# 哈尔滨工业大学

## <<数据库系统>>

## 实验报告二

(2023 年度春季学期)

| | |
|---|---|
| 姓名: | 符兴 |
| 学号: | 7203610316 |
| 学院: | 计算学部 |
| 教师: | 李东博 |

# 实验二

## 一、实验目的

在熟练掌握 MySQL 基本命令、SQL 语言以及用 C 语言编写 MySQL 操作程序的基础上，学习简单数据库系统的设计方法，包括数据库概要设计、逻辑设计。

## 二、实验环境

Windows 11 操作系统、MySQL8.0.32、PhpMyAdmin、Python3.10、PyQt5、QtCreator 等

## 三、实验过程及结果

建立一个博客网站管理的数据库系统，本数据库实现的基本功能有：插入新用户、新文章；删除用户；删除文章；查询用户评论；查询板块管理信息；查询某篇文章的情绪分布；查询某用户评论的文章等，并且完成所有的加分项。

为构建概念数据库，下面分析博客网站管理数据库实体与联系并画出 ER 图。

**1. 实体**，其中加粗的是主码

(1) 板块 plate：**板块编号**、板块名称（唯一性约束）、板块简介

(2) 板块负责人 plate_per：**身份证号**、姓名、联系方式

(3) 创作组 creation：**板块编号**、创作组编号、创作组名称（唯一性约束）、创作组简介

(4) 创作组负责人 creation_per：**身份证号**、姓名、联系方式

(5) 话题 topic：**话题编号**、话题名称（唯一性约束）、话题简介、创建时间

(6) 文章 doc：**文章编号**、文章名称、文章存储地址

(7) 用户 user：**用户编号**、用户昵称（唯一性约束）、联系方式

(8) 活动 acti：**活动编号**、活动名称（唯一性约束）、活动内容、参与方式

**2. 关系**，其中加粗的是主码

(1) 板块负责人和板块构成“管理板块”(ma_plate)联系，属于 1:n 联系

**身份证 ID**，**板块编号**，开始时间，结束时间

(2) 创作组负责人和创作组构成“管理创作组”(ma_creation)联系，属于 1:1 联系

**身份证 ID**，**板块编号**，**创作组编号**，开始时间，结束时间

(3) 用户和文章构成“评论文章”(com_doc)联系，属于 m:n 联系

**用户编号**，**文章编号**，评论时间，情绪

(4) 用户和活动构成“参与活动”(join_acti)联系，属于 m:n 联系

**用户编号**，**活动编号**，参与时间

(5) 创作组和话题构成“负责话题”(le_topic)联系，属于 m:n 联系

**板块编号**，**创作组编号**，**话题编号**，开始时间，结束时间

(6) 创作组和文章构成“发布文章”(pub_doc)联系，属于 1:n 联系

**板块编号**，**创作组编号**、**文章编号**、发布时间

(7) 文章和话题组成“参与话题”(join_topic)联系，属于 m:n 联系

**文章编号**、**话题编号**、发布时间

**3. ER 图**

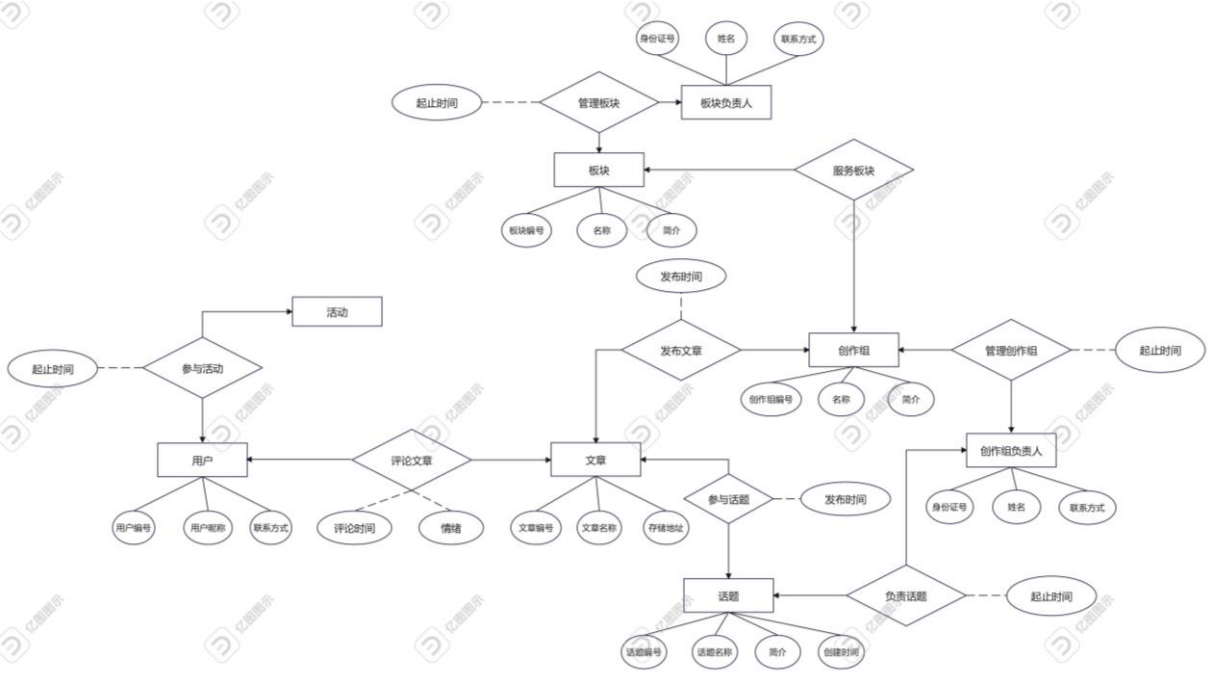


图 1 博客网络数据库 ER 图

**4. 关系的完整性约束**

(1) 主键约束：主码不允许重复

(2) 外键约束：满足参照完整性

(3) 空值约束：本关系数据库系统所有属性值非空

(4) 唯一性约束：本关系数据库中部分属性要求是唯一的，比如板块名称、创作组名称、话题名称、用户名称以及活动名称等。

**5. 插入操作**

**(1) 插入新的博客网站用户**

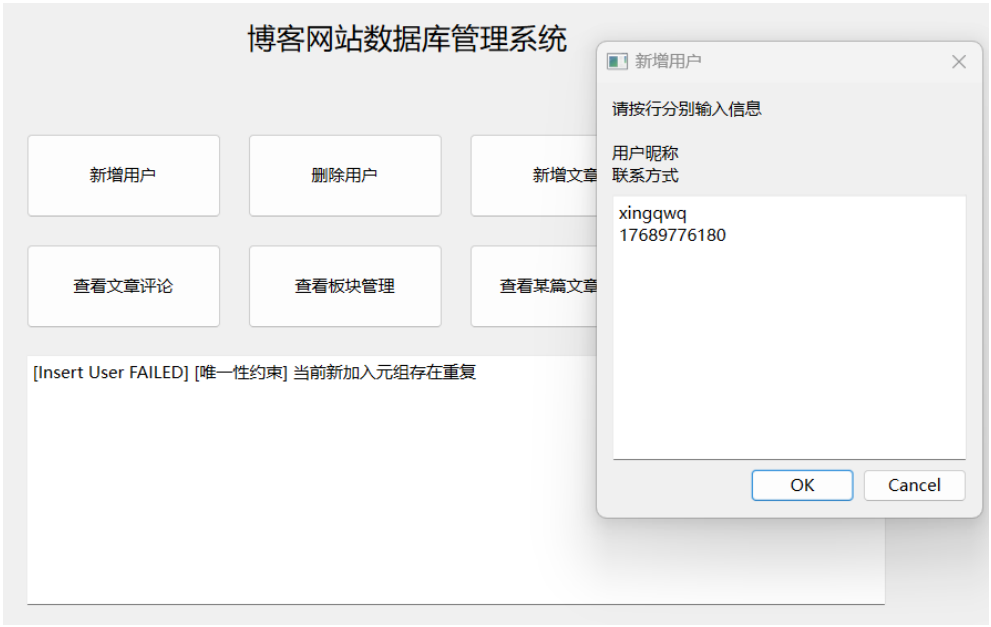


图 2 插入博客新用户

插入博客新用户时，由于用户编号使用自增主键，因此只需要给定用户昵称和联系方式；因为用户昵称具有唯一性约束，在插入具有相同用户昵称的账号时会给出错误提示。

**(2) 插入新的博客文章**

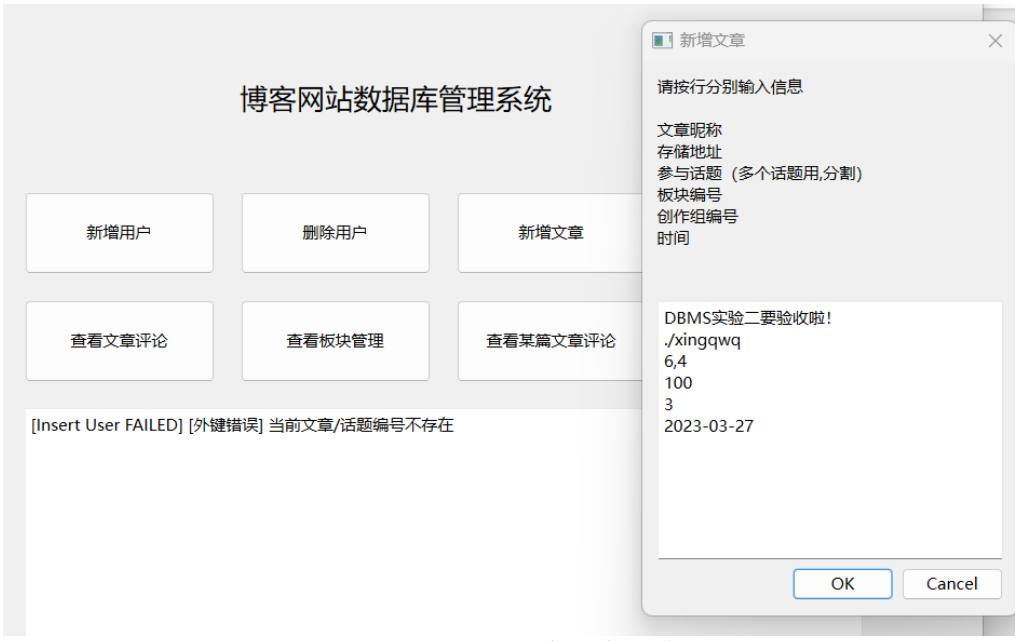


图 3 插入新的博客文章失败

插入博客文章大致需要三个步骤，新增文章实体记录、新增文章创作组联系记录、新增文章话题联系记录；因此在这里做了事务管理，将上述的三个操作看作一个原子操作，即三个步骤全部完成或三个步骤全部不完成，在任何一个步骤出错时都对数据库进行 rollback() 处理。

如图 3 所示，在插入文章时，如果选择了一个不存在的板块编号，程序会给出外键约束错误，并且对数据库进行 rollback()，删除已经添加的文章实体元组。


图 4 插入新的博客文章成功

```
def insertPage(self, name, doc, topicList, plateID, creationID, date):
    flag = 0
    try:
        # 新建文章
        sql = "INSERT INTO `doc` (`名称`, `存储地址`) VALUES ('{}', '{}');".format(name, doc)
        self.printSQL(sql)
        cur = self.dbms.newCur()
        cur.execute(sql)
        cur.execute("SELECT LAST_INSERT_ID();")
        docID = cur.fetchone()[0]
        # 添加话题
        for i in topicList:
            sql = "INSERT INTO `join_topic` (`文章编号t`, `话题编号t`, `发布时间`) VALUES ('{}', '{}','{}')".format(docID, i
            self.printSQL(sql)
            cur.execute(sql)
        # 创作组发布文章联系
        sql = "INSERT INTO `pub_doc` (`板块编号t`, `创作组编号t`, `文章编号t`, `发表时间`) VALUES ('{}', '{}', '{}', '{}');".
        cur.execute(sql)
        self.printSQL(sql)
        self.dbms.db.commit()
        print("执行成功\n")
        flag = 1
    except Exception as e:
        print("执行失败")
        if "a foreign key constraint fails" in repr(e):
            msg = "[外键错误] 当前文章/话题编号不存在"
            print("[外键错误] 当前文章/话题编号不存在")
        if "Duplicate entry" in repr(e):
            msg = "[唯一性约束] 当前新加入元组存在重复"
            print("[唯一性约束] 当前新加入元组存在重复")
        self.dbms.db.rollback()
    finally:
        cur.close()

    if flag:
        return True,"SUCC"
    else:
        return False,msg
```

图 5 插入新文章的事务管理

**6. 删除操作**

| Trigger | Event | Table | Statement | Timing | Created | sql_mode | Definer | character_set_client | collation_connection | Database Collation |
|---|---|---|---|---|---|---|---|---|---|---|
| delete_page | DELETE | doc | begin<br>DELETE FROM pub_doc WHERE 文章编号t ... | BEFORE | 2023-03-31 11:03:06.95 | | root@localhost | utf8mb4 | utf8mb4_unicode_ci | utf8mb4_0900_ai_ci |

图 6 删除文章触发器设置

由于外键约束，如果仅对文章实体元组进行删除，MySQL 会给出外键约束错误，无法删除；因此在这里使用删除触发器，在删除文章元组之前，将文章相关的关系表中的相关记录进行删除。

**7. 连接查询**


图 7 查询用户评论

```
try:
    cur = self.dbms.newCur()
    sql = "select * from user_com"
    self.printSQL(sql)
    cur.execute(sql)
    msg = ("用户评论列表",cur.fetchall())
    print("执行成功\n")
    flag = 1
except Exception as e:
    print("执行失败",e)
    msg = repr(e)
finally:
    cur.close()
```

图 8 查询用户评论的 SQL 语句

这里连接查询是 user-com_doc-doc 三个表的连接查询，得到每个用户所评论文章的文章名、时间以及情绪。由于已经提前对该查询建立视图，所以直接获取视图数据即可。

**8. 嵌套查询**


图 9 查询编号为 12 的用户“HATE”的文章列表

```
try:
    cur = self.dbms.newCur()
    sql = "select in_user.文章名, in_user.评论时间 from (select * from user_com where 情绪 = '{}') \
        as in_user where 用户编号 = '{}'".format(status,userID)
    self.printSQL(sql)
    cur.execute(sql)
    msg = ("用户编号为 {} {}的文章列表".format(userID, status),cur.fetchall())
    print("执行成功\n")
    flag = 1
except Exception as e:
    print("执行失败",e)
    msg = repr(e)
finally:
    cur.close()
```

图 10 查询某用户平衡的 SQL 语句

这里的嵌套查询是查询某位用户某种评论情绪的文章列表。如上图所示，现在 com_doc 中查询用户编号为 12 的评论，然后在其中查询出评论情绪为“HATE”的所有文章，并给出相应的文章名称以及评论时间。

**9. 分组查询**


图 11 查询文章编号为 7 的用户评论情绪分布

```
try:
    cur = self.dbms.newCur()
    sql = "select in_user.文章名, in_user.情绪, count(*) \
        from (select * from user_com where 文章编号 = '{}') \
        as in_user group by in_user.情绪".format(docID)
    self.printSQL(sql)
    cur.execute(sql)
    msg = ("文章编号为 {} 的情绪分布".format(docID),cur.fetchall())
    print("执行成功\n")
    flag = 1
except Exception as e:
    print("执行失败",e)
    msg = repr(e)
finally:
    cur.close()
```

图 12 查询某篇文章情绪分布的 SQL 语句

这里的分组查询是指查询某篇文章，并按照评论情绪进行分组，统计这篇文章的情绪分布情况。

**10. 为常见查询构建视图**


图 13 为常见查询构建视图

如图 12 所示，共为两种常见查询构建了视图。一种是查询板块管理信息，另一种是查询用户评论文章信息。

## 四、实验心得

1. 通过本次实验，了解了构建一个关系数据库的工作流程，包括构建实体，设计关系表以及关系的完整性约束等。

2. 通过本次实验，掌握使用 pymysql 对数据库进行操作，并使用异常捕获进行事务管理，以及使用 PyQt 对界面进行设计。

3. 通过本次实验，进一步了解了 MySQL 的使用，如触发器、创建视图等操作。