Introduction to ODBC and JDBC

Lecture-2

Prof. Hari Mohan Pandey
Assistant Professor, CSE Department
Amity School of Engineering & Technology
hmpandey@amity.edu

Session Objective

- At the end of this lecture learns will be able to
  - Define JDBC and its benefits
  - Explore JDBC product components.
  - JDBC architecture.
  - Sketch of Two-tier and three-tier architecture and their working.

JDBC Introduction

- The JDBC API is a Java API that can access any kind of tabular data, especially data stored in a Relational database.

- JDBC helps in:
  - Connect to data source, like a database
  - Send queries and update a statements to the database.
  - Retrieve and process the results received from the database in answer to your query.

Example-1

```
public void connetToAndQueryDatabase(String username, String password)
{
    Connection con = DriverManager.getConnection("jdbc:myDriver:myDatabase",username,
                     password);
    Statement stm = con.createStatement();
    ResultSet rs = stm.excuteQuery("SELECT a, b, c FROM Table1");
    while(rs.next())
    {
        int x = rs.getInt("a");
        String s = rs.getString("b");
        float f = rs.getFloat("c");
    }
}
```

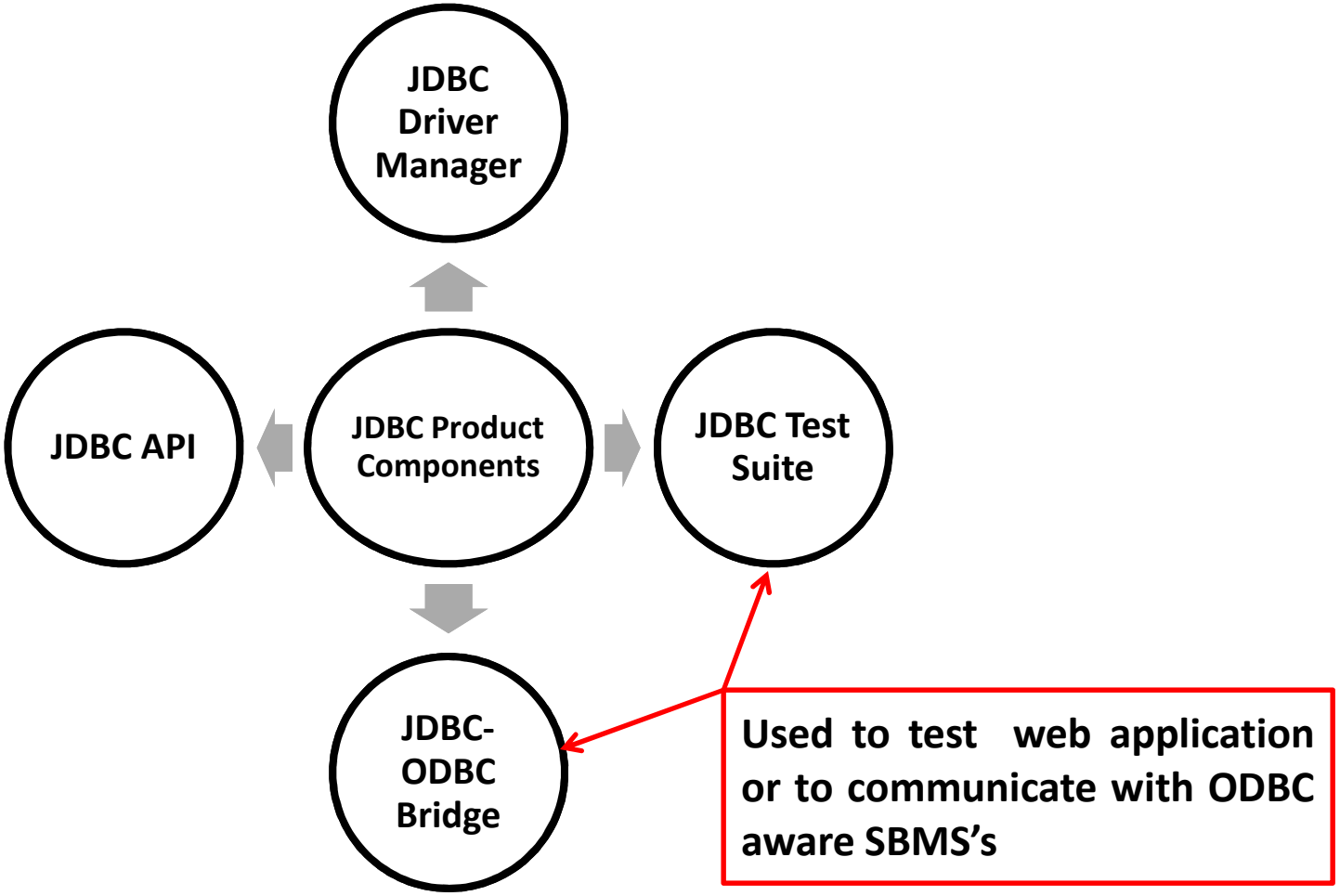
Instantiates a DriverManager object to connect to a database driver and log into the database

ResultSet object that retrieves the results of the query

A Statement object that caries SQL language query to the database

While loop retrieves and displays those results

JDBC Product Components



JDBC API

- JDBC API provides programmatic access to relational database from the Java programming language.
- Using this applications can
  - execute SQL statements,
  - retrieve results, and
  - propagate change back to an underlying data source.
- It can interact with ***multiple data sources in a distributed***, *heterogeneous environment*.
- The JDBC API is divided into two packages: **java.sql** and **javax.sql**
- Both packages are included in the Java SE and Java EE platforms.

JDBC Driver Manager

- The JDBC *DriverManager* class defines **objects** which can **connect Java applications to a JDBC driver.**
- **DriverManager** has traditionally been the **backbone** of the **JDBC architecture.**
- It is quite small and simple
- The Standard Extension packages **javax.naming** and **javax.sql** let you use a **DataSource** object registered with **Java Naming** and **Directory Interface** (JNDI) naming service ***to establish a connection with a data source.***

JDBC Test Suite

- The JDBC driver test suite helps us to determine that JDBC drivers will run the program.

- These test are not comprehensive or exhaustive, but they do exercise many of the important features in the JDBC API.

JDBC-ODBC Bridge

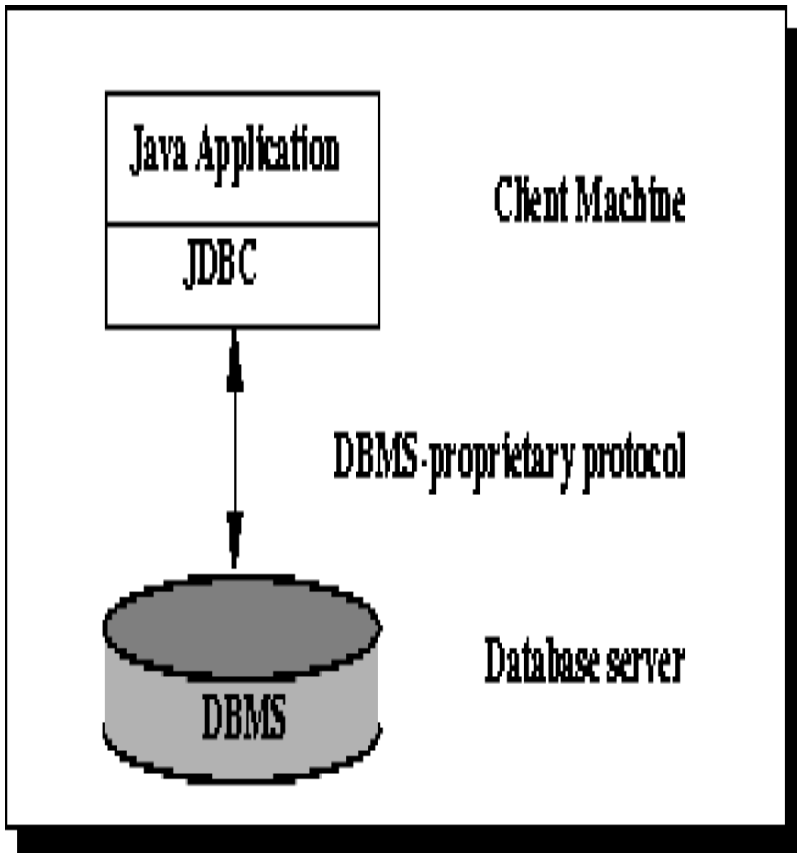
- The Java Software bridge provides JDBC access via ODBC drivers.
- We need to load ODBC binary code onto each machine that uses this driver.
- As a result, the ODBC driver is most appropriate on a corporate network where client installations are not a major problem, or for application server code written in Java in a three-tier architecture.

JDBC Architecture

- Two popular models given:
  - Two-tier model
  - Three-tier model

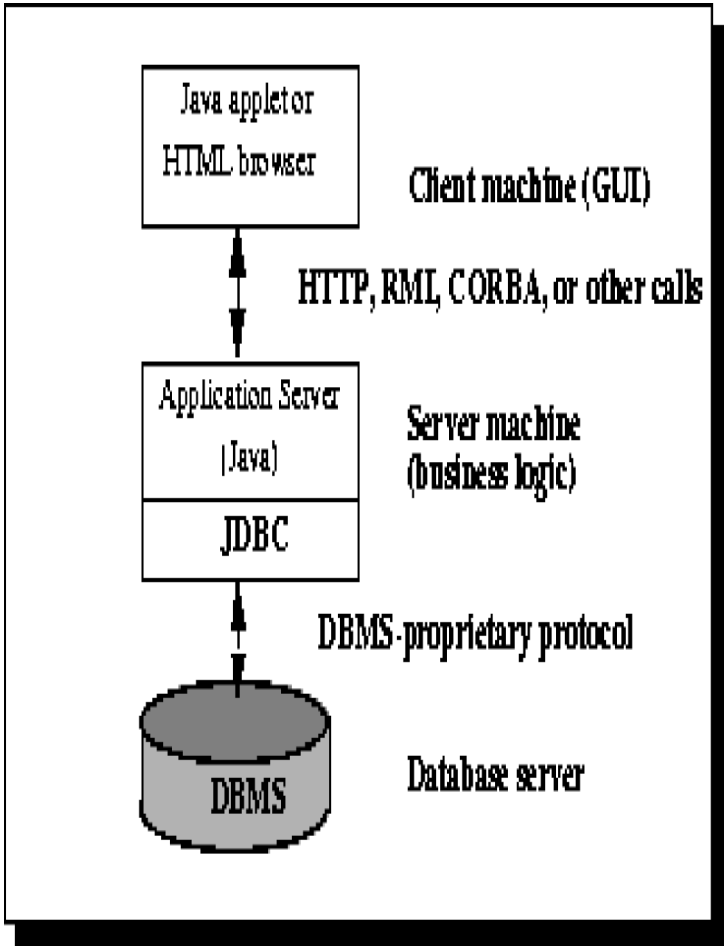
JDBC Architecture: Two-tier Model

- In this, a Java applet or application talks directly to the data source.
- This requires a JDBC driver that can communicate with the particular data source being accessed.
- A user's commands are delivered to the database or other data source, and the results of those statements are sent back to the user.
- The data source may be located on another machine to which the user is connected via a network.



JDBC Architecture: Three-tier Model

- In this model, commands are sent to a "middle tier" of services, which then sends the commands to the data source.
- The data source process the commands and sends the results back to the middle tier, which then sends them to the user.
- MIS directors finds the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data.
- It simplifies the deployment of applications.
- It shows performance advantages.



JDBC Architecture: Three-tier Model

- The middle tier has often written in languages such as C or C++, which offers fast performance.

- However, with the introduction of optimizing compilers that translate Java bytecode into efficient machine-specific code and technologies such as Enterprise JavaBeans, the Java platform is fast becoming the standard platform for middle-tier development.

- This is a big plus, making it possible to take advantage of Java's robustness, multithreading and security features.

Q & A