Chapter 13: Java Database Connectivity

Database Servers and Clients:

- Database is a repository of data.
- Data is stored permanently in the Database Server and we can retrieve later whenever needed by using query commands.
- Database client is used to retrieve data from tables and give it to the user.

JDBC:

- JDBC stands for **J**ava **D**ata**b**ase **C**onnectivity, which is a **standard Java API** for **database-independent connectivity** between the Java programming language and a wide range of databases.
- JDBC is a **specification** that provides a complete set of **interfaces** that allows for portable access to an underlying database.

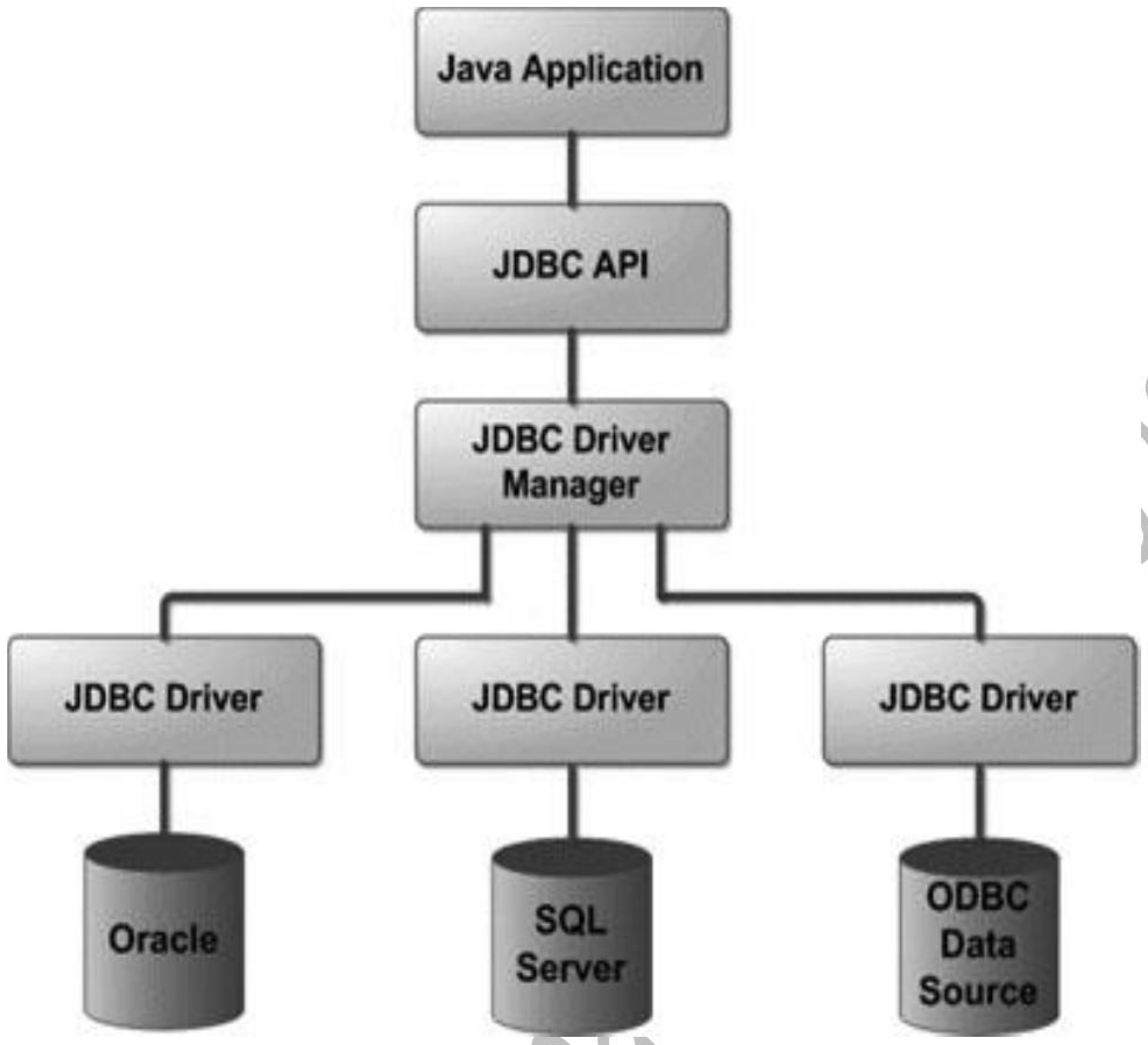
Use of JDBC:

- Java can be used to write **different types of executables**, such as:
  - **Java Applications**
  - **Java ServerPages (JSPs)**
  - **And more**
- All of these different executables are able to use a **JDBC driver** to access a database and take advantage of the stored data.

JDBC Library:

- JDBC library includes APIs for each of the tasks commonly associated with database usage:
  - Making a **connection** to a database
  - **Creating** SQL statements
  - **Executing** that SQL queries in the database
  - **Viewing & Modifying** the resulting records

JDBC Architecture:



JDBC Steps:

- Import library
- Load the driver
- Make connection
- Do some DDL / DML
- Close the connection

JDBC Statements:

- Once a connection is obtained we can interact with the database.
- The JDBC *Statement amd PreparedStatement* interfaces define the methods and properties that enable you to send SQL commands and receive data from your database.

	Statement	Prepared Statement
1	Used for normal SQL queries	Used for parameterized queries
2	Preferred when query is to be executed only once.	Preferred when particular query is to be executed multiple times
3	Performance of this interface is slow.	Performance of this interface is high.
4	Suitable for DDL operations.	Suitable for DML operations.

Statement:

- Statement is used for executing a static SQL statement and returning the results it produces.
- Before you can use a Statement object to execute a SQL statement, you need to create one using the Connection object's createStatement() method.
- **Statement createStatement() throws SQLException**
- Creates a Statement object for sending SQL statements to the database.
- SQL statements without parameters are normally executed using objects.
- If the same SQL statement is executed many times, it may be more efficient to use a PreparedStatement object.

PreparedStatement:

- A SQL statement is precompiled and stored in a object.
- This object can then be used to efficiently execute this statement multiple times.
- Before you can use a PreparedStatement object to execute SQL statement, you need to create one using the Connection object's prepareStatement( ) method
- PreparedStatement prepareStatement(String sql) throws SQLException
- Creates a PreparedStatement object for sending parameterized SQL statements to the database.
- A SQL statement with or without IN parameters can be pre-compiled and stored in a PreparedStatement object. This object can then be used to efficiently execute this statement multiple times.