

# CS307 Project 1

## Presentation

王奕童 11910104

张彤 11911831

# Content

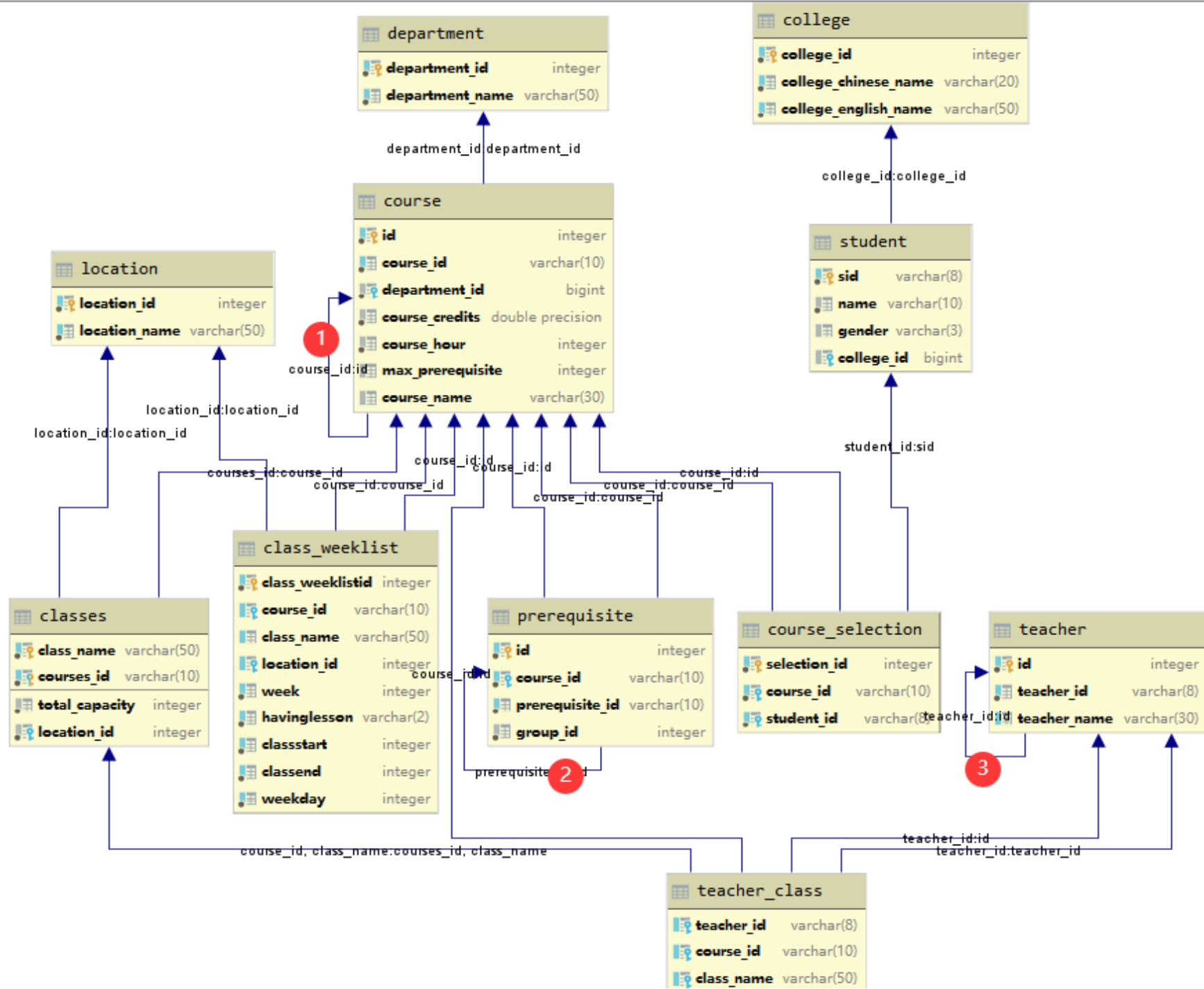
- Part0. 小组分工与贡献.....By Tong ZHANG
- Part1. 建表设计.....By Tong ZHANG
- Part2. 数据导入.....By Yitong WANG
- Part3. 文件与数据库.....By Tong ZHANG and Yitong WANG
- Part4. 总结.....By Yitong WANG

# Part0. 小组分工与贡献





- 小组成员数：2

姓名	SID	专业	贡献比
王奕童	11910104	计算机科学与技术	50%
张彤	11911831	计算机科学与技术	50%

# 建表设计



# Part 2.数据导入-脚本设计

 Client.java	2021/4/2 18:01	JAVA 文件	1 KB
 ComparingDataManipulation.java	2021/4/11 10:20	JAVA 文件	25 KB
 CSVReader.java	2021/4/10 12:18	JAVA 文件	8 KB
 DatabaseManipulation.java	2021/4/10 10:31	JAVA 文件	19 KB
 DataFactory.java	2021/4/5 10:34	JAVA 文件	1 KB
 DataManipulation.java	2021/4/10 10:31	JAVA 文件	1 KB
 Main.java	2021/4/9 20:44	JAVA 文件	11 KB
 BuildTables.sql	2021/4/6 22:00	SQL 文件	3 KB
 BuildTest.sql	2021/4/11 9:30	SQL 文件	4 KB
 Task3.sql	2021/4/9 21:47	SQL 文件	1 KB
 Task3_1.sql	2021/4/9 21:41	SQL 文件	1 KB
 TestConsole.sql	2021/4/10 12:15	SQL 文件	4 KB

- 代码可见于上传的.java文件和.sql文件

- 代码亦可见于Github仓库：

<https://github.com/YeeTone/CS307Project1-2021Spring>

- 导入json核心文件：

Main.java , DatabaseManipulation.java

- 导入sql核心文件：

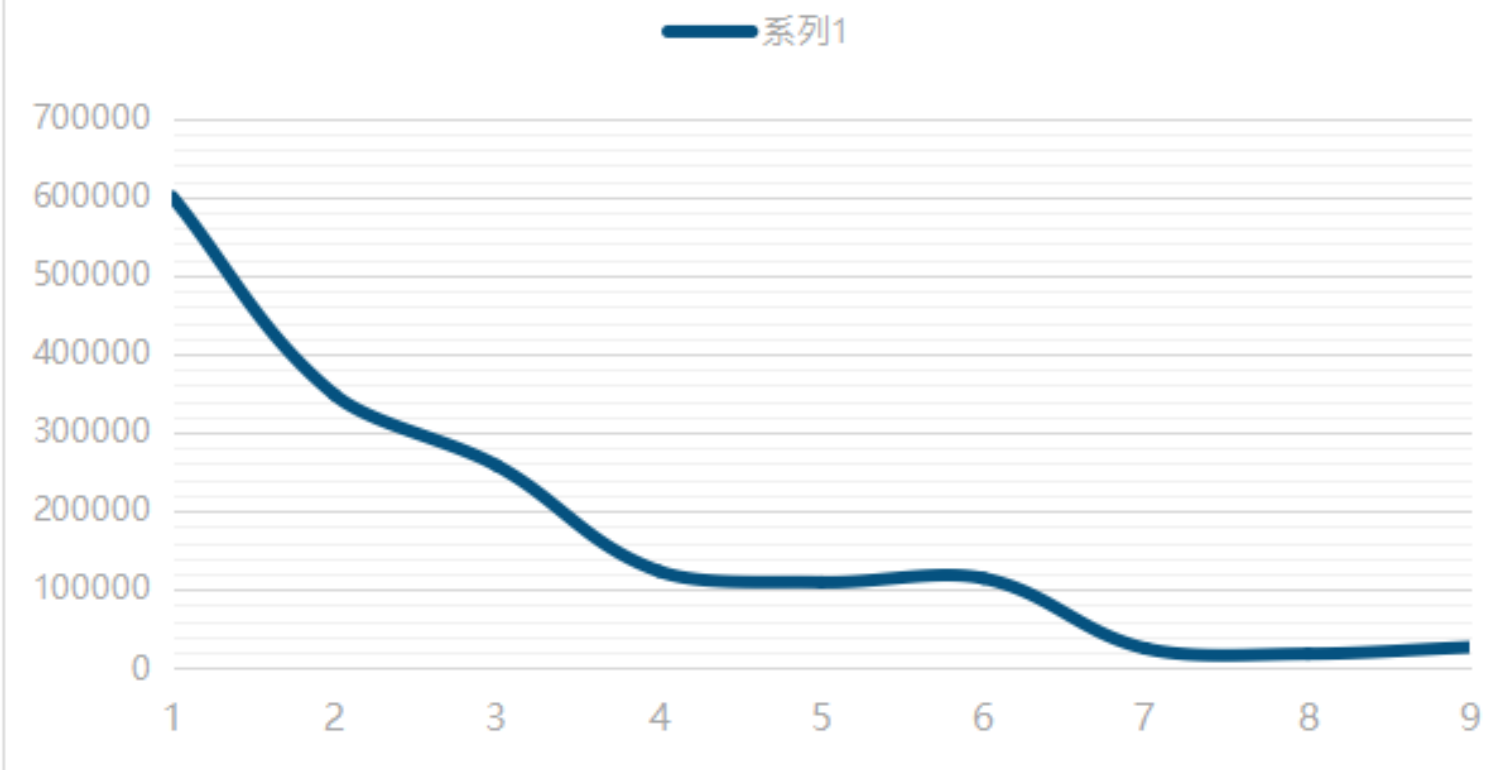
CSVReader.java

- 比较测试核心文件：

ComparingDataManipulation.java

# Part 2.数据导入-提升效率

优化次数-100W数据导入时间折线图



- 操作类型：插入数据
- 数据规模：100W
- 数据类型：学生信息
- 连接Host：localhost
- 最初版本运行时长：  
604056ms $\approx$ 10mins
- 最终版本运行时长：12066ms $\approx$ 0.2mins
- 效率提升百分比：

4873%

## Part 2.数据导入-提升效率

优化次数	优化策略	运行时间	效率提升百分比
0	无	604056ms	0%
1	哈希表避免重复创建对象	347950ms	+70%
2	使用pstm	289657ms	+17%
3	pstm循环外声明定义	257742ms	+12%
4	合并insert，批量执行	123148	<b>+110%</b>
5	移除索引约束	108991	+12%
6	增加college表索引	114056	-4%
7	预处理子查询	24737	<b>+340%</b>
8	笔记本电脑外部连接电源	15917	+55%
9	开启多线程	12066	+31%

## Part 2. 数据导入-效果验证

数据内容	Java读取结果	SQL查询结果
班级课程数量	597	597
上课周次信息	15504	15504
学生数量	3999920	3999920
学生选课信息	14000158	14000158

- 选取了部分数据进行展示比较
- 基于CSVReader.java实现测试
- 可以验证数据插入符合.json和.csv文件的内容要求
- 数据实现100%导入数据库中



## Task3 文件与数据库之用户管理权限

- 默认声明
- 创建普通用户并为其开启权限

```
1  -- 创建一个普通用户
2  create user name_login password 'xxxxxx';
3  -- 将college的owner设置为用户name_login
4  alter table college owner to name_login;
5  -- 给name_login赋予在college表上的update权限
6  grant update on college to name_login;
7  -- 给name_login撤销在college表上的update权限
8  revoke update on college from name_login;
```

## 撤销普通用户权限

```
79 revoke update on college from name_login;  
80 begin ;  
81 select * from college;  
82 ❸ update college set college_id = 23 where college_id = 10;  
83 select * from college;  
84 rollback ;  
85 select * from college;  
86
```

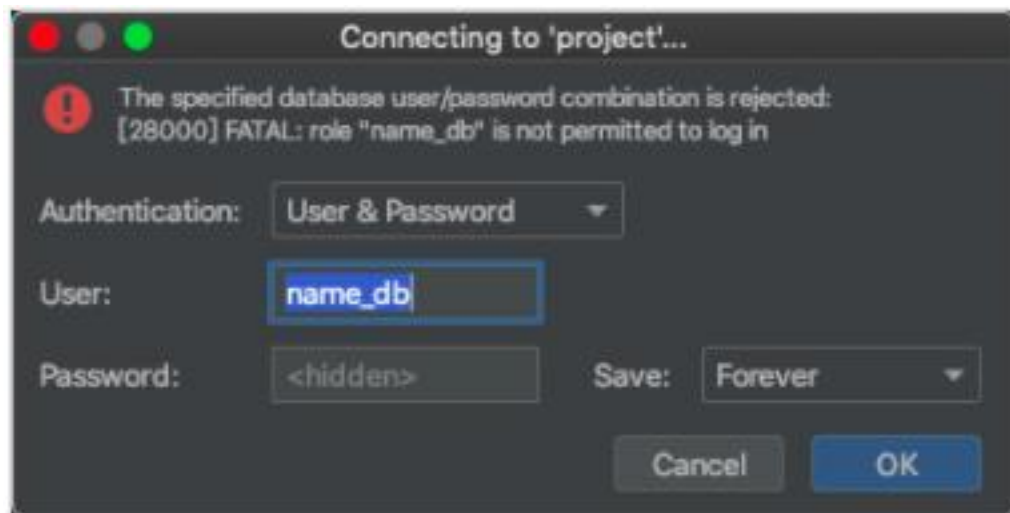
[42501] ERROR: permission denied for table college

以superuser为用户对数据库进行相关操作

数据添加 数据查询 数据更改 数据删除

# 创建用户属性

```
1  -- 创建一个可以创建数据库的角色
2  create role name_db createdb password 'xxxxxx';
3  -- 创建一个可以创建角色的角色
4  create role name_role createrole password 'xxxxxx';
```

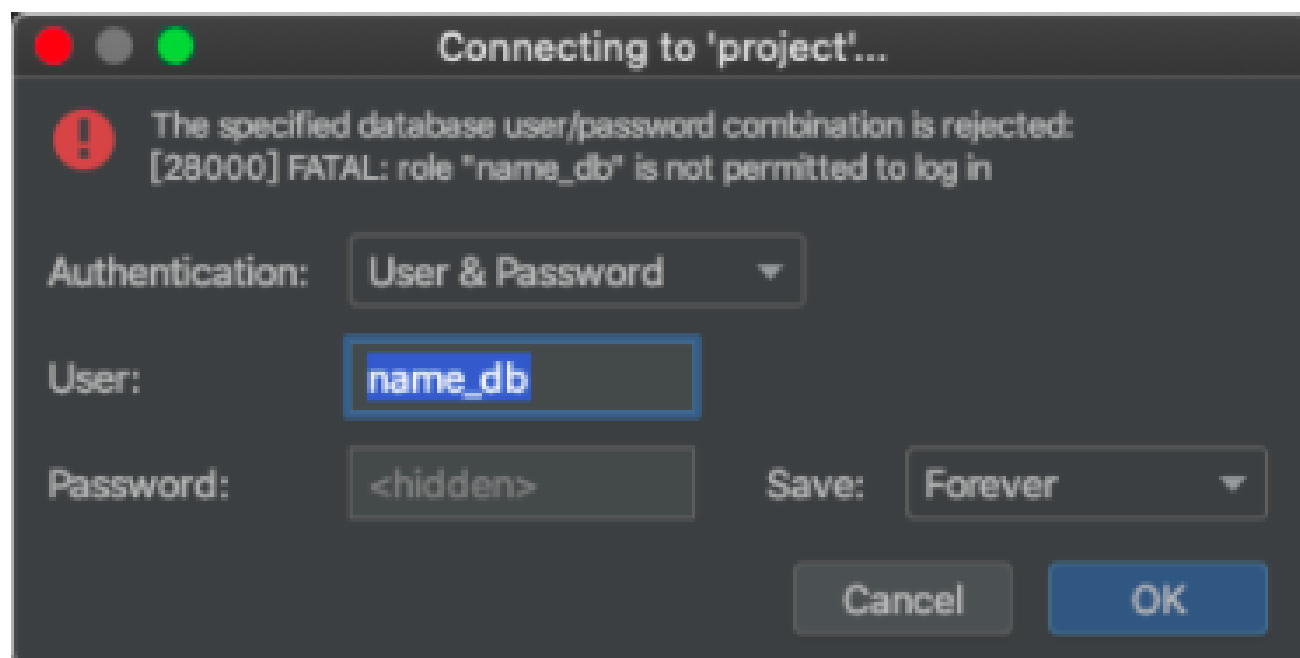


```
1  -- 创建一个可以创建数据库的角色
2  create role name_db createdb login password 'xxxxxx';
3  -- 创建一个可以创建角色的角色
4  create role name_role login createrole password 'xxxxxx';
```

## 修改用户权限

```
-- 修改用户权限  
alter role name_db with nologin ;
```

如图所示在改变其可登入权限之后，该用户无法登陆数据库



# 设置访问权限

```
1  grant select on all tables in schema public to test1
```

```
✓ grant select on all tables in schema public to test1;  
✓ select * from college;  
✓ select * from student;  
✓ select * from course_selection;  
✓ select * from classes;  
✓ select * from teacher;  
✓ select * from course;  
✓ select * from location;  
✓ select * from department;  
✓ select * from prerequisite;
```

## 用户组

```
135      -- 用户组
136  ✓    create role temporary_users;
137  ✓    grant temporary_users to test1;
138  ✓    grant temporary_users to name_login;
139  ✓    alter role test1 inherit;
```

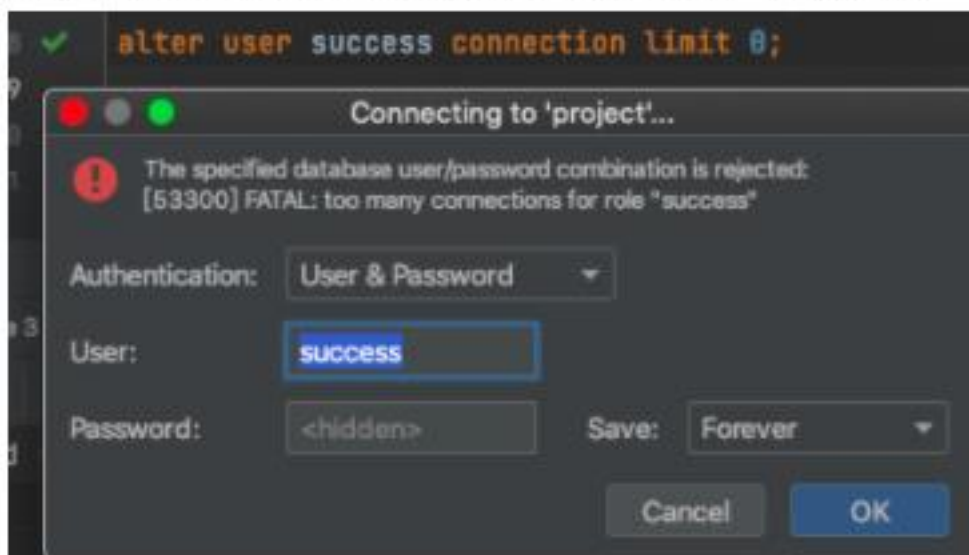
## 修改用户名



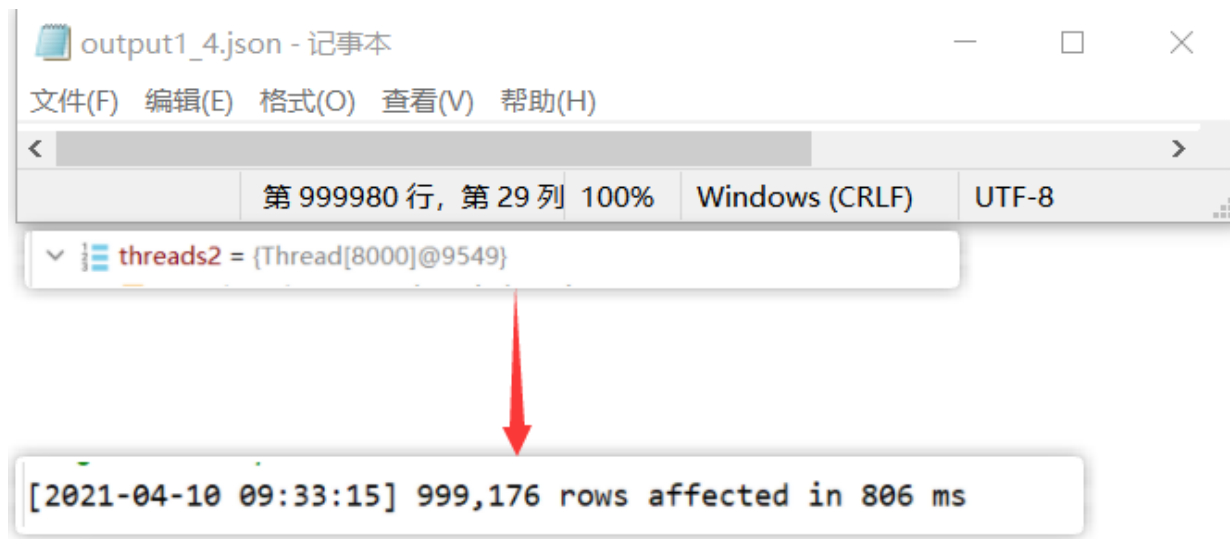
## 设置用户连接数



其中如果limit设置为0，则表示不允许登陆，如图所示：



# Part 3.实验设计-多线程高并发



- 同时开启8000个线程
- 数据插入时发生丢失

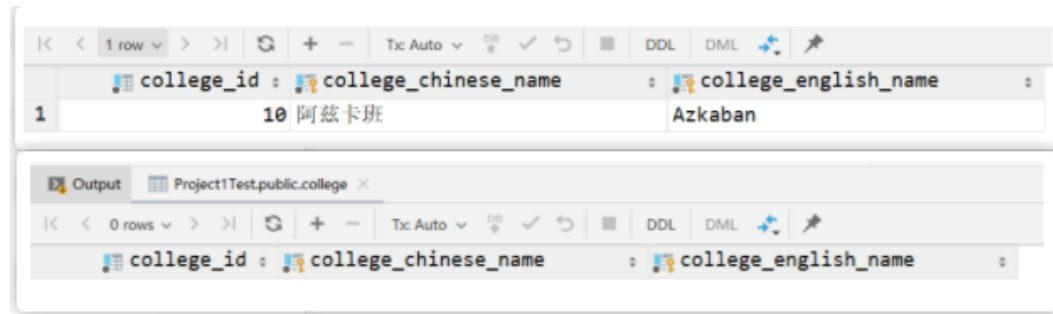
解决方案：

- 线程锁
- 线程安全容器



# Part 3.实验设计-事务管理

- 实验方法：多终端测试
- 实验现象：幻读，不可重复读



解决方案：

- 1-设置隔离级别为Serializable ==> 易死锁，效率低，不推荐

- 2-多版本并发控制 ==> 推荐

- 增：插入当前版本号对应的元素
- 删：将当前版本号对应元素标记为删除
- 改：将旧数据标记为删除，然后插入新数据
- 查：查询条件==>

create\_version <= current\_version <

5s内相同语句得到的不同的查询结果↑

# Part 3.实验设计-数据库索引

操作类型	索引前	索引后
增	76	623
删	524	625
改	2996	4350
查	1505	31

- 索引优势：

- 1.查询时间**大大缩短**

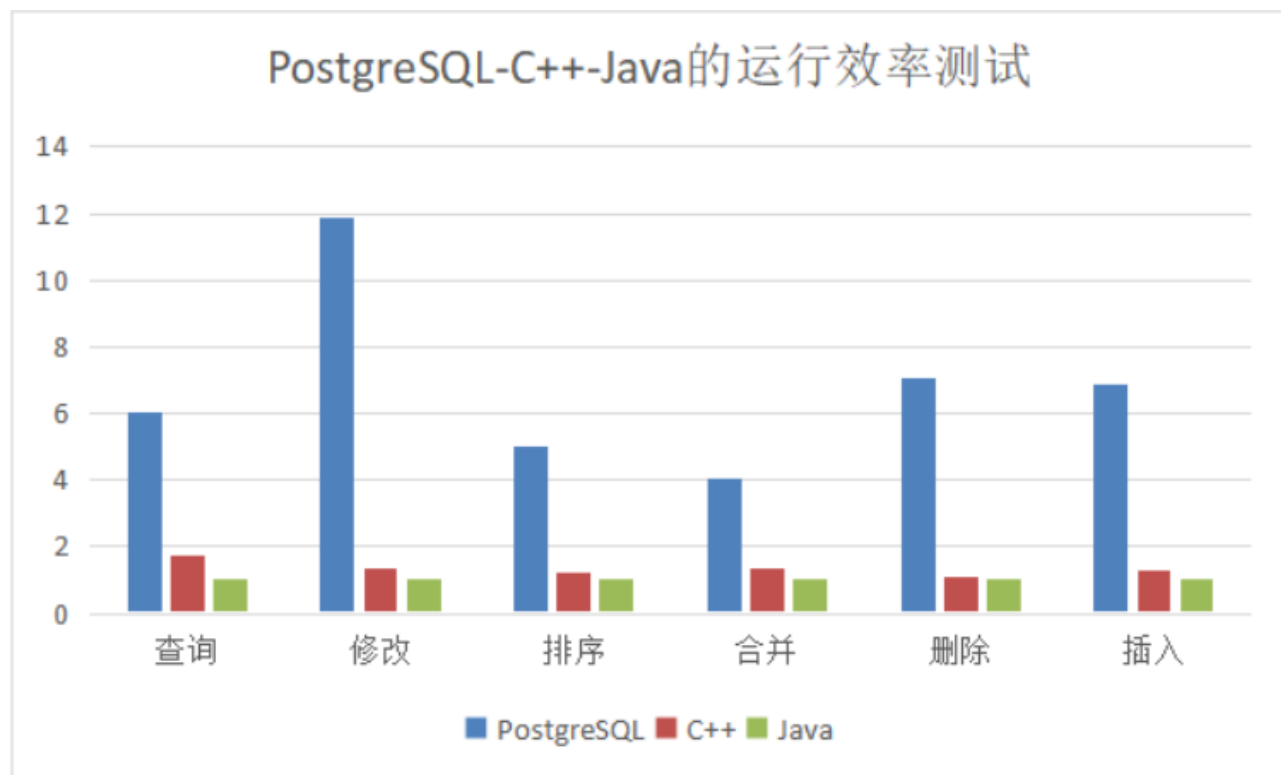
- 2.加速表的连接功能

- 索引劣势：

- 1.消耗额外内存空间

- 2.增加，删除，修改效率有所下降

# Part 3.实验设计-比较SQL与基础编程语言



- 基础操作类别：

查询、修改、排序、合并、删除、插入

- 结论：

PostgreSQL的效率**远胜**基础编程语言

# Part 3.实验设计-不同系统数据库差异

XiaoXinAir-14IIL 2020

设备名称 LAPTOP-Q2EITDTR  
处理器 Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz  
机带 RAM 16.0 GB (15.8 GB 可用)  
设备 ID  
产品 ID  
系统类型 64 位操作系统, 基于 x64 的处理器

macOS Catalina

版本 10.15.4

MacBook Air (13-inch, Early 2015)

处理器 1.6 GHz 双核 Intel Core i5

内存 8 GB 1600 MHz DDR3

图形卡 Intel HD Graphics 6000 1536 MB

	Windows	Mac
1	347950	2974721
2	114056	3958973
3	24737	53519
4	17917	21056
5	12066	16912

- 测试代码：  
ComparingDataManipulation.java
- 数据规模：100W
- 1-5：依次按照Part2中进行了插入过程的优化处理。
- 结论：
  - 1.运行内存对于交互效率有较大影响；
  - 2.算法的优化有较大的运行效率的影响。

# Part 4.Conclusion

- 学习了很多.....
  - 三大范式的理解
  - Java与数据库通讯的效率提升策略
  - Java哈希数据结构
  - 数据库高并发访问与事务管理
  - 数据库访问的用户权限管理
  - 数据库索引的优劣
  - 数据库系统效率与硬件的关系
  - 面对bug的耐心与细心
  - 面对各种报错的冷静处理能力
  - 团队之间的共同协作能力
  - .....

