Introduction to JDBC

Chen Xudong

School of Software Engineering

BJTU

2020.11.17


JDBC
SQL
SQLite
Database Programming

JDBC

JDBC

- The JDBC API is a Java API that can access data stored in a Database.
- Database is used to store and retrieve information
  - Database is one that presents information in tables with rows and columns.
  - A table is referred to as a relation in the sense that it is a collection of objects of the same type (rows).
  - A Database Management System (DBMS) handles the way data is stored, maintained, and retrieved. Generally means RDBMS(Relational Database Management System) .

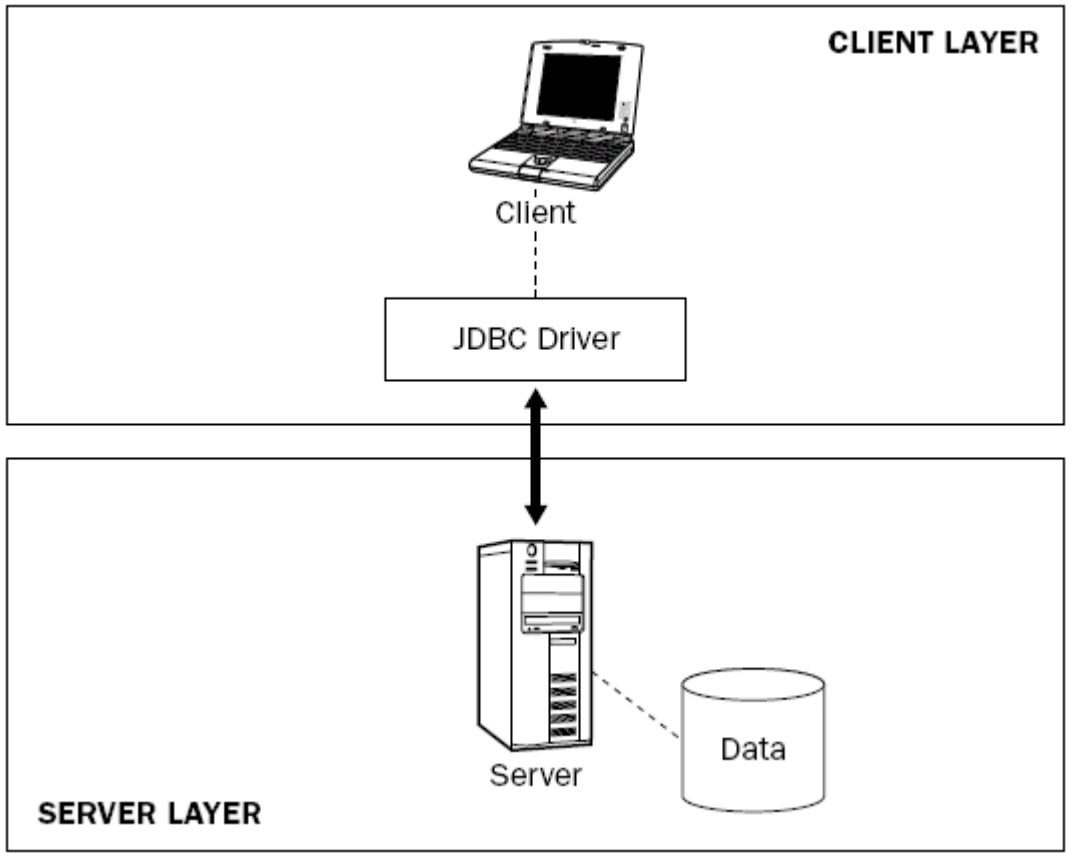
JDBC

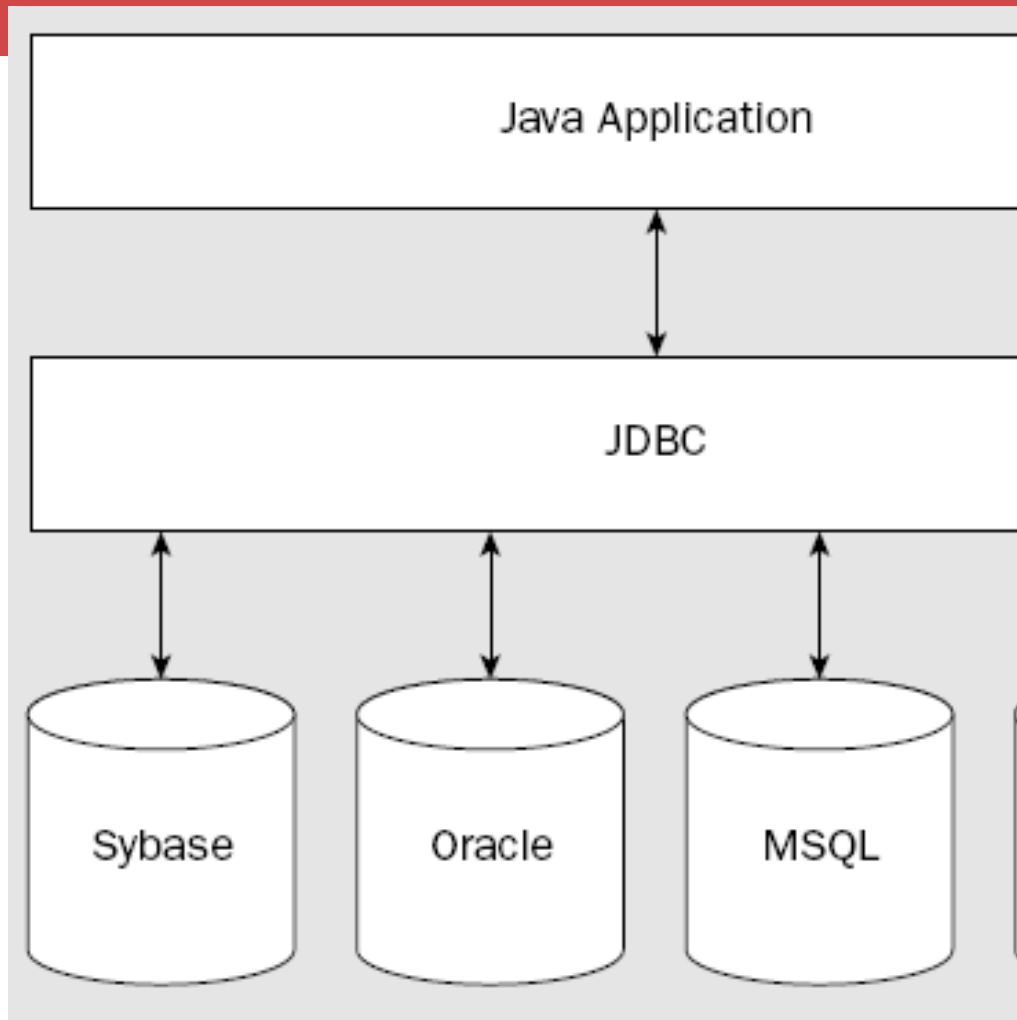
- JDBC is used to write Java applications :
  - Connect to a data source **[a database]**
  - Send queries and update statements to the database
  - Retrieve and process the results received from the database in answer to your query

JDBC API

- **The JDBC API** — The JDBC™ API provides programmatic access to relational data
  - Using the JDBC API, applications can execute SQL statements, retrieve results, and propagate changes back to an underlying data source.
  - The JDBC 4.0 API is divided into two packages: ***java.sql*** and ***javax.sql***
- The JDBC API supports both 2-tier and 3-tier processing models for database access

(1) Two-Tier Model





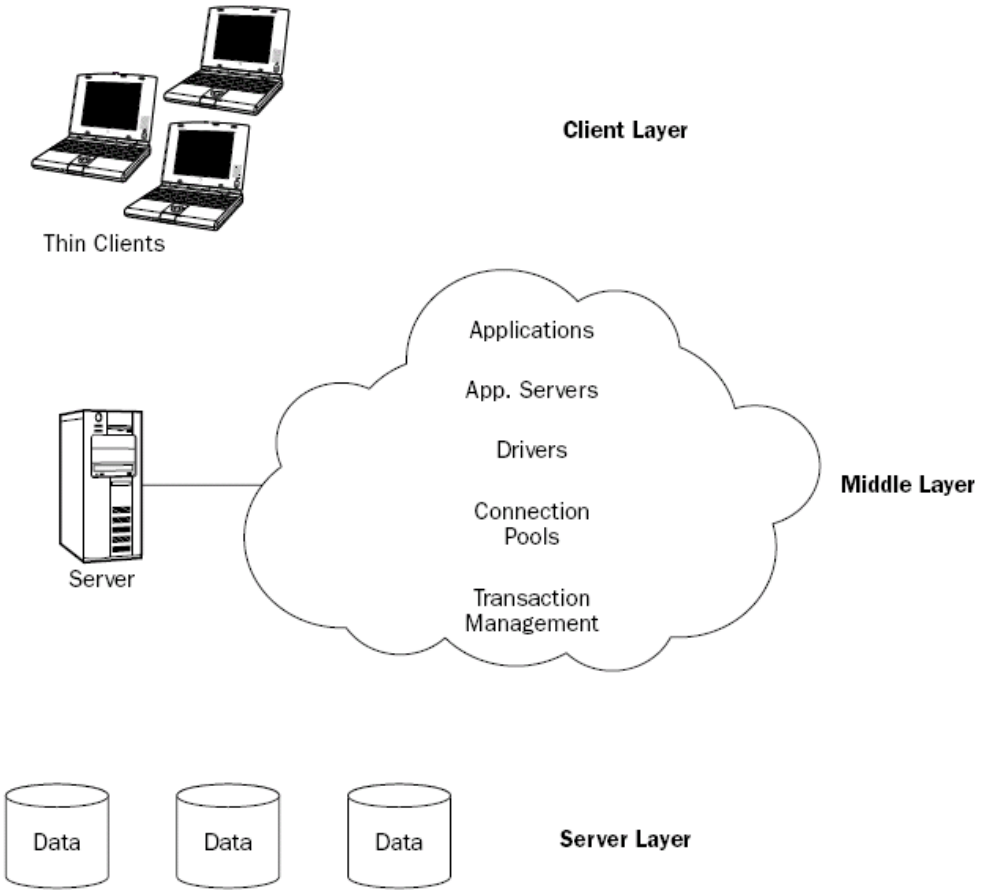
JDBC Driver

- JDBC manages to operate with a variety of different relational database systems by having an implementation of the JDBC interface for each specific database — a JDBC driver.
- A JDBC driver plays the role of the middleman between a Java program and a DBMS
  - A driver is represented by an object of type java.sql.Driver.

JDBC Driver

- 4 types of JDBC drivers：
  - ❑ JDBC-ODBC Bridge driver
    - sun.jdbc.odbc.JdbcOdbcDriver
    - work only on Windows machines, require configuration on the machine
  - ❑ Native API /partly Java
    - require installation and configuration on the machine
  - ❑ Net protocol /all-Java client
  - ❑ Native protocol /all-Java
    - No configuration on the client's machine needed
    - also known as the *thin driver*

(2)Three-Tier Model



SQL

SQL

- **Structured Query Language (**SQL) is a standard language for accessing and manipulating databases.
- Data Definition Language (DDL), Data Manipulation Language (DML), Data Query Language(DQL)
  - DDL statements : change the structure of a DB, such as CREATE TABLE and DROP TABLE.
  - DML statements : change the contents of the DB, such as INSERT, UPDATE,and DELETE
  - DQL: select data from DB,SELECT
- http://en.wikipedia.org/wiki/SQL
- http://www.w3schools.com/sql

DDL

- CREATE DATABASE myDB
  - 创建数据库myDB
- CREATE TABLE myTable(id int,name varchar(80))
  - 创建表myTable：包含id和name两列
- ALTER TABLE myTable ADD age int
  - 修改表，增加一个列定义：age
- ALTER TABLE myTable DROP age
  - 修改表，删除一列：age
- DROP TABLE myTable //删除表
- DROP DATABASE myDB//删除数据库

DML

- INSERT INTO tableName (column1, column2,...) VALUES (value1, value2,...)
  - 向表插入一条记录
- UPDATE tableName SET column1 = new WHERE column1 = old
  - 修改指定表中符合条件的记录中的指定列的值，条件由**WHERE**子句指定
- DELETE FROM tableName WHERE column1 = value
  - 删除表中符合条件的记录，条件由**WHERE**子句指定

DQL

- SELECT * FROM tableName
  - **查询一个表中的所有记录**
- SELECT column1... FROM tableName WHERE column2 = value
  - **查询表中限定条件下记录的指定列的内容**
- SELECT column1... FROM tableName WHERE column1 = value AND column2 = otherValue
  - **复杂的限定条件下，使用SELECT语句查询记录的指定列内容**
- SELECT * FROM tableName ORDER BY column1 ASC;
  - 查询结果按指定的列column1升序排序，可以多列排序。ASC为升序，降序使用DESC

Most Important SQL

- **SELECT** - extracts data from a database
- **UPDATE** - updates data in a database
- **DELETE** - deletes data from a database
- **INSERT INTO** - inserts new data into a database
- **CREATE DATABASE** - creates a new database
- **ALTER DATABASE** - modifies a database
- **CREATE TABLE** - creates a new table
- **ALTER TABLE** - modifies a table
- **DROP TABLE** - deletes a table
- **CREATE INDEX** - creates an index (search key)
- **DROP INDEX** - deletes an index

Example

- drop table zipcodes;
- create table zipcodes(zipcode varchar(6), university varchar(20), city varchar(2));
- insert into zipcodes values ('100044', '北京交通大学', '北京');
- insert into zipcodes values ('200030', '上海交通大学', '上海');
- select * from zipcodes;

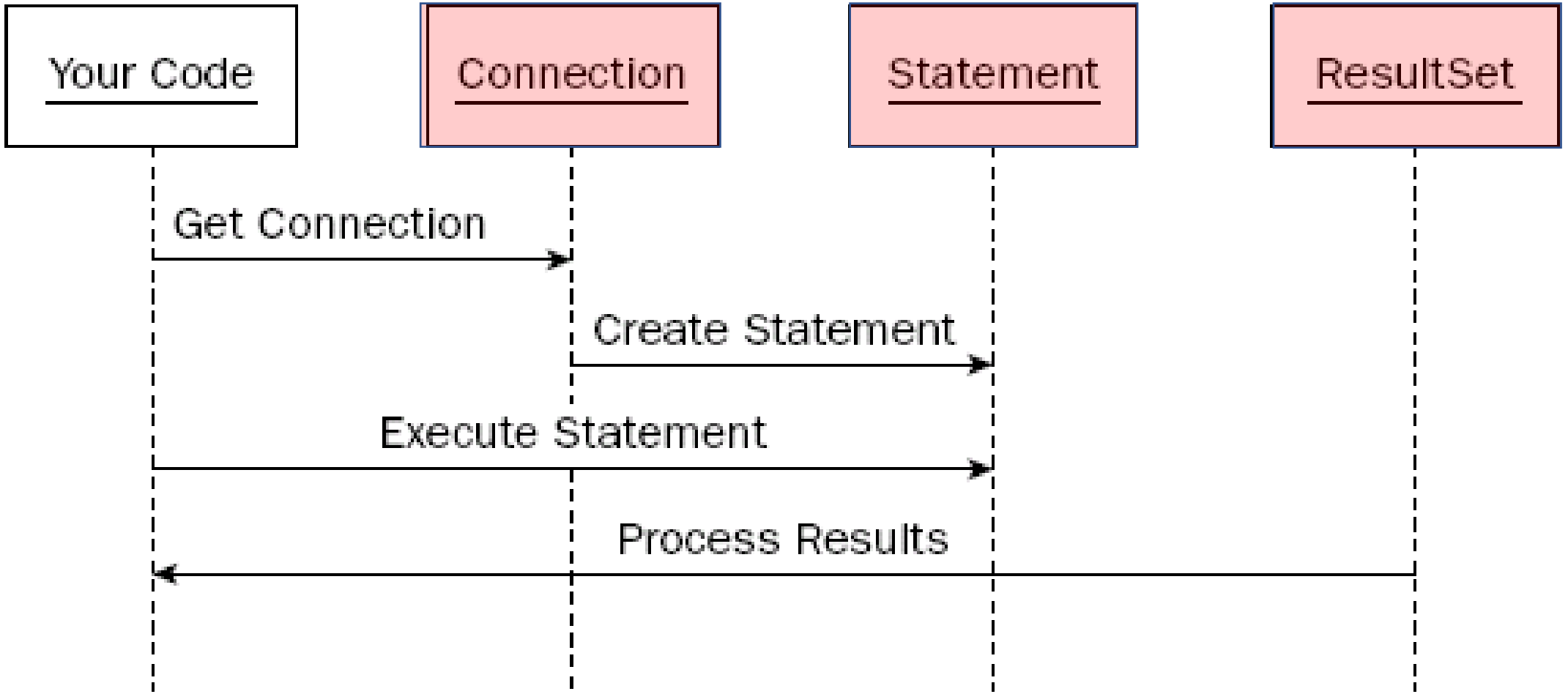
SQLite

SQLite

- https://www.sqlite.org/
- 轻型的数据库，是遵守ACID的关系型数据库管理系统，D.RichardHipp建立的公有领域项目
- 整个数据库(定义、表、索引和数据本身)都存储在一个单一的文件中
- 在嵌入式设备中，只需要几百K的内存就运行。
- 支持Windows/Linux/Unix等主流的操作系统
- 支持很多程序语言，Tcl、C#、PHP、Java等
- 相比较Mysql、PostgreSQL，它的处理速度更快。

Database Programming

Database Programming



Using JDBC in a program

1. Import the necessary classes.
2. Load the JDBC driver - Class.forName()
3. Identify the data-source.
4. Create a Connection object -DriverManager.getConnection()
5. Create a Statement object.
6. Execute a query using the Statement object.
7. Retrieve data from the returned ResultSet object.
8. Close the ResultSet, Statement object, Connection object.

JAVA中使用SQLite：基本流程

- 引用驱动 Class.forName("org.sqlite.JDBC");
- 建立连接 Connection conn = DriverManager.getConnection("jdbc:sqlite:filename");
- 执行SQL语句 Statement stat = conn.createStatement();

```
stat.executeUpdate("create table tbl1(name varchar(20), salary int);");
stat.executeUpdate("insert into tbl1values('ZhangSan',8000);");
ResultSet rs = stat.executeQuery("select * from tbl1;");
while(rs.next()){
System.out.print("name = "+ rs.getString("name")+" ");
System.out.println("salary = "+ rs.getString("salary"));
}
```

连接数据库

- 如果数据库不存在，那么它就会被创建，最后将返回一个数据库对象。
- 在当前目录中创建数据库 **test.db**

```
Class.forName("org.sqlite.JDBC");
Connection c =
DriverManager.getConnection("jdbc:sqlite:test.
db");
```

创建表

```
Statement stmt = c.createStatement();
String sql = "CREATE TABLE COMPANY " +
             "(ID INT PRIMARY KEY NOT NULL," +
             " NAME TEXT NOT NULL, " +
             " AGE INT NOT NULL, " +
             " ADDRESS CHAR(50), " +
             " SALARY REAL)";
stmt.executeUpdate(sql);
stmt.close();
```

INSERT 操作

```
String sql = "INSERT INTO COMPANY
(ID,NAME,AGE,ADDRESS,SALARY) " + "VALUES (1,
'Paul', 32, 'California', 20000.00 );";
stmt.executeUpdate(sql);

sql = "INSERT INTO COMPANY
(ID,NAME,AGE,ADDRESS,SALARY) " + "VALUES (4,
'Mark', 25, 'Rich-Mond ', 65000.00 );";
stmt.executeUpdate(sql);
```

SELECT 操作

```
ResultSet rs = stmt.executeQuery( "SELECT * FROM COMPANY;" );
while ( rs.next() ) {
  int id = rs.getInt("id");
  String name = rs.getString("name");
  int age = rs.getInt("age");
  String address = rs.getString("address");
  float salary = rs.getFloat("salary");
  System.out.println( "ID = " + id  +
          "\nNAME = " + name +
          "\nAGE = " + age +
          "\nADDRESS = " + address +
          "\nSALARY = " + salary) ;
}
rs.close();
```

UPDATE /DELETE 操作

```
stmt = c.createStatement();
String sql = "UPDATE COMPANY set SALARY =
25000.00 where ID=1;";
stmt.executeUpdate(sql);
c.commit();



String sql = "DELETE from COMPANY where ID=2;";
stmt.executeUpdate(sql);
c.commit();
```

JAVA中使用SQLite

- 先下载SQLite数据库的JDBC
  - 从 sqlite-jdbc 库下载 *sqlite-jdbc-(VERSION).jar* 的最新版本
- 将下载到的jar包添加到classpath系统环境变量中，或者在 -classpath 选项中使用

SQLiteJDBC.java

```
javac SQLiteJDBC.java
java -classpath ".; sqlite-jdbc-3.32.3.2.jar" SQLiteJDBC
```