JDBC - - Java Data Base Connectivity

1. 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. There are four types of JDBC drivers.
   JDBC-ODBC Bridge Driver,
   Native Driver,
   Network Protocol Driver, and
   Thin Driver
2. We can use JDBC API to access tabular data stored in any relational database. By the help of JDBC API, we can save, update, delete and fetch data from the database. It is like Open Database Connectivity (ODBC) provided by Microsoft.
3. 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.
4. We can use JDBC API to handle database using Java program and can perform the following activities:
   Connect to the database
   Execute queries and update statements to the database
   Retrieve the result received from the database.
5. Basic steps to use a database in JAVA
   .Establish a connection
   .Create JDBC Statements
   .Execute SQL Statements
   .GET Result Set
   .Close connection
6. **Establish a connection :**
   import java.sql.*;
   Load the vendor specific driver
   Class.forName("oracle.jdbc.driver.OracleDriver");
   Dynamically loads a driver class, for Oracle database
   Make the connection
   Connectioncon=DriverManager.getConnection("jdbc:oracle:thin:@oracleprod:1521:OPROD" , username, passwd);
   Establishes connection to database by obtaining
   a Connection object.
7. **Create JDBC statements :**
   Statement stmt = con.createStatement() ;
   Creates a Statement object for sending SQL statements to the database.
8. **Executing SQL statements:**

   String createLehigh = "Create table Lehigh " + "(SSN Integer not null, Name VARCHAR(32), " + "Marks Integer)"; stmt.executeUpdate(createLehigh);

   String insertLehigh = "Insert into Lehigh values“ + "(123456789,abc,100)";

   stmt.executeUpdate(insertLehigh);

9. **Get Resultset:**

```
String queryLehigh = "select * from Lehigh";
ResultSet rs = Stmt.executeQuery(queryLehigh);
while (rs.next())
{
 int ssn = rs.getInt("SSN");
String name = rs.getString("NAME");
int marks = rs.getInt("MARKS");
 }
```

10. Close Connection:

```
stmt.close();
con.close();
```

- **Transactions:**
- JDBC allows SQL statements to be grouped together into a single transaction.
- Transaction control is performed by the Connection object, default mode is auto-commit, I.e., each sql statement is treated as a transaction.
- We can turn off the auto-commit mode with con.setAutoCommit(false).
- And turn it back on with con.setAutoCommit(true).
- Once auto-commit is off, no SQL statement will be committed until an explicit is invoked con.commit().
- At this point all changes done by the SQL statements will be made permanent in the database.