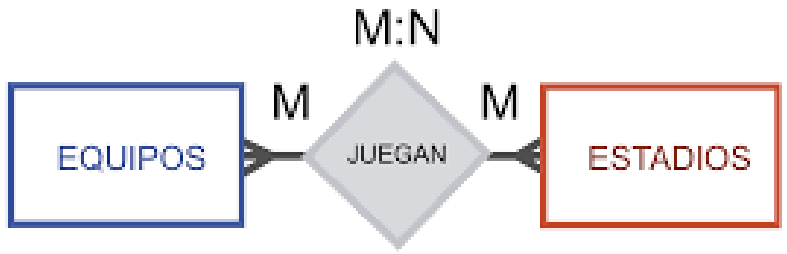
Equipo: GO

SQL-EJEMPLO DE RELACION MUCHOS A MUCHOS



INTEGRANTES:
SOLIS ROSAS, ROMARIO GONSALOS.
DE LA CRUZ MAGALLANES, SANDRA

Equipo: GO

INTRODUCCION:

Una relación de muchos a muchos se da cuando uno o más elementos de una tabla pueden estar relacionados con uno o más elementos de otra tabla. Por ejemplo: un actor puede participar en varias películas, y una película puede tener varios actores.

Este tipo de relación no puede representarse directamente entre dos tablas, por lo que se necesita una tabla intermedia que almacene las claves foráneas de ambas tablas para establecer la conexión.

En SQL, no es posible crear una relación muchos a muchos directamente porque las bases de datos relacionales están diseñadas para trabajar con claves primarias y foráneas, y una columna no puede contener múltiples valores si se quiere mantener la normalización.

- Resuelva esta relación **M:M**

Equipo: GO

Ejemplo:

Las Relación de muchos a muchos podemos Encontrarlas mediante diversos casos encontrados o analizados, por ejemplos:

- Estudiante y Cursos.
- Productos y Pedidos.
- Autores y Libros
- Actores y Películas

Podríamos implementar la relación de muchos a muchos en SQL tomando como ejemplo a actores y películas:
se crean tres tablas:

- Una tabla para **actores**
- Una tabla para **películas**
- Una tabla intermedia llamada **actores_peliculas,** que contiene las claves foráneas de ambas tablas (**id_autor y id_pelicula**), además de una clave primaria compuesta.

Esta tabla intermedia permite enlazar múltiples autores con múltiples películas, manteniendo la estructura normalizada y evitando duplicaciones.

Equipo: GO

CODIGO SQL.

```sql
CREATE DATABASE Colegio;
Use Colegio;

CREATE TABLE estudiantes (
    id_estudiante INT PRIMARY KEY,
    nombre        VARCHAR(100),
    DNI           CHAR(8),
    Fecha_Nac     DATE
)ENGINE = INNODB;
CREATE TABLE profesores (
    id_profesor INT PRIMARY KEY,
    nombre      VARCHAR(100),
    DNI         CHAR(8),
    Direccion   VARCHAR(100)
)ENGINE = INNODB;
CREATE TABLE cursos (
    id_curso INT PRIMARY KEY,
    nombre VARCHAR(100)
)ENGINE = INNODB;

CREATE TABLE curso_profesor (
    id_curso INT,
    id_profesor INT,
    PRIMARY KEY (id_curso, id_profesor),  -- Clave primaria compuesta
    CONSTRAINT id_curso_fk FOREIGN KEY (id_curso) REFERENCES cursos(id_curso),
    CONSTRAINT id_profesor_fk FOREIGN KEY (id_profesor) REFERENCES profesores(id_profesor)
)ENGINE = INNODB;
-- Tabla intermedia para relación muchos a muchos entre estudiantes y cursos
CREATE TABLE inscripciones (
    id_estudiante INT,
    id_curso INT,
    nota DECIMAL(4,2),
    PRIMARY KEY (id_estudiante, id_curso),
    CONSTRAINT id_estudiantes_fk FOREIGN KEY (id_estudiante) REFERENCES estudiantes(id_estudiante),
    CONSTRAINT id_cursos_fk FOREIGN KEY (id_curso) REFERENCES cursos(id_curso)
)ENGINE = INNODB;
```

Equipo: GO

```sql
INSERT INTO estudiantes (id_estudiante, nombre, DNI, Fecha_Nac) V
(1, 'Yoselin Martines', '12345678', '2004-05-12'),
(2, 'Maria Magallanes', '87654321', '2003-09-30'),
(3, 'Luis Huasasquiche', '11223344', '2005-01-20');
INSERT INTO profesores (id_profesor, nombre, DNI, Direccion) VALU
(1, 'Juan Sanches', '22221111', 'Av. Central 123'),
(2, 'Yulisa Napa', '33332222', 'Calle Lima 456'),
(3, 'Wuillians Fajardo', '33332222', 'Calle San Juan 456');
INSERT INTO cursos (id_curso, nombre) VALUES
(101, 'Matemáticas'),
(102, 'Historia'),
(103, 'Programación');
INSERT INTO curso_profesor (id_curso, id_profesor) VALUES
(101, 1),    -- Juan Pérez da Matemáticas
(102, 2),    -- María López da Historia
(103, 1),    -- Juan Pérez da Programación
(103, 3);    -- María López Programación
INSERT INTO inscripciones (id_estudiante, id_curso, nota) VALUES
(1, 101, 17.5),  -- Yoselin Martines en Matemáticas
(2, 102, 18.0),  -- Maria Magallanes en Historia
(3, 101, 14.0),  -- Luis Huasasquiche en Matemáticas
(1, 103, 15.5),  -- Yoselin Martines en Programación
(2, 103, 19.0);  -- Maria Magallanes en Programación
```

```sql
SELECT
    e.nombre AS estudiante,
    c.nombre AS curso,
    p.nombre AS profesor,
    i.nota
FROM inscripciones i
INNER JOIN estudiantes e ON i.id_estudiante = e.id_estudiante
INNER JOIN cursos c ON i.id_curso = c.id_curso
INNER JOIN curso_profesor cp ON c.id_curso = cp.id_curso
INNER JOIN profesores p ON cp.id_profesor = p.id_profesor
ORDER BY e.nombre, c.nombre;
```