JAVA ASSESSMENT-2

NAME: Ramireddy vishnu Vardhana Reddy

**THEORETICAL Q/A:**

**Q1) What is JDBC Driver? Please explain in detail.**

**ANS:** JDBC Driver is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers:

• **JDBC-ODBC bridge driver--** The JDBC-ODBC bridge driver uses ODBC driver to connect to the database. The JDBC-ODBC bridge driver converts JDBC method calls into the ODBC function calls. This is now discouraged because of thin driver.

• **Native-API driver (partially java driver)--** The Native API driver uses the client-side libraries of the database. The driver converts JDBC method calls into native calls of the database API. It is not written entirely in java.

• **Network Protocol driver (fully java driver)--** The Network Protocol driver uses middleware (application server) that converts JDBC calls directly or indirectly into the vendor-specific database protocol. It is fully written in java.

• **Thin driver (fully java driver)--** The thin driver converts JDBC calls directly into the vendor-specific database protocol. That is why it is known as thin driver. It is fully written in Java language.

**Q2) What are the JDBC API components?**

**ANS:** The JDBC API provides the following interfaces and classes −

• **DriverManager** − This class manages a list of database drivers. Matches connection requests from the java application with the proper database driver using communication sub protocol. The first driver that recognizes a certain subprotocol under JDBC will be used to establish a database Connection.

• **Driver** − This interface handles the communications with the database server. You will interact directly with Driver objects very rarely. Instead, you use DriverManager objects, which manages objects of this type. It also abstracts the details associated with working with Driver objects.

• **Connection** − This interface with all methods for contacting a database. The connection object represents communication context, i.e., all communication with database is through connection object only.

• **Statement** − You use objects created from this interface to submit the SQL statements to the database. Some derived interfaces accept parameters in addition to executing stored procedures.

• **ResultSet** − These objects hold data retrieved from a database after you execute an SQL query using Statement objects. It acts as an iterator to allow you to move through its data.

• **SQLException** − This class handles any errors that occur in a database application.

**Q3) What are the differences between Statement and PreparedStatement interface?**

|  |  |
| --- | --- |
| **ANS: Difference between Statement and PreparedStatement : Statement** | **PreparedStatement** |
| It is used when SQL query is to be executed only once. | It is used when SQL query is to be executed multiple times. |
| You cannot pass parameters at runtime. | You can pass parameters at runtime. |
| Used for CREATE, ALTER, DROP statements. | Used for the queries which are to be executed multiple times. |
| Performance is very low. | Performance is better than Statement. |
| It is base interface. | It extends statement interface. |
| Used to execute normal SQL queries. | Used to execute dynamic SQL queries. |
| We cannot use statement for reading binary data. | We can use Preparedstatement for reading binary data. |
| It is used for DDL statements. | It is used for any SQL Query. |
| We cannot use statement for writing binary data. | We can use Preparedstatement for writing binary data. |
| No binary protocol is used for communication. | Binary protocol is used for communication. |