JDBC

Introduction

Introduction to JDBC

Agenda

Introduction to JDBC

Objectives

At the end of this module, you will be able to:

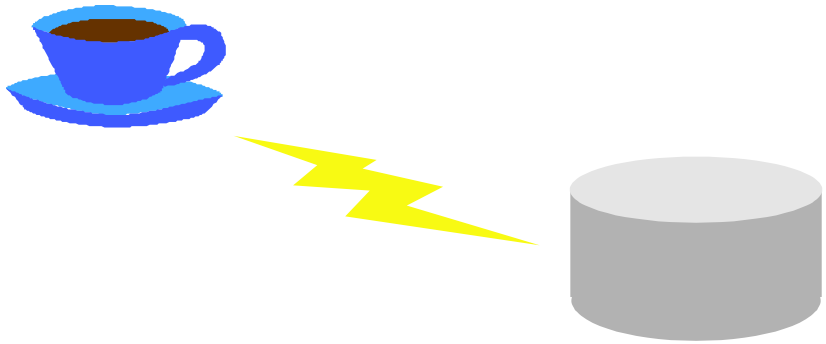
- Examine the role of JDBC in data persistence

4

Introduction to JDBC

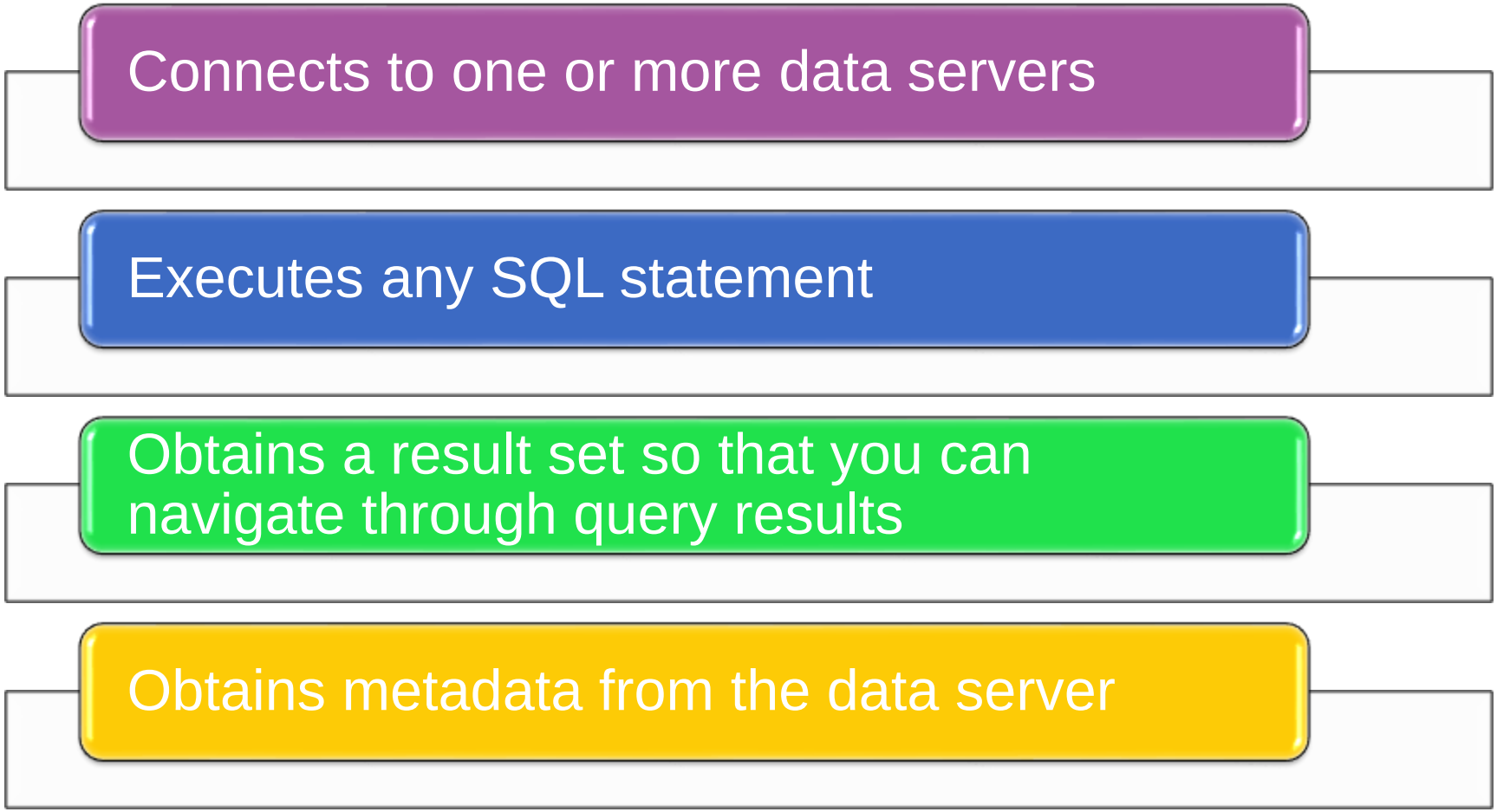
- JDBC is an API that helps a programmer to write java programs to connect to any database, retrieve the data from the database.

- java.sql package contains a set of interfaces that specify the JDBC API

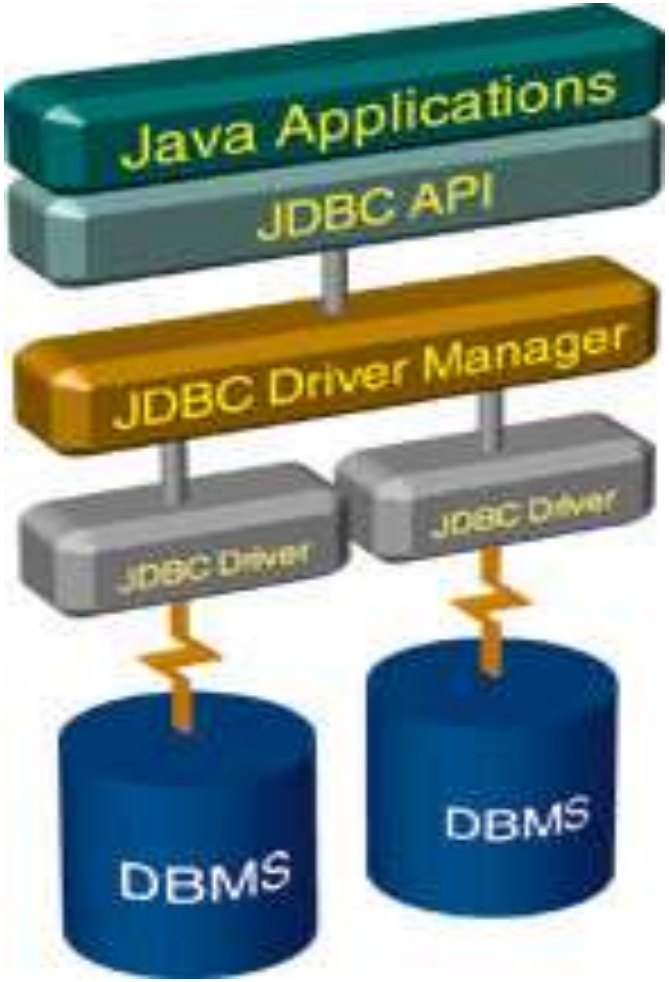


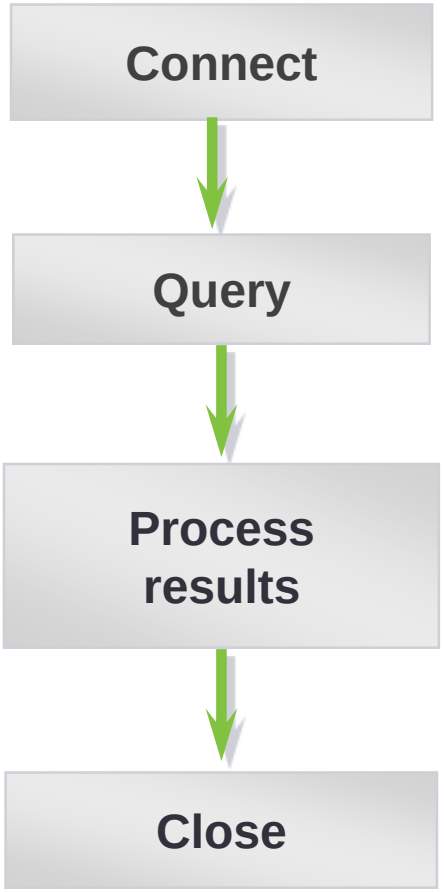
Introduction to JDBC

- Using JDBC, you can write code that:



Architecture and Querying with JDBC

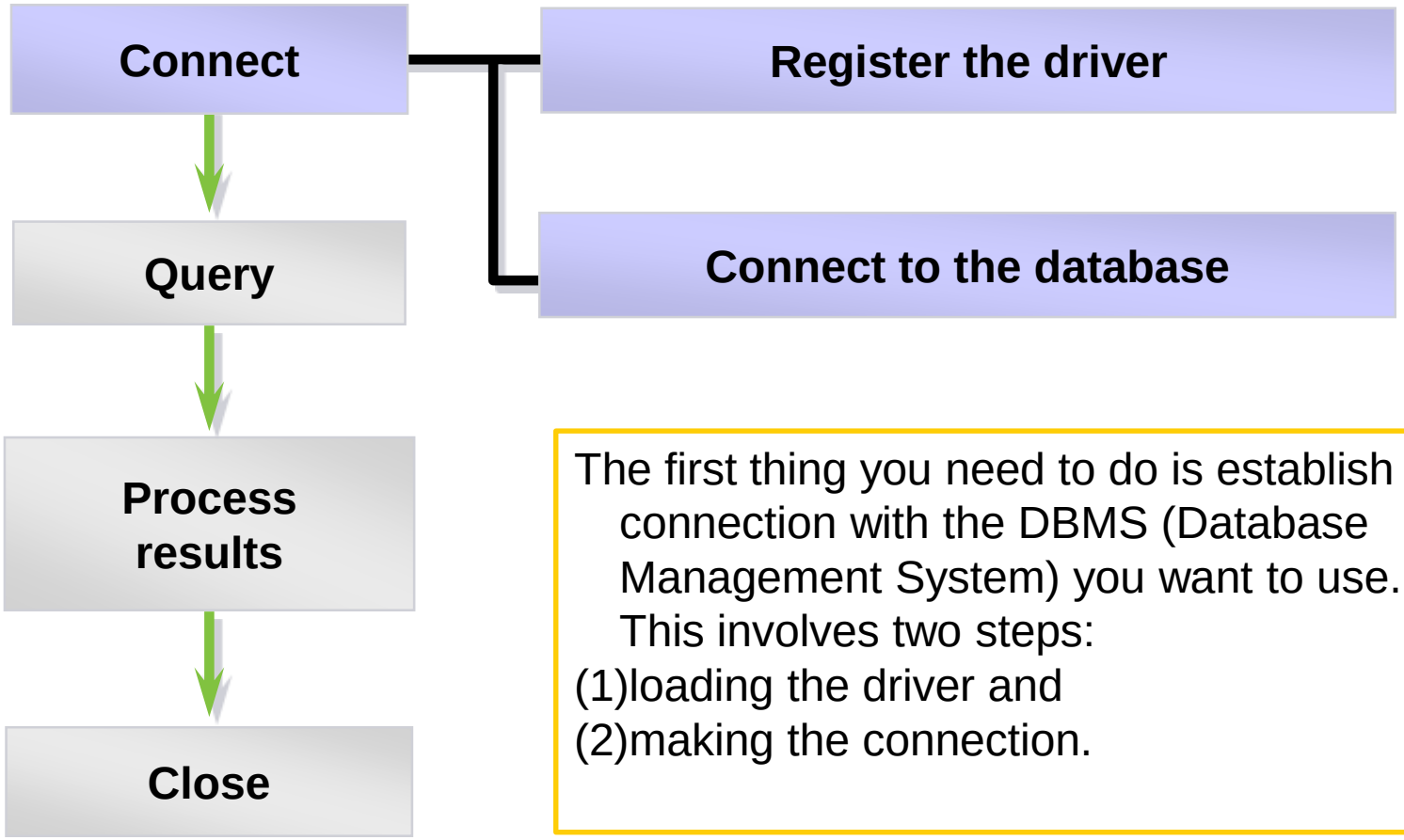




Architecture and Querying with JDBC

- The JDBC API consists of two major sets of interfaces:
- The first is the JDBC API for applications writers, and
- The second is the lower-level JDBC driver API for driver writers.
- As seen from the previous diagram, the DriverManager class is the traditional management layer of JDBC, working between the user and the drivers. It keeps track of the drivers that are available and handles establishing a connection between a database and the appropriate driver. In addition, the DriverManager class attends to things like driver login time limits and the printing of log and tracing messages.

Stage 1: Connect



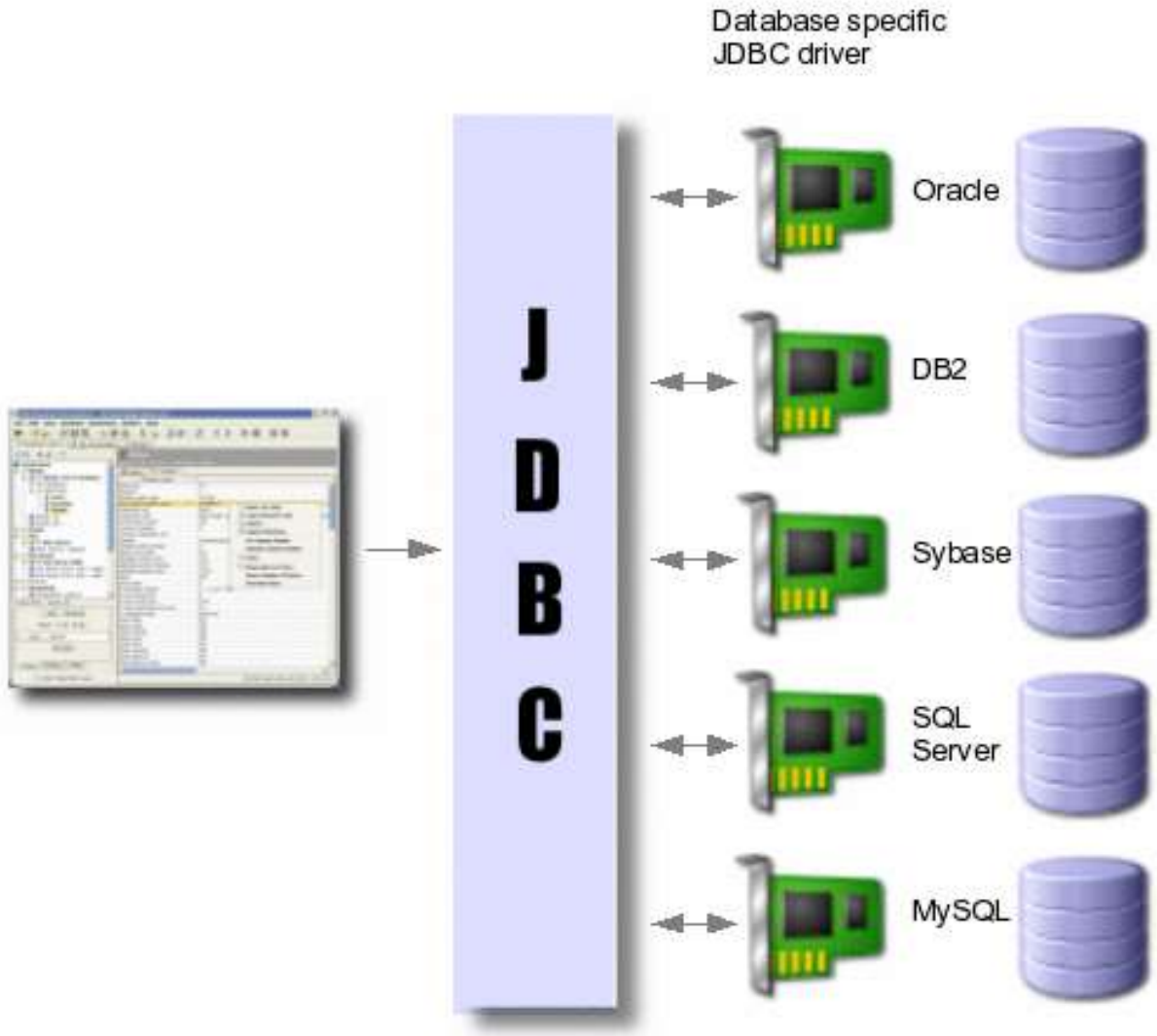
The first thing you need to do is establish a connection with the DBMS (Database Management System) you want to use. This involves two steps:
(1)loading the driver and
(2)making the connection.

9

Connect: A JDBC Driver

Is a set of classes and interfaces, written according to JDBC API to communicate with a database.

Connect: A JDBC Driver



A JDBC Driver

- The JDBC API defines a set of interfaces that encapsulate major database functionality like
- running queries
- processing results
- determining configuration information.
- A database vendor or third-party developer writes a JDBC driver, which is a set of classes that implements these interfaces for a particular database system.
- An application can use a number of drivers interchangeably.
- JDBC drivers are available for most database platforms, from a number of vendors and in a number of different flavors.
- There are four types of drivers, discussed in subsequent sections.

JDBC Driver (Contd.).

JDBC-ODBC Bridge Driver (Type I Driver)

Type 1 drivers use a bridge technology to connect a Java client to an ODBC database system.
The JDBC-ODBC Bridge from Sun is an example of a Type 1 driver.
Type 1 drivers require some sort of non-Java software to be installed on the machine running your code, and they are implemented using native code.

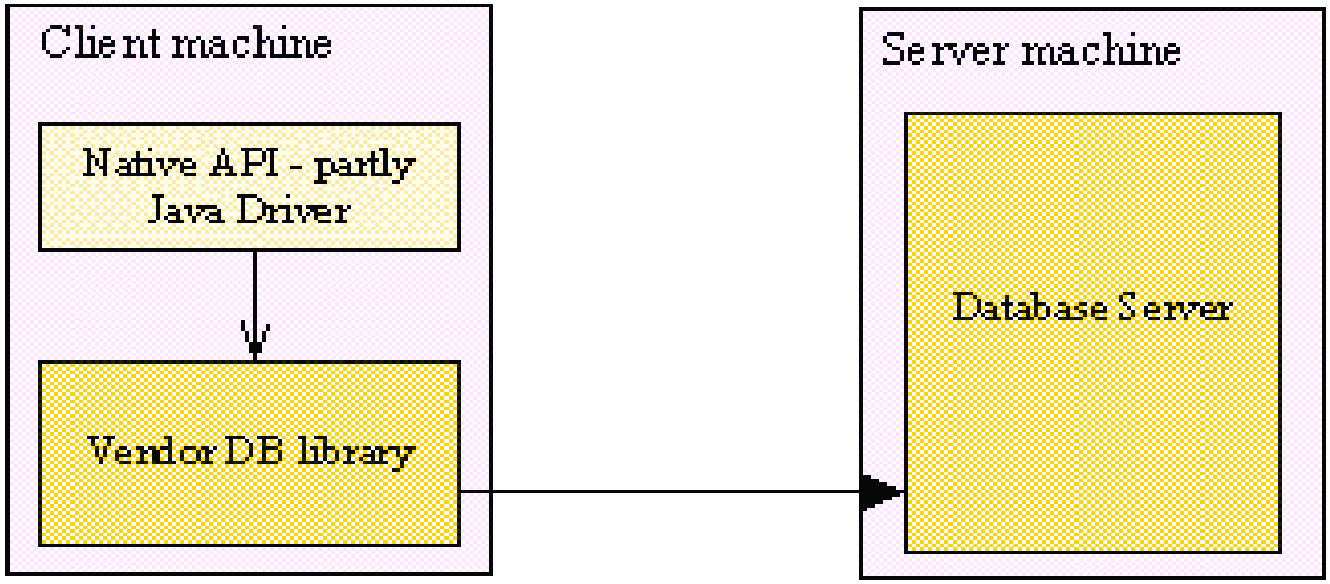
JDBC Driver (Contd.).

- **Characteristics of Type I drivers:**
  - This driver type is the JDBC-ODBC bridge
  - It is limited to running locally
  - Must have ODBC installed on computer
  - Must have ODBC driver for specific database installed on computer
  - Generally can’t run inside an applet because of Native Method calls

- **When to use?**
  - Using this driver you can access any of your existing ODBC databases or databases that don't have JDBC drivers, for example MS Access. This option is good if you already have a database system with ODBC support or for quick system prototyping.
- **The disadvantage** is that you do not get a pure Java solution. You have to install the native ODBC binaries on every system you want to use.
- The JDBC-ODBC driver is provided by JavaSoft as part of the JDK in the sun.jdbc.odbc package. Other vendors are not required to port this package. The bridge is not intended for production environments.

JDBC Driver (Contd.).

Native JDBC Driver (Type II Driver)



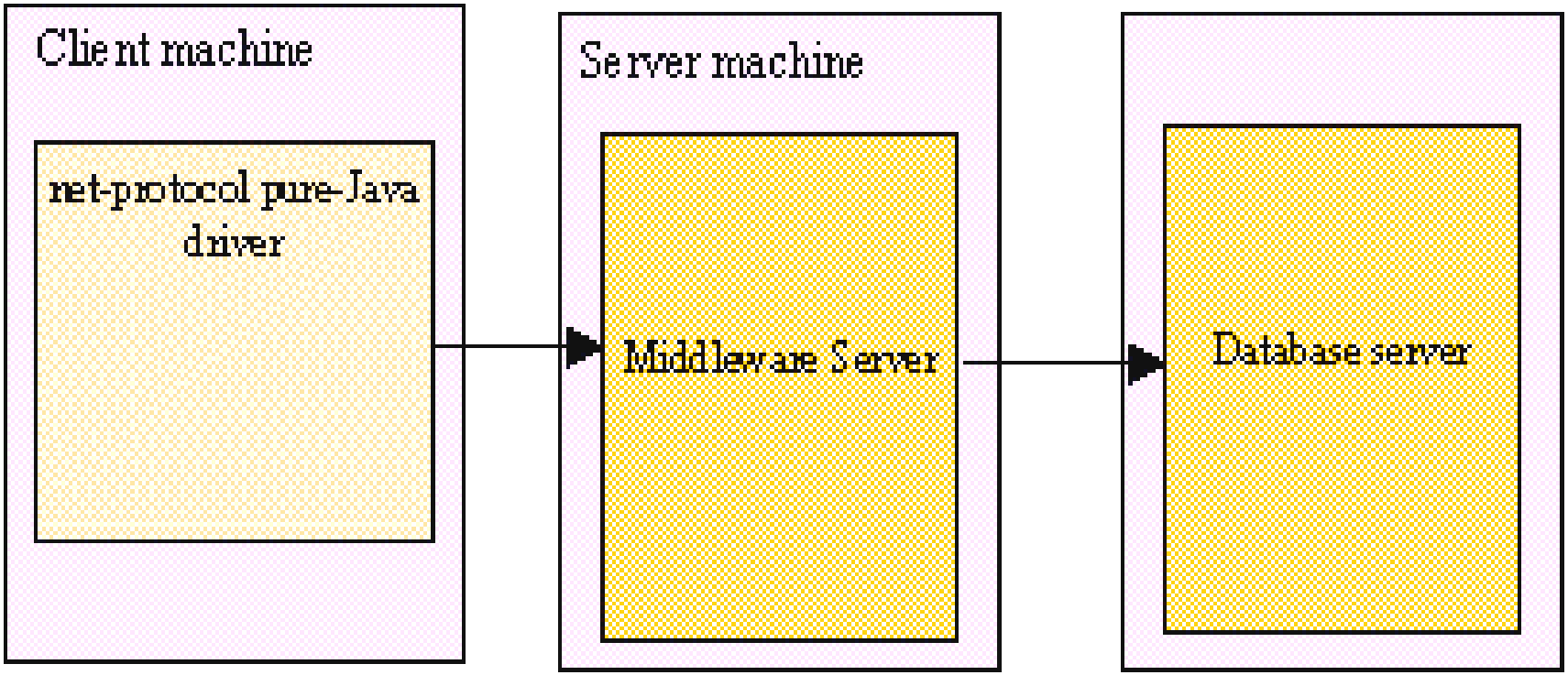
Type 2 drivers use a native code library to access a database, wrapping a thin layer of Java around the native library. For example, with Oracle databases, the native access might be through the Oracle Call Interface (OCI) libraries that were originally designed for C/C++ programmers.

JDBC Driver (Contd.).

- Type 2 drivers are implemented with native code, so they may perform better than all-Java drivers, but they also add an element of risk, as a defect in the native code can crash the Java Virtual Machine.
- **Characteristics of Type II drives:**
  - Native Database library driver
  - Uses Native Database library on computer to access database
  - Generally can't run inside an applet because of Native Method calls
  - Must have database library installed on client
  - example: DB-lib for Sybase, Oracle, MS-SQL server
- **When to use?**
- This driver simply converts the JDBC calls into the native calls for a database. Like for the JDBC-ODBC bridge you have to install the native libraries on every system. Another disadvantage is, that you can not use this driver with untrusted applets.
- But this option is faster than the ODBC bridge, because you directly interact with the database's client libraries.

JDBC Drivers (Contd.).

All Java JDBC Net Drivers (Type III Driver)



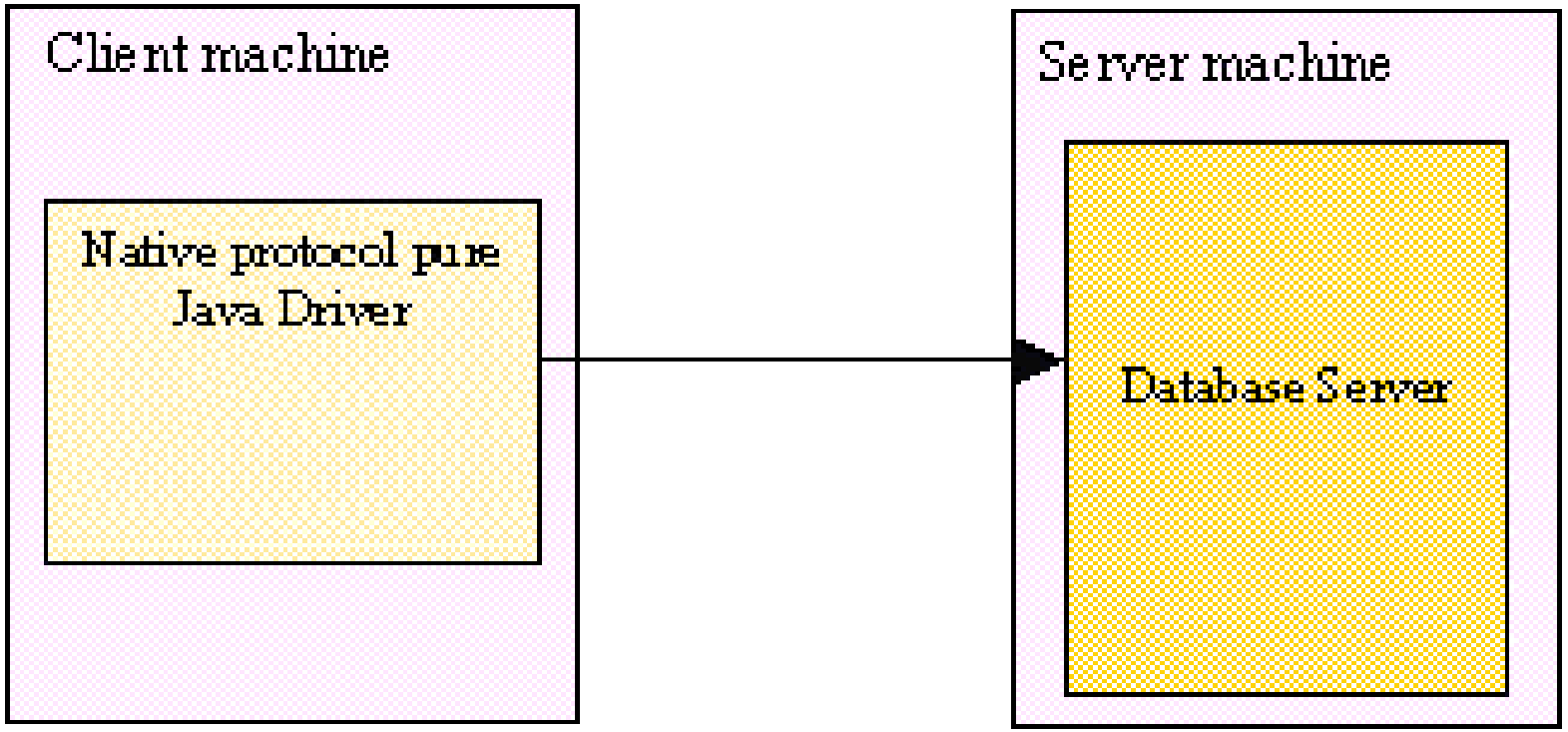
Type 3 drivers define a generic network protocol that interfaces with a piece of custom middleware. The middleware component might use any other type of driver to provide the actual database access. BEA's WebLogic product line (formerly known as WebLogic Tengah and before that as jdbcKona/T3) is an example. These drivers are especially useful for applet deployment, since the actual JDBC classes can be written entirely in Java and downloaded by the client on the fly.

JDBC Drivers (Contd.).

- **Characteristics of Type III driver:**
  - 100% Java Driver, no native methods
  - Does NOT require pre-installation on client
  - Can be downloaded and configured ‘on-the-fly’ just like any Java class file
  - Uses a proprietary protocol for talking with a middleware server
  - Middleware server converts from proprietary calls to DBMS specific calls
- **When to use?**
- Type III driver adds security, caching, and connection control. As this does not require any pre-installation, it can be used in web applications.
- Here the JDBC calls are converted into a network protocol and transmitted to a server which makes the actual database calls. This is the most flexible solution, because the clients are written in pure Java and you can access any database from the middle tier without changing the client. The JDBC drivers are developed by some companies and may be quite costly.

JDBC Drivers (Contd.).

Native Protocol All Java Drivers (Type IV Driver)



Type 4 drivers are written entirely in Java. They understand database-specific networking protocols and can access the database directly without any additional software. These drivers are also well suited for applet programming, provided that the Java security manager allows TCP/IP connections to the database server.

19

JDBC Drivers (Contd.).

- **Characteristics of Type IV driver:**
  - 100% Java Driver, no native methods
  - Does NOT require pre-installation on client
  - Can be downloaded and configured ‘on-the-fly’ just like any Java class file
  - Unlike Type III driver, talks DIRECTLY with DBMS server
  - Converts JDBC calls directly to database specific calls
- **When to use?**
- Type IV drivers need no pre-installation and are hence, truly portable.
  - In this case the JDBC calls are directly converted into the network protocol that is used by a specific database. This is the fastest alternative, because there are no additional layers included and hence most preferred in many of the J2EE or web-based applications.

QUIZ

1. **The Type I driver is also known as**
   a) **Net Driver**
   b) **Simple Driver**
   c) **JDBC-ODBC Bridge Driver**
   d) **JDBC-Oracle Bridge Driver**

2. **All the classes and interfaces used in jdbc programs are found within the inbuilt package**
   a) **java.sql**
   b) **java.net**
   c) **java.jdbc**
   d) **java.dbms**

Answers :
1) c.
2) a.

21

Summary

In this module, you were able to:

- Examine the role of JDBC in data persistence

22

Thank You