BASE DE DATOS - CURSO 02/03
PRÁCTICA 2: SQL: Lenguaje de definición de datos (DDL)

OBJETIVOS:

Enseñar al alumno las sentencias que forman el lenguaje de definición de datos de SQL, es decir, el subconjunto de órdenes que nos permitirán crear o editar esquemas de bases de datos.

MATERIAL:

Oracle 8

BIBLIOGRAFÍA:

- Oracle 8, Guía de aprendizaje. Michael Abbey, Michael Corey, McGraw-Hill 1997.
- La biblia de Oracle 8. Ediciones Anaya Multimedia, 1998.
- Guía de SQL, James R. Groff, Paul N. Weinberg, McGraw-Hill, 1998.
- Understanding the new SQL: A Complete Guide, Jim Melton, Alan R. Simon, Morgan Kaufmann Publisher, 1993.
- A Guide to SQL Standard (fourth edition). Date, C.J. and Darwen, H. Addison-Wesley, 1997.
- SQL. Manual de Referencia del Programador. Freeze, W.S. Paraninfo, 1998.

SQL: El lenguaje de definición de datos (DDL)

El lenguaje de definición de datos permite:

- Definir y crear una nueva tabla.
- Suprimir una tabla que ya no se necesita.
- Cambiar la definición de una tabla existente.
- Definir una tabla virtual (o vista) de datos.
- La concesión de privilegios sobre un objeto de la base de datos.
- La revocación de privilegios sobre un objeto de la base de datos.
- Construir un índice para hacer más rápido el acceso a la tabla.
- Controlar el almacenamiento físico de los datos por parte del DBMS.

Sentencias sobre Tablas

Crear una Tabla

```
CREATE TABLE <nombre tabla>
  ( <nombre columna> <tipo de dato>
    [NOT NULL] [UNIQUE] [CONSTRAINT <nombre restricción>][PRIMARY
    KEY] [REFERENCES][DEFAULT][CHECK]
    | [PRIMARY KEY (<lista columnas>)]
    | [FOREIGN KEY (<lista columnas>) REFERENCES (nombretabla)]
    | [UNIQUE (<lista columnas>)] [CONSTRAINT <nombre
    restricción>],[...] )
  | [CHECK (condición de búsqueda)]
```

- La sentencia **CREATE TABLE** se utiliza para crear una tabla.
- Dentro de la tabla podemos especificar una serie de columnas que contienen datos y restricciones que verifican datos y especifican otras características de la tabla.
- Cada columna debe tener asignado un tipo de dato válido.
- Puede especificarse **Not Null** para asegurar que la columna siempre contiene datos.

BASE DE DATOS - CURSO 02/03
PRÁCTICA 2: SQL: Lenguaje de definición de datos (DDL)

- También podemos especificar un valor por omisión para la columna que va utilizarse cuando se inserte una nueva fila en la tabla y no se especifique un valor, o se especifique como **Default**.

- Podemos definir una **Primary Key** (clave primaria) para la tabla o definir una columna particular o un conjunto de columnas que sean **Unique** (que sus valores no se puedan repetir en la tabla). Además podemos definir una **Foreign Key** (clave ajena) que indique que el valor de la lista de columnas debe existir en otra tabla.

Tipo de datos

Tipo de dato	Descripción
Char(tamaño)	Almacena datos de tipo carácter de longitud fija, con un máximo de 2000 caracteres)
varchar2(tamaño)	Almacena datos de tipo carácter de longitud variable, con un tamaño máximo de 4000
varchar	Actualmente es igual que char
Long	Almacena datos de tipo carácter de longitud variable, hasta 2 gigabytes. Solo se permite un Long por tabla. Una columna de tipo Long no puede utilizarse como parte de un índice. Una función almacenada no puede devolver un Long. Las cláusulas Where, Group By, Order By, Unique, o Connect By no pueden referenciar a una columna Long.
blob	Es un objeto binario de gran tamaño, siendo el tamaño máximo 4 GB (gigabytes). Normalmente un blob se utiliza para almacenar una imagen, datos de voz, o cualquier otro bloque de datos grande no estructurado.
date	Almacena fechas desde el 1 de enero del 4712 a.C. hasta el 31 de diciembre del 4712 d.C.
Integer	Un número entero que no tiene parte fraccionaria. Normalmente un Integer será un valor de 32 bits con un rango de -2147483648 a +2147483647
Smallint	Representa un número entero que no contiene parte fraccionaria. Su precisión nunca será mayor que la de un Integer. Es un valor de 16 bits entre -32768 y +32767
Number(1,d)	Almacena datos de tipo numérico, siendo "l" la longitud y "d" el número de dígitos decimales
Raw(tamaño)	Datos binarios puros con una longitud máxima de 2000 bytes. Sirven para almacenar datos de tipo binario como sonido e imágenes digitalizadas.

BASE DE DATOS - CURSO 02/03

PRÁCTICA 2: SQL: Lenguaje de definición de datos (DDL)

```
CREATE TABLE <nombre de tabla>
(nombre_columna1 tipo [restricción de columna],
.....
nombre_columnaN tipo [restricción de columna],
[restricción_de_tabla]);
```

Restricciones de columnas

NOT NULL. La columna no permitirá valores nulos.

CONSTRAINT. Permite asociar un nombre a una restricción.

DEFAULT valor. La columna tendrá un valor por defecto. El DBMS utiliza este valor cuando no se especifica un valor para dicha columna.

PRIMARY KEY. Permite indicar que esta columna forma parte de la clave primaria.

REFERENCES. Es la manera de indicar que este campo, es clave externa y hace referencia a la clave primaria de otra tabla. Esta **foreign key** es sólo de una columna.

UNIQUE. Obliga a que los valores de una columna tomen valores únicos (no puede haber dos filas con igual valor). Se implementa creando un índice para dicha(s) columna(s).

CHECK (condición). Permite indicar que condición debe de cumplir esa columna.

Restricciones de tablas

PRIMARY KEY (columna1, columna2...). Permite indicar que columna/s forma/n la clave primaria.

FOREIGN KEY (columna1, columna2...) REFERENCES NombreTabla. Indica que este campo/s, es/son clave externa y hace referencia a la clave primaria de otra tabla.

UNIQUE (columna1, columna2...). El valor combinado de una o varias columnas va a ser único.

CHECK (condición). Permite indicar que condición deben de cumplir varios campos de la tabla.

BASE DE DATOS - CURSO 02/03

PRÁCTICA 2: SQL: Lenguaje de definición de datos (DDL)

La cláusula **Foreign Key** tiene unas opciones optativas que se explican a continuación:

- Cómo debe tratar el DBMS un valor NULL en una o más columnas de la clave ajena, cuando lo compare con las filas de la tabla padre.
- Una regla de supresión opcional para la relación (CASCADE, SET NULL, SET DEFAULT, NO ACTION). De esta forma se determina la acción que se debe realizar cuando se elimina una fila padre.
- Una regla de actualización para la relación, que determina la acción que se debe realizar cuando se actualiza una parte de la clave primaria de la fila padre (CASCADE, SET NULL, SET DEFAULT, NO ACTION).

EJEMPLOS

areas(código, nombre, departamento) (código es la clave primaria)
departamentos(código_dpto, nombre) (código_dpto es la clave primaria)

La tabla areas tiene una clave ajena

areas.departamento → departamentos

```
CREATE TABLE areas
(
  codigo char(3) not null,
  nombre char(55) not null,
  departamento char(3) not null,
  Primary key(codigo),
  Foreign key(departamento) REFERENCES departamentos
  ON DELETE SET NULL ON UPDATE CASCADE);
```

ON DELETE Set Null Significa que si se borra algún departamento de la tabla departamentos el campo departamento de la tabla areas se pone como Null.

ON UPDATE CASCADE Significa que si se modifica el campo código_dpto de la tabla departamentos, también se modificara en la tabla áreas.

```
CREATE TABLE departamentos
(
  codigo_dpto char(3) not null,
  nombre char(40) not null,
  Primary key(codigo_dpto)
);
```

BASE DE DATOS - CURSO 02/03

PRÁCTICA 2: SQL: Lenguaje de definición de datos (DDL)

Renombrar una tabla

RENAME TABLE <nombre tabla existente> TO <nuevo nombre tabla>

Eliminar una tabla de la base de datos

DROP TABLE <nombre tabla> [CASCADE, RESTRICT]

DROP TABLE DEPARTAMENTOS CASCADE

DROP TABLE DEPARTAMENTOS RESTRICT

(La tabla se borra sólo si no se hace referencia a ella en ninguna restricción, i.e. en la definición de claves ajenas)

Modificar una tabla

ALTER TABLE <nombre tabla>

{ ADD <nombre columna nueva> <tipo de dato> [NOT NULL]

MODIFY <nombre columna> [SET DEFAULT | DROP DEFAULT] valor

DROP <nombre columna> [CASCADE | RESTRICT]

ADD [PRIMARY KEY (nombre columna) |

FOREIGN KEY (nombre columna) REFERENCES nombre_tabla | UNIQUE (nombre columna) | CHECK (condición)

DROP CONSTRAINT nombre-restricción [CASCADE| RESTRICT]

La sentencia **Alter Table** se utiliza para cambiar una tabla existente. Dentro de la tabla podemos **Add** (añadir) o **Drop** (borrar) columnas, restricciones, condiciones de comprobación, claves primarias y claves ajenas.

EJEMPLO

Agregar a la tabla areas el campo Responsable de tipo char(30)

alter table areas

ADD responsable char(30) not null;

Modificar el campo nombre de la tabla departamentos a char(50)

alter table departamentos

MODIFY nombre char(50);

Sentencias sobre Sinónimos

Crear un sinónimo

CREATE SYNONYM <nombre sinónimo> FOR <nombre tabla>

create synonym are for areas;

create synonym dep for departamentos;

Borrar un sinónimo

DROP SYNONYM <nombre sinónimo>

DROP SYNONYM are

EJERCICIOS

- Utilizando SQL*Plus crear las tablas correspondientes al siguiente esquema de base de datos relacional relacionado a la gestión de los préstamos de una biblioteca:

Libro (signatura, autor, titulo, editor, clase)

Usuario (carnet, nombre, direccion)

Clase (clave, tiempo_de_prestamo)

Prestamo (signatura, carnet, fecha_inicio, fecha_fin)

Con las siguientes claves ajenas:

Libro.clase → Clase

Prestamo.signatura → Libro

Prestamo.carnet → Usuario

- Agregar a la tabla Usuario el campo Fecha_Ingreso (que sea obligatorio)
- Agregar a la tabla Libro el campo Prestado (que sea obligatorio), y asignarle por defecto el valor 1.
- Al finalizar borrar las tablas creadas.