

PRÁCTICA: 2. RESTRICCIONES**OBJETIVOS:**

Aprender a implementar reglas de negocio en esquemas ORACLE mediante la inclusión de restricciones de integridad.

MATERIAL:

ORACLE 9i versión para WINDOWS XP

BIBLIOGRAFIA:

ORACLE: Introduction to ORACLE 9i:SQL. Student Guide. Volume 1. 2001.
Chapter 10.

ORACLE 9I: GUIA DE APRENDIZAJE
ABBEY, MICHAEL Y COREY, MIKE Y ABRAMSON, IAN
MCGRAW-HILL / INTERAMERICANA DE ESPAÑA, S.A.

CONTENIDO:

Introducción
Definición de Restricciones al Crear una Tabla
Restricción NOT NULL
Restricción UNIQUE
Restricción PRIMERY KEY
Restricción CHECK
Añadiendo Restricciones en Tablas Existentes
Eliminando Restricciones en Tablas Existentes
Desactivando restricciones
Activando restricciones
Restricciones en Cascada

Después de completar esta práctica, el alumno debería ser capaz de:

- Describir restricciones,
- Crear y mantener restricciones.

Esquema de Trabajo

Los ejemplos mostrados en esta práctica están basados en el siguiente esquema relacional:

Empleados (id, apellidos, nombre, salario, dep, fecha_alta, supervisor_id, email)

Departamentos (num, nombre, director_id, edificio)

Empleados.supervisor_id -> Empleados

Empleados.dep -> Departamentos

Departamentos.director_id -> Empleados

1. Introducción

Las reglas de negocio se implementan en ORACLE mediante restricciones (constraints), disparadores (triggers) o código de aplicación.

El ORACLE Server se basa en las restricciones para prevenir la entrada de datos no válidos en las tablas.

En suma, al trabajar con ORACLE, las restricciones se pueden emplear para:

- Forzar el cumplimiento de reglas por los datos de una tabla cuando se inserta una fila, se elimina o se modifica. La restricción debe ser satisfecha para que la operación DML se concluya con éxito.
- Prevenir el borrado de una tabla cuando existen dependencias desde otras tablas.
- Proveer reglas para otras herramientas (ORACLE DEVELOPER u otros CASE).

Tipos de restricciones

En ORACLE 9i son los siguientes:

NOT NULL	Una columna no puede contener un valor nulo
UNIQUE	Una columna o combinación de columnas cuyos valores deben ser únicos para todas las filas de la tabla
PRIMARY KEY	Identifica unívocamente cada fila de la tabla
FOREIGN KEY	Establece y obliga a que se cumpla una restricción de integridad entre una columna y otra columna de la tabla referenciada
CHECK	Especifica una condición que debe ser cierta

Guías para el uso

- Es conveniente asignarles un nombre autodefinido que permita referirlas de forma fácil

con posterioridad. Si el usuario no les asigna nombre el sistema las denomina "SYS_C<n>", donde <n> es un número entero diferente para cada restricción.

- Pueden ser definidas a la vez que se crea la tabla (dentro del CREATE TABLE) o posteriormente (ALTER TABLE).
- Se pueden definir a nivel de columna o a nivel de tabla.

Comprobar restricciones existentes

Con la sentencia DESCRIBE solo se muestran las restricciones NOT NULL. Para ver las demás es necesario consultar la vista USER_CONSTRAINTS del diccionario de datos. Ejemplos:

Obtener la lista de restricciones definidas en el diccionario de datos:

```
SELECT * FROM USER_CONSTRAINTS;
```

NOTA: En el listado C="CHECK", P="PRIMARY KEY", R="FOREIGN KEY". NOT NULL se representa como un caso especial de CHECK.

Igual, pero solo para la tabla Empleados:

```
SELECT * FROM USER_CONSTRAINTS
WHERE table_name='Empleados';
```

Comprobar las columnas asociadas

Las columnas asociadas con cada restricción se pueden consultar en la vista USER_CONS_COLUMNS del diccionario de datos. Ejemplo:

```
SELECT * FROM USER_CONS_COLUMNS
WHERE table_name='Empleados';
```

Otras vistas del diccionario de datos útiles para este tema son ALL_CONSTRAINTS y ALL_CONS_COLUMNS.

2. Definición de Restricciones al Crear una Tabla

Sintaxis:

```
CREATE TABLE [<esquema>.]<tabla>
(<columna> <tipo de dato> [DEFAULT <expresión>] [<restricción de columna>],
...
AS <subconsulta>
[<restricción de tabla>] [, ... ];
```

donde:

<esquema>	Nombre del propietario. Sólo es necesario indicarlo si no es el mismo que el del esquema.
<tabla>	Nombre de la tabla
<expresión>	Especifica un valor por defecto que sería usado si una operación INSERT no especifica valor
<columna>	Nombre de la columna
<tipo de dato>	Tipo de dato de la columna y su longitud
<restricción de columna>	Restricción de integridad que se define como parte de la definición de una columna particular
<restricción de tabla>	Restricción de integridad que se define fuera de las definición de las columnas, es decir, a nivel de tabla

Ejemplo:

```
CREATE TABLE Empleados (
    id          NUMBER(6),
    apellidos   VARCHAR2(40),
    nombre     VARCHAR2(20) NOT NULL,
    ...
    supervisor_id NUMBER(6),
    ...
    CONSTRAINT emp_pk PRIMARY KEY (id)
);
```

Nivel de Columna vs Nivel de Tabla

- *Columna*: Refieren a una única columna y se definen dentro de la especificación de la propia columna. Puede ser de cualquiera de los tipos de restricciones de integridad indicados en el apartado anterior.
- *Tabla*: Refieren a una o a varias columnas y se definen de forma separada a las definiciones de las columnas. Pueden ser de cualquier tipo de restricción salvo NOT NULL.

Sintaxis a nivel de columna:

[CONSTRAINT <nombre>] <tipo de restricción>

Sintaxis a nivel de tabla:

[CONSTRAINT <nombre>] <tipo de restricción> (<columna>[,...])

donde:

<nombre>	Nombre de la restricción.
<tipo de restricción>	Uno de los 5 tipos de restricciones de integridad permitidas. La sintaxis concreta se detalla en apartados siguientes.
<columna>	Nombre de columna

3. Restricción NOT NULL

También llamada de obligatoriedad.
Sólo se puede definir a nivel de columna, no de tabla.

Sintaxis:

```
[CONSTRAINT <nombre>] [NOT] NULL
```

donde:
<nombre> Nombre de la restricción

Ejemplo:

```
CREATE TABLE Empleados (  
    ...  
    nombre VARCHAR2(20) NOT NULL,  
    ...  
    Fecha_alta     DATE  
    CONSTRAINT fecha_obli NOT NULL,  
    ...  
);
```

4. Restricción UNIQUE

Es una restricción de unicidad.
Impide que pueden existir dos filas con el valor de la columna (unique key) o columnas (composite unique key).
Permite la entrada de valores nulos salvo que se establezca a la vez una restricción NOT NULL. Basta con que una de las columnas tome para el valor nulo para que se considere que se cumple la restricción de unicidad.
Las “composite unique key” sólo se pueden crear a nivel de tabla.
El ORACLE Server crea un índice de valores únicos como mecanismo para controlar este tipo de restricciones.

Sintaxis:

A nivel de columna:
[CONSTRAINT <nombre>] UNIQUE

A nivel de tabla:
[CONSTRAINT <nombre>] UNIQUE (<columna>[,...])

donde:

<nombre> Nombre de la restricción
<columna> Nombre de la columna

Ejemplo:

```
CREATE TABLE Empleados (
  ...
  apellidos        VARCHAR2(40) NOT NULL,
  nombre VARCHAR2(20),
  ...
  email            VARCHAR2(25) UNIQUE,
  ...
  CONSTRAINT apel_nom_unico UNIQUE (apellidos,nombre),
);
```

5. Restricción PRIMARY KEY

Es una restricción de clave primaria.

Sólo se puede definir una para cada tabla.

Esta restricción equivale a una restricción de unicidad (UNIQUE) y otra de obligatoriedad (NOT NULL) combinadas.

Igual que para UNIQUE, existen “primary key” y “composite primary key” (formadas por más de una columna). Éstas segundas se definen a nivel de tabla.

El ORACLE Server crea un índice de valores únicos como mecanismo para controlar la unicidad en este tipo de restricciones.

Sintaxis:

A nivel de columna:

```
[CONSTRAINT <nombre>] PRIMARY KEY
```

A nivel de tabla:

```
[CONSTRAINT <nombre>] PRIMARY KEY (<columna>[,...])
```

donde:

<nombre> Nombre de la restricción
<columna> Nombre de la columna

Ejemplo:

```
CREATE TABLE Departamentos (
  num                NUMBER(4),
  nombre VARCHAR2(30) NOT NULL,
  ...
```

```
CONSTRAINT dep_pk PRIMARY KEY (num)
);
```

6. Restricción FOREIGN KEY

Es una restricción de integridad referencial.

Designa a una o varias columnas como clave ajena y establece una relación de referencia con una clave primaria o clave única (UNIQUE) de otra tabla o de la misma.

El valor de la clave ajena debe coincidir con un valor existente en la tabla referenciada (parent table) o ser nulo.

Las claves ajenas son puramente lógicas (están basadas en valores de datos) y por tanto no son punteros físicos.

Las “composite foreign key” están formadas por más de una columna y deben ser definidas a nivel de tabla.

Sintaxis:

A nivel de columna:

```
[CONSTRAINT <nombre>]
REFERENCES [<esquema>].<tabla> [(<columna>[...])]
[ON DELETE {CASCADE | SET NULL}]
```

A nivel de tabla:

```
[CONSTRAINT <nombre>]
FOREIGN KEY (<columna>[...])
REFERENCES [<esquema>].<tabla> [(<columna>[...])]
[ON DELETE {CASCADE | SET NULL}]
```

donde:

<nombre>	Nombre de la restricción
<esquema>	Nombre del propietario. Sólo es necesario indicarlo si no es el mismo que el del esquema.
<tabla>	Nombre de la tabla padre (referenciada o destino)
<columna>	Nombre de columna

Ejemplo a nivel de columna:

```
CREATE TABLE Empleados (
    dep          NUMBER(4)
               CONSTRAINT emp_dep_fk
               REFERENCES Departamentos (num),
    ...
);
```

Ejemplo a nivel de tabla:

```
CREATE TABLE Empleados (  
    dep          NUMBER(4)  
    ...  
    CONSTRAINT emp_dep_fk  
        FOREIGN KEY (dep)  
        REFERENCES Departamentos (num)  
);
```

A nivel de columna no aparecen las palabras clave “FOREIGN KEY”. A nivel de tabla indican las columnas de la tabla origen (child table) que forman la clave ajena. Detrás de la palabra clave REFERENCES se indican la tabla y columnas de destino.

Modos de Borrado

Sólo existen tres modos de borrado, que indican lo que debe hacer ORACLE Server cuando se elimina una fila de la tabla padre:

- ON DELETE CASCADE: borra las filas dependientes de la tabla origen.
- ON DELETE SET NULL: pone las claves ajenas de las filas dependientes de la tabla origen a nulos.
- La opción por defecto, que se activa si no se incluye ON DELETE, es no permitir la acción de eliminar la fila de la tabla padre (equivale a un ON DELETE NO ACTION).

No existe la cláusula ON UPDATE (modos de modificación).

7. Restricción CHECK

Define una condición que deben cumplir todas las filas de la tabla.

La condición es igual que las condiciones de la cláusula WHERE del SELECT salvo porque no puede incluir:

- Referencias a pseudocolumnas (CURRVAL, NEXTVAL, LEVEL, ROWNUM).
- Llamadas a las funciones SYSDATE, UID, USER y USERENV.
- Consultas que refieren a otros valores en otras filas.
- Subconsultas (subqueries).

Una columna puede tener asociadas tantas restricciones CHECK como se desee.

Sintaxis:

```
[CONSTRAINT <nombre>] CHECK (<condición>)
```

donde:

<nombre> Nombre de la restricción

<condición> Una expresión que debe ser cierta para todas las filas de la tabla. A nivel de columna sólo puede referir a dicha columna. A nivel de tabla puede referir a otras columnas, pero a los valores de la misma fila.

Ejemplo a nivel de columna:

```
CREATE TABLE Empleados (
  ...
  salario      NUMBER(8,2)
              CONSTRAINT salario_positivo CHECK (salario>0),
  ...
);
```

Ejemplo a nivel de tabla:

```
CREATE TABLE Empleados (
  ...
  salario      NUMBER(8,2),
  neto         NUMBER(8,2),
  ...
  CONSTRAINT neto_max
              CHECK (neto<=salario*0'8)
);
```

8. Añadiendo Restricciones en Tablas Existentes

Se emplea la sentencia ALTER TABLE para:

- Añadir o eliminar una restricción, pero no se puede modificar su estructura o definición.
- Activar (enable) o desactivar (disable) restricciones.
- Añadir una restricción NOT NULL a una columna existente mediante la cláusula MODIFY.

Sintaxis:

```
ALTER TABLE <tabla>
  ADD [CONSTRAINT <nombre>] <tipo de restricción> (<columna>);
```

donde:

<tabla>	Nombre de la tabla
<nombre>	Nombre de la restricción
<tipo>	Tipo de la restricción
<columna>	Nombre de la columna

Ejemplo:

Añadir la restricción de clave ajena para el supervisor en la tabla de Empleados.

```
ALTER TABLE Empleados
  ADD CONSTRAINT emp_supervisor_fk
  FOREIGN KEY (supervisor_id)
  REFERENCES Empleados(id);
```

9. Eliminando Restricciones en Tablas Existentes

Se pueden consultar las vistas (views) `USER_CONSTRAINTS` y `USER_CONS_COLUMNS` del diccionario de datos para identificar el nombre de la restricción y las columnas que tienen asociadas.

Emplear la cláusula `DROP` de la sentencia `ALTER TABLE` para eliminar la restricción.

La opción `CASCADE` hace que se eliminen todas las restricciones dependientes de la elegida.

Sintaxis:

```
ALTER TABLE <tabla>
  DROP { PRIMARY KEY | UNIQUE (<columna>) | CONSTRAINT <nombre>
  [CASCADE];
```

donde:

<tabla>	Nombre de la tabla
<columna>	Nombre de la columna afectada por la restricción
<nombre>	Nombre de la restricción

Ejemplos:

Eliminar la restricción de clave ajena para el supervisor en la tabla de Empleados.

```
ALTER TABLE Empleados
  DROP CONSTRAINT emp_supervisor_fk;
```

Eliminar la restricción de clave primaria de la tabla Departamentos y la clave ajena asociada de empleados (Empleados.id-> Departamentos).

```
ALTER TABLE Departamentos
  DROP PRIMARY KEY CASCADE;
```

10. Desactivando Restricciones

Se puede desactivar la comprobación de una restricción, aunque siga estando en el esquema,

mediante la opción `DISABLE` de la sentencia `ALTER TABLE`.

También se puede incluir `DISABLE` durante la creación de la tabla (`CREATE TABLE`) para que desde el principio esté desactivada.

Desactivar una restricción `UNIQUE` o `PRIMARY KEY` supone remover el índice asociado.

Sintaxis:

```
ALTER TABLE <tabla>  
    DISABLE CONSTRAINT <nombre> [CASCADE];
```

donde:

<tabla>	Nombre de la tabla
<nombre>	Nombre de la restricción

Ejemplo:

Desactivar la restricción de clave primaria de la tabla de Empleados.

```
ALTER TABLE Empleados  
    DISABLE CONSTRAINT emp_pk CASCADE;
```

11. Activando Restricciones

Se puede activar la comprobación de una restricción mediante la opción `ENABLE` de la sentencia `ALTER TABLE`.

En el momento de la activación, todos los datos de la tabla deben satisfacer la restricción.

En el caso de restricciones `UNIQUE` o `PRIMARY KEY`, se crea automáticamente un índice asociado.

También se puede incluir `ENABLE` durante la creación de la tabla (`CREATE TABLE`).

Sintaxis:

```
ALTER TABLE <tabla>  
    ENABLE CONSTRAINT <nombre>;
```

donde:

<tabla>	Nombre de la tabla
<nombre>	Nombre de la restricción

Ejemplo:

Volver a activar la restricción de clave primaria de la tabla de Empleados.

```
ALTER TABLE Empleados
  ENABLE CONSTRAINT emp_pk;
```

12. Restricciones en Cascada

Un fenómeno que requiere especial atención es la eliminación de columnas de una tabla que son la base de restricciones que dependen unas de otras en cascada.

Para controlar este problema existe la cláusula “CASCADE CONSTRAINTS”, que se emplea junto con “DROP COLUMN”. Esta cláusula supone:

- Eliminar todas las restricciones de integridad referencial que refieren (apuntan) a claves primarias o únicas (UNIQUE) definidas sobre las columnas borradas.
- Eliminar las restricciones multicolumna (composite) definidas sobre las columnas borradas.

Ejemplo:

```
CREATE TABLE prueba1 (
  pk    NUMBER PRIMARY KEY,
  fk    NUMBER,
  col1  NUMBER,
  col2  NUMBER,
  CONSTRAINT fk_cons FOREIGN KEY (fk) REFERENCES prueba1,
  CONSTRAINT ck1 CHECK (pk>0 AND col1>0),
  CONSTRAINT ck2 CHECK (col2>0) );
```

Al ejecutar

```
ALTER TABLE prueba1 DROP (pk);
```

se produce un error porque pk es clave de la tabla padre de la integridad referencial.

Al ejecutar

```
ALTER TABLE prueba1 DROP (col1);
```

se produce un error porque col1 es empleada por la restricción multicolumna ck1.

Los errores se evitan incluyendo CASCADE CONSTRAINTS:

```
ALTER TABLE prueba1 DROP (pk) CASCADE CONSTRAINTS;
```

```
ALTER TABLE prueba1 DROP (col1) CASCADE CONSTRAINTS;
```

EJERCICIOS PRÁCTICA 7 - RESTRICCIONES

Para realizar estos ejercicios es necesario recordar la estructura del esquema de la base de datos ACADEM:

DEPARTAMENTOS (codigo, nombre)

AREAS (codigo, nombre, departamento)

PROFESORES (codigo, apellido1, apellido2, nombre_pila, activo, categoria, dedicacion, area)

ASIGNATURAS (siglas, nombre, creditos, curso, anualidad, clase, horas_teoría, horas_practica, grupos_teoría , grupos_practica, alumnos)

LOCALES (codigo, nombre, docente, capacidad, edificio, situacion)

GRUPOS (curso, clase, codigo, nombre)

DOCENCIA (id, curso, clase, grupo, siglas, profesor, local, dia, hora, periodicidad)

areas.departamento → departamentos

profesores.area → areas

docencia.curso, clase, grupo → grupos

docencia.profesor → profesores

docencia.local → locales

docencia.siglas → asignaturas

1. Cambiar la definición de la restricción de clave primaria de la tabla Departamentos para asignarle el nombre dep_pk.
2. Añadir la regla de negocio de que los profesores sólo pueden tener una dedicación de tiempo completo (TC), o parcial de 6 horas (6 h) o de 3 horas (3 h). Comprobar que se ha modelado bien intentando modificar la dedicación del profesor con codigo=1 al valor '20' y observando el mensaje de error que se produce.
3. Comprobar cuales son las restricciones que están definidas en la tabla Locales. Una de ellas no es necesaria al ser redundante. Detectarla y eliminarla. Realizar alguna operación de tipo DML para demostrar que aunque dicha restricción se ha eliminado, en realidad se sigue cumpliendo gracias a otra restricción.
4. Desactivar la restricción NOT NULL en la columna nombre de la tabla Departamentos. Comprobar mediante alguna operación UPDATE que está desactivada. Volver a intentar activarla y averiguar que efecto produce el cambio producido por la operación anterior. Resolver la situación de forma que la restricción queda activada.
5. Definir la regla de negocio de que una asignatura tiene una hora semanal de clase (de teoría y/o de práctica) por cada 3 créditos. Comprobar antes de añadirla al esquema, que dicha regla se cumple con los datos actuales.