TEMA 7 PARTE II.
CREACIÓN, MODIFICACIÓN Y ELIMINACIÓN DE OBJETOS CON SQL

7.13. MODIFICACIÓN Y ELIMINACIÓN DE OBJETOS EN UN RDBMS.

A nivel general, se emplean 2 sentencias SQL para realizar la modificación y eliminación de objetos en un RDBMS:

- `alter`: Empleada para modificar la estructura física de un objeto en un RDMBS: tablas, campos, vistas, secuencias, etc.
- `drop`: Se emplea para eliminar un objeto de la base de datos: tablas, campos, vistas, secuencias, etc.

7.13.1. Modificando la estructura de una tabla.

Para realizar la modificación de la estructura física de una tabla se emplea la sentencia `alter table`. Para eliminar una tabla se emplea la sentencia `drop table`.

Sintaxis SQL estándar:

```
alter table<table_name>
{
   [add [column] <column_definition>]   |
   [alter [column] <column_name>
      {
        set default<default> |drop default   |add<scope_clause> |
        drop scope<drop_behavior> |<alter_identity_column_option>
      }
   ] |

   [drop [column] <column_name>restrict   | cascade] |
   [add<table_constraint>] |
   [drop constraint<constraint_name>restrict | cascade]
} ;
```

El código anterior define de forma básica la sintaxis para agregar o eliminar columnas, valores por default, eliminar constraints.

Sintaxis Oracle:

La sintaxis en Oracle, contiene más opciones que la definición del SQL estándar. La sintaxis para los elementos principales es la siguiente:

```
alter table [<qualifier>.]<table_name> {
  [<change_physical_attributes>] |
  [add<column_name><datatype> [<sizel>[,<size2>]]
    [default<default_value>]
    [<column_constraint> ... ]
  ]... |

  [modify<column_name><datatype> [<sizel>[,<size2>]]
    [default<default_value>] [null | notnull]

  ]... |

  [drop {column<column_name> | (<column_name>…)}
    [cascade constraints]
  ] |

  [modify constraint<constraint_name><constraint_state>] |

  [add {
        primary key<column_name>...) |
        unique (<column_name>... ) |
        constraint<constraint_name><constraint_definition>
      }
  ] |
[drop {primary key | unique (<column_name>... ) |
       constraint<constraint_name> [cascade]
}
  ] |

  [disable | enable
    {primary key | unique (<column_name>... ) |constraint<constraint_name>
  |
    all triggers |table lock
    }
  ] |
  [rename to<new_table_name>] | [move [online] tablespace<tablespace_name>]
}
```

En resumen:

Considerar las siguientes tablas:

```
create table puesto (
      puesto_id number(3,0) constraint puesto_pk primary key,
      clave varchar2(10),
      nombre varchar2(50)
);

create table empleado (
      puesto_id number(10,0) constraint puesto_pk references
puesto(puesto_id),
      fecha_inicial date,
      rfc varchar2(13)
);
```

Ejemplo: Agregar columnas:
```
alter table empleado add (comision number(4,2));
```

Ejemplo: Modificar columnas:
```
alter table empleado modify (comision number(10,2) default 0.05);
```

Ejemplo: Eliminar columnas.
```
alter table empleado drop column comision;
```

Ejemplo: Marcar una columna como "unused"
```
alter table puesto set unused column clave;
```
Tiempo después:
```
alter table puesto drop unused columns;
```

Ejemplo: Renombrar columnas
```
alter table empleado rename column fecha_inicial to fecha_contrato;
```

Ejemplo: Marcar una tabla como tabla de solo lectura
```
alter table puesto read only;
```

Ejemplo: Agregando constraints
```
alter table empleado add constraint rfc_uk unique(rfc);
```

Ejemplo: Deshabilitar constraints
```
alter table empleado disable constraint rfc_uk;
```

Ejemplo: Habilitar constraints
```
alter table empleado enable constraint rfc_uk;
```

Ejemplo: Eliminando constraints
```
alter table empleado drop constraint rfc_uk;
```

Ejemplo: Renombrar una tabla
```
alter table empleado rename to trabajador;
```

Ejemplo: Renombrar una tabla
```
alter table empleado rename to trabajador;
```

Ejemplo: Eliminar una tabla
```
drop table puesto; -- ¿por qué Oracle marca error?: Por existir dependencias.
```

Ejemplo: Eliminar una tabla y sus referencias.
```
drop table puesto cascade constraints;
```

Ejemplo: Eliminar una tabla sin posibilidad de recuperar posteriormente.
```
drop table trabajador purge;
```

Las siguientes secciones muestran a detalle las principales opciones del SQL para modificar o eliminar la estructura de una tabla empleando Oracle.

7.13.2. Agregando columnas a una tabla.

Para agregar una columna a una tabla existente, se emplea la sentencia `ADD`.

Ejemplo:

```
alter table direccion
      add codigo_postal varchar2(5)
      add numero_interior varchar2(4);
```

Observar que se pueden agregar varias columnas en una sola instrucción de `ALTER TABLE` (no se separan por comas).

7.13.3. Modificando columnas.

Es posible modificar algunos aspectos de la definición original de una columna: el tipo de dato, incrementar o decrementar la capacidad del tipo de dato (número de caracteres, precisión o escala), especificar nuevos valores por default, o agregar el constraint `not null` a una columna.

Limitaciones:

- Para decrementar el tamaño o capacidad de un tipo de dato, la tabla debe estar vacía, la columna no debe contener valores asignados (solo nulos), o los valores existentes no deben sobrepasar el valor de la nueva capacidad.
- No es posible agregar el constraint `not null` a un campo que contiene valores nulos.
- Para cambiar el tipo de dato de una columna, si esta contiene datos, solamente se pueden hacer conversiones entre tipos de datos compatibles, por ejemplo, `char` y `varchar`. Si la columna está vacía, no hay restricciones.

Para realizar la modificación de una columna se emplea la sentencia MODIFY:

```
  modify<column_name><data type> [<sizel>[,<size2>]]
    [default<default_value>] [null | notnull]
```

Ejemplo:
Considerar la definición de la siguiente tabla:

```
create table libro(
      libro_id number(10,0) constraint pk_libro primary key,
      nombre char(50) not null,
      tipo_libro char(1),
      precio number(5,2) notnull
);
```

Los datos que contiene son los siguientes:

```
LIBRO_ID NOMBRE                                             TIPO_LIBRO PRECIO
======== ================================================== ========== ======
1        FABULAS                                            I          120.13
2        MATEMATICAS A                                      {null}     500.13
3        HISTORIA UNIVERSAL                                 C          500.14
```

- Modificar el campo `nombre`, de `char` a `varchar2`.

```
alter table libro modify nombre varchar2(100);
```

Observar que no es requerido redefinir todas las características de un campo, por ejemplo, en la definición original, el campo `nombre` está marcado como `not null`. Este constraint se sigue manteniendo posterior a la ejecución de la sentencia anterior.

- Reducir la capacidad del campo NOMBRE a un `varchar2(5);` ¿Qué pasa con esta instrucción?

```
alter table libro modify nombre varchar2(5);
```

```
ORA-01441: cannot decrease column length because some value is too big
```

- ¿Qué pasa si se reduce a `varchar2(100)`?. En Oracle, es posible siempre y cuando ninguno de los valores sobrepase al nuevo valor.

- Cambiar el campo `tipo_libro` para que asigne el valor 'L' por default. (Notar que esta instrucción no actualiza posibles valores nulos existentes):

```
alter table libro modify tipo_libro char(1) default 'l';
insert into libro (libro_id,nombre,precio) values(4,'matematicas b',600.93);
select * from libro;
```

LIBRO_ID	NOMBRE	TIPO_LIBRO	PRECIO
1	FABULAS	I	120.13
2	**MATEMATICAS A**	**{null}**	**500.13**
3	HISTORIA UNIVERSAL	C	500.14
4	**MATEMATICAS B**	**L**	**600.93**

Observar que el valor por default agregado anteriormente, se aplica a partir de la siguiente instrucción INSERT. Los registros existentes no se modifican.

- Modificar el campo `precio` de `number` a `float`.

```
alter table libro modify precio float;
```

- Regresar el campo `precio` de `float` a `number(5,2)`, observar el error.

```
alter table libro modify precio number(5,2);
```

```
ORA-01440: column to be modified must be empty to decrease precision or scale
alter table libro modify <error>precio number(5,2)
```

- Modificar el nombre del campo `precio` a `costo`:

```
alter table libro rename column precio to costo;
```

7.13.4. Eliminando columnas.

para eliminar columnas se emplea la sentencia `drop` en conjunto con `alter table`.
Ejemplo:
Eliminar el campo `tipo_libro` de la tabla `libro`:

```
alter table libro drop(tipo_libro);
```

- Es posible eliminar más de una columna dentro de la misma instrucción, se especifica la lista de columnas dentro de los paréntesis separadas por comas.
- Al eliminar una columna, los índices y constraints asociados a la columna también se eliminan.
- Si la columna es referenciada por otra tabla como FK, en Oracle, primero se deberán eliminar los constraints de referencia, de otra forma el manejador generará un error, a excepción de que se indique el borrado en cascada de los constraints que hacen referencia al campo.

Ejemplo:

Eliminar el campo `libro_id` de la tabla `libro`, la cual representa a la PK. Considere que ninguna tabla hace referencia a dicho campo.

Constraints de la tabla `libro` antes de eliminar:

```
Constraint Name Column Name   Search Condition     Status  Type Delete Rule Generated      Condition
=============== =========== ==================== ======= ==== =========== ============== =============
PK_LIBRO        LIBRO_ID    {null}               ENABLED P    {null}      USER NAME      PK (LIBRO_ID)
SYS_C0011520    NOMBRE      "NOMBRE" IS NOT NULL ENABLED C    {null}      GENERATED NAME {null}
SYS_C0011521    PRECIO      "PRECIO" IS NOT NULL ENABLED C    {null}      GENERATED NAME {null}
```

Índices de la tabla antes de eliminar:

```
Index Name Column Name Unique Type   Column Expression
========== =========== ====== ====== =================
PK_LIBRO   LIBRO_ID    UNIQUE NORMAL {null}
```

Eliminando la PK:

```
alter table libro drop column libro_id;
```

Constraints de la tabla `libro` después de eliminar:

```
ConstraintName  ColumnName   Search Condition     Status  Type Delete Rule Generated      Condition
=============== ========== ==================== ======= ==== =========== ============== =============
SYS_C0011520    NOMBRE     "NOMBRE" IS NOT NULL ENABLED C    {null}      GENERATED NAME {null}
SYS_C0011521    PRECIO     "PRECIO" IS NOT NULL ENABLED C    {null}      GENERATED NAME {null}
```

Índices de la tabla antes de eliminar:

```
Index Name Column Name Unique Type   Column Expression
========== =========== ====== ====== =================
```

Ejemplo:

Considerar nuevamente la tabla `LIBRO` original, y la definición de la siguiente tabla.

```
create table orden(
      orden_id number(10,0) constraint orden_pk primary key
);


create table orden_libro(
      orden_libro_id number(10,0) constraint orden_libro_pk primary key,
      libro_id constraint ol_libro_fk references libro(libro_id) not null,
      orden_id constraint ol_orden_fk references orden(orden_id) not null
);

insert into orden values(1);
insert into orden_libro values (1,1,1);
```

Al intentar eliminar el campo `libro_id` se genera el siguiente error:

```
ORA-12992: cannot drop parent key column
ALTER TABLE LIBRO DROP COLUMN <ERROR>LIBRO_ID
```

Existe una alternativa que permite eliminar un campo que está siendo referenciando por otras tablas. Empleando la sentencia `cascade constraints`, provoca la eliminación en cascada de todos los constraints creados en las tablas que hacen referencia al campo a eliminar.

Ejemplo:
Eliminar el campo `libro_id` que está siendo referenciado por la tabla `orden_libro`

```
alter table libro drop column libro_id cascade constraints;
```

Al ejecutar esta instrucción, desaparece el constraint de referencia que se había creado en `orden_libro`:

```
Constraint Name Column Name     Search Condition      Status  Type Delete Rule
=============== ============== ====================== ======= ==== ===========
OL_ORDEN_FK     ORDEN_ID       {null}                 ENABLED R    NO ACTION
ORDEN_LIBRO_PK  ORDEN_LIBRO_ID {null}                 ENABLED P    {null}
SYS_C0011527    LIBRO_ID       "LIBRO_ID" IS NOT NULL ENABLED C    {null}
SYS_C0011528    ORDEN_ID       "ORDEN_ID" IS NOT NULL ENABLED C    {null}
```

7.13.5. Renombrando tablas.

Para realizar el renombrado de tablas se emplea la sentencia `RENAME TO`.

Ejemplo:
```
alter table libro rename to book;
```

7.13.6. Agregando constraints.

Los siguientes ejemplos muestran la sintaxis en Oracle para agregar nuevos constraints a campos existentes.

- Agregando constraint PK a una tabla
  ```
  alter table libro add constraint libro_pk primary key(libro_id);
  ```

- Suponer que el campo `CLAVE_LIBRO` de la tabla libro debe ser único, agregar un constraint a dicho campo:
  ```
  alter table libro add constraint libro_unique unique(clave_libro);
  ```

- Agregar un nuevo constraint a la tabla `ORDEN_LIBRO` en `LIBRO_ID`, suponer que el constraint de referencia (FK) no fue especificado.
  ```
  alter table orden_libro add constraint orden_libro_fk1 foreign key
  (libro_id) references libro(libro_id);
  ```

7.13.7. Eliminando constraints.

Para eliminar un constraint se emplea la sentencia `DROP CONSTRAINT` en conjunto con `ALTER TABLE` empleando el nombre del constraint.

Ejemplo:
Eliminar el constraint de referencia en el campo `LIBRO_ID`, tabla `ORDEN_LIBRO`.

```
alter table libro drop constraint orden_libro_fk1;
```

7.13.8. Habilitando y deshabilitando constraints.

En algunas ocasiones es conveniente conservar los constraints, pero por alguna situación es necesario deshabilitar la restricción, por ejemplo, durante un proceso de carga o migración de datos, en especial cuando existe una gran cantidad de datos a procesar. En estas situaciones, normalmente se deshabilitan constraints, se procesa la información, y al concluir se reactivan. En Oracle se emplea la siguiente sintaxis para habilitar o deshabilitar un constraint.

```
{disable | enable} constraint <constraint_name>
```

Ejemplo:
Deshabilitar el constraint de referencia del campo `ORDEN_ID` en la tabla `ORDEN_LIBRO`:

```
alter table orden_libro disable constraint ol_orden_fk;
```

Objetos de un RDBMS que pueden ser habilitados o inhabilitados en Oracle:
- Procedimientos almacenados.
- Triggers
- Funciones definidas por el usuario.

7.13.9. Modificando otros objetos en Oracle.

- Modificando índices: `alter index`

```
alter index <index_name>
{[rename to<new_name>]|[rebuild tablespace<tablespace_name>]
```

Existen otras opciones, las anteriores representan las principales opciones. `rebuild` se emplea para cambiar el `tablespace` de almacenamiento del índice.

- Modificando vistas: `alter view`

```
alter view [<qualifier>.]<view_name>
{     [compile] | [add constraint<view_constraint_definition>] |
  [modify constraint<constraint_name>] |
  [drop {constraint<constraint_name> | primary key}]
};
```

`compile` se emplea para realizar una re-compilación manual de la vista cuando alguna de las tablas a las que hace referencia ha cambiado en su estructura. Actualmente está re-compilación se realiza automáticamente por el RDBMS.
Ejemplo:

```
alter view v_empleado compile;
```

Ejemplo:
Agregando un constraint tipo `unique` al campo `clave_empleado`:

```
alter view v_empleado
add constraint clave_emp_unique unique(clave_empleado);
```

- Modificando secuencias

```
alter sequence [<qualifier>.]<sequence_name>
[increment by<increment_value>]
[maxvalue <max_value> | nomaxvalue]
[minvalue <min_value> | nominvalue]
[cycle | nocycle]
[cache <value> | nocache]
[order | noorder]
```

Ejemplo:
Modificando la secuencia `empleado_seq`:

```
alter sequence empleado_seq increment by 1 nomaxvalue nocycle;
```

7.13.10. Eliminando tablas.

La sentencia `drop table` se emplea para realizar la eliminación de una tabla. Al invocar `drop table`, se eliminan también sus constraints, y sus índices. Existe un mecanismo de respaldo que permite recuperar los datos y la definición de una tabla a pesar de emplear la sentencia `drop`. En realidad, al invocar `drop` en Oracle 11g+, no se libera el espacio, se emplea una "papelera de reciclaje" y los datos pueden ser recuperados empleando la herramienta e instrucción `flashback`.

Sintaxis SQL estándar para eliminar una tabla:

```
drop table<table_name> [{cascade | restrict}]
```

Sintaxis Oracle:

```
drop table [<qualifier>.]<table_name> [cascade constraints][purge]
```

`cascade constraints` tiene el mismo efecto visto anteriormente asociado con la eliminación de un campo que es referenciado por otros. En este caso, `cascade constraints` elimina todos los constraints que hacen referencia a la tabla que se va a eliminar.

Si no se especifica `cascade constraints`, y si la tabla tiene referencias, Oracle genera un error.

`purge` elimina la tabla y libera espacio en el tablespace en un solo paso sin agregar los datos de la tabla a la papelera de reciclaje (recyclebin) por lo que ya no será posible recuperar los datos con la instrucción `flashback`.

De cualquier forma, se debe tener extremo cuidado al ejecutar instrucciones con DROP.

7.13.11. Eliminando otros objetos

- Eliminando índices

```
drop index [<qualifier>.]<index_name>;
```

- Eliminando vistas

```
drop view<view_name> [cascade constraints];
```

- Eliminando secuencias

```
drop sequence <sequence_name>[cascade | restrict]
```