MySQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto basado en SQL (Structured Query Language).

Para instalarlo y usarlo dentro del proyecto junto a express se usa:
-npm install express mysql2 body-parser bcrypt cors

Dentro del informe solo se verán las funciones principales dentro del mysql y para que sean identificadas dentro de los querys en el código

Comandos

SELECT

Sirve para consultar y mostrar datos de una o varias tablas.

```
-- Seleccionar todas las columnas SELECT * FROM
usuarios;

-- Seleccionar columnas e s p e c f i c a s SELECT nombre ,
correo FROM usuarios;
```

INSERT

Se utiliza para insertar nuevos registros en una tabla.

```
INSERT INTO usuarios ( nombre , correo , edad ) VALUES (' Juan ', ' juan@
email. com ', 25);
```

UPDATE

Permite actualizar valores de registros existentes.

```
UPDATE usuarios SET
edad = 26
WHERE  nombre  =  ' Juan ';
```

DELETE

Elimina registros de una tabla.

```
DELETE FROM usuarios WHERE
nombre = ' Juan ';
```

CREATE

Crea una base de datos o una tabla.

```
-- Crear base de datos CREATE
DATABASE empresa ;

-- Crear tabla
CREATE TABLE usuarios (
        id  INT  AUTO_INCREMENT  PRIMARY  KEY ,
        nombre VARCHAR (50) ,
        correo VARCHAR (100) , edad
        INT
);
```

ALTER

Permite modificar la estructura de una tabla.

```
-- Agregar nueva columna
ALTER TABLE usuarios ADD telefono VARCHAR (15);

-- Modificar tipo de              columna
ALTER TABLE usuarios               MODIFY  edad  SMALLINT ;

-- Eliminar columna
ALTER TABLE usuarios DROP telefono ;
```

DROP

Elimina bases de datos o tablas completas.

```
DROP DATABASE empresa ; DROP
TABLE usuarios;
```

Operadores y cl´ausulas u´tiles

WHERE

Se utiliza para filtrar registros en una consulta.

```
SELECT * FROM usuarios WHERE edad > 20;
```

ORDER BY

Ordena los resultados.

```
SELECT * FROM usuarios ORDER BY edad DESC ;
```

LIMIT

Restringe el número de resultados.

```
SELECT * FROM usuarios LIMIT 5;
```

LIKE

Permite buscar patrones en cadenas de texto.

```
SELECT * FROM usuarios WHERE nombre LIKE 'J %';
```

JOIN

Se utiliza para combinar registros de varias tablas relacionadas.

```
SELECT pedidos. id , usuarios. nombre FROM pedidos
JOIN usuarios ON pedidos. id_usuario = usuarios. id;
```

El operador *

El simbolo * se utiliza para seleccionar todas las columnas de una tabla.

```
SELECT * FROM usuarios;
```

EJ

```
const express = require("express");
const bodyParser = require("body-parser");
const mysql = require("mysql2");
const bcrypt = require("bcrypt");
const cors = require("cors");

const app = express();
app.use(cors());
app.use(bodyParser.json());

// Conexión a MySQL
const db = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "tu_password",
  database: "ejemplo_login",
});

db.connect((err) => {
  if (err) {
```

```
    console.error("Error conectando a MySQL:", err);
  } else {
    console.log("Conectado a MySQL ✅");
  }
});

// 🔹 Registro de usuario
app.post("/register", async (req, res) => {
  const { nombre, email, password } = req.body;

  // Encriptar la contraseña
  const hashedPassword = await bcrypt.hash(password, 10);

  db.query(
    "INSERT INTO usuarios (nombre, email, password) VALUES (?, ?, ?)",
    [nombre, email, hashedPassword],
    (err, result) => {
      if (err) {
        console.error(err);
        return res.status(500).json({ error: "Error registrando usuario"
});
      }
      res.json({ message: "Usuario registrado correctamente" });
    }
  );
});

// 🔹 Login de usuario
app.post("/login", (req, res) => {
  const { email, password } = req.body;

  db.query(
    "SELECT * FROM usuarios WHERE email = ?",
    [email],
    async (err, results) => {
      if (err) return res.status(500).json({ error: "Error en la base de
datos" });
      if (results.length === 0) return res.status(401).json({ error:
"Usuario no encontrado" });

      const user = results[0];

      // Verificar contraseña
      const match = await bcrypt.compare(password, user.password);
```

```
      if (!match) return res.status(401).json({ error: "Contraseña
incorrecta" });

      res.json({ message: "Inicio de sesión exitoso", user: { id: user.id,
nombre: user.nombre, email: user.email } });
    }
  );
});

// 🔹 Iniciar servidor
app.listen(3000, () => {
  console.log("Servidor corriendo en http://localhost:3000");
});
```

Este es un registro que se encarga de enviar un POST a mysql y guardar los datos del usuario registrado con su respectiva seguridad