UT_4 SQL. Ejercicios de creación de tablas

1. Crea la base de datos para una colección de películas:

GENEROS (Genero)
PELICULAS (Código, Titulo, Director, Año, Caratula, Genero, Director)
DIRECTORES (Nombre, Lugar de nacimiento, Fecha nacimiento)
ACTORES (Nombre, Lugar de nacimiento, Fecha de nacimiento, Sexo, Foto)
ACTORES-PELICULAS (Película, Actor, Foto, Año)

Debe elegirse el tipo de dato apropiado, además debe cumplirse que:

- Las **claves primarias** están **subrayadas**
- Existe una relación entre las tablas **GENEROS** y **PELICULAS**, por el campo Género.
- Existe una relación entre las tablas **DIRECTORES** y **PELICULAS**, siendo el campo **Director**, el **nombre del director**, además debe ocurrir que al **borrar un director de la tablas de directores deben borrarse las películas de dicho director.**
- Existe una relación entre las tablas **ACTORES-PELICULAS** y **PELICULAS**, siendo el campo Película el código de la Película, además al **borrar una película de la tabla PELICULAS debe borrarse también de la tabla ACTORES-PELICULAS automáticamente**.
- Existe una relación entre las tablas **ACTORES-PELICULAS** y **ACTORES**, siendo **el campo Actor el nombre del actor**.
- Los campos **Fotos**, contienen **la ruta del archivo**, por lo tanto son **VARCHAR (255).**
- El campo **Año** de la tabla ACTORES-PELICULAS debe llamarse **Anyo**.
- El **Sexo** del actor, es un código de una Letra que debe ser **M o V**, no aceptándose otro valor.

Primera Tabla Creada:

```
create table GENEROS(
GENERO varchar2(25),
constraint genero_generos_pk primary key(GENERO));
```

Segunda Tabla Creada:

```
create table DIRECTORES (
nombre varchar2(20),
lugar_de_nacimiento varchar2(25),
fecha_nacimiento date default sysdate,
constraint directores_nombre_pk primary key(nombre) );
```

Tercera Tabla creada:

```
create table PELICULAS (
codigo numeric(10) primary key,
titulo varchar2(30),
director varchar2(20) not null references DIRECTORES(nombre),
anyo char(4),
caratula varchar2(255),
generopeli varchar2(25),
constraint peliculas_genero_fk foreign key(generopeli) references GENEROS(genero));
```

Cuarta Tabla creada:

```
create table ACTORES (
        nombre varchar2(20),
        lugar_de_nacimiento varchar2(25),
        fecha_de_nacimiento date default sysdate,
        sexo char(1) constraint actores_sexo_ck check(sexo in('M','F')),
        foto varchar2(255) not null,
        constraint nomb_actores_pk primary key(nombre) );
```

Quinta Tabla creada: Antes de crearla añadimos una restriccion sobre el capo titulo para que sea UNIQUE en la tabla PELICULAS

alter table PELICULAS add UNIQUE(titulo);

CREAMOS la TABLA 5

```
create table ACTORES_PELICULAS(
pelicula varchar2(30),
actor varchar2(20) not null,
foto VARCHAR2(255) NOT NULL,
anyo CHAR(4) CONSTRAINT ANYO_ACTPELI_CK CHECK(anyo>'2000'),
constraint peli_act_fk foreign key(pelicula) references PELICULAS(titulo),
constraint act_peli_fk foreign key(pelicula) references ACTORES(nombre),
constraint peliact_actpeli_pk primary key(pelicula,actor) );
```

2. Crea una base de datos para una biblioteca:

LIBROS (ISBN, Titulo, Autor, Editorial, AñoEdición)
AUTORES (Nombre, Nacionalidad)
EDITORIAL (Nombre, Dirección)
SOCIOS (NúmeroSocio, Nombre, Dirección, Tfno, DNI)
PRESTAMOS (Codigo, Fecha, Socio, Libro)

Plantea las posibles relaciones entre las tablas.
Plantea otras restricciones posibles.

NOTA:
1. Un curso online SQL practico
2.