**1) What is JDBC?**

**JDBC (Java Database Connectivity)** is an API (Application Programming Interface) in Java that allows Java programs to interact with databases. It provides a standard interface to connect to various relational databases (such as MySQL, PostgreSQL, Oracle, etc.), execute SQL queries, and retrieve or modify data.

**Key Features of JDBC:**

* Platform-independent: Works across different operating systems.
* Supports all major relational databases (via specific database drivers).
* Allows operations like:
  + Establishing a database connection.
  + Executing SQL statements (e.g., SELECT, INSERT, UPDATE, DELETE).
  + Retrieving query results.

**2) How to connect your Java program to a database?**

To connect a Java program to a database, you need to follow these steps:

**Step 1: Import the Required JDBC Packages**

* Import the java.sql package to use JDBC classes.

Example:

import java.sql.\*;

**Step 2: Load the Database Driver**

* Load the JDBC driver for the database you want to connect to.
* Modern versions of JDBC automatically load the driver if it's on the classpath.

Example (explicit loading for older versions):

Class.forName("com.mysql.cj.jdbc.Driver"); // MySQL driver

**Step 3: Establish the Database Connection**

* Use DriverManager.getConnection() to establish a connection to the database.
* Provide:
  + **URL**: Specifies the database type, host, port, and database name.
  + **Username**: Your database username.
  + **Password**: Your database password.

Example:

Connection connection = DriverManager.getConnection(

"jdbc:mysql://localhost:3306/mydatabase", "username", "password");

**Step 4: Create a Statement or PreparedStatement**

* Use Connection.createStatement() or Connection.prepareStatement() to create a statement object for executing SQL queries.

Example:

Statement statement = connection.createStatement();

**Step 5: Execute SQL Queries**

* Use methods like executeQuery() (for SELECT) or executeUpdate() (for INSERT, UPDATE, DELETE).

// Executing a SELECT query

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

// Processing the result set

while (resultSet.next()) {

System.out.println("User ID: " + resultSet.getInt("id"));

System.out.println("User Name: " + resultSet.getString("name"));

}

**Step 6: Close the Connection**

* Always close the database connection and other resources like Statement and ResultSet to free up system resources.

Example:

resultSet.close();

statement.close();

connection.close();