**3.Explain Basic Steps in writing a Java program using JDBC?**

JDBC makes the interaction with RDBMS simple and intuitive. When a Java application needs to access database :

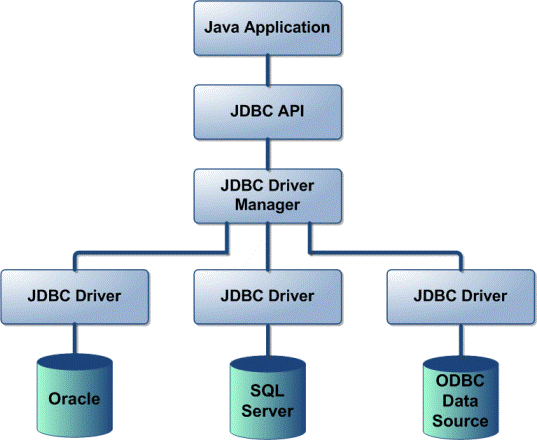
* Load the RDBMS specific JDBC driver because this driver actually communicates with the database (Incase of JDBC 4.0 this is automatically loaded).
* Open the connection to database which is then used to send SQL statements and get results back.
* Create JDBC Statement object. This object contains SQL query.
* Execute statement which returns resultset(s). ResultSet contains the tuples of database table as a result of SQL query.
* Process the result set.
* Close the connection.

**Exaplain the JDBC Architecture.**

The JDBC Architecture consists of two layers:

* The JDBC API, which provides the **application-to-JDBC Manager** connection.
* The JDBC Driver API, which supports the **JDBC Manager-to-Driver** Connection.

The JDBC API uses a driver manager and database-specific drivers to provide transparent connectivity to heterogeneous databases. The JDBC driver manager ensures that the correct driver is used to access each data source. The driver manager is capable of supporting multiple concurrent drivers connected to multiple heterogeneous databases. The location of the driver manager with respect to the JDBC drivers and the Java application is shown in Figure 1.



**What are the main components of JDBC ?**

The life cycle of a servlet consists of the following phases:

* **DriverManager**: Manages a list of database drivers. Matches connection requests from the java application with the proper database driver using communication subprotocol. The first driver that recognizes a certain subprotocol under JDBC will be used to establish a database Connection.
* **Driver**: The database communications link, handling all communication with the database. Normally, once the driver is loaded, the developer need not call it explicitly.
* **Connection**: 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** : Encapsulates an SQL statement which is passed to the database to be parsed, compiled, planned and executed.
* **ResultSet**: The ResultSet represents set of rows retrieved due to query execution.

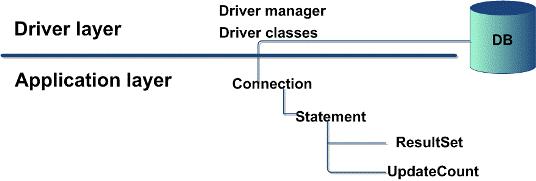
**.How the JDBC application works?**

A JDBC application can be logically divided into two layers:

1. **Driver layer**

2. **Application layer**

* Driver layer consists of DriverManager class and the available JDBC drivers.
* The application begins with requesting the DriverManager for the connection.
* An appropriate driver is choosen and is used for establishing the connection. This connection is given to the application which falls under the application layer.
* The application uses this connection to create Statement kind of objects, through which SQL commands are sent to backend and obtain the results.



**Figure 2: JDBC Application**

**3.Which is superclass of Exception?**

**"Throwable"**, the parent class of all exception related classes.

**4.What are the advantages of using exception handling?**

Exception handling provides the following advantages over "traditional" error management techniques:

* Separating Error Handling Code from "Regular" Code.
* Propagating Errors Up the Call Stack.
* Grouping Error Types and Error Differentiation.

**5.What are the types of Exceptions in Java**

There are two types of exceptions in Java, unchecked exceptions and checked exceptions.

* **Checked exceptions**: A checked exception is some subclass of Exception (or Exception itself), excluding class RuntimeException and its subclasses. Each method must either handle all checked exceptions by supplying a catch clause or list each unhandled checked exception as a thrown exception.
* **Unchecked exceptions:** All Exceptions that extend the RuntimeException class are unchecked exceptions. Class Error and its subclasses also are unchecked.