Experiment No.9
Demonstrate Database connectivity
Date of Performance:
Date of Submission:

Aim :- Write a java program to connect Java application with the MySQL database

Objective :- To learn database connectivity

Theory:

Database used : MySql

1. Driver class: The driver class for the mysql database is com.mysql.jdbc.Driver.
2. Connection URL: The connection URL for the mysql database is jdbc:mysql://localhost:3306/loan management where jdbc is the API, mysql is the database, localhost is the server name on which mysql is running, can also use IP address, 3306 is the port number and loan management is the database name.
3. Username: The default username for the mysql database is Hiren.
4. Password: It is the password given by the user at the time of installing the mysql database. Password used is " ".

To connect a Java application with the MySQL database, follow the following steps.

- First create a database and then create a table in the mysql database.
- To connect java application with the mysql database, mysqlconnector.jar file is required to be loaded.
- download the jar file mysql-connector.jar
- add the jar file to the same folder as the java program.
- Compile and run the java program to retrieve data from the database.

Conclusion: Data has been retrieved successfully from a table by establishing database connectivity of java program with mysql database.

1. Explain steps to connect a java application with the MySQL database.
   Download MySQL Connector/J: Obtain the MySQL Connector/J JDBC driver from the MySQL website or Maven repository.
   Include Connector in Project: Add the MySQL Connector/J JAR file to the project's classpath.
   Import JDBC Packages: Import the necessary JDBC packages into the Java code, including java.sql.*.
   Load JDBC Driver: Use Class.forName() to load the MySQL JDBC driver.
   Establish Connection: Connect to the MySQL database using DriverManager.getConnection() with the JDBC URL, username, and password.
   Create Statement Object: Create a Statement object to execute SQL queries.
   Execute Queries: Use Statement.execute(), executeUpdate(), or executeQuery() to run SQL queries.
   Process Results: Process query results using ResultSet if applicable.
   Close Resources: Properly close connections, statements, and result sets using their close() methods.