实验内容

- 计划课时：1.5
- 基本概念：数据库、JDBC、Statement、PreparedStatement、事务处理
- 进阶概念：JDBCUtil 与 DAO
- **基本考核**：1，3
- **特别考核：5、6、7**
- TODO：增加多表操作

题目 1:MySQL 数据库基本操作

1. 安装 MySQL，使用 root 用户登录。
2. 建立数据库 teststu，然后在 teststu 数据库中建立 student 表，字段要求如下。

学生 id:id, int,非空,主键,自增长 学号:　stuno, varchar(20), 非空 姓名:　name, varchar(20), 非空 学生年龄: age, int, 非空 性别: gender, varchar(1)， 出生日期: birthdate, date， 专业：major, varchar(20)

b. 查询学生(select 语句、order by、where)

参考命令：

```
//数据库操作
创建数据库：create database teststu;      #注意：;代表一条语句结束
删除数据库：drop database teststu;
使用数据库前要先打开数据库：use teststu;
查看当前打开的数据库：select database();

//表操作
创建表：create table students
create table students( id int not null primary key auto increment, stuno
varchar(20) not null, name varchar(20) not null, gender varchar(1)
not null, age  int, birthdate date, major varchar(20));

删除表：drop table students;
查看表结构：desc students;  //可用来查看表是否创建成功

//sql 操作
插入数据：insert into students(stuno,name,age,birthdate,gender) values('201710002', '
李四', 19, '1991-10-01','男');.  #参考 ppt，注意列名不要写错，否则会出现 unknown
column xxx in 'field list'...
查询数据：select * from students
```

参考资料：

1.MySQL 入门常用操作命令学习.txt

2.教学课件

3.MySQL 操作视频（1、2）

题目 2:使用 JDBC 连接数据库与 Statement

1. 在 Eclipse 中导入连接数据库所需要的 jar 文件。

a. 在项目下新建 lib 目录

b. 将相关 jar 包拷贝到 lib 目录

c. 在项目的 Build Path 的 Libraries 页导入该 jar 包

d. 在项目中运行 **ConnectMySQL.java**。

注意：Java 11 下需在 module-info.java 中添加 requires java.sql;

参考资料：

MySQL 操作视频（3），ConnectMySQL.java

常见问题：

No suitable driver found….-没有导入相应的数据库驱动 jar 包

Access denied….-一般是用户名密码错误

Table xxx doesn't exist..-xxx 表不存在，一般来说是表名写错

unknown column xxx in 'field list'-列名 xxx 不在表中，一般来说是列名写错

2. 使用 Statement 操作数据库

小技巧：在 Java 代码中编写 SQL 语句之前，应先在 MySQL 命令行下调试好 SQL 语句。

任务 1：针对上题的 students 表创建对应的实体类(Student 类)。将从数据库中获得数据装配成 Student 对象放入该列表（List<Student>），然后循环外输出该列表。

任务 2：编写如下方法（以下所有方法均需包含连接数据库、断开数据库的代码）

1. public static void displayAll(){....} //获取(select)并显示所有学生学号、名字和出生日期

2. public static int insert(Student stu){.....} //插入(insert)学生。返回值代表 insert 语句执行后影响的行数。如果执行成功返回 1，否则返回 0。实际上就是将 statement.executeUpdate() 的返回值返回。

3. public static List<Student> getAllByIdDesc(){.....} //获得所有学生列表（按 id 降序）。

```
select * from student order by id desc
```

4. public int deleteStudentById(int id){....}//按 id 删除(delete)某个学生，返回值为成功删除的学生的个数。即，将 executeUpdate 的返回值返回。

5. public static int updateStudentAge(int x){...}//更新(update)每个学生的年龄+x。

思考：

以上方法均包含什么共同代码？如何优化？

参考代码：

StatementTest.java、JDBCUtil.java

题目 3:PreparedStatement 与参数化查询

基本概念：

?为占位符，select * from students where id < ？ 语句中的?可以替换成所需的值。比如，

strSql = "select * from students where id < ?"；

pStatement = con.prepareStatement(strSql);

pStatement.setInt(1, 10); //这里的 1 代表第一个占位符，用 10 作为该处占位符的值。

改进：

通过前面的题目可以看到，每个方法都需要使用大量的重复语句设置 url、用户名、密码、获取连接、释放资源(ResultSet, Statement, Connection)，我们可以将重复内容封装起来实现复用。参见：“重要-数据库访问 dao 模式\utils”目录下的 **JDBCUtil.java**

任务：使用 PreparedStatement 根据用户指定的查询条件进行查询。

1. public static List<Student> getStudentByName(String name){...} // 根据 name 查找学生

select * from students where name = '小明'

select * from students where name like '周%'; //选择所有姓名以'周'开头的学生。

2. public static List<Student> getStudentsByAgeBelow(int age)

// 查找所有年龄小于 age 的所有学生并放入一个列表返回

3. public static void displayStudentBetween(String begin,String end)

//显示出生年月日在某个范围内的所有学生

Sql 语句中如果使用 between '2010-01-02' and '2011-02-23',改写成 between ? and ?

参考 sql: select * from student where birthdate between '2010-01-02' and '2011-02-23'

设置参数：使用 setString 设置日期类型参数。

4. public static double getAverageAgeAbove(int age){...} //显示所有年龄超过 age 的同学的年龄平均值(使用数据库中的 avg 函数对符合条件的记录计算)

select avg(age) avgAge from students where age>?

注意：avg(age) avgAge 指的是将年龄求平均后作为 avgAge 列输出，即列名为 avgAge

5.选做，考核：批量更新-如何批量插入 1000 个学生。

参考代码：

PreparedStatementTest1.java

JDBCUtil.java

题目 4:事务处理（可选）

问题：account 表中有两个用户 a 与 b，现在希望从 a 账户转账 100 块到 b 账户。这需要两个操作，1.从 a 减去 100。2.在 b 加上 100。假设第 1 步完成后，程序出现异常，导致第 2 步无法执行。那么实际上转账是失败的，然而 a 账户却损失了 100 块。试用事务处理相关机制，解决该问题。

说明：account 表中可以有 3 个字段：id int 主键 自增长，name 账户名 varchar(10)，balanc 余额 double。该表对应实体类是 Account 类。

编写：public void transfer(Account a, Account b, double x) throws Exception 实现从 a 到 b 转账 x，转账失败时会抛异常，异常中应包含失败的原因，比如“账户余额不足”、“其他原因”。

参考：

- Connection 中的 setAutoCommit(Boolean autoCommit)、commit()、rollback()方法。
- 改变账户余额使用 update 语句

题目 5:JDBCUtil 与 DAO 模式(特别考核)

通常，我们把对数据库的操作封装在一个类中，这种方法称之为 DAO(Data Access Object)模式。使用 DAO 模式完成数据库的相关操作。

1. 创建 model 包，在 model 包中根据 student 表创建 Student 实体类

2. 学习：数据库访问 dao 模式：

查看“**重要-数据库访问 dao 模式**”的源代码，学会使用“utils 目录”下的 JDBCUtil.java，使用方法参见“**dao**”下的 TestMain.java。同时注意参考 StudentDao 及其各种不同实现类的源代码。

3. 创建 dao 包

编写测试类 TestDaoMain，利用 JDBCUtil.java 实现简单的连接数据库返回里面的所有学生信息。

4. 编写 StudentDao 接口(里面声明了对学生的增删改查)及其实现类 StudentDaoJDBCImpl

接口包含如下方法：

增- int add(Student stu) //添加成功返回 1，否则返回-1

删- int delete(int sid) //删除成功返回 1，否则返回-1

改- int update(Student stu) //更新成功返回 1，否则返回-1

查- List<Student> findAll() //查找所有学生

Student findById(String sid),//根据 sid 查找学生

List<Student> findByName(String name),//根据姓名查找所有学生，使用%进行模糊匹配

说明：查询所有学生的信息，将 ResultSet 中的学生信息，构造成一个个的 Student 对象，并放入 List 中。编写一个函数，遍历该 List，将学生的姓名与年龄输出。

5. 编写一个函数将 List 中的所有 Student 取出，放入 Map 中，其中 key 为学生的姓名，value 为相应的学生对象。

6. 编写一个查找函数，可以根据给定的姓名,来找到对应的 Student 对象。

思考题：1.使用 DAO 模式访问数据库有什么好处？

题目 6.使用数据库连接池(选做)

C3P0 或者 DBCP 数据库连接池，注意:前面的 JDBCUtil.java 就不能直接用了，须编写新的 JDBCPoolUtil.java

参考资料：

数据库连接池目录下的参考资料

C3P0 参考资料：

http://www.mchange.com/projects/c3p0/

其他数据库连接池：

最近(2021 年)比较火热的数据库连接池：阿里巴巴的 Druid（功能丰富）以及 HikariCP(速度更快)。相关参考资料见：Spring Boot（六）：那些好用的数据库连接池们 https://zhuanlan.zhihu.com/p/139950940

题目 7：使用 DBUtils、JDBCTemplate(进阶)

Apache 提供了一个工具类库 commons-dbutils，集大简化 JDBC 编码工作量。
http://commons.apache.org/proper/commons-dbutils/

Spring 框架则提供了 JDBCTemplate，简化 JDBC 编码工作量。
http://www.cnblogs.com/Fskjb/archive/2009/11/18/1605622.html

尝试使用如上框架，改写自己的程序。