**实验5：SQLAlchemy的注入防御实验报告**

姓名：江继军 学号：21373222 班级：213921

实验日期：2024年5月2日 实验地点：一号馆406 成绩：

1. **实验目的**

1.掌握Python中对象关系映射（ORM）框架实现SQL注入攻击防御的基本原理。

2.基于MySQL数据库、pymysql和SQLAlchemy库及其ORM组件，实现数据库连接;

3.基于理论课所学知识，实现安全的数据库插入和查询功能。

**二、实验内容**

1. 环境搭建

1. 数据库环境：本地MySQL数据库。
2. 安装Python SQLAlchemy库及其支持库，用于连接和操作MySQL数据库。

2. 数据库连接和ORM类创建

1. 基于SQLAlchemy库创建引擎连接数据库。
2. 基于SQLAlchemy的ORM组件，定义两个ORM模型类“students”和“course”，用于映射对应的表格。
3. 数据库插入和查询
4. 创建数据库会话，向创建的两个表格插入数据，并提交事务。
5. 输入查询语句并打印命令执行结果，如发生异常需提示。

**三、实验环境**

分项列出主要实验环境（主机/操作系统、Python开发环境、Python版本、第三方库版本等）：

1. 操作系统：Win11
2. Python开发环境：pycharm-community
3. Python版本：3.9
4. pymysql版本：1.1.0
5. sqlalchemy版本：2.0.29

**四、实验步骤**

写出环境搭建、数据库连接、ORM类创建、数据库连接、数据插入和查询等步骤（采用1.2.3.4的序号方式列出详细步骤；写明所编写的代码；贴出效果示意图，例如数据库成功连接、表格成功创建、数据插入和查询结果截图等）。[本次报告不要超过8页]

1. 环境搭建：

本地安装MySQL数据库，创建新数据库，安装Python SQLAlchemy库，通过创建 SQLAlchemy中的引擎，验证数据库成功连接（写明关键步骤，给出连接成功截图）。

测试数据库连接是否成功代码如下：

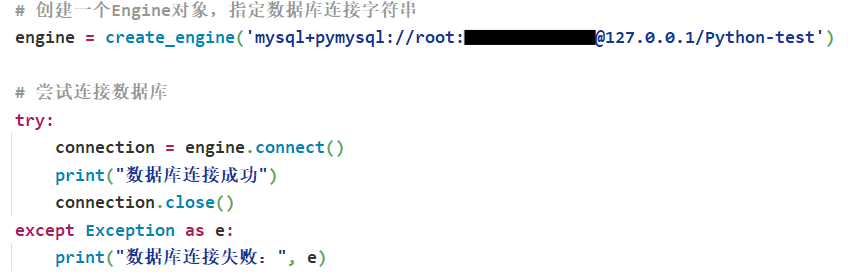


图 1 测试连接数据库成功代码

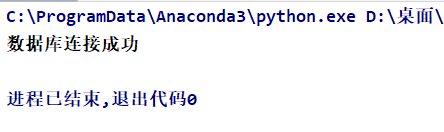


图 2 数据库连接成功

1. 定义ORM类，创建数据库表格：

基于SQLAlchemy的ORM组件，定义两个ORM模型类“students”和“course”，用于映射对应的表格，“students”类包括ID、姓名、邮件、密码，“course”包括ID、名称（写明所编写的代码，给出表格创建结果截图）。

定义ORM类，创建数据库表格代码：

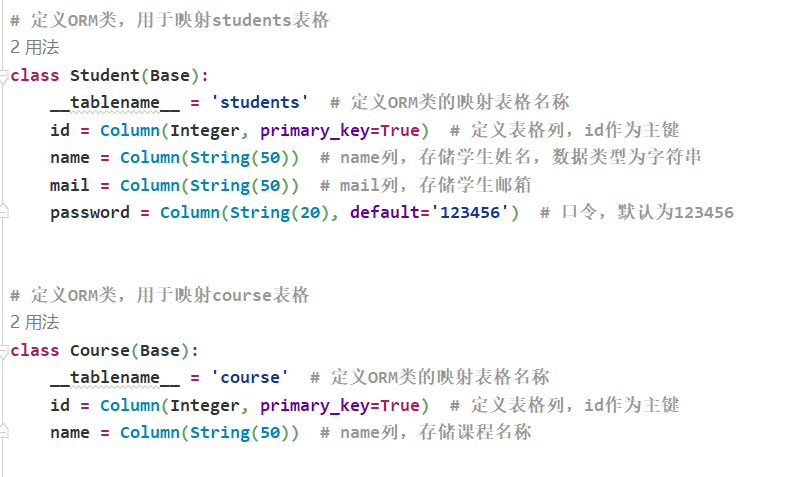


图 3 定义ORM类，创建数据库表格代码



图 4 创建表格成功截图

1. 数据库插入和查询：

创建数据库会话，向创建的两个表格插入数据，并提交事务，之后输入查询语句并打印命令执行结果，如发生异常需提示，可尝试输入多条查询语句（写明所编写的代码，给出SQL语句执行成功的结果截图）。（如尝试课程中未讲授的方法进一步提升安全性或另加功能，酌情+1~8分）。

增查数据代码：



图 5 增查数据代码

输入增查语句，查看结果：



图 6 增查内容代码

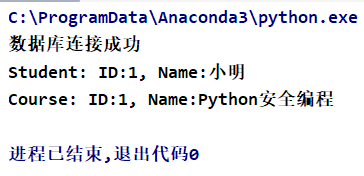


图 7 命令行输出结果

**五、问题和解决方法**

1. 数据库连接问题

创建Engine连接数据库时，字符串格式为 dialect+driver://username:password@host:port/database

* dialect：数据库，如：sqlite、mysql、oracle等
* driver：数据库驱动，用于连接数据库的，本次实验使用pymysql
* username：用户名
* password：密码
* host：IP地址
* port：端口
* database：数据库

1. SQLAlchemy中"default"和"server\_default"关键字的区别

default："default"关键字用于在Python中设置默认值。当我们创建一个新对象时，如果没有提供该列的值，那么默认值就会被使用。这个默认值是在Python中处理的，不会传递到数据库中。

server\_default："server\_default"关键字用于在数据库层面设置默认值。这意味着，如果在执行INSERT语句时未提供该列的值，数据库将使用这个默认值，而不是在Python中处理。

**六、实验总结**

通过本次实验，我对Python中的对象关系映射（ORM）框架有了更深入的理解和实践。使用SQLAlchemy库及其ORM组件，我成功实现了与MySQL数据库的连接，并创建了对应的ORM模型类来映射数据库表。在实验过程中，我学会了如何通过会话来执行数据库的插入、查询等操作，并且了解了事务的使用，确保了数据的一致性和完整性。

此外，我也意识到了安全性的重要性。在实验中，我们使用SQLAlchemy中的ORM组件访问数据库时，可直接防御SQL注入风险；自行编写语句查询时，可以通过ORM组件的Session中提供的text对象实现参数绑定，防止SQL注入风险。

总的来说，这次实验不仅提升了我的数据库操作技能，也加深了我对数据库安全和性能优化的认识。

**七、附件**

提交可执行的原程序压缩包，随电子版实验报告提交，文件名称为学号+姓名+实验5。