Lab_D：iGEM-Software数据库初探

信智学部（除了网安学院）的同学们都能看得出来，本次项目的命名参照了ICS课程的实验命名。

> 都是在寒假写代码，没啥不对劲的吧（大雾

0. 前言

就像我在“迟来的任务安排”中说的那样，~~我在过年期间~~自己把这次数据库大作业的流程走了一遍~~，同时借助了些许外力帮助~~。

咳咳！说重点！为了能让大家用**最短的时间**了解并熟悉PyMySQL的使用方法，并且写出来一个不是黑框框的数据库操作程序，本次的数据库作业采取了**发布框架、代码“完形填空”**的策略——借助些许外力，省去各位死扣tkinter写图形界面的环节，将重心放到如何使用PyMySQL库上来。

本次的数据库“实验”包含了以下几个小任务：

0. 利用大家之前已经学过的知识（同时善用互联网，下同）阅读SQL脚本文件，简要地说明我给出的SQL脚本文件的作用（写一个小文档交上来即可，推荐使用Markdown并导出成pdf）。
1. 打开Navicat，创建数据库，执行我提供的SQL脚本文件，完成数据库的建表工作。（此时你应该能看到你的数据库下面有两个表：`adm_account`和`user_info`）
2. 按照Lab_D_Release项目中注释的提示（**请务必注意找“TODO”**），补全缺失的Python或者SQL代码。
3. 按照视频教程（见第4节），完成”Lab_G“，即完成对代码在GitHub上的提交。

最后强调一件事：强烈建议大家通读完本文档一次之后再动手开始写代码！

最后强调一件事：强烈建议大家通读完本文档一次之后再动手开始写代码！

最后强调一件事：强烈建议大家通读完本文档一次之后再动手开始写代码！

1. 本次的数据库结构简介

就像我在”MySQL基础”中说的那样，一个数据库中可以包含多个数据表，一个数据表中可以包含多个数据项。

本次我们用于练习的数据库名称定为“igem001”。当然，你也可以起一个别的名字，不过切记在`connect_database.py`（见2.2节）中更改使用的数据库名称。

不妨假设我们的数据库名称就是“igem001”好了。在使用`.sql`脚本文件初始化数据库后，数据库内部包含两个数据表`adm_accounts`和`user_info`，其表结构如下图所示：

表：adm_account

id	username	password	salt

表：stu_info

id	name	gender	stu_id	GPA	major

`adm_accounts`存储4个字段：管理员的唯一id、管理员用户名、加密后的密码，用于加密的盐（salt，见第2.3节）。

`user_info`存储6个字段：学生的唯一id、学生的用户名、学生的性别、学生的学号、学生的GPA和学生的专业。

它们对应的建表语句如下：

```
CREATE TABLE `adm_account`  (
  `id` int NOT NULL AUTO_INCREMENT,
  `username` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NULL
DEFAULT NULL,
  `password` varbinary(255) NULL DEFAULT NULL,
  `salt` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NULL
DEFAULT NULL,
  PRIMARY KEY (`id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 1 CHARACTER SET = utf8mb4 COLLATE =
utf8mb4_general_ci ROW_FORMAT = Dynamic;

CREATE TABLE `stu_info`  (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(255) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NULL
DEFAULT NULL,
  `gender` varchar(255) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NULL
DEFAULT NULL,
  `stu_id` varchar(10) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NULL
DEFAULT NULL,
  `GPA` decimal(3, 2) NULL DEFAULT NULL,
  `major` varchar(255) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NULL
DEFAULT NULL,
  PRIMARY KEY (`id`) USING BTREE,
  UNIQUE INDEX `unique_std_id`(`stu_id` ASC) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 1 CHARACTER SET = utf8mb4 COLLATE =
utf8mb4_general_ci ROW_FORMAT = Dynamic;
```

2. Lab_D_Release框架简介

本次的Lab_D的框架包含四个文件：`main.py`、`connect_database.py`、`account.py`和`CRUD_database.py`。

下面分别介绍一下这四个文件：

2.1 `main.py`

`main.py`是整个项目的入口文件，我们在调试的时候只需运行这个文件即可。

这个文件中import了`Main`类（主窗口类）并构造了一个实例，运行本文件即可唤起本次数据库大作业的登录窗口。

2.2 `connect_database.py`

`connect_database.py` 实现了我们操作数据库的第一步——连接数据库。

大家可能觉得这个连接过程写得有点云里雾里，但其实这样的写法是PyMySQL包的一个使用"八股"，**基本上所有人都这么写，大家只要记下来就好。**

在这个文件中，你需要理解如何使用**connect方法**和**cursor方法**（它们返回的对象是我们操作数据库的基础），并按照我在注释中留下的TODO，正确地填写你在**安装MySQL环境时**设置的管理员用户名和密码。

我在此模块中为大家编写了测试用例，大家在实现 `connect_database` 函数之后可以直接运行该文件来检查自己的实现。假如得到和图片**类似**的输出，则证明大家的数据库建立（用Navicat，见本文档的第4节）和数据库连接（就是指 `connect_database` 函数）的实现是没有问题的，可以继续进行下面的工作。

```
((4, 'admin', b'm/\x1db\x92\xce\xc4\x93\x93\xb9\x0c\x8fNg\xa8B', 'b1b0c4d0-ca1d-11ee-95e6-e8808817ee4d'),)

进程已结束，退出代码为 0
```

怎么样，对大家来说应该不是很难吧？下面两个文件的难度稍微大一些，**建议大家尽早开始阅读代码并尝试实现。**

2.3 `account.py`

`account.py` 中包含了用户登录和注册功能以及其GUI的实现。就像我说的，大家暂时不用去考虑如何写图形用户界面，我已经替大家写好了。对于GUI界面而言，大家只需要理解一下每一条语句在干什么就可以了，我们以后再写更正式的。

在 `user_sign_up` 函数中，我给了大家一个用 `cursor` 对象去操作数据库的例子：`cursor` 的 `execute` 方法接受以字符串形式传入的SQL语句，并对其连接的数据库进行相应操作。剩下的SQL语句，就需要大家自己去完成了。另外，之前发布文档的时候和大家说过，用户的密码需要加密存储。在这个函数中，大家需要实现这一点，请参考实现MySQL AES_ENCRYPT函数带盐 - 对字段值做加密处理，稍微转个弯就可以实现了。

这里，我给出了盐和密钥的生成过程，请大家完成将用户名、密码和盐存入表中的 `INSERT` 语句。

`user_sign_in` 函数中，我们要实现密码的解密，请大家同样参考上面的链接实现。另外，我们还要实现用户的登录，请大家在TODO对应的代码段内实现用户的登录验证。

2.4 `CRUD_database.py`

恭喜各位，你们已经来到了本次实验的最后了！这个文件中包含了数据库的增删改查操作的实现，大家可能需要在这里多花一些时间，胜利就在眼前！

在这个文件中,我们实现了用于增删改查页面的图形用户界面，以及用于增删改查的一系列函数，我们来挨个看一看吧。

2.4.1 增：`insert_item`

按照正常人的思维来说，想要向一个学生信息管理系统中添加学生的信息，首先要判断学生信息是否存在。结合实际情况，我们允许重名的学生存在，但出现不能姓名和学号同时相同的情况。因此，大家需要编写按照学生的学号和姓名查询数据项的SQL语句，这是第一个TODO的内容。

在已经确认不存在信息重复的情况下，我们就需要进行信息的添加了，这便是第二个TODO的内容。

2.4.2 删：`delete_item`

删除学生信息时，要通过获取所选学生的学号和姓名来确定应删除的唯一数据项，记得使用`WHERE`子句来编写删除学生信息的SQL语句。

2.4.3 改：`update_item`

想要更改学生信息，首先要在数据表中定位到该学生所对应的数据项。由于我们可能把学生的五个信息全部更改一遍，所以，在这里我们需要查询学生的唯一id，以此来确定该更改哪个数据项。大家需要编写按照学生id查找学生并进行修改的SQL语句。

提示：`WHERE`子句上大分！

2.4.4 查：`search_item`

编写通过学号、姓名或专业查询数据项的SQL语句，注意`WHERE`子句中应该用`OR`逻辑。

祝贺大家！各位同学到这里已经完成了代码的编写工作，现在结合Navicat来调试你的程序吧！

3. Lab_G简介

3.0 前言

大家在写代码的时候，可能会遇到这种情况：在修改了一次代码后，发现程序的效果与预期不符，而想回退到上个正确版本时，却发现记不清到底有哪些地方进行了更改。纠正这样的错误将耗费大量的时间，甚至最后也不一定能成功。为了解决这些问题，我们迫切地需要一个代码版本管理工具。

基于上述原因，我向各位隆重地介绍——闻名中外的——Git工具。

3.1 介绍

Lab_G并不需要各位同学在代码中再增添任何的内容。各位同学需要按照我给出的视频教程（见第4节）完成Git环境的搭建和GitHub远程仓库（由我管理）的连接，并将自己最终的代码push到我指定的远程仓库中。除非情况十分特殊，否则，为了调动大家使用Git的积极性，我们验收的标准为各位**提交到GitHub仓库中的代码。**

由于目前仓库的搭建尚未完成，加之大家一时半会儿也写不完Lab_D，Lab_G的细节将后续在群里以文档或者QQ消息的方式给出，**请大家持续关注**！

3.2 碎碎念

可能大家会觉得为了一个小作业去学习一门全新的工具有些小题大做。但是，各位同学需要知道，在我们以后的开发过程中，将会不可避免的遇到代码版本回溯以及多人协作的问题。假如大家没有正确的使用Git（在iGEM-2024这里，尤其指GitHub）的知识，那么我们在协同推进工程的时候可能会遇到不小的阻力，有时甚至会造成无法挽回的错误。综上，我们在这个寒假还是有必要学一学怎么用Git的。考虑到各位同学的春季学期都比较繁忙，我觉得在开学前这个时间节点上让大家花两三个小时（**是的，对于基础操作而言差不多够了**）去速通一下Git（以及基础的Linux命令）的使用也是合理的——**总比大家在期中/期末周或者在暑假的时候现学要强吧**？

最后的最后，对于和我一样在信智学部的同学们，以及将来要写代码的其他方向的同学们（包括但不限于生物信息学等）来说，有的时候，学会Git对你的开发过程可以起到至关重要的作用，乃至于成为你的救命稻草。

4. 大家的任务：全流程工作流展示

好了，大家看到这里也差不多知道我们这次要干什么了。为了大家方便对照自己的进度，我在这里把大家要做的事情罗列一下：

- **请再次确认自己的环境已经搭好了。**我相信大家，但按照规范我还是要在这里提醒一下。
- 实际上，SQL脚本文件是一个文本文件。阅读它，并简单说明其作用，写成文档上交（上交到GitHub仓库中）。命名规则：`学号_姓名_SQL脚本小结.pdf`（推荐使用Typora+Markdown编写并导出成pdf）。
- 使用Navicat新建数据库并调用SQL脚本文件建立初始化的数据表（`adm_account`表中已经含了一条用于调试的初始信息：用户`admin`，密码`pw123`）。
- 完成`connect_database.py`中的TODO（**共2处**），并运行该文件以检查数据库是否正常连接。
- 完成`account.py`中的TODO（**共3处**）。
- 完成`CRUD_database.py`中的TODO（**共6处**）。
- 参考这个教程【GeekHour】一小时Git教程，将所有文件打包并上传至GitHub仓库（Lab_G）。
- 享受寒假吧！

5. 代码风格规范建议

我在这里列几条能帮助大家提高开发效率，并且写出更加美观的代码的建议：

- 保持写注释的好习惯。假如说实在懒得一个字一个字地敲，可以用Copilot等工具（顺带一提，GitHub有教育优惠，可以免费用，建议人均整一个）辅助生成注释再小改，但切记不要完全不写注释。否则，将来可能没人能看得懂你的代码，**包括你自己。**

> 第一条建议总是最重要的。

- 在命名变量（以后可能还包括函数）时，尽量使用小写字母加下划线式的命名方法，这才是Python的风格（比如说`this_is_a_variable`）。
- 尽量用英文去命名变量。用汉字命名变量也勉强可以接受（仅限Python），但不要用拼音命名变量！
- 变量名的设置尽量符合其作用并有意义——好的命名可以让你少写点注释。
- ……暂时就这么多了！

6. 最后的最后

我当时在讲"MySQL基础"的时候，仅仅停留在了在命令行界面操作数据库这个层面上，并未提及在Python中该如何调包操作数据库。假如大家对这一方面有不清楚的，可以参考这个教程（链接1、链接2）中的**部分**视频补一补基础（尤其是Day1的视频）。两个链接的内容是**一样的**，不过交替看能保证大多数视频都能找到**AI字幕**，方便速查。说"**部分**"是因为**很多内容对于这个实验来说有点太超前了**，大家**暂时**没必要看。

同时，我也把这个系列视频的Markdown课件找到并放在群里了（`python_course-master.zip`），大家需要的话可以自取。我在按照该课程学习的时候，在其中增加了一些辅助理解的注释，大家看视频的时候对照一下就能发现了。

7. 致谢

感谢各位抽出时间来阅读文档，你们是最棒的！