CS6308- Java Programming

V P Jayachitra
Assistant Professor
Department of Computer Technology
MIT Campus
Anna University

JDBC API

- JDBC API is a part of the Java platform.
- provides programmatic access to relational data (rdbms)from the Java programming language.
- Using the JDBC API, applications can execute SQL statements, retrieve results, and propagate changes back to an underlying database.
- JDBC helps you to write Java applications that manage these three programming activities:
  - Connect to a data source, like a database
  - Send queries and update statements to the database
  - Retrieve and process the results received from the database in answer to your query

JDBC Architecture

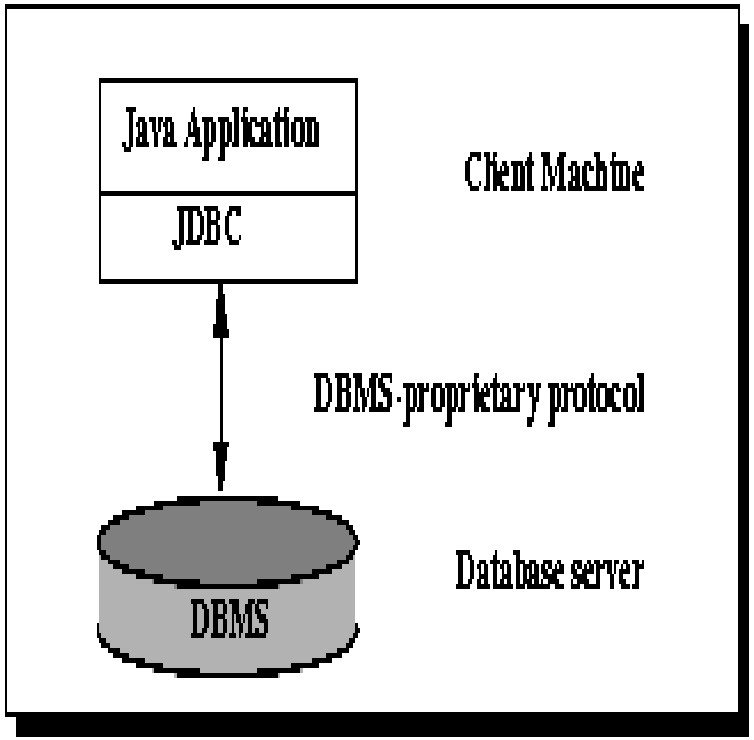


Figure 1: Two-tier Architecture for Data Access.

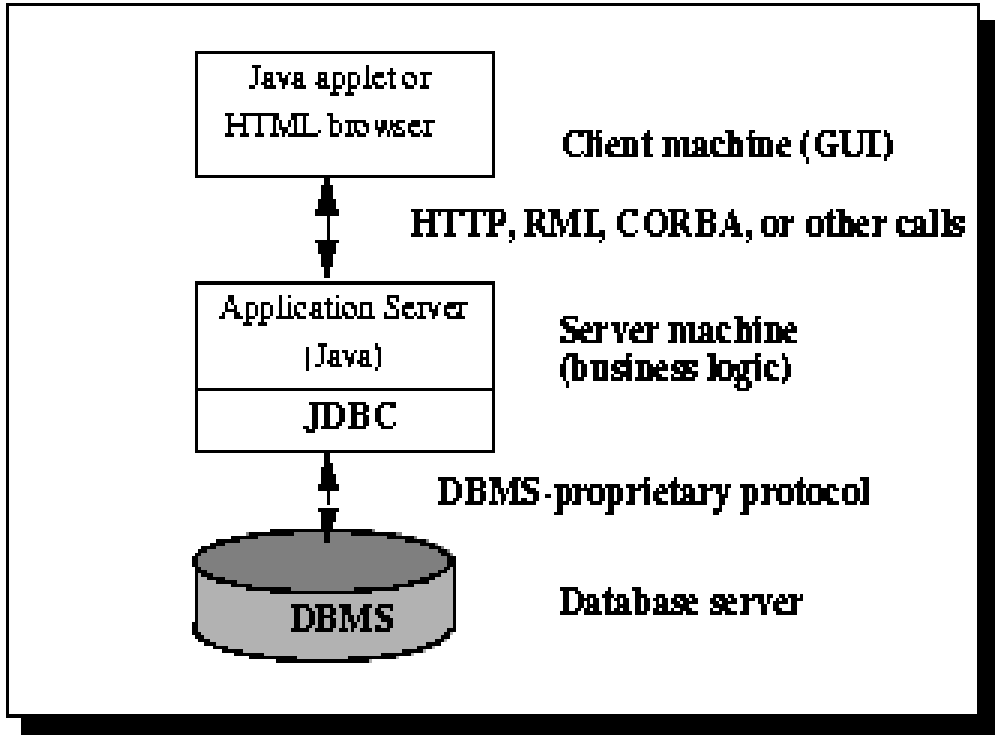


Figure 2: Three-tier Architecture for Data Access.

Two-tier vs Three tier model

two-tier model

- a Java applet or application talks directly to the data source.
- This requires a JDBC driver that can communicate with the data base.
- A user's commands are delivered to the database and the results are sent back to the user.
- client/server configuration

three-tier model

- commands are sent to a "middle tier" of services, which then sends the commands to the data base.
- The data base processes the commands and sends the results back to the middle tier, which then sends them to the user.
- Web based application

Establishing A Connection

- a connection between program(client) and the database(server) involves two steps:

1. Load the oracle driver

- Class.forName("oracle.jdbc.driver.OracleDriver")

2. **Establish connection**

```
Connection con= DriverManager.getConnection(
        "jdbc:oracle:thin:@localhost:1521:xe", "system", "123456");
```

Username and password of oracle database

Creating JDBC Statements

- A statement is used to send and execute SQL statements to a database.
- Three kinds of Statements
- **Statement:** Execute simple sql queries without parameters.
  Statement createStatement()
- **Prepared Statement:** Execute precompiled sql queries with or without parameters.
  PreparedStatement prepareStatement(String sql)
- **Callable Statement:** Execute a call to a database stored procedure.
  CallableStatement prepareCall(String sql)

Statement

- The Statement class has three methods for executing statements:

  executeQuery()

  executeUpdate()

  execute().
- For a SELECT statement,
  - the method to use is executeQuery .
- For statements that create or modify tables,
  - the method to use is executeUpdate.
  - execute() executes an SQL statement that is written as String object.

JDBC connection code fragment

- Connection con = DriverManager.getConnection("jdbc:myDriver:myDatabase", username, password);
- Statement stmt = con.createStatement();
- ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM Table1");

```
while (rs.next()) {
    int x = rs.getInt("a");
    String s = rs.getString("b");
    float f = rs.getFloat("c");
    }
```

SQL

- SQL is a structured query language for storing, manipulating and retrieving data in databases.
- The SQL SELECT Statement
  - The SELECT statement is used to select data from a database.
  - Syntax
    - SELECT *column1, column2, ...*FROM *table_name*;
    - SELECT * FROM *table_name*; --to select all the fields available in the table
  - Example
    - SELECT CustomerName, City FROM Customers;
    - SELECT * FROM Customers;

Select with WHERE Clause

- The WHERE clause is used to filter records.
- The WHERE clause is used to extract only those records that fulfill a specified condition.
- Syntax
  - SELECT *column1*, *column2, ...*FROM *table_name* WHERE *condition*;
- Example
  - SELECT * FROM Customers WHERE Country='Mexico';
  - SELECT * FROM Customers WHERE CustomerID=1;

SQL INSERT INTO Statement

The INSERT INTO statement is used to insert new records in a table.

- Syntax
  - INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...)
    VALUES (*value1*, *value2*, *value3*, ...);
  - INSERT INTO *table_name*
    VALUES (*value1*, *value2*, *value3*, ...);

```
SQL> INSERT INTO Persons
  2  values(123,'varma','surya','Chrompet','Chennai');

1 row created.

SQL> _
```

- Example
  - INSERT INTO Customers (CustomerName, City, Country)
    VALUES ('Cardinal', 'Stavanger', 'Norway');

SQL UPDATE Statement

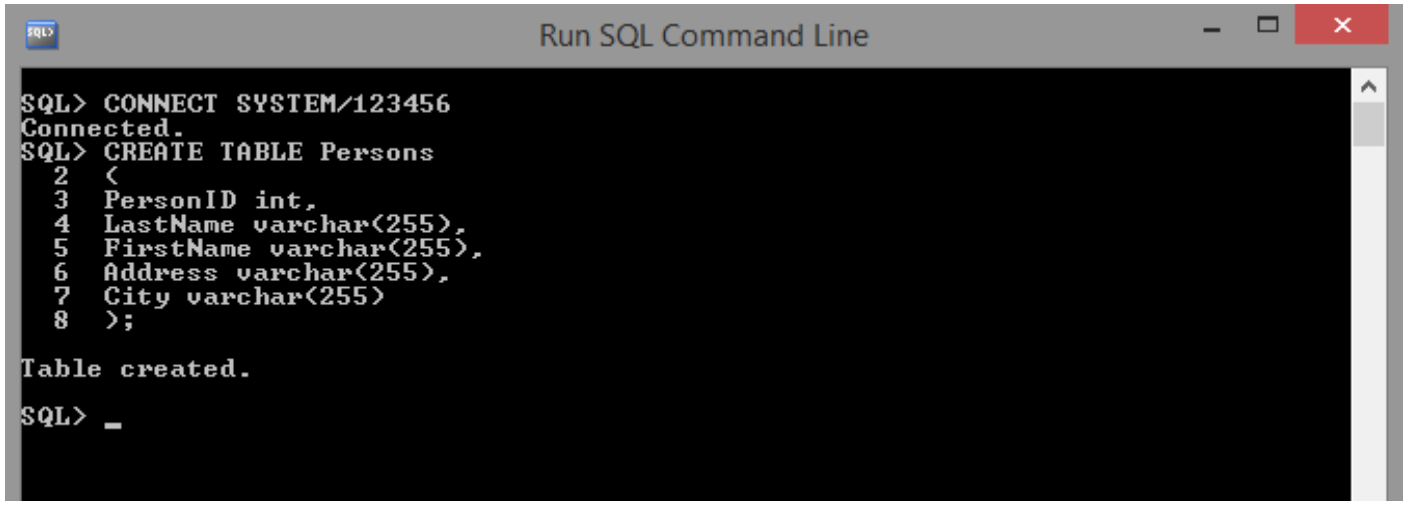
- The UPDATE statement is used to modify the existing records in a table.
- Syntax
  - UPDATE *table_name*
    SET *column1 = value1, column2 = value2, ...*
    WHERE *condition*;
- Example
  - UPDATE Customers
    SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
    WHERE CustomerID = 1;

SQL DELETE Statement

- The DELETE statement is used to delete existing records in a table.
- Syntax
  - DELETE FROM *table_name* WHERE *condition*;
- Example
  - DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

SQL CREATE TABLE Statement

- The CREATE TABLE statement is used to create a new table in a database.
- Syntax
  - CREATE TABLE *table_name* (
    column1 datatype,
    column2 datatype,
    column3 datatype,
    ....
    );



- Example

```
CREATE TABLE Persons (PersonID int,LastName varchar(55),FirstName varchar(55),
   Address varchar(55),City varchar(55)
);
```

SQL Data Types

Data type	Description
CHAR(size)	Holds a fixed length string (can contain letters, numbers, and special characters). The fixed size is specified in parenthesis. Can store up to 255 characters
VARCHAR(size)	Holds a variable length string (can contain letters, numbers, and special characters). The maximum size is specified in parenthesis. Can store up to 255 characters. **Note:** If you put a greater value than 255 it will be converted to a TEXT type
INT(size)	-2147483648 to 2147483647 normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DOUBLE(size,d)	A large number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DECIMAL(size,d)	A DOUBLE stored as a string , allowing for a fixed decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter

ESTABLISH CONNECTION TO ORACLE DATABASE

```
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.SQLException;

public class OJDBC {
   public static void main(String[] argv) {
      System.out.println("Oracle JDBC Connection Testing ");
       //load oracle driver
      try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
             }
      catch (ClassNotFoundException e) {
            System.out.println("Oracle JDBC Driver?"); e.printStackTrace(); return;       }
      System.out.println("Oracle JDBC Driver Registered!");
         //establish connection hostname:localhost port no :1521, username :system, pwd: 123456
      Connection connection = null;
      try {
            connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "123456");
            }
     catch (SQLException e) {
            System.out.println("Connection Failed! ");   e.printStackTrace();    return;}
     if (connection != null)
            System.out.println("connect to your database now!");
            else
            System.out.println("Failed to make connection!"); }}
```

```
F:\java>javac OJDBC.java

F:\java>java -cp f:/java OJDBC
Oracle JDBC Connection Testing
Oracle JDBC Driver Registered!
connect to your database now!

F:\java>_
```

Display Person table content using select query

```
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
public class OJDBC {
   public static void main(String[] argv) {
      System.out.println("Oracle JDBC Connection Testing ");
                    //load oracle driver
      try {
         Class.forName("oracle.jdbc.driver.OracleDriver");
      } catch (ClassNotFoundException e) {
         System.out.println("Oracle JDBC Driver?"); e.printStackTrace(); return;      }
      System.out.println("Oracle JDBC Driver Registered!");
                    //establish connection hostname:localhost port no :1521, username :system, pwd: 123456
      Connection connection = null;
      try {
         connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "123456
      } catch (SQLException e) {
         System.out.println("Connection Failed! "); e.printStackTrace(); return;      }
      if (connection != null)
         System.out.println("connect to your database nnow!");       else  System.out.println("Failed to make connection!");
       try{
                    Statement st=connection.createStatement(); // Execute SQL select statement to fetch records from table.
                    ResultSet rs=st.executeQuery("select PersonId, LastName, FirstName, Address,City from Persons");
                    while (rs.next()){
                              int Id = rs.getInt(1);
                              String lname = rs.getString(2);
                              String fname = rs.getString(3);
                              String address = rs.getString(4);
                              String city = rs.getString(5);
                              System.out.println("--------------------");
                              System.out.println("Id:" + Id); System.out.println("Lname:" + lname);
                              System.out.println("FName:" + fname);System.out.println("Address:" + address);
                              System.out.println("city:" + city);         }        }
      catch(SQLException e)  { System.out.println(e);}
   }
}
```

```
F:\java>java -cp f:/java OJDBC
Oracle JDBC Connection Testing
Oracle JDBC Driver Registered!
connect to your database nnow!
--------------------
Id:123
Lname:varma
FName:surya
Address:Chrompet
city:Chennai
--------------------
Id:456
Lname:darshu
FName:rajini
Address:bagya
city:vellore

F:\java>
```

Add records to Person table content using insert query

```
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
public class OJDBC {
   public static void main(String[] argv) {
      System.out.println("Oracle JDBC Connection Testing ");
                  //load oracle driver
      try {
         Class.forName("oracle.jdbc.driver.OracleDriver");
      } catch (ClassNotFoundException e) {
         System.out.println("Oracle JDBC Driver?"); e.printStackTrace(); return;      }
      System.out.println("Oracle JDBC Driver Registered!");
                  //establish connection hostname:localhost port no :1521, username :system, pwd: 123456
      Connection connection = null;
      try {
         connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "123456");
      } catch (SQLException e) {
         System.out.println("Connection Failed! "); e.printStackTrace(); return;      }
      if (connection != null)
         System.out.println("connect to your database nnow!");      else  System.out.println("Failed to make connection!");
         try{
```

```
F:\java>
F:\java>java -cp f:/java OJDBC
Oracle JDBC Connection Testing
Oracle JDBC Driver Registered!
connect to your database nnow!
rows inserted: 1
--------------------
Id:123
Lname:varma
FName:surya
Address:Chrompet
city:Chennai
--------------------
Id:456
Lname:darshu
FName:rajini
Address:bagya
city:vellore
--------------------
Id:789
Lname:ma
FName:Jack
Address:Hangzhou
city:China
```

```
Statement st=connection.createStatement();
int count=st.executeUpdate("insert into Persons values(789, 'ma', 'Jack','Hangzhou', 'China')");
System.out.println("rows inserted: " + count);
Statement st=connection.createStatement(); // Execute SQL select statement to fetch records from table.
ResultSet rs=st.executeQuery("select PersonId, LastName, FirstName, Address,City from Persons");
while (rs.next()){
            int Id = rs.getInt(1);   String lname = rs.getString(2);
            String fname = rs.getString(3);  String address = rs.getString(4);
            String city = rs.getString(5);
            System.out.println("--------------------");
            System.out.println("Id:" + Id); System.out.println("Lname:" + lname);
            System.out.println("FName:" + fname);System.out.println("Address:" + address);
            System.out.println("city:" + city);        }        }
   catch(SQLException e)  { System.out.println(e);}    }}
```

Modify Person table content using update query

```
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
public class OJDBC {
   public static void main(String[] argv) {
      System.out.println("Oracle JDBC Connection Testing ");
                  //load oracle driver
      try {
         Class.forName("oracle.jdbc.driver.OracleDriver");
      } catch (ClassNotFoundException e) {
         System.out.println("Oracle JDBC Driver?"); e.printStackTrace(); return;      }
      System.out.println("Oracle JDBC Driver Registered!");
                  //establish connection hostname:localhost port no :1521, username :system, pwd: 12345
      Connection connection = null;
      try {
         connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "1234
      } catch (SQLException e) {
         System.out.println("Connection Failed! "); e.printStackTrace(); return;      }
      if (connection != null)
         System.out.println("connect to your database nnow!");       else  System.out.println("Failed to make
```

```
--------------------
Id:789
Lname:ma
FName:Jack
Address:Hangzhou
city:China

F:\java>javac OJDBC.java

F:\java>java -cp f:/java OJDBC
Oracle JDBC Connection Testing
Oracle JDBC Driver Registered!
connect to your database nnow!
rows inserted: 1
--------------------
Id:123
Lname:varma
FName:surya
Address:Chrompet
city:Chennai
--------------------
Id:456
Lname:darshu
FName:rajini
Address:bagya
city:vellore
--------------------
Id:1
Lname:ma
FName:Jack
Address:Hangzhou
city:China
```

```
try{
      Statement st=connection.createStatement();
      int count=st.executeUpdate("update Persons set personId=1 where personId=789");
      System.out.println("rows inserted: " + count);
      Statement st=connection.createStatement(); // Execute SQL select statement to fetch records from table.
      ResultSet rs=st.executeQuery("select PersonId, LastName, FirstName, Address,City from Persons");
      while (rs.next()){
            int Id = rs.getInt(1);   String lname = rs.getString(2);
            String fname = rs.getString(3);  String address = rs.getString(4);
            String city = rs.getString(5);
            System.out.println("--------------------");
            System.out.println("Id:" + Id); System.out.println("Lname:" + lname);
            System.out.println("FName:" + fname);System.out.println("Address:" + address);
            System.out.println("city:" + city);        }        }
   catch(SQLException e)  { System.out.println(e);}
```

```java
package jdbc;
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
public class JDBCclass {
    public static void main(String args[]){
        System.out.println("Oracle JDBC Connection Testing ");
        Connection conn=null;
        ResultSet rs=null;
        try{
            conn = DriverManager.getConnection(
                    url: "jdbc:oracle:thin:@localhost:1521:orcl",  user: "system",  password: "Oracle123");
            if (conn != null)
                System.out.println("Connected to the database!");
            else {
                System.out.println("Failed to make connection!");
            }
        } catch (SQLException e) { e.printStackTrace();
        } catch (Exception e) { e.printStackTrace();
        }
        try{
            Statement st=conn.createStatement();
            rs=st.executeQuery( sql: "select studid, stuname  from student");
            while (rs.next()){
                int Id = rs.getInt( columnIndex: 1);
                String lname = rs.getString( columnIndex: 2);
                System.out.println("--------------------");
                System.out.println("Id:" + Id); System.out.println("Lname:" + lname);
                    }
        }catch(SQLException e)  { System.out.println(e);}
    } }
```