

Day 26

Assignment 2: SQL Queries using JDBC Create a table 'User' with a following schema 'User ID' and 'Password' stored as hash format (note you have research on how to generate hash from a string), accept "User ID" and "Password" as input and check in the table if they match to confirm whether user access is allowed or not as java code and explanation.

A)

Java program that demonstrates creating a table named 'User' with columns 'User ID' and 'Password' stored as hashed values. It then accepts user input for 'User ID' and 'Password', hashes the password, and checks if they match the entries in the table to confirm whether user access is allowed or not:

Java code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Scanner;

public class UserAuthentication {
    public static void main(String[] args) {
        // JDBC URL for MySQL database
        String url = "jdbc:mysql://localhost:3306/mydatabase";
        String username = "username"; // Replace with your MySQL username
        String password = "password"; // Replace with your MySQL password

        // User input
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter User ID: ");
        String userID = scanner.nextLine();
        System.out.print("Enter Password: ");
        String passwordInput = scanner.nextLine();

        // Hash the password input
        String hashedPassword = hashPassword(passwordInput);

        try {
            // Establish connection
            Connection connection = DriverManager.getConnection(url, username, password);

            // Create table if not exists
            createTable(connection);

            // Check if user access is allowed
```

```

        boolean accessAllowed = checkUserAccess(connection, userID, hashedPassword);

        if (accessAllowed) {
            System.out.println("Access granted. Welcome, " + userID + "!");
        } else {
            System.out.println("Access denied. Incorrect User ID or Password.");
        }

        // Close the connection
        connection.close();
    } catch (SQLException e) {
        System.err.println("Failed to connect to the database");
        e.printStackTrace();
    }
}

// Method to create the User table if not exists
private static void createTable(Connection connection) throws SQLException {
    String createTableSQL = "CREATE TABLE IF NOT EXISTS User (userID VARCHAR(50), password VARCHAR(64))";
    try (PreparedStatement statement = connection.prepareStatement(createTableSQL)) {
        statement.executeUpdate();
    }
}

// Method to hash the password
private static String hashPassword(String password) {
    try {
        MessageDigest digest = MessageDigest.getInstance("SHA-256");
        byte[] hash = digest.digest(password.getBytes());
        StringBuilder hexString = new StringBuilder();
        for (byte b : hash) {
            String hex = Integer.toHexString(0xff & b);
            if (hex.length() == 1) hexString.append('0');
            hexString.append(hex);
        }
        return hexString.toString();
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
        return null;
    }
}

// Method to check if user access is allowed
private static boolean checkUserAccess(Connection connection, String userID, String hashedPassword)
throws SQLException {
    String selectSQL = "SELECT * FROM User WHERE userID = ? AND password = ?";
    try (PreparedStatement statement = connection.prepareStatement(selectSQL)) {

```

```

        statement.setString(1, userID);
        statement.setString(2, hashedPassword);
        try (ResultSet resultSet = statement.executeQuery()) {
            return resultSet.next(); // Returns true if user with given ID and hashed password exists
        }
    }
}
}

```

Explanation:

Import Statements: Import necessary classes from java.sql, java.util, and java.security packages.

Database URL, Username, and Password: Replace the placeholders url, username, and password with your MySQL database connection details.

User Input: Accept user input for 'User ID' and 'Password'.

Hashing Password: The hashPassword() method hashes the password input using the SHA-256 algorithm. This hashed password will be stored in the database.

Connection Establishment: Establish a connection to the MySQL database.

Table Creation: Create a table named 'User' if it does not already exist. The table has columns 'userID' and 'password'.

Check User Access: The checkUserAccess() method checks if the provided 'User ID' and hashed password match any entries in the 'User' table. If a match is found, access is granted.

Closing Connection: Finally, close the connection using the close() method.

This program demonstrates a basic user authentication system using JDBC and MySQL, where passwords are stored as hashed values for security.