Chapter 10 JDBC

javacose@qq.com

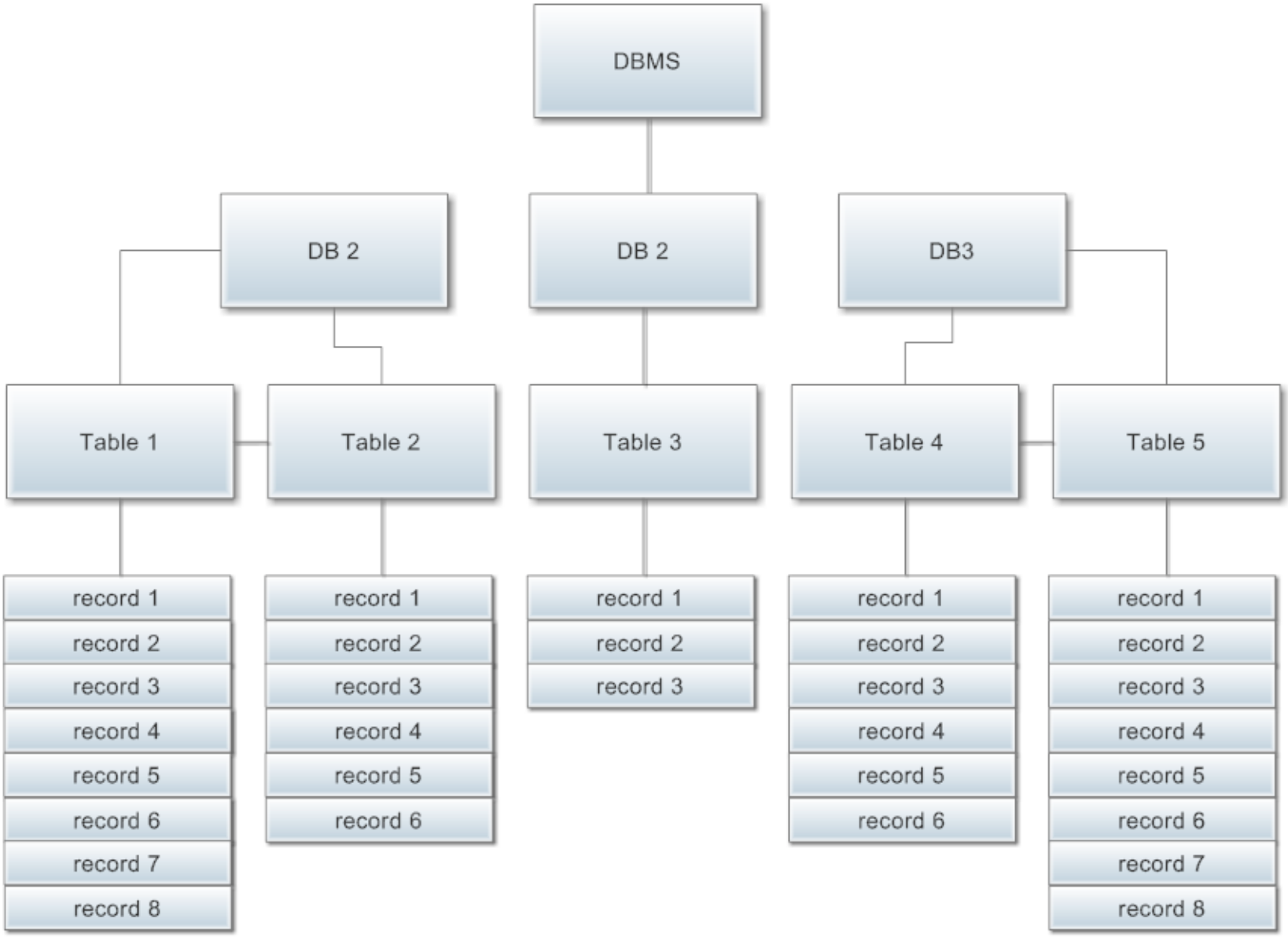
Xiang Zhang

Content

- Basic Concepts in Database
- SQL Grammar
- JDBC Principle
- JDBC Examples

Basic Concepts in Database

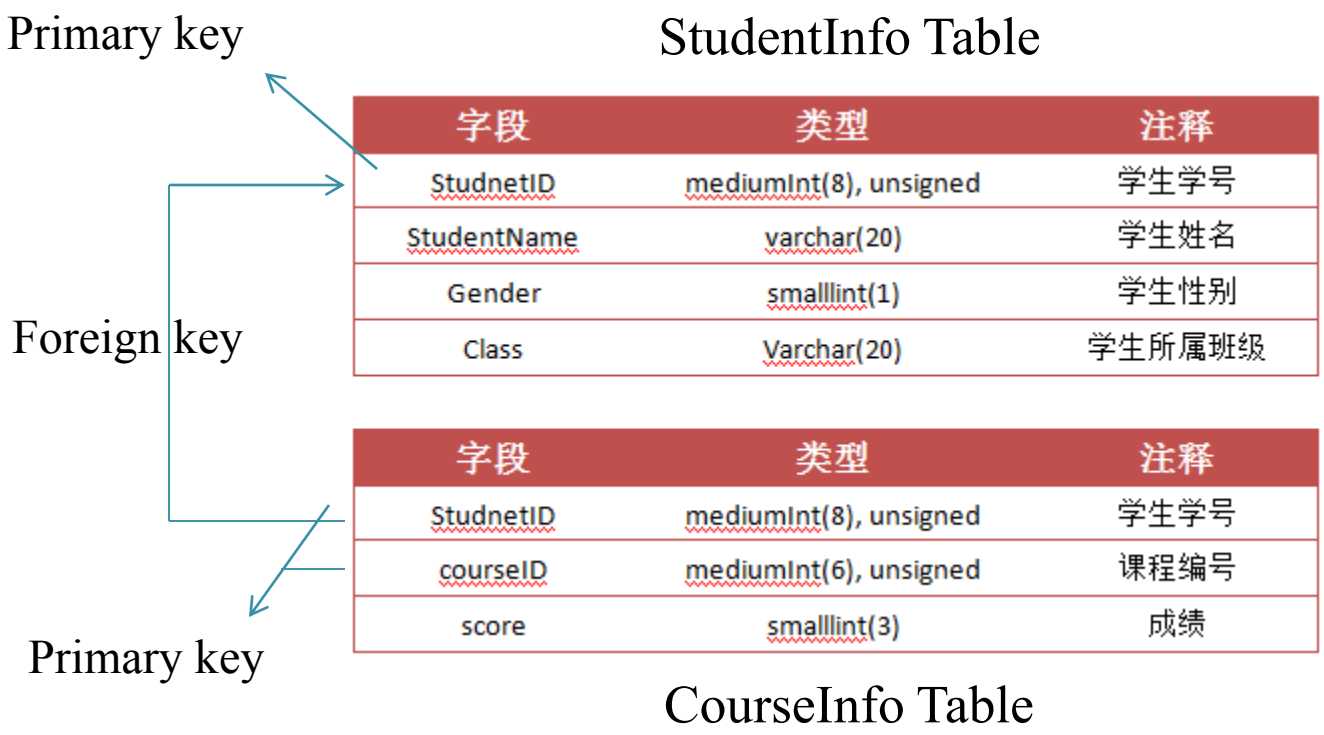
- Database and DBMS
- RDBMS (关系型数据库管理系统)
  - Store data in rows and columns
  - Rows called Record (记录) , and columns called Field (字段)
  - A set of rows and columns are called Table (表)
  - A table usually represents an Entity (实体)
  - There are Relations (关系) between Entities, → ER Map
  - Query through SQL (结构化查询语言)


DBMS
DB 2
DB 2
DB3
Table 1
Table 2
Table 3
Table 4
Table 5
record 1
record 2
record 3
record 4
record 5
record 6
record 7
record 8
record 1
record 2
record 3
record 4
record 5
record 6
record 1
record 2
record 3
record 1
record 2
record 3
record 4
record 5
record 6
record 1
record 2
record 3
record 4
record 5
record 6
record 7
record 8

字段	类型	Null	默认	额外	注释
userid	mediumint(6) unsigned	**PK**		auto_increment	用户id
username	varchar(20)	YES			用户名
password	varchar(32)	YES			密码
roleid	smallint(5)	YES	0		角色
encrypt	varchar(6)	YES			加密因子
lastloginip	varchar(15)	YES			最后登录ip
lastlogintime	int(10) unsigned	YES	0		最后登录时间
email	varchar(40)	YES			Email
realname	varchar(50)	NO			真是姓名
card	varchar(255)	NO			密保卡

userid	username	password	roleid	encrypt	lastloginip	lastlogintime	email	realname	card
1	zhangsan	******	1	***	202.119.1.2	2015.11.29.18:20:18	zs@seu.edu.cn	张三	***
2	lisi	**	1	***	202.119.1.3	2015.10.10.06:20:49	lisi@seu.edu.cn	李四	***
3	wangwu	*****	1	***	202.119.1.4	2015.10.31.09:06:06	ww@seu.edu.cn	王五	***
4	zhaoliu	*******	2	***	202.119.1.5	2015.11.05.12:00:00	zl@seu.edu.cn	赵六	***
5	jia	***	2	***	202.119.1.6	2015.11.29.10:09:00	jia@seu.edu.cn	甲	***
6	yi	*****	2	***	202.119.1.7	2015.11.28.18:20:20	yi@seu.edu.cn	乙	***
7	bin	******	3	***	202.119.1.8	2015.11.30.10:00:00	bin@seu.edu.cn	丙	***

Primary Key and Foreign Key



字段	类型	注释
StudnetID	mediumInt(8), unsigned	学生学号
StudentName	varchar(20)	学生姓名
Gender	smalllint(1)	学生性别
Class	Varchar(20)	学生所属班级

字段	类型	注释
StudnetID	mediumInt(8), unsigned	学生学号
courseID	mediumInt(6), unsigned	课程编号
score	smalllint(3)	成绩

Basic Concepts in Database

- RDBMS
  - MySQL / PostgreSQL / Berkeley DB
  - Oracle
  - SQL Server
  - DB2

Basic Concepts in Database

- NoSQL Databases
  - ElasticSearch
  - Virtuoso
  - Redis
  - MongoDB

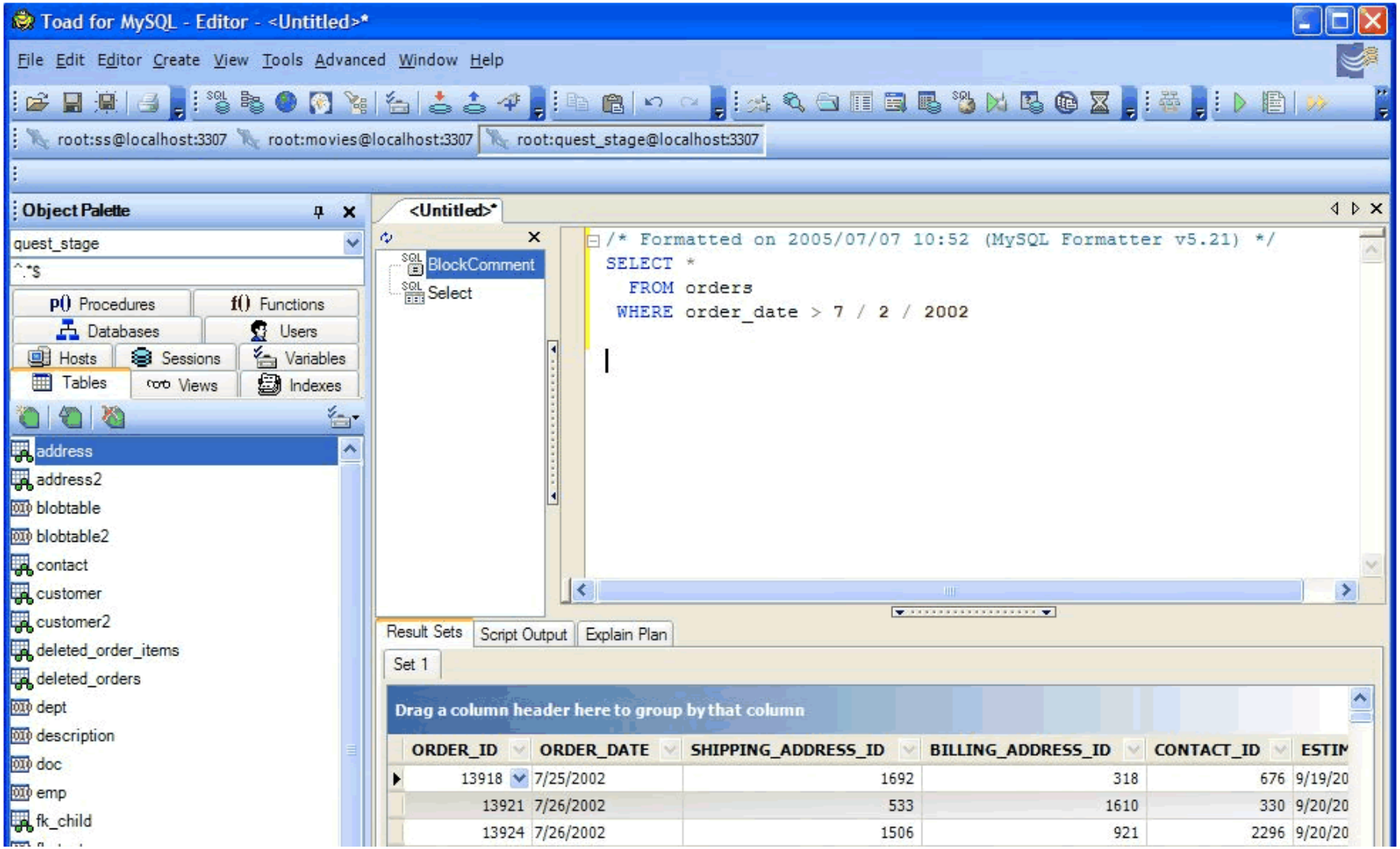
DB Ranking

- http://db-engines.com/en/ranking

341 systems in ranking, December 2018

Rank			DBMS	Database Model	Score		
Dec 2018	Nov 2018	Dec 2017			Dec 2018	Nov 2018	Dec 2017
1.	1.	1.	Oracle	Relational DBMS	1283.22	-17.89	-58.32
2.	2.	2.	MySQL	Relational DBMS	1161.25	+1.36	-156.82
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1040.34	-11.21	-132.14
4.	4.	4.	PostgreSQL	Relational DBMS	460.64	+20.39	+75.21
5.	5.	5.	MongoDB	Document store	378.62	+9.14	+47.85
6.	6.	6.	IBM Db2	Relational DBMS	180.75	+0.87	-8.83
7.	7.	↑ 8.	Redis	Key-value store	146.83	+2.66	+23.59
8.	8.	↑ 10.	Elasticsearch	Search engine	144.70	+1.24	+24.92
9.	9.	↓ 7.	Microsoft Access	Relational DBMS	139.51	+1.08	+13.63
10.	10.	↑ 11.	SQLite	Relational DBMS	123.02	+0.31	+7.82

Basic Concepts in Database



SQL Grammar

- CRUD and CRUD Applications
  - **C**reate – Insert Statement in SQL
  - **R**etrieval – Select Statement in SQL
  - **U**pdate – Update Statement in SQL
  - **D**elete – Delete Statement in SQL

SQL - Select

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT store_name FROM Store_Information
```

- Result

```
store_name
Los Angeles
San Diego
Los Angeles
Boston
```

SQL – Select Distinct

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT DISTINCT store_name FROM Store_Information
```

- Result

store_name
Los Angeles
San Diego
Boston

SQL - Select

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT store_name FROM Store_Information WHERE Sales > 1000
```

- Result

```
store_name
Los Angeles
```

SQL – Select where

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT store_name FROM Store_Information
WHERE Sales > 1000 OR (Sales < 500 AND Sales > 275)
```

- Result

```
store_name
Los Angeles
San Francisco
```

SQL – Select in

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT * FROM Store_Information
WHERE store_name IN ('Los Angeles', 'San Diego')
```

- Result

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999

SQL – Select between

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT * FROM Store_Information
WHERE Date BETWEEN 'Jan-06-1999' AND 'Jan-10-1999'
```

- Result

store_name	Sales	Date
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

SQL – Select like

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT * FROM Store_Information
WHERE store_name LIKE '%AN%'
```

- Result

store_name	Sales	Date
LOS ANGELES	$1500	Jan-05-1999
SAN FRANCISCO	$300	Jan-08-1999
SAN DIEGO	$250	Jan-07-1999

SQL – Select order by

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT store_name, Sales FROM Store_Information
ORDER BY Sales DESC
```

- Result

store_name	Sales
Los Angeles	$1500
Boston	$700
San Francisco	$300
San Diego	$250

SQL – Select count

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
SELECT COUNT(DISTINCT store_name)
FROM Store_Information
```

- Result

Count(DISTINCT store_name)
3

SQL – Delete

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
DELETE FROM Store_Information
WHERE store_name = "Los Angeles"
```

- Result

store_name	Sales	Date
San Diego	$250	Jan-07-1999
Boston	$700	Jan-08-1999

SQL – Insert

- Data

Store_Information 表格

Column Name	Data Type
store_name	char(50)
Sales	float
Date	datetime

- Query

```
INSERT INTO Store_Information (store_name, Sales, Date)
VALUES ('Los Angeles', 900, 'Jan-10-1999')
```

SQL – Update

- Data

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

- Query

```
UPDATE Store_Information SET Sales = 500
WHERE store_name = "Los Angeles" AND Date = "Jan-08-1999"
```

- Result

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$500	Jan-08-1999
Boston	$700	Jan-08-1999

Test

Store_Information 表格

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

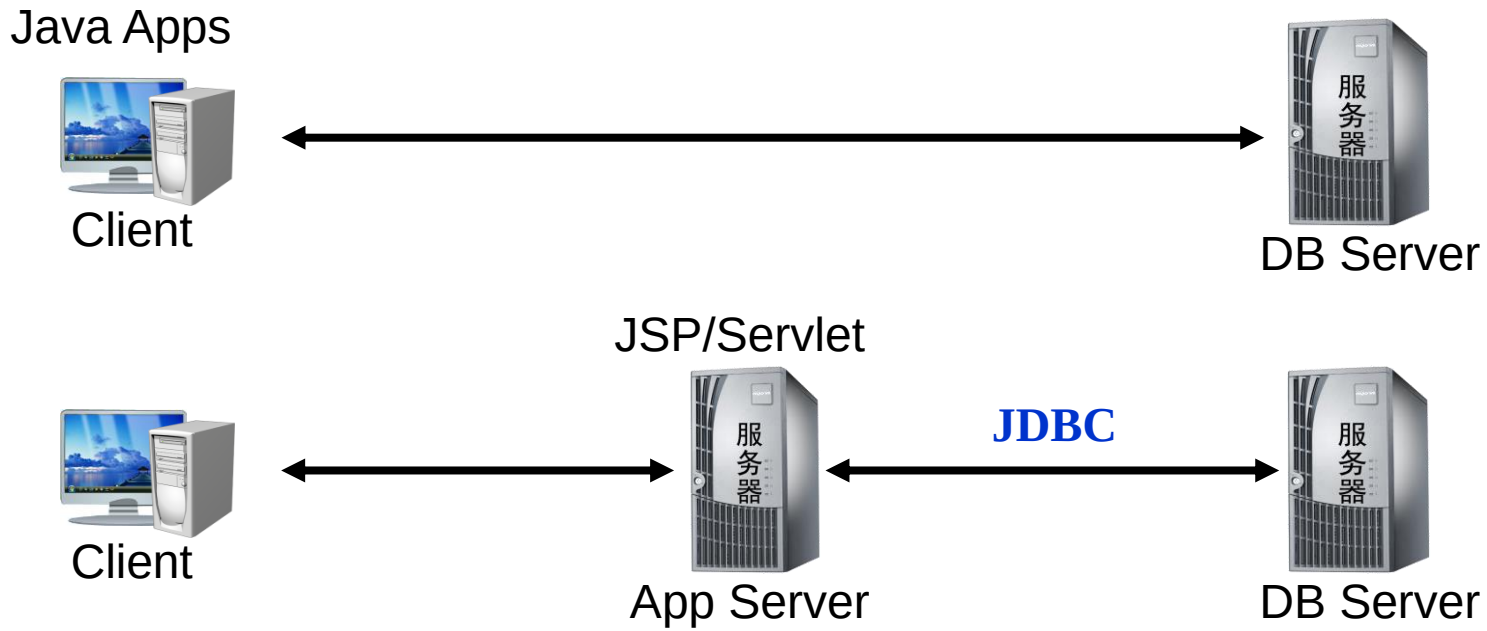
- Write following SQL query
  - Query for all store names
  - Query for the number of stores with sales > 500
  - Query for all info of stores with sales in (100, 500)
  - Query for sales of stores with name containing “an”, and rank the result descendingly according to sales
  - Insert into table a new record (any record will do)
  - Modify all the store sales in LA to 1000
  - Delete all records related to LA

Self-study

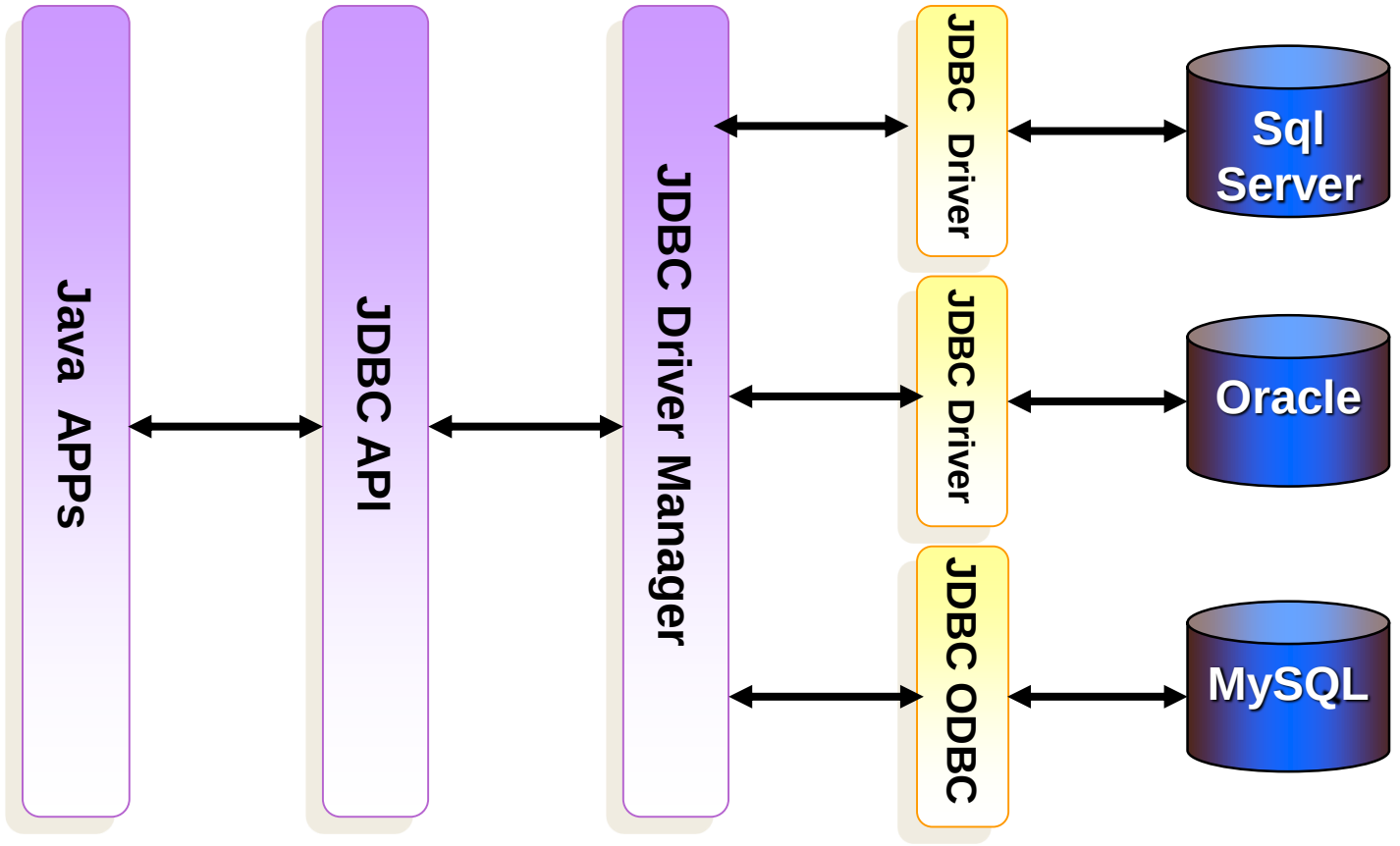
- SQL completed grammar
  - Recommendation:
  - http://sql.1keydata.com/cn/sql.php

JDBC

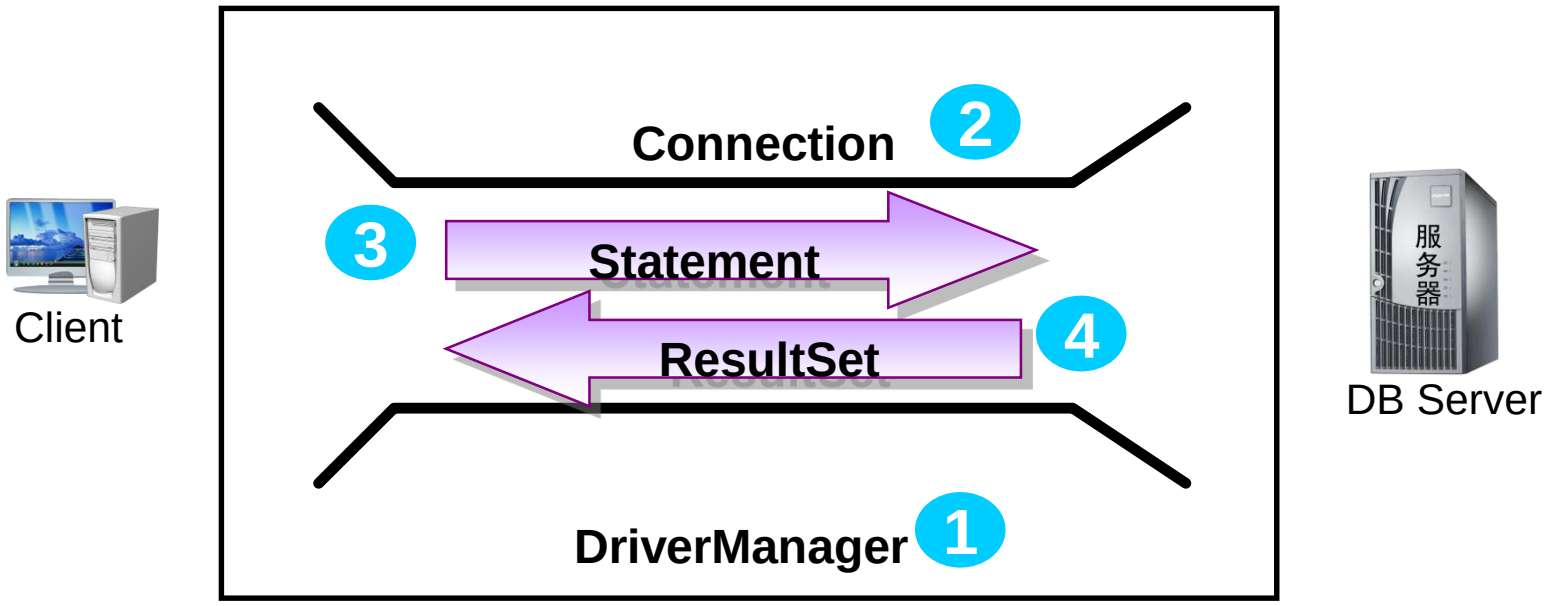
- JDBC - Java Database Connectivity
- Provides API for database access



JDBC Principle



JDBC Principle



JDBC API

- Provider: Sun
- Package
  - java.sql
  - javax.sql
- Major Classes and Interfaces
  - DriverManager Class
  - Connection Interface
  - Statement Interface
  - ResultSet Interface

JDBC Example

```
// 加载及注册JDBC驱动程序
Class.forName("com.mysql.jdbc.Driver");

//创建JDBC连接
String dbURL = "jdbc:mysql://localhost:3306/" +
     "MyDB?user=your_username&password=your_password";
Connection connection = DriverManager.getConnection(dbURL);

//创建Statement
String sqlQuery = "SELECT DISTINCT store_name " +
     "from bookstore";
Statement statement = connection.createStatement();

//执行查询并处理查询结果
ResultSet rs = statement.executeQuery(sqlQuery);
while (rs.next()) {
     System.out.println(rs.getString("store_name"));
}
```

JDBC Example

```
try{
    // 加载及注册JDBC驱动程序
    Class.forName(...);
    //创建JDBC连接
    Connection connection = ......
    //创建Statement
    Statement statement  = ......
    //创建查询并处理查询结果
    ResultSet rs = .......
}
catch (ClassNotFoundException e) {System.out.println("无法找到驱动类");}
catch (SQLException e) {e.printStackTrace();}
finally {
    try {
        rs.close();
        statement.close();
        connection.close();
    } catch (Exception e) { e.printStackTrace();}
}
```

ResultSetMetaData

```
String sqlQuery = "SELECT * FROM bookstore";
ResultSet rs = statement.executeQuery(sqlQuery);
ResultSetMetaData rsmd = rs.getMetaData();
for(int i=1; i<=rsmd.getColumnCount(); i++){
    System.out.print(rsmd.getColumnName(i) + "\t");
}
```

```
<terminated> MySQLTest [Java Application] C:\Pr
store_name      Sales    Date
```

Two Way of Executing Statement

- executeUpdate
  - For queries with no results returned, usually Insert \ Delete \ Update
- executeQuery
  - For queries with results returned, usually Select

```
Statement stmt = conn.createStatement();
stmt.executeUpdate("DELETE FROM bookstore WHERE store_name = \"Boston\" ");

Statement stmt = conn.createStatement();
Result rs = stmt.executeQuery("SELECT * FROM bookstore");
```

PreparedStatement Interface

- Derived from Statement interface
- For repeatedly executed SQL
- Usually for queries with no result, such as Insert \ Delete \ Update
- Together with addBatch() and executeBatch()

PreparedStatement

```
//构建PreparedStatement
String sql = "INSERT INTO tab (id, name, score) values (?, ?, ?)";
PreparedStatement pstmt = connection.prepareStatement(sql);

//加入一条记录
pstmt.setInt(1, 71108398);
pstmt.setString(2, "张三");
pstmt.setInt(3, 80);
pstmt.addBatch();
pstmt.clearParameters();

//加入另一条记录
pstmt.setInt(1, 71108399);
pstmt.setString(2, "李四");
pstmt.setInt(3, 21);
pstmt.addBatch();
pstmt.clearParameters();
...
pstmt.executeBatch();
pstmt.clearBatch();
```

Self Study

- NOSQL Databases
  - Document Store
  - Key-value DB
  - Graph DB

Self-study

- Database Connection Pool
  - *Resource Pool* design pattern
- Database
  - Installation and test of MySQL server
  - MySQL client