# Práctica 7. CREACIÓN DE OBJETOS EN SQL

## 0. Introducción

Esta práctica se dedica al estudio de las sentencias SQL de definición de datos (DDL) que permiten definir y borrar objetos dentro de la BD. Algunos objetos, como las tablas y secuencias, ya han sido introducidos en prácticas anteriores. En el caso de las tablas, ampliaremos la sintaxis vista en la práctica 2 para la sentencia CREATE con la definición de restricciones de clave primaria, claves externas y otras. Se recuerda que la sintaxis de las operaciones DROP y ALTER sobre las tablas también está incluida en la práctica 2.

Cada una de las sentencias DDL tienen cláusulas ampliadas que pueden ser consultadas dentro de los manuales de referencia de Oracle y MySQL.

## 1. Restricciones en tablas

### *1.1 Definición de restricciones*

Las tablas, como ya definimos en la práctica 2 son objetos de la base de datos que almacenan los datos. La información sobre la estructura de cada tabla se almacena en el diccionario de datos. La sintaxis de la sentencia de creación de una tabla es la siguiente:

```
CREATE TABLE nombre_tabla
({nombre_atributo tipo [DEFAULT valor]{restriccion_atributo},
      ...}
 { restriccion_de_tabla ,}
);
```

Las restricciones son condiciones que limitan los valores válidos de uno (restricciones de atributo) o más atributos (restricciones de tabla) en una tabla. Sirven para expresar las reglas de integridad y semánticas para los esquemas de relación de la base de datos.

**Sintaxis Oracle**

```
[CONSTRAINT       nombre_de_restricción]
tipo_de_restricción [DISABLE] [EXCEPTIONS INTO tabla_destino]
```

**Sintaxis MySQL**

```
[CONSTRAINT       nombre_de_restricción]
tipo_de_restricción
```

- **nombre_de_restricción** identifica la restricción sobre la tabla o atributo dentro del Diccionario de Datos; si se omite, ORACLE genera un nombre con el formato SYS_Cn donde n es un entero que identifica de forma única la restricción en la Base de Datos.
- La opción **DISABLE** permite deshabilitar la comprobación de una restricción. Puede ser útil para acelerar determinadas operaciones, aunque supone una degradación de la seguridad en la semántica de los datos, por lo que suele realizarse un paso posterior de comprobación y se activará de nuevo la restricción con la sentencia:

```
ALTER TABLE nombre_tabla
            [CONSTRAINT nombre_de_restricción] tipo_restricción;
```

- La opción **EXCEPTIONS** sirve para almacenar en nombre_tabla_destino el conjunto de tuplas que no verifiquen la restricción una vez se haya vuelto a activar con ALTER TABLE. La tabla *tabla_destino* debe existir antes de usar esta acción.
- Los **tipos de restricciones** que utilizaremos más frecuentemente son los siguientes:

- **NOT NULL**: Impide que el atributo pueda contener valores nulos.
- **UNIQUE**: Designa un atributo o combinación de atributos como clave única. Sirve para definir claves alternativas.
- **PRIMARY KEY**: Permite la definición de claves primarias. Designa a un atributo o combinación de atributos (restricción de tabla) como clave primaria de la tabla. Naturalmente, todos sus atributos deben ser NOT NULL (Primera regla de integridad de la entidad) y la combinación de los mismos única. Sin embargo, ni la restricción NOT NULL ni UNIQUE se deben especificar para esos atributos.
- **REFERENCES**: Permite la definición de claves externas.
- **CHECK (condición)**: Permite establecer condiciones sobre el dominio válido de los atributos. Aunque MySQL permite su declaración, actualmente no es aplicable.

## *1.2 Definición de claves primarias*

Oracle permite la definición de uno o varios atributos como clave primaria de una tabla, evitando que la combinación de los valores de dichos atributos se puedan duplicar dentro de ella y optimizando las operaciones de consulta, modificación y borrado sobre la tabla que utilicen como criterio de selección dichos atributos.

**Sintaxis 1**: Restricción de atributo.

```
nombre_atributo tipo PRIMARY KEY
```

**Sintaxis 2**: Restricción de tabla.

```
PRIMARY KEY ( lista_atributos )
```

## *1.3 Definición de claves externas*

Oracle permite la definición de claves externas formadas por uno o varios atributos sobre una tabla, exigiendo que se verifique la regla de integridad referencial.

**Sintaxis 1**: Restricción de atributo.

```
nombre_atributo tipo
REFERENCES nombre_tabla [ ( atributo ) ]
[ON DELETE CASCADE / ON DELETE SET NULL]
```

**Sintaxis 2**: Restricción de tabla.

```
FOREIGN KEY ( lista_atributos )
REFERENCES nombre_tabla [(lista_atributos)]
[ON DELETE CASCADE / ON DELETE SET NULL]
```

- *FOREIGN KEY*: Define un atributo o combinación de atributos como clave externa de una tabla, exigiendo que se verifique la regla de integridad referencial.
- *REFERENCES*: Indica la tabla y su clave correspondiente a la que se refiere la clave externa.
- *ON DELETE SET NULL:* Si se realiza un borrado en la tabla de referencia (tabla padre), se pondrán a nulo los valores de los atributos que forman la clave externa para el conjunto de tuplas que tengan el mismo valor de clave que la de la tupla borrada.
- *ON DELETE CASCADE:* Se mantiene la integridad referencial mediante el borrado de todas las tuplas dependientes al realizar un borrado en la tabla de referencia. Si se omite la cláusula ON DELETE, no permitirá el borrado de las tuplas en la tabla de referencia para las que existan tuplas en nuestra tabla con los mismos valores sobre la clave externa.

## *1.4 Ejemplos*

- Creación de una tabla de profesores donde se guarde información del dni, primer apellido, nombre y cargo que ocupa. Ademas tendrá que cumplir las siguientes restricciones:
  - El campo dni identifica a cada tupla de la tabla
  - Los campos nombre y primer apellido deben estar siempre rellenos.

```
CREATE TABLE Profesor
(dni   char(9)  PRIMARY KEY,
 apellido1 varchar2(20) NOT NULL,
 nombre   varchar2(15) NOT NULL,
 cargo      varchar2(10));
```

- Creación de una tabla de asignatura donde se guarde información del código de la asignatura, su nombre, el número de créditos y el dni del responsable de la asignatura. Ademas tendrá que cumplir las siguientes restricciones:
  - El campo código_asignatura identifica cada tupla de la tabla
  - Los campos nombre y créditos deben estar siempre rellenos; créditos debe tomar un valor >= 1.
  - El nombre de la asignatura no se puede repetir dentro de la tabla (clave alternativa)
  - El valor del dni en responsable debe existir como valor del atributo dni de una tupla de la tabla de profesores (clave ajena).
  - Si causara baja un profesor y ese profesor fuera responsable de una o varias asignaturas, para esas asignaturas el campo responsable debe quedar sin valor asignado.

```
CREATE TABLE Asignatura
(codigo_asignatura number(2),
 nombre varchar2(20) NOT NULL UNIQUE,
 créditos   number(2) NOT NULL CHECK (creditos >=1),
 responsable char(9),
 PRIMARY KEY(codigo_asignatura),
 FOREIGN KEY(responsable) REFERENCES Profesor(dni) ON DELETE SET NULL);
```

# 2. Indices

Un índice es un objeto de la BD que contiene una entrada por cada valor existente en las columnas indexadas de la tabla y proporciona acceso directo y rápido a las tuplas.

## *2.1 Creación de un índice*

Para crear un índice dentro de nuestro esquema, una de las siguientes condiciones debe ser cierta:

- La tabla o cluster a ser indexado debe encontrarse en nuestro esquema
- Poseer privilegios de CREATE INDEX en la tabla que va a ser indexada o tener el privilegio CREATE ANY INDEX

Para crear un índice en otro esquema, se debe tener el privilegio CREATE ANY INDEX.

**Sintaxis básica**:

```
CREATE [UNIQUE] INDEX [esquema.]indice
   ON [esquema.]tabla ( atributo1 , atributo2 , ... );
```

- *UNIQUE*: Especifica que el valor del atributo o de la combinación de valores de los atributos indexados en la tabla debe ser único

## *2.2 Borrado de un índice*

**Sintaxis**:

```
DROP INDEX nombre_de_indice;
```

# 3. Vistas

Las vistas son tablas virtuales formadas a través de una consulta a partir de otras tablas y/o vistas existentes en la base de datos. Suelen utilizarse para realizar los esquemas externos asociados a la base de datos.

## *3.1 Creación de una vista*

**Sintaxis**:

```
CREATE VIEW nombre_de_vista [ ( lista_alias ) ]
      AS subconsulta
      [ WITH CHECK OPTION [CONSTRAINT nombre_de_restricción] ] ;
```

- *lista_alias*: Lista de nombres, separados por comas, dados a los atributos que componen la vista y que se corresponden uno a uno con los de la subconsulta.
- *WITH CHECK OPTION*: Especifica que las inserciones y actualizaciones ejecutadas sobre la vista no pueden generar tuplas que la vista no pueda seleccionar.
- *CONSTRAINT nombre_de_restricción*: Permite asignar un nombre dentro del Diccionario de Datos a la restricción WITH CHECK OPTION definida para esta vista (sólo Oracle).

Ejemplo: Crear una vista que contenga todos los alumnos de la titulacion 103 sobre una tabla alumnos2 de nuestra propiedad

```
CREATE VIEW alumnos103
AS
      SELECT *
      FROM alumnos2
      WHERE id_titulacion=103
      WITH CHECK OPTION;
```

## *3.2 Borrado de una vista*

**Sintaxis**

```
DROP VIEW nombre_de_vista;
```

Ejemplo: Para borrar la vista anterior basta ejecutar la sentencia:

```
DROP VIEW alumnos103;
```

Como ejercicio, vuelve a crear una vista con el mismo nombre, pero ahora sin la cláusula WITH CHECK OPTION y ejecuta la anterior sentencia de inserción, ¿qué ocurre ahora?.

## *3.3 Restricciones de las vistas*

En una vista definida en Oracle no se pueden utilizar los valores de secuencias con NEXTVAL o CURRVAL .

Las sentencias INSERT, UPDATE y DELETE sólo se podrán llevar a cabo sobre vistas si éstas no provienen de una consulta que contenga:

- Joins
- Operadores de conjuntos de tuplas (UNION, MINUS o INTERSECT)
- GROUP BY
- DISTINCT
- Alguna expresión en la lista de atributos seleccionados.

# Anexo I: Ejercicios Propuestos

## *Operaciones con tablas*

1. Crea una tabla denominada MARCAS2 con los mismos atributos y el mismo contenido que la tabla Marcas del esquema practbd. Después de crearla, añade la restricción de clave primaria sobre el atributo, ya que la sentencia CREATE TABLE AS... no copia todas las restricciones.
2. Crea la tabla MODELOS2 con los mismos atributos y el mismo contenido que la tabla Modelos del esquema practbd. Una vez creada, añade las siguientes restricciones:
    - Los valores del cod_marca deben existir para el atributo cod_marca de la tabla MARCAS2
    - Los atributos cod_marca y cod_modelo identifican a cada tupla de la tabla
    - No se permiten valores duplicados ni vacíos para el atributo nombre del modelo
    - La potencia del modelo estará entre 60 y 300
3. Crea una tabla denominada PERSONAS2 con los siguientes atributos:
    - Dni: dni de la persona. Clave primaria
    - Nombre: nombre de la persona.
    - Apellido1: primer apellido de la persona.
    - Apellido2. segundo apellido de la persona.
4. Copia a PERSONAS2 la información de las personas de Cáceres que hay en la tabla de Personas del esquema practbd. Consulta la tabla PERSONAS2 para comprobar su contenido.
5. Da de alta una nueva persona en PERSONAS2 con tus propios datos.
6. Crea una tabla denominada VEHICULOS2 con los siguientes atributos:
    - Matricula: matricula del vehículo. Clave primaria.
    - Fecmat: fecha de matriculación. Debe tener siempre un valor.
    - Propietario: propietario del vehículo. Su valor, si está relleno, debe existir en la tabla de PERSONASCC.
    - Cod_marca: código de la marca del vehículo. Debe estar siempre relleno.
    - Cod_modelo: código del modelo del vehículo. Debe estar siempre relleno.

Además, el modelo de vehículos (cod_marca y cod_modelo) debe existir en la tabla de MODELOS2.

7. Copia en VEHICULOS2 los vehículos de personas de Cáceres, obteniendo esta información de las tablas del esquema practbd.
8. Haz lo mismo que en el ejercicio anterior pero para las personas de Badajoz. ¿Por qué esta operación te devuelve error? ¿De qué tipo de error se trata?
9. Crea una secuencia smarca para generar nuevos valores de marca, que comience en el número 10 y pueda llegar a 999.
10. En la tabla de MARCAS2 añade la marca HONDA, usando la secuencia definida anteriormente.
11. En la tabla de MODELOS2 añade un nuevo modelo CRV 2.0 que es de la marca HONDA y tendrá un valor 1 como código de modelo y 170 caballos de potencia.
12. Da de alta en VEHICULOS2 un vehículo de la marca HONDA y modelo CRV 2.0 con matricula 9999XYZ y matriculado en el día de hoy.
13. Indica en la tabla VEHICULOS2 que el vehículo de matricula 9999XYZ es de tu propiedad.
14. Realiza las operaciones necesarias para dar de alta a un compañero de clase como propietario de un vehículo de matricula 3333HHH con fecha de matriculación del 1 de enero de 2012 cuyo modelo es el VECTRA 2.0.

## *Creación de índices*

15. Optimizar las consultas por apellidos y nombre en la tabla de PERSONAS2.

## *Trabajo con vistas*

16. Crear una vista MODELOS_POTENTES sobre la tabla Modelos2 que muestre los datos de los modelos de vehículos de potencia igual o superior a 150 caballos. Consulta el contenido de la vista.
17. Intenta actualizar, usando la vista MODELOS_POTENTES, la potencia del modelo VECTRA 2.0 a 140 caballos. Si se produce algún tipo de error (diferente de un error sintáctico), razona por qué.
18. Si el ejercicio anterior no te ha dado ningún tipo de error, consulta la vista MODELOS_POTENTES y comprueba que no te muestra el modelo VECTRA 2.0, que sí aparece en una consulta a la tabla MODELOS2. Actualiza la tabla MODELOS2 para poner de nuevo la potencia del modelo VECTREA 2.0 a 150 caballos.
19. Borra la vista MODELOS_POTENTES y repite el ejercicio 16 para crear la vista de manera que no se permita actualizar filas que la vista no podría luego mostrar. Después, ejecuta de nuevo la operación de actualización del ejercicio 17 para comprobar que no te permite realizarla.
20. Crea una vista VEH_OPEL_HONDA con los nombres completos de propietarios, las matrículas de vehículos y su fecha de matriculación para los vehículos de marca OPEL y HONDA, de tal manera que se muestre el nombre completo del propietario bajo una sola columna.
21. Realiza una consulta sobre la vista anterior para obtener los nombres de propietarios de los vehículos matriculados en el año 2013, ordenando por el nombre alfabéticamente

## *Borrado de tuplas y objetos*

22. Borra de VEHICULOS2 todos los vehículos de marca OPEL.
23. Borra todos los objetos que has creado en estos ejercicios: vistas, índice, secuencia y tablas.