《数据库原理》实验报告

实验名称　　数据库接口实验

班　　级　　2014211304

学　　号　　2014211218

姓　　名　　史文翰

实验四 数据库接口实验

一、 实验目的

1．通过实验了解通用数据库应用编程接口 ODBC 的基本原理和实现机制，熟悉主要的 ODBC 接口的语法和使用方法；

2．利用 C 语言(或其它支持 ODBC 接口的高级程序设计语言)编程实现简单的数据库应用程序，掌握基于 ODBC 的数据库访问的基本原理和方法

3．学习 java 语言，并采用 jdbc 接口方式对数据库进行访问

二、 实验环境

MySQL 5.7 on win10 x64 with Python 3.6

三、 实验内容

1. 以教科书第四章关于 SQL 语言相关内容为基础，课后查阅、自学 ODBC 接口有关内容，包括 ODBC 的体系结构、工作原理、数据访问过程、主要 API 接口的语法和使用方法等。

2．. 以实验二建立的学生数据库为基础，编写 C 语言(或其它支持 ODBC 接口的高级程序设计语言) 数据库应用程序，按照如下步骤访问数据库

(a) Step1. ODBC 初始化，为 ODBC 分配环境句柄

(b) Step2. 建立应用程序与 ODBC 数据源的连接

(c) Step3. 利用 SQLExecDirect 语句，实现数据库应用程序对数据库的建立、查询、修改、删除等操作

(d) Step4. 检索查询结果集

(e) Step5. 结束数据库应用程序

四、 实验步骤及结果分析

1、 连接数据库

```
conn = MySQLdb.connect(
        host='localhost',
        port=3306,
        user='shiwenhan',
        passwd='abc123',
        db='stu',
        charset='utf8'
)
```

在 import MySQLdb 后，利用 connect 连接本地服务器的数据库。host 指定了服务器所处的位置，port 指定了访问端口，user 是数据库的用户名，passwd 是该用户的密码，db 是数据库名称，最后也是最重要的，我们要显示指定该数据库的字符集，因为它不是默认的。指定的字符集应该和数据库本身的设定吻合。

```
mysql> status
--------------
C:\Program Files\MySQL\MySQL Server 5.7\bin\mysql.exe  Ver 14.14 Distrib 5.7.17, for Win64 (x86_64)

Connection id:          5
Current database:       stu
Current user:           root@localhost
SSL:                    Not in use
Using delimiter:        ;
Server version:         5.7.17-log MySQL Community Server (GPL)
Protocol version:       10
Connection:             localhost via TCP/IP
Server characterset:    utf8
Db     characterset:    utf8
Client characterset:    utf8
Conn.  characterset:    utf8
TCP port:               3306
Uptime:                 5 days 10 hours 12 min 17 sec

Threads: 1  Questions: 165  Slow queries: 0  Opens: 135  Flush tables: 1  Open tables: 128  Queries p
0
--------------
```

2、 获取数据库游标

cur = conn.cursor()

cursor 方法返回一个游标对象，这个游标是 python 和 mysql 数据的桥梁。

3、 获取数据

（1） 查询

*c = cur.execute('select * from sel')*

info = cur.fetchmany(c)

for i in info:

print(i)

```
t_mysql
F:\Python36\python.exe G
('30201', 'C03', 40)
('30201', 'C04', 88)
('30201', 'C05', 93)
('30202', 'C03', 40)
('30202', 'C04', 40)
('30203', 'C03', 57)
('30203', 'C04', 50)
('30203', 'C05', 40)
('30204', 'C03', 54)
('30204', 'C04', 50)
('30204', 'C05', 40)
('30206', 'C03', 40)
('30206', 'C04', 40)
('30206', 'C05', 50)
('30207', 'C03', 82)
```

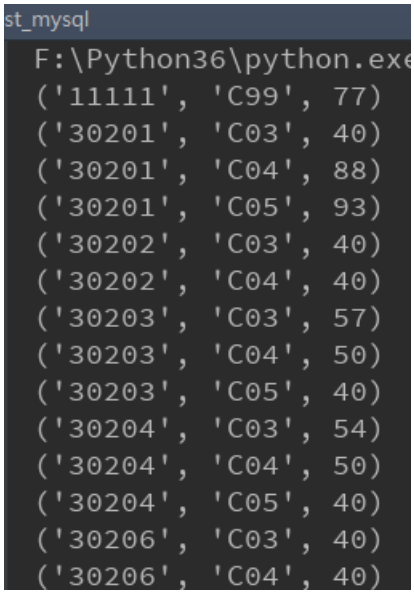
可利用 fetchmany 方法一次性获得多个数据，info 的本质是一个 list，list 中的每一个单元都是表中的一个元组，遍历并打印这个 list 即可看到查询结果。

请注意：对于增删改这三个命令，在 execute 后，需要执行一次 commit 命令来将这个事务提交到数据库本身，否则我们虽然能得到正确的结果，但实际

上数据库本身并没有发生变化（因为事务并没有被提交）。

(2) 插入

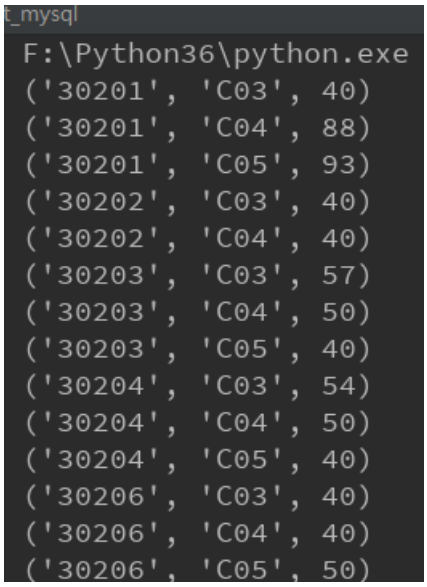
c = cur.execute("insert into sel values('11111', 'C99', 77)")



可以看到第一行就是我们增加的数据，我们同样用查询的方式来打印结果。

(3) 删除

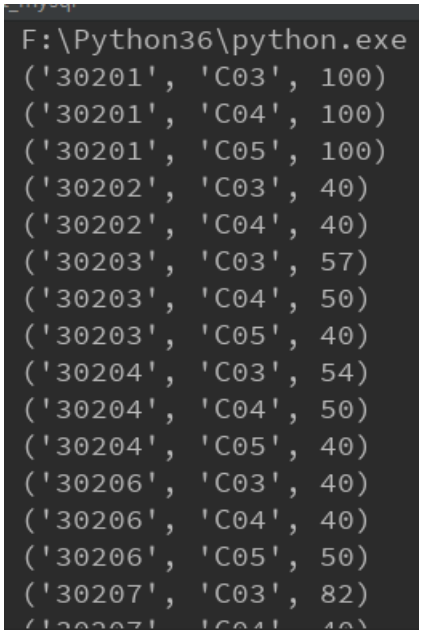
c = cur.execute("delete from sel where stu_no = '11111'")



我们将刚刚插入的数据删除，得到了如上的结果，数据确实已被删除。

(4) 更新

c = cur.execute("update sel set grade = 100 where stu_no = '30201'")



我们试图将学号为 30201 的几个成绩均改为 100 分，前三行表示了更改之后的结果。

4、 释放游标

cur.close()

5、 释放相关资源并断开数据库连接

conn.close()

五、 实验小结

本次实验将两种我之前学过的语言联系到了一起：Python 和 Mysql，这种数据接口为使用脚本语言直接操作 Mysql 提供了方便。

本次实验有两点需要特别注意：其一，即使你已经将 python3、mysql 和 python console 统一为 utf8 编码，但在执行 connect 语句的时候，必须显示说明被连接的数据库的字符集，即增加 *charset='utf8'*；其二，在进行增改查的命令时，在 execute 之后必须增加 commit 操作，才能将事务提交到数据库，最终完成数据库的修改。