**Comandos básicos de SQL**

**Obtener información sobre la base de datos Mysql:**

*show databases;*  – Listar todas las bases de datos.

*use [database];* – Usar esa base de datos.

*show tables;*  – Listar todas las tablas de una base de datos.

*show table status;* – Muestra información sobre las tablas de la base de datos.

*describe [table]*; – Muestra la estructura de una tabla de la base de datos.

**Manejo de bases de datos Mysql:**

*drop table [table];* – Elimina la tabla, incluyendo registros y estructura.

*drop table if exists [table];* – Elimina la tabla de la base de datos, pero antes verifica que exista.

*truncate table [table];* – Elimina los registros, pero mantiene la estructura de la tabla.

*rename table [table] to [nuevo nombre de tabla];* – Renombrar una tabla de la base de datos.

**Crear Tabla**

El comando **CREATE TABLE** requiere que especifiques el nombre de la tabla y su encabezado, el cual es una lista de definiciones de campos separadas por coma. Ejemplo:

*CREATE TABLE empleados (*

*id\_empleado INT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,*

*curp VARCHAR(20) UNIQUE,*

*nombre VARCHAR(150),*

*email VARCHAR(100),*

*id\_depto INT (20)*

*fecha\_nacimiento DATE*

*FOREIGN KEY (id\_depto) REFERENCES departamentos (id\_depto)*

*);*

El nombre de la tabla es empleados. El encabezado de la tabla es todo lo que va entre paréntesis. Y cada línea dentro de los paréntesis corresponde a una definición de campo.

Además, nota que cada «definición de campo» está formada a su vez por el nombre del campo, tipo de dato y restricciones (en ese orden).

Por ejemplo, en la definición de campo *curp VARCHAR(20) UNIQUE:*

curp es el nombre del campo.

VARCHAR es el tipo de dato.

(20) es una restricción que indica el máximo de caracteres permitidos en el campo.

UNIQUE es una restricción que indica que solo un registro de ese campo podrá tener ese valor, es otras palabras, esta restricción no permite que el campo tenga valores duplicados.

La restricción "foreign key" tiene las cláusulas "on delete" y "on update" que son opcionales.

Estas cláusulas especifican cómo debe actuar MySQL frente a eliminaciones y modificaciones de las tablas referenciadas en la restricción.

Las opciones para estas cláusulas son las siguientes:

"no action": indica que si intentamos eliminar o actualizar un valor de la clave primaria de la tabla referenciada (TABLA2) que tengan referencia en la tabla principal (TABLA1), se genere un error y la acción no se realice; es la opción predeterminada.

"cascade": indica que si eliminamos o actualizamos un valor de la clave primaria en la tabla referenciada (TABLA2), los registros coincidentes en la tabla principal (TABLA1), también se eliminen o modifiquen; es decir, si eliminamos o modificamos un valor de campo definido con una restricción "primary key" o "unique", dicho cambio se extiende al valor de clave externa de la otra tabla (integridad referencial en cascada).

"set null": Establece con el valor null en el campo de la clave foránea.

"restrict": La acción "restrict" significa que la aplicación tiene prohibido eliminar (para ON DELETE RESTRICT) o modificar (para ON UPDATE RESTRICT) una clave principal cuando existe una o más claves secundarias asignadas a ella.

**Agregar campos nuevos**

Puedes agregar un nuevo campo usando el comando ALTER TABLE con la cláusula ADD COLUMN. Por ejemplo, vamos a agregar el campo ap\_pat a nuestra tabla empleados:

*ALTER TABLE empleados ADD COLUMN ap\_pat VARCHAR(100);*

El comando ALTER TABLE va seguido del nombre de la tabla, luego de la cláusula ADD COLUMN y por último la definición del campo, es decir, el nombre del campo seguido del tipo de dato y sus restricciones (así como cuando creaste la tabla).

Por defecto, los campos se agregarán al final, pero también puedes agregarlos en una posición específica, usando FIRST para añadirlos al inicio o AFTER para añadirlos después de otro campo.

Ejemplo:

ALTER TABLE empleados ADD COLUMN ap\_pat VARCHAR(100) AFTER nombre;

**Renombrar un campo**

Puedes cambiar el nombre de un campo usando el comando ALTER TABLE con la cláusula CHANGE COLUMN. Por ejemplo, vamos a renombrar el campo ap\_pat a ap\_paterno:

*ALTER TABLE* empleados *CHANGE COLUMN ap\_pat ap\_paterno VARCHAR(150);*

La cláusula CHANGE COLUMN te pide que especifiques el nombre actual del campo y luego el nuevo nombre que darás a ese campo, seguido del tipo de dato y las restricciones (en caso de tenerlas).

Aunque solo desees cambiar el nombre del campo, si usas la cláusula CHANGE COLUMN debes especificar su definición, ya que ésta también será modificada.

**Cambiar tipo de datos y/o restricciones**

El comando ALTER TABLE con la cláusula MODIFY COLUMN nos permite cambiar el tipo de dato y/o restricciones de un campo, solo basta con colocar su nombre y su nueva definición:

*ALTER TABLE empleados MODIFY COLUMN ap\_pat VARCHAR(100);*

*ALTER TABLE categorias MODIFY COLUMN codigo INT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,*

Aunque usando la cláusula CHANGE COLUMN también puedes lograr esto, la cláusula MODIFY COLUMN es la más recomendable si solo deseas cambiar la definición de un campo.

Para mover un campo de posición podemos hacerlo haciendo uso de AFTER

*ALTER TABLE empleados MODIFY COLUMN ap\_mat VARCHAR(100) AFTER ap\_pat;*

**Eliminar campos**

Para eliminar un campo debes usar el comando ALTER TABLE seguido del nombre de la tabla, luego la cláusula DROP COLUMN y por último el nombre del campo que deseas eliminar. Por ejemplo:

*ALTER TABLE empleados DROP COLUMN email;*

**Insertar datos en una tabla**

Para insertar datos en una tabla se usa el comando INSERT INTO. Por ejemplo, insertemos un departamento en la tabla *departamentos*:

*INSERT INTO departamentos (id\_depto, nombre)*

*VALUES ('D-1', 'División de estudios');*

El comando INSERT INTO va seguido del nombre de la tabla y su encabezado (lista de campos) entre paréntesis, luego se coloca la palabra reservada VALUES y, finalmente, la lista de valores a insertar encerrada entre paréntesis.

El orden de los campos y valores es muy importante. Asegúrate que cada valor a insertar corresponda (en el mismo orden) con el campo al que pertenece.

**Consultar datos de una tabla**

Para consultar los datos de las tablas se usa el comando SELECT. Por ejemplo, vamos a consultar la tabla empleado para obtener nuestro registro insertado:

*SELECT id, nombre, ap\_pat FROM empleados;*

El comando SELECT va seguido de la lista de campos de la tabla (separados por coma), luego se escribe la palabra reservada FROM, y por último, el nombre de la tabla que deseas consultar.

Puedes usar un comodín (\*) para consultar todos los campos de la tabla, sin embargo, es una buena práctica construir las consultas especificando los campos deseados.

También podemos usar la cláusula WHERE para filtrar los resultados que cumplan con una condición.

SELECT id, nombre FROM empleados WHERE ap\_pat = 'Gonzalez';

**Operador JOIN**

El operador JOIN nos permite enlazar dos tablas por medio de un campo en común, el cual puede ser un campo clave o no.

Como resultado, obtendremos todos los registros de ambas tablas donde el campo en común tenga el mismo valor.

Supongamos que queremos obtener todos los empleados que están asignados a un departamento:

*SELECT E.id\_empleado, E.nombre, E.ap\_pat, E.id\_depto, D.nombre AS departamento FROM empleados AS E JOIN departamentos AS D ON E.id\_depto = D.id\_depto;*

La cláusula AS nos permite asignar un «apodo» a los nombres de campos y tablas. Esto resulta muy útil para colocar un nombre alternativo a campos/tablas o evitar conflictos con nombres de campos repetidos.

El operador JOIN se puede usar con el comando SELECT después de la cláusula FROM. Va seguido del nombre de la tabla que se desea «enlazar», luego la cláusula ON, y por último se comparan los campos en común.

En otras palabras, la tabla empleados se enlaza con la tabla departamentos por medio de los campos id\_depto e id\_depto.

Si ejecutamos la siguiente consulta:

*SELECT \* FROM empleados AS E JOIN departamentos AS D ON E.id\_depto = D.id\_depto;*

La lista de campos de nuestro resultado estará ordenada con todos los campos de la tabla employees primero y luego con todos los campos de la tabla branches.

**Actualizar datos en una tabla**

La actualización de datos nos permite modificar los valores que han sido insertados dentro de los campos de nuestras tablas. Podemos lograr esto usando el comando UPDATE.

UPDATE departamentos SET nombre = 'Sistemas y Computacion ' WHERE id = ‘D1’;

El comando UPDATE va seguido del nombre de la tabla que deseas actualizar, luego la cláusula SET con los campos y su nuevo valor (separados por coma), y por último la cláusula WHERE, para aplicar la modificación de estos campos sólo en los registros que cumplan con las condiciones definidas.

Si no incluyes la cláusula WHERE se modificará el campo con el nuevo valor en TODOS los registros de la tabla.

**Eliminar datos en una tabla**

El comando DELETE nos permite eliminar registros de forma física en nuestras tablas, es decir, borrará el registro de la tabla completamente.

Por ejemplo, vamos a eliminar un depto en la tabla departamento:

*DELETE FROM departamentos WHERE id\_depto = ‘D1’;*

El comando DELETE va seguido de la cláusula FROM y el nombre de la tabla donde deseamos eliminar los registros, luego, se agrega la cláusula WHERE para aplicar esta operación solamente en los registros que cumplan con las condiciones definidas.

Si no incluyes la cláusula WHERE se eliminarán TODOS los registros de la tabla.

Verifica que los registros que deseas eliminar no estén relacionados con registros en otras tablas, de lo contrario, la instrucción podría fallar