管理信息系统（MIS）设计与开发

1 实验目的

1. 熟悉数据库设计与开发的基本步骤和流程。
2. 熟练掌握数据库开发语言与工具。
3. 体会数据模型和数据模式在数据库设计中的作用，以及与数据库管理系统之间的关系。

2 实验平台与工具

1. Windows、Linux 操作系统
2. Java、Python 等语言
3. 数据库管理系统 PostgreSQL
4. ER 设计工具（如 EZDML，**http://www.ezdml.com/**）。

3 实验内容与要求

按照“图书借阅系统数据库设计.pdf”完成数据库管理系统设计与开发的全过程。

1. 具有图形化的用户交互界面。
2. 实现完整的业务功能。
3. 可以粘贴必要的关键代码，但不要粘贴大段过多的代码。

4. 要求

（1）独立完成，严禁相互抄袭（如有发现抄袭和被抄袭均判为 0 分），以及从网络上直接摘抄别人的观点和总结（该行为将影响报告成绩）。

（2）实验报告符合学术写作的排版要求，请参考群文件中的“报告模板.docx”和“参考文献格式.docx”的排版格式。

（3）实验报告内容详实，采用图文混合的方式叙述安装和配置过程。

Tip：Win+Shift+S 在 Windows 中可以快速截屏。

（4）报告文件见附件，提交报告时请以附件形式插入到超星作业中。

实验报告

报告标题：管理信息系统（MIS）设计与开发
学号：
姓名：
日期：

一、实验环境

1. 操作系统：
Microsoft Windows 10 家庭中文版（10.0.19045 内部版本 19045）

2. 数据库管理软件（含版本号）：
postgresql-16.0-1-windows-x64

3. 设计与开发工具：
ER 设计工具（如 EZDML，**http://www.ezdml.com/**）。

二、实验内容及其完成情况

所有实验 4 包含的内容均可简化表达，不需要详细描述

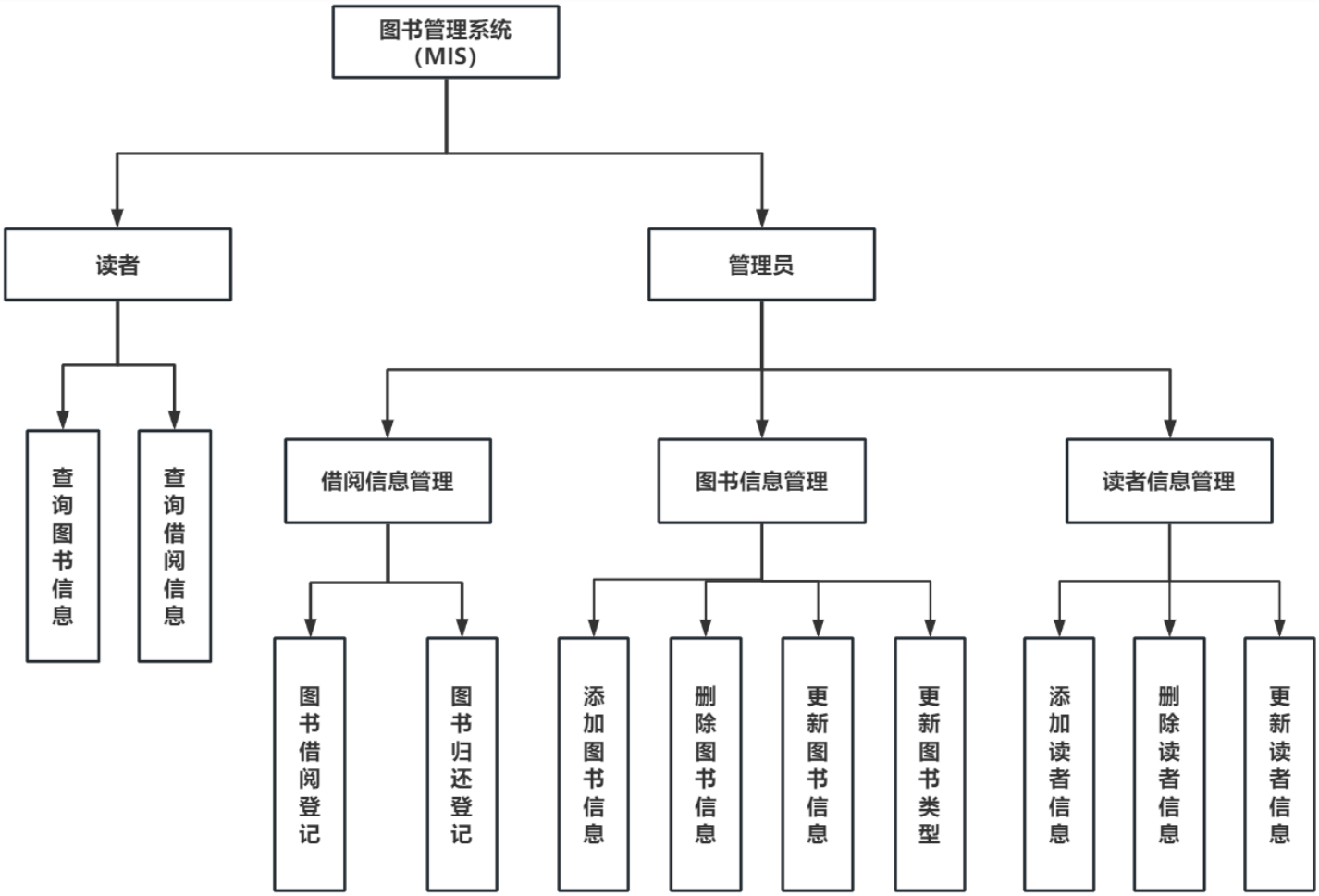
1. MIS 功能介绍（简要介绍即可）

文档中要求，系统应该实现以下功能：图书管理员可以维护图书信息，包括增加新书、修改图书信息、办理图书借阅登记、图书归还登记、过期图书处理、丢失图书处理及读者借阅证件信息的维护等；而读者可以实现借书、还书、查阅图书信息、查询借书信息等。

因此使用 python 语言设计 MIS，功能表如下。

读者	查询图书信息	
	查询借阅信息	
管理员	图书借阅登记	借阅信息管理
	图书归还登记	
	添加图书信息	图书信息管理
	删除图书信息	
	更新图书信息	
	更新图书类型	
	添加读者信息	读者信息管理
	删除读者信息	
	更新读者信息	

2. MIS 系统基本框架图



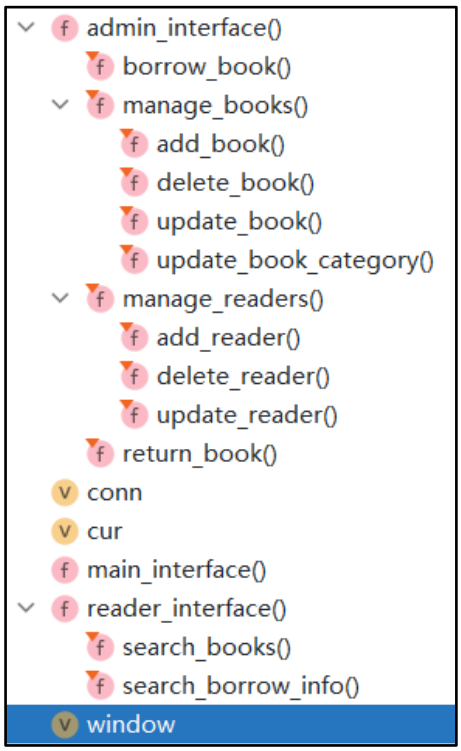
3. 逻辑结构设计

逻辑结构如下图所示，在主界面 main_interface()中创建管理员、读者按钮，分别导向管理员界面 admin_interface()和读者界面 reader_interface()；

其中，管理员界面又设有图书管理界面 manage_books()和读者信息管理界面 manage_readers()，以及借书、还书这两最常用的功能；图书管理界面包括对图书信息和图书类型信息的增删改，读者信息管理界面包括对读者证件信息的增删改；

读者界面为查询功能，查询图书信息以及个人的借阅信息。

本 MIS 设计要求：借书、还书操作仅由管理员登记完成，即读者无法个人完成借阅登记。



4. **物理设计与实施（可选）**

利用实验 4 构建的数据库 LibraryBorrowingSystem，在 python 中连接该数据库，对其进行相关操作，操作后关闭连接。

```
# 创建数据库连接
conn = psycopg2.connect(database="LibraryBorrowingSystem", user="postgres",
                        password="tara090729", host="localhost", port="5432")
cur = conn.cursor()

# 创建主窗口
window = tk.Tk()
# 运行主界面
main_interface()
window.mainloop()

# 关闭数据库连接
cur.close()
conn.close()
```

5. **业务功能的设计与实现**

首先构建主界面，创建窗口，设置“管理员”与“读者”两个按钮，指向对应页面。

```python
# 创建主界面
def main_interface():
    window.title("图书管理系统")  # 设置窗口标题为"图书管理系统"
    window.geometry("200x150")  # 设置窗口大小为400x300

    label = tk.Label(window, text="图书管理系统", font=tkFont.Font(size=15))
    label.pack()

    # 创建管理员按钮
    admin_button = tk.Button(window, text="管理员", command=admin_interface)
    admin_button.pack(side="top", pady=10)

    # 创建读者按钮
    reader_button = tk.Button(window, text="读者", command=reader_interface)
    reader_button.pack()
```

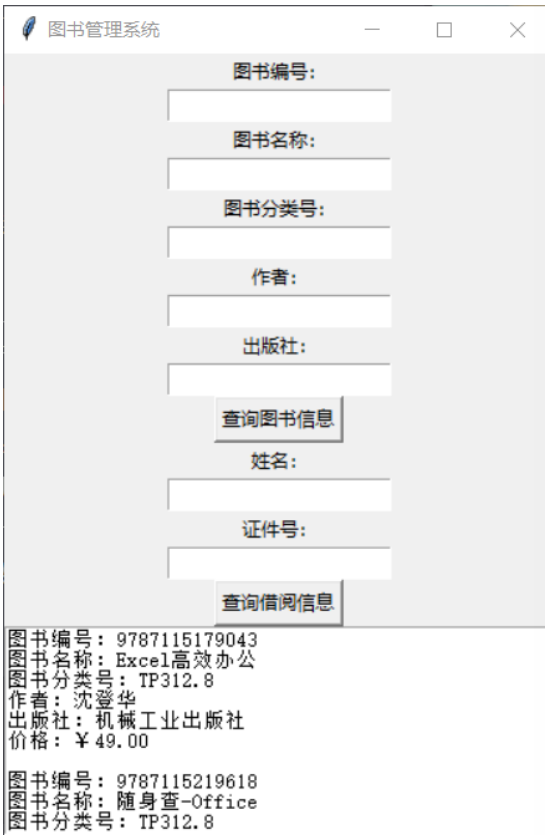
点击“读者”，进入读者界面。

- **功能 1：查询图书信息**

  根据输入框获取到的字符串，构建查询语句，执行查询语句，将结果输出在下方方框中。其中，图书名称、作者、出版社可以是关键字查询；同时，查询条件可以是 0 个、1 个，也可以是多个。

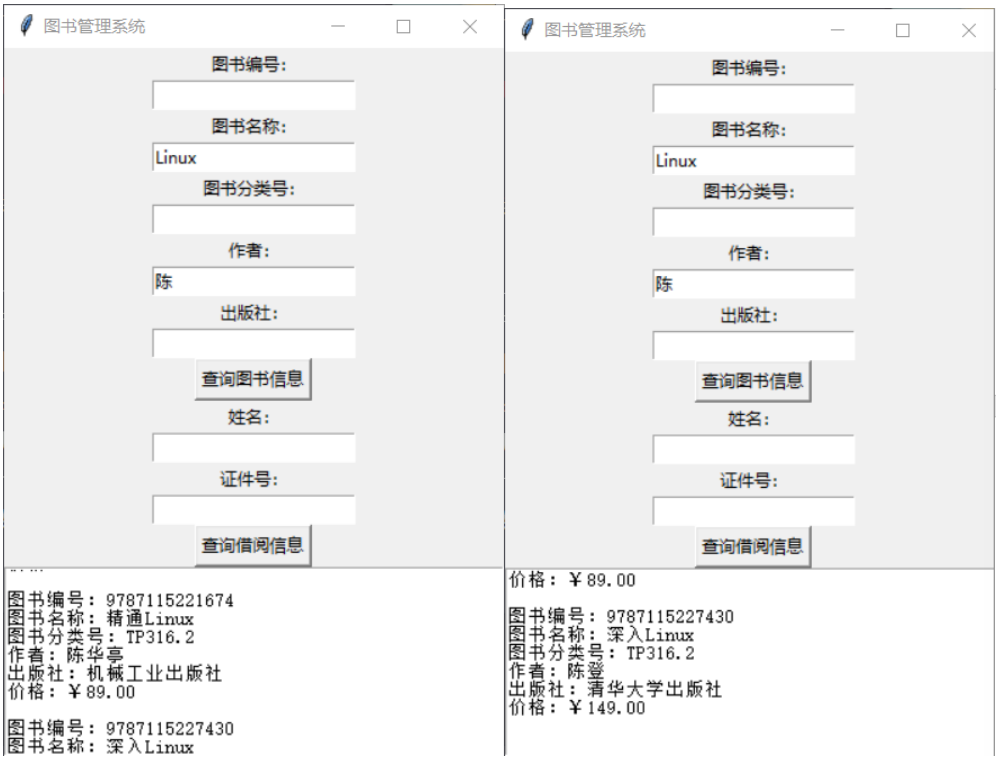
```
# 创建查询图书函数
def search_books():
    # 获取查询条件
    book_id = book_id_entry.get()
    book_name = book_name_entry.get()
    book_category = book_category_entry.get()
    book_author = book_author_entry.get()
    book_publisher = book_publisher_entry.get()

    # 构建查询语句
    query = "SELECT * FROM 图书 WHERE 1=1"
    if book_id:
        query += f" AND 图书编号 = '{book_id}'"
    if book_name:
        query += f" AND 图书名称 LIKE '%{book_name}%'"
    if book_category:
        query += f" AND 图书分类号 = '{book_category}'"
    if book_author:
        query += f" AND 作者 LIKE '%{book_author}%'"
    if book_publisher:
        query += f" AND 出版社 LIKE '%{book_publisher}%'"

        # 执行查询
        cursor = conn.cursor()
        cursor.execute(query)
        books = cursor.fetchall()
        cursor.close()

        # 显示查询结果
        result_text.delete("1.0", tk.END)
        if books:
            for book in books:
                result_text.insert(tk.END,
                f"图书编号：{book[0]}\n图书名称：{book[1]}\n图书分类号
        else:
            result_text.insert(tk.END, "未找到符合条件的图书。")
```

  - 无关键词查询



■ 多个关键词查询

数据库中有两本关于 Linux 的书，作者都姓陈，查询结果如下。可以看到，关键词查询将两本书都查了出来。



- **功能 2：查询借阅信息**

可以根据证件号（主码）查询借阅信息，同时，也支持根据姓名查找，在代码中要先用姓名在“读者”表查找对应的证件号，然后去“借阅”表查询借阅信息。

```
# 如果只输入了姓名，则查询对应的证件号
if reader_name and not reader_id:
    reader_query = f"SELECT 证件号 FROM 读者 WHERE 姓名 LIKE '%{reader_name}%'"
    cursor.execute(reader_query)
    reader_ids = cursor.fetchall()

    borrow_info = []
    for reader_id in reader_ids:
        borrow_query = f"SELECT * FROM 借阅 WHERE 证件号 = '{reader_id[0]}'"
        cursor.execute(borrow_query)
        borrow_info.extend(cursor.fetchall())
```

  - 根据证件号查询

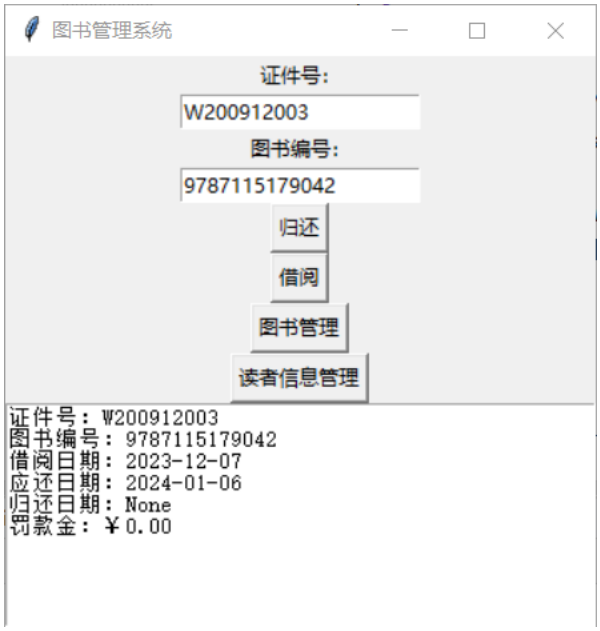
  - 根据姓名查询

在主界面点击“管理员”，进入管理员界面。

- **功能 3：图书借阅登记**

借书时必须要输入读者的证件号和所借阅图书的图书编号，因为这两者是“借阅”表的主码；要获取当前的日期以及 30 天后的日期，作为借阅日期和应还日期，向“借阅”表插入数据。

```
# 获取当前日期和应还日期
borrow_date = datetime.now().strftime("%Y-%m-%d")
return_date = (datetime.now() + timedelta(days=30)).strftime("%Y-%m-%d")

borrow_data = (reader_id, book_id, borrow_date, return_date)
insert_query = "INSERT INTO 借阅(证件号,图书编号,借阅日期,应还日期) VALUES (%s,%s,%s,%s)"
cur.execute(insert_query, borrow_data)
conn.commit()
```

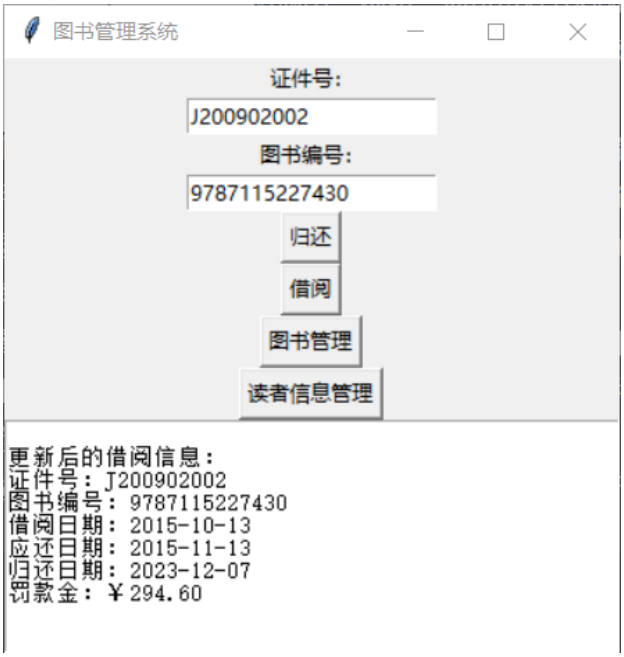


- **功能 4：图书归还登记**

在读者归还图书时，需要给归还日期赋值，同时根据归还日期和应归日期计算罚款金，超期每天需支付 0.1 元，未超期则罚款金置零。

```
# 如果找到对应借阅信息
if borrow_info:
    # 展示借阅信息
    result_text.delete("1.0", tk.END)
    result_text.insert(tk.END,
                       f"证件号：{borrow_info[0]}\n图书编号：{borrow_info[1]}\n借阅日期：{borrow.

    # 获取当前日期
    return_date = datetime.now().strftime("%Y-%m-%d")
    # 计算超期天数和罚款金额
    due_date = datetime.strptime(borrow_info[3].strftime("%Y-%m-%d"), "%Y-%m-%d")
    days_overdue = (datetime.strptime(return_date, "%Y-%m-%d") - due_date).days
    fine = 0.1 * days_overdue
    if fine <= 0:
        fine = 0.00

    # 更新借阅信息
    cur.execute("UPDATE 借阅 SET 归还日期 = %s, 罚款金 = %s WHERE 证件号= %s AND 图书编号 = %s",
                (return_date, fine, reader_id, book_id))
    conn.commit()
```

根据“借阅”表相关信息，归还一本已超期未归的书，会计算出相应的罚款金。



在管理员界面点击“图书管理”，进入图书信息管理界面。

- **功能 5：添加图书信息**

本系统要求，输入“图书”表的所有属性，才能添加图书，否则会提示输入完整图书信息。

- **功能 6：删除图书信息**

本系统要求，只能根据图书编号（主码）对图书进行删除，以此来处理过期图书或图书丢失问题。

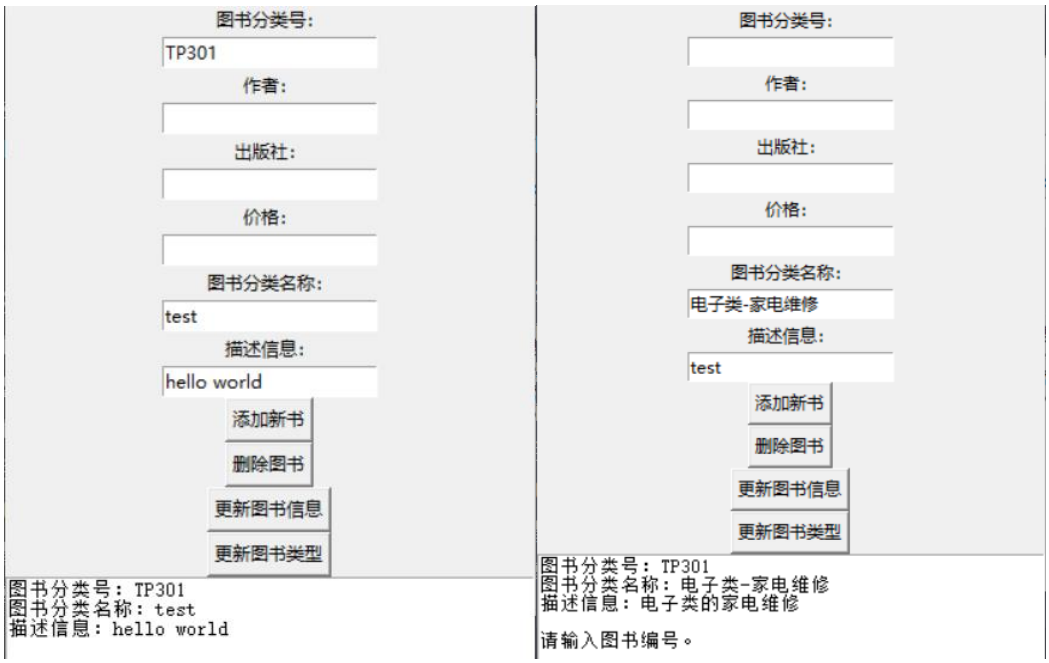
- **功能 7：更新图书信息**

本系统要求，必须有图书编号（主码）才能对图书信息进行更新，而其他属性可以为 1 个或多个。

其他更新模式，例如知道图书名称更新图书作者、出版社、价格，知道作者更新图书名称等等，情况过多，操作类似读者界面的用“姓名”查询“借阅信息”，分部执行，因此在此不做演示。

- **功能 8：更新图书类型**

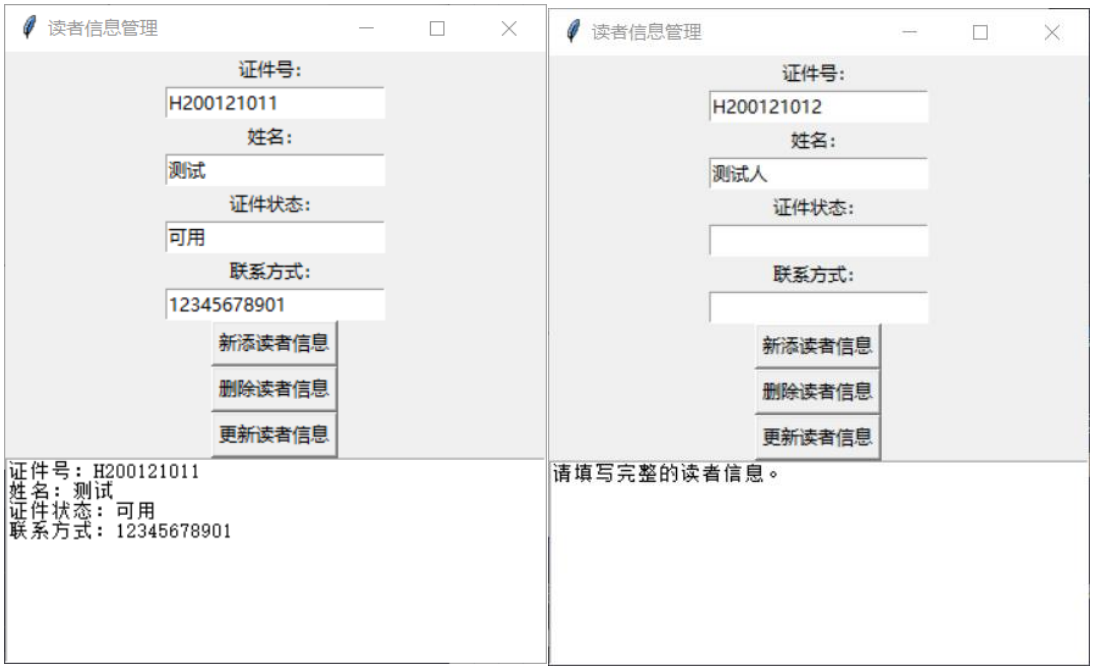
同理功能 7 对图书信息的更新，本系统要求必须有图书分类号（主码）才能更新，否则会给出提示。



在管理员界面点击“读者信息”，进入读者信息管理界面。

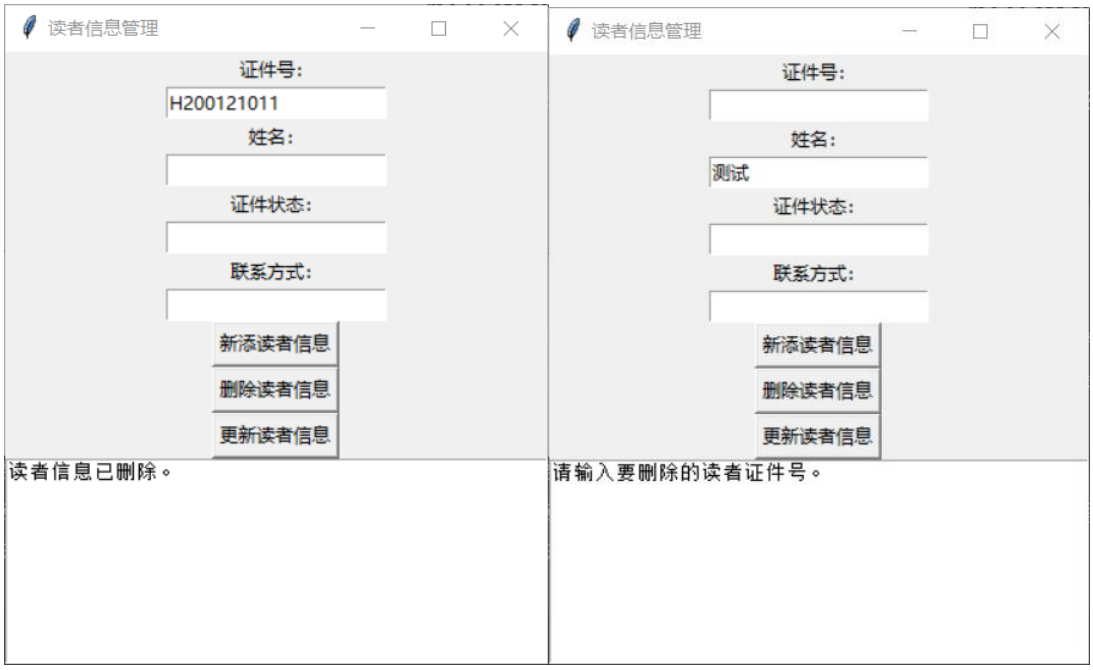
- **功能 9：添加读者信息**

同理功能 5， 本系统要求，输入“读者”表的所有属性，才能添加读者信息，否则会提示。



- **功能 10：删除读者信息**

同功能 6，本系统要求，只能根据证件号（主码）对读者借阅信息进行删除。



- **功能 11：更新读者信息**

同理功能 7 对图书信息的更新，本系统要求必须有证件号（主码）才能更新，否则会给出提示。

三、实验总结

（可以总结实验中出现的问题以及解决的思路，也可以列出没有解决的问题）

- **问题 1：**

由于没有学过 java 或者 python 关于界面的相关设计，因此起出有些无从下手，同时，也不清楚该如何将数据库的数据与界面相连接。

在查阅了相关资料后，选择用 python 的 tkinter 库构建图形界面，用 psycopg2 连接数据库。给出数据库的相关信息，端口、密码等等，得以连接，在后续很多查询等操作时，都会将语句传递给 cur=conn.cursor()的相关函数实现操作，例如 cur.execute(query)。

- 需注意，在用 pgAdmin4 或者 SQL shell 打开了该数据库后，才能连接上。

```
# 创建数据库连接
conn = psycopg2.connect(database="LibraryBorrowingSystem", user="postgres",
                        password="tara090729", host="localhost", port="5432")
cur = conn.cursor()

# 创建主窗口
window = tk.Tk()
# 运行主界面
main_interface()
window.mainloop()

# 关闭数据库连接
cur.close()
conn.close()
```

与此同时，也学习了 tkinter 库中最基本的窗口、界面组件的创建和调用。

```
# 创建管理员窗口
admin_window = tk.Toplevel(window)

# 创建界面组件
reader_id_label = tk.Label(admin_window, text="证件号：")
reader_id_label.pack()
reader_id_entry = tk.Entry(admin_window)
reader_id_entry.pack()

reader_id = reader_id_entry.get()
reader_name = reader_name_entry.get()
reader_status = reader_status_entry.get()
reader_contact = reader_contact_entry.get()
```

- **问题 2：**

在读者查询个人的借阅信息时，存在只记得姓名不记得证件号的情况，需要添加一层查询，帮助读者根据姓名先查到对应的证件号，然后根据证件号给出借阅信息。但这里也会出现同名读者的信息一并查出的情况。

```
# 如果只输入了姓名，则查询对应的证件号
if reader_name and not reader_id:
    reader_query = f"SELECT 证件号 FROM 读者 WHERE 姓名 LIKE '%{reader_name}%'"
    cursor.execute(reader_query)
    reader_ids = cursor.fetchall()

    borrow_info = []
    for reader_id in reader_ids:
        borrow_query = f"SELECT * FROM 借阅 WHERE 证件号 = '{reader_id[0]}'"
        cursor.execute(borrow_query)
        borrow_info.extend(cursor.fetchall())
```

对于这些非主码的增删改查操作，实际上是十分复杂的，需要讨论非常多种情况。由于课时限制以及独自一个人考虑不周的限制，无法将所有情况进行讨论，因此只选择了一项相对代表性的任务来进行实现，其他情况可以类比。

在代码的实现中，这些复杂情况都可以由多个 if 分类讨论+query 语句的修改实现。其中利用到的 sql 语言在实验 3/4 中均有体现，因此在此次实验不再作为重点展示。

- **问题 3：**

在管理员处理读者归还图书时，对罚款金的设置需要注意，添加判断，若归还日期在应还日期之前，即罚款金 fine<0，则需要置零再更新，而不能用负数，会违反“借阅”表的相关定义。

```
# 获取当前日期
return_date = datetime.now().strftime("%Y-%m-%d")
# 计算超期天数和罚款金额
due_date = datetime.strptime(borrow_info[3].strftime("%Y-%m-%d"), "%Y-%m-%d")
days_overdue = (datetime.strptime(return_date, "%Y-%m-%d") - due_date).days
fine = 0.1 * days_overdue
if fine <= 0:
    fine = 0.00

# 更新借阅信息
cur.execute("UPDATE 借阅 SET 归还日期 = %s, 罚款金 = %s WHERE 证件号= %s AND 图书编号 = %s",
            (return_date, fine, reader_id, book_id))
conn.commit()
```

- **问题 3：**

图形的界面看起来有点长，尝试过使用 label.pack(side=tk.RIGHT, padx=10, pady=10)等语句，但修改多次后效果不佳，遂放弃。