【MySQL】多表查询(四)

> 🚚MySQL学习·第四站~
> 🚩本文已收录至专栏：MySQL通关路
> ❤️‍🩹文末附全文思维导图，感谢各位点赞收藏支持~
> ⭐学习汇总贴，超详细思维导图：【MySQL】学习汇总(完整思维导图)

之前我们介绍DQL语句，也就是数据查询语句的时候，介绍的查询操作都是单表查询，他的功能当然不仅局限于此，我们还可以一次性对多个表的数据进行查询操作，也就是接下来要介绍的多表查询。

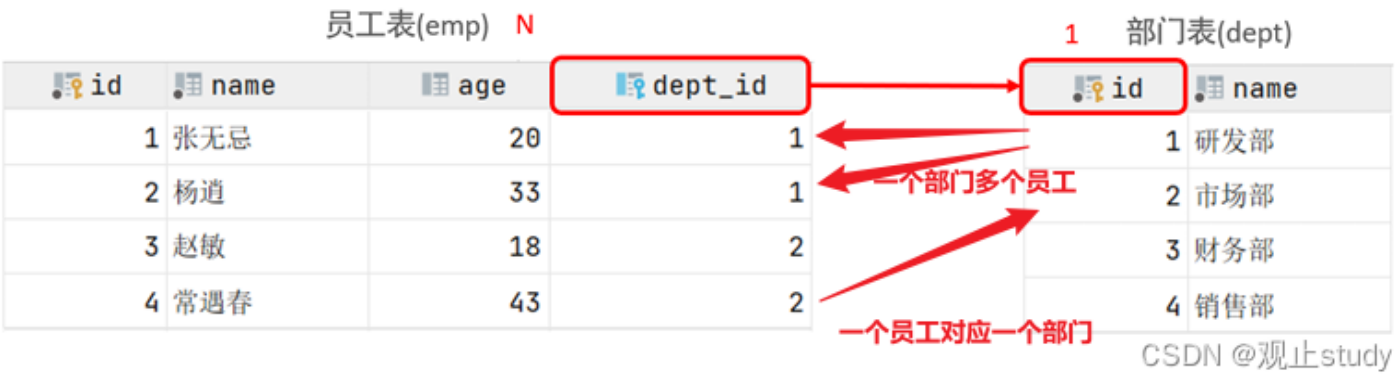
一.多表关系

在我们的项目开发中，在进行数据库表结构设计时，会根据业务需求及业务模块之间的关系，分析并设计表结构，由于业务之间经常会存在相互关联关系，所以由此设计出的各个表结构之间也存在着各种联系，基本上分为三种：

- 一对多(多对一)
- 多对多
- 一对一

(1) 一对多

- 业务示例: 部门与员工的关系
- 关系介绍: 一个部门对应多个员工，一个员工对应一个部门
- 实现方式: 我们通常会在 **'多'** 的一方建立**外键**，指向 **'一'** 的一方的**主键**



(2) 多对多

- 业务示例: 学生与课程的关系
- 关系介绍: 一个学生可以选修多门课程，一门课程也可以供多个学生选择
- 实现方式: **建立第三张中间表，中间表至少包含两个外键，分别关联两方主键**

(3) 一对一

- 业务示例: 用户与用户详情的关系
- 关系介绍: 一对一关系，**多用于单表拆分**，将一张表的基础字段放在一张表中，其他详情字段放在另 一张表中，以提升操作效率
- **实现方式: 在任意一方加入外键，关联另外一方的主键，并且设置外键为唯一的(UNIQUE)**

外键唯一。不唯一可能会导致出现一对多的异常情况

id	name	age	gender	phone
1	黄渤	45	1	18800001111
2	冰冰	35	2	18800002222
3	码云	55	1	18800008888
4	李彦宏	50	1	18800009999

用户基本信息表(tb_user)

id	degree	major	primaryschool	middleschool	university	userid
1	本科	舞蹈	静安区第一小学	静安区第一中学	北京舞蹈学院	1
2	硕士	表演	朝阳区第一小学	朝阳区第一中学	北京电影学院	2
3	本科	英语	杭州市第一小学	杭州市第一中学	杭州师范大学	3
4	本科	应用数学	阳泉第一小学	阳泉区第一中学	清华大学	4

用户教育信息表(tb_user_edu)

二.多表查询

(1) 引入

多表查询就是指一次性从多张表中查询数据。

原来我们查询单表数据，执行的SQL形式为：`select 字段列表 from 表名;`

现在我们想要执行多表查询，就只需要使用逗号分隔多张表即可，如：`select 字段列表 from 表名1, 表名2;`

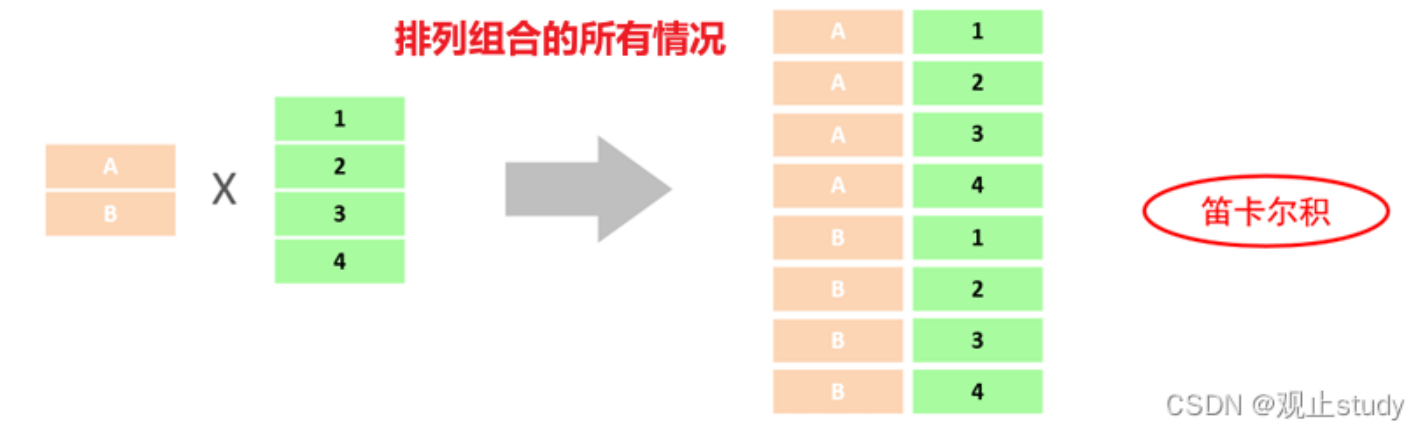
但是我们这样使用却发现存在问题：的确同时查到了多张表的数据，但是数据形式和我们想要的并不一样,它排列组合了两张表中的所有数据项!

select * from emp,dept;

数据项为 emp x dept 表集合

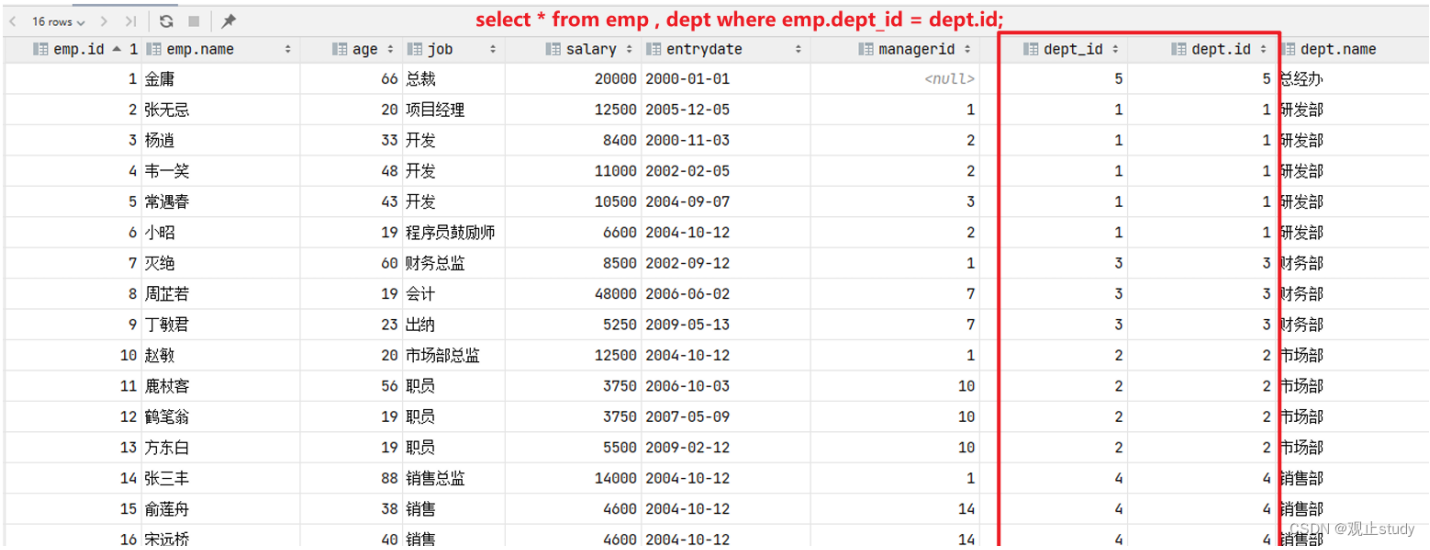
emp.id	emp.name	age	job	salary	entrydate	managerid	dept_id	dept.id	dept.name
1	金庸	66	总裁	20000	2000-01-01	<null>	5	6	人事部
1	金庸	66	总裁	20000	2000-01-01	<null>	5	5	总经办
1	金庸	66	总裁	20000	2000-01-01	<null>	5	4	销售部
1	金庸	66	总裁	20000	2000-01-01	<null>	5	3	财务部
1	金庸	66	总裁	20000	2000-01-01	<null>	5	2	市场部
1	金庸	66	总裁	20000	2000-01-01	<null>	5	1	研发部
2	张无忌	20	项目经理	12500	2005-12-05	1	1	6	人事部
2	张无忌	20	项目经理	12500	2005-12-05	1	1	5	总经办
2	张无忌	20	项目经理	12500	2005-12-05	1	1	4	销售部
2	张无忌	20	项目经理	12500	2005-12-05	1	1	3	财务部
2	张无忌	20	项目经理	12500	2005-12-05	1	1	2	市场部
2	张无忌	20	项目经理	12500	2005-12-05	1	1	1	研发部
3	杨逍	33	开发	8400	2000-11-03	2	1	6	人事部
3	杨逍	33	开发	8400	2000-11-03	2	1	5	总经办

例如我们查询员工、部门表，本来我们预期是每个员工对应其所在的部门，但事实确实，每个员工都对应了所有部门。这种现象也称之为 **笛卡尔积** 。



因此，在多表查询中，我们需要根据业务情况进行连接查询，消除无效的笛卡尔积，只保留两张表关联部分的有效数据。

例如在上述示例，我们通过`表名.字段名`指定员工表的外键等于部门表的主键即可获得预期数据~



16 rows

```
select * from emp , dept where emp.dept_id = dept.id;
```

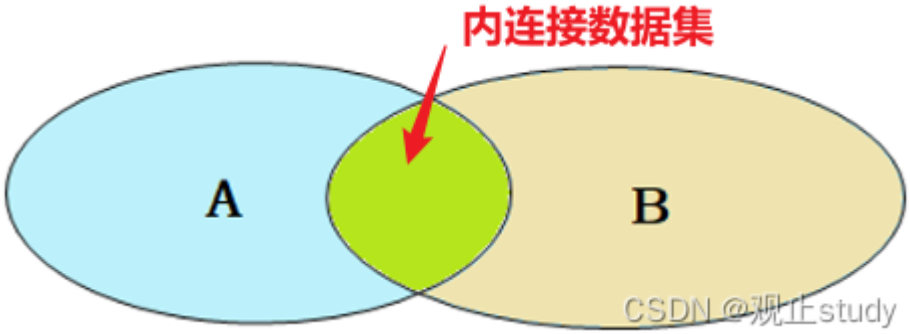
emp.id 1	emp.name	age	job	salary	entrydate	managerid	dept_id	dept.id	dept.name
1	金庸	66	总裁	20000	2000-01-01	<null>	5	5	总经办
2	张无忌	20	项目经理	12500	2005-12-05	1	1	1	研发部
3	杨逍	33	开发	8400	2000-11-03	2	1	1	研发部
4	韦一笑	48	开发	11000	2002-02-05	2	1	1	研发部
5	常遇春	43	开发	10500	2004-09-07	3	1	1	研发部
6	小昭	19	程序员鼓励师	6600	2004-10-12	2	1	1	研发部
7	灭绝	60	财务总监	8500	2002-09-12	1	3	3	财务部
8	周芷若	19	会计	48000	2006-06-02	7	3	3	财务部
9	丁敏君	23	出纳	5250	2009-05-13	7	3	3	财务部
10	赵敏	20	市场部总监	12500	2004-10-12	1	2	2	市场部
11	鹿杖客	56	职员	3750	2006-10-03	10	2	2	市场部
12	鹤笔翁	19	职员	3750	2007-05-09	10	2	2	市场部
13	方东白	19	职员	5500	2009-02-12	10	2	2	市场部
14	张三丰	88	销售总监	14000	2004-10-12	1	4	4	销售部
15	俞莲舟	38	销售	4600	2004-10-12	14	4	4	销售部
16	宋远桥	40	销售	4600	2004-10-12	14	4	4	销售部

特别说明:

- 我们不光可以为字段起别名，同样可以使用相同的语法给表起别名，`table as 别名或table 别名`
- 一旦为表起了别名，就不能再使用表名来指定对应的字段了，此时只能够使用别名来指定字 段
- 在获取表字段时，我们可以使用`表名.字段名`来进行指定。

(2) 内连接

内连接查询的是两张表交集部分的数据(也就是绿色部分的数据)。语法分为两种: 隐式内连接、显式内连接。



(2.1) 隐式内连接

- 语法

```
SELECT 字段列表 FROM 表1,表2 WHERE 限制条件;
```

- 示例

```
# 查询每一个员工的姓名 , 及关联的部门的名称
select e.name,d.name from emp e,dept d where e.dept_id = d.id;
```

```
select e.name,d.name from emp e , dept d where e.dept_id = d.id;
```

信息 | 结果1 | 概况 | 状态

name	name1
金庸	总经办
张无忌	研发部
杨逍	研发部
韦一笑	研发部
常遇春	研发部
小昭	研发部

(2.2) 显式外连接

- 语法

```
SELECT 字段列表 FROM 表1 [ INNER ] JOIN 表2 ON 连接条件 ... ;
```

- 示例（与上述相同）

```
# 查询每一个员工的姓名 , 及关联的部门的名称
select e.name, d.name from emp e inner join dept d on e.dept_id = d.id;
```

```
select e.name, d.name from emp e join dept d  on e.dept_id = d.id;
```

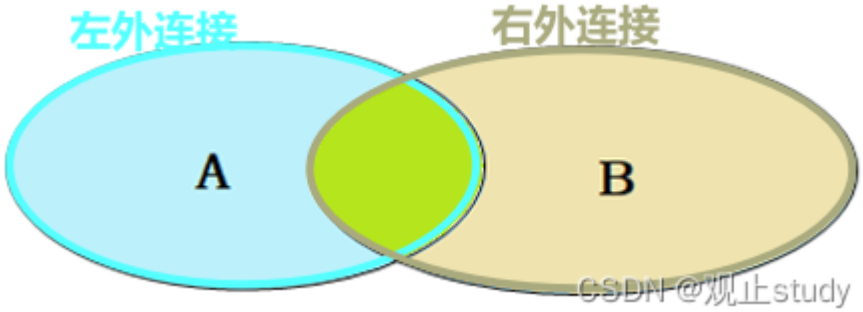
信息 | 结果1 | 概况 | 状态

name	name1
金庸	总经办
张无忌	研发部
杨逍	研发部
韦一笑	研发部
常遇春	研发部
小昭	研发部

可以看到显式内连接与隐式内连接查询结果一致

(3) 外连接

外连接分为：左外连接 和 右外连接。左外连接相当于查询表1(左表)的所有数据，当然也包含表1和表2交集部分的数据。右外连接相当于查询表2(右表)的所有数据，当然也包含表1和表2交集部分的数据。



(3.1) 左外连接

- 语法

```
SELECT 字段列表 FROM 表1 LEFT [ OUTER ] JOIN 表2 ON 条件 ... ;
```

- 示例:

```
# 查询emp表的所有数据, 和对应的部门信息
select e.*, d.name from emp e left outer join dept d on e.dept_id = d.id;
```

```
select e.*, d.name from emp e left outer join dept d on e.dept_id = d.id;
```

结果1 | 概况 | 状态

	name	age	job	salary	entrydate	managerid	dept_id	name1
1	金庸	66	总裁	20000	2000-01-01	(Null)	5	总经办
2	张无忌	20	项目经理	12500	2005-12-05	1	1	研发部
3	杨逍	33	开发	8400	2000-11-03	2	1	研发部
4	韦一笑	48	开发	11000	2002-02-05	2	1	研发部
5	常遇春	43	开发	10500	2004-09-07	3	1	研发部
6	小昭	19	程序员鼓励师	6600	2004-10-12	2	1	研发部
7	灭绝	60	财务总监	8500	2002-09-12	1	3	财务部
8	周芷若	19	会计	48000	2006-06-02	7	3	财务部
9	丁敏君	23	出纳	5250	2009-05-13	7	3	财务部
10	赵敏	20	市场部总监	12500	2004-10-12	1	2	市场部
11	鹿杖客	56	职员	3750	2006-10-03	10	2	市场部
12	鹤笔翁	19	职员	3750	2007-05-09	10	2	市场部
13	方东白	19	职员	5500	2009-02-12	10	2	市场部
14	张三丰	88	销售总监	14000	2004-10-12	1	4	销售部
15	俞莲舟	38	销售	4600	2004-10-12	14	4	销售部
16	宋远桥	40	销售	4600	2004-10-12	14	4	销售部
17	陈友谅	42	(Null)	2000	2011-10-12	1	(Null)	(Null)

不仔细看会觉得和内连接没啥区别，但是仔细看会发现我们null数据也能够正常显示出来，而内连接则无法查询到null，这是因为我们外连接会获取到一张表的全部数据，而内连接只获取交集部分数据。

(3.2) 右外连接

- 语法

```
SELECT 字段列表 FROM 表1 RIGHT [ OUTER ] JOIN 表2 ON 条件 ... ;
```

- 示例

```
# 查询emp表的所有数据, 和对应的部门信息
select e.*, d.name from emp e left outer join dept d on e.dept_id = d.id;
```

```sql
select d.*, e.* from emp e right outer join dept d on e.dept_id = d.id;
```

息 | 结果1 | 概况 | 状态

id	name	id1	name1	age	job	salary	entrydate	managerid	dept_id
1	研发部	6	小昭	19	程序员鼓励师	6600	2004-10-12	2	1
1	研发部	5	常遇春	43	开发	10500	2004-09-07	3	1
1	研发部	4	韦一笑	48	开发	11000	2002-02-05	2	1
1	研发部	3	杨逍	33	开发	8400	2000-11-03	2	1
1	研发部	2	张无忌	20	项目经理	12500	2005-12-05	1	1
2	市场部	13	方东白	19	职员	5500	2009-02-12	10	2
2	市场部	12	鹤笔翁	19	职员	3750	2007-05-09	10	2
2	市场部	11	鹿杖客	56	职员	3750	2006-10-03	10	2
2	市场部	10	赵敏	20	市场部总监	12500	2004-10-12	1	2
3	财务部	9	丁敏君	23	出纳	5250	2009-05-13	7	3
3	财务部	8	周芷若	19	会计	48000	2006-06-02	7	3
3	财务部	7	灭绝	60	财务总监	8500	2002-09-12	1	3
4	销售部	16	宋远桥	40	销售	4600	2004-10-12	14	4
4	销售部	15	俞莲舟	38	销售	4600	2004-10-12	14	4
4	销售部	14	张三丰	88	销售总监	14000	2004-10-12	1	4
5	总经办	1	金庸	66	总裁	20000	2000-01-01	(Null)	5
6	人事部	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)

这么一看感觉左右外连接查询结果都差不多，确实如此，通常左外连接和右外连接是可以相互替换的，只需要调整在连接查询时SQL中，表结构的先后顺序就可以了。而我们在日常开发使用时，更偏向于左外连接。

(4) 自连接

自连接查询，顾名思义，就是**自己连接自己**，也就是把一张表进行连接查询多次。需要注意的是在自连接查询中，必须要为表起别名，要不然我们不清楚所指定的条件、返回的字段，到底是哪一张表的字段。

- 自连接查询，可以是内连接查询，也可以是外连接查询，其关键点在于 **自己连接自己**

```sql
SELECT 字段列表 FROM 表A 别名A JOIN 表A 别名B ON 条件 ... ;
```

- 示例

```sql
# 查询员工 及其 所属领导的名字,如果员工没有领导, 也需要查询出来
select a.name '员工', b.name '领导' from emp a left join emp b on a.managerid =
b.id;
```

```sql
select a.name '员工', b.name '领导' from emp a
left join emp b on a.managerid = b.id;
```

自己连接查询自己

信息 结果1 概况 状态

员工	领导
金庸	(Null)
张无忌	金庸
杨逍	张无忌
韦一笑	张无忌
常遇春	杨逍
小昭	张无忌
灭绝	金庸
周芷若	灭绝
丁敏君	灭绝
赵敏	金庸
鹿杖客	赵敏
鹤笔翁	赵敏

(5) 联合查询

对于联合查询，就是把多次查询的结果合并起来，形成一个新的查询结果集。

- 语法说明

```sql
SELECT 字段列表 FROM 表A  ...   # 查询结果集1
UNION [ ALL ]
SELECT 字段列表 FROM 表B  ....;  # 查询结果集2
```

- 注意事项
  - 对于联合查询的多张表的**字段列表必须保持一致**，**字段类型也需要保持一致**，如果不一致将会报错。
  - `union all` 会将**全部的数据直接合并**在一起，`union` 会对合并之后的**数据去重**。
- 示例代码

```sql
select * from emp where salary < 5000
union all
select * from emp where age > 50;
```

信息 结果1 概况 状态

id	name	age	job	salary	entrydate	managerid	dept id
11	鹿杖客	56	职员	3750	2006-10-03	10	2
12	鹤笔翁	19	职员	3750	2007-05-09	10	2
15	俞莲舟	38	销售	4600	2004-10-12	14	4
16	宋远桥	40	销售	4600	2004-10-12	14	4
17	陈友谅	42	(Null)	2000	2011-10-12	1	(Null)
1	金庸	66	总裁	20000	2000-01-01	(Null)	5
7	灭绝	60	财务总监	8500	2002-09-12	1	3
11	鹿杖客	56	职员	3750	2006-10-03	10	2
14	张三丰	88	销售总监	14000	2004-10-12	1	4

salary < 5000

age > 50

`union all` 查询出来的结果，仅仅只对数据集进行简单的合并，查询结果中可能会存在重复数据项，使用 `union` 即可去除重复数据项。

(6) 子查询

(6.1) 介绍

SQL语句中**嵌套SELECT语句**，称为嵌套查询，又称子查询。

```
# 示例语法
SELECT * FROM t1 WHERE column1 = (SELECT column1 FROM t2);
```

子查询外部的语句可以是`INSERT / UPDATE / DELETE / SELECT`的任何一个。嵌套`SELECT语句`可以位于`SELECT之后`、`FROM之后`、`WHERE之后`。

- 子查询根据结果不同，分为：
  - **标量**子查询（子查询结果为**单个值**）
  - **列**子查询（子查询结果为**一列**）
  - **行**子查询（子查询结果为**一行**）
  - **表**子查询（子查询结果为**多行多列**）

(6.2) 标量子查询

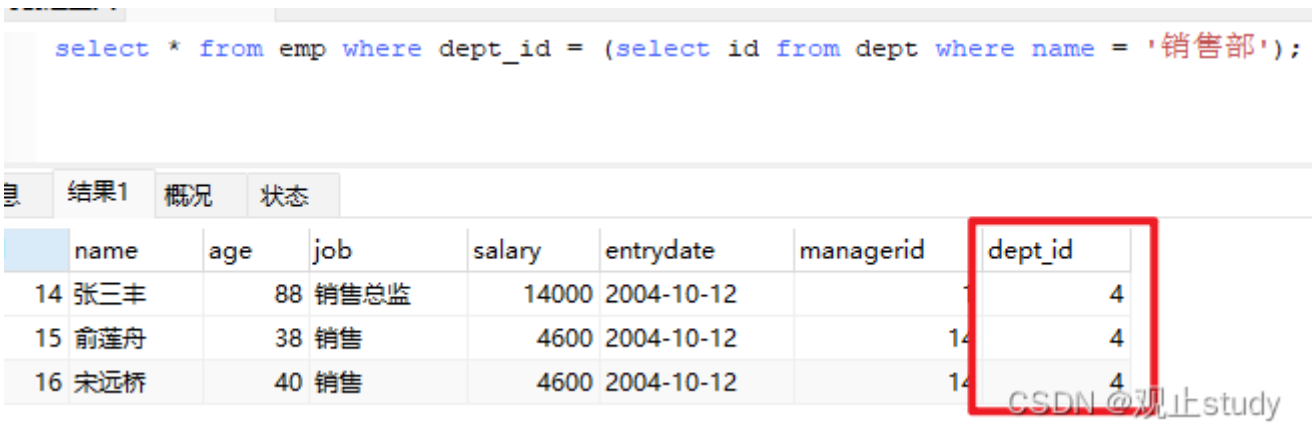
子查询返回的结果是**单个值**（例如数字、字符串、日期等）。 常用的操作符有：`= <> > >= < <=`

- 使用示例：查询 "销售部" 的所有员工信息

```
# 拆分1： 查询 "销售部" 部门ID，返回单个id值
select id from dept where name = '销售部';

# 拆分2：根据部门ID，查询员工信息
select id from dept where name = xxx;

# 完整版
 select * from emp where dept_id = (select id from dept where name = '销售部');
```



(6.3) 列子查询

子查询返回的结果是**一列**（可以是多行）。常用的操作符：`IN 、NOT IN 、 ANY 、SOME 、 ALL`

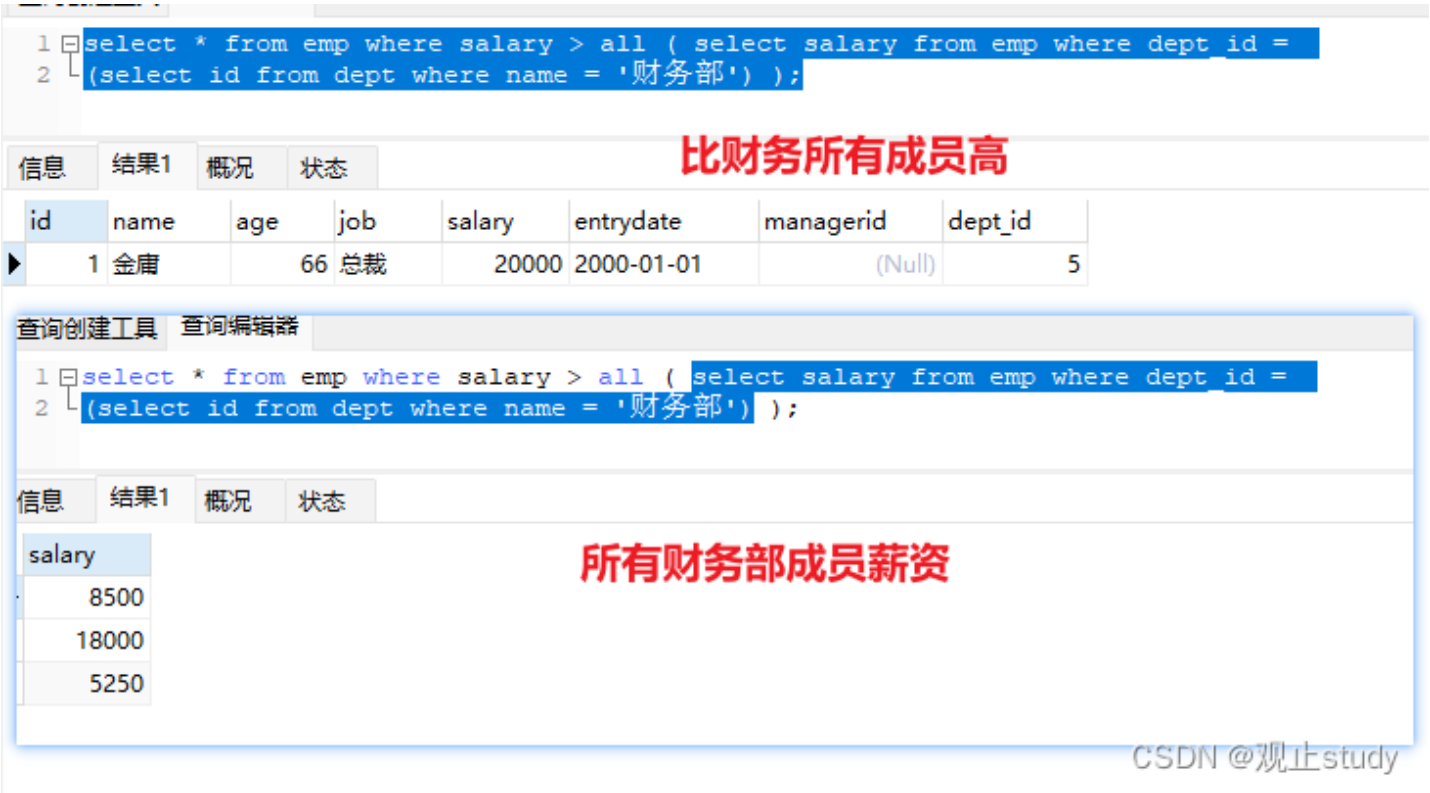
- 使用示例：查询比 财务部 所有人工资都高的员工信息

```sql
# 拆分1: 查询财务部id
select id from dept where name = '财务部';

# 拆分2:  查询财务部员工工资
select salary from emp where dept_id = 拆分1;
select salary from emp where dept_id = (select id from dept where name = '财务部');

#  拆分3 :  比 财务部 所有人工资都高的员工信息
select * from emp where salary > all (拆分2);

# 合并
select * from emp where salary > all ( select salary from emp where dept_id =
(select id from dept where name = '财务部') );
```



(6.4) 行子查询

子查询返回的结果是**一行**（可以是多列）。常用的操作符：`= 、<> 、IN 、NOT IN`

- 使用示例：查询与 "张无忌" 的薪资及直属领导相同的员工信息；

```sql
# 拆分1：查询 "张无忌" 的薪资及直属领导
select salary, managerid from emp where name = '张无忌';

# 拆分2：查询与 "张无忌" 的薪资及直属领导相同的员工信息
select * from emp where (salary,managerid) = (xx,xx);

# 合并
select * from emp where (salary,managerid) = (select salary, managerid from emp
where name = '张无忌');
```

```
select * from emp where (salary,managerid) = (select salary, managerid from emp
where name = '张无忌');
```

相同

信息 | 结果1 | 概况 | 状态

id	name	age	job	salary	entrydate	managerid	dept_id
2	张无忌	20	项目经理	12500	2005-12-05	1	1
10	赵敏	20	市场部总监	12500	2004-10-12	1	2

查询创建工具 | 查询编辑器

```
select * from emp where (salary,managerid) = (select salary, managerid from emp
where name = '张无忌');
```

信息 | 结果1 | 概况 | 状态

查询个人薪资以及领导id

salary	managerid
12500	1

(6.5) 表子查询

子查询返回的结果是**多行多列**。常用的操作符：IN

- 使用示例：查询与 "鹿杖客" , "宋远桥" 的职位和薪资相同的员工信息

```
# 拆分1：查询 "鹿杖客" , "宋远桥" 的职位和薪资
select job, salary from emp where name = '鹿杖客' or name = '宋远桥';

# 拆分2：查询与 "鹿杖客" , "宋远桥" 的职位和薪资相同的员工信息
select * from emp where (job,salary) in ( xxx );

 # 合并
 select * from emp where (job,salary) in (select job, salary from emp where name = '鹿杖客' or name
= '宋远桥');
```

查询创建工具 | 查询编辑器

```
select * from emp where (job,salary) in
(select job, salary from emp where name = '鹿杖客' or name = '宋远桥');
```

信息 | 结果1 | 概况 | 状态

id	name	age	job	salary	entrydate	managerid	dept_id
11	鹿杖客	56	职员	3750	2006-10-03	10	2
12	鹤笔翁	19	职员	3750	2007-05-09	10	2
15	俞莲舟	38	销售	4600	2004-10-12	14	4
16	宋远桥	40	销售	4600	2004-10-12	14	4

查询创建工具 | 查询编辑器

```
select * from emp where (job,salary) in
(select job, salary from emp where name = '鹿杖客' or name = '宋远桥');
```

信息 | 结果1 | 概况 | 状态

job	salary
职员	3750
销售	4600

最后值得一提是：多表查询业务能实现相关需求的SQL往往会很多，写法也多种多样，总之，能满足我们的需求，查询出符合条件的记录即可~

三.全文概览