Ejercicios Tema 3

Creación de Tablas en MySQL: TAREA 2122

```
/* ************************************************
**********   ACTIVIDAD 1   **********************
************************************************** */


/*Elimina si existe la base de datos conferencias_bd03*/
DROP DATABASE IF EXISTS conferencias_bd03;


/*Crea la base de datos conferencias_bd03.*/
CREATE DATABASE conferencias_bd03;


/*Activamos la base de datos*/
USE conferencias_bd03;


/*Creación de las tablas. Comenzamos por las que no tienen claves foráneas */
-- Creación tabla personal
CREATE TABLE personal (
            codPersonal INT PRIMARY KEY,
   dni CHAR(9)  UNIQUE ,
   nombre VARCHAR (20) NOT  NULL,
   apellidos VARCHAR (40) NOT NULL,
   fecNacimiento DATE,
   telefono CHAR(9) ,
   direccion VARCHAR (40)
);
```

```
-- Creación tabla conferencia
CREATE TABLE conferencia (
        codConferencia INT PRIMARY KEY,
   titulo VARCHAR (40) NOT NULL
);

/*Comenzamos a crear las tablas que contienen claves primarias*/
-- Creación tabla administrativo
CREATE TABLE administrativo (
        codAdministrativo INT PRIMARY KEY,
   CONSTRAINT administ_pers_FK
                FOREIGN KEY (codAdministrativo) REFERENCES personal (codPersonal) ON UPDATE CASCADE
);

-- Creación tabla jefe
CREATE TABLE jefe (
        codJefe INT PRIMARY KEY,
   fechaIncorporacion DATE,
   CONSTRAINT jefe_pers_FK
                FOREIGN KEY (codJefe) REFERENCES personal (codPersonal) ON UPDATE CASCADE
      -- cuando un personal que es JEFE es eliminado, debería eliminarse el JEFE de esta tabla y también en cascada el JEFE referenciado en ASISTENCIA
                        ON DELETE CASCADE
);

-- Creación tabla edicion
CREATE TABLE edicion (
        codConferencia INT,
   numero INT,
```

```
   lugar VARCHAR (20) NOT NULL,
   fecha DATETIME DEFAULT NOW(), -- ponemos por defecto la fecha actual
   aforo INT default 0,
   PRIMARY KEY (codConferencia, numero),
   CONSTRAINT edicion_confer_FK
                  FOREIGN KEY (codConferencia) REFERENCES conferencia (codConferencia)
);


-- Creación tabla asistencia
CREATE TABLE asistencia(
         codJefe INT,
   codConferencia INT,
   numero INT,
   PRIMARY KEY (codJefe, codConferencia, numero),
   CONSTRAINT asist_edicion_FK
                  FOREIGN KEY (codConferencia, numero) REFERENCES edicion (codConferencia,
numero),
         CONSTRAINT asist_jefe_FK
                  FOREIGN KEY (codJefe) REFERENCES jefe (codJefe) ON DELETE CASCADE ON
UPDATE CASCADE
);
/* NOTA:
notese que si se actualiza un codPersonal de un jefe  en la tabla personal, este código se podrá
actualizar en la tabla jefe siempre que no esté referenciado en la tabla de asistencia,
para poder actualizarlo hay que añadir también la condición ON UPDATE CASCADE en la Fk
codJEfe de la tabla asistencia


/* ************************************************
**********   ACTIVIDAD 2   **********************
************************************************ */


/* Añadimos la nueva columna a la tabla administrativo y
```

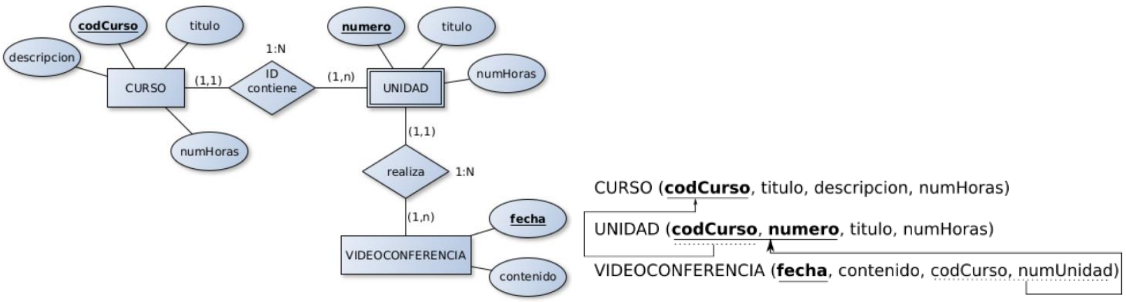
```
 la nueva restricción para indicar la nueva clave foránea */
ALTER TABLE administrativo ADD
(codJefeSupervisor INT,
CONSTRAINT admin_jefe_FK
        FOREIGN KEY (codJefeSupervisor) REFERENCES jefe (codJefe) -- podríamos poner ON DELETE CASCADE
                                                                /* esta condición no la he evaluado en la tarea,
                                                                        si eliminamos un jefe eliminamos este registro en cascada

                        Si no ponemos dicha acción, tendremos en cuenta que no vamos a poder eliminar nunca un personal Jefe que
                                                                        tenga un registro referenciado en esta tabla, ya que por defecto la FK es ON DELETE RESTRICT*/
);

-- Creamos la tabla categoría
CREATE TABLE categoria (
        codCategoria INT PRIMARY KEY,
   nombre VARCHAR (30) NOT NULL
);

/* Añadimos la nueva columna a la tabla conferencia y
 la nueva restricción para indicar la nueva clave foránea */
ALTER TABLE conferencia ADD
(codCategoria INT NOT NULL,
CONSTRAINT confer_categ_FK
        FOREIGN KEY (codCategoria) REFERENCES categoria (codCategoria)
);

/* ************************************************
**********   ACTIVIDAD 3   **********************
```

```
********************************************** */


-- Creamos el índice

CREATE INDEX index_pers_apell_nomb ON personal (apellidos, nombre);

-- show index from conferencias_bd03.personal;

-- creamos la vista

CREATE VIEW personalVista AS SELECT nombre, apellidos FROM personal WHERE fecNacimiento
< '1980-01-01';


-- Cambiamos el nombre a la columna fecNacimiento de la tabla personal

ALTER TABLE personal CHANGE COLUMN fecNacimiento fecNac DATE;

-- ALTER TABLE personal            RENAME COLUMN fecNacimiento TO fecNac;


-- Cambiamos el nombre a la columna fechaIncorporacion de la tabla jefe

ALTER TABLE jefe CHANGE COLUMN fechaIncorporacion fecInc DATE;

-- ALTER TABLE jefe RENAME COLUMN fechaIncorporacion TO fecInc;


-- ojo: al cambiar el nombre de la columna que afecta a la vista, la vista queda inutilizada, por lo
que podríamos modificarla aunque no se ha tenido en cuenta en la tarea

ALTER VIEW personalVista AS SELECT nombre, apellidos FROM personal WHERE fecNac < '1980-
01-01';


-- Creación del usuario

CREATE USER IF NOT EXISTS 'usuario_bd03'@'localhost' IDENTIFIED BY '1234';


-- asignamos los permisos al usuario en la BD y en todas sus tablas.

GRANT INSERT, SELECT ON conferencias_bd03.* TO 'usuario_bd03'@'localhost';


-- eliminamos la base de datos.

DROP DATABASE conferencias_bd03;
```

Creación de Tablas en MySQL: TAREA 2223

Queremos almacenar la información relacionada con la gestión personal de los cursos que ofrecemos en nuestra empresa, así como las distintas videoconferencias que se ofrecen a los alumnos. Ya tenemos el Modelo E-R elaborado y el paso al Modelo Relacional. Ahora queremos pasar las tablas creadas al SGBD MySQL, para ello sigue todos los pasos que se indican a continuación.

El Diagrama Entidad Relación y el Modelo Relacional son los siguientes:



Partiendo de dichas tablas, debes realizar los siguientes apartados utilizando MySQL Workbench:

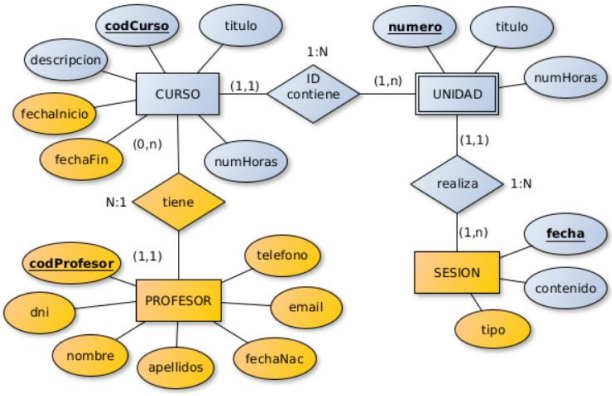
Apartado 1:

Escribe las siguientes sentencias en el orden apropiado utilizando el lenguaje DDL de SQL para que funcionen correctamente en MySQL.

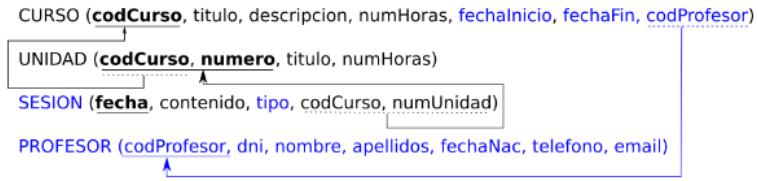
1. Elimina si existe la base de datos cursos_bd03
2. Crea la base de datos cursos_bd03.
3. Para la creación de tablas debes crear una sentencia SQL para cada una de ellas teniendo en cuenta los siguientes subapartados:
   - Elegir el nombre, tipo de dato MySQL y el tamaño más adecuado para cada campo teniendo en cuenta los valores que éstos pueden almacenar.
   - Definir las claves primarias identificadas en cada una de las tablas.
   - Definir las claves ajenas identificadas en cada una de las tablas referenciando a los campos y tablas correspondientes y teniendo en cuenta las siguientes reglas de integridad referencial:
     - Cuando se elimine una unidad de un curso, se deben eliminar las videoconferencias asociadas
     - Cuando un valor de la tabla principal se actualice, se actualizarán los valores a los que referencie.
   - El campo codCurso estará formado por 4 caracteres.
   - El título de los cursos no puede tener valores duplicados.
   - El título de las unidades y el contenido de las videoconferencias no pueden tomar valores nulos.
   - El numHoras de las unidades tiene por defecto el valor 0
   - La fecha de las videoconferencias deben ser mayores del 1 de enero de 2020 (Las fechas se almacenarán con el formato YYYY-mm-dd hh:mm:ss).

Apartado 2:

Si queremos añadir una nueva funcionalidad de forma que almacenemos los profesores que imparten los cursos. Y además queremos dar la posibilidad que para las unidades además de videoconferencias, existan otro tipo de sesiones. Nuestro diagrama entidad-relación y el Modelo Relacional quedarían de la siguiente forma:



El Modelo Relacional es:



Debes crear una sentencia para cada subapartado utilizando el lenguaje DDL de SQL para que funcionen correctamente en MySQL. Para cada subapartado NO puedes eliminar ni tablas ni la base de datos para volverlas a crear de nuevo (ya que podrían contener datos):

1. Renombra la tabla videoconferencia por sesion.
2. Crea el campo tipo de la tabla SESION eligiendo el tipo de dato más adecuado para que sólo se puedan almacenar los siguientes valores: "Videoconferencia", "Sesión Presencial" o "Cuestionario"
3. Crea la tabla PROFESOR, teniendo en cuenta los siguientes subapartados:
   - Elegir el nombre, tipo de dato MySQL y el tamaño más adecuado para cada campo teniendo en cuenta los valores que éstos pueden almacenar.
   - Definir la clave primaria identificada en la tabla.
4. Modifica la tabla CURSO, teniendo en cuenta los siguientes subapartados:
   - Añade los campos fechaInicio y fechaFin, elegiendo el tipo de dato MySQL y el tamaño más adecuado para cada campo teniendo en cuenta los valores que éstos pueden almacenar.
   - Definir la claves ajena identificada en la tabla referenciando al campo y tabla correspondiente.
5. Crea un índice formado por los campos apellidos y nombre de PROFESOR (Ojo es un índice formado por dos campos, no dos índices)

Apartado 3:

Escribe cada una de las sentencias en SQL que se piden a continuación haciendo uso del lenguaje DDL y DCL de SQL:

1. Crea una vista de forma que sólo se muestren los nombres y apellidos de los profesores cuya fecha de nacimiento está comprendida entre ' 1980-01-01' y '2000-01-01'
2. Crea otra vista de forma que se muestren todos los títulos de los cursos, junto con su descripción y número de horas ordenados ascendentemente por el título del curso.
3. Crea un usuario llamado admin_bd03, con la contraseña "2023" para que se pueda conectar desde cualquier equipo.
4. Conceder los privilegios al usuario creado en el apartado anterior para que pueda actualizar datos en la tabla SESION.

SOLUCIÓN:

```
/* APARTADO 1 */
-- eliminamos si existe la BD
drop database if exists cursos_bd03;
-- creamos la base de datos
create database cursos_bd03;
-- seleccionamos nuestra BD
use cursos_bd03;
-- tabla CURSO
create table curso (
   codCurso char(4) PRIMARY KEY, -- PK estará formado por 4 caracteres
   titulo varchar(20) UNIQUE NOT NULL, -- no podemos repetir el título
   descripcion TEXT null,
   numHoras SMALLINT unsigned DEFAULT '0'
);
-- tabla UNIDAD
create table unidad (
          codCurso char(4), -- debe estar formado por 4 caracteres al igual que en la tabla curso
   numero tinyint unsigned,
   titulo varchar(40) not null, -- no puede admitir valores nulos.
   numHoras SMALLINT unsigned NOT NULL default (0), -- por defecto 0
   PRIMARY KEY (codCurso, numero), -- PK debe ponerse aparte
   CONSTRAINT fk_curso_unidad FOREIGN KEY (codCurso) REFERENCES curso (codCurso) -- FK codCurso
   ON UPDATE CASCADE -- si se actualiza codCurso debe actualizarse en esta tabla
```

```
);


-- tabla VIDEOCONFERENCIA
create table videoconferencia (
        fecha TIMESTAMP PRIMARY KEY check (fecha > '2000-01-01'), -- PK, se pide restricción > 1/1/2000
   contenido TEXT not null, -- no admite valores nulos
   codCurso char(4) NOT NULL, -- debe ser del mismo tipo que la tabla unidades
   numUnidad tinyint unsigned not null, -- debe ser del mismo tipo que la tabla unidades
   CONSTRAINT unidad_videoc_fk FOREIGN KEY (codCurso, numUnidad) REFERENCES unidad (codCurso, numero)
                ON UPDATE CASCADE ON DELETE CASCADE -- es una única FK formada por dos campos
);


/* APARTADO 2 */
-- renombramos la tabla videoconferencias
RENAME TABLE videoconferencia to sesion;
-- otra forma: ALTER TABLE videoconferencia RENAME sesion;


-- añadimos el campo tipo a la tabla sesion
alter table sesion add column tipo enum ('videoconferencia','sesion presencial', 'cuestionario') default ('videoconferencia'); -- no piden valor por defecto pero lo he incluido a modo de ejemplo


-- tabla profesor
create table profesor (
        codProfesor smallint NOT NULL PRIMARY KEY, -- PK
   dni char(9) not null UNIQUE,  -- aunque no se indica expresamente en la tarea, conviene ponerlo con valor único
   nombre varchar(20) NOT NULL,
   apellidos varchar(40) NOT NULL,
   telefono char(9) NULL,
   email varchar(30) NULL,
```

```
   fechaNac date NULL
);


-- añadir dos campos a la tabla curso y FK
alter table curso
        add column (fechaInicio date, fechaFin date, codProfesor smallint NOT NULL),
   add constraint profesor_curso_fk foreign key (codProfesor) references profesor (codProfesor)
;
   -- no lo voy a tener en cuenta pero se podría añadir
   -- ON UPDATE CASCADE;


-- creamos Indice
create index index_prof_apell_nomb ON profesor (apellidos, nombre);
-- otra forma:
ALTER TABLE profesor
        ADD INDEX idx_NombreCompletoProfesor(apellidos, nombre);


/* APARTADO 3*/
-- creación de la primera vista
create view profVista as select nombre, apellidos from profesor where fechaNac BETWEEN '1980-01-01' and '2000-01-01';


-- eliminación de la vista anterior y creación de otra vista.
drop view profVista;
create view cursosOrdenados as select titulo, descripcion, numHoras from curso order by titulo asc;


-- creación del usuario
create user IF NOT EXISTS admin_bd03@'%' identified by '2023'; -- podemos prescindir de %
-- permisos al usuario
grant UPDATE on cursos_bd03.sesion to admin_bd03@'%'; -- no es necesario poner el alias de la BD ni el %
```

Creación de Tablas en MySQL: EXAMEN 2122 Feb

```
/* EXAMEN 2022 Feb */

/* EJERCICIO 1 */

-- Eliminamos la BD si ya existe
DROP DATABASE IF EXISTS feb_2022;

-- Creamos la BD
CREATE DATABASE feb_2022;

-- Seleccionamos y usamos la BD
USE feb_2022;

-- Creamos las tablas
CREATE TABLE museo (
        codigo CHAR(4) PRIMARY KEY,
   nombre VARCHAR (50) UNIQUE NOT NULL,
   descripcion TEXT,
   superficie FLOAT CHECK (superficie > 0),
   fecha_creacion TIMESTAMP NOT NULL DEFAULT current_timestamp,
   estilo VARCHAR (50)
);

CREATE TABLE sala (
        numero INT,
   codigo_museo CHAR(4),
   visitable BOOLEAN,
```

```
   CONSTRAINT SA_PK PRIMARY KEY (numero, codigo_museo),
   CONSTRAINT SA_COD_FK FOREIGN KEY (codigo_museo) REFERENCES MUSEO(codigo) ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE autor (
        dni CHAR(9) PRIMARY KEY,
   nombre VARCHAR(100),
   biografia TEXT,
   anyo_nacimiento YEAR
);


CREATE TABLE pieza (
        referencia CHAR(8) PRIMARY KEY,
   titulo VARCHAR (100),
   genero enum ("clásico", "moderno", "contemporáneo"),
   fecha_hora_creacion DATETIME,
   dni_autor CHAR(9),
   CONSTRAINT PI_DNI_FK FOREIGN KEY (dni_autor) REFERENCES autor(dni) ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE expone(
        referencia_pieza CHAR(8),
   numero_sala INT,
   codigo_museo CHAR(4),
   CONSTRAINT EXP_PK PRIMARY KEY (referencia_pieza, numero_sala, codigo_museo),
   CONSTRAINT EXP_REF_FK FOREIGN KEY (referencia_pieza) REFERENCES pieza(referencia) ON UPDATE CASCADE ON DELETE CASCADE,
   CONSTRAINT EXP_NUM_FK FOREIGN KEY (numero_sala) REFERENCES sala(numero) ON UPDATE CASCADE ON DELETE CASCADE
);
```

```
-- Crear un índice para el título de pieza
CREATE INDEX Indice_Piezas ON pieza(titulo);

-- Crear un usuario con el mismo nombre y contraseña: visitante
CREATE USER visitante identified by "visitante";

-- Asigna permiso de consulta sobre la tabla SALA al usuario visitante
GRANT SELECT ON sala TO visitante;

-- Modifica en la tabla MUSEO el nombre del campo "Estilo" por "Tipo"
ALTER TABLE museo RENAME COLUMN estilo TO tipo;
```