**Database Connectivity:**

Connection con=DriverManager.getConnection("jdbc:mysql://" + host + ":" + port + "/demo", "root", "root");

Statement s=con.createStatement();

ResultSet rs=s.executeQuery("select \* from credentials where scenario ='rewardscard'");

Rs.next();

**Sample Program**:

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

public class jdbcconection {

public static void main(String[] args) throws SQLException, ClassNotFoundException {

// TODO Auto-generated method stub

String host="localhost";

String port= "3306";

Connection con=DriverManager.getConnection("jdbc:mysql://" + host + ":" + port + "/demo", "root", "root");

Statement s=con.createStatement();

ResultSet rs=s.executeQuery("select \* from credentials where scenario ='rewardscard'");

while(rs.next())

{

WebDriver driver= new FirefoxDriver();

driver.get("https://login.salesforce.com");

driver.findElement(By.xpath(".//\*[@id='username']")).sendKeys(rs.getString("username"));

driver.findElement(By.xpath(".//\*[@id='password']")).sendKeys(rs.getString("password"));

}

}

}

**Constructor**:

* Constructor is a method without return type with class name.
* This block of code gets executed when an object is created/instantiated.
* If the constructor is not created explicitly in a class, then the default constructor will be executed from java lib.
* Implicit constructor is created without parameters.
* Parametrized constructor will be created with parameters/arguments but when an object is created, then parameters should be given in the constructor to execute the parameterized constructor.
* **package** javaCocepts;
* **import** org.testng.annotations.Test;
* **public** **class** ConstructorDemo {
* **public** ConstructorDemo(){
* System.***out***.println("Im the implicit constructor");
* }
* **public** ConstructorDemo(**int** a, **int** b) {
* System.***out***.println("Im the constructor with parameters");
* }
* @Test
* **public** **void** method(){
* //ConstructorDemo obj=new ConstructorDemo();
* ConstructorDemo obj=**new** ConstructorDemo(3, 9);
* }
* }