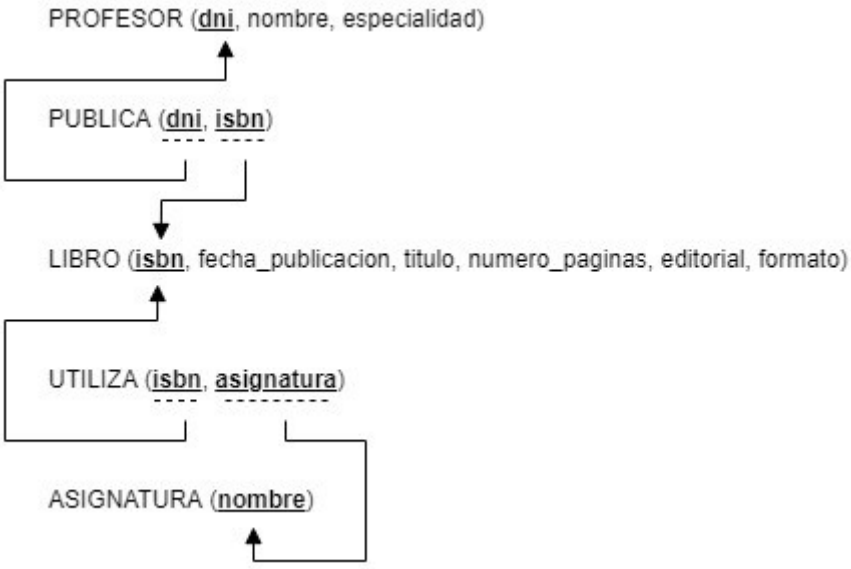
# CREACIÓN DE MODELO FÍSICO UTILIZANDO MARIADB

A partir del siguiente modelo relacional vamos a crear el modelo físico utilizando MariaDB.

La tarea se puede realizar tanto en la terminal de PowerShell como con Workbench MySQL.




1. **Crear el espacio para la base de datos y activar su uso.**

La sentencia para crear una base de datos es:

```
CREATE [OR REPLACE] {DATABASE | SCHEMA} [IF NOT EXISTS] db_name
    [create_specification] ...

create_specification:
    [DEFAULT] CHARACTER SET [=] charset_name
  | [DEFAULT] COLLATE [=] collation_name
  | COMMENT [=] 'comment'
```

Analizando la sintaxis vemos que CREATE DATABASE y CREATE SCHEMA son sinónimos. También que aún siendo opcional el parámetro, no se puede usar simultáneamente REPLACE e IF NOT EXISTS.

La opción REPLACE realiza internamente la eliminación completa de la base de datos en caso de existir para inmediatamente crearla. Es decir, borra todo rastro de la base de datos.

La opción IF NOT EXISTS crea la base de datos solo si no existe previamente. Si no se utiliza este parámetro y la base de datos existiera, la sentencia arrojaría un warning (advertencia) en lugar de un error.

Un error informa que algo se hizo realmente mal, describe el problema y detiene el proceso. Un warning no lo detiene, pero advierte que algo sucedió que no se esperaba pero que no es lo suficientemente crítico como para terminar el proceso.

```
CREATE DATABASE IF NOT EXISTS publicacion CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish_ci;
USE publicacion;
```

Con estas sentencias creamos la base de datos publicación en caso de no existir. Como vamos a usar el español, utilizamos el juego de caracteres utf8mb4 y como reglas de comparación utf8mb4_spanish_ci. Esto último aplica las reglas del español moderno y no se distingue entre mayúsculas y minúsculas.

La sentencia USE nos lleva hasta este nuevo espacio de trabajo y mientras que no cambiemos de espacio, todas las sentencias se aplicarán a la base de datos publicación

## 2. Crear las tablas

Para crear una tabla se puede optar por alguna de las siguientes sentencias:

```
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    (create_definition,...) [table_options    ]... [partition_options]
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    [(create_definition,...)] [table_options    ]... [partition_options]
    select_statement
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
   { LIKE old_table_name | (LIKE old_table_name) }

select_statement:
    [IGNORE | REPLACE] [AS] SELECT ...   (Some legal select statement)
```

```
create_definition:
  { col_name column_definition | index_definition | period_definition | CHECK (expr) }

column_definition:
  data_type
    [NOT NULL | NULL] [DEFAULT default_value | (expression)]
    [ON UPDATE [NOW | CURRENT_TIMESTAMP] [(precision)]]
    [AUTO_INCREMENT] [ZEROFILL] [UNIQUE [KEY] | [PRIMARY] KEY]
    [INVISIBLE] [{WITH|WITHOUT} SYSTEM VERSIONING]
    [COMMENT 'string'] [REF_SYSTEM_ID = value]
    [reference_definition]
  | data_type [GENERATED ALWAYS]
  AS { { ROW {START|END} } | { (expression) [VIRTUAL | PERSISTENT | STORED] } }
      [UNIQUE [KEY]] [COMMENT 'string']

constraint_definition:
   CONSTRAINT [constraint_name] CHECK (expression)
```

A la vista de la sintaxis, la creación de una tabla necesita como mínimo el nombre de la tabla y los detalles de las columnas.

Para saber más sobre las diferentes opciones consulta la documentación de MariaDB (https://mariadb.com/kb/en/create-table/).

Al igual que otros SGBDs, MaríaDB no crea una clave ajena (foránea) si la tabla y la clave primaria a la que referencian no existe. Por lo tanto hay dos opciones:

- Crear las tablas comenzando por las que no referencian a nadie seguidas de las que sólo referencian a las ya creadas hasta completar el esquema.
- Creamos todas las tablas en el orden que se quiera sin incluir las referencias a las claves ajenas y posteriormente las añadimos utilizando la sentencia ALTER TABLE.

Utilizaremos el segundo método.

```
CREATE OR REPLACE TABLE profesor(
dni             CHAR(9)         NOT NULL    PRIMARY KEY,
nombre          VARCHAR(100)    NOT NULL,
especialidad    VARCHAR(50)     NOT NULL
);

CREATE OR REPLACE TABLE publica(
dni             CHAR(9)         NOT NULL,
isbn            CHAR(13)        NOT NULL,
CONSTRAINT PRIMARY KEY (dni,isbn)
);

CREATE OR REPLACE TABLE libro(
isbn            CHAR(13)        NOT NULL    PRIMARY KEY,
fchPublicacion  DATE            NOT NULL,
titulo          VARCHAR(150)    NOT NULL,
numPaginas      INTEGER         NOT NULL,
editorial       VARCHAR(100)    NOT NULL,
formato         VARCHAR(50)     NOT NULL
);

CREATE OR REPLACE TABLE utiliza(
isbn            CHAR(13)        NOT NULL,
asignatura      VARCHAR(50)     NOT NULL,
CONSTRAINT PRIMARY KEY (isbn,asignatura)
);

CREATE OR REPLACE TABLE asignatura(
nombre          VARCHAR(50)     NOT NULL    PRIMARY KEY
);
```

Con estas sentencias hemos creado todas las tablas pero sin establecer las claves ajenas, aunque las columnas necesarias están ahí.

Con respecto a los tipos de datos, comentar que:

- Tanto dni como isbn se han considerado texto de longitud fija (9 caracteres para dni y 13 para isbn) porque se quieren conservar los ceros a la izquierda en caso de que se introduzcan.
- Los campos de texto restante se han considerado de longitud variable y se ha estimado su tamaño.

3. **Crear las referencias de claves ajenas** (si no se hizo en el paso anterior)

Con la sentencia ALTER TABLE modificaremos la estructura de las tablas para establecer las relaciones entre las tablas mediante la definición de las claves foráneas.

La sintaxis de este comando incluye numerosas opciones por lo que debes consultarla con más detalle en la documentación de MariaDB (https://mariadb.com/kb/en/alter-table/).

En nuestro ejercicio utilizaremos la opción que permite añadir (ADD) una restricción (CONSTRAINT) de clave ajena (FOREIGN KEY).

```
ALTER TABLE publica ADD CONSTRAINT pub_dni_FK  FOREIGN KEY (dni)  REFERENCES profesor(dni)
ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE publica ADD CONSTRAINT pub_isbn_FK FOREIGN KEY (isbn) REFERENCES libro(isbn)
ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE utiliza ADD CONSTRAINT FOREIGN KEY (isbn)       REFERENCES libro(isbn)
ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE utiliza ADD CONSTRAINT FOREIGN KEY (asignatura) REFERENCES asignatura(nombre)
ON UPDATE CASCADE ON DELETE CASCADE;
```

La palabra reservada REFERENCES indica cuál es la tabla padre y el campo de ésta en la que se debe comprobar la integridad referencial.

También se pueden definir las acciones que se deben llevar a cabo en caso de borrado (DELETE) o actualización (UPDATE) en la tabla padre del campo referenciado.

En nuestro caso, como los campos son a su vez clave primaria de la tabla a la que pertenecen no hemos permitido que se queden nulos para la acción de borrado.

Por otro lado, observa que a la restricción se le puede asignar un nombre. Este se utiliza en los mensajes de error y debe ser único en la base de datos.

La secuencia completa para generar esta bases de datos es pues:

```
CREATE DATABASE IF NOT EXISTS publicacion CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish_ci;
USE publicacion;
CREATE OR REPLACE TABLE profesor(
dni             CHAR(9)         NOT NULL    PRIMARY KEY,
nombre          VARCHAR(100)    NOT NULL,
especialidad    VARCHAR(50)     NOT NULL
);
CREATE OR REPLACE TABLE publica(
dni             CHAR(9)         NOT NULL,
isbn            CHAR(13)        NOT NULL,
CONSTRAINT PRIMARY KEY (dni,isbn)
);
CREATE OR REPLACE TABLE libro(
isbn            CHAR(13)        NOT NULL    PRIMARY KEY,
fchPublicacion  DATE            NOT NULL,
titulo          VARCHAR(150)    NOT NULL,
numPaginas      INTEGER         NOT NULL,
editorial       VARCHAR(100)    NOT NULL,
formato         VARCHAR(50)     NOT NULL
);
CREATE OR REPLACE TABLE utiliza(
isbn            CHAR(13)        NOT NULL,
asignatura      VARCHAR(50)     NOT NULL,
CONSTRAINT PRIMARY KEY (isbn,asignatura)
);
CREATE OR REPLACE TABLE asignatura(
nombre          VARCHAR(50)     NOT NULL    PRIMARY KEY
);
ALTER TABLE publica ADD CONSTRAINT pub_dni_FK  FOREIGN KEY (dni)  REFERENCES profesor(dni)
ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE publica ADD CONSTRAINT pub_isbn_FK FOREIGN KEY (isbn) REFERENCES libro(isbn)
ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE utiliza ADD CONSTRAINT FOREIGN KEY (isbn)       REFERENCES libro(isbn)
ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE utiliza ADD CONSTRAINT FOREIGN KEY (asignatura) REFERENCES asignatura(nombre)
ON UPDATE CASCADE ON DELETE CASCADE;
```