**What is JDBC?**

The term JDBC stands for **Java Database Connectivity**. JDBC is a specification from Sun microsystems. JDBC is an API(Application programming interface) in Java that helps users to interact or communicate with various databases.

The **classes** and **interfaces** of JDBC API allow the application to send the request to the specified database.

The JDBC API is a Java API that can access any kind of tabular data, especially data stored in a **Relational Database**. JDBC helps to write Java applications that manage these three programming activities:

* Connect to a data source, like a database
* Send queries and update statements to the database
* Retrieve and process the results received from the database in answer to your query

**The JDBC API:** It allows the Java Programs to perform the execution of the SQL statements and then get the results.

A few of the crucial interfaces and classes defined in the JDBC API are the following:

* Drivers
* DriverManager
* **Statement**
* **Connection**
* CallableStatement
* PreparedStatement
* **ResultSet**

**Create database databasename**

use swastikdb//select database

SELECT \* FROM swastikdb.tblstudent;//fetch all records from tblStudent

insert into tblstudent values(1,'Sunil Chaudhary','suneel@gmail.com','Male')//insert record

select \* from tblstudent

insert into tblStudent values(3,'Ramesh','ramesh@gmail.com','Male')//insert

update tblStudent set Name='Sunil Chaudhary1' where Id=1//update rectord

delete from tblstudent where Id=1//delete

**Statement:**

* Statement is an interface in JDBC used to **execute SQL queries** against a database.
* It is primarily used for executing **static SQL queries**, which do not have dynamic or user-provided values.

**There are three types of Statement objects:**

1. **Statement**: Used for executing simple SQL queries without parameters.
2. **PreparedStatement**: Used for precompiled SQL queries **with parameters**, improving performance and security by preventing SQL injection.
3. **CallableStatement**: Used for executing database **stored procedures**.

Statement objects are created using a Connection object, like this:

Statement statement = connection.createStatement();

you can execute SQL queries using the **execute()**, **executeQuery()**, and **executeUpdate()** methods of the Statement interface. For example:

ResultSet resultSet = statement.executeQuery("SELECT \* FROM my\_table");

Once you have executed a query, you can retrieve the results using a **ResultSet**

**ResultSet:**

* **ResultSet** is another JDBC interface that represents the result set of a database query.
* It allows you to navigate and access the data retrieved from the database after executing a query.
* You can use methods like **next()**, **getInt()**, **getString()**, etc., to navigate through the result set and retrieve specific column values.
* Typically, you obtain a **ResultSet** object by calling the **executeQuery**() method on a **Statement** or **PreparedStatement**, as shown in the example above.

Here's an example of how to retrieve data from a **ResultSet**:

 ResultSet resultSet = statement.executeQuery("SELECT name, age FROM employees");

 while (resultSet.next()) {

 String name = resultSet.getString("name");

 int age = resultSet.getInt("age");

 System.out.println("Name: " + name + ", Age: " + age);

}

**Steps For Connectivity Between Java Program and Database**

1. Import the Packages
2. Load the drivers using the *forName() method*
3. Register the drivers *using DriverManager*
4. Establish a connection*using the Connection class object*
5. Create a statement
6. Execute the query
7. Close the connections

***Question: 1 How do you execute SQL queries in JDBC***Ans:

To process an SQL statement, you need to follow the steps given below:

1. Establish the connection.
2. Create a statement.
3. Execute the statement/query.
4. Process the result.
5. Close the connection.

**1. Establishing a Connection**

To process SQL statements first of all you need to establish connection with the desired DBMS or, file System or, other data sources.

* To do so, Register the JDBC driver class, corresponding to the DataSource you need to the DriverManager using the **registerDriver()** method.

*Driver myDriver = new com.mysql.jdbc.Driver();*

*DriverManager.registerDriver(myDriver);*

You can also register the driver using the **forName()** method. This method loads the specified class in to the memory and it automatically gets registered.

Class.forName("com.mysql.jdbc.Driver");

After registering the Driver class, get the Connection object using the **getConnection()** method.

This method accepts a database URL (an address that points to your database), Username and, password and, returns a connection object.

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/collegedb?useSSL=false","root","root");

**2.Creating a Statement**

The Statement interface represents an SQL statement and JDBC provides 3 kinds of Statements

* **Statement:** A general purpose statement which does not accept any parameters.
* **PreparedStatement:** A precompiled SQL statement which accepts input parameters.
* **Callable Statement:** This is used to call the stored procedures.

conn.createStatement();

            conn.prepareStatement(query);

            conn.prepareCall(query)

**3.Executing the Statements**

After creating the statement objects, you need to execute them. To execute the statements, the Statement interface provides three methods namely, **execute(), executeUpdate()** and, **executeQuery().**

* **execute():** Used to execute SQL DDL statements, it returns a boolean value specifying whether the ResultSet object can be retrieved.
* **executeUpdate():** Used to execute statements such as insert, update, delete. It returns an integer value representing the number of rows affected.
* **executeQuery():** Used to execute statements that returns tabular data (example select). It returns an object of the class ResultSet.

stmt.execute(query);

stmt.executeQuery(query);

stmt.execute(query);

**4. Processing the Result**

Once you execute the statements/queries you will get the result of the respective query as a return value from **execute()** (boolean value) or, **executeQuery()** (ResultSet) or, **executeUpdate()** (integer value) methods.

 while(rs.next())

            System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));

            con.close();

            }

### 5.Close Connection

*conn.close();*

**DML Operation in JAVA**

**Executing Select Statement**

import java.sql.\*;

public class App {

    public static void main(String[] args) throws Exception {

        try {

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/swastik?useSSL=false","root","root");

            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select \* from student");

            while(rs.next())

            System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));

            con.close();

            }

            catch(Exception e){ System.out.println(e);}

            }

    }

Output:

1 Sunil Chaudhary Kathmandu

2 Bikash Shrestha Kathmandu

3 Dinesh Gautam Kathmandu

4 Hira Kahdma Kathmandu

**Insert Statement**import java.sql.\*;  
public class App {

    public static void main(String[] args) throws Exception {

        try {

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/swastik?useSSL=false","root","root");

String sql="insert into student(ID,Name,Address) values(?,?,?)";

            PreparedStatement ps = con.prepareStatement(sql);

            ps.setInt(1, 5);

            ps.setString(2, "Suman");

            ps.setString(3, "Kathmandu");

            ps.executeUpdate();

            con.close();

            System.out.println("Data Inserted");

            }

            catch(Exception e)

{

System.out.println(e);  
}

            }

    }

**Update Statement**

import java.sql.\*;

public class App {

    public static void main(String[] args) throws Exception {

        try {

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/swastik?useSSL=false","root","root");

            String sql=" update tblstudent set Name=?, Email=? where Id=?";

            PreparedStatement ps = con.prepareStatement(sql);

            ps.setInt(3, 1);

            ps.setString(2, "suneel@hotmail.com");

            ps.setString(1, "suneel chaudhary");

            ps.executeUpdate();

            con.close();

            System.out.println("Data Updated");

            }

            catch(Exception e){ System.out.println(e);}

            }

    }

**Delete Statement**

import java.sql.\*;

public class App {

    public static void main(String[] args) throws Exception {

        try {

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/swastik?useSSL=false","root","root");

            String sql="delete from student where id = ?";

            PreparedStatement ps = con.prepareStatement(sql);

            ps.setInt(1, 1);  //id =1

            ps.executeUpdate();

            con.close();

            System.out.println("Data Deleted");

            }

            catch(Exception e){ System.out.println(e);}

            }

    }

**Exam question:Insert 3 student record and display all student**

Create class CRUDStudent.java

import java.sql.\*;

public class CRUDStudent {

    public void InsertStudent(int id, String name, String email,String gender)

    {

        try {

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/swastik?useSSL=false","root","root");

String sql="insert into student(ID,Name,Address) values(?,?,?)";

            PreparedStatement ps = con.prepareStatement(sql);

            ps.setInt(1, id);

            ps.setString(2, name);

            ps.setString(3, email);

            ps.setString(4, gender);

            ps.executeUpdate();

            con.close();

            }

            catch(Exception e){ System.out.println(e);}

    }

    public void DisplayRecord()

    {

        try {

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/swastik?useSSL=false","root","root");

            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select \* from student");

            while(rs.next())

            System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3)+"  "+rs.getString(4));

            con.close();

            }

            catch(Exception e){ System.out.println(e);}

            }

}

App.java

import java.util.\*;

public class App {

    public static void main(String[] args) throws Exception {

       CRUDStudent obj=new CRUDStudent();

       Scanner sc=new Scanner(System.in);

       for (int i = 0; i < 5; i++) {

        System.out.println("Enter Record Student#: "+(i+1));

        System.out.print("Enter Id: ");

        int id=sc.nextInt();

        System.out.print("Enter Name: ");

        String name=sc.next();

        System.out.print("Enter Email: ");

        String email=sc.next();

        System.out.print("Enter Gender: ");

        String gender=sc.next();

        obj.InsertStudent(id, name, email, gender);

        System.out.println("Record Inserted");

       }

       obj.DisplayRecord();

       sc.close();

    }

}