

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

ORACLE

Academy

Database Foundations

6-3

Data Definition Language (DDL)

ORACLE
Academy



Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

Objetivos

- Esta lição abrange os seguintes objetivos:
 - Identificar as etapas necessárias para criar tabelas de banco de dados
 - Descrever a finalidade do DDL (Data Definition Language)
 - Listar as operações DDL necessárias para criar e manter as tabelas de um banco de dados



Objetos de Banco de Dados

Objeto	Descrição
Tabela	É a unidade de armazenamento básica, composta por linhas
View	Representa logicamente subconjuntos de dados de uma ou mais tabelas
Sequência	Gera valores numéricos
Índice	Melhora o desempenho de algumas consultas
Sinônimo	Fornece um nome alternativo para um objeto

Observação: neste curso, iremos criar e recuperar informações da unidade básica de armazenamento, as tabelas. Mais objetos de banco de dados estão disponíveis, mas eles não são abordados neste curso.

Regras de Nomenclatura de Tabelas e Colunas

- Os nomes de tabelas e colunas devem:
 - Começar com uma letra
 - Ter de 1 a 30 caracteres
 - Conter apenas A–Z, a–z, 0–9, _, \$ e #
 - Não duplicar o nome de outro objeto pertencente ao mesmo usuário
 - Não ser uma palavra reservada do servidor Oracle

Observação: os nomes não fazem distinção entre maiúsculas e minúsculas. Por exemplo, `EMPLOYEES` é tratado da mesma forma que `eMPLOYEES` ou `eMpLOYEES`. No entanto, os identificadores com aspas fazem distinção entre maiúsculas e minúsculas.

Para obter uma lista completa de palavras reservadas, consulte:

https://docs.oracle.com/cd/B28359_01/appdev.111/b31231/appb.htm#CJHIIICD

Instrução CREATE TABLE

- Para emitir uma instrução CREATE TABLE, você deve ter:
 - O privilégio CREATE TABLE
 - Uma área de armazenamento

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



ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

6

Para criar uma tabela, o usuário deve ter o privilégio `CREATE TABLE` e uma área de armazenamento na qual criar objetos. O administrador de banco de dados (DBA) usa instruções DCL (Data Control Language) para conceder privilégios aos usuários.

Na sintaxe:

- `schema` é o mesmo que o nome do proprietário.
- `table` é o nome da tabela.
- `DEFAULT expr` especificará um valor padrão se um valor for omitido na instrução `INSERT`.
- `column` é o nome da coluna.
- `datatype` é o tipo de dados e o tamanho da coluna.

Observação: o privilégio `CREATE ANY TABLE` é necessário para criar uma tabela em qualquer esquema que não seja o do usuário.

Instrução CREATE TABLE

- Especifique na instrução:
 - Nome da tabela
 - Nome da coluna, tipo de dados e tamanho da coluna
 - Restrições de integridade (opcional)
 - Valores padrão (opcional)

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



Criando Tabelas

- Crie a tabela:

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

- Para confirmar se a tabela foi criada, execute o comando DESCRIBE

Observação: Para exibir a lista de tabelas que você possui, consulte o dicionário de dados. Por exemplo,
`select table_name from user_tables;`

Para obter mais informações sobre tabelas do dicionário de dados, consulte:
<https://docs.oracle.com/database/121/GMSWN/apc.htm#GMSWN600>

Criando Tabelas

- Confirme a criação da tabela:

```
DESCRIBE dept;
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT	DEPTNO	NUMBER	-	2	0	-		-	-
	DNAME	VARCHAR2	14	-	-	-		-	-
	LOC	VARCHAR2	13	-	-	-		-	-
	CREATE_DATE	DATE	7	-	-	-		SYSDATE	-

Tipos de Dados

Tipo de Dados	Descrição
VARCHAR2(size)	Dados de caracteres de comprimento variável (É necessário especificar um tamanho máximo, o tamanho mínimo é 1.) Tamanho máximo: 32767 bytes
CHAR(size)	Dados de caracteres de comprimento fixo em bytes. (O tamanho padrão e mínimo é 1; o tamanho máximo é 2.000)
NUMBER(p, s)	Dados numéricos de comprimento variável. A precisão é p e a escala é s. (A precisão é o número total de dígitos decimais, e a escala é o número de dígitos à direita da casa decimal; a precisão pode variar de 1 a 38, e a escala pode variar de -84 a 127.)
DATE	Valores de data e hora até o segundo mais próximo entre 1º de janeiro de 4712 a.C e 31 de dezembro de 9999 d. C.
LONG	Dados de caracteres de comprimento variável (até 2 GB)

Tipos de Dados

Tipo de Dados	Descrição
CLOB	Um objeto de caracteres grande (CLOB) que contém caracteres de um ou de vários bytes. O tamanho máximo é $(4 \text{ GB} - 1) * (\text{DB_BLOCK_SIZE})$; armazena dados do conjunto de caracteres nacionais.
NCLOB	CLOB que contém caracteres Unicode. Tanto conjuntos de caracteres de largura variável como de largura fixa são suportados e ambos usam o conjunto de caracteres nacionais de banco de dados. O tamanho máximo é $(4 \text{ GB} - 1) * (\text{tamanho do bloco de banco de dados})$; armazena dados do conjunto de caracteres nacionais.
RAW (Tamanho)	Dados binários brutos de tamanho em bytes. É necessário especificar o tamanho para um valor RAW. Tamanho máximo: 32767 bytes se MAX_SQL_STRING_SIZE = EXTENDED 4000 bytes se MAX_SQL_STRING_SIZE = LEGACY
LONG RAW	Dados binários brutos de comprimento variável de até 2 GB.
BLOB	Objeto binário grande. O tamanho máximo é $(4 \text{ GB} - 1) * (\text{parâmetro de inicialização DB_BLOCK_SIZE (8 TB a 128 TB)})$.
BFILE	Dados binários armazenados em um arquivo externo (até 4 GB).
ROWID	String de base 64 que representa o endereço exclusivo de uma linha em sua tabela. Esse tipo de dados destina-se principalmente a valores retornados pela pseudocoluna ROWID

Exemplo: Criando uma Tabela com Diferentes Tipos de Dados

```
CREATE TABLE    print_media(  
  product_id    NUMBER(6) ,  
  id            NUMBER(6) ,  
  desc         VARCHAR2(100) ,  
  composite     BLOB ,  
  msourcetext   CLOB ,  
  finaltext     CLOB ,  
  photo        BLOB ,  
  graphic       BFILE  
);
```

Tipos de Dados de Data



Tipo de Dados	Descrição
TIMESTAMP	Permite o armazenamento de tempo como uma data com segundos fracionais. Armazena o valor do ano, do mês, do dia, da hora, dos minutos e dos segundos dos tipos de dados DATE, bem como o valor de segundos fracionais. Há muitas variações desse tipo de dados, como WITH TIMEZONE e WITH LOCALTIMEZONE.
INTERVAL YEAR TO MONTH	Permite o armazenamento de tempo como um intervalo de anos e meses. Usado para representar a diferença entre dois valores de data e hora nos quais as únicas partes significativas são o ano e o mês.
INTERVAL DAY TO SECOND	Permite o armazenamento de tempo como um intervalo de dias, horas, minutos e segundos; usado para representar a diferença precisa entre dois valores de data e hora.
TIMESTAMP WITH TIME ZONE	Variação de TIMESTAMP que inclui o nome de uma região ou deslocamento de fuso horário em seu valor.

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

13

Você pode usar vários tipos de dados de data.

Exemplo de `TIMESTAMP WITH TIMEZONE`:

```
CREATE TABLE table_tstz (c_id NUMBER, c_tstz TIMESTAMP WITH  
TIME ZONE);  
  
INSERT INTO table_tstz VALUES(1, '01-JAN-2003 2:00:00 AM -  
07:00');
```

Exemplos: Tipos de Dados de Data

- Exemplo do tipo de dados TIMESTAMP:

```
CREATE TABLE table_ts(  
    c_id NUMBER(6),  
    c_ts TIMESTAMP  
);
```

```
INSERT INTO table_ts  
VALUES(1, '01-JAN-2003 2:00:00');
```

Exemplos: Tipos de Dados de Data

- Exemplo de uma tabela com as colunas TIMESTAMP, INTERVAL YEAR TO MONTH e INTERVAL DAY TO SECOND:

```
CREATE TABLE time_table(  
    start_time      TIMESTAMP,  
    duration_1      INTERVAL DAY (6) TO SECOND (5),  
    duration_2      INTERVAL YEAR TO MONTH  
);
```

Opção DEFAULT

- Especifique um valor padrão para uma coluna durante a ação CREATE TABLE
- Essa opção impede que valores nulos sejam inseridos nas colunas quando uma linha é inserida sem um valor para a coluna

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

- Valores literais, expressões ou funções SQL são permitidos

Considere o seguinte exemplo, no qual a instrução insere o valor `NULL` em vez do valor padrão:

```
INSERT INTO hire_dates values(45, NULL);
```

No próximo exemplo, a instrução insere o `SYSDATE` para a coluna `HIRE_DATE` porque esse é o valor `DEFAULT`:

```
INSERT INTO hire_dates(id) values(35);
```


Opção DEFAULT

- O nome de outra coluna ou uma pseudocoluna são valores não permitidos
- O tipo de dados padrão deve corresponder ao tipo de dados da coluna

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

Table created.

Cenário de Caso: Criando Tabelas

Que tal criar as
tabelas para o banco
de dados de
biblioteca
simplificado?



ORACLE
Academy

Dfo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

18

Cenário de Caso: Criando Tabelas

```
CREATE TABLE BOOK_TRANSACTIONS
( ID      VARCHAR2(6),
  TRAN_DATE DATE DEFAULT SYSDATE,
  TYPE    VARCHAR2(10),
  BOOK_ID VARCHAR2(6),
  MEMBER_ID NUMBER(4)
);
```

```
CREATE TABLE AUTHORS
( ID      NUMBER(3),
  NAME    VARCHAR2(60)
);
```

```
CREATE TABLE PUBLISHERS
( ID      NUMBER(2),
  NAME    VARCHAR2(100)
);
```

```
CREATE TABLE MEMBERS
( ID          NUMBER(4),
  FIRST_NAME  VARCHAR2(50),
  LAST_NAME   VARCHAR2(50),
  STREET_ADDRESS VARCHAR2(50),
  CITY        VARCHAR2(20),
  STATE       VARCHAR2(2),
  ZIP         VARCHAR2(10)
);
```

```
CREATE TABLE BOOKS
( ID      VARCHAR2(6),
  TITLE   VARCHAR2(255),
  PUBLISHER_ID NUMBER(2),
  AUTHOR_ID NUMBER(3)
);
```

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

19

Cenário de Caso: Criando Tabelas



**Criando
Tabelas**

```
CREATE TABLE authors(  
  id      NUMBER(3),  
  name    VARCHAR2(60)  
);  
  
CREATE TABLE members(  
  id          NUMBER(4),  
  first_name   VARCHAR2(50),  
  last_name    VARCHAR2(50),  
  street_address VARCHAR2(50),  
  city         VARCHAR2(20),  
  state        VARCHAR2(2),  
  zip         VARCHAR2(10)  
);
```

**Criação
bem-sucedida
de tabelas**



Results	Explain	Describe	Saved SQL	History
Table created.				
0.03 seconds				

Cenário de Caso: Criando Tabelas



**Criando
Tabelas**

```
CREATE TABLE publishers(  
  id NUMBER(2),  
  name VARCHAR2(100) NOT NULL  
);  
  
CREATE TABLE books(  
  id VARCHAR2(6),  
  title VARCHAR2(255) NOT NULL,  
  publisher_id NUMBER(2),  
  author_id NUMBER(3)  
);
```

**Criação
bem-sucedida
de tabelas**



Results	Explain	Describe	Saved SQL	History
Table created.				
0.03 seconds				

Incluindo Restrições

- As restrições impõem regras no nível da tabela
- As restrições garantem a consistência e a integridade do banco de dados
- Os tipos de restrição a seguir são válidos:
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK



Restrições de Integridade dos Dados

Restrições	Descrição
NOT NULL	A coluna não pode conter um valor nulo.
UNIQUE	Os valores de uma coluna ou de uma combinação de colunas devem ser exclusivos para todas as linhas da tabela.
PRIMARY KEY	A coluna (ou uma combinação de colunas) deve conter o valor AND IS NOT NULL exclusivo para todas as linhas.
FOREIGN KEY	A coluna (ou uma combinação de colunas) deve estabelecer e impor uma referência a uma coluna ou uma combinação de colunas em outra (ou na mesma) tabela.
CHECK	Uma condição deve ser verdadeira.

Diretrizes sobre Restrições

- Nomeie uma restrição (caso contrário, o servidor Oracle gerará um nome no formato SYS_Cn)
- Será mais fácil fazer referência às restrições se você atribuir um nome significativo a elas. (Ex. employee_employee_id_pk)
- Crie uma restrição em um dos seguintes momentos:
 - Ao mesmo tempo em que criar a tabela
 - Após a criação da tabela
- Defina uma restrição no nível da coluna ou da tabela
- Exiba uma restrição no dicionário de dados

Constraint	Type
SYS_C0014370	Primary Key

Por exemplo, ao criar uma tabela, se você especificar uma coluna para ser a chave primária sem usar a palavra reservada "CONSTRAINT", o Oracle gerará um nome de restrição, conforme mostrado aqui:

```
CREATE TABLE DEPT_SAMPLE (DEPT_ID NUMBER(2) PRIMARY KEY, DEPARTMENT_ID VARCHAR2(50));
```


Diretrizes sobre Restrições

- As restrições no nível da coluna são incluídas durante a definição da coluna
- As restrições no nível da tabela são definidas no final da definição da tabela e devem fazer referência à coluna ou às colunas às quais a restrição está relacionada
- Funcionalmente, uma restrição no nível de coluna é igual a uma restrição no nível de tabela
- Restrições NOT NULL podem ser definidas somente no nível da coluna
- As restrições que se aplicam a mais de uma coluna devem ser definidas no nível da tabela

Definindo Restrições

- Sintaxe de CREATE TABLE com CONSTRAINTS:

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

Na sintaxe:

- `schema` é o mesmo que o nome do proprietário.
- `table` é o nome da tabela.
- `DEFAULT expr` especificará um valor padrão a ser usado se um valor for omitido na instrução `INSERT`.
- `column` é o nome da coluna.
- `datatype` é o tipo de dados e o tamanho da coluna.
- `column_constraint` é uma restrição de integridade como parte da definição da coluna.
- `table_constraint` é uma restrição de integridade como parte da definição da tabela.

Definindo Restrições

- Sintaxe de restrição no nível da coluna:

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

- Sintaxe de restrição no nível da tabela:

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

Exemplos: Definindo Restrições

- Restrição no nível da coluna:

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

- Restrição no nível da tabela:

```
CREATE TABLE employees(  
  employee_id NUMBER(6),  
  first_name VARCHAR2(20),  
  ...  
  job_id VARCHAR2(10),  
  CONSTRAINT emp_emp_id_pk PRIMARY KEY (employee_id)  
);
```

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

28

Neste exemplo, a restrição de chave primária usa o UID designado para essa entidade e cria a chave primária, a qual pode ser criada no nível da coluna ou da tabela. Mais detalhes sobre restrições de chave primária são fornecidos nos slides 33-34.

Observação: os exemplos contidos neste e nos próximos slides mostram somente uma parte do código usado para criar a tabela employees, e portanto, não podem ser executados da forma como estão.

Restrição NOT NULL

- Garante que valores nulos não sejam permitidos para a coluna:

EMPLOYEE_ID	FIRST_NAME	SALARY	COMMISSION_PCT	DEPARTMENT_ID	EMAIL	PHONE_NUMBER	HIRE_DATE
100	Steven	24000	-	90	SKING	515.123.4567	17-Jun-1987
101	Neena	17000	-	90	NKOCHHAR	515.123.4568	21-Sep-1989
102	Lex	17000	-	90	LDEHAAN	515.123.4569	13-Jan-1993
200	Jennifer	4400	-	10	JWHALEN	515.123.4444	17-Sep-1987
205	Shelley	12000	-	110	SHIGGINS	515.123.8080	07-Jun-1994
206	William	8300	-	110	WGIETZ	515.123.8181	07-Jun-1994
141	Trenna	3500	-	50	TRAJS	650.121.8009	17-Oct-1995

↑
Restrição NOT NULL
(A Chave Primária impõe
a restrição NOT NULL.)

Restrição
NOT NULL

↑ Ausência de restrição NOT NULL (Qualquer
linha pode conter um valor nulo para essa
coluna.)

A criação de restrições NOT NULL impõe os atributos obrigatórios no design.

Restrição NOT NULL

- Pode ser definida SOMENTE no nível 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 CONSTRAINT hire_date_nn  
                  NOT NULL ,  
  ...  
);
```

Restrição UNIQUE

- Uma restrição de integridade UNIQUE exige que todos os valores de uma coluna ou de um conjunto de colunas sejam exclusivos;
- Se a restrição UNIQUE tiver mais de uma coluna, esse grupo de colunas será denominado chave exclusiva composta
- As restrições UNIQUE permitem a entrada de nulos
- Um nulo em uma coluna (ou em todas as colunas de uma chave UNIQUE composta) sempre satisfaz a uma restrição UNIQUE

Observação: devido ao mecanismo de pesquisa das restrições UNIQUE em mais de uma coluna, não é possível ter valores idênticos nas colunas não nulas de uma restrição de chave UNIQUE composta parcialmente nula.

Restrição UNIQUE

Restrição UNIQUE

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	EMAIL
100	King	SKING
101	Kochhar	NKOCHHAR
102	De Haan	LDEHAAN
200	Whalen	JWHALEN
205	Higgins	SHIGGINS

...



INSERT INTO

208 Smith JSMITH

Permitido

209 Smith JSMITH

Não permitido: já existe

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

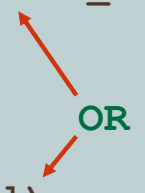
Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

32

Restrição UNIQUE

- Definida no nível da tabela ou da coluna:

```
CREATE TABLE employees (  
    employee_id      NUMBER(6),  
    last_name        VARCHAR2(25),  
    email            VARCHAR2(25) CONSTRAINT  
                                emp_email_uk UNIQUE,  
    salary            NUMBER(8,2),  
    commission_pct   NUMBER(2,2),  
    hire_date        DATE,  
    ...  
    CONSTRAINT emp_email_uk UNIQUE(email)  
);
```



Uma chave exclusiva composta é definida no nível da tabela. Por exemplo:

```
ALTER TABLE DEPT_SAMPLE ADD CONSTRAINT unq_dept_det UNIQUE (DEPT_ID, DEPARTMENT_NAME) ;
```

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

Restrição PRIMARY KEY

- Uma restrição PRIMARY KEY cria uma chave primária para a tabela
- Somente uma chave primária pode ser criada para cada tabela
- A restrição PRIMARY KEY é uma coluna ou um conjunto de colunas que identifica de forma exclusiva cada linha de uma tabela
- Nenhuma coluna que faça parte da chave primária pode conter um valor nulo
- Uma chave primária composta deve ser criada no nível da tabela

Por exemplo:

```
create table dept(  
  dept_id number(8),  
  dept_name varchar2(30),  
  loc_id number(4),  
  constraint pk_dept primary key(dept_id,loc_id));
```

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

Restrição 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

... Não permitido
(valor nulo)

 INSERT INTO

NULL	Public Accounting	124	2500
50	Finance	124	1500

 Not allowed (50 already exists)

- Observação: consulte no slide 27 exemplos de codificação de restrição de chave primária.

Restrição FOREIGN KEY

- A restrição FOREIGN KEY (ou de integridade referencial) designa uma coluna ou uma combinação de colunas como uma chave estrangeira
- Estabelece um relacionamento com uma chave primária na mesma tabela ou em outra
- Veja a seguir as diretrizes para restrições de chave estrangeira:
 - Um valor de chave estrangeira deve corresponder a um valor existente na tabela pai ou ser NULL
 - As chaves estrangeiras se baseiam em valores de dados e são ponteiros puramente lógicos, em vez de físicos

Restrição FOREIGN KEY

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
...		...	

FOREIGN KEY

EMPLOYEE_ID	SALARY	DEPARTMENT_ID
100	24000	90
101	17000	90
102	17000	90
...		

INSERT INTO

200	Ford	9
200	Ford	60

Não permitido
(9 não existe)

Permitido

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

37

Restrição FOREIGN KEY

- Definida no nível da tabela:

```
CREATE TABLE employees (  
    employee_id      NUMBER(6) ,  
    last_name        VARCHAR2(25) ,  
    email            VARCHAR2(25) ,  
    salary           NUMBER(8,2) ,  
    commission_pct   NUMBER(2,2) ,  
    hire_date        DATE ,  
    ...  
    department_id    NUMBER(4) ,  
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)  
        REFERENCES departments(department_id)  
);
```

Uma chave estrangeira composta deve ser criada usando a definição no nível da tabela.

No slide, o exemplo define uma restrição FOREIGN KEY na coluna DEPARTMENT_ID da tabela EMPLOYEES, usando a sintaxe no nível da tabela. O nome da restrição é EMP_DEPT_FK.

A chave estrangeira também poderá ser definida no nível da coluna, desde que a restrição se baseie em uma única coluna. A sintaxe é diferente uma vez 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) ,  
    ...  
)
```

Restrição FOREIGN KEY

- Definida no nível da coluna:

```
CREATE TABLE employees (  
  employee_id      NUMBER(6) ,  
  last_name        VARCHAR2(25) ,  
  email            VARCHAR2(25) ,  
  salary           NUMBER(8,2) ,  
  commission_pct   NUMBER(2,2) ,  
  hire_date        DATE ,  
  ...  
  department_id    NUMBER(4) CONSTRAINT emp_dept_fk  
                  REFERENCES departments(department_id)  
) ;
```

Uma chave estrangeira composta deve ser criada no nível da tabela; por exemplo:

```
CREATE TABLE supplier  
( sup_id numeric(15) not null,  
  sup_name varchar2(45) not null,  
  contact_name varchar2(45),  
  CONSTRAINT sup_pk PRIMARY KEY (sup_id, sup_name)  
) ;
```

Restrição FOREIGN KEY: Palavras-chave

- **FOREIGN KEY:** define a coluna da tabela filho no nível da restrição da tabela
- **REFERENCES:** identifica a tabela e a coluna na tabela pai
- **ON DELETE CASCADE:** exclui as linhas dependentes na tabela filho quando uma linha na tabela pai é excluída
- **ON DELETE SET NULL:** converte valores de chave estrangeira dependentes em nulos

Sem as opções **ON DELETE CASCADE** ou **ON DELETE SET NULL**, a linha da tabela pai não poderá ser excluída se estiver referenciada na tabela filho. E essas palavras-chave não podem ser usadas na sintaxe no nível da coluna.

Restrição CHECK

- Define uma condição que cada linha deve atender
- Não pode fazer referência a colunas de outras tabelas

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

Para satisfazer à restrição, cada linha da tabela deve tornar a condição TRUE ou desconhecida (devido a um valor nulo).

Uma única coluna pode ter várias restrições CHECK que façam referência à coluna em sua definição. Não há limite para o número de restrições CHECK que você pode definir em uma coluna.

A restrição CHECK pode ser definida no nível da tabela ou da coluna.

CREATE TABLE: Exemplo de Restrição CHECK

```
CREATE TABLE employees(  
    employee_id      NUMBER(6),  
    last_name        VARCHAR2(25),  
    email            VARCHAR2(25),  
    salary            NUMBER(8,2),  
    commission_pct   NUMBER(2,2),  
    hire_date        DATE,  
    ...  
    CONSTRAINT hire_date_min CHECK  
        (hire_date > '01-JAN-2018')  
);
```

Cenário de Caso: Criando Tabelas

Que tal adicionar as
restrições às tabelas
do banco de dados de
biblioteca
simplificado?



Cenário de Caso: Adicionando Restrições

```
CREATE TABLE authors(  
  id          NUMBER(3),  
  name        VARCHAR2(60),  
  CONSTRAINT  atr_id_pk PRIMARY KEY (ID)  
);
```

```
CREATE TABLE members(  
  id          NUMBER(4),  
  first_name   VARCHAR2(50),  
  last_name    VARCHAR2(50),  
  street_address VARCHAR2(50),  
  city         VARCHAR2(20),  
  state        VARCHAR2(2),  
  zip          VARCHAR2(10),  
  CONSTRAINT  mbr_id_pk PRIMARY KEY (ID)  
);
```

Cenário de Caso: Adicionando Restrições

```
CREATE TABLE publishers(  
  id          NUMBER(2),  
  name        VARCHAR2(100) NOT NULL,  
  CONSTRAINT plr_id_pk PRIMARY KEY (ID)  
);  
  
CREATE TABLE books(  
  id          VARCHAR2(6),  
  title       VARCHAR2(255) NOT NULL,  
  publisher_id NUMBER(2),  
  author_id   NUMBER(3),  
  CONSTRAINT bok_id_pk PRIMARY KEY (ID),  
  CONSTRAINT bok_atr_fk FOREIGN KEY (author_id)  
                        REFERENCES authors(id),  
  CONSTRAINT bok_plr_fk FOREIGN KEY (publisher_id)  
                        REFERENCES publishers(id)  
);
```

Cenário de Caso: Adicionando Restrições

```
CREATE TABLE book_transactions(  
    id                VARCHAR2(6) ,  
    tran_date         DATE DEFAULT SYSDATE NOT NULL ,  
    type              VARCHAR2(10) ,  
    book_id           VARCHAR2(6) ,  
    member_id         NUMBER(4) ,  
    CONSTRAINT btn_id_pk PRIMARY KEY (ID) ,  
    CONSTRAINT bok_btn_fk FOREIGN KEY (book_id)  
                                REFERENCES books(id) ,  
    CONSTRAINT bok_mbr_fk FOREIGN KEY (member_id)  
                                REFERENCES members(id)  
);
```

Data Definition Language

- A criação de tabelas faz parte do DDL (Data Definition Language) do SQL
- Outras instruções **DDL** incluem:
 - **ALTER**: para modificar a estrutura de um objeto
 - **DROP**: para remover um objeto do banco de dados
 - **RENAME**: para renomear um objeto de banco de dados

Instrução ALTER TABLE

- Use a instrução ALTER TABLE para alterar a estrutura da tabela:
 - Adicionar uma coluna
 - Modificar uma definição de coluna existente
 - Definir um valor padrão para a nova coluna
 - Eliminar uma coluna
 - Renomear uma coluna
 - Alterar uma tabela para o status somente leitura

Depois de criar uma tabela, talvez você precise alterar a sua estrutura por alguma das seguintes razões:

- Você omitiu uma coluna.
- A definição ou o nome da coluna precisa ser alterado.
- É necessário remover colunas.
- Você deseja colocar a tabela no modo somente leitura.

Instrução ALTER TABLE

- Use a instrução ALTER TABLE para adicionar, modificar ou eliminar colunas:

```
ALTER TABLE table
ADD          (column data type [DEFAULT expr]
              [, column data type]...);
```

```
ALTER TABLE table
MODIFY       (column data type [DEFAULT expr]
              [, column data type]...);
```

```
ALTER TABLE table
DROP (column [, column] ...);
```

Na sintaxe:

- *table* é o nome da tabela.
- ADD | MODIFY | DROP é o tipo de modificação.
- *column* é o nome da coluna.
- *data type* é o tipo de dados e o tamanho da coluna.
- DEFAULT *expr* especifica o valor padrão de uma coluna.

Adicionando uma Coluna

- Use a cláusula ADD para adicionar colunas:

```
ALTER TABLE employees  
ADD termination_date DATE;
```

- A nova coluna se torna a última:

EMPLOYEE_ID	LAST_NAME	HIRE_DATE	TERMINATION_DATE
100	King	17-Jun-1987	-
101	Kochhar	21-Sep-1989	-
102	De Haan	13-Jan-1993	-
200	Whalen	17-Sep-1987	-

Observação: se uma tabela já contiver linhas quando uma coluna for adicionada, a nova coluna será inicialmente nula ou terá o valor padrão para todas as linhas. Você poderá adicionar uma coluna `NOT NULL` obrigatória a uma tabela que contenha dados nas outras colunas somente se especificar um valor padrão. Você pode adicionar uma coluna `NOT NULL` a uma tabela vazia sem o valor padrão.

Consulte no slide 9 o código para criar essa tabela.

Modificando uma Coluna

- Você pode alterar o tipo de dados, o tamanho e o valor padrão de uma coluna:

```
ALTER TABLE employees  
MODIFY first_name VARCHAR2(30);
```

- Um valor padrão alterado afeta somente as inserções subsequentes na tabela
- As modificações estão sujeitas a certas condições

Veja a seguir as diretrizes para modificar uma coluna:

- Você pode aumentar a largura ou a precisão de uma coluna numérica.
- Você pode aumentar a largura de colunas de caracteres.
- Você poderá reduzir a largura de uma coluna se...
 - A coluna contiver apenas valores nulos.
 - A tabela não tiver linhas.
 - A redução na largura da coluna não for inferior aos valores existentes nessa coluna.
- Você poderá alterar o tipo de dados se a coluna contiver apenas valores nulos. A exceção são as conversões de CHAR em VARCHAR2, que podem ser feitas com dados nas colunas.
- Você só poderá converter uma coluna CHAR no tipo de dados VARCHAR2 ou converter uma coluna VARCHAR2 no tipo de dados CHAR se a coluna contiver valores nulos ou se você não alterar o tamanho.
- Uma alteração no valor padrão de uma coluna afeta somente as inserções subsequentes na tabela.
- Você pode adicionar uma restrição NOT NULL usando as cláusulas MODIFY.

Eliminando uma Coluna

- Use a cláusula **DROP COLUMN** para eliminar colunas desnecessárias:

```
ALTER TABLE employees  
DROP (termination_date);
```

Table altered.

EMPLOYEE_ID	LAST_NAME	HIRE_DATE
100	King	17-Jun-1987
101	Kochhar	21-Sep-1989
102	De Haan	13-Jan-1993
200	Whalen	17-Sep-1987

...

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

52

Veja a seguir as diretrizes para eliminar uma coluna:

- A coluna pode ou não conter dados.
- Com a instrução **ALTER TABLE DROP COLUMN**, é possível eliminar apenas uma coluna por vez.
- A tabela deve ter, pelo menos, uma coluna após a alteração.
- Após eliminar uma coluna, não é possível recuperá-la.
- Uma chave primária que é referenciada por outra coluna não poderá ser eliminada, a menos que a opção cascata seja adicionada.
- A eliminação de uma coluna poderá demorar se a coluna tiver um grande número de valores. Nesse caso, talvez seja melhor defini-la como não usada e eliminá-la quando houver menos usuários no sistema. Dessa forma, você evitará bloqueios estendidos.

Opção SET UNUSED

- A opção SET UNUSED marca uma ou mais colunas como não usadas para que elas possam ser eliminadas quando a demanda de recursos do sistema for menor
- Use a opção SET UNUSED para marcar uma ou mais colunas como não usadas
- Use a opção DROP UNUSED COLUMNS para remover as colunas marcadas como não usadas

As colunas não usadas são tratadas como se tivessem sido eliminadas, embora os dados das colunas permaneçam nas linhas da tabela.

Após marcar uma coluna como não usada, você não terá acesso a ela. Uma consulta `SELECT *` não recupera dados de colunas não usadas. Além disso, os nomes e os tipos das colunas marcadas como não usadas não são exibidos durante uma instrução `DESCRIBE`, e você pode adicionar uma nova coluna com o mesmo nome que uma coluna não usada.

Você pode especificar a palavra-chave `ONLINE` para indicar que são permitidas operações DML (Data Manipulation Language) na tabela ao marcar a coluna ou as colunas como `UNUSED`. O exemplo de código a seguir mostra o uso de `SET UNUSED COLUMN`, que define permanentemente uma coluna como não usada, com a inclusão da palavra-chave `ONLINE`:

```
ALTER TABLE dept80 SET UNUSED(hire_date) ONLINE;
```

A informação `SET UNUSED` é armazenada na view de dicionário `USER_UNUSED_COL_TABS`.

Observação: as diretrizes para definir uma coluna como `UNUSED` são semelhantes às usadas para eliminar uma coluna.

Opção SET UNUSED

```
ALTER TABLE <table_name>
SET UNUSED(<column_name> [ , <column_name>]);
OR
ALTER TABLE <table_name>
SET UNUSED COLUMN <column_name> [ , <column_name>];
```

```
ALTER TABLE dept
SET UNUSED (dname);
```

```
ALTER TABLE <table_name>
DROP UNUSED COLUMNS;
```

```
ALTER TABLE dept
DROP UNUSED COLUMNS;
```

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

54

Ao definir uma coluna como `UNUSED`, você tem a opção de eliminar essa coluna.

Você pode usar `DROP UNUSED COLUMNS` para remover da tabela todas as colunas marcadas atualmente como não usadas. Você pode usar essa instrução quando quiser recuperar o espaço em disco extra das colunas não usadas na tabela. Se a tabela não contiver colunas não usadas, a instrução será retornada sem erros.

Observação: uma instrução `DROP UNUSED COLUMNS` subsequente remove fisicamente todas as colunas não usadas de uma tabela, de maneira semelhante a uma instrução `DROP COLUMN`.

Cenário de Caso: Alterando Tabelas



Corpo Docente

Sean, ao revisar a tabela **AUTHORS**, percebi o seguinte:

O campo de endereço de e-mail do autor está faltando.

É necessário aumentar o tamanho da coluna de nome do autor.

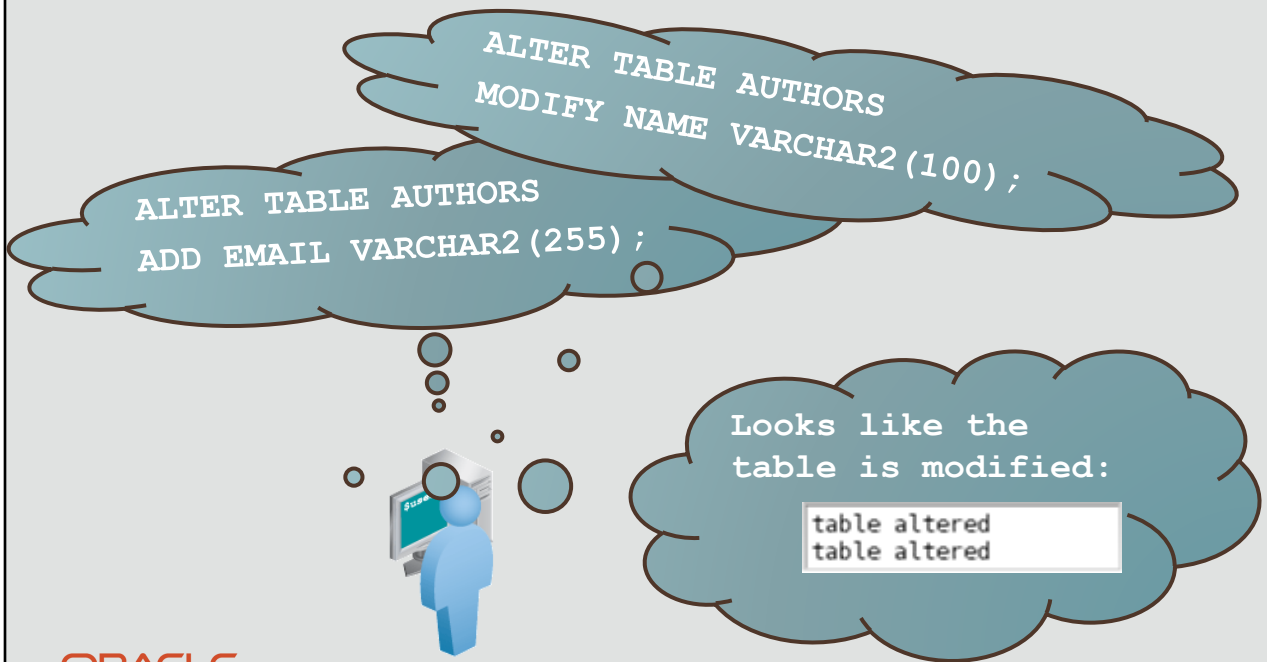
Você pode fazer essas alterações?

Claro, posso fazer isso. Como a modificação está adicionando uma nova coluna e aumentando o seu tamanho, isso não deve ser um problema.



Aluno

Cenário de Caso: Alterando Tabelas



Tabelas Somente Leitura

- Você pode usar a sintaxe ALTER TABLE para:
 - Colocar uma tabela no modo somente leitura, o que impede alterações DDL ou DML durante a manutenção da tabela
 - Colocar a tabela no modo de leitura/gravação

```
ALTER TABLE dept READ ONLY;
```

```
-- melakukan pemeliharaan tabel, lalu  
-- mengembalikan mode tabel ke baca/tulis
```

```
ALTER TABLE dept READ WRITE;
```

Veja a seguir as diretrizes para colocar uma tabela no modo somente leitura:

- Você pode especificar `READ ONLY` para colocar uma tabela no modo somente leitura.
- Quando uma tabela está no modo somente leitura, não é possível emitir instruções DML que afetem a tabela nem qualquer instrução `SELECT ... FOR UPDATE`.
- Você poderá emitir instruções DDL desde que elas não modifiquem nenhum dado da tabela.
- Operações em índices associados à tabela são permitidas quando a tabela está no modo somente leitura.
- Especifique `READ/WRITE` a fim de retornar uma tabela somente leitura para o modo de leitura/gravação.

Observação: é possível eliminar uma tabela que esteja no modo `READ ONLY`. Como o comando `DROP` é executado somente no dicionário de dados, o acesso ao conteúdo da tabela não é necessário. O espaço usado pela tabela não será recuperado até que o tablespace seja definido como de leitura/gravação novamente. Em seguida, as alterações necessárias poderão ser feitas nos cabeçalhos de segmentos de bloco etc.

Eliminando uma Tabela

- Move uma tabela para a lixeira
- Removerá a tabela e seus dados se a cláusula PURGE for especificada
- Invalida objetos dependentes e remove privilégios de objeto na tabela

```
DROP TABLE dept;
```

Table dropped.

ORACLE
Academy

DFo 6-3
Linguagem de Definição de Dados (DDL)

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados.

58

A menos que você especifique a cláusula PURGE, a instrução DROP TABLE não resultará na liberação de espaço de volta para o tablespace para uso por outros objetos, e o espaço continuará a ser contado em relação à cota de espaço do usuário. A eliminação de uma tabela invalida objetos dependentes e remove privilégios de objeto na tabela.

Quando você elimina uma tabela, o banco de dados perde todos os dados da tabela bem como todos os índices associados a ela.

Sintaxe

```
DROP TABLE table [PURGE]
```

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

Veja a seguir as diretrizes para eliminar uma tabela:

- Todos os dados são excluídos da tabela.
- Todas as views e sinônimos permanecem, mas eles são inválidos.
- É feito commit de todas as transações pendentes.
- Somente o criador da tabela ou um usuário com o privilégio DROP ANY TABLE pode remover uma tabela.

Exercício do Projeto

- DFo_6_3_Project
 - Banco de Dados da Oracle Baseball League
 - Usar o DDL para criar e manter tabelas de banco de dados



Resumo

- Nesta lição, você deverá ter aprendido a:
 - Identificar as etapas necessárias para criar tabelas de banco de dados
 - Descrever a finalidade do DDL
 - Listar as operações DDL necessárias para criar e manter as tabelas de um banco de dados



The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

ORACLE

Academy