JDBC

JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database.

Why Should We Use JDBC

Before JDBC, ODBC API was the database API to connect and execute the query with the database. But, ODBC API uses ODBC driver which is written in C language (i.e. platform dependent and unsecured). That is why Java has defined its own API (JDBC API) that uses JDBC drivers (written in Java language).

We can use JDBC API to handle database using Java program and can perform the following activities:

1. Connect to the database
2. Execute queries and update statements to the database
3. Retrieve the result received from the database.

What is API

API (Application programming interface) is a document that contains a description of all the features of a product or software. It represents classes and interfaces that software programs can follow to communicate with each other. An API can be created for applications, libraries, operating systems, etc.

Connection interface

A Connection is a session between a Java application and a database. It helps to establish a connection with the database.

The Connection interface is a factory of Statement, PreparedStatement, and DatabaseMetaData, i.e., an object of Connection can be used to get the object of Statement and DatabaseMetaData. The Connection interface provide many methods for transaction management like commit(), rollback(), setAutoCommit() etc.

Prof. Usman Ahmad
Govt. Islamia Graduate College, Gujranwala.

The commit() method of the Connection interface saves all the modifications made since the last commit. If any issue occurs after the commit you can revert all the changes done till this commit by invoking the rollback() method.

setAutoCommit(true); enables auto-commit mode, which means that each statement is once again committed automatically when it is completed.

Statement interface

The **Statement interface** provides methods to execute queries with the database. The statement interface is a factory of ResultSet i.e. it provides factory method to get the object of ResultSet.

Commonly used methods of Statement interface:

The important methods of Statement interface are as follows:

1. **public ResultSet executeQuery(String sql):** is used to execute SELECT query. It returns the object of ResultSet.
2. **public int executeUpdate(String sql):** is used to execute specified query, it may be create, drop, insert, update, delete etc.
3. **public boolean execute(String sql):** is used to execute queries that may return Multiple results.
4. **public int[] executeBatch():** is used to execute batch of commands.

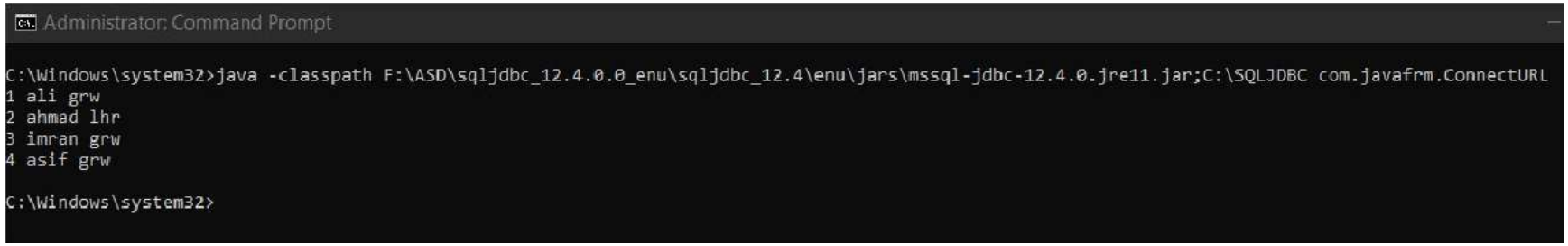
Java Database Connectivity with 5 Steps

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

- Create connection
- Create statement
- Execute queries
- Close connection

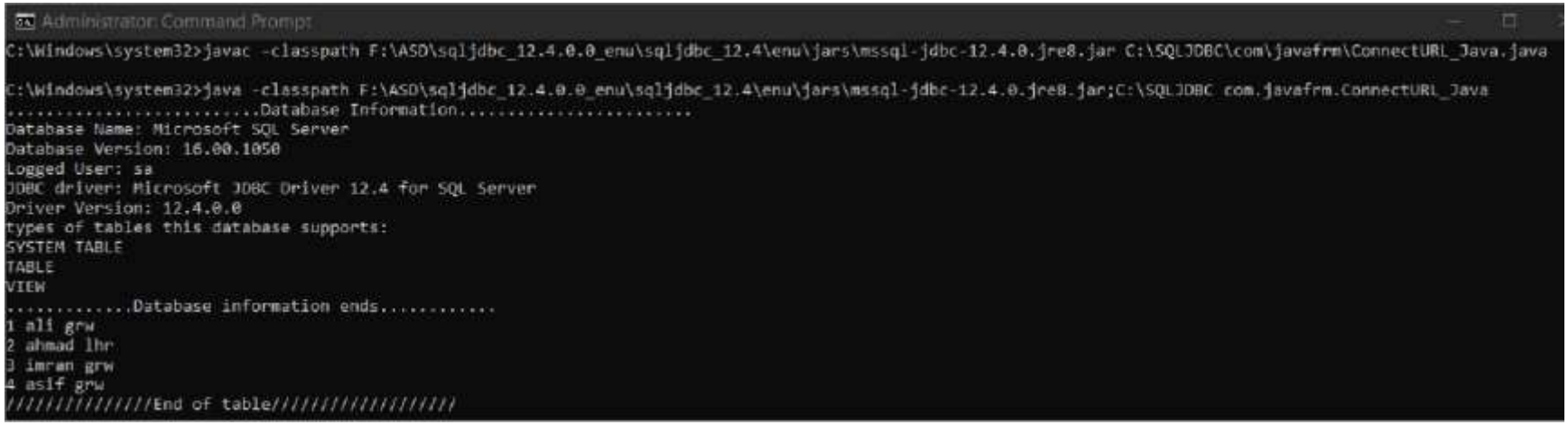
Example to Connect Java Application with Ms. SQL Server

```
package com.javafrm;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
public class ConnectURL {
   public static void main(String[] args) {

      String connectionUrl = "jdbc:sqlserver://localhost:49697;databaseName=BSIT;
integratedSecurity=false;encrypt=true;trustServerCertificate=true;user=sa;password=root";

      try (Connection con = DriverManager.getConnection(connectionUrl); Statement stmt = con.createStatement();) {
         String SQL = "SELECT * FROM emp";
         ResultSet rs = stmt.executeQuery(SQL);

         // Iterate through the data in the result set and display it.
         while (rs.next()) {
            System.out.println(rs.getInt("empid") + " " + rs.getString("ename") + " " + rs.getString("address"));
         }
      }
// Handle any errors that may have occurred.
      catch (SQLException e) {
         e.printStackTrace();
      }
   }
}
```




```
package com.javafrm;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.DatabaseMetaData;
import java.util.Properties;

public class ConnectURL_Java{
   public static void main(String[] args) {

     String connectionUrl =
"jdbc:sqlserver://localhost:49697;databaseName=BSIT;authenticationScheme=NTLM;integratedSecurity=false;encrypt=true;trustServ
erCertificate=true;user=sa;password=root";

        try (Connection con = DriverManager.getConnection(connectionUrl);Statement stmt = con.createStatement();) {

        DatabaseMetaData metaData = con.getMetaData();
        String product_name = metaData.getDatabaseProductName();
        System.out.println(".....Database Information.....");
        System.out.println("Database Name: " + product_name);
        System.out.println("Database Version: " + metaData.getDatabaseProductVersion());
```

```
        System.out.println("Logged User: " + metaData.getUserName());
        System.out.println("JDBC driver: " + metaData.getDriverName());
        System.out.println("Driver Version: " + metaData.getDriverVersion());

        ResultSet mdrs = metaData.getTableTypes();
        System.out.println("types of tables this database supports: ");
        while (mdrs.next()) {
                System.out.println(mdrs.getString("TABLE_TYPE"));
        }

        System.out.println(".....Database information ends.....");

        String SQL = "SELECT * FROM emp";
      ResultSet rs = stmt.executeQuery(SQL);
        while (rs.next()) {
                System.out.println(rs.getInt("empid") + " " + rs.getString("ename") + " " + rs.getString("address"));
        }
        System.out.println("///////////////End of table//////////////////");
        Properties prop = System.getProperties();
        System.out.println("JVM Vendor========>: " + prop.getProperty("java.verndor"));

        rs.close();
        stmt.close();
        con.close();
        }

        catch (SQLException e) {
                e.printStackTrace();
        }
    }//end of main
}//end of class
```


Administrator: Command Prompt
C:\Windows\system32>javac -classpath F:\ASD\sqljdbc_12.4.0.0_enu\sqljdbc_12.4\enu\jars\mssql-jdbc-12.4.0.jre8.jar C:\SQLJDBC\com\javafrm\ConnectURL_Java.java
C:\Windows\system32>java -classpath F:\ASD\sqljdbc_12.4.0.0_enu\sqljdbc_12.4\enu\jars\mssql-jdbc-12.4.0.jre8.jar;C:\SQLJDBC com.javafrm.ConnectURL_Java
.....Database Information.....
Database Name: Microsoft SQL Server
Database Version: 16.00.1050
Logged User: sa
JDBC driver: Microsoft JDBC Driver 12.4 for SQL Server
Driver Version: 12.4.0.0
types of tables this database supports:
SYSTEM TABLE
TABLE
VIEW
.....Database information ends.....
1 ali grw
2 ahmad lhr
3 imran grw
4 asif grw
///////////////End of table//////////////////