

数据库系统概论

Introduction to Database System

8.3 JDBC编程

8.3.1 JDBC工作原理概述

8.3.2 JDBC APIs基础

8.3.3 使用JDBC操纵数据库的工作流程

8.3.1 JDBC工作原理概述

❖JDBC (Java Database Connection)

- 由于不同的数据库管理系统的存在，在某个关系数据库管理系统下编写的应用程序就不能在另一个关系数据库管理系统下运行
- 许多应用程序需要共享多个部门的数据资源，访问不同的关系数据库管理系统

JDBC工作原理概述（续）

JDBC概念：

- JDBC就是使用Java语言操作关系型数据库的一套标准接口（Application Programming Interface, API）
- 全称：Java DataBase Connectivity, Java数据库连接

JDBC工作原理概述（续）

同一套Java代码，操作不同的关系型数据库

Java代码



JDBC工作原理概述（续）

JDBC本质：

- （1）官方（**sun**公司）定义的一套操作所有关系型数据库的规则，即接口；
- （2）各个数据库厂商去实现这套接口，提供数据库驱动**jar**包；
- （3）我们可以使用这套接口（**JDBC**）编程，真正执行的代码是驱动**jar**包中的实现类。

JDBC好处：

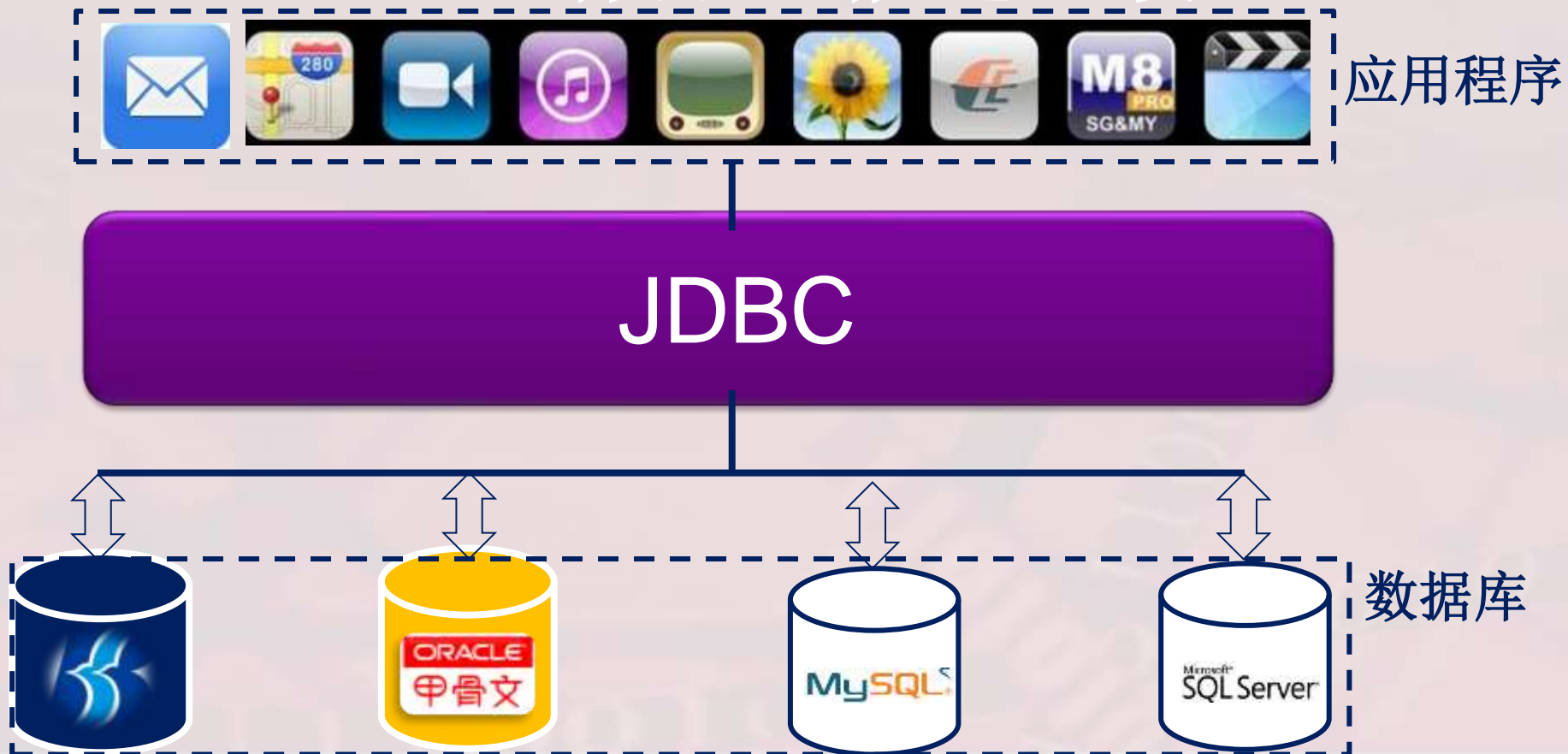
- （1）各数据库厂商使用相同的接口，**Java**代码不需要针对不同数据库分别开发；
- （2）可随时替换底层数据库，访问数据库的**Java**代码基本不变。

JDBC工作原理概述（续）

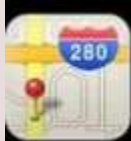
❖ JDBC应用系统的体系结构

1. 用户应用程序
2. JDBC驱动程序管理器
3. 数据源

JDBC工作原理概述（续）



JDBC工作原理概述（续）



应用程序

JDBC应用程序编程接口

JDBC

驱动程序管理器

数据库驱动程序1

数据库驱动程序2

数据库驱动程序3

数据库驱动程序4



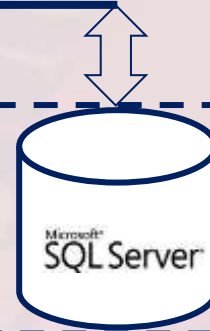
Kingbase
数据源



Oracle
数据源



MySQL
数据源



Microsoft
SQL Server
数据源

数据库

JDBC工作原理概述（续）

❖ JDBC驱动程序管理器

驱动程序管理器对用户透明的，管理应用程序和驱动程序之间的通信。应用程序对数据库的各种操作请求是通过调用**JDBC**驱动程序管理器提供的标准接口完成。**如果应用程序要操纵不同DBMS下的数据库，就要动态地链接到不同的驱动程序上。**

主要功能：

- （1）装载**JDBC**驱动程序。
- （2）选择和连接正确的驱动程序。
- （3）管理数据源。

JDBC工作原理概述（续）

❖ 数据源

数据源是用户最终需要访问的数据，包含数据库位置、数据库类型、访问数据库的用户名和密码等信息，实际上是一种数据连接的抽象。

8.3 JDBC编程

8.3.1 JDBC工作原理概述

8.3.2 JDBC APIs基础

8.3.3 使用JDBC操纵数据库的工作流程

8.3.2 JDBC API基础

❖ JDBC中的常用类

- JDBC进行应用程序开发涉及到的所有类都包含在 **java.sql** 包中
- 不同的JDBC版本接口名和使用略有差异

8.3.2 JDBC API基础

❖JDBC 常用类

使用JDBC进行应用程序开发涉及的所有类都包含在java.sql包中，常用的有：

类名	路径	备注
驱动程序类	java.sql.Driver	由各数据库厂商提供
驱动程序管理类	java.sql.DriverManager	作用于应用程序与驱动程序之间
数据库连接类	java.sql.Connection	用于建立与指定数据库的连接
静态SQL语句执行类	java.sql.Statement	用于执行静态SQL语句并返回结果
动态SQL语句执行类	java.sql.PreparedStatement	用于执行含参SQL语句并返回结果
存储过程语句执行类	java.sql.CallableStatement	用于执行存储过程语句并返回结果
结果集处理类	java.sql.ResultSet	用于检索结果集中的数据

8.3.2 JDBC API基础

❖ DriverManager

DriverManager(驱动管理类)作用:

- 1.注册驱动
- 2.获取数据库连接

8.3.2 JDBC API基础

DriverManager

1. 注册驱动

```
Class.forName("com.mysql.jdbc.Driver");
```

- 查看 Driver 类源码

```
static {  
    try {  
        DriverManager.registerDriver(new Driver());  
    } catch (SQLException var1) {  
        throw new RuntimeException("Can't register driver!");  
    }  
}
```

提示:

- MySQL 5之后的驱动包, 可以省略注册驱动的步骤
- 自动加载jar包中META-INF/services/java.sql.Driver文件中的驱动类

8.3.2 JDBC API基础

DriverManager

2. 获取连接

```
static Connection getConnection(String url, String user, String password)
```

- 参数

1. url: 连接路径

语法: jdbc:mysql://ip地址(域名):端口号/数据库名称?参数键值对1&参数键值对2...

示例: jdbc:mysql://127.0.0.1:3306/db1

细节:

- 如果连接的是本机mysql服务器, 并且mysql服务默认端口是3306, 则url可以简写为: jdbc:mysql:///数据库名称?参数键值对
- 配置 useSSL=false 参数, 禁用安全连接方式, 解决警告提示

2. user: 用户名

3. password: 密码

8.3.2 JDBC API基础

Connection

- Connection(数据库连接对象)作用:
 1. 获取执行 SQL 的对象
 2. 管理事务

8.3.2 JDBC API基础

Connection

1. 获取执行 SQL 的对象

- 普通执行SQL对象

```
Statement createStatement()
```

- 预编译SQL的执行SQL对象：防止SQL注入

```
PreparedStatement prepareStatement (sql)
```

- 执行存储过程的对象

```
CallableStatement prepareCall (sql)
```

8.3.2 JDBC API基础

Statement

- Statement作用：
 1. 执行SQL语句
- 执行SQL语句

`int executeUpdate(sql):` 执行DML、DDL语句

➤ 返回值: (1) DML语句影响的行数 (2) DDL语句执行后, 执行成功也可能返回 0

`ResultSet executeQuery(sql):` 执行DQL 语句

➤ 返回值: `ResultSet` 结果集对象

8.3.2 JDBC API基础

ResultSet

- ResultSet(结果集对象)作用:
 1. 封装了DQL查询语句的结果

ResultSet stmt.executeQuery(sql): 执行DQL 语句, 返回 ResultSet 对象

8.3.2 JDBC API基础

❖数据类型

■**SQL数据类型**：用于数据源，例如创建关系模式时指定属性的数据类型。

•**VARCHAR、CHAR、BIT、NUMERIC、DATE**等

■**Java数据类型**：用于应用程序的**Java**代码，例如程序中定义的变量的数据类型。

•**String、boolean、BigDecimal、byte、short、int**等

❖由数据库管理系统的驱动程序完成自身数据类型和**JDBC**标准数据类型的映射

数据类型（续）

❖ SQL数据类型和java数据类型之间的转换规则

	SQL数据类型	Java数据类型
SQL数据类型	数据源之间转换	应用程序变量传递给语句参数
Java数据类型	从数据库中读取数据赋值给应用程序变量	应用程序变量之间转换

8.3 JDBC编程

8.3.1 JDBC工作原理概述

8.3.2 JDBC APIs基础

8.3.3 使用JDBC操纵数据库的工作流程

8.3.3 使用JDBC操纵数据库的工作流程

❖ 搭建开发环境的流程（以MySQL数据库服务器为例）


■ 步骤1：下载MySQL数据库：

<https://dev.mysql.com/downloads/mysql/>

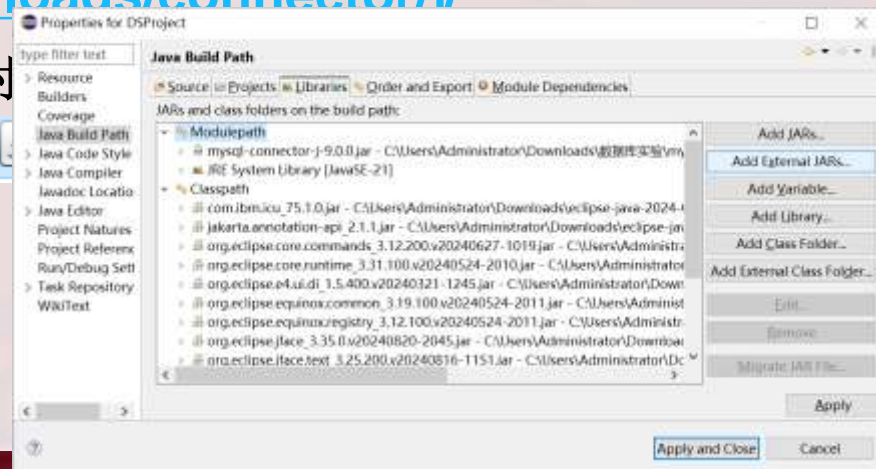
■ 步骤2：下载MySQL连接器：

<https://dev.mysql.com/downloads/connector/j/>

● 注意根据MySQL的版本选择对

 mysql-connector-j-9.0.0.zip

■ 步骤3：加载到Java项目中



8.3.3 使用JDBC操纵数据库的工作流程

❖ 操作数据库的基本工作流程

- 步骤1：加载驱动程序
- 步骤2：建立与数据库的连接
 - 定义连接的URL
 - 利用生成的URL建立与数据库的连接
- 步骤3：执行SQL语句
 - 创建语句执行类对象
 - 执行SQL语句，可以通过executeQuery、executeUpdate、execute三种方式执行。
- 步骤4：处理结果集
- 步骤5：释放资源

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤1：加载驱动程序

- 驱动程序在JDBC API中实现定义数据交互的接口
- [例8.7]对MySQL、Kingbase、Oracle、SQL Server加载数据库驱动

```
Class.forName("com.mysql.cj.jdbc.Driver");           /* MySQL */  
Class.forName("com.kingbase.Driver");                 /* Kingbase */  
Class.forName("oracle.jdbc.OracleDriver");           /* Oracle */  
Class.forName("com.microsoft.jdbc.sqlserver.SQLServerDriver"); /* SQL Server */
```

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤2：获取连接（设置待连接数据库的URL，建立与数据库连接）

- 加载驱动后，可以通过URL地址与数据库建立连接
- URL包含了连接数据库所需的协议、子协议和数据库名称，定义格式为：<协议名>:<子协议名>:<数据库名称>

■ 【例8.8】定义与MySQL、Kingbase、Oracle、SQL Server数据库连接的URL

`strURL = "jdbc:mysql://" + 服务器名 + ":" + 端口号 + "/" + 数据库名;`

`strURL = "jdbc:kingbase://" + 服务器名 + ":" + 端口号 + "/" + 数据库名;`

`strURL = "jdbc:oracle:thin:@" + 服务器名 + ":" + 端口号 + ":" + 数据库名`

`strURL = "jdbc:microsoft:sqlserver://" + 服务器名 + ":" + 端口号 + ":" + 数据库名`

MySQL、Kingbase、Oracle、SQL Server的默认端口号分别为3306、54321、1521、1433

8.3.3 使用JDBC操纵数据库的工作流程

❖ [例8.9]建立与Kingbase数据库的连接，假定服务器地址为192.168.0.118，端口为54321，数据库名为DB-Student，用户名为Info001，密码为123456

- `String strURL = "jdbc:kingbase:// 192.168.0.118:54321/DB-Student";`
- `Connection conn= DriverManager.getConnection(strURL,"Info001","123456");`

8.3.3 使用JDBC操纵数据库的工作流程

- ❖ [例-补充]建立与MySQL数据库的连接，假定服务器地址为127.0.0.1(MySQL服务器位于本机)，端口为3306，数据库名为dsdb，用户名为root，密码为db123456

```
Class.forName("com.mysql.cj.jdbc.Driver");  
Connection conn = DriverManager.getConnection(  
    "jdbc:mysql://127.0.0.1:3306/dsdb",  
    "root",  
    "db123456");
```

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤3：执行SQL语句

■ 静态语句执行类对象(Statement)

```
Statement stm = conn.createStatement();
```

■ 派生执行类对象：

- 动态语句执行类(**PreparedStatement**)：执行动态的**SQL**语句（含参数），也就是说允许程序在执行时灵活地调整**SQL**的内容，可以有效防止**SQL**注入攻击。
- 存储过程执行类(**CallableStatement**)：执行数据库存储过程

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤3：执行SQL语句

■ 执行方法

- **ResultSet executeQuery():** 执行数据库查询语句
- **int executeUpdate():** 处理增、删、改以及定义语句
- **boolean execute():** 处理存储过程或动态SQL语句

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤3：执行SQL语句

■ 【8.10】使用JDBC向课堂评价表中插入一条记录

```
PreparedStatement stmt = conn.prepareStatement("INSERT INTO SC  
VALUES(?,?,?,?,?,?)");
```

/* 生成PreparedStatement类对象中的动态参数，注意第六个字段Feedback，未设置输入值 */

```
stmt.setString(1, " 20180001 ");          /*设置学生学号*/  
stmt.setString(2, "19950018");            /*设置职工号*/  
stmt.setString(3, "81001-01");            /*设置教学班号*/  
stmt.setString(4, "老师讲得很出色");      /*设置学生评价意见*/  
stmt.setBoolean(5, true);                 /*设置学生评价意见类型*/  
stmt.executeUpdate();
```

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤4：处理结果集

■ ResultSet: 结果集合类对象

- **.GetXXX(参数)**
 - 获取元组的属性值，**XXX**代表某种数据类型，比如**string**
 - 可以指定参数为列号(**JDBC**的列从**1**开始)或列名
- **.next()** –判断是否有下一行，返回**true**或者**false**
- 游标(**cursor**): 当结果集刚刚生成时，游标指向第一行数据之前，通过**next()**、**previous()**等操作来移动游标获取结果集中的每一行数据。

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤4：基于该结果集，处理用户逻辑

■ [8.12]遍历教师职工号为19950018，教学班号为81001-01的学生课程评价详情

```
String SQL = "SELECT Sno, Assess, CAtype, Feedback FROM ClassAssess  
WHERE Tno='19950018' AND TCo = '81001-01';  
ResultSet rs = stmt.executeQuery(SQL);  
while(rs.next()){  
    String Sno = rs.getString("Sno");           /*等价于rs.getString(1)*/  
    String strAssess = rs.getString("Assess");    /*等价于rs.getString(2) */  
    Boolean bCAtype= rs.getBoolean ("CAtype");   /*等价rs.getBoolean (3) */  
    String strFeedback = rs.getString("Feedback"); /*等价于rs.getString(4) */  
    System.out.printf("[%s,%b,%s]%n", strAssess, bCAtype, strFeedback);  
}
```

8.3.3 使用JDBC操纵数据库的工作流程

❖ 步骤5：释放资源

- 执行结束后，将与数据库进行交互的对象释放
- 释放资源有标准的顺序：
 - 关闭结果集：**ResultSet.close()**
 - 关闭语句执行类对象：**Statement.close()**
 - 释放数据库连接对象：**Connection.close()**

示例--基于JDBC实现数据库访问

- ❖ 建立与MySQL数据库的连接，假定服务器地址为127.0.0.1(MySQL服务器位于本机)，端口为3306，数据库名为dsdb，用户名为root，密码为db123456；访问其中login表格中的所有数据。

示例--基于JDBC实现数据库访问

```
import java.sql.*;
public class Test {
    public static void main(String[] args) {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection conn = DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:33061/dsdb",
                "root",
                "db123456");
            Statement stm = conn.createStatement();
            ResultSet rs = stm.executeQuery("select * from login");
            while(rs.next()) {
                System.out.println(rs.getString(1)+"\t"+rs.getString(2));
            }
            System.out.println("connection close");
            rs.close();
            stm.close();
            conn.close();
        } catch (SQLException se) {
            se.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

name	password
A ^B C varchar(50)	A ^B C varchar(16)
test	123456

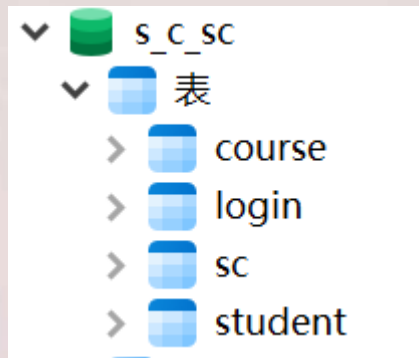
```
test      123456
connection close
```

基于JDBC的DBS开发示例—实验三

- ❖ 本实验要求利用**Java**编程语言，使用**JDBC**开发一个小型的学生选课管理系统，实现对数据的访问和增删改查操作。
- ❖ 该系统功能包括：学生登录查看课程和自己的选课情况；教师登录查看课程信息、学生信息和学生选课信息，并对这些信息进行管理。

基于JDBC的DBS开发示例—实验三

❖ 数据库包括的表格如下：



课程表 (**Course**)：学校开设的课程信息。

学生表 (**Student**)：学校的学生信息。

选课表 (**SC**)：学生的选课信息。

登录表 (**login**)：学生和老师的登录账号、密码信息。

基于JDBC的DBS开发示例—实验三

❖ 系统界面参考：



登录

欢迎进入学生成绩管理系统

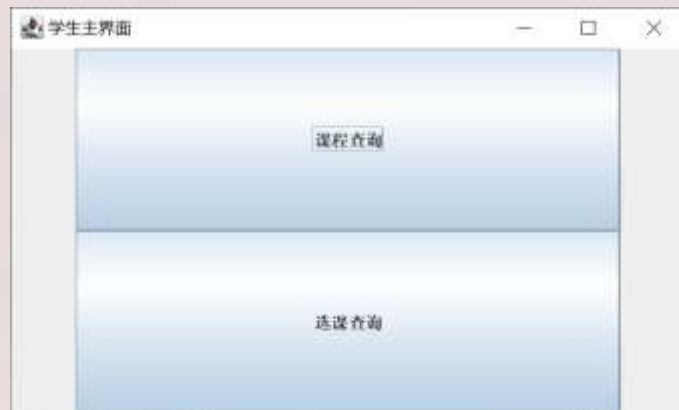
账号: T0001

密码:

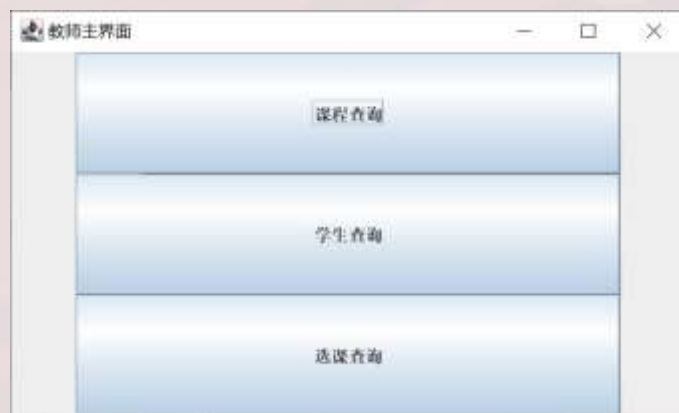
☐ Student ☐ Teacher

登录 取消

输入账号、密码，选择相应角色，点击登录按钮进行登录，成功则跳转到主界面



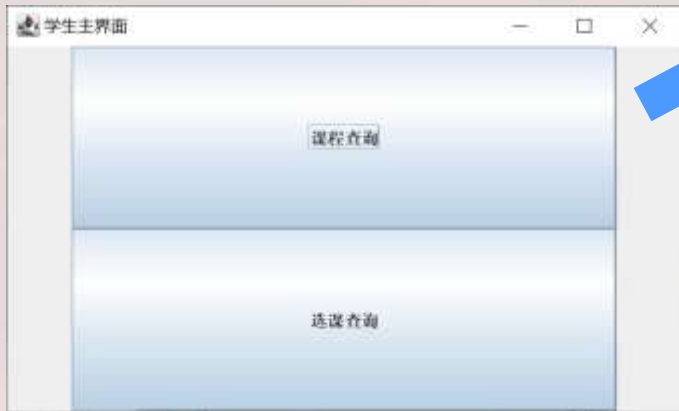
学生
主界面



教师
主界面

基于JDBC的DBS开发示例—实验三

❖ 系统界面参考:



课程查询

Cno	Cname	Ccredit	Cpno
81001	程序设计基础与C语言	4	81001
81002	数据库原理	4	81002
81003	数据库系统概论	4	81003
81004	信息检索概论	4	81001
81005	操作系统	4	81002
81006	Python语言	3	81003
81007	离散数学	4	81003
81008	人工智能概论	4	81003
81009	机器学习概论	2	81003
81010	机器学习时代中国特色社会主及理...	2	81003
81011	测试	3	81005

课程号 查找 刷新

在对应界面中直接显示所有数据

课程查询

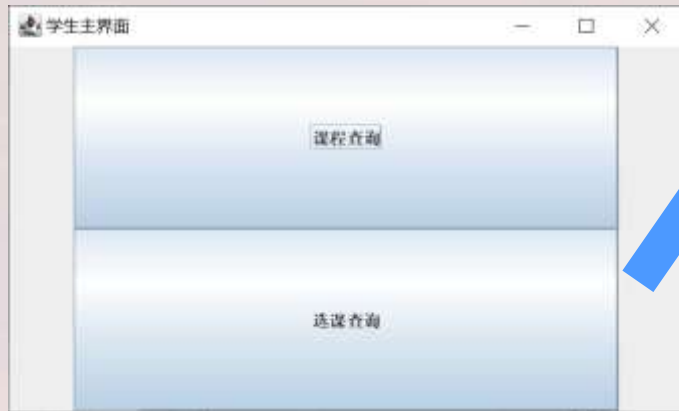
Cno	Cname	Ccredit	Cpno
81001	程序设计基础与C语言	4	
81006	Python语言	3	81002

课程名 查找 刷新

选择查找的列并输入信息，可以进行查找

基于JDBC的DBS开发示例—实验三

❖ 系统界面参考：



Cname	Grade	Semester	Teachingclass
程序设计基础与C语言	85	20251	81001-01
数据结构	96	20252	81002-01
数据库系统概论	87	20261	81003-01

课程名 ▼ 查找 刷新

在对应界面中直接显示所有数据

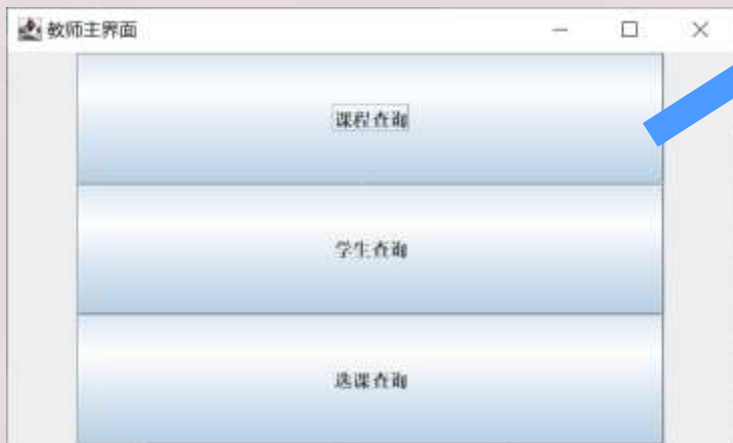
Cname	Grade	Semester	Teachingclass
程序设计基础与C语言	85	20251	81001-01
数据结构	96	20252	81002-01

开课学期 ▼ 25 查找 刷新

选择查找的列并输入信息，可以进行查找

基于JDBC的DBS开发示例—实验三

❖ 系统界面参考：



课程列表

增加 保存修改 删除

Cno	Cname	Ccredit	Cpno
81001	程序设计基础与C语言	4	81001
81002	数据结构	4	81002
81003	数据库系统概论	4	81002
81004	信息系统概论	4	81003
81005	操作系统	4	81001
81006	Python语言	3	81002
81007	离散数学	4	81003
81008	大数据技术概论	4	81003
81009	思想道德与法治	2	81009
81010	习近平新时代中国特色社会主义思想	2	81009
81011	测试	3	81005

课程号 查找 刷新

在对应界面中直接显示所有数据

课程列表

增加 保存修改 删除

Cno	Cname	Ccredit	Cpno
81001	程序设计基础与C语言	4	
81006	Python语言	3	81002

课程号 Python语言 查找 刷新

选择查找的列并输入信息，可以进行查找；点击“刷新”，显示所有信息

基于JDBC的DBS开发示例—实验三

❖ 系统界面参考：

课程列表

增加 保存修改 删除

Cno	Cname	Ccredit	Cpno
81001	程序设计基础与C语言	4	
81002	数据结构	4	81001
81003	数据库系统概论	4	81001 81002
81004	信息系统概论	4	
81005	操作系统	4	81001
81006	Python语言	3	81002
81007	离散数学	4	
81008	大数据技术概论	4	
81009	思想道德与法治	2	
81010	习近平新时代中国特色社会主义思想	2	
81011	测试	3	

课程号: [] 查找 刷新

点击“增加”按钮，新增一门课程

点击“保存修改”，对在表格中修改的数据进行保存

点击“删除”，可以删除选中的一行数据

添加课程

课程号: []

课程名: []

学分: []

先修课程: []

OK Cancel

课程列表

增加 保存修改 删除

Cno	Cname	Ccredit	Cpno
81001	程序设计基础与C语言	4	
81002	数据结构	4	81001
81003	数据库系统概论	4	81001 81002
81004	信息系统概论	4	
81005	操作系统	4	81001
81006	Python语言	3	81002
81007	离散数学	4	
81008	大数据技术概论	4	
81009	思想道德与法治	2	
81010	习近平新时代中国特色社会主义思想	2	
81011	测试	3	

课程号: [] 查找 刷新

消息: 保存成功! 确定