BASES DE DATOS Y SISTEMAS DE INFORMACIÓN

Tema 2.3

Miquel Gómez Corral Ing. Informática ETSINF, UPV

Tema 2.3

Lenguaje de definición de Datos (DDL)

Este subconjunto de SQL permite crear, modificar y eliminar componentes de bases de datos. Se usan instrucciones para definir: Relaciones/tablas, Vistas y permisos.

DEFINICIÓN DE TABLA

CREATE TABLE *nombre_tabla* (elemento1, elemento2,…,elementoN)

Elementos: Para los elementos hay que definir una "definición de atributo" y el "tipo de datos"

- **Definición de atributo**: **Nombre atributo** + **Valor por defecto** "DEFAULT {valor | NULL} + restricciones:

CONSTRAINT nombre restricción +
{**NOT NULL** | **UNIQUE** | **PRIMARY KEY** | **CHECK** (condición búsqueda) | **REFERENCES** *nombre_relación* a la que hace referencia y opcionalmente [(*nombre_atributo*)] //Clave ajena
[**ON DELETE** {CASCADE| SET NULL| SET DEFAULT| NO ACTION}] //Relaciones intefridad
[**ON UPDATE** {CASCADE| SET NULL| SET DEFAULT| NO ACTION}] //Relaciones intefridad
[cuándo_comprobar] :: [[NOT] DEFERRABLE] [INITIALLY {IMMEDIATE | DEFERRED}]

Constraint es el nombre de las restricciones, después se le puede especificar el qué se le añade: *no sea nulo, sea único… y si es o no* **calve ajena**. En esto, se le puede especificar *qué hacer al borrar y al actualizar*. También **cuando comprobar estos cambios**, pudiendo especificar 2 modos (inmmediate y deferred), y si se puede cambiar el modo (deferrable o NOT deferrable). *Lo normal es que esté en deferred, se comprueba después de cada transacción y si incumple algo anula la transacción entera.*

- **Tipo de datos**: {VARCHAR [(n)]| CHAR [(n)]| INTEGER [(n [,n])] | DATE}. En cada uno se le puede especificar un número "(n)" que indique el tamaño.

Ejemplos

```
CREATE TABLE puerto (
      nompuerto VARCHAR2(35) CONSTRAINT PK_puerto PRIMARY KEY,
      altura NUMBER(4),
      categoria CHAR(1),
      pendiente NUMBER(3,2),
      netapa NUMBER(2) NOT NULL CONSTRAINT FK_puerto_eta REFERENCES etapa (netapa),
      dorsal NUMBER(3) CONSTRAINT FK_puerto_cicli REFERENCES ciclista (dorsal)
);
```

```
CREATE TABLE llevar (
      codigo CHAR(3) CONSTRAINT FK_llevar_mai
                              REFERENCES maillot (codigo),
      dorsal NUMBER(3) NOT NULL CONSTRAINT FK_llevar_cicli
                        REFERENCES ciclista (dorsal),
      netapa NUMBER(2) CONSTRAINT FK_llevar_etapa
                              REFERENCES etapa (netapa),
      CONSTRAINT PK lle PRIMARY KEY (netapa, codigo) //se pone al final xq afecta a varias cosas
);
```

```
CREATE TABLE Asignatura (
       Cod_asg CHAR(5)
             CONSTRAINT pk_asignatura PRIMARY KEY DEFERRABLE INITIALY DEFERRED,
       Nombre VARCHAR(50) NUT NULL UNIQUE,
       Semestre CHAR(2) NOT NULL
             CHECK (semestre IN ('1A','1B','2A','2B'…))
       …
)
```

MODIFICACIÓN DE LA DEFINICIÓN DE TABLA

```
ALTER TABLE nombre_tabla
      { ADD definición_atributo
      | MODIFY [COLUMN ] (nombre_atributo)
            { DROP DEFAULT
            | SET DEFAULT {literal | funcion_sistema | NULL}
            | ADD definicion_restriccion
            | DROP nombre_restriccion }
      | DROP [COLUMN ] nombre_atributo {RESTRICT | CASCADE}}
      | ADD restricción de tabla
      | DROP restricción de tabla}
```

Claves ajenas cruzadas: Es posible que si hay **claves ajenas cruzadas** haya que **relejar la creación de una tabla** y **ponerle esa clave ajena después**.

Las opciones que tenemos para modificar una tabla son: añadir un atributo, modificar una columna existente (eliminan o cambiando el valor por defecto y/o añadiendo o eliminando restricciones ya creadas), eliminar columnas, añadir restricciones a toda la tabla o borrarlas.

Ejemplos

```
ALERT TABLE Departamento

      ADD CONSTRAINT fk_departamento_profesor FOREIGN KEY (director)
                              REFERENCES Profesor(dni) DEFERRABLE INITIALLY IMMEDIATE;
```

```
ALTER TABLE Puerto

      ADD provincia VARCHAR(30) NOT NULL;
```

BORRADO DE TABLA

```
DROP TABLE nom_relación {CASCADE CONSTRAINTS}
```

Se pone "**CASCADE CONSTRAINTS**" si se quiere eliminar TODAS las claves ajenas que apuntaban a esa relación.

DEFINICIÓN DE VISTAS

Son "tablas virtuales" que muestran el **resultado de una consulta**, pero **NO se guardan en disco**: Muestra solo una parte de la Base de Datos. *Se pueden crear, algunas modificar con "INSERT INTO nombreVista…" y borrar con "DROP VIWE nombre…".*

```
CREATE VIEW Etapas_con_puertos [Nom_Atributo1,… Nom_AtributoN]AS CONSULTA SELECT
```

Se pone el nombre de la tabla y después el nombre de cada atributo que te va a devolver a la consulta (si quieres).

WITH CHECK OPTION: Se puede añadir opcionalmente después de la consulta para que **no te deje añadir filas que NO cumplan** la definición de la consulta SELECT. *Si te digo que tenga >20 años y me pones uno con 10 NO deja.*

Vista actualizable: Una vista es actualizable si **cada fila de la vista se corresponde directamente a una fila de una tabla existente**. *Esto quiere decir que, por ejemplo, UNIONS, JOINS, igualar valores… y cosas así no cuadran*. Se puede hacer inserciones, modificados y borrados sobre esta vista haciendo que se actualice también la tabla original.

Actualizable

```
CREATE VIEW Etapa_larga AS
SELECT * FROM Etapa
WHERE km>100;
```

NO Actualizable

```
CREATE VIEW Ganadores AS
SELECT C.dorsal,C.nombre,COUNT(*) AS Etapas_ganadas
FROM Ciclista C, Etapa E
WHERE C.dorsal=E.dorsal
GROUP BY C.dorsal, C.nombre
```

LA GESTIÓN DE AUTORIZACIONES EN SQL *Privilegios*

```
GRANT {ALL|SELECT|INSERT|DELETE|UPDATE} ON nom_tabl TO {User1, User2…|PUBLIC}
```

```
REVOKE {ALL|SELECT|INSERT|DELETE|UPDATE} ON nom_tabl FROM {User1,User2…|PUBLIC}
```

WITH GRANT OPTION: Se puede añadir esa clausula al final de las operaciones para decir que al usuario que se le de ese privilegio sobre algo, él puede también darlo o quitarlo después a otra persona.

```
GRANT SELECT ON Puerto TO Pepe;
```

```
REVOKE SELECT ON Puerto FROM Pepe;
```