*En esta sección, aprenderemos cómo manipular los datos en las tablas de una base de datos mediante SQL. La manipulación de datos es una función principal de SQL, y es importante tener cuidado al realizar cambios en las tablas para garantizar la integridad de los datos y evitar consecuencias no deseadas en otras tablas relacionadas mediante claves foráneas.*

*4. 1 Creando una base de datos y una tabla*

*Para crear una base de datos, se utiliza el comando CREATE DATABASE seguido del nombre de la base de datos que se quiere crear. La sintaxis utilizada es:*

| *CREATE DATABASE nombre\_de\_la\_base\_de\_datos;* |
| --- |

*Por ejemplo, si queremos crear una base de datos que se llame escuela, utilizaremos la siguiente instrucción:*

| *CREATE DATABASE escuela;* |
| --- |

*Para crear una tabla, se utiliza el comando CREATE TABLE seguido del nombre de la tabla y los nombres y tipos de datos de las columnas que se quieren crear. La sintaxis a utilizar es la siguiente:*

| *CREATE TABLE nombre\_de\_la\_tabla (  columna1 tipo\_de\_dato,  columna2 tipo\_de\_dato,  columna3 tipo\_de\_dato, ... );* |
| --- |

*Por ejemplo, para crear una tabla de estudiantes en nuestra base de datos "escuela" con las columnas "id", "nombre", "apellido", "fecha\_nacimiento" y “email”, utilizamos la siguiente sintaxis:*

| *CREATE TABLE estudiantes (  id INT PRIMARY KEY,  nombre VARCHAR(50),  apellido VARCHAR(50),  fecha\_nacimiento DATE,  email VARCHAR(50) );* |
| --- |

*Donde:*

* *CREATE TABLE es la sentencia que indica que vamos a crear una tabla.*
* *“estudiantes” es el nombre que le damos a la tabla.*
* *“id” INT, “nombre” VARCHAR(50), apellido VARCHAR(50), “fecha\_nacimiento” DATE, “email” VARCHAR(50)) son las columnas de la tabla y su tipo de dato correspondiente.*
* *En este caso, "id" es un número entero (INT), "nombre", “apellido” y “email” son cadenas de caracteres (VARCHAR) de hasta 50 caracteres, "fecha\_nacimiento" es una fecha.*

*Una vez creada la tabla, podemos insertar datos en ella utilizando la sentencia INSERT INTO.*

*Para eliminar una base de datos en SQL, se utiliza el comando DROP DATABASE, seguido del nombre de la base de datos que se desea eliminar. Por ejemplo:*

| *DROP DATABASE nombre\_de\_base\_de\_datos;* |
| --- |

*Es importante tener en cuenta que esta acción eliminará completamente la base de datos y todos sus datos, y no se puede deshacer. Por lo tanto, es recomendable hacer una copia de seguridad de la base de datos antes de eliminarla.*

*Para eliminar una tabla en SQL, se utiliza el comando DROP TABLE, seguido del nombre de la tabla que se desea eliminar. Por ejemplo:*

| *DROP TABLE nombre\_de\_la\_tabla* |
| --- |

*Al igual que con la eliminación de una base de datos, es importante tener en cuenta que esta acción eliminará completamente la tabla y todos sus datos, y no se puede deshacer. Por lo tanto, es recomendable hacer una copia de seguridad de la tabla antes de eliminarla.*

*4.1 Insertar datos en tablas*

*Para insertar datos en una tabla, utilizamos la sentencia INSERT INTO. Esta sentencia permite agregar nuevos datos a las tablas de la base de datos. La sintaxis básica de la sentencia INSERT es la siguiente:*

| *INSERT INTO nombre\_de\_tabla (columna1, columna2, columna3, ...) VALUES (valor1, valor2, valor3, ...)* |
| --- |

*Donde:*

* *“nombre\_de\_tabla”: Es el nombre de la tabla en la que se quiere insertar datos.*
* *“columna1”, “columna2”, “columna3”, ...: Son los nombres de las columnas de la tabla en la que se quieren insertar datos. Es opcional y solo se usa si se quieren especificar las columnas a las que se les van a insertar datos. Si se omiten, se asumirá que se insertarán datos en todas las columnas.*
* *“valor1”, “valor2”, “valor3”, ...: Son los valores que se quieren insertar en las columnas correspondientes.*

*Por ejemplo, para insertar un nuevo registro en una tabla llamada alumnos con los datos “id”, “nombre”, “apellido”, “fecha\_nacimiento” y “email”, se puede utilizar la siguiente sintaxis:*

| *INSERT INTO estudiantes (id, nombre, apellido, fecha\_nacimiento, email) VALUES (21, 'Javier, 'Jiménez, '1997-04-08', 'javier.jimenez@example.com');* |
| --- |

*Si se quiere insertar más de un registro a la vez, se puede utilizar la siguiente sintaxis:*

| *INSERT INTO nombre\_de\_tabla (columna1, columna2, columna3, ...) VALUES (valor1, valor2, valor3, ...),*  *(valor1, valor2, valor3, ...),*  *(valor1, valor2, valor3, ...);* |
| --- |

*Por ejemplo, para insertar tres nuevos registros en la tabla estudiantes, se puede utilizar la siguiente sintaxis:*

| *INSERT INTO estudiantes (id, nombre, apellido, fecha\_nacimiento, email) VALUES  (22, 'Juan', 'Pérez', '1999-03-20', 'juan.perez@example.com'),  (23, 'María', 'García', '1998-06-12', 'maria.garcia@example.com'),  (24, 'Carlos', 'Sánchez', '2000-01-01', 'carlos.sanchez@example.com'),* |
| --- |

*Es importante recordar que los valores deben estar en el mismo orden en que se especificaron las columnas. Si se omite alguna columna, se debe omitir también su valor correspondiente en la lista de valores.*

*Además, es posible insertar datos utilizando una subconsulta. Por ejemplo, si se quiere insertar datos en una tabla llamada compras que contenga el id de un cliente y el id de un producto comprado, se puede utilizar la siguiente sintaxis:*

| *INSERT INTO compras (id\_cliente, id\_producto) SELECT id\_cliente, id\_producto FROM clientes WHERE nombre = 'Juan Pérez';* |
| --- |

*En este caso, se insertarán en la tabla compras los id del cliente que tenga el nombre Juan Perez y el id de los productos que haya comprado. La subconsulta devuelve los valores que se quieren insertar.*

*En conclusión, la sentencia INSERT es una herramienta muy útil para agregar nuevos datos a una tabla. Es importante tener en cuenta la sintaxis y el orden de los valores para evitar errores al momento de insertar datos. Además, es posible utilizar subconsultas para insertar datos a partir de información de otras tablas.*

*4.3 Actualizar datos en tablas*

*Para actualizar datos en una tabla, utilizamos la sentencia UPDATE. p}Permite modificar información existente en una tabla. Podemos actualizar uno o varios campos en una o varias filas de una tabla, utilizando condiciones con la cláusula WHERE. La sintaxis básica para actualizar datos en una tabla en SQL es la siguiente:*

| *UPDATE nombre\_tabla SET columna1 = valor1, columna2 = valor2, ... WHERE condicion;* |
| --- |

*Donde:*

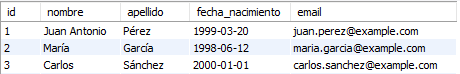
* *“nombre\_tabla”: es el nombre de la tabla que se desea actualizar.*
* *“columna1”, “columna2”, ... : son los nombres de las columnas que se desean actualizar.*
* *“valor1”, “valor2”, ... : son los valores nuevos que se desean asignar a las columnas.*
* *“condicion”: es una expresión que indica qué registros deben ser actualizados.*

*Es importante tener en cuenta que la cláusula WHERE es opcional, pero si no se utiliza, todos los registros de la tabla serán actualizados.*

*A continuación se presenta un ejemplo de cómo actualizar datos en una tabla de estudiantes:*

| *UPDATE estudiantes SET nombre = 'Juan Antonio' WHERE id = 1;* |
| --- |

*En este ejemplo, se está actualizando el nombre del alumno con ID 1. La expresión WHERE id = 1 indica que sólo se actualizará el registro correspondiente al empleado con ID 1. Si revisamos la tabla estudiantes nuevamente, podremos ver que el nombre del alumno ha cambiado:*

**

*También es posible actualizar datos utilizando subconsultas, por ejemplo:*

| *UPDATE calificaciones SET calificacion = calificacion - 0.2 WHERE curso\_id = 1;* |
| --- |

*En este ejemplo, se está actualizando la calificación de todos los alumnos en el curso de matemáticas (id = 1) disminuyendo dos décimas su calificación.*

*.*

*Es importante tener en cuenta que al actualizar datos en una tabla, se debe tener cuidado de no modificar accidentalmente información importante o de afectar la integridad de la base de datos. Por lo tanto, se recomienda siempre hacer una copia de seguridad antes de realizar cualquier actualización de datos masiva.*

*4.4 Eliminar datos de tablas*

*Para eliminar datos de una tabla, utilizamos la sentencia DELETE. Podemos eliminar una o varias filas de una tabla, utilizando condiciones con la cláusula WHERE. Es importante tener en cuenta que, al eliminar una fila, se pierde la información relacionada con ella, por lo que es esencial hacer una copia de seguridad antes de eliminar datos. La sintaxis básica de la sentencia DELETE es la siguiente:*

| *DELETE FROM table\_name WHERE condition;* |
| --- |

*Donde:*

* *DELETE FROM es la cláusula que indica que se desea eliminar datos de una tabla.*
* *table\_name es el nombre de la tabla de la que se eliminarán los datos.*
* *WHERE es una cláusula opcional que se utiliza para especificar una condición para la eliminación de datos. Si se omite la cláusula WHERE, se eliminarán todos los registros de la tabla.*

*Un ejemplo del uso de la sentencia DELETE es el siguiente.*

| *DELETE FROM estudiantes WHERE id = 21;* |
| --- |

*Esta sentencia eliminará el registro cuyo id sea igual a 21 en la tabla estudiantes.*

*Si se desea, en cambio, eliminar todos los registros de la tabla, se puede omitir la cláusula WHERE:*

| *DELETE FROM estudiantes;* |
| --- |

*Cabe mencionar que la sentencia DELETE es una operación destructiva y no se puede deshacer. Por lo tanto, es importante tener precaución al utilizarla y asegurarse de que se esté eliminando la información correcta.*

*En conclusión, hemos aprendido cómo manipular datos en SQL. Aprendimos cómo crear una tabla utilizando la sintaxis CREATE TABLE y definimos las columnas y sus tipos de datos. También vimos cómo insertar datos en una tabla utilizando la sintaxis INSERT INTO y cómo actualizar esos datos utilizando la sintaxis UPDATE. Además, aprendimos cómo eliminar datos utilizando la sintaxis DELETE FROM.*

*Es importante tener en cuenta que antes de realizar cualquier manipulación de datos, debemos asegurarnos de que la estructura de la tabla sea correcta y que los datos sean válidos. También es importante hacer una copia de seguridad de los datos antes de realizar cualquier modificación.*