```
SELECT ROUND(MAX(salary),0) "Maximum",  
       ROUND(MIN(salary),0) "Minimum",  
       ROUND(SUM(salary),0) "Sum",  
       ROUND(AVG(salary),0) "Average"  
FROM   employees;
```

5. Modifique a consulta em lab_04_04.sql para exibir o salário mínimo, o salário máximo, a soma dos salários e o salário médio de cada tipo de cargo. Salve novamente

lab_04_04.sql como lab_04_05.sql. Execute a instrução em lab_04_05.sql.

```
SELECT job_id, ROUND(MAX(salary),0) "Maximum",  
       ROUND(MIN(salary),0) "Minimum",  
       ROUND(SUM(salary),0) "Sum",  
       ROUND(AVG(salary),0) "Average"  
FROM   employees  
GROUP BY job_id;
```

6. Crie uma consulta para exibir o número de pessoas com o mesmo cargo.

```
SELECT job_id, COUNT(*)  
FROM   employees  
GROUP BY job_id;
```

Generalize a consulta para que um cargo seja solicitado ao usuário no departamento de recursos humanos. Salve o script no arquivo lab_04_06.sql.

```
SELECT job_id, COUNT(*)  
FROM   employees  
WHERE  job_id = '&job_title'  
GROUP BY job_id;
```

Exercício 4: Soluções (continuação)

7. Determine o número de gerentes sem listá-los. Atribua o label `Number of Managers` à coluna. *Dica: Use a coluna `MANAGER_ID` para determinar o número de gerentes.*

```
SELECT COUNT(DISTINCT manager_id) "Number of Managers"
FROM   employees;
```

8. Obtenha a diferença entre o salário mais alto e o mais baixo. Atribua o label `DIFFERENCE` à coluna.

```
SELECT   MAX(salary) - MIN(salary) DIFFERENCE
FROM     employees;
```

Se tiver tempo, faça os seguintes exercícios:

9. Crie um relatório para exibir o número do gerente e o salário do funcionário com menor remuneração subordinado a esse gerente. Exclua todas as pessoas cujos gerentes sejam desconhecidos. Exclua todos os grupos em que o salário mínimo seja US\$ 6.000 ou inferior. Classifique a saída em ordem decrescente de salário.

```
SELECT   manager_id, MIN(salary)
FROM     employees
WHERE    manager_id IS NOT NULL
GROUP BY manager_id
HAVING   MIN(salary) > 6000
ORDER BY MIN(salary) DESC;
```

Se deseja mais um desafio, faça estes exercícios:

10. Crie uma consulta que exiba o número total de funcionários e, desse total, mostre o número de funcionários admitidos em 1995, 1996, 1997 e 1998. Crie cabeçalhos de colunas apropriados.

```
SELECT   COUNT(*) total,
          SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1995,1,0))"1995",
          SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1996,1,0))"1996",
          SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1997,1,0))"1997",
          SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1998,1,0))"1998"
FROM     employees;
```

Exercício 4: Soluções (continuação)

11. Crie uma consulta matriz que exiba o cargo, o salário relativo a esse cargo com base no número do departamento e o salário total desse cargo para os departamentos 20, 50, 80 e 90. Atribua um cabeçalho apropriado a cada coluna.

```
SELECT    job_id "Job",  
          SUM(DECODE(department_id , 20, salary)) "Dept 20",  
          SUM(DECODE(department_id , 50, salary)) "Dept 50",  
          SUM(DECODE(department_id , 80, salary)) "Dept 80",  
          SUM(DECODE(department_id , 90, salary)) "Dept 90",  
          SUM(salary) "Total"  
FROM      employees  
GROUP BY  job_id;
```

Exercício 5: Soluções

1. Crie uma consulta para o departamento de recursos humanos a fim de obter os endereços de todos os departamentos. Use as tabelas `LOCATIONS` e `COUNTRIES`. Mostre o ID do local, o endereço, a cidade, o estado e o país na saída. Use `NATURAL JOIN` para gerar os resultados.

```
SELECT location_id, street_address, city, state_province, country_name
FROM   locations
NATURAL JOIN countries;
```

2. O departamento de recursos humanos precisa de um relatório de todos os funcionários. Crie uma consulta para exibir o sobrenome, o número do departamento e o nome do departamento de todos os funcionários.

```
SELECT last_name, department_id, department_name
FROM   employees
JOIN   departments
USING (department_id);
```

3. O departamento de recursos humanos precisa de um relatório dos funcionários em Toronto. Exiba o sobrenome, o cargo, o número do departamento e o nome do departamento de todos os funcionários que trabalham em Toronto.

```
SELECT e.last_name, e.job_id, e.department_id, d.department_name
FROM   employees e JOIN departments d
ON     (e.department_id = d.department_id)
JOIN   locations l
ON     (d.location_id = l.location_id)
WHERE  LOWER(l.city) = 'toronto';
```

4. Crie um relatório para exibir o sobrenome e o número dos funcionários, bem como o sobrenome e o número dos respectivos gerentes. Atribua às colunas os labels `Employee`, `Emp#`, `Manager` e `Mgr#`, respectivamente. Inclua a instrução SQL no arquivo de texto `lab_05_04.sql`.

```
SELECT w.last_name "Employee", w.employee_id "EMP#",
       m.last_name "Manager", m.employee_id  "Mgr#"
FROM   employees w join employees m
ON     (w.manager_id = m.employee_id);
```

5. Modifique `lab_05_04.sql` para exibir todos os funcionários, inclusive King, que não possui gerente. Ordene os resultados pelo número do funcionário. Inclua a instrução SQL no arquivo de texto `lab_05_05.sql`. Execute a consulta em `lab_05_05.sql`.

```
SELECT w.last_name "Employee", w.employee_id "EMP#",
       m.last_name "Manager", m.employee_id  "Mgr#"
FROM   employees w
LEFT   OUTER JOIN employees m
ON     (w.manager_id = m.employee_id);
```

Exercício 5: Soluções (continuação)

6. Crie um relatório para o departamento de recursos humanos que exiba os sobrenomes e os números de departamento dos funcionários, bem como todos os funcionários que trabalham no mesmo departamento como um funcionário específico. Atribua um label apropriado a cada coluna. Salve o script no arquivo lab_05_06.sql.

```
SELECT e.department_id department, e.last_name employee,  
       c.last_name colleague  
FROM   employees e JOIN employees c  
ON     (e.department_id = c.department_id)  
WHERE  e.employee_id <> c.employee_id  
ORDER BY e.department_id, e.last_name, c.last_name;
```

7. O departamento de recursos humanos precisa de um relatório sobre níveis de cargos e salários. Para se familiarizar com a tabela JOB_GRADES, primeiro mostre a estrutura dessa tabela. Em seguida, crie uma consulta que exiba o nome, o cargo, o nome do departamento, o salário e o nível de todos os funcionários.

```
DESC JOB_GRADES  
  
SELECT e.last_name, e.job_id, d.department_name,  
       e.salary, j.grade_level  
FROM   employees e JOIN departments d  
ON     (e.department_id = d.department_id)  
JOIN   job_grades j  
ON     (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

Se deseja mais um desafio, faça estes exercícios:

8. O departamento de recursos humanos deseja determinar os nomes de todos os funcionários admitidos após Davies. Crie uma consulta para exibir o nome e a data de admissão de todos os funcionários admitidos após Davies.

```
SELECT e.last_name, e.hire_date  
FROM   employees e JOIN employees davies  
ON     (davies.last_name = 'Davies')  
WHERE  davies.hire_date < e.hire_date;
```

Exercício 5: Soluções (continuação)

9. O departamento de recursos humanos precisa obter os nomes e as datas de admissão de todos os funcionários admitidos antes dos respectivos gerentes, além dos nomes e das datas de admissão desses gerentes. Salve o script no arquivo `lab_05_09.sql`.

```
SELECT w.last_name, w.hire_date, m.last_name, m.hire_date
FROM   employees w JOIN employees m
ON     (w.manager_id = m.employee_id)
WHERE  w.hire_date < m.hire_date;
```

Exercício 6: Soluções

1. O departamento de recursos humanos precisa de uma consulta que solicite ao usuário o sobrenome de um funcionário. A consulta exibe o sobrenome e a data de admissão de todos os funcionários no mesmo departamento do funcionário cujo nome foi fornecido (excluindo esse funcionário). Por exemplo, se o usuário informar Zlotkey, serão exibidos todos os funcionários que trabalham com Zlotkey (excluindo ele próprio).

```
UNDEFINE Enter_name

SELECT last_name, hire_date
FROM   employees
WHERE  department_id = (SELECT department_id
                        FROM   employees
                        WHERE  last_name = '&Enter_name')
AND    last_name <> '&Enter_name';
```

2. Crie um relatório que exiba o número e o sobrenome de todos os funcionários cujo salário é maior que o salário médio. Classifique os resultados em ordem crescente de salário.

```
SELECT employee_id, last_name
FROM   employees
WHERE  salary > (SELECT AVG(salary)
                 FROM   employees)
ORDER BY salary;
```

3. Crie uma consulta que exiba o número e o sobrenome de todos os funcionários que trabalham em um departamento com funcionários cujos sobrenomes contêm a letra *u*. Inclua a instrução SQL no arquivo de texto lab_06_03.sql. Execute a consulta.

```
SELECT employee_id, last_name
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   employees
                        WHERE  last_name like '%u%');
```

4. O departamento de recursos humanos precisa de um relatório que exiba o sobrenome, o número do departamento e o ID do cargo de todos os funcionários cujo ID de local do departamento é 1700.

```
SELECT last_name, department_id, job_id
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   departments
                        WHERE  location_id = 1700);
```

Modifique a consulta para que um ID de local seja solicitado ao usuário. Salve-a no arquivo lab_06_04.sql.

```
SELECT last_name, department_id, job_id
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   departments
                        WHERE  location_id = &Enter_location);
```


Exercício 6: Soluções (continuação)

5. Crie um relatório para o departamento de recursos humanos que exiba o sobrenome e o salário de todos os funcionários subordinados a King.

```
SELECT last_name, salary
FROM   employees
WHERE  manager_id = (SELECT employee_id
                     FROM   employees
                     WHERE  last_name = 'King');
```

6. Crie um relatório para o departamento de recursos humanos que exiba o número do departamento, o sobrenome e o ID do cargo de todos os funcionários no departamento executivo.

```
SELECT department_id, last_name, job_id
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   departments
                        WHERE  department_name = 'Executive');
```

Se tiver tempo, faça o seguinte exercício:

7. Modifique a consulta em lab_06_03.sql para exibir o número, o sobrenome, bem como o salário de todos os funcionários que ganham mais que o salário médio e trabalham em um departamento com funcionários cujos sobrenomes contêm a letra *u*. Salve novamente lab_06_03.sql como lab_06_07.sql. Execute a instrução em lab_06_07.sql.

```
SELECT employee_id, last_name, salary
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   employees
                        WHERE  last_name like '%u%')
AND    salary > (SELECT AVG(salary)
                 FROM   employees);
```

Exercício 7: Soluções

1. O departamento de recursos humanos precisa de uma lista de IDs dos departamentos que não contêm o ID de cargo ST_CLERK. Use operadores de conjunto para criar esse relatório.

```
SELECT department_id
FROM   departments
MINUS
SELECT department_id
FROM   employees
WHERE  job_id = 'ST_CLERK';
```

2. O departamento de recursos humanos precisa de uma lista de países nos quais não há departamentos. Exiba o ID e o nome dos países. Use operadores de conjunto para criar esse relatório.

```
SELECT country_id, country_name
FROM   countries
MINUS
SELECT l.country_id, c.country_name
FROM   locations l,   countries c
WHERE  l.country_id = c.country_id;
```

3. Produza uma lista de cargos dos departamentos 10, 50 e 20, nessa ordem. Exiba o ID de cargo e o ID de departamento usando operadores de conjunto.

```
COLUMN dummy NOPRINT
SELECT job_id, department_id, 'x' dummy
FROM   employees
WHERE  department_id = 10
UNION
SELECT job_id, department_id, 'y' dummy
FROM   employees
WHERE  department_id = 50
UNION
SELECT job_id, department_id, 'z' dummy
FROM   employees
WHERE  department_id = 20
ORDER BY dummy;
COLUMN dummy PRINT
```

4. Crie um relatório que liste os IDs de funcionário e os IDs de cargo dos funcionários que, no momento, estão no mesmo cargo que ocupavam quando foram admitidos pela empresa (ou seja, eles mudaram de cargo, mas agora voltaram para o cargo original).

```
SELECT   employee_id, job_id
FROM     employees
INTERSECT
SELECT   employee_id, job_id
FROM     job_history;
```

Exercício 7: Soluções (continuação)

5. O departamento de recursos humanos precisa de um relatório com as seguintes especificações:

- Sobrenome e ID do departamento de todos os funcionários da tabela EMPLOYEES, mesmo que não pertençam a um departamento
- ID e nome de todos os departamentos da tabela DEPARTMENTS, mesmo que não tenham funcionários

Crie uma consulta composta para isso.

```
SELECT last_name,department_id,TO_CHAR(null)
FROM   employees
UNION
SELECT TO_CHAR(null),department_id,department_name
FROM   departments;
```

Exercício 8: Soluções

O departamento de recursos humanos deseja criar instruções SQL para inserir, atualizar e deletar dados de funcionários. Como protótipo, use a tabela MY_EMPLOYEE antes de fornecer as instruções ao departamento de recursos humanos.

Insira dados na tabela MY_EMPLOYEE.

1. Execute a instrução no script lab_08_01.sql para criar a tabela MY_EMPLOYEE a ser usada no exercício.

```
CREATE TABLE my_employee
(id NUMBER(4) CONSTRAINT my_employee_id_nn NOT NULL,
last_name VARCHAR2(25),
first_name VARCHAR2(25),
userid VARCHAR2(8),
salary NUMBER(9,2));
```

2. Descreva a estrutura da tabela MY_EMPLOYEE para identificar os nomes de coluna.

```
DESCRIBE my_employee
```

3. Crie uma instrução INSERT para adicionar a primeira linha de dados à tabela MY_EMPLOYEE usando estes dados de amostra. Não liste as colunas na cláusula INSERT.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	cnewman	750
5	Ropeburn	Audrey	aropebur	1550

```
INSERT INTO my_employee
VALUES (1, 'Patel', 'Ralph', 'rpatel', 895);
```

Exercício 8: Soluções (continuação)

4. Preencha a tabela MY_EMPLOYEE com a segunda linha de dados de amostra da lista anterior. Desta vez, liste as colunas explicitamente na cláusula INSERT.

```
INSERT INTO my_employee (id, last_name, first_name,  
                        userid, salary)  
VALUES (2, 'Dancs', 'Betty', 'bdancs', 860);
```

5. Confirme a adição à tabela.

```
SELECT  *  
FROM    my_employee;
```

Exercício 8: Soluções (continuação)

6. Crie uma instrução insert no arquivo de script dinâmico reutilizável `loademp.sql` para carregar linhas na tabela `MY_EMPLOYEE`. Concatene a primeira letra do primeiro nome e os sete primeiros caracteres do sobrenome para gerar o ID do usuário. Salve esse script no arquivo `lab_08_06.sql`.

```
SET ECHO OFF
SET VERIFY OFF
INSERT INTO my_employee
VALUES (&p_id, '&p_last_name', '&p_first_name',
       lower(substr('&p_first_name', 1, 1) ||
       substr('&p_last_name', 1, 7)), &p_salary);
SET VERIFY ON
SET ECHO ON
UNDEFINE p_first_name
UNDEFINE p_last_name
```

7. Para preencher a tabela com as próximas duas linhas dos dados de amostra, execute a instrução insert no script criado.

```
SET ECHO OFF
SET VERIFY OFF
INSERT INTO my_employee
VALUES (&p_id, '&p_last_name', '&p_first_name',
       lower(substr('&p_first_name', 1, 1) ||
       substr('&p_last_name', 1, 7)), &p_salary);
SET VERIFY ON
SET ECHO ON
UNDEFINE p_first_name
UNDEFINE p_last_name
```

8. Confirme as adições à tabela.

```
SELECT  *
FROM my_employee;
```

9. Torne as adições de dados permanente.

```
COMMIT;
```

Atualize e delete dados na tabela `MY_EMPLOYEE`.

10. Altere o sobrenome do funcionário 3 para Drexler.

```
UPDATE my_employee
SET    last_name = 'Drexler'
WHERE  id = 3;
```

Exercício 8: Soluções (continuação)

11. Altere o salário de todos os funcionários com salário inferior a US\$ 900 para US\$ 1.000.

```
UPDATE my_employee
SET     salary = 1000
WHERE   salary < 900;
```

12. Verifique as alterações na tabela.

```
SELECT last_name, salary
FROM   my_employee;
```

13. Delete Betty Dancs da tabela MY_EMPLOYEE.

```
DELETE
FROM   my_employee
WHERE  last_name = 'Dancs';
```

14. Confirme as alterações na tabela.

```
SELECT *
FROM   my_employee;
```

15. Submeta todas as alterações pendentes a commit.

```
COMMIT;
```

Controle a transação de dados na tabela MY_EMPLOYEE.

16. Preencha a tabela com a última linha dos dados de amostra usando as instruções no script criado na etapa 6. Execute as instruções no script.

```
SET ECHO OFF
SET VERIFY OFF
INSERT INTO my_employee
VALUES (&p_id, '&p_last_name', '&p_first_name',
       lower(substr('&p_first_name', 1, 1) ||
       substr('&p_last_name', 1, 7)), &p_salary);
SET VERIFY ON
SET ECHO ON
UNDEFINE p_first_name
UNDEFINE p_last_name
```

17. Confirme a adição à tabela.

```
SELECT *
FROM   my_employee;
```

Exercício 8: Soluções (continuação)

18. Marque um ponto intermediário no processamento da transação.

```
SAVEPOINT step_18;
```

19. Esvazie a tabela inteira.

```
DELETE  
FROM my_employee;
```

20. Confirme se a tabela está vazia.

```
SELECT *  
FROM my_employee;
```

21. Descarte a operação DELETE mais recente sem descartar a operação INSERT anterior.

```
ROLLBACK TO step_18;
```

22. Verifique se a nova linha permanece intacta.

```
SELECT *  
FROM my_employee;
```

23. Torne a adição de dados permanente.

```
COMMIT;
```


Exercício 9: Soluções

1. Crie a tabela DEPT com base no quadro de instâncias de tabela a seguir. Inclua a sintaxe no script lab_09_01.sql e execute a instrução no script para criar a tabela. Verifique se a tabela foi criada.

```
CREATE TABLE dept
(id    NUMBER(7) CONSTRAINT department_id_pk PRIMARY KEY,
name  VARCHAR2(25));

DESCRIBE dept
```

2. Preencha a tabela DEPT com dados da tabela DEPARTMENTS. Inclua somente as colunas necessárias.

```
INSERT INTO dept
SELECT  department_id, department_name
FROM    departments;
```

3. Crie a tabela EMP com base no quadro de instâncias de tabela a seguir. Inclua a sintaxe no script lab_09_03.sql e execute a instrução no script para criar a tabela. Verifique se a tabela foi criada.

```
CREATE TABLE emp
(id          NUMBER(7),
last_name    VARCHAR2(25),
first_name   VARCHAR2(25),
dept_id      NUMBER(7)
CONSTRAINT emp_dept_id_FK REFERENCES dept (id)
);

DESCRIBE emp
```

4. Crie a tabela EMPLOYEES2 com base na estrutura da tabela EMPLOYEES. Inclua somente as colunas EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY e DEPARTMENT_ID. Nomeie as colunas da nova tabela como ID, FIRST_NAME, LAST_NAME, SALARY e DEPT_ID, respectivamente.

```
CREATE TABLE employees2 AS
SELECT  employee_id id, first_name, last_name, salary,
        department_id dept_id
FROM    employees;
```

5. Elimine a tabela EMP.

```
DROP TABLE emp;
```

Exercício 10: Soluções

Parte 1

1. A equipe do departamento de recursos humanos deseja ocultar alguns dados da tabela EMPLOYEES. Ela deseja uma view denominada EMPLOYEES_VU com base nos números e nomes de funcionário, bem como nos números de departamento da tabela EMPLOYEES. Também deseja atribuir EMPLOYEE como o cabeçalho do nome do funcionário.

```
CREATE OR REPLACE VIEW employees_vu AS
  SELECT employee_id, last_name employee, department_id
  FROM employees;
```

2. Verifique se a view funciona. Exiba o conteúdo da view EMPLOYEES_VU.

```
SELECT    *
FROM      employees_vu;
```

3. Usando a view EMPLOYEES_VU, crie uma consulta para o departamento de recursos humanos a fim de exibir todos os nomes de funcionário e números de departamento.

```
SELECT    employee, department_id
FROM      employees_vu;
```

Exercício 10: Soluções (continuação)

Parte 1

4. O departamento 50 precisa de acesso aos dados de seus funcionários. Crie uma view denominada DEPT50 que contenha os números e os sobrenomes, bem como os números de departamento de todos os funcionários do departamento 50. A equipe solicitou a atribuição dos labels EMPNO, EMPLOYEE e DEPTNO às colunas da view. Para fins de segurança, não permita que um funcionário seja redesignado a outro departamento por meio da view.

```
CREATE VIEW dept50 AS

  SELECT    employee_id empno, last_name employee,
            department_id deptno
  FROM      employees
  WHERE     department_id = 50
  WITH CHECK OPTION CONSTRAINT emp_dept_50;
```

5. Exiba a estrutura e o conteúdo da view DEPT50.

```
DESCRIBE dept50

SELECT    *
FROM      dept50;
```

6. Teste a view. Tente redesignar Matos ao departamento 80.

```
UPDATE    dept50
SET        deptno = 80
WHERE     employee = 'Matos';
```

Exercício 10: Soluções (continuação)

Parte 2

7. Você precisa de uma sequência que possa ser usada com a coluna de chave primária da tabela DEPT. A sequência deve começar com o valor 200 e ter o valor máximo 1000. Incremente a sequência em 10. Nomeie-a como DEPT_ID_SEQ.

```
CREATE SEQUENCE dept_id_seq
  START WITH 200
  INCREMENT BY 10
  MAXVALUE 1000;
```

8. Para testar a sequência, crie um script para inserir duas linhas na tabela DEPT. Nomeie o script como lab_10_08.sql. Certifique-se de utilizar a sequência criada para a coluna ID. Adicione dois departamentos: Education e Administration. Confirme as adições. Execute os comandos no script.

```
INSERT INTO dept
VALUES (dept_id_seq.nextval, 'Education');

INSERT INTO dept
VALUES (dept_id_seq.nextval, 'Administration');
```

9. Crie um índice não exclusivo na coluna DEPT_ID da tabela DEPT.

```
CREATE INDEX dept_name_idx ON dept (name);
```

10. Crie um sinônimo para a tabela EMPLOYEES. Nomeie o sinônimo como EMP.

```
CREATE SYNONYM emp FOR EMPLOYEES;
```

Exercício 11: Soluções

1. Para uma tabela especificada, crie um script que informe os nomes de colunas, os tipos de dados, os tamanhos dos tipos de dados e a permissão ou não de valores nulos. Solicite o nome da tabela ao usuário. Forneça apelidos apropriados às colunas DATA_PRECISION e DATA_SCALE. Salve esse script no arquivo lab_11_01.sql.

```
SELECT column_name, data_type, data_length,  
       data_precision PRECISION, data_scale SCALE, nullable  
FROM   user_tab_columns  
WHERE  table_name = UPPER('&tab_name');
```

2. Crie um script que informe o nome da coluna, o nome da constraint, o tipo de constraint, a condição de pesquisa e o status de uma tabela especificada. Junte as tabelas USER_CONSTRAINTS e USER_CONS_COLUMNS para obter todas essas informações. Solicite o nome da tabela ao usuário. Salve o script no arquivo lab_11_02.sql.

```
SELECT ucc.column_name, uc.constraint_name, uc.constraint_type,  
       uc.search_condition, uc.status  
FROM   user_constraints uc JOIN user_cons_columns ucc  
ON      uc.table_name = ucc.table_name  
AND     uc.constraint_name = ucc.constraint_name  
AND     uc.table_name = UPPER('&tab_name');
```

3. Adicione um comentário à tabela DEPARTMENTS. Em seguida, consulte a view USER_TAB_COMMENTS para verificar se o comentário está presente.

```
COMMENT ON TABLE departments IS  
    'Company department information including name, code, and location.';  
  
SELECT COMMENTS  
FROM   user_tab_comments  
WHERE  table_name = 'DEPARTMENTS';
```

4. Localize os nomes de todos os sinônimos existentes no esquema.

```
SELECT *  
FROM   user_synonyms;
```

Exercício 11: Soluções (continuação)

5. Você precisa determinar os nomes e as definições de todas as views do seu esquema. Crie um relatório que recupere informações (o texto e o nome da view) da view de dicionário de dados USER_VIEWS.

Observação: Já existe outra view. A view EMP_DETAILS_VIEW foi criada como parte do seu esquema.

Observação: Para exibir mais conteúdo de uma coluna LONG, use o comando SET LONG *n* do iSQL*Plus, em que *n* é o valor do número de caracteres da coluna LONG a ser exibida.

```
SET LONG 600

SELECT    view_name, text
FROM      user_views;
```

6. Obtenha os nomes das seqüências. Crie uma consulta em um script para exibir as seguintes informações sobre as seqüências: nome da seqüência, valor máximo, tamanho do incremento e último número. Nomeie o script como lab_11_06.sql. Execute a instrução no script.

```
SELECT    sequence_name, max_value, increment_by, last_number
FROM      user_sequences;
```

Exercício C: Soluções

1. Crie uma consulta para o departamento de recursos humanos a fim de obter os endereços de todos os departamentos. Use as tabelas LOCATIONS e COUNTRIES. Mostre o ID do local, o endereço, a cidade, o estado e o país na saída.

```
SELECT location_id, street_address, city, state_province, country_name
FROM   locations, countries
WHERE  locations.country_id = countries.country_id;
```

2. O departamento de recursos humanos precisa de um relatório de todos os funcionários. Crie uma consulta para exibir o sobrenome, o número do departamento e o nome do departamento de todos os funcionários.

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE  e.department_id = d.department_id;
```

3. O departamento de recursos humanos precisa de um relatório dos funcionários em Toronto. Exiba o sobrenome, o cargo, o número do departamento e o nome do departamento de todos os funcionários que trabalham em Toronto.

```
SELECT e.last_name, e.job_id, e.department_id, d.department_name
FROM   employees e, departments d , locations l
WHERE  e.department_id = d.department_id
AND    d.location_id = l.location_id
AND    LOWER(l.city) = 'toronto';
```

4. Crie um relatório para exibir o sobrenome e o número dos funcionários, bem como o sobrenome e o número dos respectivos gerentes. Atribua às colunas os labels Employee, Emp#, Manager e Mgr#, respectivamente. Inclua a instrução SQL no arquivo de texto lab_c_04.sql.

```
SELECT w.last_name "Employee", w.employee_id "EMP#",
       m.last_name "Manager", m.employee_id  "Mgr#"
FROM   employees w, employees m
WHERE  w.manager_id = m.employee_id;
```

5. Modifique lab_c_04.sql para exibir todos os funcionários, inclusive King, que não possui gerente. Ordene os resultados pelo número do funcionário. Inclua a instrução SQL no arquivo de texto lab_c_05.sql. Execute a consulta em lab_c_05.sql.

```
SELECT w.last_name "Employee", w.employee_id "EMP#",
       m.last_name "Manager", m.employee_id  "Mgr#"
FROM   employees w, employees m
WHERE  w.manager_id = m.employee_id (+);
```

Exercício C: Soluções (continuação)

6. Crie um relatório para o departamento de recursos humanos que exiba os sobrenomes e os números de departamento dos funcionários, bem como todos os funcionários que trabalham no mesmo departamento como um funcionário específico. Atribua um label apropriado a cada coluna. Salve o script no arquivo `lab_c_06.sql`.

```
SELECT e.department_id department, e.last_name employee,  
       c.last_name colleague  
FROM   employees e, employees c  
WHERE  e.department_id = c.department_id  
AND    e.employee_id <> c.employee_id  
ORDER BY e.department_id, e.last_name, c.last_name;
```

7. O departamento de recursos humanos precisa de um relatório sobre níveis de cargos e salários. Para se familiarizar com a tabela `JOB_GRADES`, primeiro mostre a estrutura dessa tabela. Em seguida, crie uma consulta que exiba o nome, o cargo, o nome do departamento, o salário e o nível de todos os funcionários.

```
DESC JOB_GRADES  
  
SELECT e.last_name, e.job_id, d.department_name,  
       e.salary, j.grade_level  
FROM   employees e, departments d, job_grades j  
WHERE  e.department_id = d.department_id  
AND    e.salary BETWEEN j.lowest_sal AND j.highest_sal;
```

Se deseja mais um desafio, faça estes exercícios:

8. O departamento de recursos humanos deseja determinar os nomes de todos os funcionários admitidos após Davies. Crie uma consulta para exibir o nome e a data de admissão de todos os funcionários admitidos após Davies.

```
SELECT e.last_name, e.hire_date  
FROM   employees e , employees davies  
WHERE  davies.last_name = 'Davies'  
AND    davies.hire_date < e.hire_date;
```


Exercício C: Soluções (continuação)

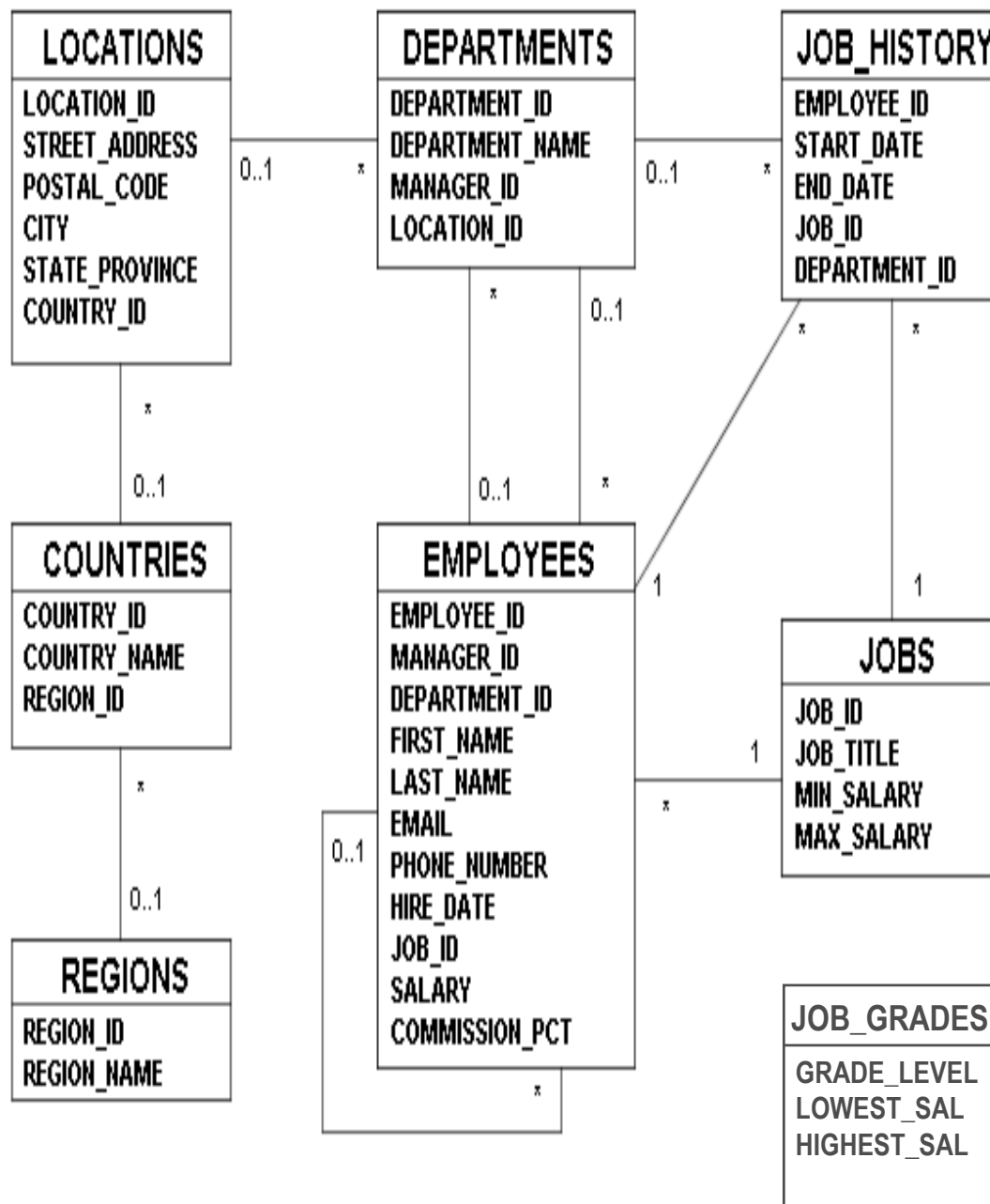
9. O departamento de recursos humanos precisa obter os nomes e as datas de admissão de todos os funcionários admitidos antes dos respectivos gerentes, além dos nomes e das datas de admissão desses gerentes. Atribua às colunas os labels Employee, Emp Hired, Manager e Mgr Hired, respectivamente. Salve o script no arquivo lab_c_09.sql.

```
SELECT w.last_name, w.hire_date, m.last_name, m.hire_date
FROM   employees w , employees m
WHERE  w.manager_id = m.employee_id
AND    w.hire_date <  m.hire_date;
```

B

Dados e Descrições de Tabelas

Conjunto de Dados de HR (Human Resources)



Conjunto de Dados de HR (Human Resources)

O esquema HR (Human Resources) faz parte do Esquema Comum Oracle que pode ser instalado em um banco de dados Oracle. Os exercícios deste curso utilizam os dados do esquema HR.

Descrições de Tabelas

REGIONS contém linhas que representam uma região (como Américas, Ásia e outras).

COUNTRIES contém linhas de países, cada um associado a uma região.

LOCATIONS contém os endereços de escritórios, depósitos e/ou locais de produção específicos de uma empresa em determinado país.

DEPARTMENTS mostra detalhes dos departamentos onde os funcionários trabalham. Cada departamento pode ter um relacionamento que representa seu respectivo gerente na tabela EMPLOYEES.

EMPLOYEES contém detalhes de cada funcionário que trabalha em um departamento. Alguns funcionários não podem ser designados a departamento algum.

JOBS contém os tipos de cargo que podem ser ocupados por cada funcionário.

JOB_HISTORY contém o histórico de cargos dos funcionários. Se um funcionário mudar de departamento no mesmo cargo ou mudar de cargo no departamento, uma nova linha será inserida nesta tabela com as informações do cargo antigo do funcionário.

JOB_GRADES identifica uma faixa salarial por nível de cargo. As faixas salariais não se sobrepõem.

Tabela COUNTRIES

```
DESCRIBE countries
```

Name	Null?	Type
COUNTRY_ID	NOT NULL	CHAR(2)
COUNTRY_NAME		VARCHAR2(40)
REGION_ID		NUMBER

```
SELECT * FROM countries;
```

CO	COUNTRY_NAME	REGION_ID
CA	Canada	2
DE	Germany	1
UK	United Kingdom	1
US	United States of America	2

Tabela DEPARTMENTS

DESCRIBE departments

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

SELECT * FROM departments;

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

8 rows selected.

Tabela EMPLOYEES

DESCRIBE employees

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE
100	Steven	King	SKING	515.123.4567	17-JUN-87
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93
103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90
104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-91
107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-99
124	Kevin	Mourgos	KMOURGOS	650.123.5234	16-NOV-99
141	Trenna	Rajs	TRAJS	650.121.8009	17-OCT-95
142	Curtis	Davies	CDAVIES	650.121.2994	29-JAN-97
143	Randall	Matos	RMATOS	650.121.2874	15-MAR-98
144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-98
149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-00
174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-96
176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	24-MAR-98
178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-MAY-99
200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-87
201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-96
202	Pat	Fay	PFAY	603.123.6666	17-AUG-97
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94
206	William	Gietz	WGIEZT	515.123.8181	07-JUN-94

20 rows selected.

Banco de Dados Oracle 10g: Fundamentos de SQL I B-6

Tabela EMPLOYEES (continuação)

JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
AD_PRES	24000			90
AD_VP	17000		100	90
AD_VP	17000		100	90
IT_PROG	9000		102	60
IT_PROG	6000		103	60
IT_PROG	4200		103	60
ST_MAN	5800		100	50
ST_CLERK	3500		124	50
ST_CLERK	3100		124	50
ST_CLERK	2600		124	50
ST_CLERK	2500		124	50
SA_MAN	10500	.2	100	80
SA_REP	11000	.3	149	80
SA_REP	8600	.2	149	80
SA_REP	7000	.15	149	
AD_ASST	4400		101	10
MK_MAN	13000		100	20
MK_REP	6000		201	20
AC_MGR	12000		101	110
AC_ACCOUNT	8300		205	110

20 rows selected.

Tabela JOBS

```
DESCRIBE jobs
```

Name	Null?	Type
JOB_ID	NOT NULL	VARCHAR2(10)
JOB_TITLE	NOT NULL	VARCHAR2(35)
MIN_SALARY		NUMBER(6)
MAX_SALARY		NUMBER(6)

```
SELECT * FROM jobs;
```

JOB_ID	JOB_TITLE	MIN_SALARY	MAX_SALARY
AD_PRES	President	20000	40000
AD_VP	Administration Vice President	15000	30000
AD_ASST	Administration Assistant	3000	6000
AC_MGR	Accounting Manager	8200	16000
AC_ACCOUNT	Public Accountant	4200	9000
SA_MAN	Sales Manager	10000	20000
SA_REP	Sales Representative	6000	12000
ST_MAN	Stock Manager	5500	8500
ST_CLERK	Stock Clerk	2000	5000
IT_PROG	Programmer	4000	10000
MK_MAN	Marketing Manager	9000	15000
MK_REP	Marketing Representative	4000	9000

12 rows selected.

Tabela JOB_GRADES

```
DESCRIBE job_grades
```

Name	Null?	Type
GRADE_LEVEL		VARCHAR2(3)
LOWEST_SAL		NUMBER
HIGHEST_SAL		NUMBER

```
SELECT * FROM job_grades;
```

GRA	LOWEST_SAL	HIGHEST_SAL
A	1000	2999
B	3000	5999
C	6000	9999
D	10000	14999
E	15000	24999
F	25000	40000

6 rows selected.

Tabela JOB_HISTORY

```
DESCRIBE job_history
```

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
START_DATE	NOT NULL	DATE
END_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
DEPARTMENT_ID		NUMBER(4)

```
SELECT * FROM job_history;
```

EMPLOYEE_ID	START_DAT	END_DATE	JOB_ID	DEPARTMENT_ID
102	13-JAN-93	24-JUL-98	IT_PROG	60
101	21-SEP-89	27-OCT-93	AC_ACCOUNT	110
101	28-OCT-93	15-MAR-97	AC_MGR	110
201	17-FEB-96	19-DEC-99	MK_REP	20
114	24-MAR-98	31-DEC-99	ST_CLERK	50
122	01-JAN-99	31-DEC-99	ST_CLERK	50
200	17-SEP-87	17-JUN-93	AD_ASST	90
176	24-MAR-98	31-DEC-98	SA_REP	80
176	01-JAN-99	31-DEC-99	SA_MAN	80
200	01-JUL-94	31-DEC-98	AC_ACCOUNT	90

10 rows selected.

Tabela LOCATIONS

DESCRIBE locations

Name	Null?	Type
LOCATION_ID	NOT NULL	NUMBER(4)
STREET_ADDRESS		VARCHAR2(40)
POSTAL_CODE		VARCHAR2(12)
CITY	NOT NULL	VARCHAR2(30)
STATE_PROVINCE		VARCHAR2(25)
COUNTRY_ID		CHAR(2)

SELECT * FROM locations;

LOCATION_ID	STREET_ADDRESS	POSTAL_CODE	CITY	STATE_PROVINCE	CO
1400	2014 Jabberwocky Rd	26192	Southlake	Texas	US
1500	2011 Interiors Blvd	99236	South San Francisco	California	US
1700	2004 Charade Rd	98199	Seattle	Washington	US
1800	460 Bloor St. W.	ON M5S 1X8	Toronto	Ontario	CA
2500	Magdalen Centre, The Oxford Science Park	OX9 9ZB	Oxford	Oxford	UK

Tabela REGIONS

```
DESCRIBE regions
```

Name	Null?	Type
REGION_ID	NOT NULL	NUMBER
REGION_NAME		VARCHAR2(25)

```
SELECT * FROM regions;
```

REGION_ID	REGION_NAME
1	Europe
2	Americas
3	Asia
4	Middle East and Africa

Sintaxe de Join Oracle

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Ao concluir esta lição, você será capaz de:

- Criar instruções **SELECT** para acessar dados de mais de uma tabela com equijoins e não-equijoins
- Usar joins externas para exibir dados que geralmente não atendem a uma condição de join
- Juntar uma tabela a si própria com uma auto-join

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Esta lição explica como obter dados de mais de uma tabela. Uma *join* é usada para exibir informações de várias tabelas. Portanto, você pode *juntar* tabelas para exibir informações de mais de uma tabela.

Observação: Para obter informações sobre joins, consulte “SQL Queries and Subqueries: Joins” no manual *Oracle SQL Reference*.

Obtendo Dados de Várias Tabelas

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
...		
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700



EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
...		
102	90	Executive
205	110	Accounting
206	110	Accounting

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Dados de Várias Tabelas

Às vezes, é necessário usar dados de mais de uma tabela. No exemplo do slide, o relatório exibe dados de duas tabelas distintas:

- Os IDs de funcionário estão na tabela EMPLOYEES.
- Os IDs de departamento estão nas tabelas EMPLOYEES e DEPARTMENTS.
- Os nomes de departamento estão na tabela DEPARTMENTS.

Para gerar o relatório, você precisa vincular as tabelas EMPLOYEES e DEPARTMENTS e acessar os dados dessas duas tabelas.

Produtos Cartesianos

- **Um produto cartesiano será formado quando:**
 - Uma condição de join for omitida
 - Uma condição de join for inválida
 - Todas as linhas da primeira tabela se unirem a todas as linhas da segunda tabela
- **Para evitar um produto cartesiano, inclua sempre uma condição de join válida na cláusula WHERE.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Produtos Cartesianos

Quando uma condição de join é inválida ou completamente omitida, o resultado é um *produto cartesiano*, no qual todas as combinações de linhas são exibidas. Todas as linhas da primeira tabela são unidas a todas as linhas da segunda tabela.

Um produto cartesiano tende a gerar um grande número de linhas e o resultado raramente é útil. Inclua sempre uma condição de join válida, a menos que exista uma necessidade específica de combinar todas as linhas de todas as tabelas.

Os produtos cartesianos são úteis em alguns testes quando é necessário gerar muitas linhas para simular um volume razoável de dados.

Gerando um Produto Cartesiano

EMPLOYEES (20 linhas)

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
...		
202	Fay	20
205	Higgins	110
206	Gietz	110

20 rows selected.

DEPARTMENTS (8 linhas)

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700

8 rows selected.

**Produto cartesiano:
20 x 8 = 160 linhas**

EMPLOYEE_ID	DEPARTMENT_ID	LOCATION_ID
100	90	1700
101	90	1700
102	90	1700
103	60	1700
104	60	1700
107	60	1700
...		

160 rows selected.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Produtos Cartesianos (continuação)

Um produto cartesiano será gerado se uma condição de join for omitida. O exemplo do slide exibe o sobrenome e o nome do departamento dos funcionários com base nas tabelas EMPLOYEES e DEPARTMENTS. Como não foi especificada uma condição de join, todas as linhas (20) da tabela EMPLOYEES são unidas a todas as linhas (8) da tabela DEPARTMENTS, gerando 160 linhas na saída.

```
SELECT last_name, department_name dept_name
FROM employees, departments;
```

LAST_NAME	DEPT_NAME
King	Administration
Kochhar	Administration
De Haan	Administration

...

160 rows selected.

Tipos de Join

Joins de propriedade Oracle (8i e releases anteriores)

- Equijoin
- Não-equijoin
- Join externa
- Auto-join

Joins compatíveis com o SQL:1999

- Join cruzada
- Join natural
- Cláusula USING
- Join externa integral
(ou de dois lados)
- Condição arbitrária
de join para join
externa

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Tipos de Join

Antes da release do Banco de Dados Oracle9i, a sintaxe de join era proprietária.

Observação: A sintaxe de join compatível com o SQL:1999 não oferece benefícios de desempenho em relação à sintaxe de join de propriedade Oracle existente nas releases anteriores. Para obter informações detalhadas sobre a sintaxe de join compatível com o SQL:1999, consulte a Lição 5.

Unindo Tabelas com a Sintaxe Oracle

Use uma join para consultar dados de mais de uma tabela:

```
SELECT    table1.column, table2.column
FROM      table1, table2
WHERE     table1.column1 = table2.column2;
```

- **Crie a condição de join na cláusula WHERE.**
- **Adicione o nome da tabela como prefixo ao nome da coluna quando o mesmo nome de coluna aparecer em mais de uma tabela.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Definindo Joins

Quando são necessários dados de mais de uma tabela do banco de dados, é usada uma condição de *join*. É possível unir linhas de uma tabela a linhas de outra, de acordo com valores comuns existentes em colunas correspondentes (isto é, geralmente colunas de chave primária e estrangeira).

Para exibir os dados de duas ou mais tabelas relacionadas, crie uma condição de join simples na cláusula WHERE.

Na sintaxe:

table1.column indica a tabela e a coluna das quais os dados são recuperados
table1.column1 = é a condição que une (ou relaciona) as tabelas
table2.column2

Diretrizes

- Quando criar uma instrução SELECT para unir tabelas, preceda o nome da coluna pelo nome da tabela por uma questão de clareza e para melhorar o acesso ao banco de dados.
- Se o mesmo nome de coluna aparecer em mais de uma tabela, adicione o nome da tabela como um prefixo ao nome da coluna.
- Para unir *n* tabelas, você precisará de, pelo menos, *n* - 1 condições de join. Por exemplo, para unir quatro tabelas, é necessário um mínimo de três joins. Essa regra não será aplicada se a tabela tiver uma chave primária concatenada; nesse caso, será necessário mais de uma coluna para identificar cada linha com exclusividade.

Equijoins

EMPLOYEES

EMPLOYEE_ID	DEPARTMENT_ID
200	10
201	20
202	20
124	50
141	50
142	50
143	50
144	50
103	60
104	60
107	60
149	80
174	80
176	80

...

Chave estrangeira

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
20	Marketing
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
60	IT
60	IT
60	IT
80	Sales
80	Sales
80	Sales

...

Chave primária

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Equijoins

Para determinar o nome do departamento de um funcionário, compare o valor da coluna `DEPARTMENT_ID` na tabela `EMPLOYEES` com os valores de `DEPARTMENT_ID` na tabela `DEPARTMENTS`. O relacionamento entre as tabelas `EMPLOYEES` e `DEPARTMENTS` é uma *equijoin* – isto é, os valores da coluna `DEPARTMENT_ID` nas duas tabelas devem ser iguais. Com frequência, esse tipo de join envolve complementos de chave primária e estrangeira.

Observação: As equijoins são chamadas de *joins simples* ou *joins internas*.

Recuperando Registros com Equijoins

```
SELECT employees.employee_id, employees.last_name,  
       employees.department_id, departments.department_id,  
       departments.location_id  
FROM   employees, departments  
WHERE  employees.department_id = departments.department_id;
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID	LOCATION_ID
200	Whalen	10	10	1700
201	Hartstein	20	20	1800
202	Fay	20	20	1800
124	Mourgos	50	50	1500
141	Rajs	50	50	1500
142	Davies	50	50	1500
143	Matos	50	50	1500
144	Vargas	50	50	1500

19 rows selected.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Recuperando Registros com Equijoins

No exemplo do slide:

- A cláusula **SELECT** especifica os nomes das colunas a serem recuperadas:
 - Sobrenome do funcionário, número do funcionário e número do departamento, que são colunas da tabela **EMPLOYEES**
 - Número do departamento, nome do departamento e ID do local, que são colunas da tabela **DEPARTMENTS**
- A cláusula **FROM** especifica as duas tabelas que o banco de dados deve acessar:
 - Tabela **EMPLOYEES**
 - Tabela **DEPARTMENTS**
- A cláusula **WHERE** especifica como as tabelas serão unidas:

EMPLOYEES.DEPARTMENT_ID = DEPARTMENTS.DEPARTMENT_ID

Como a coluna **DEPARTMENT_ID** é comum às duas tabelas, é necessário adicionar o nome da tabela como prefixo ao nome dessa coluna para evitar ambigüidade.

Condições Adicionais de Pesquisa com o Operador AND

EMPLOYEES

LAST_NAME	DEPARTMENT_ID
Whalen	10
Hartstein	20
Fay	20
Mourgos	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Hunold	60
Ernst	60

...

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
20	Marketing
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
60	IT
60	IT

...

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Condições Adicionais de Pesquisa

Além da join, você pode especificar critérios na cláusula WHERE para restringir as linhas a serem consideradas em uma ou mais tabelas da join. Por exemplo, para exibir o número e o nome do departamento do funcionário Matos, você precisa de uma condição adicional na cláusula WHERE.

```
SELECT last_name, employees.department_id,
       department_name
FROM   employees, departments
WHERE  employees.department_id = departments.department_id
AND    last_name = 'Matos';
```

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Matos	50	Shipping

Qualificando Nomes de Colunas Ambíguos

- Use prefixos de tabela para qualificar nomes de colunas presentes em várias tabelas.
- Use prefixos de tabela para melhorar o desempenho.
- Use apelidos de colunas para diferenciar colunas com nomes idênticos, mas localizadas em tabelas distintas.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Qualificando Nomes de Colunas Ambíguos

Você precisa qualificar os nomes das colunas na cláusula WHERE com o nome da tabela para evitar ambigüidades. Sem os prefixos das tabelas, a coluna DEPARTMENT_ID poderá ser originária da tabela DEPARTMENTS ou EMPLOYEES. É necessário adicionar o prefixo da tabela para executar a consulta.

Se não houver nomes de colunas comuns entre as duas tabelas, não será preciso qualificar as colunas. No entanto, o uso do prefixo da tabela melhora o desempenho, pois você informa ao servidor Oracle exatamente onde localizar as colunas.

O requisito para qualificar nomes de colunas ambíguos também é aplicado a colunas que possam gerar ambigüidade em outras cláusulas, como a cláusula SELECT ou ORDER BY.

Usando Apelidos de Tabelas

- Use apelidos de tabelas para simplificar consultas.
- Use prefixos de tabela para melhorar o desempenho.

```
SELECT e.employee_id, e.last_name, e.department_id,  
       d.department_id, d.location_id  
FROM   employees e, departments d  
WHERE  e.department_id = d.department_id;
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Usando Apelidos de Tabelas

A qualificação dos nomes de colunas com nomes de tabelas pode consumir muito tempo, especialmente se os *nomes das tabelas* forem longos. Você pode usar os apelidos das tabelas em vez dos nomes. Assim como um apelido de coluna fornece outro nome a uma coluna, um apelido de tabela fornece um outro nome a uma tabela. Os apelidos de tabelas ajudam a reduzir o tamanho do código SQL, utilizando menos memória.

Observe como os apelidos de tabelas são identificados na cláusula FROM do exemplo. O nome da tabela é especificado por inteiro, seguido por um espaço e, depois, o apelido da tabela. A tabela EMPLOYEES recebeu o apelido e, e a tabela DEPARTMENTS, o apelido d.

Diretrizes

- Um apelido de tabela pode conter até 30 caracteres, mas é recomendável especificar o menor nome possível.
- Se um apelido de tabela for usado para um nome de tabela específico na cláusula FROM, ele deverá ser substituído pelo nome da tabela em toda a instrução SELECT.
- Os apelidos de tabelas devem ser significativos.
- Um apelido de tabela é válido somente para a instrução SELECT atual.

Unindo Mais de Duas Tabelas

EMPLOYEES

LAST_NAME	DEPARTMENT_ID
King	90
Kochhar	90
De Haan	90
Hunold	60
Ernst	60
Lorentz	60
Mourgos	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Zlotkey	80
Abel	80
Taylor	80

...
20 rows selected.

DEPARTMENTS

DEPARTMENT_ID	LOCATION_ID
10	1700
20	1800
50	1500
60	1400
80	2500
90	1700
110	1700
190	1700

8 rows selected.

LOCATIONS

LOCATION_ID	CITY
1400	Southlake
1500	South San Francisco
1700	Seattle
1800	Toronto
2500	Oxford

Para unir n tabelas, você precisa de, pelo menos, $n-1$ condições de join. Por exemplo, para unir três tabelas, são necessárias, no mínimo, duas joins.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Condições Adicionais de Pesquisa

Às vezes, talvez seja necessário unir mais de duas tabelas. Por exemplo, para exibir o sobrenome, o nome do departamento e a cidade de cada funcionário, você precisa unir as tabelas EMPLOYEES, DEPARTMENTS e LOCATIONS.

```
SELECT e.last_name, d.department_name, l.city
FROM   employees e, departments d, locations l
WHERE  e.department_id = d.department_id
AND    d.location_id = l.location_id;
```

LAST_NAME	DEPARTMENT_NAME	CITY
Hunold	IT	Southlake
Ernst	IT	Southlake
Lorentz	IT	Southlake
Mourgos	Shipping	South San Francisco
Rajs	Shipping	South San Francisco
Davies	Shipping	South San Francisco

...
19 rows selected.

Não-Equijoins

EMPLOYEES

LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000
Hunold	9000
Ernst	6000
Lorentz	4200
Mourgos	5800
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500
Zlotkey	10500
Abel	11000
Taylor	8600

...
20 rows selected.

JOB_GRADES

GRA	LOWEST_SAL	HIGHEST_SAL
A	1000	2999
B	3000	5999
C	6000	9999
D	10000	14999
E	15000	24999
F	25000	40000

← O salário na tabela **EMPLOYEES** deve estar compreendido entre o menor e o maior salário na tabela **JOB_GRADES**.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Não-Equijoins

Uma não-equijoin é uma condição de join que contém algo diferente de um operador de igualdade.

O relacionamento entre as tabelas **EMPLOYEES** e **JOB_GRADES** é um exemplo de uma não-equijoin. Em um relacionamento entre as duas tabelas, os valores da coluna **SALARY** da tabela **EMPLOYEES** devem estar compreendidos entre os valores das colunas **LOWEST_SALARY** e **HIGHEST_SALARY** da tabela **JOB_GRADES**. O relacionamento é obtido com um operador diferente de igualdade (=).

Recuperando Registros com Não-Equijoins

```
SELECT e.last_name, e.salary, j.grade_level
FROM   employees e, job_grades j
WHERE  e.salary
      BETWEEN j.lowest_sal AND j.highest_sal;
```

LAST_NAME	SALARY	GRA
Matos	2600	A
Vargas	2500	A
Lorentz	4200	B
Mourgos	5800	B
Rajs	3500	B
Davies	3100	B
Whalen	4400	B
Hunold	9000	C
Ernst	6000	C

...
20 rows selected.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Não-Equijoins (continuação)

O exemplo do slide cria uma não-equijoin para avaliar o nível salarial de um funcionário. O salário deve estar compreendido *entre* qualquer par de faixas de salário mais baixo e mais alto.

É importante observar que todos os funcionários aparecem uma única vez quando essa consulta é executada. Os funcionários não são repetidos na lista. Existem dois motivos para isso:

- Nenhuma das linhas da tabela de níveis de cargos contém níveis sobrepostos. Isto é, o valor do salário de um funcionário somente pode estar entre os valores de salário mais alto e mais baixo de uma das linhas da tabela de níveis salariais.
- Os salários de todos os funcionários estão compreendidos entre os limites fornecidos pela tabela de níveis de cargos. Isto é, nenhum funcionário recebe menos que o valor mais baixo contido na coluna `LOWEST_SAL` ou mais que o valor mais alto contido na coluna `HIGHEST_SAL`.

Observação: É possível usar outras condições (como `<=` e `>=`), mas `BETWEEN` é a mais simples. Quando usar `BETWEEN`, lembre-se de informar primeiro o valor mais baixo e depois o valor mais alto.

Foram especificados apelidos de tabelas no exemplo do slide por questões de desempenho, e não para evitar uma possível ambigüidade.

Joins Externas

DEPARTMENTS

DEPARTMENT_NAME	DEPARTMENT_ID
Administration	10
Marketing	20
Shipping	50
IT	60
Sales	80
Executive	90
Accounting	110
Contracting	190

8 rows selected.

EMPLOYEES

DEPARTMENT_ID	LAST_NAME
90	King
90	Kochhar
90	De Haan
60	Hunold
60	Ernst
60	Lorentz
50	Mourgos
50	Rajs
50	Davies
50	Matos
50	Vargas
80	Zlotkey

...
20 rows selected.

**Não há funcionários
no departamento 190.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Retornando Registros sem Correspondência Direta com Joins Externas

Se não atender a uma condição de join, a linha não aparecerá no resultado da consulta. Por exemplo, na condição de equijoin das tabelas EMPLOYEES e DEPARTMENTS, o nome da funcionária Grant não é exibido, pois não existe um ID de departamento para ela na tabela EMPLOYEES. Em vez de conter 20 funcionários, o conjunto de resultados conterá 19 registros.

```
SELECT e.last_name, e.department_id, d.department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id;
```

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	10	Administration
Hartstein	20	Marketing
Fay	20	Marketing
Mourgos	50	Shipping

...
19 rows selected.

Sintaxe de Joins Externas

- Use uma join externa para ver as linhas que não atendem à condição de join.
- O operador de join externa é o sinal de adição (+).

```
SELECT table1.column, table2.column  
FROM   table1, table2  
WHERE  table1.column(+) = table2.column;
```

```
SELECT table1.column, table2.column  
FROM   table1, table2  
WHERE  table1.column = table2.column(+);
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Usando Joins Externas para Retornar Registros sem Correspondência Direta

Para retornar as linhas sem correspondência, utilize um operador de *join externa* na condição de join. O operador é um sinal de adição entre parênteses (+), colocado ao “lado” da join em que há falta de informações. Esse operador cria uma ou mais linhas nulas, às quais é possível unir uma ou mais linhas da tabela com registros correspondentes.

Na sintaxe:

table1.column = é a condição que une (ou relaciona) as tabelas

table2.column (+) é o símbolo de join externa, que pode ser colocado em qualquer lado da condição da cláusula WHERE, mas não em ambos. (Coloque o símbolo de join externa após o nome da coluna na tabela sem as linhas correspondentes.)

Usando Joins Externas

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE  e.department_id(+) = d.department_id ;
```

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	10	Administration
Hartstein	20	Marketing
Fay	20	Marketing
Mourgos	50	Shipping
Rajs	50	Shipping
Davies	50	Shipping
Matos	50	Shipping
...		
Gietz	110	Accounting
		Contracting

20 rows selected.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Usando Joins Externas para Retornar Registros sem Correspondência Direta (continuação)

O exemplo do slide exibe sobrenomes, IDs de departamento e nomes de departamento. O departamento Contracting não tem funcionários. Nenhum valor é mostrado na saída.

Restrições de Joins Externas

- O operador de join externa só pode aparecer em *um* lado da expressão – o lado no qual faltam informações. Ele retorna as linhas de uma tabela sem uma correspondência direta na outra tabela.
- Uma condição que envolva uma join externa não pode usar o operador IN ou ser vinculada a outra condição pelo operador OR.

Auto-Joins

EMPLOYEES (WORKER)

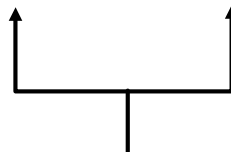
EMPLOYEE_ID	LAST_NAME	MANAGER_ID
100	King	
101	Kochhar	100
102	De Haan	100
103	Hunold	102
104	Ernst	103
107	Lorentz	103
124	Mourgos	100

...

EMPLOYEES (MANAGER)

EMPLOYEE_ID	LAST_NAME
100	King
101	Kochhar
102	De Haan
103	Hunold
104	Ernst
107	Lorentz
124	Mourgos

...



**MANAGER_ID na tabela WORKER é igual a
EMPLOYEE_ID na tabela MANAGER.**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Unindo uma Tabela a Ela Própria

Às vezes, é necessário unir uma tabela a ela própria. Para obter o nome do gerente de cada funcionário, você precisa unir a tabela EMPLOYEES a ela própria; esse tipo de join é denominado *auto-join*.

Por exemplo, para obter o nome do gerente de Lorentz, você precisa:

- Localizar Lorentz na tabela EMPLOYEES examinando a coluna LAST_NAME.
- Localizar o número do gerente de Lorentz examinando a coluna MANAGER_ID. O número do gerente de Lorentz é 103.
- Localizar o nome do gerente com o valor de EMPLOYEE_ID 103 examinando a coluna LAST_NAME. O número de funcionário de Hunold é 103, portanto, Hunold é o gerente de Lorentz.

Nesse processo, você examinará a tabela duas vezes. Na primeira vez, você examinará a tabela para localizar Lorentz na coluna LAST_NAME e o valor de MANAGER_ID 103. Na segunda vez, você examinará a coluna EMPLOYEE_ID para localizar 103 e a coluna LAST_NAME para localizar Hunold.

Unindo uma Tabela a Ela Própria

```
SELECT worker.last_name || ' works for '
       || manager.last_name
FROM   employees worker, employees manager
WHERE  worker.manager_id = manager.employee_id ;
```

WORKER.LAST_NAME 'WORKSFOR' MANAGER.LAST_NAME
Kochhar works for King
De Haan works for King
Mourgos works for King
Zlotkey works for King
Hartstein works for King
Whalen works for Kochhar
Higgins works for Kochhar
Hunold works for De Haan
Ernst works for Hunold
...

19 rows selected.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Unindo uma Tabela a Ela Própria (continuação)

O exemplo do slide une a tabela EMPLOYEES a ela própria. Para simular duas tabelas na cláusula FROM, existem dois apelidos, w e m, para a mesma tabela, EMPLOYEES.

Neste exemplo, a cláusula WHERE contém a join que significa “onde o número do gerente de um funcionário corresponde ao número de funcionário do gerente”.

Sumário

Neste apêndice, você aprendeu a usar joins para exibir dados de várias tabelas com a sintaxe de propriedade Oracle nas versões 8i e anteriores.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Sumário

Há várias maneiras de unir tabelas.

Tipos de Join

- Produtos cartesianos
- Equijoins
- Não-equijoins
- Joins externas
- Auto-joins

Produtos Cartesianos

Um produto cartesiano resulta na exibição de todas as combinações de linhas. Para obter esse resultado, omita a cláusula WHERE.

Apelidos de Tabelas

- Os apelidos de tabelas aceleram o acesso ao banco de dados.
- Eles podem ajudar a reduzir o código SQL, preservando a memória.

Exercício C: Visão Geral

Este exercício aborda a criação de consultas para unir tabelas usando a sintaxe Oracle.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Exercício C: Visão Geral

O objetivo deste exercício é proporcionar várias atividades de junção de tabelas com a sintaxe Oracle abordada neste apêndice.

Exercício C

1. Crie uma consulta para o departamento de recursos humanos a fim de gerar os endereços de todos os departamentos. Use as tabelas `LOCATIONS` e `COUNTRIES`. Mostre o ID do local, o endereço, a cidade, o estado e o país na saída. Use `NATURAL JOIN` para gerar os resultados.

LOCATION_ID	STREET_ADDRESS	CITY	STATE_PROVINCE	COUNTRY_NAME
1400	2014 Jabberwocky Rd	Southlake	Texas	United States of America
1500	2011 Interiors Blvd	South San Francisco	California	United States of America
1700	2004 Charade Rd	Seattle	Washington	United States of America
1800	460 Bloor St. W.	Toronto	Ontario	Canada
2500	Magdalen Centre, The Oxford Science Park	Oxford	Oxford	United Kingdom

2. O departamento de recursos humanos precisa de um relatório de todos os funcionários. Crie uma consulta para exibir o sobrenome, o número do departamento e o nome do departamento de todos os funcionários.

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	10	Administration
Hartstein	20	Marketing
Fay	20	Marketing
Mourgos	50	Shipping
Rajs	50	Shipping
Davies	50	Shipping
Vargas	50	Shipping
■ ■ ■		
De Haan	90	Executive
Higgins	110	Accounting
Gietz	110	Accounting

19 rows selected.

Exercício C (continuação)

- O departamento de recursos humanos precisa de um relatório dos funcionários baseados em Toronto. Exiba o sobrenome, o cargo, o número do departamento e o nome do departamento de todos os funcionários que trabalham em Toronto.

LAST_NAME	JOB_ID	DEPARTMENT_ID	DEPARTMENT_NAME
Hartstein	MK_MAN	20	Marketing
Fay	MK_REP	20	Marketing

- Crie um relatório para exibir o sobrenome e o número dos funcionários, bem como o sobrenome e o número dos respectivos gerentes. Atribua às colunas os labels Employee, Emp#, Manager e Mgr#, respectivamente. Inclua a instrução SQL no arquivo de texto lab_c_04.sql.

Employee	EMP#	Manager	Mgr#
Kochhar	101	King	100
De Haan	102	King	100
Mourgos	124	King	100
Zlotkey	149	King	100
Hartstein	201	King	100
Whalen	200	Kochhar	101
Higgins	205	Kochhar	101
Hunold	103	De Haan	102
Ernst	104	Hunold	103
Lorentz	107	Hunold	103
Rajs	141	Mourgos	124
Davies	142	Mourgos	124
Matos	143	Mourgos	124
Vargas	144	Mourgos	124
Employee	EMP#	Manager	Mgr#
Abel	174	Zlotkey	149
Taylor	176	Zlotkey	149
Grant	178	Zlotkey	149
Fay	202	Hartstein	201
Gietz	206	Higgins	205

19 rows selected.

Exercício C (continuação)

5. Modifique `lab_c_04.sql` para exibir todos os funcionários, inclusive King, que não possui gerente. Ordene os resultados pelo número do funcionário. Inclua a instrução SQL no arquivo de texto `lab_c_05.sql`. Execute a consulta em `lab_c_05.sql`.

Employee	EMP#	Manager	Mgr#
King	100		
Kochhar	101	King	100
De Haan	102	King	100
Hunold	103	De Haan	102
Ernst	104	Hunold	103
Lorentz	107	Hunold	103
Mourgos	124	King	100

■ ■ ■

20 rows selected.

6. Crie um relatório para o departamento de recursos humanos que exiba os sobrenomes e os números de departamento dos funcionários, bem como todos os funcionários que trabalham no mesmo departamento como um funcionário específico. Atribua um label apropriado a cada coluna. Salve o script no arquivo `lab_c_06.sql`.

DEPARTMENT	EMPLOYEE	COLLEAGUE
20	Fay	Hartstein
20	Hartstein	Fay
50	Davies	Matos
50	Davies	Mourgos
50	Davies	Rajs
50	Davies	Vargas
50	Matos	Davies
50	Matos	Mourgos
50	Matos	Rajs
50	Matos	Vargas
50	Mourgos	Davies
50	Mourgos	Matos
50	Mourgos	Rajs
50	Mourgos	Vargas

■ ■ ■

42 rows selected.

Exercício C (continuação)

7. O departamento de recursos humanos precisa de um relatório sobre níveis de cargos e salários. Para se familiarizar com a tabela `JOB_GRADES`, primeiro mostre a estrutura dessa tabela. Em seguida, crie uma consulta que exiba o sobrenome, o cargo, o nome do departamento, o salário e o nível de todos os funcionários.

Name	Null?	Type
GRADE_LEVEL		VARCHAR2(3)
LOWEST_SAL		NUMBER
HIGHEST_SAL		NUMBER

LAST_NAME	JOB_ID	DEPARTMENT_NAME	SALARY	GRA
Matos	ST_CLERK	Shipping	2600	A
Vargas	ST_CLERK	Shipping	2500	A
Lorentz	IT_PROG	IT	4200	B
Mourgos	ST_MAN	Shipping	5800	B
Rajs	ST_CLERK	Shipping	3500	B
Davies	ST_CLERK	Shipping	3100	B
Whalen	AD_ASST	Administration	4400	B

■ ■ ■

19 rows selected.

Se quiser tomar parte em mais um desafio, faça estes exercícios:

8. O departamento de recursos humanos deseja determinar os nomes de todos os funcionários admitidos após Davies. Crie uma consulta para exibir o nome e a data de admissão de todos os funcionários admitidos após Davies.

LAST_NAME	HIRE_DATE
Lorentz	07-FEB-99
Mourgos	16-NOV-99
Matos	15-MAR-98
Vargas	09-JUL-98
Zlotkey	29-JAN-00
Taylor	24-MAR-98
Grant	24-MAY-99
Fay	17-AUG-97

8 rows selected.

Exercício C (continuação)

9. O departamento de recursos humanos precisa obter o nome e a data de admissão de todos os funcionários admitidos antes dos respectivos gerentes, além do nome e da data de admissão desses gerentes. Atribua às colunas os labels Employee, Emp Hired, Manager e Mgr Hired, respectivamente. Salve o script no arquivo lab_c_09.sql.

LAST_NAME	HIRE_DATE	LAST_NAME	HIRE_DATE
Whalen	17-SEP-87	Kochhar	21-SEP-89
Hunold	03-JAN-90	De Haan	13-JAN-93
Rajs	17-OCT-95	Mourgos	16-NOV-99
Davies	29-JAN-97	Mourgos	16-NOV-99
Matos	15-MAR-98	Mourgos	16-NOV-99
Vargas	09-JUL-98	Mourgos	16-NOV-99
Abel	11-MAY-96	Zlotkey	29-JAN-00
Taylor	24-MAR-98	Zlotkey	29-JAN-00
Grant	24-MAY-99	Zlotkey	29-JAN-00

9 rows selected.

Usando o SQL*Plus

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

Ao concluir este apêndice, você será capaz de:

- **Efetuar login no SQL*Plus**
- **Editar comandos SQL**
- **Formatar a saída com comandos SQL*Plus**
- **Interagir com arquivos de script**

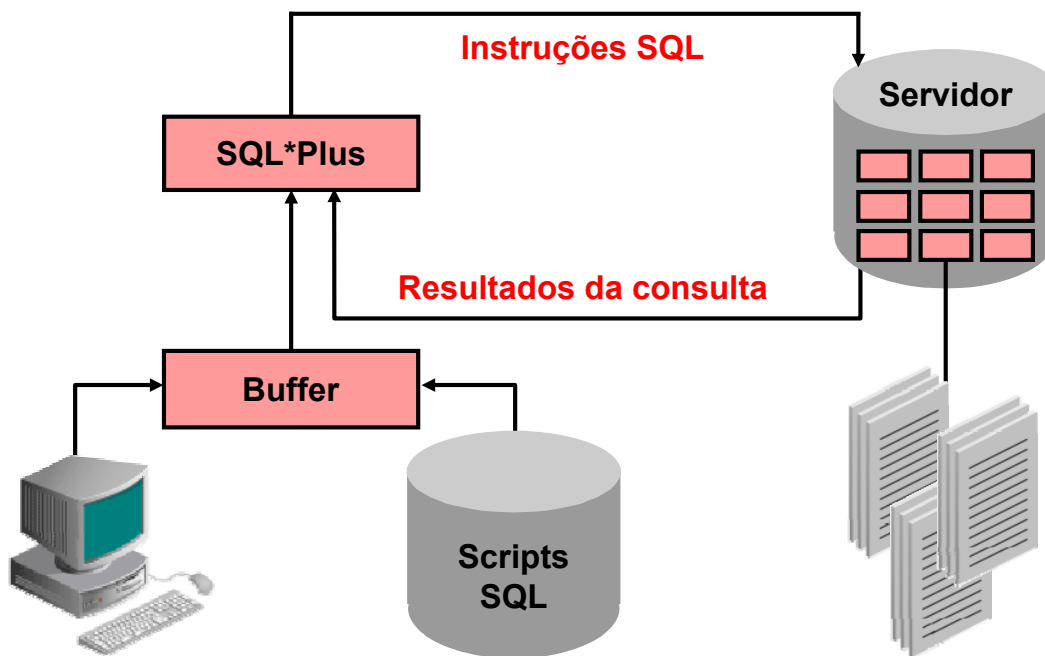
ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Objetivos

É possível criar instruções `SELECT` para uso contínuo. Este apêndice também aborda a utilização de comandos SQL*Plus para executar instruções SQL. Você aprenderá a formatar a saída com comandos SQL*Plus, editar comandos SQL e salvar scripts no SQL*Plus.

Interação entre SQL e SQL*Plus



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

SQL e SQL*Plus

SQL é uma linguagem de comandos para a comunicação com o Oracle9i Server a partir de qualquer ferramenta ou aplicação. O Oracle SQL contém várias extensões. Quando você informa uma instrução SQL, ela é armazenada em uma parte da memória denominada *buffer SQL* e permanece nesse local até que uma nova instrução SQL seja especificada. O SQL*Plus é uma ferramenta Oracle que reconhece as instruções SQL e as submete ao Oracle9i Server para execução. Ele contém sua própria linguagem de comandos.

Recursos de SQL

- Pode ser usada por vários usuários, incluindo aqueles com pouca ou nenhuma experiência em programação
- É uma linguagem não procedural
- Reduz o tempo necessário para criar e manter sistemas
- É uma linguagem semelhante ao idioma inglês

Recursos de SQL*Plus

- Aceita entrada ad hoc de instruções
- Aceita entrada SQL de arquivos
- Fornece um editor de linha para a modificação de instruções SQL
- Controla definições de ambiente
- Formata resultados de consultas em relatórios básicos
- Acessa bancos de dados locais e remotos

Instruções SQL e Comandos SQL*Plus

SQL

- Uma linguagem
- padrão ANSI
- Não é possível abreviar palavras-chave.
- As instruções manipulam dados e definições de tabelas no banco de dados.

Instruções SQL



Buffer SQL



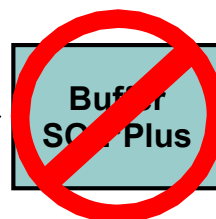
SQL*Plus

- Um ambiente
- Propriedade Oracle
- É possível abreviar palavras-chave.
- Os comandos não permitem a manipulação de valores no banco de dados.

Comandos SQL*Plus



Buffer SQL*Plus



ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

SQL e SQL*Plus (continuação)

Esta tabela compara SQL e SQL*Plus:

SQL	SQL*Plus
É uma linguagem para a comunicação com o servidor Oracle a fim de acessar os dados	Reconhece instruções SQL e as envia ao servidor
É baseada no padrão ANSI (American National Standards Institute) SQL	É a interface proprietária da Oracle para a execução de instruções SQL
Manipula dados e definições de tabelas no banco de dados	Não permite a manipulação de valores no banco de dados
É incluída no buffer SQL em uma ou mais linhas	É incluído em uma linha de cada vez; não é armazenado no buffer SQL
Não tem um caractere de continuação	Utiliza um traço (–) como caractere de continuação se o comando ultrapassa uma linha
Não pode ser abreviada	Pode ser abreviado
Utiliza um caractere de finalização para executar os comandos imediatamente	Não requer caracteres de finalização; executa os comandos imediatamente
Utiliza functions para aplicar formatação	Utiliza comandos para formatar dados

Visão Geral do SQL*Plus

- Fazer login no SQL*Plus.
- Descrever a estrutura de tabelas.
- Editar uma instrução SQL.
- Executar uma instrução SQL no SQL*Plus.
- Salvar instruções SQL em arquivos e anexar instruções SQL a arquivos.
- Executar os arquivos salvos.
- Carregar comandos do arquivo para o buffer e editá-los.

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

SQL*Plus

O SQL*Plus é um ambiente no qual você pode:

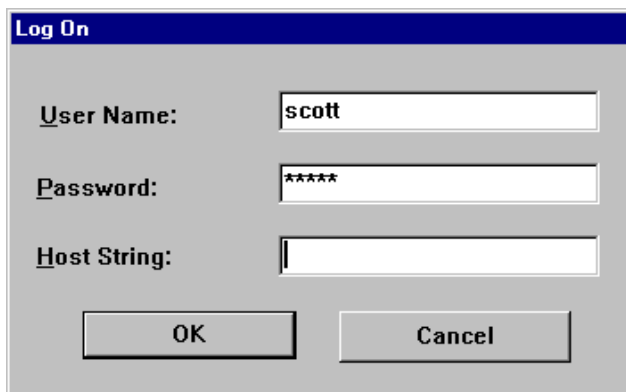
- Executar instruções SQL para recuperar, modificar, adicionar e remover dados no banco de dados
- Formatar, armazenar e imprimir resultados de consultas na forma de relatórios, bem como realizar cálculos com base nesses resultados
- Criar arquivos de script a fim de armazenar instruções SQL para uso futuro

É possível dividir os comandos SQL*Plus nestas principais categorias:

Categoria	Objetivo
Ambiente	Afeta o comportamento geral das instruções SQL na sessão
Formato	Formata os resultados da consulta
Manipulação de arquivos	Salva, carrega e executa arquivos de script
Execução	Envia instruções SQL do buffer SQL para o servidor Oracle
Edição	Modifica as instruções SQL no buffer
Interação	Cria e especifica variáveis para instruções SQL, imprime valores de variáveis e imprime mensagens na tela
Diversos	Conecta-se ao banco de dados, manipula o ambiente SQL*Plus e exibe definições de colunas

Efetuando Login no SQL*Plus

- De um ambiente Windows:



- De uma linha de comandos:

```
sqlplus [username[/password  
[@database]]]
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Efetuando Login no SQL*Plus

O modo de acesso ao SQL*Plus depende do tipo de sistema operacional ou do ambiente Windows em execução.

Para efetuar login de um ambiente Windows:

1. Selecione Start > Programs > Oracle > Application Development > SQL*Plus.
2. Informe o nome do usuário, a senha e o nome do banco de dados.

Para efetuar login de um ambiente de linha de comandos:

1. Efetue logon na sua máquina.
2. Informe o comando SQL*Plus mostrado no slide.

Na sintaxe:

username Seu nome de usuário do banco de dados
Password Sua senha do banco de dados (Ela estará visível se você informá-la aqui.)
@database SA string de conexão do banco de dados

Observação: Para garantir a integridade da senha, não a informe no prompt do sistema operacional. Nesse caso, informe somente seu nome de usuário. Informe a senha no prompt de senha.

Após o login no SQL*Plus, você verá a seguinte mensagem (se estiver usando o SQL*Plus versão 9i):

SQL*Plus: Release 9.0.1.0.0 - Development on Tue Jan 9 08:44:28 2001

(c) Copyright 2000 Oracle Corporation. All rights reserved.

Development Program (WDP) eKit materials are provided for WDP in-class use only. Copying eKit materials is strictly prohibited and is in violation of Oracle copyright. All WDP eKit materials must be used as eKit materials and not be used as eKit materials with their name and e-mail. Contact OracleWDP_ww@oracle.com if you have not received your personalized eKit.

Exibindo a Estrutura de Tabelas

Use o comando SQL*Plus **DESCRIBE** para exibir a estrutura de uma tabela:

```
DESC[RIBE] tablename
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Exibindo a Estrutura de Tabelas

No SQL*Plus, você pode exibir a estrutura de uma tabela com o comando DESCRIBE. O resultado do comando é a exibição de nomes de colunas e tipos de dados, bem como a indicação informando se uma coluna deve conter dados.

Na sintaxe:

tablename O nome de qualquer tabela, view ou sinônimo existente e acessível ao usuário

Para descrever a tabela JOB_GRADES, use este comando:

```
SQL> DESCRIBE job_grades
```

Name	Null?	Type
-----	-----	-----
GRADE_LEVEL		VARCHAR2 (3)
LOWEST_SAL		NUMBER
HIGHEST_SAL		NUMBER

Exibindo a Estrutura de Tabelas

```
SQL> DESCRIBE departments
```

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER (4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2 (30)
MANAGER_ID		NUMBER (6)
LOCATION_ID		NUMBER (4)

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Exibindo a Estrutura de Tabelas (continuação)

O exemplo do slide exibe as informações sobre a estrutura da tabela DEPARTMENTS.

No resultado:

Null? Especifica se uma coluna deve conter dados (NOT NULL indica que uma coluna deve conter dados.)

Type Exibe o tipo de dados de uma coluna

Esta tabela descreve os tipos de dados:

Tipo de Dados	Descrição
NUMBER (<i>p</i> , <i>s</i>)	Valor numérico com um número máximo de <i>p</i> dígitos e <i>s</i> dígitos à direita da vírgula decimal
VARCHAR2 (<i>s</i>)	Valor de caractere de tamanho variável cujo tamanho máximo é igual a <i>s</i>
DATE	Valor de data e horário entre 1º de janeiro de 4712 A.C. e 31 de dezembro de 9999 D.C.
CHAR (<i>s</i>)	Valor de caractere de tamanho fixo cujo tamanho é igual a <i>s</i>

Comandos de Edição do SQL*Plus

- **A[PPEND] *text***
- **C[HANGE] / *old* / *new***
- **C[HANGE] / *text* /**
- **CL[EAR] BUFF[ER]**
- **DEL**
- **DEL *n***
- **DEL *m n***

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Oracle University and Impacta Tecnologia use only.

Comandos de Edição do SQL*Plus

Os comandos SQL*Plus são informados uma linha por vez e não são armazenados no buffer SQL.

Comando	Descrição
A[PPEND] <i>text</i>	Adiciona texto ao final da linha atual
C[HANGE] / <i>old</i> / <i>new</i>	Altera o texto <i>old</i> para o texto <i>new</i> na linha atual
C[HANGE] / <i>text</i> /	Deleta <i>text</i> da linha atual
CL[EAR] BUFF[ER]	Deleta todas as linhas do buffer SQL
DEL	Deleta a linha atual
DEL <i>n</i>	Deleta a linha <i>n</i>
DEL <i>m n</i>	Deleta as linhas <i>m</i> a <i>n</i> inclusive

Diretrizes

- Se você pressionar [Enter] antes de executar um comando, o SQL*Plus solicitará um número de linha.
- Para encerrar o buffer SQL, informe um dos caracteres finalizadores (ponto-e-vírgula ou barra) ou pressione [Enter] duas vezes. O prompt do SQL será exibido.

Comandos de Edição do SQL*Plus

- **I [NPUT]**
- **I [NPUT] *text***
- **L [IST]**
- **L [IST] *n***
- **L [IST] *m n***
- **R [UN]**
- ***n***
- ***n text***
- ***0 text***

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Comandos de Edição do SQL*Plus (continuação)

Comando	Descrição
I [NPUT]	Insere um número indefinido de linhas
I [NPUT] <i>text</i>	Insere uma linha que consiste em <i>text</i>
L [IST]	Lista todas as linhas no buffer SQL
L [IST] <i>n</i>	Lista uma linha (especificada por <i>n</i>)
L [IST] <i>m n</i>	Lista uma faixa de linhas (<i>m</i> a <i>n</i>) inclusive
R [UN]	Exibe e executa a instrução SQL atual no buffer
<i>n</i>	Especifica a linha que se tornará a atual
<i>n text</i>	Substitui a linha <i>n</i> por <i>text</i>
<i>0 text</i>	Insere uma linha antes da linha 1

Observação: É possível informar somente um comando SQL*Plus para cada prompt do SQL. Os comandos SQL*Plus não são armazenados no buffer. Para continuar um comando SQL*Plus na próxima linha, finalize a primeira linha com um hífen (-).

Usando LIST, n e APPEND

```
SQL> LIST
```

```
1  SELECT last_name  
2* FROM    employees
```

```
SQL> 1
```

```
1* SELECT last_name
```

```
SQL> A , job_id
```

```
1* SELECT last_name, job_id
```

```
SQL> L
```

```
1  SELECT last_name, job_id  
2* FROM    employees
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Usando LIST, n e APPEND

- Use o comando L[IST] para exibir o conteúdo do buffer SQL. O asterisco (*) ao lado da linha 2 no buffer indica que essa é a linha atual. Todas as edições feitas se aplicam à linha atual.
- Para alterar o número da linha atual, informe o número (n) da linha a ser editada. A nova linha atual será exibida.
- Use o comando A[PPEND] para adicionar texto à linha atual. A linha recém-editada será exibida. Verifique o novo conteúdo do buffer com o comando LIST.

Observação: É possível abreviar vários comandos SQL*Plus, inclusive LIST e APPEND, apenas à primeira letra desses comandos. LIST pode ser abreviado como L; APPEND pode ser abreviado como A.

Usando o Comando CHANGE

```
SQL> L
```

```
1* SELECT * from employees
```

```
SQL> c/employees/departments
```

```
1* SELECT * from departments
```

```
SQL> L
```

```
1* SELECT * from departments
```

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Usando o Comando CHANGE

- Use L [IST] para exibir o conteúdo do buffer.
- Use o comando C [HANGE] para alterar o conteúdo da linha atual no buffer SQL. Neste caso, substitua a tabela employees pela tabela departments. A nova linha atual será exibida.
- Use o comando L [IST] para exibir o novo conteúdo do buffer.

Comandos de Arquivo do SQL*Plus

- **SAVE *filename***
- **GET *filename***
- **START *filename***
- **@ *filename***
- **EDIT *filename***
- **SPOOL *filename***
- **EXIT**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Comandos de Arquivo do SQL*Plus

As instruções SQL comunicam-se com o servidor Oracle. Os comandos SQL*Plus controlam o ambiente, formatam os resultados de consultas e gerenciam arquivos. Você pode utilizar os comandos descritos nesta tabela:

Comando	Descrição
SAV[E] <i>filename</i> [.ext] [REP [LACE] APP [END]]	Salva o conteúdo atual do buffer SQL em um arquivo. Use APPEND para adicionar o conteúdo a um arquivo existente e REPLACE para sobregravar um arquivo existente. A extensão default é .sql.
GET <i>filename</i> [.ext]	Grava o conteúdo de um arquivo salvo anteriormente no buffer SQL. A extensão default do nome do arquivo é .sql.
STA[RT] <i>filename</i> [.ext]	Executa um arquivo de comandos salvo anteriormente
@ <i>filename</i>	Executa um arquivo de comandos salvo anteriormente (igual a START)
ED[IT]	Acessa o editor e salva o conteúdo do buffer no arquivo afiedt.buf
ED[IT] [<i>filename</i> [.ext]]	Acessa o editor para editar o conteúdo de um arquivo salvo
SPO[OL] [<i>filename</i> [.ext]] OFF OUT	Armazena os resultados da consulta em um arquivo. OFF fecha o arquivo de spool. OUT fecha o arquivo de spool e envia os resultados do arquivo para a impressora.
EXIT	Encerra o SQL*Plus

Usando os Comandos SAVE e START

```
SQL> L
  1  SELECT last_name, manager_id, department_id
  2* FROM    employees
SQL> SAVE my_query
```

```
Created file my_query
```

```
SQL> START my_query
```

LAST_NAME	MANAGER_ID	DEPARTMENT_ID
King		90
Kochhar	100	90
...		
20 rows selected.		

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

SAVE

Use o comando SAVE para armazenar o conteúdo atual do buffer em um arquivo. Dessa maneira, você poderá armazenar os scripts usados com frequência para uso futuro.

START

Use o comando START para executar um script no SQL*Plus.

EDIT

Use o comando EDIT para editar um script existente. Esse procedimento abre um editor com o arquivo de script. Após fazer as alterações, encerre o editor e retorne à linha de comandos do SQL*Plus.

Sumário

Neste apêndice, você aprendeu a usar o SQL*Plus como um ambiente para:

- **Executar instruções SQL**
- **Editar instruções SQL**
- **Formatar a saída**
- **Interagir com arquivos de script**

ORACLE

Copyright © 2004, Oracle. Todos os direitos reservados.

Sumário

O SQL*Plus é um ambiente de execução que pode ser usado para enviar comandos SQL ao servidor de banco de dados, bem como para editar e salvar esses comandos. É possível executar comandos no prompt do SQL ou em um arquivo de script.

Índice

Observação: Um número ou uma letra em negrito refere-se a uma lição ou a um apêndice inteiro.

A

Adicionando Dados por Meio de uma View 10-15

ANSI (American National Standards Institute) I-11, 1-12, I-25, I-28, I-30,
1-22, 03-26, 03-54, 05-04, 08-27, D-04

Apelido 01-04, 01-08, 01-14-17, 01-36, 02-04, 02-07, 02-20, 02-21, 02-31
03-46, 04-12, 04-25, 05-08, 05-11, 05-12, 05-20, 05-29, 07-17, 07-21,
09-33, 10-07, 10-09, 10-11, C-11, C-12, C-15, C-20, C-21

Apelido de Coluna 01-04, 01-08, 01-14-17, 01-36, 02-04, 02-07, 02-20, 02-21,
02-31, 03-46, 04-12, 04-25, 05-08, 05-11, 07-17, 07-21,
09-33, 10-07, 10-09, 10-11, C-11,

Apelido de Tabela 05-12, 05-20, 05-29, C-12, C-15, C-20, C-21

Aplicações de Data Warehouse I-09

Argumentos de Function 03-03, 03-05, 03-15, 03-50

Argumentos (Functions) 03-03, 03-05, 03-15, 03-50

Atributos I-19, I-20, I-23, 11-14

Área Global do Sistema I-27

B

Banco de Dados 01-02, 01-03, 01-21, 01-22, 01-23, 01-34, 01-36, 02-02, 02-06,
03-10, 03-17, 03-19, 03-20, 03-26, 04-13, 04-16, 05-29, 06-07, 08-03, 08-11,
08-13, 08-18, 08-24, 08-25, 08-31, 08-32, 08-37, 08-38, 08-39, 09-02, 09-03,
09-04, 09-05, 09-06, 09-18, 09-33, 09-34, 09-35, 09-36, 09-37, 10-03, 10-05,
10-15, 10-16, 10-19, 10-21, 10-22, 10-24, 10-29, 10-31, 10-32, 10-34, 10-37,
10-38, 10-39, 10-40, 10-41, 11-03, 11-04, 11-05, 11-07, 11-09, 11-10, 11-12,
11-17, 11-18, 11-19, C-06, C-07, C-09, C-21, D-03, D-04, D-05, D-06, D-15,
I-02, I-03, I-04, I-05, I-06, I-06, I-08, I-09, I-10, I-11, I-13, I-14, I-15, I-16, I-23

Banco de Dados Oracle 10g I-03, I-04, I-05, I-06, I-27, I-30

Banco de Dados Relacional I-02, I-03, I-09, I-14, I-15, I-23, I-25, I-27, I-28, I-30

Botão Execute (iSQL*Plus) 01-07, 01-25, 01-33

Botão Load 01-31, 01-32

Índice

C

Cálculos 01-09, 01-14, 01-23, 01-37, 03-03, 03-14, 03-20, 03-25, 03-60, 03-61, D-05

Cardinalidade I-20

Chave Estrangeira 05-09, 08-07, 09-17, 09-25, 09-26, 09-27, 09-31, 10-33, 11-12, 11-13, C-7, C-8, I-21

Chave exclusiva composta 09-22, 09-23

Ciclo de Vida de Desenvolvimento do Sistema I-11, I-12, I-17

Cláusula FROM 01-04, 01-09, 02-04, 03-14, 04-13, 04-16, 05-12, 06-04, C-09, C-12, C-20

Cláusula GROUP BY 04-02, 04-11, 04-12, 04-13, 04-14, 04-16, 04-17, 04-20, 04-21, 04-24, 06-12, 10-13, 10-14, 10-15

Cláusula HAVING 04-02, 04-18, 04-19, 04-20, 04-21, 04-22, 04-24, 04-25, 06-04, 06-11, 06-19

Cláusula ON 05-05, 05-13, 05-14, 05-15, 05-16, 05-17, 05-18, 05-22, 07-12, 10-16

Cláusula ORDER BY 02-20, 02-21, 02-23, 02-31, 02-32, 03-05, 04-14, 04-24, 06-06, 07-17, 07-21, 07-23, C-11

Cláusula UNION 07-12

Cláusula USING 05-04, 05-08, 05-09, 05-10, 05-11, C-06

Cláusula VALUES 08-05, 08-07, 08-11, 08-22

Cláusula WHERE 02-03, 02-04, 02-05, 02-06, 02-07, 02-08, 02-10, 02-14, 02-22, 02-23, 02-26, 02-27, 02-31, 02-32,

Classificado 07-08, 07-11, 07-18, 07-21, 07-23

Classificação 2

Ordem de Classificação Default 02-21, 04-16

Palavra-chave DESC 02-21

Coluna COMMENTS 11-06, 11-19

Coluna GROUP BY 04-14, 04-16

Colocação da Subconsulta 06-04, 06-06

Comando AUTOCOMMIT 08-29

Comando DESCRIBE 01-26, 08-07, 09-08, 09-33, 10-08, 10-12, 11-11, D-07

Comando VERIFY 02-30

Banco de Dados Oracle 10g: Fundamentos de SQL I Índice-2

Índice

C

Compatível com SQL: 1999 05-04, 05-18, 05-30, C-06

Condição de Faixa BETWEEN 02-09

Condição IN 02-07, 02-10, 05-02, 05-04, 05-13, 05-18, 05-19, 05-21, 05-22,
05-26, 05-27, 10-35, C-02, C-04, C-05, C-06, C-07, C-13, C-14, C-16, C-17

Condição IS NOT NULL 02-13

Condição IS NULL 02-13

Condição LIKE 02-11, 02-12

Condição Lógica 02-14

Condições NULL 02-13, 02-31

Conjuntos de Linhas 04-03

CONSTRAINT_TYPE 09-19, 11-13

CONSTRAINTS 9, 01-12, 08-07, 08-17, 08-21, 10-16, 10-36, 11-11, 11-12, 11-13,
11-14, 11-20

- Constraint CHECK 08-07, 09-28, 10-20, 11-12, 11-13
- Definindo 09-19, 09-20
- Constraint NOT NULL 01-26, 09-19, 09-21, 09-22, 11-10
- Constraint PRIMARY KEY 09-20, 09-24
- Constraint Somente Leitura 10-18
- REFERENCES 01-25, 01-32, 01-34, 01-35, 09-25, 09-26,
09-27, 09-28, 09-29, 11-13
- Constraint de Integridade Referencial 08-21, 09-25
- Constraint UNIQUE 09-22, 09-23
- Constraint de Integridade de Chave UNIQUE 09-22

Constraint CHECK 08-07, 09-28, 10-20, 11-12, 11-13

Constraint de Integridade de Chave UNIQUE 09-22

Constraint de Integridade 08-07, 08-21, 09-17, 09-19, 09-22, 09-25, 09-30, 09-31,
09-32, 10-16, 11-03

Constraint de Integridade Referencial 08-21, 09-25

Índice

C

Constraint NOT NULL 01-26, 09-19, 09-21, 09-22, 11-10
Constraint PRIMARY KEY 09-20, 09-24
Constraint Somente Leitura 10-18
Constraint UNIQUE 09-22, 09-23
Consistência de Leitura 08-31, 08-37, 08-38
Consulta Externa 06-03, 06-04, 06-05, 06-09, 06-10, 06-12, 06-13, 06-20
Consulta Interna ou Instrução Select Interna 06-03, 06-04, 06-05, 06-07, 06-09,
06-13, 06-14, 06-17, 06-20
Conversão Explícita de Tipos de Dados 03-26, 03-29, 03-30, 03-31
Conversão Implícita de Tipos de Dados 03-26, 03-27, 03-28
Criar Arquivos de Script 01-23, D-05
CURRVAL 09-07, 09-28, 10-25, 10-27

D

Dados de Mais de uma Tabela 05-02, 05-03, 05-05, 05-30, C-02, C-03, C-07
Data 01-08, 01-09, 01-17, 01-28, 02-06, 02-07, 02-10, 02-11, 02-20, 02-21,
02-23, 02-26, 03-02, 03-03, 03-04, 03-06, 03-14, 03-15, 03-17, 03-19, 03-20,
03-21, 03-22, 03-23, 03-24, 03-25, 03-27, 03-29, 03-29, 03-30, 03-31, 03-32,
03-33, 03-35, 03-37, 03-41, 03-42, 03-43, 03-44, 03-46, 03-48, 03-60, 03-61,
04-05, 04-07, 04-24, 07-04, 07-19, 08-02, 08-03, 08-06, 08-08, 08-09,
08-09, 08-12, 08-13, 08-14, 08-15, 08-16, 08-22, 08-23, 08-27, 08-28, 08-32,
08-33, 08-37, 08-38, 08-39, 08-40, 09-07, 09-08, 09-11, 09-12, 09-14, 09-17,
09-23, 09-26, 09-27, 09-28, 09-29, 09-30, 09-33, 10-06, 10-07, 10-16, 10-35,
11-04, 11-07, 11-08, 11-15
DBMS I-02, I-13, I-14, I-21, I-23, I-26, I-27

Índice

E

Equijoins 05-09, 05-19, 05-20, 05-29, C-08, C-09, C-14, C-15, C-21

Esquema 09-02, 09-05, 09-06, 09-19, 09-36, 10-01, 10-32, 10-38, 11-02, 11-03, 11-04, 11-07, 11-21

Estruturas de Banco de Dados 09-05, 09-08

Estruturas de Dados 09-03

Executar SQL 01-02, 01-23, 01-36, D-02, D-05, D-15

Exibição de Data Default 02-06, 03-17

Expressão CASE 03-54, 03-55, 03-56, 03-57, 03-60

Expressões Aritméticas 01-09, 01-13, 01-16, 02-04

F

Falha do Sistema 08-24, 08-29

Formato de Data RR 03-43, 03-44

Functions **3, 4**

02-04, 04-01, 04-03, 04-04, 04-05, 04-06, 04-07, 04-10, 04-11, 04-12, 04-13, 04-17, 04-18, 04-20, 04-23, 04-24, 04-25, 06-10, 06-19, 07-19, 08-08, 09-07, 09-28, 10-06, 10-12, 10-13, 10-14, 10-15, 11-03, 11-08, I-03, I-07, I-10

Function ADD_MONTHS 03-22, 03-23, 03-46, 03-60

Function AVG 04-04, 04-06, 04-07, 04-10, 04-13, 04-14, 04-18, 04-21, 04-23, 04-24, 06-11, 10-12

Functions de Manipulação de Maiúsculas e Minúsculas 03-07

Functions de Manipulação de Caracteres 03-07, 03-11, 03-12

Function COALESCE 03-06, 03-47, 03-52, 03-53, 03-60, 04-05

Function CONCAT 01-16, 01-18, 03-07, 03-08, 03-11, 03-12, 03-38, 03-46, 03-60, 03-61, C-07

Functions de Conversão 03-02, 03-04, 03-06, 03-09, 03-26, 03-27, 03-28, 03-29, 03-30, 03-31, 03-48, 03-50, 03-60, 07-19, 08-06

Índice

F

Function COUNT 04-08

Function DECODE 03-06, 03-54, 03-56, 03-57, 03-58, 03-59, 03-60, 11-04

Functions de Grupo em uma Subconsulta 06-10

Function INITCAP 03-07, 03-08, 03-09, 03-10, 03-60

Function INSTR 03-07, 03-08, 03-11, 03-12, 03-60

Function LAST_DAY 03-22, 03-23, 03-60

Function LENGTH 03-07, 03-08, 03-11, 03-12, 03-51, 03-60

Function LOWER 03-02, 03-07, 03-08, 03-09, 03-10, 03-10, 03-37, 03-60

Function LPAD 03-07, 03-08, 03-11

Function MAX 04-03, 04-04, 04-06, 04-07, 04-19, 04-2, 04-23, 04-24

Function MIN 04-04, 04-06, 04-07, 04-12, 04-16, 04-24

Function TO_CHAR 03-32, 03-37, 03-38, 03-40

Functions TO_NUMBER ou TO_DATE 03-41

Function SUBSTR 03-07, 03-08, 03-11, 03-12, 03-46, 03-60

Function SUM 04-02, 04-04, 04-06, 04-07, 04-12, 04-16, 04-22, 04-24

Function STDDEV 04-04, 04-07, 04-24

Function SYSDATE 03-19, 03-21, 03-23, 03-24, 03-60, 08-08, 08-08, 09-07, 09-28

Functions de uma Única Linha 03-04, 03-05, 03-06, 03-45, 03-60, 04-03

Functions de Número 03-06, 03-13

Function NVL 03-06, 03-47, 03-48, 03-49, 03-50, 03-52, 03-60, 04-05, 04-10

Function NVL2 03-06, 03-47, 03-50, 03-60, 04-05

Function TRIM 03-07, 03-08, 03-11

Function TRUNC 03-13, 03-15, 03-22, 03-24, 03-59, 03-60, 08-21

Function UPPER 03-02, 03-07, 03-08, 03-09, 03-10, 03-46, 03-60

Índice

F

Function MOD 03-16

Function MONTHS_BETWEEN 03-06, 03-22, 03-23, 03-60

Function NEXT_DAY 03-22, 03-23, 03-46, 03-60

Function NULLIF 03-06, 03-47, 03-51, 03-60

Function NVL 03-06, 03-47, 03-48, 03-49, 03-50, 03-52, 03-60, 04-05, 04-10

Function NVL2 03-06, 03-47, 03-50, 03-60, 04-05

Function ROUND 03-13, 03-14, 03-15, 03-22, 03-24, 03-40, 03-60, 11-16

Functions ROUND e TRUNC 03-24

Function STDDEV 04-04, 04-07, 04-24

Function SUBSTR 03-07, 03-08, 03-11, 03-12, 03-46, 03-60

Function SUM 04-02, 04-04, 04-06, 04-07, 04-12, 04-16, 04-22, 04-24

Function SYSDATE 03-19, 03-21, 03-23, 03-24, 03-60, 08-08, 08-08, 09-07, 09-28

Function TO_CHAR 03-32, 03-37, 03-38, 03-40

Function TO_NUMBER ou TO_DATE 03-41

Function TRIM 03-07, 03-08, 03-11

Function TRUNC 03-13, 03-15, 03-22, 03-24, 03-59, 03-60, 08-21

Function UPPER 03-02, 03-07, 03-08, 03-09, 03-10, 03-46, 03-60

Function DECODE 03-06, 03-54, 03-56, 03-57, 03-58, 03-59, 03-60, 11-04

Functions de Manipulação de Maiúsculas e Minúsculas 3-07

Functions de Manipulação de Caracteres 03-07, 03-11, 03-12

Function ADD_MONTHS 03-22, 03-23, 03-46, 03-60

Function AVG 04-04, 04-06, 04-07, 04-10, 04-13, 04-14, 04-18, 04-21, 04-23, 04-24, 06-11, 10-12

Function INITCAP 03-07, 03-08, 03-09, 03-10, 03-60

Function INSTR 03-07, 03-08, 03-11, 03-12, 03-60

Índice

F

Function MAX 04-03, 04-04, 04-06, 04-07, 04-19, 04-21, 04-23, 04-24

Function MIN 04-04, 04-06, 04-07, 04-12, 04-16, 04-24

Function LOWER 03-02, 03-07, 03-08, 03-09, 03-10, 03-10, 03-37, 03-60

Function LPAD 03-07, 03-08, 03-11

Function LAST_DAY 03-22, 03-23, 03-60

Function LENGTH 03-07, 03-08, 03-11, 03-12, 03-51, 03-60

Function MOD 03-16

Function MONTHS_BETWEEN 03-06, 03-22, 03-23, 03-60

Functions de Várias Linhas 03-04

Functions Aninhadas 03-45, 03-61

Function NEXT_DAY 03-22, 03-23, 03-46, 03-60

Function NULLIF 03-06, 03-47, 03-51, 03-60

Functions de Conversão 03-02, 03-04, 03-06, 03-09, 03-26, 03-27, 03-28, 03-29, 03-30, 03-31, 03-48, 03-50, 03-60, 07-19, 08-06

Function COUNT 04-08

Function COALESCE 03-06, 03-47, 03-52, 03-53, 03-60, 04-05

Function ROUND 03-13, 03-14, 03-15, 03-22, 03-24, 03-40, 03-60, 11-16

Functions ROUND e TRUNC 03-24

Function CONCAT 01-16, 01-18, 03-07, 03-08, 03-11, 03-12, 03-38, 03-46, 03-60, 03-61, C-07

Functions de Grupo em uma Subconsulta 06-10

Functions de Grupo 03-04, 04-01, 04-02, 04-03, 04-04, 04-05, 04-06, 04-07, 04-10, 04-11, 04-12, 04-13, 04-17, 04-18, 04-20, 04-23, 04-24, 04-25, 06-10, 06-19, 10-12, 10-13, 10-14, 10-15

Functions de Grupo Aninhadas 04-23

Valores Nulos 04-05, 04-10

G

Gerar Números Exclusivos 10-03, 10-22

Índice

I

Índice	10 , 11-03, 11-07, 11-08
Tipos de Índices	10-33
Quando Criar um Índice	10-35
Índice Não Exclusivo	10-33, 10-41
Índice Exclusivo	09-23, 09-24, 10-33, 10-35, 10-41
Identificador Exclusivo	I-19, I-20
Incorporando uma Subconsulta à Instrução CREATE VIEW	10-07
Instrução DELETE	08-18, 08-19, 08-20, 08-21, 08-37, 08-39
Instrução COMMENT	11-19
Instrução DCL	08-24, 08-25, 08-29
Instrução DDL	08-21, 08-24, 08-25, 08-29, 08-36, 09-01, 09-02, 09-05, 09-08, 09-35, 11-07, I-28
Instrução ALTER TABLE	09-34
Instrução COMMIT	08-02, 08-24, 08-25, 08-26, 08-27, 08-29, 08-31, 08-32, 08-33, 08-35, 08-36, 08-37, 08-38, 08-39, 09-08, 09-35, 10-28, 10-35
iSQL*Plus	01-02, 01-07, 01-08, 01-14, 01-20, 01-21, 01-22, 01-23, 01-24, 01-25, 01-26, 01-28, 01-30, 01-32, 01-33, 01-34, 01-35, 01-36, 01-37, 02-02, 02-22, 02-23, 02-24, 02-25, 02-26, 02-28, 02-29, 02-30, 02-31, 03-32, 07-21, 08-07, 08-25, 08-29, 09-33, 10-08, 10-12, 11-11, 11-15, I-02
Botão Execute (iSQL*Plus)	01-07, 01-25, 01-33
Recursos do iSQL*Plus	01-21
Botão Load	01-31, 01-32
Load Script	01-25, 01-30
SET VERIFY ON	02-30
Substituição com E Comercial Único	02-23

Índice

I

- Instrução DROP INDEX 10-36
- Instrução DROP SYNONYM 10-39
- Instrução DROP TABLE 09-35, 09-36
- Instrução DROP VIEW 10-19
- Instruções DCL (Data Control Language) 09-05
- Instruções DDL (Data Definition Language) 09-02, 09-05
- Instruções DML (Data Manipulation Language) **8**
 - 09-02, 10-06, 10-07, 10-13, 10-14, 10-15, 10-16, 10-17, 10-18, 10-35, I-28
- Instrução ALTER SEQUENCE 10-29, 10-30
- Instrução CREATE INDEX 10-34
- Instrução CREATE PUBLIC SYNONYM 10-39
- Instrução CREATE SEQUENCE 10-23
- Instrução CREATE SYNONYM 10-37, 10-38, 10-39
- Instrução CREATE TABLE 09-05, 09-32, 09-36, 09-37
- Instrução INSERT 08-05, 08-11, 08-22, 08-23, 09-08, 09-19, 10-27
- Instrução SAVEPOINT 08-02, 08-27, 08-28, 08-32, 08-36, 08-39, I-28
- Instrução SELECT 01-04, 01-06, 01-16, 01-20, 03-14, 04-12, 04-13, 04-14,
04-16, 07-08, 07-15, 07-18, C-09, C-11
- Instrução SELECT - Recursos Avançados 06-02
- Instrução ROLLBACK 08-02, 08-21, 08-25, 08-26, 08-27, 08-28, 08-29,
08-31, 08-34, 08-35, 08-36, 08-39, 10-28, I-28
- Instrução SELECT Aninhada 06-04, 06-20
- Instrução UPDATE 08-13, 08-14, 08-15 08-16
- INTERVAL YEAR TO MONTH 09-14
- Instância Oracle I-27
- ISO (International Standards Organization) I-28

Banco de Dados Oracle 10g: Fundamentos de SQL I Índice-10

Índice

J

Java I-04, I-07, I-10, I-27
Joins Cruzadas 05-04, 05-05, 05-28, 05-29, C-06
Join Tridimensional 05-18
Join FULL OUTER 05-05, 05-22, 05-25
Join RIGHT OUTER 05-22, 05-24
Join LEFT OUTER 05-22, 05-23

L

Linhas Duplicadas 01-20, 04-08, 07-08, 07-11, 07-12, 07-18, 07-23
Literal 01-17, 01-18, 01-19, 02-04, 02-11, 02-12, 03-11, 03-35, 03-51, 03-55, 07-20, 09-07
Load Script 01-25, 01-30
Lógica IF-THEN-ELSE 03-54, 03-55, 03-57, 03-60

M

Modelo de Formato 03-22, 03-24, 03-32, 03-33, 03-35, 03-37, 03-40, 03-41, 03-42
Modificador 03-41, 03-42
Modo fx 03-41, 03-42

N

Não-equijoin 05-19, 05-20, 05-29, C-06, C-14, C-15, C-21
Nomeação 09-04, 09-18, 11-05
NEXTVAL 09-07, 09-28, 10-25, 10-27

Índice

O

OLTP I-09
ON DELETE CASCADE 09-27
ON DELETE SET NULL 09-27
Opção CYCLE (com Sequências) 10-24, 10-29
Opção DEFAULT 09-07, 10-23, 10-24
Opção ESCAPE 02-12
Operador INTERSECT 07-03, 07-13, 07-14, 07-23
Opção OR REPLACE 10-08, 10-11
Opção READ ONLY 10-17
Operações DML em Dados por Meio de uma View 10-13
Operadores Aritméticos 01-09, 01-10, 03-20, 03-21, 03-60
Operador de Uma Única Linha 06-04, 06-06, 06-08, 06-12, 06-14, 06-19
Operadores de conjunto 7
Operador ALL 06-16, 07-11, 07-12, 07-18, 07-23
Operador ANY 06-15
Operador NOT 02-17, 02-31, 06-16
Operador MINUS 07-15, 07-16, 07-23, 07-24
Operador SOME 06-15
Operador UNION ALL 07-11, 07-12, 07-18, 07-23
Operador UNION 07-08, 07-09, 07-19, 07-20, 07-21, 07-23, 07-24
Oracle Enterprise Manager 10g Grid Control I-05, I-08, I-30
ORDBMS I-02
Ordem 02-18, 02-20, 02-21, 02-23, 02-27, 02-28, 02-31, 02-32
Ordem de Classificação Default 02-21, 04-16
Ordem de Linhas 02-20, 07-02, 07-21, I-23
Ordem de Precedência 01-11, 02-18

Banco de Dados Oracle 10g: Fundamentos de SQL I Índice-12

Índice

P

Padrões (American National Standards Institute) I-11, 1-12, I-25, I-28, I-30, 1-22, 03-26, 03-54, 05-04, 08-27, D-04

Palavra-chave DESC 02-21

Palavra-chave DISTINCT 01-20, 04-09, 07-11, 10-13, 10-14, 10-15

Palavras-chave 01-04, 01-05, 01-07, 01-14, 01-15, 01-16, 01-20, 01-22, 02-21, 04-09, 05-06, 07-11, 08-07, 09-26, 09-27, 10-13, 10-14, 10-15, I-04

Palavras-chave NATURAL JOIN 05-06, C-06

Pesquisa com Curinga 02-11

Prefixos de Tabela 05-11, 05-12, C-12

Privilégio DROP ANY INDEX 10-36

Privilégio DROP ANY VIEW 10-19

Processamento Condicional 03-54

Processamento de Transações On-line I-09

Produto Cartesiano 05-02, 05-26, 05-27, 05-28, 05-29, C-04, C-05, C-21

Programação Orientada a Objeto I-09

Projeção 01-03

Pseudocolunas NEXTVAL e CURRVAL 10-25

R

Recuperar Dados de uma View 10-10

Recursos Avançados da Instrução SELECT 06-02

RDBMS I-02, I-14, I-21, I-23, I-26, I-27

REFERENCES 01-25, 01-32, 01-34, 01-35, 09-25, 09-26, 09-27, 09-28, 09-29, 11-13

Relacionamento entre Entidades I-17, I-19, I-20

Relacional de Objeto I-02, I-06, I-09, I-13, I-30

Regras de Precedência 01-10, 01-11, 02-18, 02-19

Restringir as Linhas 02-02, 02-04, 04-19, C-10

Resultados Sumariados de Grupos 04-16

Retornar um Valor 03-03, 03-06, 03-14, 03-22

Rollback no Nível de Instrução 08-36

Banco de Dados Oracle 10g: Fundamentos de SQL I Índice-13

Índice

S

- Servidor de Aplicações Oracle 10g I-05, I-07, I-30
- Símbolos de Cerquilha 03-40
- Sistema de Gerenciamento de Banco de Dados Relacional de Objeto I-02, I-09, I-30
- Sistema de Gerenciamento de Banco de Dados Relacional I-02, I-09, I-14, I-27, I-30
- Strings de caracteres 01-16, 01-17, 01-18, 02-06, 02-15, 03-09, 03-11, 03-35, 03-38
- Structured Query Language 01-02, I-25, I-26
- Subconsultas de Várias Linhas 06-02, 06-06, 06-07, 06-14, 06-15, 06-16
- Subconsultas de Várias Colunas 06-07
- Subconsulta AS 09-32, 10-07
- Subconjuntos Lógicos 10-04
- Sub-SELECT 06-04
- Subconsultas em instruções UPDATE 08-16
- Subconsultas para Deletar Linhas 08-20
- Substituição com E Comercial 2-27
 - E Comercial Duplo (&&) 02-28
- Seleção 01-03, 02-03
- Seqüências 10
 - Seqüências de Armazenamento em Cache 10-28
 - CURRVAL 09-07, 09-28, 10-25, 10-27
 - Opção CYCLE 10-24, 10-29
 - Gerar Números Exclusivos 10-03, 10-22
 - NEXTVAL 09-07, 09-28, 10-25, 10-27
 - Pseudocolunas NEXTVAL e CURRVAL 10-25

Índice

S

- Seqüências de Armazenamento em Cache 10-28
- SET VERIFY ON 02-30
- SGA I-27
- Substituição com E Comercial Único (&) 02-23
- Subconsultas de Uma Única Linha 06-02, 06-06, 06-07, 06-08, 06-09
- Subconsulta 6
 - Palavra-chave AS 09-32, 10-07
 - Incorporando uma Subconsulta à Instrução CREATE VIEW 10-07
 - Functions de Grupo em uma Subconsulta 06-10
 - Consulta Interna ou Instrução Select Interna 06-03, 06-04, 06-05, 06-07, 06-09, 06-13, 06-14, 06-17, 06-20
 - Instrução SELECT Aninhada 06-04, 06-20
 - Consulta Externa 06-03, 06-04, 06-05, 06-09, 06-10, 06-12, 06-13, 06-20
 - Colocação da Subconsulta 06-04, 06-06
 - Subconsultas de Uma Única Linha 06-02, 06-06, 06-07, 06-08, 06-09
 - Sub-SELECT 06-04
 - Subconsultas em instruções UPDATE 08-16
 - Subconsultas para Deletar Linhas 08-20
- Sinônimo 06-15, 09-03, 09-06, 09-35, 10-02, 10-03, 10-37, 10-38, 10-39, 10-40, 10-41, 11-03, 11-08, 11-18, 11-20, 11-21

Índice

T

Tabela DUAL 03-14, 03-19

Tipo de Dados DATE 03-06, 03-22, 03-25, 03-61, 04-07, 09-12

Exibição de Data Default 02-06, 03-17

Tipo de Dados de Data/Horário 09-11, 09-12, 09-14

Tipos de Dados 01-09, 01-26, 01-27, 03-03, 03-25, 03-26, 03-48, 03-50, 03-60,
03-61, 04-05, 04-07, 05-06, 05-08, 08-07, 08-11, 09-02, 09-09, 09-11, 09-12,
09-14, 09-36, 11-10, 11-11, D-07, D-08

Tipos de Índices 10-33

Tipo de Dados NUMBER 03-38, 08-06

Transações 08-02, 08-24, 08-25, 08-27, 08-36, 08-40, 09-35

Tupla I-23

U

Unindo Tabelas **6, C**

Joins Cruzadas 05-04, 05-05, 05-28, 05-29, C-06

Equijoins 05-09, 05-19, 05-20, 05-29, C-08, C-09, C-14,
C-15, C-21

Join FULL OUTER 05-05, 05-22, 05-25

Join LEFT OUTER 05-22, 05-23

Join Tridimensional 05-18

Índice

V

Valor Nulo 01-12, 01-13, 01-16, 02-13, 02-21, 03-47, 03-48, 03-49, 03-50, 03-51, 03-57, 04-05, 04-08, 04-09, 04-10, 06-13, 06-17, 07-08, 07-13, 08-07, 09-07, 09-21, 09-24, 10-35

Variáveis de Substituição 02-22, 02-23, 02-26, 02-27, 02-30, 02-31, 02-32, 08-10

VARIANCE 04-04, 04-07, 04-24

View CAT 11-08

View de Dicionário DBA_OBJECTS 11-05

View de Dicionário USER_COL_COMMENTS 11-19

View de Dicionário USER_CONS_COLUMNS 11-12, 11-14, 11-20

View de Dicionário USER_CONSTRAINTS 11-12, 11-13, 11-14, 11-20

View de Dicionário USER_OBJECTS 11-03, 11-05, 11-06, 11-07, 11-08, 11-17, 11-20

View de Dicionário USER_SEQUENCES 11-16, 11-17, 11-20

View de Dicionário USER_SYNONYMS 11-18

View de Dicionário USER_TAB_COLUMNS 11-03, 11-10, 11-11, 11-20

View de Dicionário USER_TAB_COMMENTS 11-19

View de Dicionário USER_TABLES 11-09, 11-10

View de Dicionário USER_VIEWS 11-15, 11-20

Índice

V

Views 10

Adicionando Dados por Meio de uma View 10-15

Operações DML em Dados por Meio de uma View 10-13

Diretrizes para Criar uma View 10-08

Views: Simples e Complexas 10-06

View de Dicionário ALL_COL_COMMENTS 11-19

View de Dicionário ALL_OBJECTS 11-05, 11-7

View de Dicionário ALL_TAB_COMMENTS 11-19

Views de Dicionário 11, 09-05, 09-18, 10-04, 10-36

ALL_COL_COMMENTS 11-19

ALL_OBJECTS 11-05, 11-7

ALL_TAB_COMMENTS 11-19

CAT 11-08

DBA_OBJECTS 11-05

View de Dicionário USER_COL_COMMENTS 11-19

View de Dicionário USER_CONS_COLUMNS 11-12, 11-14, 11-20

View de Dicionário USER_CONSTRAINTS 11-12, 11-13, 11-14, 11-20

View de Dicionário USER_OBJECTS 11-03, 11-05, 11-06, 11-07,
11-08, 11-17, 11-20

View de Dicionário USER_SEQUENCES 11-16, 11-17, 11-20

View de Dicionário USER_SYNONYMS 11-18

View de Dicionário USER_TAB_COLUMNS 11-03, 11-10, 11-11, 11-20

View de Dicionário USER_TAB_COMMENTS 11-19

View de Dicionário USER_TABLES 11-09, 11-10

View de Dicionário USER_VIEWS 11-15, 11-20

W

WITH CHECK OPTION 10-07, 10-08, 10-16, 11-13

X

XML I-04, I-06, I-27

Banco de Dados Oracle 10g: Fundamentos de SQL I Índice-18