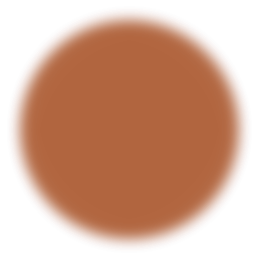
**JAC444 - Lecture 13**

Java DataBase Connectivity Segment 2 - Query DB

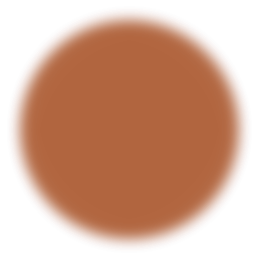
**JDBC technology**

Four steps are used in working with JDBC

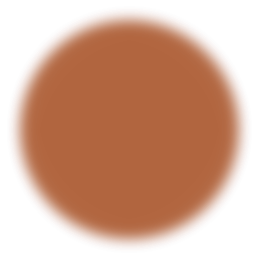
Connect to the database



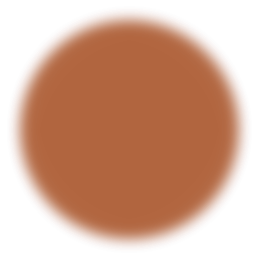
3



1



2



4

Create a statement and execute the query

Look at the result set

Close connection

**Establish the Connection**

* DriverManager has a static method called getConnection
* It returns a Connection object

**Connection connection =**

**DriverManager.getConnection(**

***String url,***

***String user,***

***String passwd***

**);**

**Create Statement**

* There is a statement object created from the **Connection** object.
* A **Statement** object is used to send queries and command to database.

**Statement stmt = conn.createStatement();**

**String query = “SELECT \* FROM MyTable”; ResultSet rs = stmt.executeQuery(query);**

**stmt.executeUpdate(..);** to modify a database

**stmt.execute();** execute arbitrary command **stmt.setQueryTime();** set delay to wait for results

**Process the Results**

* TheResultSet class implements a collection of type Set and you can use it to process one row at the time.

**ResultSet rs = stmt.executeQuery(query); while ( rs.next() ) {**

**System.out.println( rs.getString(…) );**

**}**

* There is a class **ResultSetMetaData**that helps you determine the number, names and types of column in the **ResultSet**

**ResultSetMetaData rsm = rs.getMetaData(); int colCount = rsm.getColumnCount();**

**String colName = rsm.getColumnName(col); int colType = rsm.getColumnType();**

**Process Exceptions**

**try { // Code that could generate an exception goes here.**

**} catch(SQLException ex) {**

**System.err.println("SQLException:“+ ex.getMessage()); }**

**try { Class.forName("myDriverClassName");**

**} catch(java.lang.ClassNotFoundException e) {**

**System.err.print("ClassNotFoundException: ");**

**System.err.println(e.getMessage());**

**}**

**Conclusion**

1. JDBC technology is an API that lets you access virtually any tabular data source from the Java programming language.
2. With a JDBC technology-enabled driver, a developer can easily connect all corporate data even in a heterogeneous environment.
3. Servlet JDBC is the further extension of the servlet functionality by the integration of servlet programming technique for the interactive access and update of a remote database engine using JDBC technology.
4. Java Database Connectivity is a programming interface that lets developers using the Java programming language gain access to a wide range of databases and other data sources