



Fundamentos de bases de datos

6-3

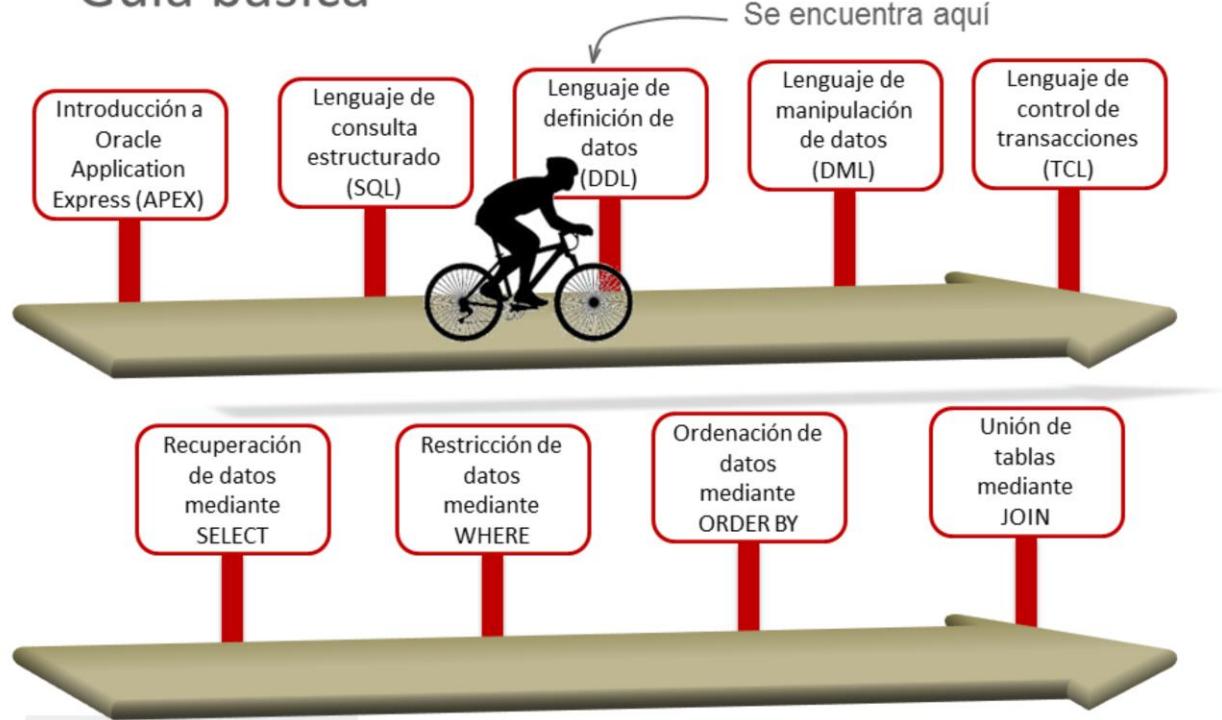
Lenguaje de definición de datos (DDL)



ORACLE® ACADEMY

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

Guía básica



ORACLE

ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

Objetivos

En esta lección se abordan los siguientes objetivos:

- Identificar los pasos necesarios para crear tablas de base de datos
- Describir la finalidad del lenguaje de definición de datos (DDL)
- Mostrar las operaciones DDL necesarias para crear y mantener las tablas de una base de datos



Objetos de base de datos

Objeto	Descripción
Tabla	Es la unidad básica de almacenamiento; consta de filas.
Vista	Representa de forma lógica subconjuntos de datos de una o más tablas.
Secuencia	Genera valores numéricos.
Índice	Mejora el rendimiento de algunas consultas.
Sinónimo	Ofrece un nombre alternativo para un objeto.

Nota: En este curso, vamos a crear y recuperar información de la unidad básica de almacenamiento, las tablas. Hay más objetos de base de datos disponibles que los enumerados, pero no se tratan en este curso.

Reglas de nomenclatura para tablas y columnas

Los nombres de tabla y de columna deben:

- Empezar por una letra
- Tener entre 1 y 30 caracteres
- Contener solo A–Z, a–z, 0–9, _, \$ y #
- No ser un duplicado de otro nombre de objeto propiedad del mismo usuario
- No ser una palabra reservada del servidor de Oracle



Nota: Los nombres no son sensibles a mayúsculas/minúsculas. Por ejemplo, EMPLOYEES se considera lo mismo que eMPloyees o eMployees. Sin embargo, los identificadores entre comillas son sensibles a mayúsculas/minúsculas.

Para obtener una lista completa de las palabras reservadas, consulte:

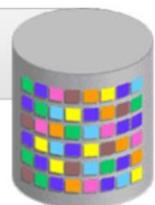
https://docs.oracle.com/cd/B28359_01/appdev.111/b31231/appb.htm#CJHIIICD

Sentencia CREATE TABLE

Para emitir una sentencia CREATE TABLE, se debe disponer de lo siguiente:

- El privilegio CREATE TABLE
- Un área de almacenamiento

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



ACADEMY

DFO 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

7

Para crear una tabla, un usuario debe tener el privilegio CREATE TABLE y un área de almacenamiento en la que crear los objetos. El administrador de la base de datos (DBA) utiliza sentencias de lenguaje de control de datos (DCL) para otorgar privilegios a los usuarios.

En la sintaxis:

- schema es el mismo nombre que el del propietario.
- table es el nombre de la tabla.
- DEFAULT expr especifica un valor por defecto si se omite un valor en la sentencia INSERT.
- column es el nombre de la columna.
- datatype es el tipo de dato y la longitud de la columna.

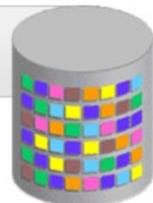
Nota: Se necesita el privilegio CREATE ANY TABLE para crear una tabla en cualquier esquema distinto del esquema del usuario.

Sentencia CREATE TABLE

Especifique en la sentencia:

- Nombre de la tabla
- Nombre de columna, tipo de dato de columna, tamaño de columna
- Restricciones de integridad (opcional)
- Valores por defecto (opcional)

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



ACADEMY

DFO 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

8

Creación de tablas

- Crear una tabla:

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

- Para confirmar que se ha creado la tabla, ejecute el comando DESCRIBE.

Nota: Puede ver la lista de las tablas que posee realizando consultas en el diccionario de datos. Por ejemplo, `select table_name from user_tables;`

Para obtener más información sobre las tablas del diccionario de datos, consulte:
<https://docs.oracle.com/database/121/GMSWN/apc.htm#GMSWN600>

Creación de tablas

- Confirmar la creación de la tabla:

```
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 dato

Tipo de dato	Descripción
VARCHAR2(size)	Datos de caracteres de longitud variable (Se debe especificar un tamaño máximo; el tamaño mínimo es 1). Tamaño máximo: 32767 bytes si MAX_SQL_STRING_SIZE = EXTENDED 4000 bytes si MAX_SQL_STRING_SIZE = LEGACY
CHAR(size)	Datos de tipo carácter de longitud fija de longitud (size) en bytes. (El tamaño por defecto y mínimo es 1; el tamaño máximo es 2000)
NUMBER(p, s)	Datos numéricos de longitud variable. La precisión es p, y la escala, s. (La precisión es el número total de dígitos decimales, y la escala el número de dígitos a la derecha del punto decimal; la precisión puede ir de 1 a 38, y la escala de -84 a 127).
DATE	Valores de fecha y hora hasta el segundo más próximo entre el 1 de enero del 4712 a.C. y el 31 de diciembre del 9999 d.C.
LONG	Datos de caracteres de longitud variable (hasta 2 GB).



ACADEMY

DFo 6-3

Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

11

Tipos de dato

Tipo de dato	Descripción
CLOB	Objeto grande de caracteres (CLOB) que contiene caracteres de un solo byte o multibyte. El tamaño máximo es $(4 \text{ GB} - 1) * (\text{DB_BLOCK_SIZE})$; almacena datos del juego de caracteres nacional.
NCLOB	CLOB que contiene caracteres Unicode. Se soportan los juegos de caracteres de ancho fijo y variable y ambos utilizan el juego de caracteres nacional de la base de datos. El tamaño máximo es $(4 \text{ GB} - 1) * (\text{tamaño de bloque de base de datos})$; almacena datos del juego de caracteres nacional.
RAW (Size)	Datos binarios raw con una longitud en bytes especificada por <i>size</i> . Debe especificar <i>size</i> para un valor RAW. El tamaño (<i>size</i>) máximo es: 32767 bytes si <code>MAX_SQL_STRING_SIZE = EXTENDED</code> 4000 bytes si <code>MAX_SQL_STRING_SIZE = LEGACY</code>
LONG RAW	Datos binarios raw de longitud variable hasta 2 GB.
BLOB	Objeto grande binario. El tamaño máximo es $(4 \text{ GB} - 1) * (\text{parámetro de inicialización DB_BLOCK_SIZE (de 8 TB a 128 TB)})$.
BFILE	Datos binarios almacenados en un archivo externo (hasta 4 GB).
ROWID	Cadena de base 64 que representa la dirección única de una fila en su tabla correspondiente. Este tipo de dato es principalmente para valores devueltos por la pseudocolumna ROWID



ACADEMY

DFo 6-3

Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

12

Ejemplo: Creación de una tabla con diferentes tipos de datos

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



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

13

Tipos de dato de fecha



Tipo de dato	Descripción
TIMESTAMP	Permite almacenar los datos de tiempo como fecha con segundos fraccionarios. Almacena el valor de año, mes, día, hora, minuto y segundo del tipo de dato DATE, así como el valor para segundos fraccionarios. Existen diversas variaciones de este tipo de dato, como, por ejemplo, WITH TIMEZONE y WITH LOCALTIMEZONE.
INTERVAL YEAR TO MONTH	Permite almacenar el tiempo como un intervalo de años y meses. Se utiliza para representar la diferencia entre dos valores de fecha y hora en los que las únicas partes significativas son el año y el mes.
INTERVAL DAY TO SECOND	Permite almacenar el tiempo como un intervalo de días, horas, minutos y segundos; se utiliza para representar la diferencia exacta entre dos valores de fecha y hora.
TIMESTAMP WITH TIME ZONE	Variante de TIMESTAMP que incluye el nombre de la región de zona horaria o el desplazamiento de zona horaria en su valor.

Puede utilizar varios tipos de dato de fecha.

Ejemplo 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');
```

Ejemplos: Tipos de dato de fecha

- Ejemplo de tipo de dato TIMESTAMP:

```
CREATE TABLE table_ts(c_id NUMBER(6), c_ts TIMESTAMP);  
INSERT INTO table_ts VALUES(1, '01-JAN-2003 2:00:00');
```

- Ejemplo de una tabla con las columnas 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);
```



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 15

Opción DEFAULT

- Especifica un valor por defecto para una columna durante la operación CREATE TABLE.
- Esta opción evita que se introduzcan valores nulos en las columnas si se inserta una fila sin un valor para la columna.

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

- Los valores literales, expresiones o funciones SQL son valores válidos.



Observe el siguiente ejemplo, donde la sentencia inserta el valor NULL en lugar del valor por defecto:

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

En el siguiente ejemplo, la sentencia inserta SYSDATE para la columna HIRE_DATE, ya que es el valor DEFAULT:

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

Opción DEFAULT

- El nombre de otra columna o una pseudocolumna son valores no válidos.
- El tipo de dato por defecto debe coincidir con el tipo de dato de la columna.

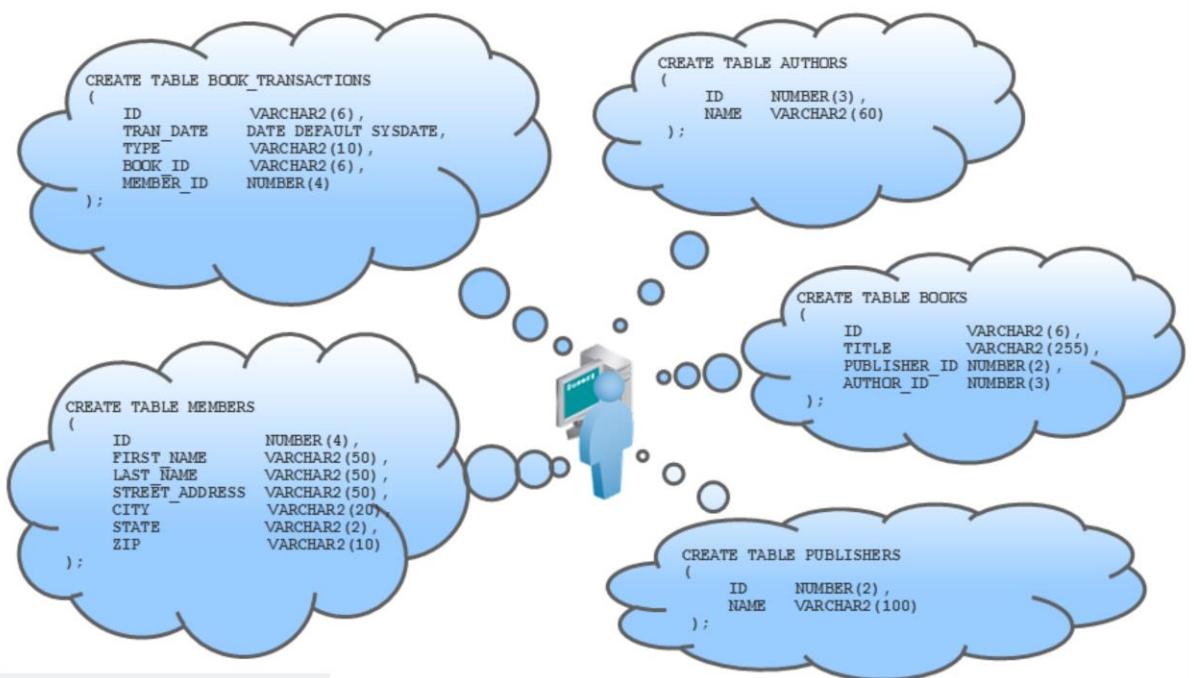
```
CREATE TABLE hire_dates
  (id          NUMBER(8),
   hire_date DATE DEFAULT SYSDATE);
```

Table created.

Escenario de caso: Creación de tablas



Escenario de caso: Creación de tablas



ORACLE

ACADEMY

DFo 6-3

Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

19

Escenario de caso: Creación de tablas



Creación
de tablas

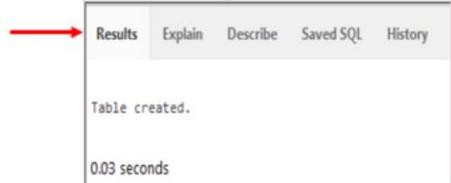
```
CREATE TABLE AUTHORS
(
ID          NUMBER(3),
NAME        VARCHAR2(60)
);

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)
);

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)
);
```

Creación
correcta de
tablas



Inclusión de restricciones

- Las restricciones aplican reglas en el nivel de tabla.
- Las restricciones garantizan la consistencia e integridad de la base de datos.
- Los siguientes tipos de restricciones son válidos:
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK



Restricciones de integridad de datos

Restricciones	Descripción
NOT NULL	La columna no puede contener un valor nulo.
UNIQUE	Los valores de una columna o una combinación de columnas deben ser únicos para todas las filas de la tabla.
PRIMARY KEY	La columna (o una combinación de columnas) debe contener el valor AND IS NOT NULL único para todas las filas.
FOREIGN KEY	La columna (o una combinación de columnas) debe establecer y aplicar una referencia a una columna o una combinación de columnas de otra tabla (o de la misma).
CHECK	Una condición debe ser true.

Directrices de restricción

- Asignar un nombre a una restricción (de lo contrario, el servidor de Oracle genera un nombre con el formato SYS_Cn).
- Es fácil hacer referencia a las restricciones si se les asigna un nombre significativo. (Por ejemplo, employee_employee_id_pk)
- Crear una restricción en uno de los siguientes momentos:
 - En el momento de la creación de la tabla
 - Después de la creación de la tabla
- Definir una restricción en el nivel de columna o de tabla.
- Ver una restricción en el diccionario de datos.

Constraint	Type
SYS_C0014370	Primary Key

Por ejemplo, al crear una tabla, si especifica una columna para que sea la clave primaria sin utilizar la palabra reservada "CONSTRAINT", Oracle genera un nombre de restricción, como se muestra a continuación:

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

Directrices de restricción

- Las restricciones de nivel de columna se incluyen al definir la columna.
- Las restricciones de nivel de tabla se definen al final de la definición de tabla y deben hacer referencia a la columna o las columnas a las que pertenece la restricción.
- Funcionalmente, una restricción de nivel de columna es lo mismo que una restricción de nivel de tabla.
- Las restricciones NOT NULL solo se pueden definir en el nivel de columna.
- Las restricciones que se aplican a más de una columna se deben especificar en el nivel de tabla.

Definición de restricciones

- Sintaxis de CREATE TABLE con CONSTRAINTS:

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

En la sintaxis:

- schema es el mismo nombre que el del propietario.
- table es el nombre de la tabla.
- DEFAULT expr especifica un valor por defecto que se utiliza si se omite un valor en la sentencia INSERT.
- column es el nombre de la columna.
- datatype es el tipo de dato y la longitud de la columna.
- column_constraint es una restricción de integridad como parte de la definición de columna.
- table_constraint es una restricción de integridad como parte de la definición de tabla.

Definición de restricciones

- Sintaxis de restricción de nivel de columna:

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

- Sintaxis de restricción de nivel de tabla:

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

Ejemplos: Definición de restricciones

- Restricción de nivel de columna:

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

- Restricción de nivel de tabla:

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

En este ejemplo, la restricción de clave primaria utiliza el UID designado para dicha entidad y crea la clave primaria (esta se puede crear en el nivel de columna o de tabla); en las diapositivas 33 a 34 encontrará más información sobre las restricciones de clave primaria.

Nota: Los ejemplos de esta **diapositiva y los de las siguientes** solo muestran una parte del código utilizado para crear la tabla employees y, por lo tanto, no se pueden ejecutar tal y como aparecen.

Restricción NOT NULL

- Garantiza que no se permiten los valores nulos para la columna:

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	515.122.4569	13-Jan-1993
200	Jennifer	4400	-	10	JWHALEN	515.123.4444	17-Sep-1987
205	Shelley	12000	-	110	SHIGINS	515.123.8000	07-Jun-1994
206	William	8300	-	110	WGIETZ	515.123.8181	07-Jun-1994
149	Eleni	10500	.2	80	EZLOTKEY	011.44.1344.429018	29-Jan-2000
174	Ellen	11000	.3	80	EABEL	011.44.1644.429267	11-May-1996
176	Jonathon	8600	.2	80	JTAYLOR	011.44.1644.429265	24-Mar-1998
178	Kimberely	7000	.15	-	KGRANT	011.44.1644.429263	24-May-1999

↑
Restricción NOT NULL
(La clave primaria aplica la restricción NOT NULL)

↑
Restricción
NOT NULL

Ausencia de la restricción NOT NULL (cualquier fila puede contener un valor nulo para esta columna)

La creación de restricciones NOT NULL aplica los atributos obligatorios en el diseño.

Restricción NOT NULL

- **Solo** se puede definir en el nivel de columna:

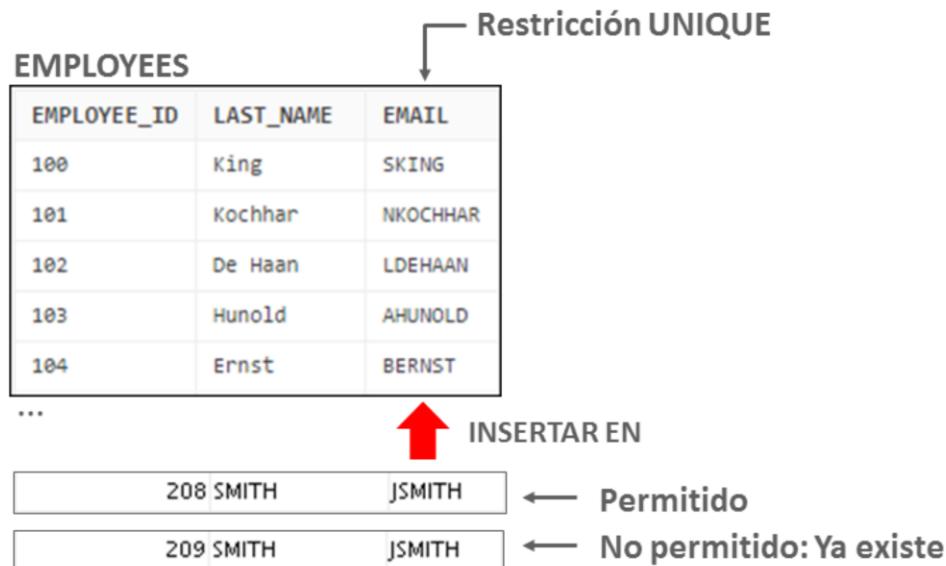
```
CREATE TABLE employees (
    employee_id      NUMBER(6),
    last_name        VARCHAR2(25) NOT NULL,
    email            VARCHAR2(25),
    salary           NUMBER(8,2),
    commission_pct   NUMBER(2,2),
    hire_date        DATE CONSTRAINT hire_date_nn NOT NULL,
    ...
)
```

Restricción UNIQUE

- Una restricción de integridad de clave UNIQUE requiere que todos los valores de la columna o de un juego de columnas sean únicos.
- Si la restricción UNIQUE tiene más de una columna, el grupo de columnas se denomina clave única compuesta.
- Las restricciones UNIQUE permiten la entrada de valores nulos.
- Un valor nulo en una columna (o en todas las columnas de una clave UNIQUE compuesta) cumple siempre una restricción UNIQUE.

Nota: Debido al mecanismo de búsqueda de restricciones UNIQUE en más de una columna, no puede tener valores idénticos en las columnas no nulas de una restricción de clave UNIQUE compuesta parcialmente nula.

Restricción UNIQUE



Restricción UNIQUE

- Definida en el nivel de tabla o de columna:

```
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));
```



Una clave única compuesta se define en el nivel de tabla. Por ejemplo:

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

Nota: El servidor de Oracle aplica la restricción UNIQUE mediante la creación implícita de un índice único en la columna o columnas de clave única.

Restricción PRIMARY KEY

- Una restricción PRIMARY KEY crea una clave primaria para la tabla.
- Solo se puede crear una clave primaria para cada tabla.
- La restricción PRIMARY KEY es una columna o un juego de columnas que identifica de forma única cada fila de una tabla.
- Ninguna columna que forme parte de la clave primaria puede contener un valor nulo.
- Se debe crear una clave primaria compuesta en el nivel de la tabla.

Por ejemplo:

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

Nota: Puesto que la unicidad forma parte de la definición de restricción de clave primaria, el servidor de Oracle aplica la unicidad mediante la creación implícita de un índice único en la columna o columnas de clave primaria.

Restricción PRIMARY KEY

DEPARTMENTS 

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400
50	Shipping	121	1500
60	IT	103	1400
70	Public Relations	204	2700
80	Sales	145	2500

No permitido
(valor nulo) 

INSERTAR EN 

(null)	Public Accounting	124	2500
50	Finance	124	1500

 No permitido (50 ya existe)

Nota: Consulte en la diapositiva 27 ejemplos de codificación de restricción de clave primaria.



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

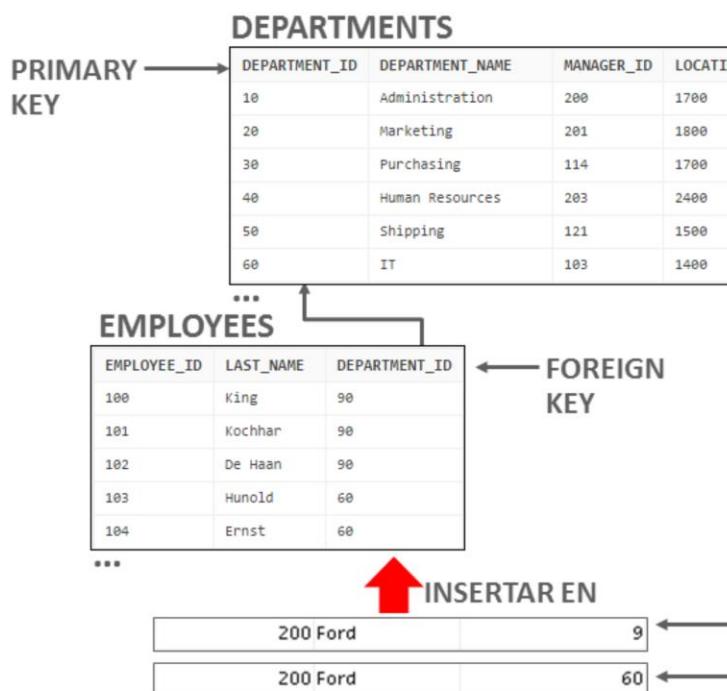
Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

34

Restricción FOREIGN KEY

- La restricción FOREIGN KEY (o integridad referencial) designa una columna o una combinación de columnas como clave ajena.
- Establece una relación con una clave primaria en la misma tabla o en una diferente.
- A continuación, se muestran las directrices para las restricciones de clave ajena:
 - El valor de clave ajena debe coincidir con un valor existente de la tabla principal o ser un valor NULL.
 - Las claves ajenas se basan en los valores de datos y son punteros puramente lógicos, en lugar de físicos.

Restricción FOREIGN KEY



Restricción FOREIGN KEY

- Definida en el nivel de tabla:

```
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));
```

Las claves ajenas compuestas se deben crear mediante la definición de nivel de tabla.

En la diapositiva, el ejemplo define una restricción FOREIGN KEY en la columna DEPARTMENT_ID de la tabla EMPLOYEES, mediante la sintaxis de nivel de tabla. El nombre de la restricción es EMP_DEPT_FK.

La clave ajena se puede definir también en el nivel de columna, siempre que la restricción esté basada en una sola columna. La sintaxis difiere en que las palabras clave FOREIGN KEY no aparecen. Por ejemplo:

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

Restricción FOREIGN KEY

- Definida en el nivel de columna:

```
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));
```

Se debe crear una clave ajena compuesta en el nivel de la tabla, por ejemplo:

```
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)
);
```

Restricción FOREIGN KEY: Palabras clave

- FOREIGN KEY: Define la columna en la tabla secundaria en el nivel de restricción de tabla.
- REFERENCES: Identifica la tabla y la columna en la tabla principal.
- ON DELETE CASCADE: Suprime las filas dependientes de la tabla secundaria cuando se suprime una fila de la tabla principal.
- ON DELETE SET NULL: Convierte los valores de clave ajena dependiente en nulos.



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 39

Sin las opciones ON DELETE CASCADE u ON DELETE SET NULL, la fila de la tabla principal no se puede suprimir si se hace referencia a ella en la tabla secundaria. Y estas palabras clave no se pueden utilizar en la sintaxis en el nivel de columna.

Restricción CHECK

- Define una condición que debe cumplir cada fila.
- No puede hacer referencia a columnas de otras tablas.

```
CREATE TABLE employees
(
    salary NUMBER(8,2) CONSTRAINT emp_salary_min
                  CHECK (salary > 0),
    ...
)
(CHECK constraint shown here at column level)
```

Para cumplir la restricción, en cada fila de la tabla se debe definir la condición como TRUE o desconocida (debido a un valor nulo).

Una sola columna puede tener varias restricciones CHECK que hagan referencia a la columna en su definición. No hay ningún límite en cuanto al número de restricciones CHECK que puede definir en una columna.

Las restricciones CHECK se pueden definir en el nivel de tabla o de columna.

CREATE TABLE: Ejemplo de Restricción 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'));
```

(CHECK constraint shown here at table level)

Escenario de caso: Creación de tablas



ORACLE

ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 42

Escenario de caso: Adición de restricciones

```
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)
) ;
```



ACADEMY

DFo 6-3

Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

43

Escenario de caso: Adición de restricciones

```
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)
) ;

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)
) ;
```



DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 44

Lenguaje de definición de datos

- La creación de tablas forma parte del lenguaje de definición de datos de SQL
- Entre otras sentencias DDL se incluyen:
 - ALTER: Para modificar la estructura de un objeto
 - DROP: Para eliminar un objeto de la base de datos
 - RENAME: Para cambiar el nombre de un objeto de base de datos



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 45

Sentencia ALTER TABLE

Utilice la sentencia ALTER TABLE para cambiar la estructura de tabla:

- Agregar una columna
- Modificar una definición de columna existente
- Definir un valor por defecto para la nueva columna
- Borrar una columna
- Cambiar el nombre de una columna
- Cambiar una tabla al estado de solo lectura

Después de crear una tabla, puede que necesite cambiar la estructura de la tabla por cualquiera de las siguientes razones:

- Ha omitido una columna.
- Debe cambiar la definición de columna o su nombre.
- Debe eliminar columnas.
- Desea definir la tabla en modo de solo lectura.

Sentencia ALTER TABLE

- Utilizar la sentencia ALTER TABLE para agregar, modificar y borrar columnas:

```
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] ...);
```

En la sintaxis:

- *table* es el nombre de la tabla.
- ADD | MODIFY | DROP es el tipo de modificación.
- *column* es el nombre de la columna.
- *data type* es el tipo de dato y la longitud de la columna.
- DEFAULT *expr* especifica el valor por defecto de una columna.

Adición de columnas

- Puede utilizar la cláusula ADD para agregar columnas:

```
ALTER TABLE dept  
ADD job_id VARCHAR2(9);
```

- La nueva columna se convierte en la última:

EMPLOYEE_ID	LAST_NAME	HIRE_DATE	JOB_ID
100	King	17-JUN-03	-
101	Kochhar	21-SEP-05	-
102	De Haan	13-JAN-01	-
103	Hunold	03-JAN-06	-

Nota: Si una tabla ya contiene filas cuando se agrega una columna, la nueva columna será inicialmente nula o utilizará el valor por defecto para todas las filas. Solo puede agregar una columna NOT NULL obligatoria a una tabla que contenga datos en las demás columnas si especifica un valor por defecto. Puede agregar una columna NOT NULL a una tabla vacía sin el valor por defecto.

Consulte la diapositiva 9 para consultar el código para crear esta tabla.

Modificación de columnas

- Puede cambiar el tipo de dato, tamaño y valor por defecto de una columna:

```
ALTER TABLE dept  
MODIFY dname VARCHAR2(30) ;
```

- El cambio de un valor por defecto solo afecta a las inserciones posteriores en la tabla.
- Las modificaciones están sujetas a determinadas condiciones.

A continuación, se muestran las directrices para modificar una columna:

- Puede aumentar el ancho o la precisión de una columna numérica.
- Puede aumentar el ancho de las columnas de caracteres.
- Puede reducir el ancho de una columna si:
 - La columna solo contiene valores nulos.
 - La tabla no tiene filas.
 - La disminución del ancho de columna no es inferior a los valores existentes en dicha columna.
- Puede cambiar el tipo de dato si la columna solo contiene valores nulos. La única excepción son las conversiones de CHAR a VARCHAR2, que se pueden realizar con los datos de las columnas.
- Solo puede convertir una columna CHAR al tipo de dato VARCHAR2 o una columna VARCHAR2 al tipo de dato CHAR si la columna contiene valores nulos o si no cambia el tamaño.
- El cambio a un valor por defecto de una columna solo afecta a las inserciones posteriores en la tabla.
- Puede agregar una restricción NOT NULL mediante las cláusulas MODIFY.

Borrado de columnas

- Usar la cláusula `DROP COLUMN` para borrar columnas que ya no son necesarias:

```
ALTER TABLE dept  
DROP (job_id);
```

Table altered.

EMPLOYEE_ID	LAST_NAME	annsal	HIRE_DATE
100	King	24000	17-JUN-03
101	Kochhar	17000	21-SEP-05
102	De Haan	17000	13-JAN-01
103	Hunold	9000	03-JAN-06
104	Ernst	6000	21-MAY-07
105	Austin	4800	25-JUN-05
106	Pataballa	4800	05-FEB-06



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 50

A continuación, se muestran las directrices para borrar una columna:

- La columna puede o no contener datos.
- Con la sentencia `ALTER TABLE DROP COLUMN`, solo se puede borrar una columna en cada ocasión.
- La tabla debe tener al menos una columna después de modificarla.
- Después de borrar una columna, no se puede recuperar.
- Una clave primaria a la que hace referencia otra columna no se puede borrar, a menos que se agregue la opción de cascada.
- El borrado de una columna puede tardar un rato si tiene muchos valores. En este caso, puede ser mejor definirla para que no se utilice y borrarla cuando haya menos usuarios en el sistema. De esta forma, se evitan los bloqueos ampliados.

Opción SET UNUSED

- La opción SET UNUSED marca una o más columnas como no utilizadas para que se puedan borrar simultáneamente cuando la demanda de recursos del sistema sea menor.
- Puede utilizar la opción SET UNUSED para marcar una o más columnas como no utilizadas.
- Puede utilizar la opción DROP UNUSED COLUMNS para eliminar las columnas marcadas como no utilizadas.



ACADEMY

DFO 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 51

Las columnas no utilizadas se tratan como si se hubieran borrado, aunque sus datos de columna permanezcan en las filas de la tabla.

Después de que una columna se marque como no utilizada, no tendrá acceso a dicha columna. Las consultas SELECT * no recuperan datos de las columnas marcadas como no utilizadas. Además, los nombres y tipos de columnas marcados como no utilizados no se muestran durante la sentencia DESCRIBE, y puede agregar a la tabla una nueva columna con el mismo nombre que la columna no utilizada.

Puede especificar la palabra clave ONLINE para indicar que se permiten las operaciones de lenguaje de manipulación de datos (DML) en la tabla al marcar la columna o columnas como UNUSED. El siguiente ejemplo de código muestra el uso de SET UNUSED COLUMN, que define una columna como no utilizada para siempre mediante la adición de la palabra clave ONLINE:

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

La información de SET UNUSED se almacena en la vista de diccionario USER_UNUSED_COL_TABS.

Nota: Las instrucciones para definir una columna como UNUSED son similares a las instrucciones para borrar una columna.

Opción 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 <table_name>
DROP UNUSED COLUMNS;
```

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

ALTER TABLE dept
DROP UNUSED COLUMNS;
```



Al definir una columna como UNUSED, tiene la opción de borrar esa columna.

Puede utilizar DROP UNUSED COLUMNS para eliminar de la tabla todas las columnas que estén marcadas actualmente como no utilizadas. Puede utilizar esta sentencia cuando desee reclamar el espacio en disco adicional de las columnas no utilizadas en la tabla. Si la tabla no contiene columnas no utilizadas, la sentencia no devuelve ningún error.

Nota: Una opción DROP UNUSED COLUMNS posterior elimina físicamente todas las columnas no utilizadas de una tabla, de forma similar a DROP COLUMN.

Escenario de caso: Modificación de tablas



Profesor

Sean, estaba examinando la tabla AUTHORS y me he dado cuenta de que:

- Falta el campo de dirección de correo electrónico del autor.
- Hay que aumentar longitud de la columna de nombre del autor.

¿Puede realizar estos cambios?

Claro, puedo hacerlo. La modificación consiste en agregar una nueva columna y aumentar la longitud de columna, así que no debe haber ningún problema.



Alumno



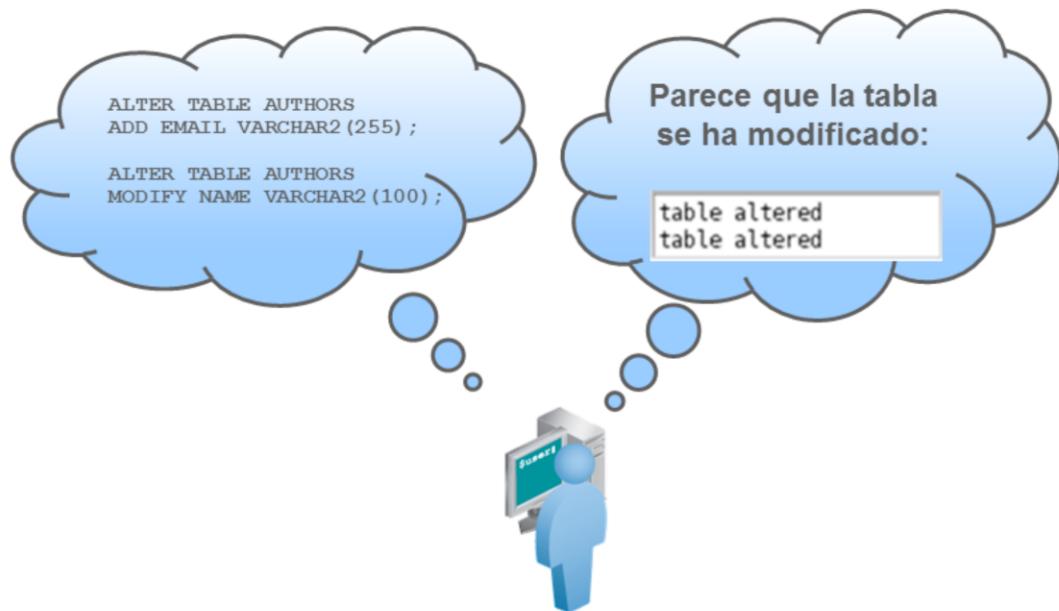
ACADEMY

DFo 6-3

Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 53

Escenario de caso: Modificación de tablas



Tablas de solo lectura

Puede utilizar la sintaxis de ALTER TABLE para:

- Definir una tabla en modo de solo lectura para evitar cambios de DDL o DML durante el mantenimiento de la tabla
- Volver a definir la tabla en modo de lectura/escritura

```
ALTER TABLE dept READ ONLY;  
-- perform table maintenance and then  
-- return table back to read/write mode  
  
ALTER TABLE dept READ WRITE;
```

A continuación, se muestran las directrices para definir una tabla en modo de solo lectura:

- Puede especificar READ ONLY para definir una tabla en modo de solo lectura.
- Cuando una tabla está en modo de solo lectura, no se pueden emitir sentencias DML que afecten a la tabla o cualquier sentencia SELECT... FOR UPDATE.
- Puede emitir sentencias DDL siempre y cuando no modifique los datos de la tabla.
- Se permiten operaciones sobre los índices asociados a la tabla cuando la tabla está en modo de solo lectura.
- Especifique READ/WRITE para volver a definir una tabla de solo lectura en modo de lectura/escritura.

Nota: Si es necesario, puede borrar una tabla en modo READ ONLY. El comando DROP se ejecuta solo en el diccionario de datos, por lo que no es necesario el acceso al contenido de la tabla. El espacio utilizado por la tabla no se reclamará hasta que el tablespace se vuelva a definir en modo de lectura/escritura y, a continuación, se podrán realizar los cambios necesarios en las cabeceras de segmentos de bloque, etc.

Borrado de una tabla

- Mueve una tabla a la papelera de reciclaje.
- Elimina la tabla y sus datos si se especifica la cláusula PURGE.
- Invalida los objetos dependientes y elimina privilegios de objeto en la tabla.

```
DROP TABLE dept;
```

```
Table dropped.
```



A menos que especifique la cláusula PURGE, la sentencia DROP TABLE no vuelve a liberar espacio en los tablespaces para que lo utilicen otros objetos, y el espacio sigue contando en la cuota de espacio del usuario. El borrado de una tabla invalida objetos dependientes y elimina privilegios de objeto en la tabla.

Al borrar una tabla, la base de datos pierde todos los datos de la tabla y todos los índices asociados a esta.

Sintaxis

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

En la sintaxis, *table* es el nombre de la tabla.

A continuación, se muestran las directrices para borrar una tabla:

- Se suprimen todos los datos de la tabla.
- Se mantienen las vistas y los sinónimos, pero no son válidos.
- Se confirman las transacciones pendientes.
- Solo el creador de la tabla o un usuario con el privilegio DROP ANY TABLE puede eliminar una tabla.

Ejercicio del proyecto

DFo_6_3_Project

Base de datos de la tienda Oracle Baseball League:

Uso de DDL para crear y mantener tablas de base de datos



ACADEMY

DFo 6-3
Lenguaje de definición de datos (DDL)

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados. 57

Resumen

En esta lección, debe haber aprendido a hacer lo siguiente:

- Identificar los pasos necesarios para crear tablas de base de datos
- Describir el objetivo del DDL
- Mostrar las operaciones DDL necesarias para crear y mantener las tablas de una base de datos



