Universidad
Europea de Madrid

LAUREATE INTERNATIONAL UNIVERSITIES

SQL. MANIPULACIÓN DE DATOS

MODIFICACIÓN Y ELIMINACIÓN DE TABLAS

Índice

Presentación

Anteriormente aprendimos a crear el esquema de las tablas de la base de datos. Vimos que en la propia sentencia de definición de la tabla se podían incluir cláusulas que permiten definir las restricciones de integridad que el SGBD debe comprobar al modificar la instancia de la base de datos.
Hay más operaciones que debe permitirnos realizar el LDD del SQL, entre ellas la modificación y eliminación del esquema de una tabla previamente creada.

En este tema aprenderás a:

- **Realizar modificaciones** del esquema de una tabla a través de la sentencia **ALTER TABLE**.
- **Conocer una estrategia alternativa** para conseguir modificar el esquema de una tabla, ya que no todos los dialectos de SQL disponen de la sentencia ALTER TABLE. Depende de la implementación concreta que se utilice en el SGBD que se esté utilizando.
- **Eliminar tablas**.
- Aplicar estos conocimientos en un caso práctico: el **ejemplo de las aerolíneas**.

La sentencia alter table: añadir y eliminar columnas

ALTER TABLE permite modificar el esquema de una tabla que ha sido creada previamente. Se compone de seis opciones que permiten: añadir y eliminar **columnas** a la tabla, añadir y eliminar **restricciones**, y añadir y eliminar **un valor por defecto**. Las vemos a continuación y en las siguientes pantallas.

Añadir una nueva columna a la tabla: ADD COLUMN

La sintaxis es similar al modo de definir una columna cuando se usa el CREATE TABLE.

```
ALTER TABLE <tabla> ADD COLUMN <columna> <tipo>
    <resDatoRequerido> <resDominio>;
```

- **<tabla>** es el nombre de la tabla que se quiere modificar.
- **<columna>** es el nombre del nuevo atributo que se quiere añadir.
- **<tipo>** tipo del atributo.

Eliminar una columna existente de la tabla: DROP COLUMN

```
ALTER TABLE <tabla> DROP COLUMN <columna> <acción>;
```

- **<tabla>** es el nombre de la tabla que se quiere modificar.
- **<columna>** es el nombre del atributo que se quiere eliminar.
- **<acción>** indica si debe o no propagarse la eliminación a otras tablas donde se referencia esta columna.
  Existen dos posibles acciones:
  - **RESTRICT**: si la columna está siendo referenciada por otro objeto de la BD, no se borra. Es la opción por defecto. De esta forma si se intenta eliminar una columna que está siendo referenciada, el SGBD dará un error y no ejecutará la sentencia ALTER TABLE. Para que se ejecute la sentencia se tendrá que eliminar previamente la referencia.
  - **CASCADE**: la operación DROP se propaga afectando a los objetos que pudieran hacer referencia a la columna. No solo se borra esta columna sino todas las referencias que aparezcan a este atributo en el resto de la BD.

La sentencia alter table: añadir y eliminar restricciones

Al crear tablas en SQL se debe definir qué atributo ó conjunto de atributos forman la clave primaria y, en caso de existir, qué atributo es clave externa en otra tabla de la base de datos. Estas restricciones de entidad (PRIMARY KEY) y referencia (FOREIGN KEY) también se pueden añadir y eliminar con las sentencia ALTER TABLE.

Añadir una nueva restricción a la tabla: ADD <restricción></restricción>

```
ALTER TABLE <tabla> ADD <defRestricciónTabla>;
```

- **<tabla>** es el nombre de la tabla que se quiere modificar.
- **<defRestriccionTabla>** es una cláusula de tipo PRIMARY KEY, FOREIGN KEY, UNIQUE ó CHECK que definimos en la lección anterior.

Añadir restricción

Ejemplo

```
ALTER TABLE clientes_andaluces ADD FOREIGN KEY (nombre_cli) REFERENCES
cliente(nombre_cli);
ALTER TABLE clientes_andaluces ADD PRIMARY KEY (calle);
```

Eliminar una restricción de la tabla: DROP <restricción></restricción>

```
ALTER TABLE <tabla> DROP <restriccion>;
```

- **<tabla>** es el nombre de la tabla que se quiere modificar.
- **<restriccion>** puede ser PRIMARY KEY, FOREIGN KEY.

Ejemplo Eliminar restricción

Eliminar restricción

```
ALTER TABLE clientes_andaluces DROP PRIMARY KEY;
ALTER TABLE clientes_andaluces DROP FOREIGN KEY
clientes_andaluces_ibfk_1;
```

clientes_andaluces_ibfk_1 es el nombre que le ha dado el SGBD a la restricción de clave externa establecida en la tabla clientes_andaluces

La sentencia alter table: añadir y eliminar valores por defecto

Al crear una tabla con la sentencia CREATE TABLE es posible indicar si queremos que una columna tenga un valor por defecto en el caso de que no se especifique ninguno al hacer el INSERT.
Con la sentencia ALTER TABLE podemos establecer ó eliminar valores por defecto definidos sobre atributos.

Asignar un valor por defecto a una columna: SET DEFAULT

```
ALTER TABLE<tabla> ALTER <columna> SET DEFAULT <opciónPredeterminada>
```

Donde:

- **<columna>** es el nombre del atributo al que queremos añadir el valor por defecto.
- **<opcionPredeterminada>** es el valor que queremos establecer por defecto para esta columna.

Asignar un valor por defecto

Asignar un valor por defecto

```
ALTER TABLE cliente ALTER estadoCivil SET DEFAULT 'casado';
```

Eliminar un valor por defecto de una columna: DROP DEFAULT

```
ALTER TABLE <tabla> ALTER <columna> DROP DEFAULT;
```

Eliminar un valor por defecto

Eliminar un valor por defecto

```
ALTER TABLE cliente ALTER dirección DROP DEFAULT;
```

Eliminar tablas

Para eliminar una tabla de la base de datos, junto con todas las tuplas o filas en ella contenidas, se usa la instrucción **DROP TABLE**:

```
DROP TABLE <tabla> <acción>
```

Donde:

- **<tabla>** es el nombre de la tabla que se quiere modificar.
- **<acción>**. En el caso de que haya referencias externas a esta tabla, <acción> indica qué se hará con estas referencias. Puede ser **RESTRICT** o **CASCADE**.

RESTRICT

La tabla se elimina sólo si no hay objetos que hagan referencia a ella. Es la opción por defecto. De esta forma no se permitirá eliminar ninguna tabla que contenga referencias externas.

CASCADE

Elimina la tabla y los objetos que hagan referencia a ella, propagando la eliminación a estos y así sucesivamente.

Ejemplo

```
CREATE TABLE adulto_con_bebe (

            dni varchar(9) NOT NULL DEFAULT '',
            nombre_bebe varchar(20) DEFAULT NULL,
            fecha_nac_bebe date DEFAULT NULL,
            PRIMARY KEY (dni),
            FOREIGN KEY (dni) REFERENCES pasajero (dni)
            )
DROP TABLE pasajero RESTRICT;
```

Rtdo: Cannot delete or update a parent row: a foreign key constraint fails

Ejemplo de aerolíneas I

Vamos a crear las tablas de la base de datos de **aerolíneas** teniendo en cuenta las restricciones a las que están sometidas en el mundo real.

Tabla pasajero

```
PASAJERO (dni, nombre, apellido1, apellido2)
```

- Suponemos que almacenamos la letra del dni, por lo que estará compuesto de 9 caracteres exactamente (usamos el tipo *char(9)*).
- **Dni** es la clave primaria de la relación por lo que tiene que ser única y no puede contener valores nulos.
- **Nombre**, **apellido1** y **apellido2** son también cadenas de caracteres. Permitiremos que apellido2 no se introduzca, no así para el nombre y el apellido1, ya que en el formulario de entrada de datos nombre y apellido 1 eran datos obligatorios.

Tabla adulto con bebe

```
ADULTO_CON_BEBE (dni, nombre_bebe, fecha_nac_bebe)
```

- **Dni** es clave primaria de esta relación y clave externa de pasajeros.
- **Nombre_bebe** es una cadena de caracteres y **fecha_nac_bebe** es de tipo fecha. Ninguna de las dos son campos obligatorios en el formulario por lo que suponemos que pueden estar vacíos.

Sentencia tabla pasajero

Sentencia tabla adulto_con_bebe

Sentencia tabla pasajero

```
CREATE TABLE pasajeros (
    dni CHAR(9) NOT NULL UNIQUE,
    nombre VARCHAR(20) NOT NULL,
    apellido1 VARCHAR(20) NOT NULL,
    apellido2 VARCHAR (20),
    PRIMARY KEY (dni));
```

Sentencia tabla adulto_con_bebe

```
CREATE TABLE adulto_con_bebe (
    dni CHAR(9) NOT NULL UNIQUE,
    nombre_bebe VARCHAR(20),
    fecha_nac_bebe DATE,
    PRIMARY KEY (dni),
    FOREIGN KEY (dni) REFERENCES pasajeros(dni));
```

Ejemplo de aerolíneas II

Tabla niño

```
NIÑO ( dni,  fecha_nac_niño)
```

Tabla tarjeta_crédito

```
TARJETA_CREDITO (n_tarjeta, tipo, caducidad.mes, caducidad.año,
titular, país, cod_seguridad)
```

N_tarjeta	El número de una tarjeta de crédito está compuesta por 16 dígitos. Utilizaremos, por tanto, una cadena de caracteres de longitud fija. Además es la clave primaria de la relación por lo que no puede contener valores nulos y debe ser única.
Tipo	Cuando se rellena el tipo de tarjeta de crédito en el formulario de reserva del billete se obliga al cliente a elegir en una lista desplegable con los valores que se muestran en la imagen. Se va a definir por tanto una restricción de dominio. En este punto se puede utilizar la cláusula **check** y/o crear el dominio concreto con la sentencia **CREATE DOMAIN**. Utilizaremos la primera opción.
Caducidad mes	Es un dato obligatorio que el usuario selecciona de una lista desplegable, tal y como se muestra en la figura. Como se puede comprobar es un dato de 3 caracteres cuyos valores están restringidos. Definimos una cadena de longitud fija con una cláusula **check**.
Caducidad_año	Dato numérico de 4 dígitos, obligatorio.
Titular	Cadena de caracteres de longitud variable. Campo obligatorio.
País	En el formulario aparece una lista desplegable con todos los países y el cliente selecciona su país de origen. Si queremos que el usuario introduzca un país existente podemos crear el dominio con todos estos valores ó crear una tabla de países y definir aquí **países** como clave externa. Utilizamos esta última opción.
Cod_seguridad	El código de seguridad de una tarjeta está compuesto por 3 ó 4 dígitos. Utilizamos una cadena de caracteres de longitud variable de 4 dígitos. Es obligatorio.

Sentencia tabla niño

Sentencia tabla tarjeta_credito

Sentencia tabla niño

Dni es la clave primaria de esta tabla y clave externa de pasajeros.

Fecha_nac_niño es de tipo date y suponemos que no es obligatorio introducir su valor.

```
CREATE TABLE niño (
    dni CHAR(9) NOT NULL UNIQUE,
    fecha_nac_niño DATE,
    PRIMARY KEY (dni),
    FOREIGN KEY (dni) REFERENCES pasajeros(dni));
```

Sentencia tabla tarjeta_credito

```
CREATE TABLE tarjeta_de_credito (
                n_tarjeta INT(16) NOT NULL UNIQUE,
                tipo VARCHAR(15) NOT NULL CHECK VALUE IN
('VISA','AMERICAN EXPRESS','UATP/AIRPLUS','DINERS CLUB','MASTER
CARD','CARTE BALNCHE','ACCESS'),
                caducidad_mes CHAR(3) NOT NULL CHECK VALUE IN
('ENE','FEB','MAR','ABR','MAY','JUN','JUL','AGO','SEP','OCT','NOV',
'DIC'),
                caducidad_año INT(4) NOT NULL,
                titular VARCHAR(50) NOT NULL,
                pais VARCHAR(20) NOT NULL,
                cod_seguridad VARCHAR(4) NOT NULL,
                PRIMARY KEY(n_tarjeta),
                FOREIGN KEY (pais) REFERENCES paises(pais));
```

Ejemplo de aerolíneas III

Tabla cliente

CLIENTE (correo_e, teléfono)

Tabla vuelos

```
VUELO (cod_vuelo, clase, avión, salida.fecha, salida.hora,
salida.ciudad, salida.aeropuerto, salida.terminal, llegada.fecha,
llegada.hora, llegada.ciudad, llegada.aeropuerto, llegada.terminal,
empresa, duración)
```

- **Cod_vuelo**: compuesto por 5 caracteres alfanuméricos. Clave primaria de la relación.
- **Clase**: puede ser business, turista, económica. Obligatorio.
- **Avión**: cadena de caracteres de longitud variable. No obligatorio.
- **Salida_fecha** y **llegada.fecha**: ambos de tipo fecha y obligatorios.
- **Salida_hora** y **llegada_hora**: ambos de tipo time y obligatorios.
- **Salida.ciudad** y **llegada.ciudad**: ambos cadena de caracteres de longitud variable y obligatorios.
- **Salida.aeropuerto** y **llegada.aeropuerto**: hacemos una tabla con los nombres de todos los aeropuertos del mundo y definimos estos dos campos como claves externas. De esta forma nos aseguramos que contengan valores válidos. Obligatorios.
- **Salida.terminal** y **llegada.terminal**: cadena de caracteres de longitud variable y obligatorio.
- **Empresa**: cadena de caracteres de longitud variable. No obligatorio.
- **Duración**: formato de horas+minutos (ej. 16h 30m). Lo dividimos en dos campos duración_horas y duración_min de tipo smallint. No obligatorios.

En detalle **Sentencia tabla cliente**

En detalle **Sentencia tabla vuelo**

Sentencia tabla cliente

Correo_e es una cadena de caracteres de longitud variable. Es clave primaria de la relación.

Teléfono es una cadena de longitud variable, como máximo 11 dígitos. Campo obligatorio.

```
CREATE TABLE clientes (
    correo_e VARCHAR(30) NOT NULL UNIQUE,
    telefono INT(11) NOT NULL,
    PRIMARY KEY (correo_e));
```

Sentencia tabla vuelo

```
CREATE TABLE vuelos (
    cod_vuelo VARCHAR(5) NOT NULL UNIQUE,
    clase VARCHAR(10) NOT NULL CHECK VALUE IN ('business', 'turista',
'económica'),
    avion VARCHAR(20),
    salida_fecha DATE NOT NULL,
    salida_hora TIME NOT NULL,
    salida_ciudad VARCHAR(15) NOT NULL,
    salida_aeropuerto VARCHAR(20) NOT NULL,
    salida_terminal VARCHAR(4) NOT NULL,
    llegada_fecha DATE NOT NULL,
    llegada_hora TIME NOT NULL,
    llegada_ciudad VARCHAR(15) NOT NULL,
    llegada_aeropuerto VARCHAR(20) NOT NULL,
    llegada_terminal VARCHAR(4) NOT NULL,
    empresa VARCHAR(15),
    duracion_horas SMALLINT,
    duracion_min SMALLINT,
    PRIMARY KEY(cod_vuelo),
    FOREIGN KEY (salida_aeropuerto) REFERENCES aeropuerto(nombre),
    FOREIGN KEY (llegada_aeropuerto) REFERENCES aeropuerto(nombre));
```

Ejemplo de aerolíneas IV

Tabla aparece

```
APARECE (dni, cod_reserva, comida_especial, n_billete)
```

- **Dni:** clave externa que hace referencia al campo dni de pasajeros. Es clave primaria de la relación junto con cod_reserva, por tanto es un campo obligatorio y único. Mismo dominio que en pasajeros.
- **Cod_reserva:** clave externa de reserva. Clave principal de la tabla junto con dni. Lo definimos, por tanto, obligatorio y único.
- **Comida_especial:** este campo está asociado en el formulario a una lista desplegable que contiene los valores: vegetariana, baja en calorías, sin preferencia, musulmana, sin lactosa. Creamos una restricción de dominio para comprobar que el valor introducido sea alguno de estos. No es un campo obligatorio. Establecemos como valor por defecto *sin preferencia*.
- **N_billete.** Es un dato numérico de 4 dígitos. Es obligatorio.

Tabla líneas

```
LINEAS (cod_reserva, concepto, tarifa, tasas, gastos_gestión,
n_pasajeros)
```

- **Cod_reserva:** clave externa de reservas y clave primaria de esta relación junto con concepto.
- **Concepto**: puede contener 3 valores: adulto, adulto con bebe o niño. Es clave primaria de la relación.
- **Tarifa**: número con dos decimales. Obligatorio.
- **Tasas**: número con dos decimales. Obligatorio.
- **Gastos gestión**: número con dos decimales. Obligatorio.
- **N_pasajeros**: entero obligatorio.
- **Total**: número con dos decimales. Obligatorio.

Sentencia tabla aparece

Sentencia tabla líneas

Sentencia tabla aparece

```
CREATE TABLE aparece (
    dni_pasajero CHAR(9) NOT NULL UNIQUE,
    reserva CHAR(6) NOT NULL UNIQUE,
    comida_especial VARCHAR(15) DEFAULT 'sin preferencia' CHECK VALUE
IN ('vegetariana', 'baja en calorías', 'sin preferencia', 'musulmana',
'sin lactosa'),
    n_billete INT(4) NOT NULL,
    PRIMARY KEY (dni_pasajero, reserva),
    FOREIGN KEY (dni_pasajero) REFERENCES pasajeros(dni),
    FOREIGN KEY (reserva) REFERENCES reservas(cod_reserva));
```

Sentencia tabla líneas

```
CREATE TABLE lineas (
    cod_reserv CHAR(6) NOT NULL UNIQUE,
    concepto VARCHAR(15) NOT NULL UNIQUE CHECK VALUE IN
('ADULTO','ADULTO_CON_BEBE','NIÑO'),
    tarifa DEC(9,2) NOT NULL,
    tasas DEC(9,2) NOT NULL,
    gastos_gestion DEC(9,2) NOT NULL,
    n_pasajeros SMALLINT NOT NULL,
    total DEC(10,2) NOT NULL,
    PRIMARY KEY (cod_reserv, concepto),
    FOREIGN KEY (cod_reserv) REFERENCES reservas(cod_reserva));
```

Ejemplo de aerolíneas V

Tabla compuesta_por

```
COMPUESTA_POR (cod_reserva, cod_vuelo)
```

- **Cod_reserva**. Clave externa y clave primaria junto con cod_vuelo
- **Cod_vuelo**. Lo mismo que la anterior.

Tabla reserva

```
RESERVA (cod_reserva, origen, destino, tipo, fecha_sal, fecha_llegada,
precio_total, correo_e, n_tarjeta)
```

- **Cod_reserva**: es una cadena de 6 caracteres alfanuméricos. Es la clave de la relación.
- **Origen**: es una ciudad. Utilizaremos cadenas de caracteres de longitud variable. Campo obligatorio.
- **Destino**: es una ciudad. Utilizaremos cadenas de caracteres de longitud variable. Campo obligatorio.
- **Fecha_sal**: utilizamos el tipo de dato fecha. Es obligatorio.
- **Fecha_llegada:** igual que el anterior.
- **Precio_total**: número real con dos decimales. Obligatorio.
- **Correo_cli**: cadena de caracteres de longitud variable. Clave externa de clientes.
- **Tarjeta_cli**: mismo tipo que en tarjeta_de_credito. Clave externa de tarjeta_de_credito.

Sentencia tabla compuesta_por

Sentencia tabla reserva

Sentencia tabla compuesta_por

```
CREATE TABLE compuesta_por (
    cod_reserva CHAR(6) NOT NULL UNIQUE,
    cod_vuelo VARCHAR(5) NOT NULL UNIQUE,
    PRIMARY KEY (cod_reserva, cod_vuelo),
    FOREIGN KEY (cod_reserva) REFERENCES reservas,
    FOREIGN KEY (cod_vuelo) REFERENCES vuelos);
```

Sentencia tabla reserva

```
CREATE TABLE reservas (
    cod_reserva CHAR(6) NOT NULL UNIQUE,
    origen VARCHAR(15) NOT NULL,
    destino VARCHAR(15) NOT NULL,
    fecha_sal DATE NOT NULL,
    fecha_llegada DATE NOT NULL,
    precio_total DEC(10,2) NOT NULL,
    correo_cli VARCHAR(30) NOT NULL,
    tarjeta_cli INT(16) NOT NULL,
    PRIMARY KEY (cod_reserva),
    FOREIGN KEY (correo_cli) REFERENCES clientes(correo_e),
    FOREIGN KEY (tarjeta_cli) REFERENCES
tarjeta_de_credito(n_tarjeta));
```

Resumen

En este tema hemos visto cómo se puede **modificar el esquema de una relación** previamente construida utilizando la sentencia ALTER TABLE. Con ella se puede:

- Añadir y eliminar columnas con la cláusula ADD COLUMN y DROP COLUMN.
- Añadir y eliminar restricciones de entidad. ADD PRIMARY KEY, DROP PRIMARY KEY.
- Añadir y eliminar restricciones de referencias. ADD FOREIGN KEY, DROP FOREIGN KEY.
- Añadir y eliminar otras restricciones.
- Añadir y eliminar valores establecidos por defecto. ADD DEFAULT, DROP DEFAULT.

También hemos visto cómo **eliminar tablas** previamente creadas utilizando la sentencia DROP TABLE.

Para ilustrar las sentencias del LDD estudiadas en este tema y en el anterior se ha utilizado, de nuevo, el **ejemplo de las aerolíneas**. Con esas tablas se han mostrado las sentencias CREATE TABLE que se pueden utilizar para implementar este caso práctico.