Mantén eso en mente...

- Las palabras clave SQL NO distinguen entre mayúsculas y minúsculas: select es lo mismo que SELECT

En este tutorial escribiremos todas las palabras clave SQL en mayúsculas.

Herramienta para trabajar de manera online

https://www.mycompiler.io/es/new/sql

¿

Punto y coma después de declaraciones SQL?

Algunos sistemas de bases de datos requieren un punto y coma al final de cada declaración SQL.

El punto y coma es la forma estándar de separar cada declaración SQL en sistemas de bases de datos que permiten ejecutar más de una declaración SQL en la misma llamada al servidor.

En este tutorial, usaremos punto y coma al final de cada declaración SQL.

Algunos de los comandos SQL más importantes

- SELECT- extrae datos de una base de datos
- UPDATE- actualiza datos en una base de datos
- DELETE- elimina datos de una base de datos
- INSERT INTO- inserta nuevos datos en una base de datos
- CREATE DATABASE- crea una nueva base de datos
- ALTER DATABASE- modifica una base de datos
- CREATE TABLE- crea una nueva tabla
- ALTER TABLE- modifica una tabla
- DROP TABLE- elimina una tabla
- CREATE INDEX- crea un índice (clave de búsqueda)
- DROP INDEX- elimina un índice

EJEMPLO DE LA CREACION DE UNA TABLA Y DE INSERTAR LOS DATOS

```
create table usuarios (
  nombre varchar(30),
  clave varchar(10)
);

insert into usuarios(nombre,clave) values ('MarioPerez','Marito');
insert into usuarios(nombre,clave) values ('MariaGarcia','Mary');
insert into usuarios(nombre,clave) values
('DiegoRodriguez','z8080');

select nombre,clave from usuarios;
SELECT * FROM usuarios;
```

Ejercicio completo de la creación de dos tablas y su tabla relacional.

Entidades:

- **Estudiante**
  - **Atributos:**
    - **ID (identificador único)**
    - **Nombre**
    - **Edad**
- **Curso**
  - **Atributos:**
    - **ID (identificador único)**
    - **Nombre**
    - **Duración**

Relación:

- **Inscripción**
  - **Esta relación vincula estudiantes con cursos.**
  - **Un estudiante puede estar inscrito en varios cursos.**
  - **Un curso puede tener varios estudiantes inscritos.**

```
-- Creación de la tabla Estudiante
CREATE TABLE Estudiante (
    ID INT PRIMARY KEY,
    Nombre VARCHAR(50),
    Edad INT
);
-- Creación de la tabla Curso
CREATE TABLE Curso (
    ID INT PRIMARY KEY,
    Nombre VARCHAR(100),
    Duracion INT -- En horas, por ejemplo
);
-- Creación de la tabla Inscripcion
CREATE TABLE Inscripcion (
    EstudianteID INT,
    CursoID INT,
    FOREIGN KEY (EstudianteID) REFERENCES Estudiante(ID),
    FOREIGN KEY (CursoID) REFERENCES Curso(ID),
    PRIMARY KEY (EstudianteID, CursoID)
);
```

Insertar datos en las tablas.

```
-- Insertar datos en la tabla Estudiante
INSERT INTO Estudiante (ID, Nombre, Edad) VALUES
```

```
(1, 'Juan', 20),

(2, 'María', 22),

(3, 'Pedro', 21);

-- Insertar datos en la tabla Curso

INSERT INTO Curso (ID, Nombre, Duracion) VALUES

(101, 'Introducción a SQL', 30),

(102, 'Consultas Avanzadas', 40),

(103, 'Administración de Bases de Datos', 50);

-- Insertar datos en la tabla Inscripcion

INSERT INTO Inscripcion (EstudianteID, CursoID) VALUES

(1, 101), -- Juan inscrito en Introducción a SQL

(2, 102), -- María inscrita en Consultas Avanzadas

(3, 103), -- Pedro inscrito en Administración de Bases de Datos

(1, 102); -- Juan inscrito también en Consultas Avanzadas
```

Consultas básicas

Consultar todos los estudiantes:

```
SELECT * FROM Estudiante;
```

Consultar todos los cursos:

```
SELECT * FROM Curso;
```

Consultar todas las inscripciones:

```
SELECT * FROM Inscripcion;
```

Consultar el nombre y la edad de todos los estudiantes:

```
SELECT Nombre, Edad FROM Estudiante;
```

Consultar el nombre y la duración de todos los cursos:

```
SELECT Nombre, Duracion FROM Curso;
```

Consultar los estudiantes inscritos en un curso específico (por ejemplo, "Introducción a SQL"):

```
SELECT e.Nombre FROM Estudiante e INNER JOIN Inscripcion i ON e.ID = i.EstudianteID INNER JOIN Curso c ON i.CursoID = c.ID WHERE c.Nombre = 'Introducción a SQL';
```

Consultar los cursos en los que está inscrito un estudiante específico (por ejemplo, "Juan"):

```
SELECT c.Nombre FROM Curso c INNER JOIN Inscripcion i ON c.ID = i.CursoID INNER JOIN Estudiante e ON i.EstudianteID = e.ID WHERE e.Nombre = 'Juan';
```

Taller de Diseño de Base de Datos y SQL

Parte 1: Diseño del Modelo Entidad-Relación (MER)

Escenario: Imagina que estás diseñando una base de datos para una biblioteca. La biblioteca tiene libros, cada libro tiene un título, un autor y un género. Los socios de la biblioteca pueden tomar prestados libros, y cada préstamo tiene una fecha de inicio y una fecha de devolución. Los socios de la biblioteca están registrados en el sistema con su nombre, dirección y número de socio.

Ejercicio: Diseña un Modelo Entidad-Relación (MER) para este escenario. Identifica las entidades, atributos y relaciones necesarias, así como cualquier restricción de integridad que consideres relevante.

Parte 2: Traducción del MER a SQL

Ejercicio: Usando el MER que diseñaste en la Parte 1, tradúcelo a código SQL. Crea las tablas necesarias para implementar el MER en una base de datos relacional y proporciona las sentencias SQL para insertar datos de ejemplo en estas tablas.

Instrucciones:

1. Diseña tu MER para el escenario proporcionado.
2. Traduce tu MER a código SQL, creando las tablas necesarias.
3. Inserta datos de ejemplo en estas tablas.
4. Copia y pega tanto el diseño del MER como el código SQL en un documento de Word, junto con los comandos de inserción de datos y sus respectivos resultados