课程设计名称[[1]](#footnote-0)🞼

参与同学（冯古豪 2000013175 吴清月 2000013057）

(北京大学 信息科学技术学院, 北京 100871);

E-mail: [fenguhao@stu.pku.edu.cn](mailto:fenguhao@stu.pku.edu.cn) , 2000013057@stu.pku.edu.cn

摘要: 本次数据库大作业实现了一个简单的论坛系统。这个论坛系统支持用户注册、登录、发帖、回帖和关注等许多功能，并采用B/S结构和网页前端，用户能方便地执行各项操作。另外，我们还充分使用了数据库的一些高级功能如存储过程、视图等来辅助系统的实现。

# 引言

在这次数据库大作业中，我们实现了一个论坛系统。论坛系统为用户提供了一个开放的交流平台，是在现实生活中有着很重要的应用场景。在论坛系统中，需要设计合适的表和关系来存储用户信息、帖子、评论等数据，这涉及到数据库设计中的实体关系模型、关系规范化等理论知识的应用；用户需要能够浏览帖子、发表评论等，这就需要使用SQL，编写有效的查询语句来查询、更新数据库。通过设计一个论坛系统，我们能够将课程中学到的数据库理论知识应用到实际项目中，提升我们设计复杂数据库系统的能力。

我们实现的论坛系统，支持用户的注册、登录、发帖、回帖和关注他人等一系列操作。我们采用B/S结构，使用网页作为前端，让用户能够很方便执行各项操作。在实现的这个系统中，我们锻炼了自己设计复杂系统的能力，并且对之前数据库学习到的理论知识有了更加深入的理解。

本文的组织：在第二部分中，我们介绍了本次大作业所使用的工具，包括数据库和前后端框架；第三部分描述了论坛系统框架和功能；第四部分给出相关的关系数据库模式和相应的操作；第五部分总结了全文，并且描述了完成本课程设计的收获。

# 相关工作

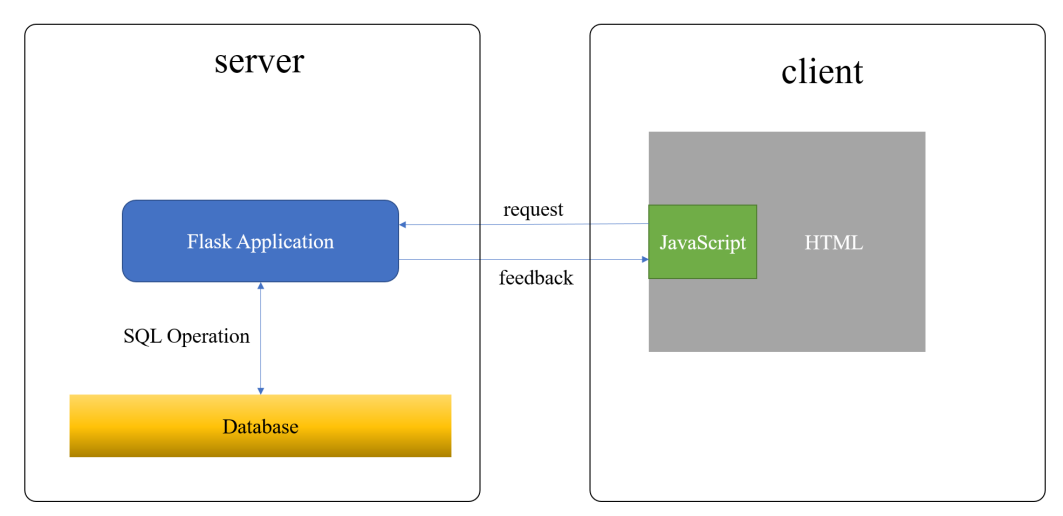
在构建前端部分，我们选择了使用Python的Flask框架作为服务器的搭建工具，并使用Bootstrap框架来构建前端页面，此外，我们还使用了JavaScript来实现一些逻辑操作，以增强前端的交互性和动态性。。Flask框架为我们提供了一个快速而灵活的开发环境，使得我们可以轻松地处理服务器端的逻辑。同时，Bootstrap框架为我们提供了丰富的前端组件和样式，使得我们可以轻松地构建出现代化和响应式的用户界面。

为了实现与数据库的交互，我们选择了SQL Server作为我们的数据库系统，并使用Python的pyodbc库来实现与数据库的连接和交互。在处理复杂的字符串操作方面，我们利用了Python的re模块。re模块为我们提供了正则表达式的功能，使得我们可以方便地进行复杂的字符串匹配、查找和替换操作。这对于处理文本数据和提取特定模式的信息非常有用。此外，在数据库设计方面，我们使用了PowerDesigner软件帮助我们进行数据库的建模、设计和管理。通过PowerDesigner，我们能够更好地管理数据库设计流程，确保数据库结构的合理性和一致性。

# 课程设计的系统框架

在这一部分中，我们将详细介绍我们实现的整个系统和系统的功能。

我们的系统包括4个部分：网页代码（均在/server/templates目录下），服务器程序（server/hello.py），数据库接口程序（SQL/utils.py）和数据库（SQL/目录下的其它内容）。在整体接口上，我们使用Python的Flask库来编写web server，在客户端通过html文件中的javascript函数来与服务端进行交互，在服务端通过python代码与数据库进行交互。如下图所示。



## 网页设计

在网页设计上，我们为应用设计了6个不同的网页：主页（main.html），游客页面（visitor.html），个人信息（user.html），关注列表（users.html），最近帖子（posts.html），帖子详情（post.html）。它们使用同一个模板（base.html），在此之上进行每种网页各自的设计。当用户访问特定网址的时候，服务器会将用户定向到对应的页面。特别的，如果用户未登录，除主页外所有的网址都会重定向到游客界面（visitor.html），提醒用户登录。

## 网页与服务器的交互

在网页与服务器的交互上，除基础的网址访问外，我们还设计了7个不同的请求：登录（login），注册（signup），登出（logout），发帖（postnew），回复（reply），关注（follow）和取消关注（disfollow）。它们都是通过javascript中的触发函数来实现。

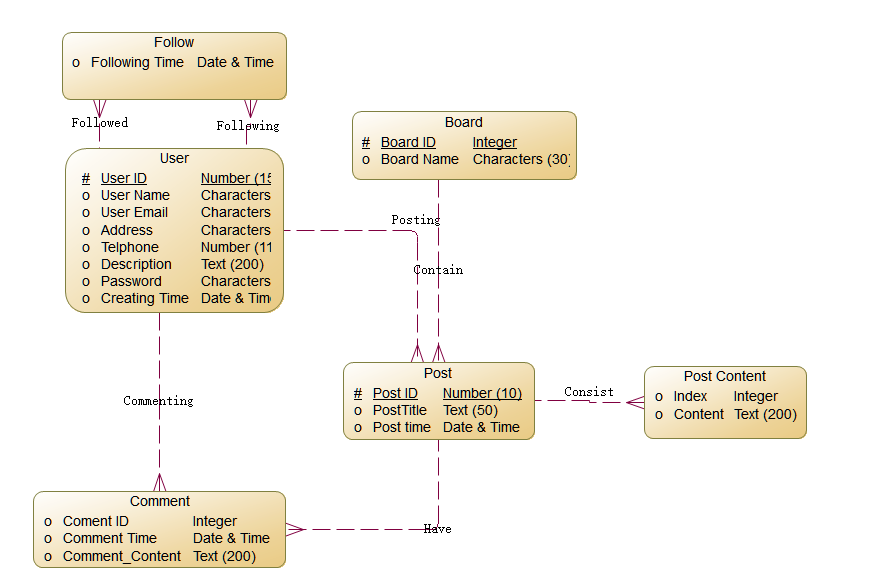
为了在一定程度上减小网页延迟，我们将尽可能多的操作放在客户端进行，如注册时检查确认密码和原密码是否相同，回帖时检查内容是否为空等等。只有当需要与数据库交互时，才向服务器发送请求。此外，若服务器由于某些错误产生未知异常，javascript函数也对此进行了处理。

## 服务器与数据库的交互

在服务器代码和数据库的交互上，我们使用python中的pyodbc库，将对数据库的访问封装到python代码（/SQL/utils.py）中，如查找用户信息，帖子信息，注册新用户，回帖等等，方便服务器代码进行调用。

# 课程设计对应的关系数据库模式

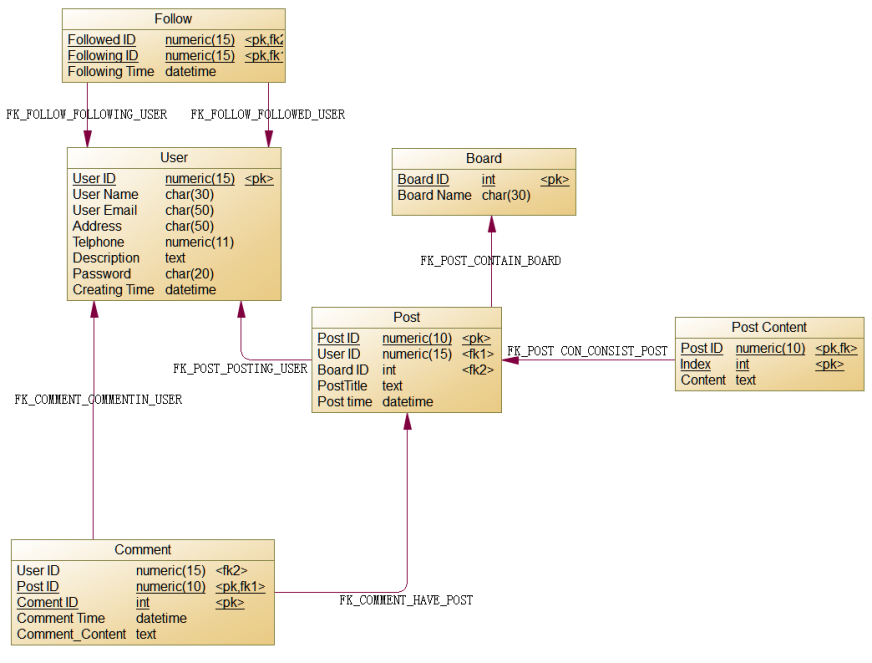
## ER图

我们使用PowerDesigner软件管理数据库设计流程。首先，我们使用PowerDesigner设计E-R图，如下图所示。

这个E-R图描述了一个简单的论坛管理系统。在该系统中，存在六个主要实体：用户（User）、版面（Board）、帖子（Post）、帖子内容（Post Content）、评论（Comment）以及关注关系（Follow）。用户实体包含了用户的基本信息如地址、电话和创建时间等等，同时还存储了用户用来登陆论坛系统的用户名（User Name）和密码（Password）等信息，每一个用户由User ID唯一标识。版面（Board）实体包含了主键（Board ID）和版面名。帖子（Post）实体包含了主键（Post ID）和帖子的标题以及发帖的时间。为了处理有些帖子长度过大的情况，我们没有简单的将帖子的内容作为一个属性存在帖子实体中。我们如果采用设置上限的方法处理帖子长度过大的情况，那么就会出现如果上限太大会极度浪费空间，上限太小又不太合理的矛盾。因此，我们另外建立了一张表来存储帖子的内容，我们将帖子的内容切分成若干个小段，每一段至多200字，然后按照顺序给每一段一个索引，表中的每一行只要存储索引和对应段的内容即可。因此，帖子内容（Post Content）实体包含索引和对应段的内容两个属性。评论（Comment）实体包含评论号（Comment ID），评论时间和评论内容三个属性。其中，评论号标识这个评论是在某一个帖子下的第个回复。关注关系（Follow）实体仅仅包含关注时间一个属性。

在这张E-R图中，我们还标识了不同实体之间的关系。一个用户可以发布多个帖子和多个评论，因此用户和帖子之间，用户和评论之间都存在一对多的关系。一个版面可以包含多个帖子，因此用户和帖子之间存在一对多的关系。一个帖子的内容被截成了很多段，同时也会有多个评论，因此帖子和内容之间，帖子和评论之间都存在一对多的关系。一个用户可以在不同时间关注他人，也可以在不同时间被他人关注，因此用户和关注关系之间存在着两个一对多的关系。

## 关系数据库模式描述

基于上述的E-R图，我们使用PowerDesigner软件自动生成了物理模式图，并且，我们对这个物理模式图稍作修改就得到了我们最终的物理模式图，如图所示，图中包含了每张表的属性，并标识了主键和外键。我们设计的物理模式图达到了4NF的范式要求，不存在非主属性之间的多值依赖。

在论坛系统实现中，我们需要根据用户名去查询用户的信息，因此，为了加快查询速度，我们在表User中单独为用户名（User Name）创建了非聚簇索引。为了保障信息的安全，我们对SQL注入攻击进行了防范, 对于用户名和密码，我们限定使用字母和数字，我们python的re模块进行检测。同时，我们限制了用户名和密码的最大长度。为了方便查询用户关注的其它用户的信息，创建了一个视图来简化查询，这个视图follow\_view是对User表和Follow表进行条件连接得到的，用来查询被关注者的信息。同时，为了保障安全，我们构建了一个存储过程get\_password来访问密码，我们保证前端只能够通过这个存储过程来访问密码。

## 关系数据库模式之上的操作

我们使用pyodbc来实现与数据库的交互，针对前端每个具体的功能分别构建了具体的函数，在服务器中，我们直接调用这些函数读取和修改数据。

# 总结

在这个大作业中，我们实现了一个简单的论坛系统。我们充分运用了课程中学到的数据库理论知识，来设计整体的数据库系统，同时，尝试着使用了一些辅助软件来管理整个设计的流程。在整个项目中，我们对于课程中学习到的数据库的基础理论知识也有了更深的理解，构建复杂的系统极大锻炼了我们的团队协作能力，也为我们之后的学习工作积累了宝贵的经验。同时，在这个项目中，我充分意识到了，当面对一个复杂系统，仔细分析和理解用户的需求，设计合理的结构和关系的重要性。

References:

1. Ashish Kumar Gupta, Dan Suciu: Stream Processing of XPath Queries with Predicates. In: Alon Y. Halevy, Zachary G. Ives, AnHai Doan., eds Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, USA, June 9-12, ACM, 2003, 419-430

1. [↑](#footnote-ref-0)