



BASES DE DATOS

Práctica 2

Restricciones

UCLM-ESI



Objetivos

- Aprender a implementar **reglas de negocio** en un esquema ORACLE mediante la inclusión de **restricciones de integridad**.



Bibliografía

- 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 PRIMARY KEY
- Restricción CHECK
- Añadiendo Restricciones en Tablas Existentes
- Eliminando Restricciones en Tablas Existentes
- Desactivando restricciones
- Activando restricciones
- Restricciones en Cascada



Esquema de Trabajo

- Los ejemplos empleados 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



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.
- 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).



Introducción

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



Introducción

Guía de 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**.



Introducción

Consultar el diccionario de datos

- Obtener la lista de restricciones definidas en el esquema:

```
SELECT * FROM USER_CONSTRAINTS;
```

- Igual, pero solo para la tabla Empleados:

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

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

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



Definición de Restricciones al Crear una Tabla

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



Definición de Restricciones al Crear una 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)  
);
```



Definición de Restricciones al Crear una Tabla Nivel de Columna vs Nivel de Tabla

- **Columna**
 - Se refiere 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.

[CONSTRAINT <nombre>] <tipo de restricción>

- **Tabla**
 - Se refiere 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.

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



Restricción NOT NULL

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

[CONSTRAINT <nombre>] [NOT] NULL

- Ejemplo:

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



Restricción UNIQUE

- Es una restricción de **unicidad**.
- Impide que puedan 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 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.



Restricción UNIQUE

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

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



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.



Restricción PRIMARY KEY

- **Sintaxis:**

- A nivel de columna:

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

- A nivel de tabla:

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

- **Ejemplo:**

```
CREATE TABLE Departamentos (  
    num          NUMBER(4),  
    nombre       VARCHAR2(30) NOT NULL,  
    ...  
    CONSTRAINT dep_pk PRIMARY KEY (num)  
);
```



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.



Restricción FOREIGN KEY

- A Nivel de Columna:
- **Sintaxis:**
[CONSTRAINT <nombre>
REFERENCES [<esquema>].<tabla> [(<columna>[,...])]
[ON DELETE {CASCADE | SET NULL}]
- **Ejemplo:**
CREATE TABLE Empleados (
 dep NUMBER(4)
 CONSTRAINT emp_dep_fk
 REFERENCES Departamentos (num),
 ...
);



Restricción FOREIGN KEY

- A Nivel de Tabla:
- **Sintaxis:**
[CONSTRAINT <nombre>
FOREIGN KEY (<columna>[,...])
REFERENCES [<esquema>].<tabla> [(<columna>[,...])]
[ON DELETE {CASCADE | SET NULL}]
- **Ejemplo:**
CREATE TABLE Empleados (
 dep NUMBER(4)
 ...
 CONSTRAINT emp_dep_fk
 FOREIGN KEY (dep)
 REFERENCES Departamentos (num)
);



Restricción FOREIGN KEY

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**:
 - se activa si no se incluye ON DELETE
 - Consiste en 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).



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.



Restricción CHECK

- **Sintaxis:**
[CONSTRAINT <nombre>] CHECK (<condición>)
- **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)
);



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.



Añadiendo Restricciones en Tablas Existentes

- **Sintaxis:**

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



Eliminando Restricciones en Tablas Existentes

- 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];
```



Eliminando Restricciones en Tablas Existentes

- **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;
```



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];
```

- **Ejemplo:**

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

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



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>;
```
- **Ejemplo:**
Volver a activar la restricción de clave primaria de la tabla de Empleados.

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



Restricciones en Cascada

- 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.



Restricciones en Cascada

```
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) );
```

Error al ejecutar

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

Error al ejecutar

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

Correcto al añadir la opción **CASCADE CONSTRAINTS**:

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