

华东师范大学软件工程学院实验报告

实验课程：数据库系统及其应用	年级：2018	实验成绩：
实验名称：期末Web项目	姓名：孙宇洪	实验日期：2020.6.10
实验编号：8	学号：10174503104	实验时间：/
指导教师：宫学庆	组号：/	

项目介绍

- 本项目采用Python中的Flask框架，实现了一个简单的笔记（Note-taking）记录Web App。
- 数据库采用的是轻量级的SQLite，方便本地部署与操作。
- 用户在App上进行注册后，可以添加、编辑、删除自己的笔记，并且为自己的笔记添加Tags。
- 用户可以通过标签或内容作为关键字查询对应笔记。

Github项目地址

<https://github.com/Yuki-Asuuna/ECNU-DB-Project>

环境要求

1. Python 3+以上的解释器
2. pip3以上版本

安装方法

- 安装相关依赖包

```
pip install -r requirements.txt
```

- 解压源文件
- cd到解压目录，输入以下命令并回车：

```
python run.py
```

- 访问127.0.0.1:5000

演示

建议在IE浏览器下运行（CHROME会出现界面变形的情况）

登陆界面：

127.0.0.1:5000

☆

< Take Notes >

注册 登陆

A Simple Note Taking App

目前总共有 5 位注册用户

© 2020 孙宇洪

[My blog](#) [My Github](#)

注册帐号：

127.0.0.1:5000/signup/

☆

< Take Notes >

注册 登陆

Username*

Email*

Password*

Confirm Password*

Signup

© 2020 孙宇洪

[My blog](#) [My Github](#)

登陆（可以使用测试账户test 密码123）：

127.0.0.1:5000/login/

☆

< Take Notes >

注册 登陆

Username*

Password*

Login

点击此创建账户 注册

或者你可以使用测试账户（用户名：test，密码：123）

© 2020 孙宇洪

[My blog](#) [My Github](#)

创建笔记：

< Take Notes >

创建新笔记

查看所有笔记

创建新标签

查看所有标签

Welcome, test ▾

Note Title:

mytest

Your Note:

Hello, world!

Preview:
Hello, world!

Add Note

© 2020 孙宇洪

My blog My Github

查询笔记

< Take Notes >

创建新笔记

查看所有笔记

创建新标签

查看所有标签

Welcome, test ▾

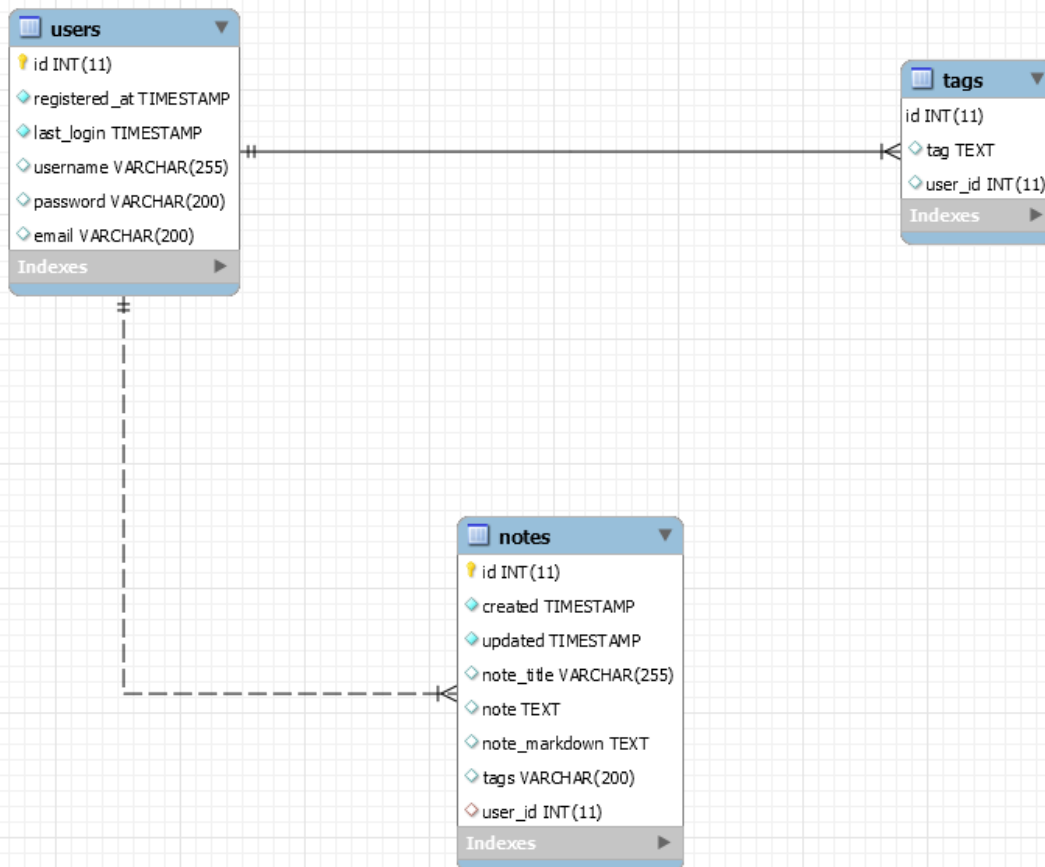
Search Note

#	标题	修改时间	标签	删除
1	mytest	Jul 23, 2020 13:45:54		

© 2020 孙宇洪

My blog My Github

模式设计（E-R图）



```

CREATE TABLE `notes` (
  `id` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
  `created` TIMESTAMP NOT NULL DEFAULT (strftime('%Y-%m-%d %H:%M:%S', 'now',
'localtime')),
  `updated` TIMESTAMP NOT NULL DEFAULT (strftime('%Y-%m-%d %H:%M:%S', 'now',
'localtime')),
  `note_title` VARCHAR(255),
  `note` TEXT,
  `note_markdown` TEXT,
  `tags` VARCHAR(200),
  `user_id` INTEGER,
  FOREIGN KEY(user_id) REFERENCES users(id)
);

CREATE TRIGGER `triggerDate` AFTER UPDATE ON `notes`
BEGIN
  update `notes` SET `updated` = (strftime('%Y-%m-%d %H:%M:%S', 'now',
'localtime')) WHERE id = NEW.id;
END;

CREATE TABLE `users` (
  `id` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
  `registered_at` TIMESTAMP NOT NULL DEFAULT (strftime('%Y-%m-%d %H:%M:%S',
'now', 'localtime')),
  `last_login` TIMESTAMP NOT NULL DEFAULT (strftime('%Y-%m-%d %H:%M:%S', 'now',
'localtime')),
  `username` VARCHAR(255),
  `password` VARCHAR(200),
  `email` VARCHAR(200)
  
```

```
);

CREATE TRIGGER `triggerUserLogin` AFTER UPDATE ON `users`
BEGIN
    update `users` SET `last_login` = (strftime('%Y-%m-%d %H:%M:%S', 'now',
'localtime')) WHERE id = NEW.id;
END;

CREATE TABLE `tags` (
  `id` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
  `tag` TEXT,
  `user_id` INTEGER,
  FOREIGN KEY(user_id) REFERENCES users(id)
);
```

表与表之间的关系相对简单，化简易得该模式满足3NF的要求，消除了大部分冗余。

数据示例：

1.user

```
users(1, '2020-07-21 21:50:20', '2020-7-22
13:26:35', 'sunyh1999', '123', 'sunyh1999@126.com')
```

用户信息:

注册于:

Jul 21,2020 21:50:20

最后登录时间:

Jul 22,2020 13:26:35

Email: 

sunyh1999@126.com


用户名:

sunyh1999

2.tags

```
tag(1, 'python', 1)
```

提醒 你可以通过点击标签来过滤所有带有标签的笔记

#	Tag	Delete
1	python	

3.notes

```
notes(1, '2020-07-21 22:05:20', '2020-07-21  
22:05:20', 'mytest', '12344444', '', 'python', 1)
```

详细信息:

添加于:

Jul 21, 2020 22:05:20

更新于:

Jul 22, 2020 13:34:19

标题:

mytest

 Edit

 Delete

12344444

相关查询/操作

- 连接数据库

```
def get_database_connection():  
    '''  
        Creates a connection between selected database  
    '''  
    import sqlite3  
    sqlite_file = 'notes.db'  
    file_exists = os.path.isfile(sqlite_file)  
    conn = sqlite3.connect(sqlite_file)  
    if not file_exists:  
        create_sqlite_tables(conn)  
    return conn
```

- 创建关系模式

```
def create_sqlite_tables(conn):  
    '''  
        Creates a sqlite table as specified in schema_sqlite.sql file  
    '''  
    cursor = conn.cursor()  
    with open('schema_sqlite.sql', 'r') as schema_file:  
        cursor.executescript(schema_file.read())  
    conn.commit()
```

- 查询用户数

```
select count(*)  
from users;
```

- 查询用户是否存在

```
SELECT * FROM users WHERE username=? AND password=?;
```

- 更新用户的最后登录时间

```
UPDATE users SET last_login=(strftime('%Y-%m-%d %H:%M:%S', 'now',  
'localtime')) WHERE id=?;
```

- 创建用户

```
INSERT INTO users (username,password,email) VALUES(?,?,?);
```

- 查询用户相关信息

```
SELECT * FROM users WHERE id=?;
```

- 查询tag相关信息

```
SELECT tag FROM tags WHERE id=?;
```

诸如此类的查询/更新操作还有很多，限于篇幅在此不再列举，详情见目录下的functionss.py。

关于优化

本app关系模式上的逻辑是比较简单的。通过观察常见查询可以发现，大多数的查询都是围绕 user_id、tag_id、note_id来展开的，而这些属性恰好是它们的主键（默认主键建立索引），因此不需要对其它属性增加额外索引。

关于前端/功能

前端的开发基于Bootstrap框架，用了一些基本的组件。由于本人美术水平实在是不太行，所以界面相对也比较简陋。

前端与后端的交互通过flask框架中的route装饰器实现。表单通过flask-wtf实现。登陆状态通过flask-login插件实现。

template列表：

文件名	描述
add_note.html	添加笔记
profile.html	个人信息页面
profile_setting.html	个人信息修改
signup.html	注册
view_note.html	查看笔记
view_tag.html	查看tag列表
add_tag.html	添加tag
change_email.html	修改email
change_password.html	修改密码
edit_note.html	编辑笔记
edit_tag.html	编辑tag
homepage.html	回到主页
index.html	主页
login.html	登陆

参考资料

[1].pallets/flask: The Python micro framework for building web applications..<https://github.com/pallets/flask>

[2].使用 WTForms 进行表单验证 — Flask 中文文档 (1.1.1) . <https://dormousehole.readthedocs.io/en/latest/patterns/wtforms.html>

[3]render_template渲染模板及jinja2 - 后端开发——Flask初体验 - SegmentFault 思否.<https://segmentfault.com/a/1190000012817254>

[4].《Flask+Web开发实战：入门、进阶与原理解析（李辉著+）》

[5].《Flask Web开发：基于Python的Web应用开发实战》

[6]. Bootstrap.<https://getbootstrap.com/2.3.2/>