JDBC APIS

Chapter 2

JDBC APIs:

JDBC APIs:

If any java application or an applet wants to connect with a database then there are various classes and interfaces available in java.sql package.

Depending on the requirements these classes and interfaces can be used.

Some of them are list out the below which are used to perform the various tasks with database as well as for connection.

Class or Interface	Description
Java.sql.Connection	Create a connection with specific database
Java.sql.DriverManager	The task of DriverManager is to manage the database driver
Java.sql.Statement	It executes SQL statements for particular connection and retrieve the results
Java.sql.PreparedStatement	It allows the programmer to create prepared SQL statements
Java.sql.CallableStatement	It executes stored procedures
Java.sql.ResultSet	This interface provides methods to get result row by row generated by SELECT statements

The example program for Statement interface and its methods are given in next chapter for different databases.

The Connection interface:

The Connection interface used to connect java application with particular database.

After crating the connection with database we can execute SQL statements for that particular connection using object of Connection and retrieve the results.

The interface has few methods that makes changes to the database temporary or permanently. The some methods are as given below.

Method	Description
void close()	This method frees an object of type Connection from database and other JDBC resources.
void commit()	This method makes all the changes made since the last commit or rollback permanent. It throws SQLExeception.
Statement createStatement()	This method creates an object of type Statement for sending SQL statements to the database. It throws SQLExeception.
boolean isClosed()	Return true if the connection is close else return false.
CallableStatement prepareCall(String s)	This method creates an object of type CallableStatement for calling the stored procedures from database. It throws SQLExeception.
PreparedStatement prepareStatement(String s)	This method creates an object of type PrepareStatement for sending dynamic (with or without IN parameter) SQL statements to the database. It throws SQLExeception.
void rollback()	This method undoes all chang[illegible]de to the database.

The example program for Connection interface and its methods are given in next chapter for different databases.

Statement Interface:

The Statement interface is used for to execute a static query.

It's a very simple and easy so it also calls a "**Simple Statement**".

The statement interface has several methods for execute the SQL statements and also get the appropriate result as per the query sent to the database.

Some of the most common methods are as given below

Method	Description
void close()	This method frees an object of type Statement from database and other JDBC resources.
boolean execute(String s)	This method executes the SQL statement specified by s. The getResultSet() method is used to retrieve the result.
ResultSet getResultet()	This method retrieves the ResultSet that is generated by the execute() method.
ResultSet executeQuery(String s)	This method is used to execute the SQL statement specified by s and returns the object of type ResultSet.
int getMaxRows()	This method returns the maximum number of rows those are generated by the executeQuery() method.
Int executeUpdate(String s)	This method executes the SQL statement specified by s. The SQL statement may be a SQL insert, update and delete statement.

The example program for Statement interface and its methods are given in next chapter for different databases.

The Prepared Statement Interface:

The Prepared Statement interface is used to execute a dynamic query (parameterized SQL statement) with IN parameter.

IN Parameter:-

In some situation where we need to pass different values to an query then such values can be specified as a "?" in the query and the actual values can be passed using the setXXX() method at the time of execution.

Syntax :

setXXX(integer data ,XXX value);

Where XXX means a data type as per the value we want to pass in the query.

For example,

```
String query = "Select * from Data where ID = ? and Name = ? ";
PreparedStatement ps = con.prepareStatement(query);
        ps.setInt(1, 1);
        ps.setString(2, "Ashutosh Abhangi");
```

The Prepared statement interface has several methods to execute the parameterized SQL statements and retrieve appropriate result as per the query sent to the database.

Some of the most common methods are as given below

Method	Description
void close()	This method frees an object of type Prepared Statement from database and other JDBC resources.
boolean execute()	This method executes the dynamic query in the object of type **Prepared Statement**.The getResult() method is used to retrieve the result.
ResultSet executeQuery()	This method is used to execute the dynamic query in the object of type **Prepared Statement** and returns the object of type ResultSet.
Int executeUpdate()	This method executes the SQL statement in the object of type **Prepared Statement**. The SQL statement may be a SQL insert, update and delete statement.
ResultSetMetaData getMetaData()	The ResultSetMetaData means a deta about the data of ResultSet.This method retrieves an object of type ResultSetMetaData that contains information about the columns of the ResultSet object that will be return when a query is execute.
int getMaxRows()	This method returns the maximum number of rows those are generated by the executeQuery() met

The example program for Prepared Statement interface and its methods are given in next chapter for different databases.