

# 04.高级查询

---

## 一、单元概述

通过本章的学习能够理解MySQL数据库中分组查询的含义，掌握常用分组函数的使用，掌握GROUP BY子句的使用规则，掌握分组后数据结果的条件过滤，掌握SELECT语句执行过程，理解子查询的含义，掌握单行子查询和多行子查询的使用

## 二、教学重点与难点

**重点：** 掌握常用分组函数的使用 掌握GROUP BY子句的使用规则 掌握HAVING子句的使用规则 掌握子查询的使用规则

**难点：** SELECT语句执行过程 HAVING和WHERE的区别 单行子查询和多行子查询

## 4.1 为什么使用分组函数

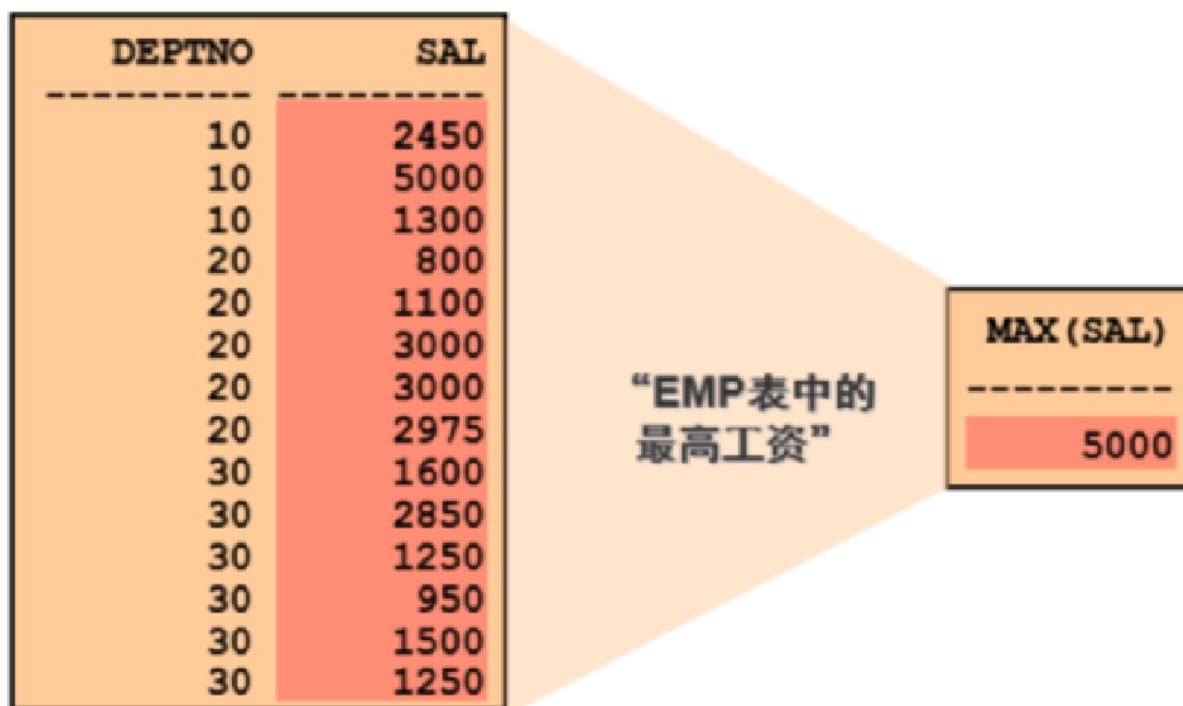
●请思考如下问题？

- 查询所有员工的每个月工资总和，平均工资？
- 查询工资最高和最低的工资是多少？
- 查询公司的总人数？

○查询有奖金的总人数?

○.....

- 分组函数是对数据行的集合进行操作并按组给出一个结果，这个结果可直接输出，或者用来做判断条件



The diagram illustrates a table with two columns: DEPTNO and SAL. The table contains 16 rows of data. A callout box labeled "EMP表中的最高工资" points to the row where DEPTNO is 10 and SAL is 5000. Another callout box labeled "MAX (SAL)" shows the result of the MAX function applied to the SAL column, which is 5000.

DEPTNO	SAL
10	2450
10	5000
10	1300
20	800
20	1100
20	3000
20	3000
20	2975
30	1600
30	2850
30	1250
30	950
30	1500
30	1250

“EMP表中的最高工资”

MAX (SAL)

5000

## 4.2 分组函数概述

### 4.2.1 分组函数

- 分组函数是对表中一组记录进行操作，每组只返回一个结果，即首先要对表记录进行分组，然后再进行操作汇总，每组返回一个结果，分组时可能是整个表分为一组，也可能根据条件分成多组。

- 分组函数常用到以下五个函数：
- - MIN
  - MAX
  - SUM
  - AVG
  - COUNT

## 4.2.2 分组函数的语法

```
SELECT      [column,] group_function(column)
FROM        table
[WHERE      condition]
[GROUP BY   column]
[HAVING
group_function(column)expression
[ORDER BY   column |
group_function(column)expression];
```

## 4.3 分组函数的使用

### 4.3.1 MIN函数和MAX函数

- MIN和MAX函数主要是返回每组的最小值和最大值。
- - MIN([DISTINCT | ALL] column | expression)
  - MAX([DISTINCT | ALL] column | expression)
- MIN和MAX可以用于任何数据类型

- 查询入职日期最早和最晚的日期

```
SELECT      MIN(hiredate), MAX(hiredate)
FROM        emp;
```

MIN(HIRED	MAX(HIRED
1980-12-17	1987-05-23

### 4.3.2 SUM函数和AVG函数

- SUM和AVG函数分别返回每组的总和及平均值。
- - SUM([DISTINCT | ALL] column | expression)
  - AVG([DISTINCT | ALL] column | expression)
- SUM和AVG函数都是只能够对数值类型的列或表达式操作。
- 查询职位以SALES开头的所有员工平均工资、最低工资、最高工资、工资和。

```
SELECT      AVG(sal), MAX(sal), MIN(sal),
SUM(sal)
FROM        emp
WHERE       job LIKE 'SALES%';
```

AVG(SAL)	MAX(SAL)	MIN(SAL)	SUM(SAL)
1400	1600	1250	5600

### 4.3.3 COUNT函数

## 1. COUNT函数语法

- COUNT函数的主要功能是返回满足条件的每组记录条数。

```
COUNT( * | {[DISTINCT | ALL] column |  
expression})
```

- COUNT(\*): 返回表中满足条件的行记录数
- 查询部门30有多少个员工

```
SELECT      COUNT(*)  
FROM        emp  
WHERE       deptno = 30;
```

COUNT(*)
6

## 2. COUNT函数使用

- COUNT( [DISTINCT | ALL] column | expression): 返回满足条件的非空(NULL)行的数量
- 查询部门30有多少个员工领取奖金

```
SELECT      COUNT(comm)  
FROM        emp  
WHERE       deptno = 30;
```

## 3. 组函数中DISTINCT

- DISTINCT会消除重复记录后再使用组函数

- 查询有员工的部门数量

```
SELECT      COUNT(DISTINCT deptno)
FROM        emp;
```

#### 4. 分组函数中空值处理

- 除了COUNT (\*) 之外，其它所有分组函数都会忽略列中的空值，然后再进行计算。

```
SELECT AVG(comm)
FROM    emp;
```

#### 5. 在分组函数中使用IFNULL函数

- IFNULL 函数可以使分组函数强制包含含有空值的记录

```
SELECT AVG(IFNULL(comm,0))
FROM    emp;
```

### 练习

1. 查询部门20的员工，每个月的工资总和及平均工资。
2. 查询工作在CHICAGO的员工人数，最高工资及最低工资。
3. 查询员工表中一共有几种岗位类型。

## 4.4 创建数据组

- 求各部门平均工资，按照部门进行分组

DEPTNO	SAL		
10	2450		
10	5000	2916.6667	
10	1300		
20	800		
20	1100	2175	
20	3000		
20	3000		
20	2975		
30	1600		
30	2850	1566.6667	
30	1250		
30	950		
30	1500		
30	1250		

“按部门  
分组求出  
各部门的  
平均工资”

DEPTNO	AVG (SAL)
10	2916.6667
20	2175
30	1566.6667

## 4.4.1 用GROUP BY子句创建数据组

- 通过 GROUP BY 子句可将表中满足WHERE条件的记录按照指定的列划分成若干个小组
- 其中GROUP BY子句指定要分组的列

```
SELECT      column, group_function(column)
FROM        table
[WHERE      condition]
[GROUP BY   group_by_expression]
[ORDER BY   column];
```

## 4.4.2 GROUP BY 子句的使用

- 查询每个部门的编号，平均工资

```
SELECT    deptno, AVG(sal)
FROM      emp
GROUP BY  deptno;
```

DEPTNO	AVG(SAL)
10	2916.6667
20	2175
30	1566.6667

### 4.4.3 GROUP BY 子句的使用规则

- 在SELECT列表中除了分组函数那些项，所有列都必须包含在GROUP BY 子句中。

```
SELECT    deptno, AVG(sal)
FROM      emp
GROUP BY  deptno;
```

DEPTNO	AVG(SAL)
10	2916.6667
20	2175
30	1566.6667

- GROUP BY 所指定的列并不是必须出现在SELECT 列表中。

```
SELECT    AVG(sal)
FROM      emp
GROUP BY  deptno;
```



AVG(SAL)
2916.6667
2175
1566.6667

## 4.4.4 按多个列分组

- 查询每个部门每个岗位的工资总和。

DEPTNO	JOB	SAL
10	MANAGER	2450
10	PRESIDENT	5000
10	CLERK	1300
20	CLERK	800
20	CLERK	1100
20	ANALYST	3000
20	ANALYST	3000
20	MANAGER	2975
30	SALESMAN	1600
30	MANAGER	2850
30	SALESMAN	1250
30	CLERK	950
30	SALESMAN	1500
30	SALESMAN	1250

“求出每个部门内的每个工种的薪水合计”

DEPTNO	JOB	SUM(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	6000
20	CLERK	1900
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	5600

- 查询每个部门每个岗位的工资总和

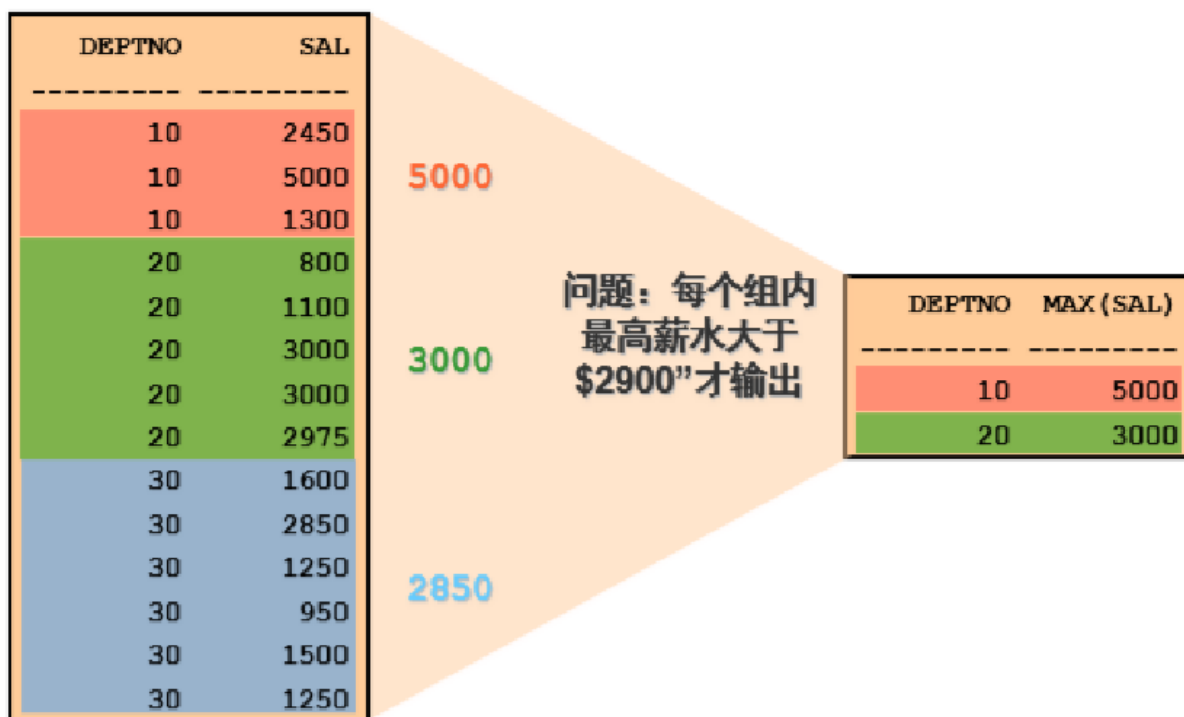
```
SELECT deptno, job, sum(sal)
FROM emp
GROUP BY deptno, job;
```

DEPTNO	JOB	SUM(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	6000
20	CLERK	1900
...		

## 练习

1. 查询每个部门的部门编号，部门名称，部门人数，最高工资，最低工资，工资总和，平均工资。
2. 查询每个部门，每个岗位的部门编号，部门名称，岗位名称，部门人数，最高工资，最低工资，工资总和，平均工资。
3. 查询每个经理所管理的人数，经理编号，经理姓名，要求包括没有经理的人员信息。

## 4.5 排除组结果



## 4.5.1 使用组函数的非法的查询

- 不能在 WHERE 子句中限制组
- 可以通过 HAVING 子句限制组

```
SELECT      deptno, max(sal)
FROM        emp
WHERE       max(sal) > 2900
GROUP BY    deptno;
```

## 4.5.2 用 HAVING 子句排除组结果

- 使用 HAVING 子句限制组
- ○ 记录已经分组.
- ○ 使用过组函数.

- 与 HAVING 子句匹配的结果才输出

```
SELECT      column, group_function
FROM        table
[WHERE      condition]
[GROUP BY   group_by_expression]
[HAVING     group_condition]
[ORDER BY   column];
```

### 4.5.3 HAVING 子句的使用

- 查询每个部门最高工资大于2900的部门编号，最高工资

```
SELECT      deptno, max(sal)
FROM        emp
GROUP BY    deptno
HAVING      max(sal)>2900;
```

DEPTNO	MAX(SAL)
10	5000
20	3000

```
SELECT      job, SUM(sal) PAYROLL
FROM        emp
WHERE       job NOT LIKE 'SALES%'
GROUP BY    job
HAVING      SUM(sal)>5000
ORDER BY    SUM(sal);
```

JOB	PAYROLL
ANALYST	6000
MANAGER	8275

## 4.5.4 SELECT语句执行过程

- 通过案例解释SELECT语句的执行过程。

```
SELECT      deptno,job,avg(sal)
FROM        emp
WHERE       job in
('SALESMAN','MANAGER','CLERK')
GROUP BY   deptno,job
HAVING avg(sal)>1000
ORDER BY   3 DESC;
```

DEPTNO	JOB	AVG(SAL)
20	MANAGER	2975
30	MANAGER	2850
10	MANAGER	2450
30	SALESMAN	1400
10	CLERK	1300

- SELECT语句执行过程：
  - 通过FROM子句中找到需要查询的表；
  - 通过WHERE子句进行非分组函数筛选判断；
  - 通过GROUP BY子句完成分组操作；
  - 通过HAVING子句完成组函数筛选判断；

5. 通过SELECT子句选择显示的列或表达式及组函数;
6. 通过ORDER BY子句进行排序操作。

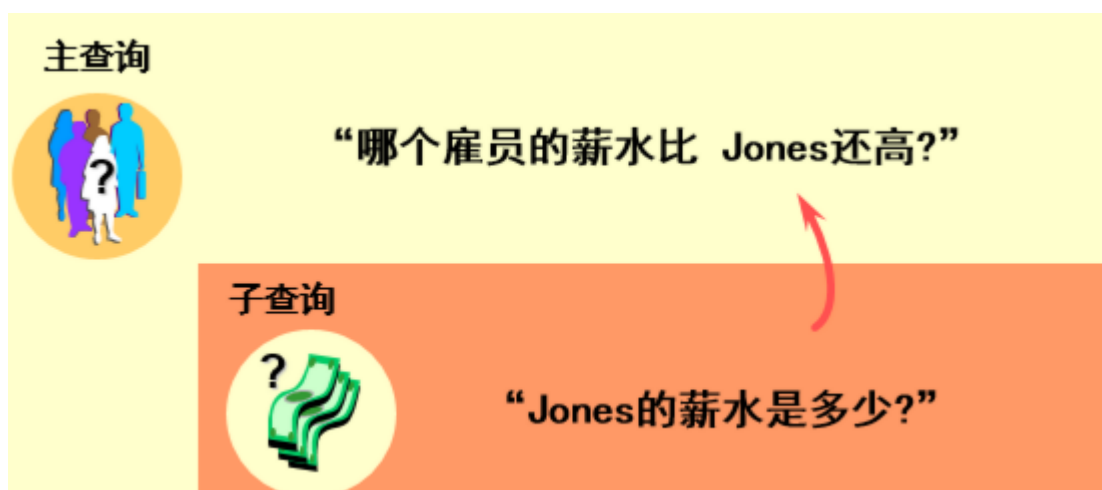
## 练习

1. 查询部门人数大于2的部门编号, 部门名称, 部门人数。
2. 查询部门平均工资大于2000, 且人数大于2的部门编号, 部门名称, 部门人数, 部门平均工资, 并按照部门人数升序排序。

## 4.6 子查询概述

### 4.6.1 为什么使用子查询

- 思考如下问题?
  - ○ 查询工资比Jones工资高的员工信息?
  - ○ 查询工资最低的员工姓名?
- “谁的薪水比 Jones 还高呢?”



- 子查询语法

```
SELECT      select_list
FROM        table
WHERE       expr operator
            (SELECT      select_list
              FROM        table);
```

- 括号内的查询叫做子查询，也叫内部查询，先于主查询执行。
- 子查询的结果被主查询（外部查询）使用
- expr operator包括比较运算符
  - ○ 单行运算符：>、=、>=、<、<>、<=
  - ○ 多行运算符：IN、ANY、ALL
- 子查询可以嵌于以下SQL子句中：
  - ○ WHERE子句
  - ○ HAVING子句
  - ○ FROM子句

## 4.6.2 使用子查询

- 查询出比JONES为雇员工资高的其他雇员

```

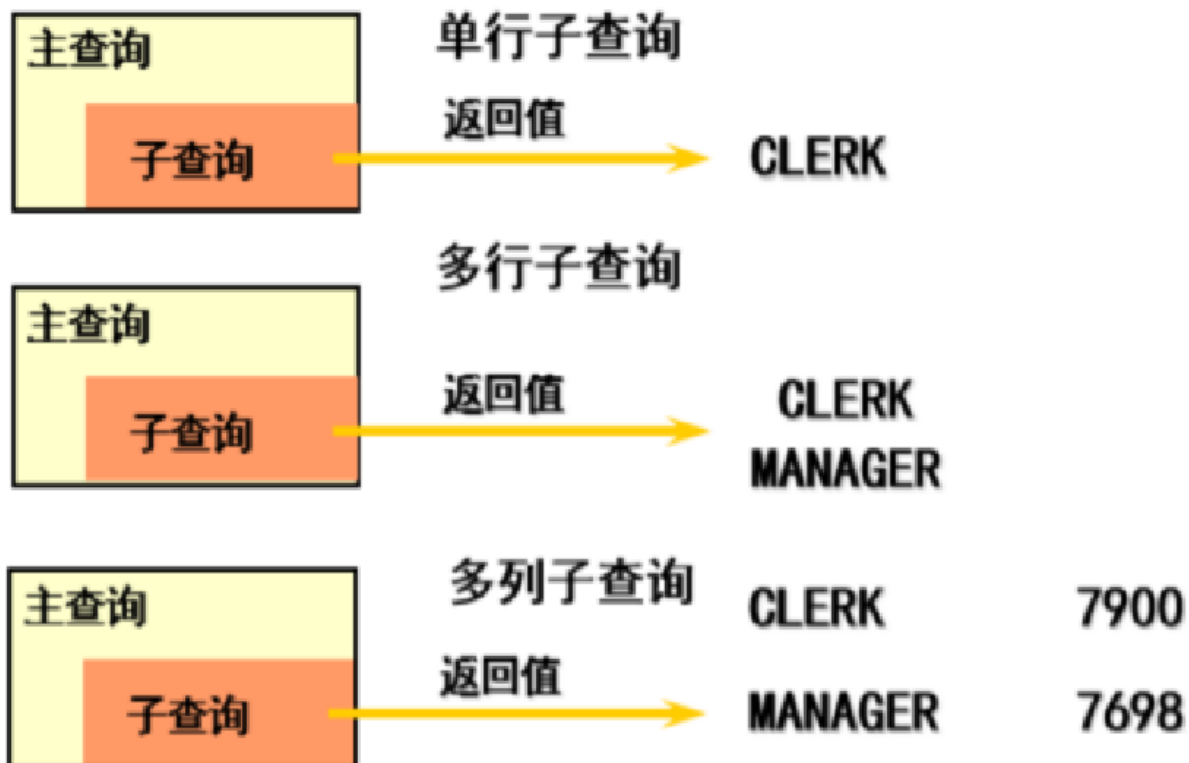
SELECT  ename
FROM    emp
WHERE   sal >
        (SELECT  sal
         FROM    emp
         WHERE   ename='JONES');

```

ENAME
KING
FORD
SCOTT

### 4.6.3 子查询类型

根据子查询返回的行和列数量，分为：





## 4.6.4 子查询使用指导

- 子查询要用括号括起来
- 将子查询放在比较运算符的右边
- 对于单行子查询要使用单行运算符
- 对于多行子查询要使用多行运算符

## 4.7 单行子查询

### 4.7.1 单行子查询概述

- 子查询只返回一行一列
- 使用单行运算符

### 4.7.2 单行子查询语句

- 显示和雇员7369从事相同工作并且工资大于雇员7876的雇员的姓名和工作。

```
SELECT    ename, job
FROM      emp
WHERE     job =
          (SELECT job      FROM emp WHERE empno =
7369)
          AND sal >
          (SELECT sal FROM emp WHERE empno =
7876);
```

### 4.7.3 子查询中使用组函数

- 查询工资最低的员工姓名，岗位及工资

```
SELECT      ename, job, sal
FROM        emp
WHERE       sal =
           (SELECT      MIN(sal) FROM emp);
```

### 4.7.4 HAVING子句中使用子查询

- 查询部门最低工资比20部门最低工资高的部门编号及最低工资

```
SELECT      deptno, MIN(sal)
FROM        emp
GROUP BY    deptno
HAVING      MIN(sal) >
           (SELECT      MIN(sal)
            FROM        emp
            WHERE        deptno = 20);
```

### 练习

1. 查询入职日期最早的员工姓名，入职日期
2. 查询工资比SMITH工资高并且工作地点在CHICAGO的员工姓名，工资，部门名称

3. 查询入职日期比20部门入职日期最早的员工还要早的员工姓名，入职日期

## 4.8 多行子查询

### 4.8.1 多行子查询概述

- 子查询返回记录的条数 可以是一条或多条。
- 和多行子查询进行比较时，需要使用多行操作符，多行操作符包括：
  - IN
  - ANY
  - ALL
- IN操作符和以前介绍的功能一致，判断是否与子查询的任意一个返回值相同。

### 4.8.2 IN使用

```
SELECT      ename, sal
FROM        emp
WHERE       empno IN (SELECT mgr
                      FROM   emp);
```

ENAME	SAL
SCOTT	3000.00
KING	5000.00
JONES	2975.00
FORD	3000.00
CLARK	2450.00
BLAKE	2850.00

### 4.8.3 ANY的使用

- ANY：表示和子查询的任意一行结果进行比较，有一个满足条件即可。
- ○ < ANY：表示小于子查询结果集中的任意一个，即小于最大值就可以。
- ○ > ANY：表示大于子查询结果集中的任意一个，即大于最小值就可以。
- ○ = ANY：表示等于子查询结果中的任意一个，即等于谁都可以，相当于IN。
- 案例1 查询是经理的员工姓名，工资。

```
SELECT      ename, sal
FROM        emp
WHERE       empno = ANY (SELECT mgr
                           FROM   emp);
```

ENAME	SAL
JONES	2975.00
BLAKE	2850.00
CLARK	2450.00
SCOTT	3000.00
KING	5000.00
FORD	3000.00

- 案例2 查询部门编号不为10，且工资比10部门任意一名员工工资高的员工编号，姓名，职位，工资。

```
SELECT empno, ename, job, sal
FROM emp
WHERE sal > ANY (SELECT sal
                  FROM emp
                  WHERE deptno = 10)
AND deptno <> 10;
```

## 4.8.4 ALL的使用

- ALL：表示和子查询的所有行结果进行比较，每一行都必须都满足条件。
- ○ < ALL:表示小于子查询结果集中的所有行，即小于最小值。
- ○ >ALL:表示大于子查询结果集中的所有行，即大于最大值。

○ = ALL :表示等于子查询结果集中的所有行,即等于所有值, 通常无意义。

- 案例1 查询部门编号不为20, 且工资比20部门所有员工工资高的员工编号, 姓名, 职位, 工资。

```
SELECT empno, ename, job, sal
FROM emp
WHERE sal > ALL (SELECT sal
                  FROM emp
                  WHERE deptno= 20)
AND deptno <> 20;
```

- 案例2 查询部门编号不为10, 且工资比10部门所有员工工资低的员工编号, 姓名, 职位, 工资。

```
SELECT empno, ename, job, sal
FROM emp
WHERE sal < ALL (SELECT sal
                  FROM emp
                  WHERE deptno= 10)
AND deptno <> 10;
```

EMPNO	ENAME	JOB	SAL
7369	SMITH	CLERK	800.00
7521	WARD	SALESMAN	1250.00
7654	MARTIN	SALESMAN	1250.00
7876	ADAMS	CLERK	1100.00
7900	JAMES	CLERK	950.00

## 练习

1. 查询入职日期比10部门任意一个员工晚的员工姓名、入职日期，不包括10部门员工
2. 查询入职日期比10部门所有员工晚的员工姓名、入职日期，不包括10部门员工
3. 查询职位和10部门任意一个员工职位相同的员工姓名，职位，不包括10部门员工

## 4.9 子查询中的空值

- 查询不是经理的员工姓名。

```
SELECT      ename
FROM        emp
WHERE       empno NOT IN
           (SELECT mgr
            FROM   emp);
```

- 子查询返回的结果中含有空值
- 上面的SQL语句试图查找出没有下属的雇员，逻辑上，这个SQL语句应该会返回8条记录，但是却一条也没返回，why?
- 因为子查询的结果中有一条空值，这条空值导致主查询没有记录返回。这是因为所有的条件和空值比较结果都是空值。因此无论什么时候只要空值有可能成为子查询结果集中的一部分，就不能使用NOT IN 运算符

## 4.10 在 FROM 子句中使用子查询

- 查询比自己部门平均工资高的员工姓名，工资，部门编号，部门平均工资

```
SELECT  a.ename, a.sal, a.deptno, b.salavg
FROM    emp a, (SELECT  deptno, AVG(sal)
                salavg
                        FROM    emp
                        GROUP BY deptno) b
WHERE   a.deptno = b.deptno AND    a.sal >
b.salavg;
```

ename	sal	deptno	salavg
KING	5000.00	10	2916.666667
JONES	2975.00	20	2175.000000
SCOTT	3000.00	20	2175.000000
FORD	3000.00	20	2175.000000
ALLEN	1600.00	30	1566.666667
BLAKE	2850.00	30	1566.666667

## 4.11 本章小结

- MIN函数和MAX函数
- SUM函数和AVG函数
- COUNT函数
- 组函数中空值处理
- 通过GROUP BY子句进行分组汇总



- HAVING子句的使用
- 单行子查询
- 多行子查询
- 子查询中空值问题

## 4.12 课后作业

1. 查询部门平均工资在2500元以上的部门名称及平均工资。
2. 查询员工岗位中不是以“SA”开头并且平均工资在2500元以上的岗位及平均工资，并按平均工资降序排序。
3. 查询部门人数在2人以上的部门名称、最低工资、最高工资。
4. 查询岗位不为SALESMAN，工资和大于等于2500的岗位及每种岗位的工资和。
5. 显示经理号码和经理姓名，这个经理所管理员工的最低工资，没有经理的KING也要显示，不包括最低工资小于3000的，按最低工资由高到低排序
6. 查询工资高于编号为7782的员工工资，并且和7369号员工从事相同工作的员工的编号、姓名及工资。
7. 查询工资最高的员工姓名和工资。
8. 查询部门最低工资高于10号部门最低工资的部门的编号、名称及部门最低工资。
9. 查询员工工资为其部门最低工资的员工的编号和姓名及工资。
10. 显示经理是KING的员工姓名，工资。

11. 显示比员工SMITH参加工作时间晚的员工姓名，工资，参加工作时间。