

Integrating MySQL with Java

CLO: 2

Objectives

- Integrate MySQL with Java
- Perform SQL queries using code

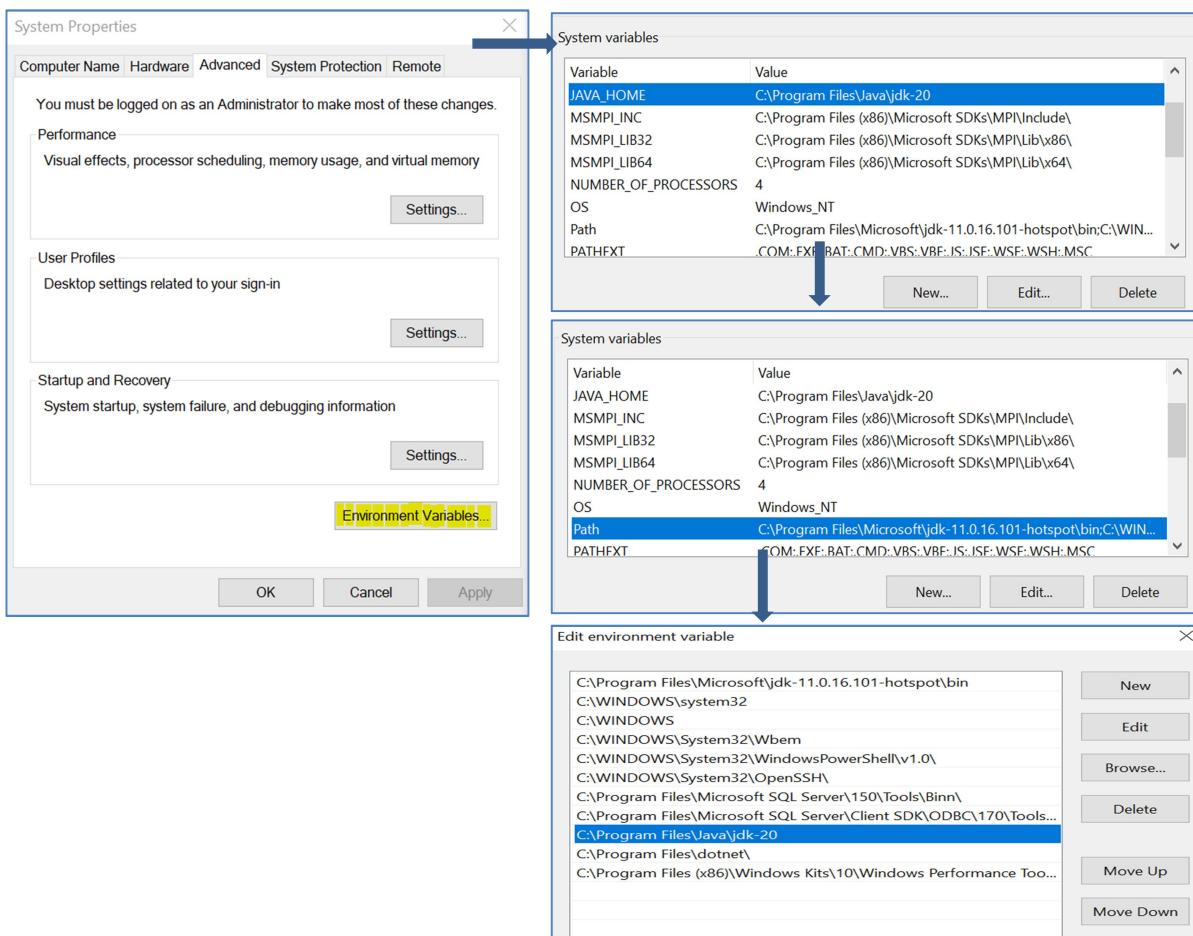
This manual provides step-by-step instructions for integrating MySQL with Java to perform SQL operations programmatically. It covers environment setup, establishing a database connection, and executing SQL queries like SELECT, INSERT, UPDATE, and DELETE using JDBC (Java Database Connectivity).

Installation and Configuration

Ensure the following software is installed on your system before proceeding:

1. Java Development Kit (JDK):

- Download from [Oracle JDK](#).
- Make sure to set your JAVA_HOME environment variable.



2. MySQL:

- o Download and install from [MySQL Downloads](#).

The screenshot shows the MySQL Community Downloads page. A modal window titled "MySQL Installer - Community" is open, displaying a progress bar and the message "Please wait while Windows configures MySQL Installer - Community". Below the modal, the MySQL Installer 8.0.39 download page is visible, showing two download options: "Windows (x86, 32-bit), MSI Installer" (8.0.39, 2.1M) and "Windows (x86, 32-bit), MSI Installer" (8.0.39, 303.0M). Both links have "Download" buttons. At the bottom of the page, there is a note about MD5 checksums and GnuPG signatures.

The screenshot shows the MySQL Installer interface. On the left, a sidebar lists "Choosing a Setup Type", "Installation" (which is selected), "Product Configuration", and "Installation Complete". The main area is titled "Installation" and displays a table of products being installed. The table includes columns for Product, Status, Progress, and Notes. The status for MySQL Server 8.0.31 is "Installing" with 0% progress. Other products listed include MySQL Workbench 8.0.31, MySQL Shell 8.0.31, MySQL Router 8.0.31, Connector/ODBC 8.0.31, Connector/C++ 8.0.31, Connector/J 8.0.31, Connector/.NET 8.0.31, MySQL Documentation 8.0.31, and Samples and Examples 8.0.31, all marked as "Ready to Install".

3. MySQL Connector/J:

- o The JDBC driver required for Java MySQL connectivity.

The screenshot shows the MySQL Product Configuration page. It displays a table of products with columns for Product, Version, Architecture, and Quick Action. The "Connector/J" row is highlighted with a blue background, indicating it is selected. The version is 8.0.31 and the architecture is X86. The "Quick Action" column contains a "Reconfigure" link. Below the table, a "Product Information" section provides details: Product Home (<https://dev.mysql.com/downloads/connector/j/>), Release Changes (<https://dev.mysql.com/doc/relnotes/connector-j/8.0/en/news-8-0-31.html>), Install Path (C:\Program Files (x86)\MySQL\Connector J 8.0), and Install Date (16-Oct-22).

4. DB Configuration

<div style="border: 1px solid #ccc; padding: 10px;"> <p>Type and Networking</p> <p>Server Configuration Type</p> <p>Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.</p> <p>Config Type: Development Computer</p> <p>Connectivity</p> <p>Use the following controls to select how you would like to connect to this server.</p> <p><input checked="" type="checkbox"/> TCP/IP Port: 3306 X Protocol Port: 33060 <input type="checkbox"/> Open Windows Firewall ports for network access <input type="checkbox"/> Named Pipe Pipe Name: MYSQL <input type="checkbox"/> Shared Memory Memory Name: MYSQL</p> <p>Advanced Configuration</p> <p>Select the check box below to get additional configuration pages where you can set advanced and logging options.</p> <p><input type="checkbox"/> Show Advanced and Logging Options</p> </div>	<div style="border: 1px solid #ccc; padding: 10px;"> <p>Type and Networking</p> <p>Server Configuration Type</p> <p>Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.</p> <p>Config Type: Development Computer</p> <p>Connectivity</p> <p>Use the following controls to select how you would like to connect to this server.</p> <p><input checked="" type="checkbox"/> TCP/IP Port: 3306 X Protocol Port: 33060 <input type="checkbox"/> Open Windows Firewall ports for network access <input type="checkbox"/> Named Pipe Pipe Name: MYSQL <input type="checkbox"/> Shared Memory Memory Name: MYSQL</p> <p>Advanced Configuration</p> <p>Select the check box below to get additional configuration pages where you can set advanced and logging options.</p> <p><input type="checkbox"/> Show Advanced and Logging Options</p> </div>								
<input type="button" value="Next >"/> <input type="button" value="Cancel"/>									
<div style="border: 1px solid #ccc; padding: 10px;"> <p>Authentication Method</p> <p><input checked="" type="radio"/> Use Strong Password Encryption for Authentication (RECOMMENDED) MySQL 8 supports a new authentication based on improved stronger SHA256-based password methods. It is recommended that all new MySQL Server installations use this method going forward.</p> <p>Attention: This new authentication plugin on the server side requires new versions of connectors and clients which add support for this new 8.0 default authentication (caching_sha2_password authentication).</p> <p>Currently MySQL 8.0 Connectors and community drivers which use libmysqlclient 8.0 support this new method. If clients and applications cannot be updated to support this new authentication method, the MySQL 8.0 Server can be configured to use the legacy MySQL Authentication Method below.</p> <p><input type="radio"/> Use Legacy Authentication Method (Retain MySQL 5.x Compatibility) Using the old MySQL 5.x legacy authentication method should only be considered in the following cases:</p> <ul style="list-style-type: none"> - If applications cannot be updated to use MySQL 8 enabled Connectors and drivers. - For cases where re-compilation of an existing application is not feasible. - An updated, language specific connector or driver is not yet available. <p>Security Guidance: When possible, we highly recommend taking needed steps towards upgrading your applications, libraries, and database servers to the new stronger authentication. This new method will significantly improve your security.</p> </div>	<div style="border: 1px solid #ccc; padding: 10px;"> <p>Accounts and Roles</p> <p>Root Account Password Enter the password for the root account. Please remember to store this password in a secure place.</p> <p>MySQL Root Password: <input type="password"/> Repeat Password: <input type="password"/> Password strength: Medium</p> <p>MySQL User Accounts Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="padding: 2px;">MySQL User Name</th> <th style="padding: 2px;">Host</th> <th style="padding: 2px;">User Role</th> <th style="padding: 2px; text-align: right;">Add User</th> </tr> </thead> <tbody> <tr> <td style="height: 40px;"></td> <td style="height: 40px;"></td> <td style="height: 40px;"></td> <td style="text-align: right; height: 40px;"><input type="button" value="Edit User"/> <input type="button" value="Delete"/></td> </tr> </tbody> </table> </div>	MySQL User Name	Host	User Role	Add User				<input type="button" value="Edit User"/> <input type="button" value="Delete"/>
MySQL User Name	Host	User Role	Add User						
			<input type="button" value="Edit User"/> <input type="button" value="Delete"/>						
<input type="button" value="< Back"/> <input type="button" value="Next >"/> <input type="button" value="Cancel"/>									
<div style="border: 1px solid #ccc; padding: 10px;"> <p>Windows Service</p> <p><input checked="" type="checkbox"/> Configure MySQL Server as a Windows Service</p> <p>Windows Service Details Please specify a Windows Service name to be used for this MySQL Server instance. A unique name is required for each instance.</p> <p>Windows Service Name: <input type="text" value="MySQL80"/></p> <p><input checked="" type="checkbox"/> Start the MySQL Server at System Startup</p> <p>Run Windows Service as ... The MySQL Server needs to run under a given user account. Based on the security requirements of your system you need to pick one of the options below.</p> <p><input checked="" type="radio"/> Standard System Account Recommended for most scenarios.</p> <p><input type="radio"/> Custom User An existing user account can be selected for advanced scenarios.</p> </div>	<div style="border: 1px solid #ccc; padding: 10px;"> <p>Server File Permissions</p> <p>MySQL Installer can secure the server's data directory by updating the permissions of files and folders located at:</p> <p>C:\ProgramData\MySQL\MySQL Server 8.0\Data</p> <p>Do you want MySQL Installer to update the server file permissions for you?</p> <p><input checked="" type="radio"/> Yes, grant full access to the user running the Windows Service (if applicable) and the administrators group only. Other users and groups will not have access. <input type="radio"/> Yes, but let me review and configure the level of access. <input type="radio"/> No, I will manage the permissions after the server configuration</p> </div>								
<input type="button" value="< Back"/> <input type="button" value="Next >"/> <input type="button" value="Cancel"/>									

Logging Options

Please select the logs you want to activate for this server in addition to the error log. On production computers, it can be beneficial to separate the log files from the data. Specify a file name to save the logs in the data directory (default) or browse to a different location. You must provide an absolute path when specifying the path together with the file name.

Error Log: DESKTOP-8TQ5QNO.err

General Log
The general query log is a general record of what the MySQL Server is doing. It should only be used to track down issues.
File Path: DESKTOP-8TQ5QNO.log

Slow Query Log
The slow query log consists of SQL statements that took more than the given value of seconds to execute. It is recommended to turn this log on.
File Path: DESKTOP-8TQ5QNO-slow.log Seconds: 10

Binary Log
The binary log contains all database events and is used for replication and data recovery operations. Enabling the log has a performance impact on the server. Enter the log name without a file extension.
File Path: DESKTOP-8TQ5QNO-bin

Apply Configuration

Click [Execute] to apply the changes

Configuration Steps Log

- Writing configuration file
- Updating Windows Firewall rules
- Adjusting Windows service
- Initializing database (may take a long time)
- Updating permissions for the data folder and related server files
- Starting the server
- Applying security settings
- Updating the Start menu link

Apply Configuration

The configuration operation has completed.

Configuration Steps Log

- Writing configuration file
- Updating Windows Firewall rules
- Adjusting Windows service
- Initializing database (may take a long time)
- Updating permissions for the data folder and related server files
- Starting the server
- Applying security settings
- Updating the Start menu link

The configuration for MySQL Server 8.0.31 was successful. Click Finish to continue.

MySQL Router Configuration

Bootstrap MySQL Router for use with InnoDB Cluster

This wizard can bootstrap MySQL Router to direct traffic between MySQL applications and InnoDB Cluster. Applications that connect to the router will be automatically directed to an available read/write or read-only member of the cluster.

The bootstrapping process requires a connection to InnoDB Cluster. In order to register the MySQL Router for monitoring, use the current Read/Write instance of the cluster.

Hostname:

Port: 3306

Management User: root

Password:

MySQL Router requires specification of a base port (between 80 and 65532). The first port is used for classic read/write connections. The other ports are computed sequentially after the first port. If any port is indicated to be in use, please change the base port.

Classic MySQL protocol connections to InnoDB Cluster:

Read/Write: 6446

Read Only: 6447

X Protocol connections to InnoDB Cluster:

Read/Write: 6448

Read Only: 6449

Advanced Options

Server ID: 1

A unique numeric identifier used in a replication topology. If binary logging is enabled, a Server ID must be specified.

Table Names Case:

Lower Case (default):

This option sets the configuration variable lower_case_table_names = 1.

Preserve Given Case:

This option sets the configuration variable lower_case_table_names = 2.

Installation Complete

The installation procedure has been completed.

Start MySQL Workbench after setup

Start MySQL Shell after setup

The MySQL Shell is an advanced MySQL client application that can be used to work with single MySQL Server instances. Further, it can be used to create and manage InnoDB Cluster, an integrated solution for high availability and scalability of MySQL databases, without requiring advanced MySQL expertise.



Refer to the following links for documentation, tutorials and examples on MySQL Shell:

[MySQL Shell Documentation](#)

[Setting up a Real World Cluster Blog](#)

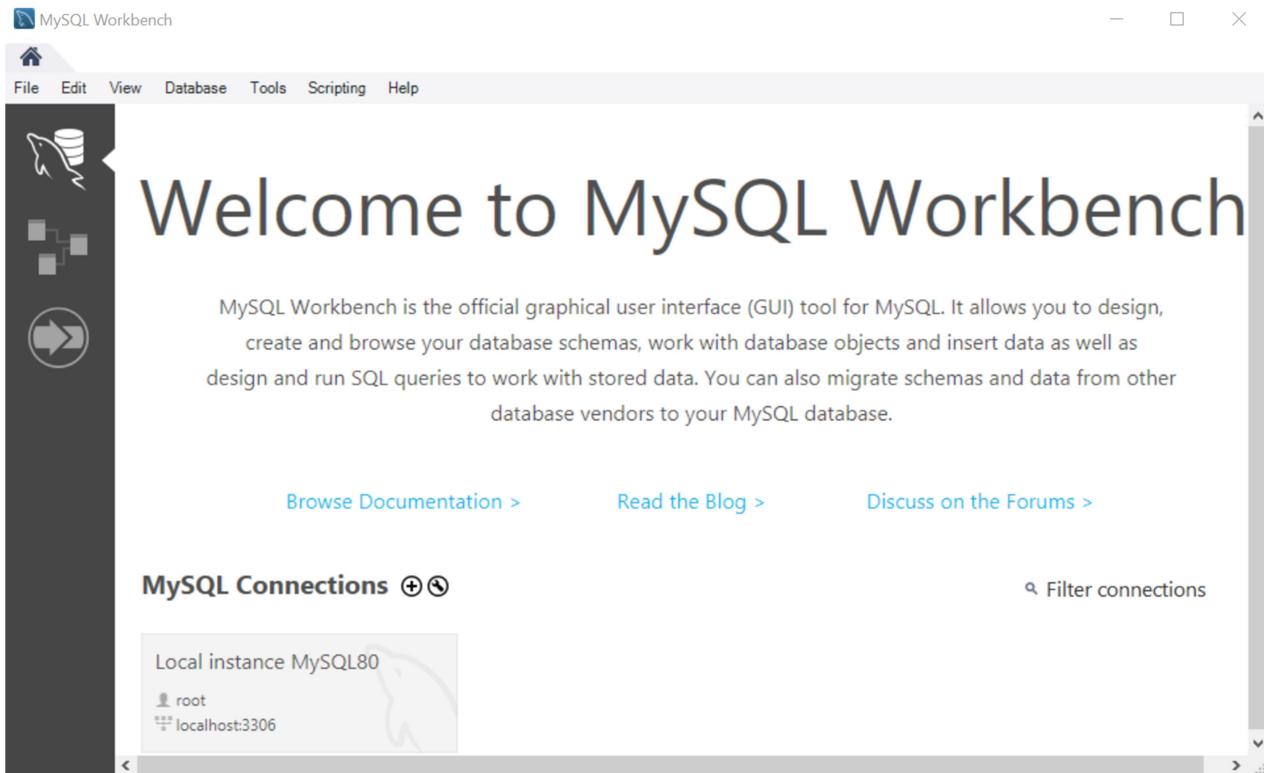
[The All New MySQL InnoDB ReplicaSet Blog](#)

[Changing Cluster Options Live Blog](#)

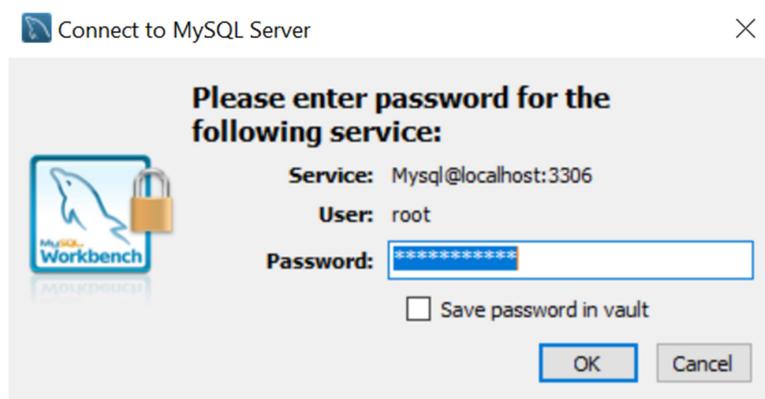
5. IDE (Integrated Development Environment):
- o Use an IDE like **Eclipse** or **IntelliJ IDEA** for Java development.

Setting up the MySQL Database

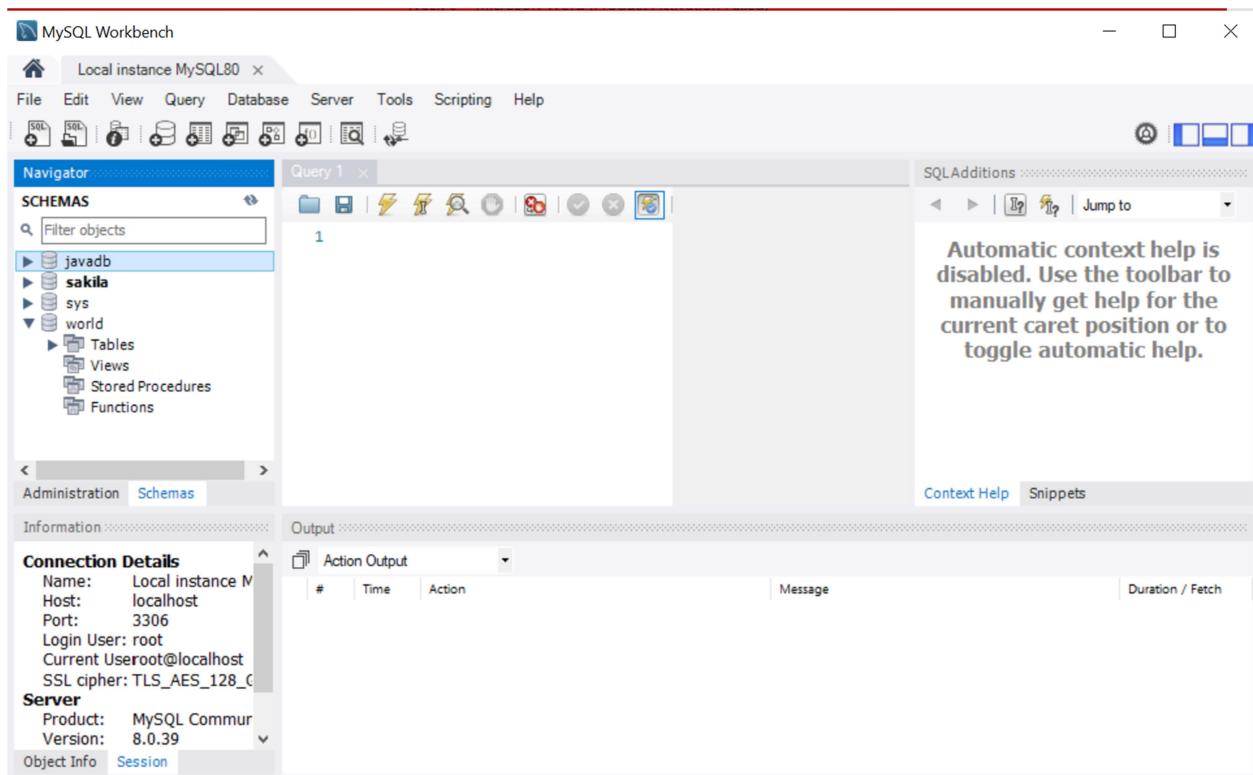
1. Open MySql Workbench



2. Create or connect to already existing MySQL Connection



3. Go-through MySQL workbench platform



4. Create SQL tab for executing queries or use the one created by default. Write SQL Command to see all the available Databases

The screenshot shows the results of a SQL query in the Query 1 tab. The query is:

```
1 show databases;
```

The results are displayed in the Result Grid:

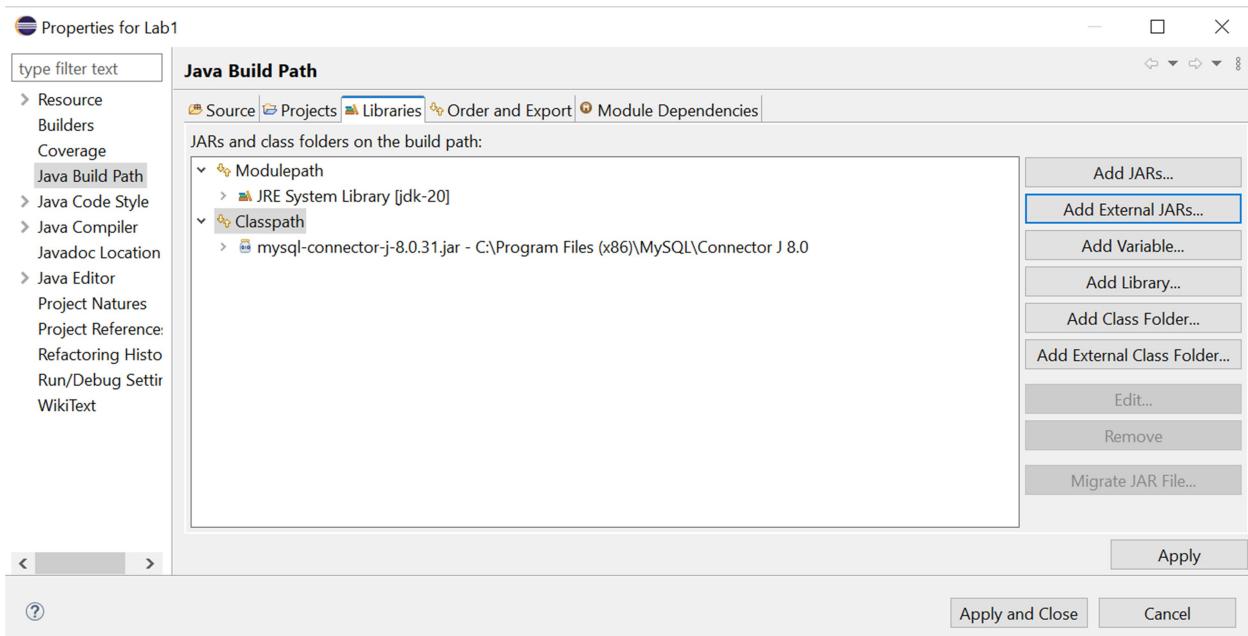
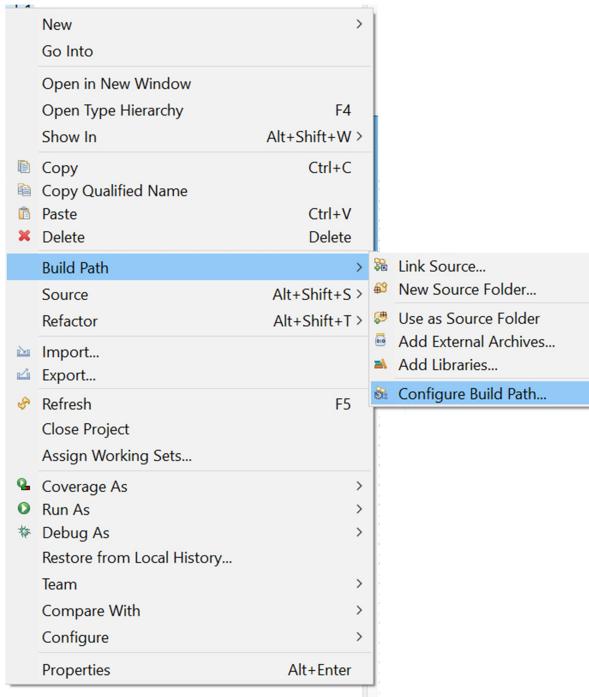
Database
information_schema
javadb
mysql
performance_schema
sakila
sys
world

5. Practice executing all the SQL commands (especially mentioned on cheat sheet) using MySQL Workbench platform. (Weighted evaluation)

Setting up Java Project

1. Add MySQL Connector to Project:

- In **Eclipse**: Right-click on your project → Build Path → Configure Build Path → Add External JARs → Select the MySQL Connector .jar file.
- In **IntelliJ IDEA**: File → Project Structure → Libraries → Add Library → Select the MySQL Connector .jar file.



Database integration

There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:

1. Import JDBC Libraries
2. Create connection
3. Create statement
4. Execute queries
5. Close connection

```
package com.java.practice.mysql;

import java.sql.Timestamp;
//Step 1: Import JDBC Libraries
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

public class mysql_queries {
    // Database URL, User name, and Password
    private static final String DB_URL = "jdbc:mysql://localhost:3306/sakila";
    private static final String USER = "root";
    private static final String PASSWORD = "blackParrot3";

    public static void main(String[] args) {
        Connection connection = null;
        Statement statement = null;

        try {
            // Step 2: Create Connection

            // a. Register JDBC driver (automatically done in newer versions)
            Class.forName("com.mysql.cj.jdbc.Driver");

            // b. Open a connection
            System.out.println("Connecting to database...");
            connection = DriverManager.getConnection(DB_URL, USER, PASSWORD);
            System.out.println("Successfully connected to MySQL!");

            // Step 3: Create statement
            statement = connection.createStatement();

            // Step 4: Execute SQL SELECT query
            String sql = "SELECT * FROM actor";
            ResultSet resultSet = statement.executeQuery(sql);

            // Process the result set
            while (resultSet.next()) {
                int id = resultSet.getInt("actor_id");
                String first_name = resultSet.getString("first_name");
                String last_name = resultSet.getString("last_name");
                Timestamp last_update = resultSet.getTimestamp("last_update");

                System.out.println("ID: " + id + ", First Name: " + first_name +
                    ", Last Name: " + last_name + ", Last Update: " + last_update);
            }
            resultSet.close();

        } catch (SQLException | ClassNotFoundException e) {
            e.printStackTrace();
        } finally {
            // Step 5: Close connection
            try {
                if (connection != null) {
                    connection.close();
                }
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }
}
```

Output

```
Connecting to database...
Successfully connected to MySQL!
ID: 1, First Name: PENELOPE, Last Name: GUINNESS, Last Update: 2006-02-15 04:34:33.0
ID: 2, First Name: NICK, Last Name: WAHLBERG, Last Update: 2006-02-15 04:34:33.0
ID: 3, First Name: ED, Last Name: CHASE, Last Update: 2006-02-15 04:34:33.0
ID: 4, First Name: JENNIFER, Last Name: DAVIS, Last Update: 2006-02-15 04:34:33.0
ID: 5, First Name: JOHNNY, Last Name: LOLLOBRIGIDA, Last Update: 2006-02-15 04:34:33.0
ID: 6, First Name: BETTE, Last Name: NICHOLSON, Last Update: 2006-02-15 04:34:33.0
ID: 7, First Name: GRACE, Last Name: MOSTEL, Last Update: 2006-02-15 04:34:33.0
ID: 8, First Name: MATTHEW, Last Name: JOHANSSON, Last Update: 2006-02-15 04:34:33.0
ID: 9, First Name: JOE, Last Name: SWANK, Last Update: 2006-02-15 04:34:33.0
ID: 10, First Name: CHRISTIAN, Last Name: GABLE, Last Update: 2006-02-15 04:34:33.0
ID: 11, First Name: ZERO, Last Name: CAGE, Last Update: 2006-02-15 04:34:33.0
ID: 12, First Name: KARL, Last Name: BERRY, Last Update: 2006-02-15 04:34:33.0
ID: 13, First Name: UMA, Last Name: WOOD, Last Update: 2006-02-15 04:34:33.0
ID: 14, First Name: VIVIEN, Last Name: BERGEN, Last Update: 2006-02-15 04:34:33.0
ID: 15, First Name: CUBA, Last Name: OLIVIER, Last Update: 2006-02-15 04:34:33.0
ID: 16, First Name: FRED, Last Name: COSTNER, Last Update: 2006-02-15 04:34:33.0
ID: 17, First Name: HELEN, Last Name: VOIGHT, Last Update: 2006-02-15 04:34:33.0
ID: 18, First Name: DAN, Last Name: TORN, Last Update: 2006-02-15 04:34:33.0
ID: 19, First Name: BOB, Last Name: FAWCETT, Last Update: 2006-02-15 04:34:33.0
ID: 20, First Name: LUCILLE, Last Name: TRACY, Last Update: 2006-02-15 04:34:33.0
ID: 21, First Name: KIRSTEN, Last Name: PALTROW, Last Update: 2006-02-15 04:34:33.0
ID: 22, First Name: ELVIS, Last Name: MARX, Last Update: 2006-02-15 04:34:33.0
ID: 23, First Name: SANDRA, Last Name: KILMER, Last Update: 2006-02-15 04:34:33.0
ID: 24, First Name: CAMERON, Last Name: STREEP, Last Update: 2006-02-15 04:34:33.0
ID: 25, First Name: KEVIN, Last Name: BLOOM, Last Update: 2006-02-15 04:34:33.0
ID: 26, First Name: RIP, Last Name: CRAWFORD, Last Update: 2006-02-15 04:34:33.0
ID: 27, First Name: JULIA, Last Name: MCQUEEN, Last Update: 2006-02-15 04:34:33.0
```

Best Practices

1. **Always Close Resources:** Use `finally` blocks (or Java 7+ `try-with-resources`) to close connections, statements, and result sets.
 2. **Prepared Statements:** Use `PreparedStatement` instead of `Statement` to avoid SQL injection attacks.
 3. **Error Handling:** Implement robust error handling for database connectivity issues.
 4. **Transactions:** Use transactions for atomic operations (group multiple queries in one logical unit).