视图

2025 年 4 月 11 日

一、主要任务：

1. 体验 view

2. 创建一个'Comp.Sci.' 学院的学生视图 std_view, 该视图包括学生 id、name 和 dept_name

（1）查询'Comp.Sci.' 学院的 J 开头的名字。

（2）插入一个 Jack 学生, 并查询。

（3）插入一个 Lee 学生, 并查询。

3. 采用任何程序语言实现 rank() 排名功能, 以 student 表的 tot_cred 大小排名并实现按某列分组排名。

二、实现方法：

1. 用 create view 语句可以创建视图，为多次实现该 sql 语句，在前面加入 drop view if exist，然后就可以使用 select 语句进行查询了.

2. 直接向视图里插入数据，再次进行查询，然后检查 student 原表内是否有添加。

3. 用 like'J%' 语句，可以查找到名字的第一个字母。

4. 使用 rank() over (order by) as cre 可以生成排序序列。

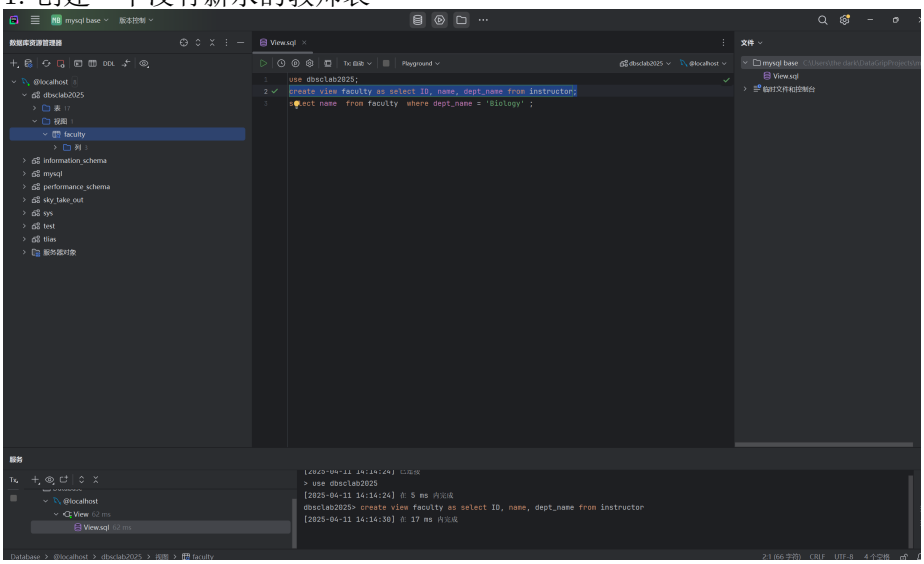
5. 加入 partition by 可以按照分组内部排序。

6. 最后按照 dept_name 和 cre 排名进行查询即可。

7. 自定义 rank 函数：使用 java 的 ssm 框架进行 rank 函数的定义，在 mapper 层使用 select * from student 将全部 student 数据取出来，将其封装在 ArrayList 中传给 service 层,service 层使用 TreeMap<String,ArrayList<Student» 在学生与学院间建立联系，并且 treeset 会自动根据学院名称进行排序，将学生全部插入后对每个 Arraylist 进行 sort 排序，这样就可以完成根据两个字段的分别排序

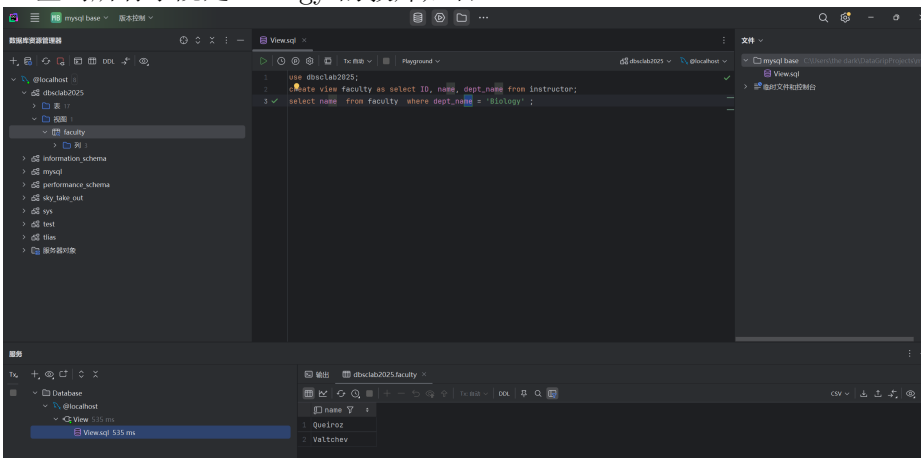
三、成果展示：

1. 创建一个没有薪水的教师表



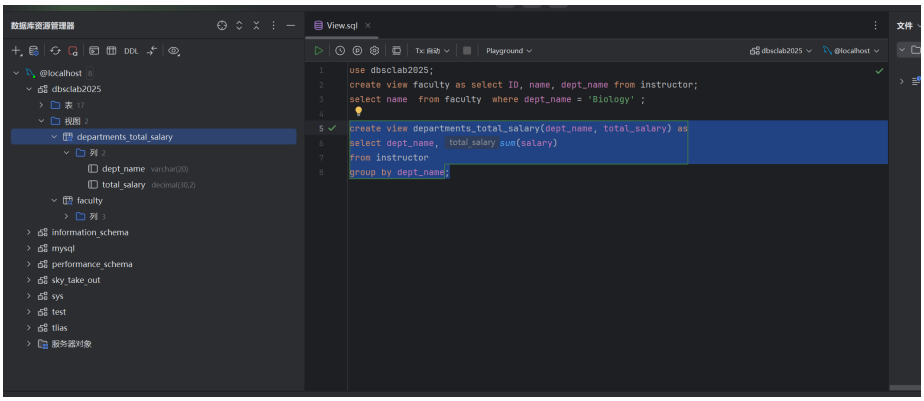
从右边栏可看到视图已经创建完毕

2. 查询所有学院是 biology 的教师姓名



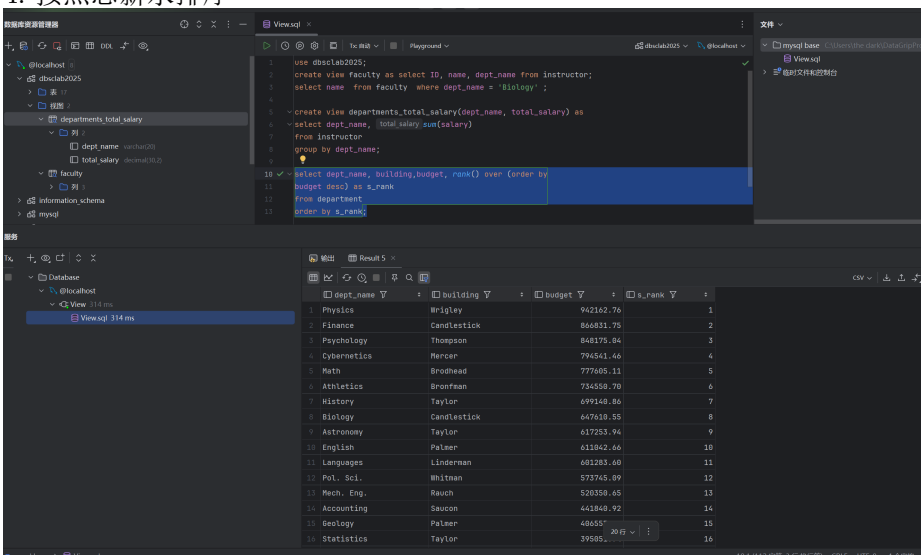
可以看到已经查询完毕

3. 创建学院和总薪水的视图

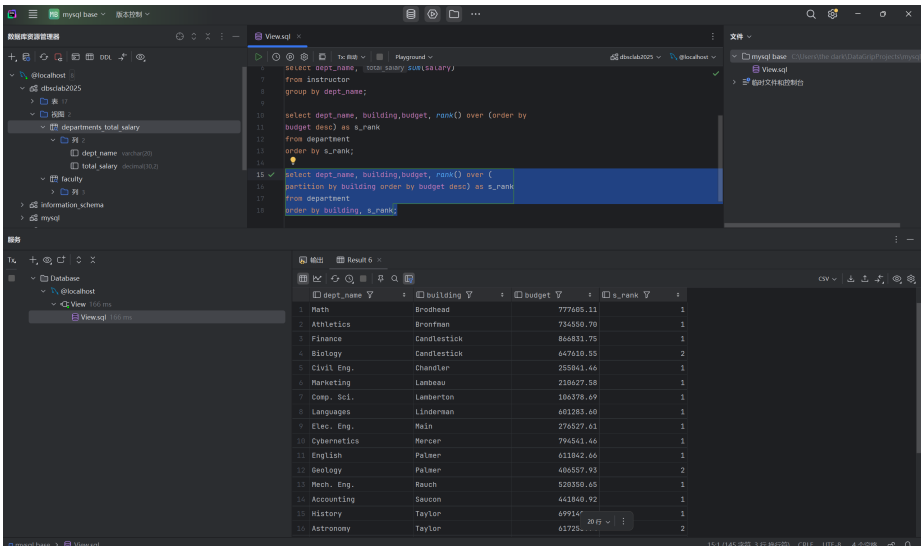


可以看到已经创建完毕

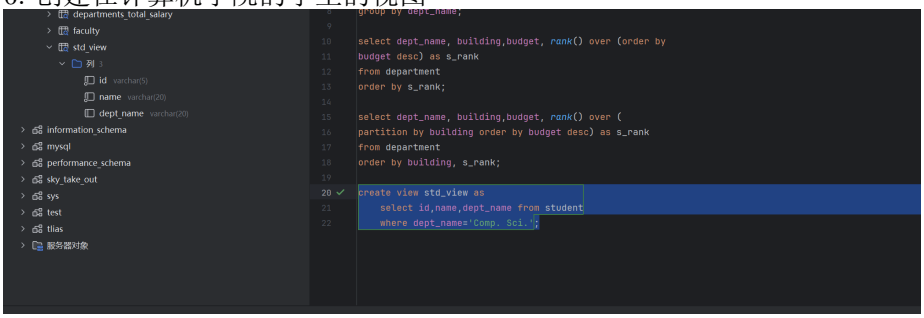
4. 按照总薪水排序



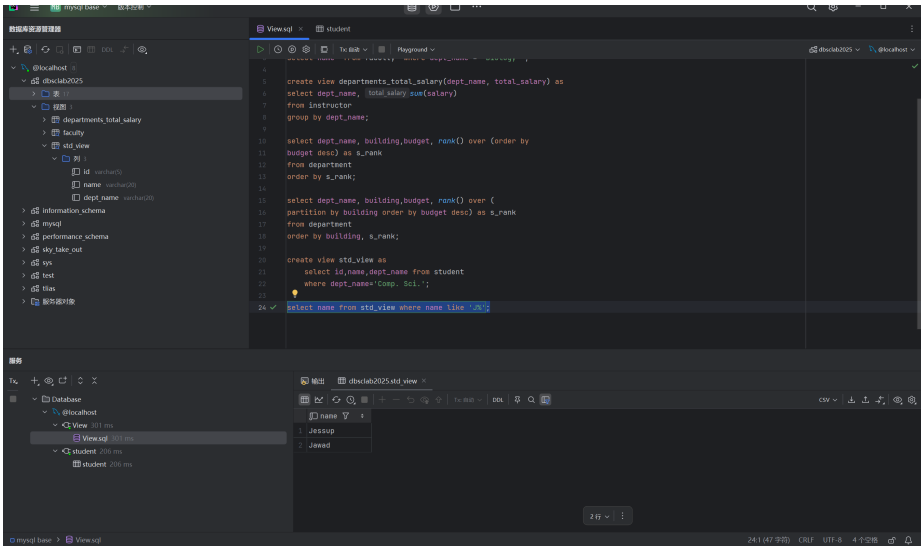
5. 分别查询按照 building 和 s_rank 排序的结果



6. 创建在计算机学院的学生的视图

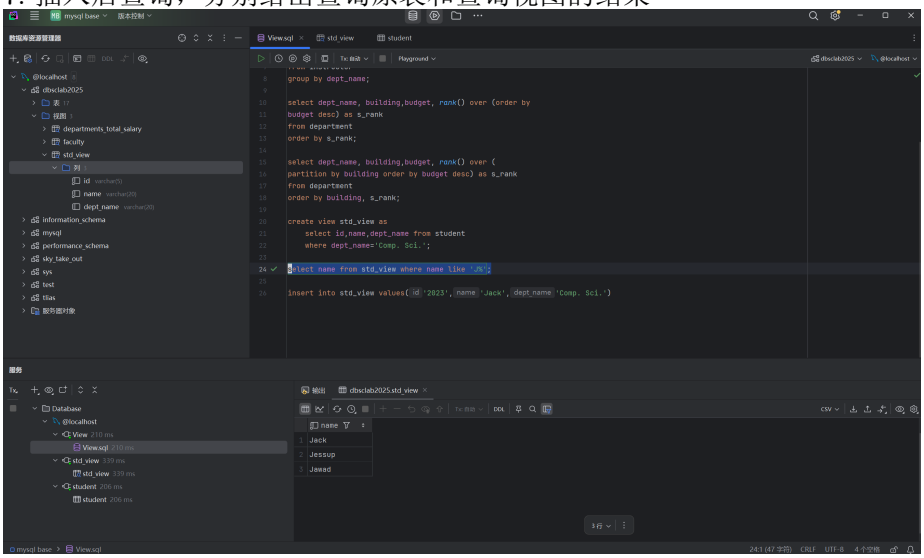


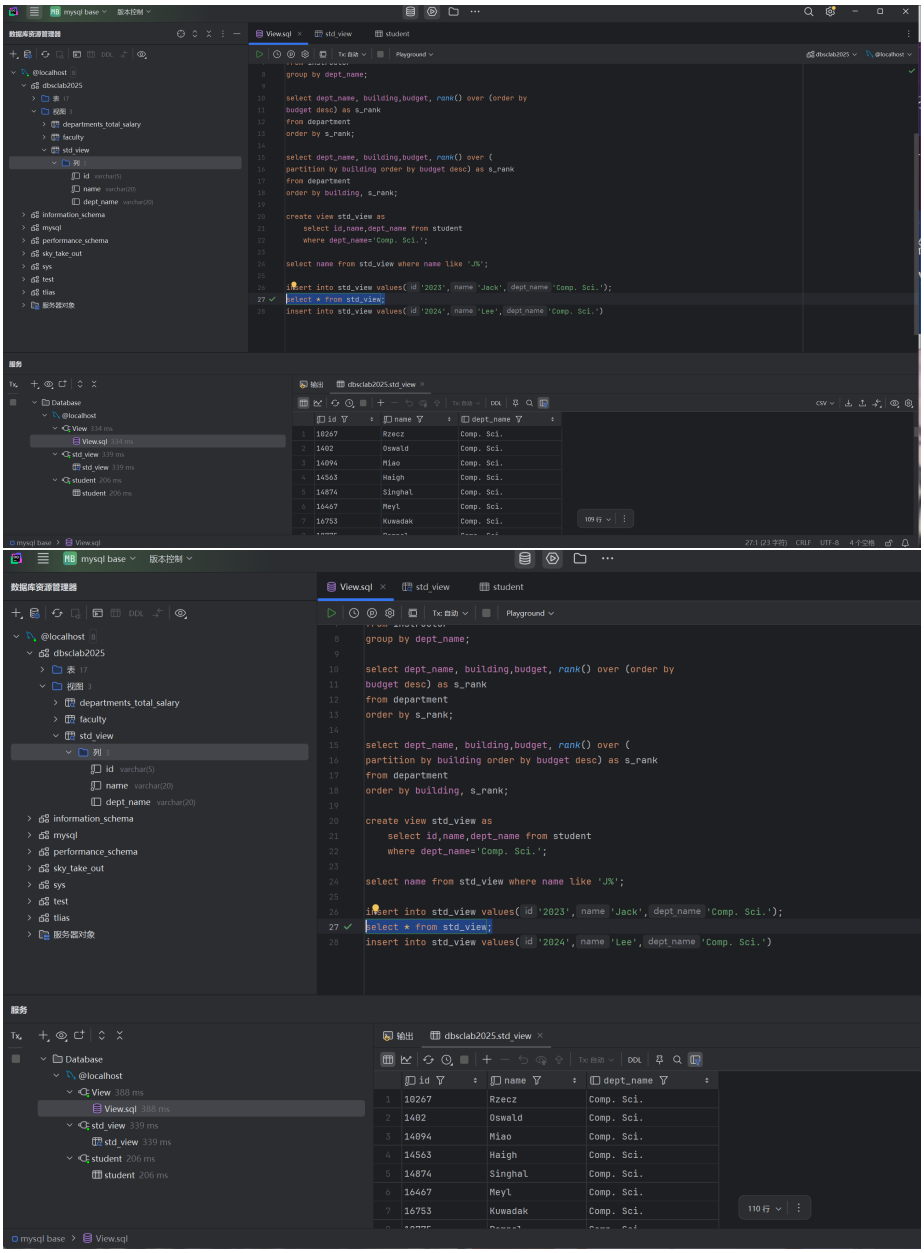
从其中查询开头是 J 的同学

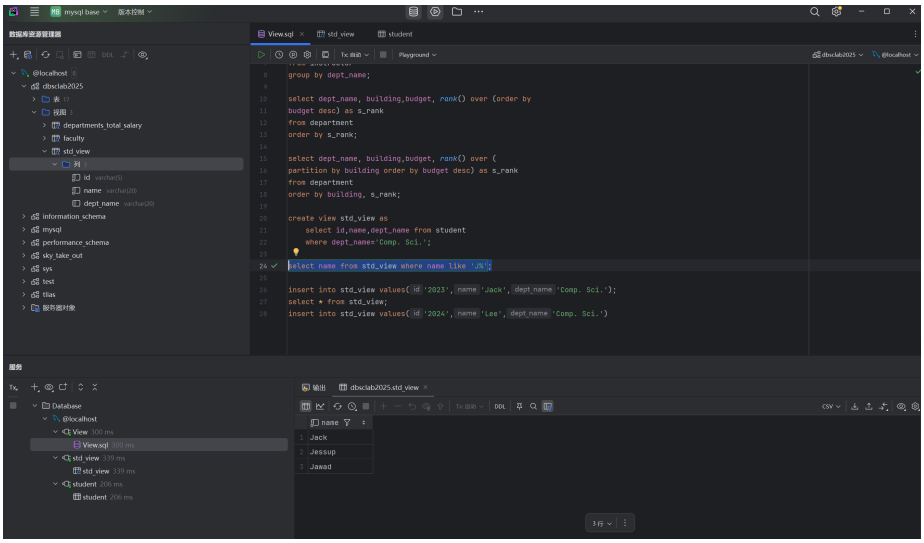


可见可以正确找到

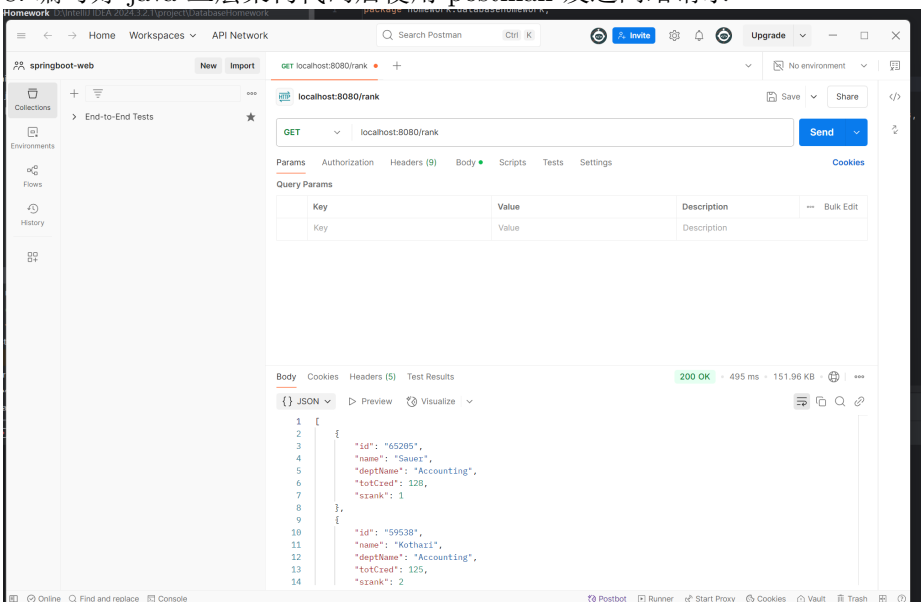
7. 插入后查询，分别给出查询原表和查询视图的结果



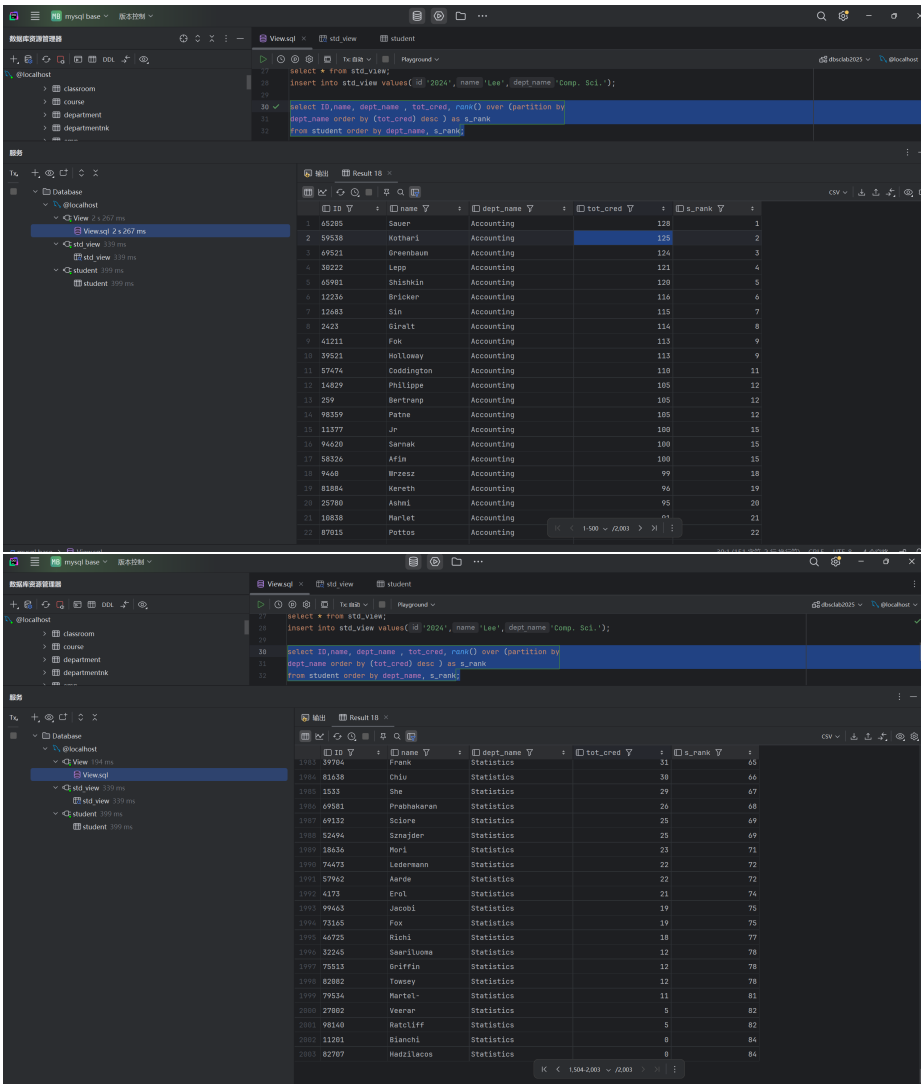

group by dept_name;
select dept_name, building,budget, rank() over (order by budget desc) as s_rank
from department
order by s_rank;
select dept_name, building,budget, rank() over (partition by building order by budget desc) as s_rank
from department
order by building, s_rank;
create view std_view as
select id,name,dept_name from student
where dept_name='Comp. Sci.';
select name from std_view where name like 'J%';
insert into std_view values('2023','Jack','Comp. Sci.');
select * from std_view;
insert into std_view values('2024','Lee','Comp. Sci.')
10267 Rzecz Comp. Sci.
1402 Oswald Comp. Sci.
14094 Miao Comp. Sci.
14563 Haigh Comp. Sci.
14874 Singhal Comp. Sci.
16467 Meyl Comp. Sci.
16753 Kuwadak Comp. Sci.

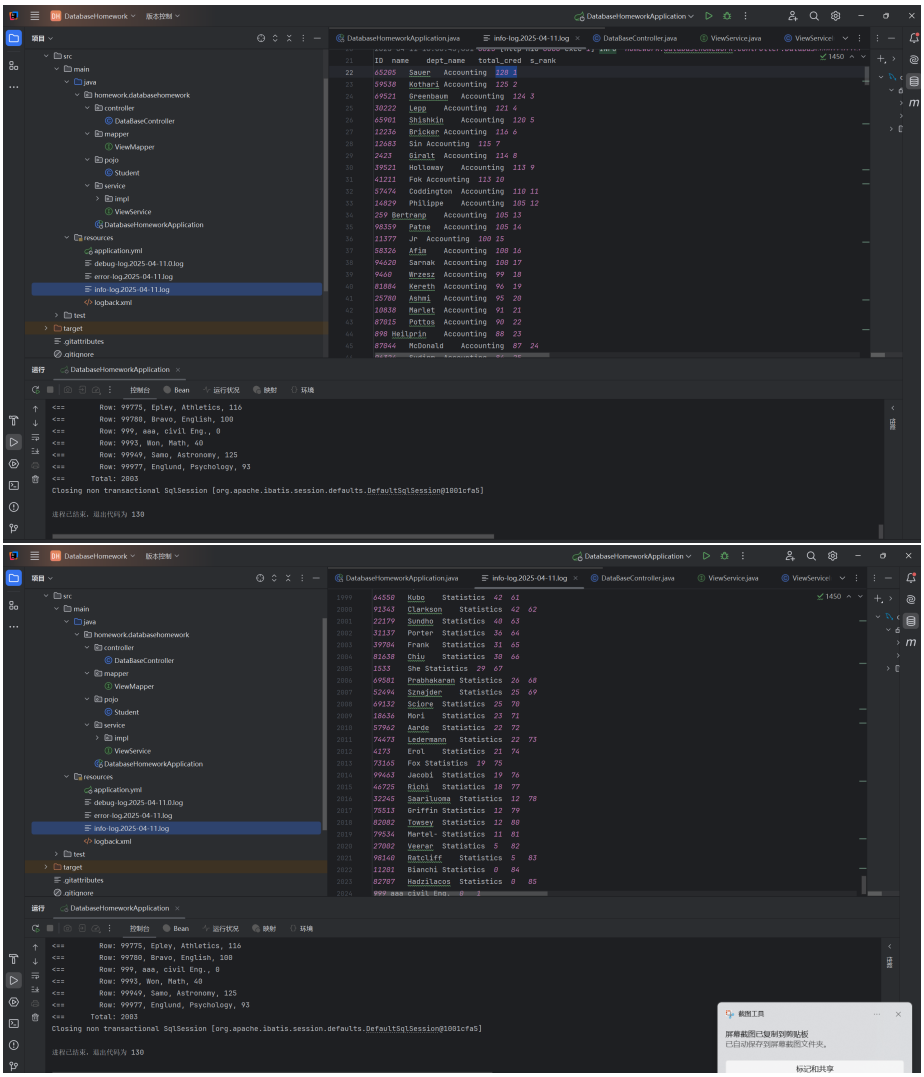


8. 编写好 java 三层架构代码后使用 postman 发送网络请求



将排序好的结果记录到 log 日志中，与 datagrip 查询出的结果对比


select * from std_view;
insert into std_view values(id '2024', name 'Lee', dept_name 'Comp. Sci.');
select ID,name, dept_name , tot_cred, rank() over (partition by
dept_name order by (tot_cred) desc) as s_rank
from student order by dept_name, s_rank;
ID name dept_name tot_cred s_rank
1 65285 Sauer Accounting 128 1
2 59538 Kothari Accounting 125 2
3 69521 Greenbaum Accounting 124 3
4 30222 Lepp Accounting 121 4
5 65981 Shishkin Accounting 120 5
6 12236 Bricker Accounting 116 6
7 12683 Sin Accounting 115 7
8 2423 Giralt Accounting 114 8
9 41211 Fok Accounting 113 9
10 39521 Holloway Accounting 113 9
11 57474 Coddington Accounting 110 11
12 14829 Philippe Accounting 105 12
13 259 Bertranp Accounting 105 12
14 98359 Patne Accounting 105 12
15 11377 Jr Accounting 100 15
16 94620 Sarnak Accounting 100 15
17 58326 Afin Accounting 100 15
18 9468 Mrzesz Accounting 99 18
19 81884 Kereth Accounting 96 19
20 25780 Ashmi Accounting 95 20
21 10838 Marlet Accounting 21
22 87015 Pottos Accounting 22



四、结论：

1. 在需要排名时，使用 rank()over(orderby...) 可以生成排序序列。

2. 创造视图后插入数据会直接插到真实表中，视图中不存在的属性设成 null。

3. 即使视图中无法显示，也可以通过视图插入的方法直接插到真实表中。

4. 使用 desc 可以逆序排列

5. 用 like 进行模糊匹配是很高效的

6. 使用 mybatis 框架操作数据库是十分方便的

五、java 的 rank 函数源代码：

Controller 层代码：接收网络请求并且向 service 层发送数据请求

```java
package homework.databasehomework.controller;

import homework.databasehomework.pojo.Student;
import homework.databasehomework.service.ViewService;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;

import java.util.ArrayList;

@RestController
@Slf4j
public class DataBaseController {
    @Autowired
    private ViewService viewService;

    @GetMapping("/rank")
    public ArrayList<Student> rank() {
        ArrayList<Student> students=viewService.rank();
        StringBuilder output=new StringBuilder();
        output.append("\nID\tname\tdept_name\ttotal_cred\
            ts_rank\n");
        for(Student student:students){
            output.append(String.format("%s\t%s\t%s\t%d\t%d\n",
                    student.getId(),student.getName(),student.
                        getDeptName(),student.getTotCred(),
                        student.getSRank()));
        }
        log.info(output.toString());
        return students;
    }
}
```

mapper 层代码，负责与数据库连接进行查询

```java
package homework.databasehomework.mapper;

import homework.databasehomework.pojo.Student;
import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Select;

import java.util.ArrayList;

@Mapper
public interface ViewMapper {

    @Select("select␣ID,␣name,␣dept_name,␣tot_cred␣from␣student"
        )
    public ArrayList<Student> rank();
}
```

service 层代码，负责业务逻辑处理

```java
package homework.databasehomework.service.impl;

import homework.databasehomework.mapper.ViewMapper;
import homework.databasehomework.pojo.Student;
import homework.databasehomework.service.ViewService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.TreeMap;
import java.util.TreeSet;

@Service
public class ViewServiceImpl implements ViewService {
    @Autowired
    private ViewMapper viewMapper;

    @Override
    public ArrayList<Student> rank() {
        ArrayList<Student> input=viewMapper.rank();
        TreeMap<String, ArrayList<Student>> map=new TreeMap<>()
```

```
            ;
        for(Student student:input){
            if(map.containsKey(student.getDeptName())){
                map.get(student.getDeptName()).add(student);
            }
            else
            {
                map.put(student.getDeptName(),new ArrayList<>()
                    );
                map.get(student.getDeptName()).add(student);
            }
        }
        for(ArrayList<Student> set:map.values()){
            set.sort(Student::compareTo);
            int i=1;
            for(Student student:set){
                student.setSRank(i++);
            }
        }
        ArrayList<Student> output=new ArrayList<>();
        for(ArrayList<Student> set:map.values()){
            for(Student student:set){
                output.add(student);
            }
        }
        return output;
    }
}
```

```
package homework.databasehomework.service;

import homework.databasehomework.pojo.Student;

import java.util.ArrayList;

public interface ViewService {
    ArrayList<Student> rank();
}
```

实体类，负责封装数据

```java
package homework.databasehomework.pojo;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@AllArgsConstructor
@NoArgsConstructor
public class Student implements Comparable<Student> {
    private String id;
    private String name;
    private String deptName;
    private int sRank;
    private int totCred;
    @Override
    public int compareTo(Student o) {
        return  o.totCred-this.totCred ;
    }
}
```