Web Technology
(KCS-602)
Unit 4
Prepared Statements

Prepared By

Abhishek Kesharwani

Assistant Professor,UCER Naini,Allahabad

Index

- JDBC Connection

5 Steps to connect to the database in java

There are 5 steps to connect any java application with the database in java using JDBC.

They are as follows

- Register the driver class
- Creating connection
- Creating statement
- Executing queries
- Closing connection

1) Register the driver class

The forName() method of Class class is used to register the driver class.

This method is used to dynamically load the driver class.

Syntax of forName() method

public static void forName(String className)

throws ClassNotFoundException

2) Create the connection object

The getConnection() method of DriverManager class is used to establish connection with the database.

Syntax of getConnection() method

1) **public static** Connection getConnection(String url)

throws SQLException

2) **public static** Connection getConnection(String url ,String name,String password)

throws SQLException

3) Create the Statement object

The createStatement() method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database.

Syntax of createStatement() method

public Statement createStatement()

throws SQLException

Example to create the statement object

Statement stmt=con.createStatement();

4) Execute the query

The executeQuery() method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table.

Syntax of executeQuery() method

public ResultSet executeQuery(String sql)

throws SQLException

Example to execute query

```
ResultSet rs=stmt.executeQuery("select * from emp");
while(rs.next()){
System.out.println(rs.getInt(1)+" "+rs.getString(2));
}
```

5) Close the connection object

By closing connection object statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection.

Syntax of close() method

public void close()**throws** SQLException

Example to close connection

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

Example to connect to the mysql database

we are using MySql as the database. So we need to know following informations for the mysql database:

- **Driver class:** The driver class for the mysql database is **com.mysql.jdbc.Driver**.
- **Connection URL:** The connection URL for the mysql database is **jdbc:mysql://localhost:3306/united**
- where jdbc is the API, mysql is the database, localhost is the server name on which mysql is running, we may also use IP address, 3306 is the port number and united is the database name.
- **Username:** The default username for the mysql database is **root**.
- **Password:** Password is given by the user at the time of installing the mysql database. In this example, we are going to use admin as the password.

Let's first create a table in the mysql database, but before creating table, we need to create database first.

- create database united;
- use united;
- create table emp(id **int**(10),name varchar(40), age **int**(3));

```
import java.sql.*;
class MysqlCon{
public static void main(String args[]){
try{
Class.forName("com.mysql.jdbc.Driver");
  Connection con=DriverManager.getConnection(
"jdbc:mysql://localhost:3306/united","root",“admin");
  Statement stmt=con.createStatement();
ResultSet rs=stmt.executeQuery("select * from emp");
while(rs.next())
System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getS
    tring(3));
con.close();
}catch(Exception e){ System.out.println(e);}
 }
}
```

To connect java application with the mysql database mysqlconnector.jar file is required. loaded.

DriverManager class

The DriverManager class acts as an interface between user and drivers.

It keeps track of the drivers that are available and handles establishing a connection between a database and the appropriate driver.

public static Connection getConnection(String url,String userName,String password)

is used to establish the connection with the specified url, username and password.

Connection interface

The Connection interface is a factory of Statement, PreparedStatement, and DatabaseMetaData i.e. object of Connection can be used to get the object of Statement and DatabaseMetaData.

The Connection interface provide many methods for transaction management like commit(),rollback() etc.

By default, connection commits the changes after executing queries.

Commonly used methods of Connection interface:

1) **public Statement createStatement():** creates a statement object that can be used to execute SQL queries.
2) **public void commit():** saves the changes made since the previous commit/rollback permanent.
3) **public void rollback():** Drops all changes made since the previous commit/rollback.

Statement interface

The **Statement interface** provides methods to execute queries with the database.

Commonly used methods of Statement interface:

1) **public ResultSet executeQuery(String sql):** is used to execute SELECT query. It returns the object of ResultSet.
2) **public int executeUpdate(String sql):** is used to execute specified query, it may be create, drop, insert, update, delete etc.
3) **public boolean execute(String sql):** is used to execute queries that may return multiple results.

ResultSet interface

The object of ResultSet maintains a cursor pointing to a particular row of data. Initially, cursor points to before the first row.

By default, ResultSet object can be moved forward only and it is not updatable.

Commonly used methods of ResultSet interface

1) public boolean next():

is used to move the cursor to the one row next from the current position.

2) public boolean previous():

is used to move the cursor to the one row previous from the current position.

3) public boolean first():

is used to move the cursor to the first row in result set object.

4) public boolean last():

is used to move the cursor to the last row in result set object.

5) public boolean absolute(int row):

is used to move the cursor to the specified row number in the ResultSet object.

7) public int getInt(int columnIndex):

is used to return the data of specified column index of the current row as int.

8) public int getInt(String columnName):

is used to return the data of specified column name of the current row as int.

9) public String getString(int columnIndex):

is used to return the data of specified column index of the current row as String.

10) public String getString(String columnName):

is used to return the data of specified column name of the current row as String.

AKTU Semester Questions

- What are the various types of JDBC drivers? Write steps to connect database with the web application using JDBC.(2019-20)