牛客网SQL编程部分

查找最晚入职员工的所有信息

CREATE TABLE `employees` (

`emp_no` int(11) NOT NULL,

`birth_date` date NOT NULL,

`first_name` varchar(14) NOT NULL,

`last_name` varchar(16) NOT NULL,

`gender` char(1) NOT NULL,

`hire_date` date NOT NULL,

PRIMARY KEY (`emp_no`));

```
SELECT *
FROM employees
WHERE hire_date = (select max(hire_date) from employees);--晚入职的当天未必就一个人，也许有多人，使用排序并限制得只能取得指定数量的结果
```

查找入职员工时间排名倒数第三的员工所有信息

CREATE TABLE `employees` (

`emp_no` int(11) NOT NULL,

`birth_date` date NOT NULL,

`first_name` varchar(14) NOT NULL,

`last_name` varchar(16) NOT NULL,

`gender` char(1) NOT NULL,

`hire_date` date NOT NULL,

PRIMARY KEY (`emp_no`));

emp_no	birth_date	first_name	last_name	gender	hire_date

```
SELECT *
FROM employees
WHERE hire_date=(
    SELECT distinct  hire_date
    FROM employees
    ORDER BY hire_date DESC
    LIMIT 2,1);
--#distinct 是去重复的
--#desc 倒序
--LIMIT 2,1 是从0开始的，第三个即是2；如果给定两个参数，第一个参数指定第一个返回记录行的偏移量，第二个参数指定返回记录行的最大数目。
```

查找各个部门当前(to_date='9999-01-01')领导当前薪水详情以及其对应部门编号dept_no

CREATE TABLE `dept_manager` (

`dept_no` char(4) NOT NULL,

`emp_no` int(11) NOT NULL,

`from_date` date NOT NULL,

`to_date` date NOT NULL,

PRIMARY KEY (`emp_no`,`dept_no`));

CREATE TABLE `salaries` (

`emp_no` int(11) NOT NULL,

`salary` int(11) NOT NULL,

`from_date` date NOT NULL,

`to_date` date NOT NULL,

PRIMARY KEY (`emp_no`,`from_date`));

emp_no	salary	from_date	to_date	dept_no

```
SELECT s.*, d.dept_no
FROM salaries s ，dept_manager d  --别名
WHERE s.to_date='9999-01-01'--salaries.to_date是经理来到这个公司的日期，还是有点区别的
AND d.to_date='9999-01-01'--dept_manager.to_date是经理来到这个部门时候的日期，
AND s.emp_no = d.emp_no;

/*
方法二
select s.*,d.dept_no
from salaries s
left/inner/right join dept_manager d
on d.emp_no=s.emp_no
where s.to_date='9999-01-01'
and d.to_date='9999-01-01';
*/
```

查找所有已经分配部门的员工的last_name和first_name

```
CREATE TABLE `dept_emp` (
`emp_no` int(11) NOT NULL,
`dept_no` char(4) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
CREATE TABLE `employees` (
`emp_no` int(11) NOT NULL,
`birth_date` date NOT NULL,
`first_name` varchar(14) NOT NULL,
`last_name` varchar(16) NOT NULL,
`gender` char(1) NOT NULL,
`hire_date` date NOT NULL,
PRIMARY KEY (`emp_no`));
```

last_name	first_name	dept_no

```
/*
SELECT employees.last_name, first_name, dept_emp.dept_no
FROM dept_emp inner join employees --内连接查询
ON  dept_emp.emp_no = employees.emp_no;
*/


SELECT employees.last_name,employees.first_name,dept_emp.dept_no
FROM dept_emp,employees
```

```
where dept_emp.emp_no=employees.emp_no;
```

查找所有员工的last_name和first_name以及对应部门编号dept_no，也包括展示没有分配具体部门的员工

```
CREATE TABLE `dept_emp` (
`emp_no` int(11) NOT NULL,
`dept_no` char(4) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
CREATE TABLE `employees` (
`emp_no` int(11) NOT NULL,
`birth_date` date NOT NULL,
`first_name` varchar(14) NOT NULL,
`last_name` varchar(16) NOT NULL,
`gender` char(1) NOT NULL,
`hire_date` date NOT NULL,
PRIMARY KEY (`emp_no`));
```

last_name	first_name	dept_no

```
SELECT employees.last_name,employees.first_name,dept_emp.dept_no
FROM employees LEFT JOIN dept_emp
ON employees.emp_no = dept_emp.emp_no;
```

--INNER JOIN 两边表同时有对应的数据，即任何一边缺失数据就不显示。

--LEFT JOIN 会读取左边数据表的全部数据，即便右边表无对应数据。

--RIGHT JOIN 会读取右边数据表的全部数据，即便左边表无对应数据。

/*

注意on与where有什么区别，两个表连接时用on，在使用left jion时，on和where条件的区别如下：

1、 on条件是在生成临时表时使用的条件，**它不管on中的条件是否为真，都会返回左边表中的记录。**

2、where条件是在临时表生成好后，再对临时表进行过滤的条件。**这时已经没有left join的含义**（必须返回左边表的记录）了，条件不为真的就全部过滤掉。

*/

查找所有员工入职时候的薪水情况，给出emp_no以及salary， 并按照emp_no进行逆序

```
CREATE TABLE `employees` (
`emp_no` int(11) NOT NULL,
`birth_date` date NOT NULL,
`first_name` varchar(14) NOT NULL,
`last_name` varchar(16) NOT NULL,
`gender` char(1) NOT NULL,
`hire_date` date NOT NULL,
PRIMARY KEY (`emp_no`));
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
```

```
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`from_date`));
```

emp_no	salary

```
SELECT e.emp_no,s.salary
FROM employees AS e,salaries AS s
WHERE e.emp_no = s.emp_no       --需先找到employees.emp_no在salaries表中对应的记录salaries.emp_no
AND e.hire_date = s.from_date   --入职时候salaries.from_date 和employees.hire_date的值应该要相等
ORDER BY e.emp_no DESC;         --按照emp_no值逆序排列
```

查找薪水涨幅超过15次的员工号emp_no以及其对应的涨幅次数t

```
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`from_date`));
```

emp_no	t

```
SELECT s.emp_no,COUNT(emp_no) AS t
FROM salaries AS s
GROUP BY emp_no    --用COUNT()函数和GROUP BY语句可以统计同一emp_no值的记录条数
HAVING t > 15;--由于WHERE后不可跟COUNT()函数，故用HAVING语句来限定t>15的条件
```

```
--GROUP BY应该发生在SELECT之后，你想想，都还没选择完，怎么对结果筛选?
-- GROUP BY 进行分组，然后再根据分组条件用 SELECT 选择出对应记录，最后再由 HAVING 的限制条件显示出 t>15 的记录。

/* 注意： 严格来说，下一条salary高于本条才算涨幅，但本题只要出现了一条记录就算一次涨幅，
salary相同可以理解为涨幅为0，salary变少理解为涨幅为负 */
```

找出所有员工当前(to_date='9999-01-01')具体的薪水salary情况，对于相同的薪水只显示一次,并按照逆序显示

```
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`from_date`));
```

salary

```
SELECT s.salary
FROM salaries AS s
WHERE s.to_date='9999-01-01'--当前(to_date='9999-01-01')
GROUP BY s.salary  --分组，可以去重
order by s.salary desc;--对SELECT的结果排序，要求逆序排列，则在最后应使用ORDER BY salary DESC

--慢些
/*
SELECT DISTINCT s.salary --相同薪水显示一次，则使用SELECT DISTINCT可去除重复值
```

```
FROM salaries AS s
WHERE s.to_date='9999-01-01'
--GROUP BY s.salary
order by s.salary desc;*/
```

获取所有部门当前manager的当前薪水情况，给出dept_no, emp_no以及salary，当前表示to_date='9999-01-01'

```
CREATE TABLE `dept_manager` (
`dept_no` char(4) NOT NULL,
`emp_no` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`from_date`));
```

dept_no	emp_no	salary

```
SELECT d.dept_no, d.emp_no, s.salary
FROM salaries AS s INNER JOIN dept_manager AS d --先用INNER JOIN连接两张表，限制条件是两张表的emp_no相同
ON d.emp_no = s.emp_no
AND d.to_date = '9999-01-01'
```

```
AND s.to_date = '9999-01-01'
/*
要获取当前manager的当前salary情况，再加上限制条件d.to_date = '9999-01-01' AND s.to_date = '9999-01-01'即可
（因为同一emp_no在salaries表中对应多条涨薪记录，而当s.to_date = '9999-01-01'时是该员工当前的薪水记录）*/
```

获取所有非manager的员工emp_no

```
CREATE TABLE `dept_manager` (
`dept_no` char(4) NOT NULL,
`emp_no` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
CREATE TABLE `employees` (
`emp_no` int(11) NOT NULL,
`birth_date` date NOT NULL,
`first_name` varchar(14) NOT NULL,
`last_name` varchar(16) NOT NULL,
`gender` char(1) NOT NULL,
`hire_date` date NOT NULL,
PRIMARY KEY (`emp_no`));
```

emp_no

```
--方法一：使用NOT IN选出在employees但不在dept_manager中的emp_no记录
SELECT e.emp_no
FROM employees AS e
```

```
WHERE e.emp_no NOT IN (SELECT emp_no FROM dept_manager);


--先使用LEFT JOIN连接两张表，再从此表中选出dept_no值为NULL对应的emp_no记录
/*
SELECT e.emp_no
FROM employees AS e LEFT JOIN dept_manager AS d
ON e.emp_no == d.emp_no
WHERE dept_no IS NULL;*/
```

获取所有员工当前的manager，如果当前的manager是自己的话结果不显示，当前表示to_date='9999-01-01'。

结果第一列给出当前员工的emp_no,第二列给出其manager对应的manager_no。

```
CREATE TABLE `dept_emp` (
`emp_no` int(11) NOT NULL,
`dept_no` char(4) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
CREATE TABLE `dept_manager` (
`dept_no` char(4) NOT NULL,
`emp_no` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
```

emp_no	manager_no

```
SELECT de.emp_no,dm.emp_no AS manager_no
FROM dept_emp AS de INNER JOIN dept_manager AS dm
ON de.dept_no = dm.dept_no    --用INNER JOIN连接两张表，因为要输出自己的经理，得知自己与经理的部门要相同
WHERE dm.to_date = '9999-01-01' --再用WHERE限制当前员工与当前经理的条件,当前的manager,所以最后的to_date要使用manager表中的字段
AND de.to_date = '9999-01-01'
AND de.emp_no <> dm.emp_no --部门一定有manager，员工不一定有manager,如果manager是自己，那么不显示。
```

获取所有部门中当前员工薪水最高的相关信息，给出dept_no, emp_no以及其对应的salary

```
CREATE TABLE `dept_emp` (
`emp_no` int(11) NOT NULL,
`dept_no` char(4) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`dept_no`));
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
```

PRIMARY KEY (`emp_no`,`from_date`));

dept_no	emp_no	salary

1、先用INNER JOIN连接两张表，**限制条件是两张表的emp_no相同**，即d.emp_no = s.emp_no；

2、选取每个员工当前的工资水平，用d.to_date = '9999-01-01' AND s.to_date = '9999-01-01'作条件限制，因为此表中每条最新记录的 to_date 都用 9999-01-01 表示；

3、用GROUP BY d.dept_no**将每个部门分为一组，用MAX()函数选取每组中工资最高者；**

4、将salaries用s代替，dept_emp用d代替，最后将MAX(s.salary)用salary代替后输出。

```
SELECT d.dept_no, s.emp_no, MAX(s.salary) AS salary
FROM salaries AS s INNER JOIN dept_emp As d
ON d.emp_no = s.emp_no
WHERE d.to_date = '9999-01-01' AND s.to_date = '9999-01-01'
GROUP BY d.dept_no
```

从titles表获取按照title进行分组，每组个数大于等于2，给出title以及对应的数目t。

CREATE TABLE IF NOT EXISTS "titles" (

`emp_no` int(11) NOT NULL,

`title` varchar(50) NOT NULL,

`from_date` date NOT NULL,

`to_date` date DEFAULT NULL);

title	t

此题应注意以下三点：

1、用COUNT()函数和GROUP BY语句可以统计同一title值的记录条数

2、根据题意，输出每个title的个数为t，故用AS语句将COUNT(title)的值转换为t

3、由于WHERE后不可跟COUNT()函数，故用**HAVING语句来限定t>=2的条件**

```
SELECT title, COUNT(title) AS t
FROM titles
GROUP BY title
HAVING t >= 2
```

从titles表获取按照title进行分组，每组个数大于等于2，给出title以及对应的数目t。

注意对于重复的emp_no进行忽略。

```
CREATE TABLE IF NOT EXISTS "titles" (
`emp_no` int(11) NOT NULL,
`title` varchar(50) NOT NULL,
`from_date` date NOT NULL,
`to_date` date DEFAULT NULL);
```

title	t

1、先用GROUP BY title将表格**以title分组**，再用COUNT(DISTINCT emp_no)可以统计同一title值且不包含重复emp_no值的记录条数

2、根据题意，输出每个title的个数为t，故用AS语句将COUNT(DISTINCT emp_no)的值转换为t

3、由于WHERE后不可跟COUNT()函数，故用HAVING语句来限定t>=2的条件

```
SELECT title, COUNT(DISTINCT emp_no) AS t
FROM titles
GROUP BY title
HAVING t >= 2
```

查找employees表所有emp_no为奇数，且last_name不为Mary的员工信息，并按照hire_date逆序排列

CREATE TABLE `employees` (

`emp_no` int(11) NOT NULL,

`birth_date` date NOT NULL,

`first_name` varchar(14) NOT NULL,

`last_name` varchar(16) NOT NULL,

`gender` char(1) NOT NULL,

`hire_date` date NOT NULL,

PRIMARY KEY (`emp_no`));

emp_no	birth_date	first_name	last_name	gender	hire_date

三点需要注意：

1、员工号为奇数，则emp_no取余应为1

2、last_name不为Mary，用‘！=’表示

3.根据hire_date逆序排列，用desc

```
SELECT *
FROM employees
WHERE emp_no % 2 = 1    --emp_no为奇数
AND last_name != 'Mary' –last_name不为Mary的员工信息
ORDER BY hire_date DESC –按照hire_date逆序排列
```

统计出当前各个title类型对应的员工当前薪水对应的平均工资。结果给出title以及平均工资avg。

```
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`from_date`));
CREATE TABLE IF NOT EXISTS "titles" (
`emp_no` int(11) NOT NULL,
`title` varchar(50) NOT NULL,
`from_date` date NOT NULL,
`to_date` date DEFAULT NULL);
```

title	avg

```
SELECT title,AVG(salary)
FROM titles INNER JOIN salaries
ON titles.emp_no = salaries.emp_no
WHERE titles.to_date = '9999-01-01'    -- 工资是会变化的， to_date='9999-0101'表示的是当前的工资
AND salaries.to_date = '9999-01-01'
GROUP BY title


--where 是建表前先筛选，having是建表后再判断
--where要求必须在group by 前面..意思是先过滤再分组   而having是必须在group by后面连用 是分组后的过滤
```

获取当前（to_date='9999-01-01'）薪水第二多的员工的emp_no以及其对应的薪水salary

```
CREATE TABLE `salaries` (
`emp_no` int(11) NOT NULL,
`salary` int(11) NOT NULL,
`from_date` date NOT NULL,
`to_date` date NOT NULL,
PRIMARY KEY (`emp_no`,`from_date`));
```

emp_no	salary

```
SELECT emp_no,salary
FROM salaries
WHERE to_date='9999-01-01'  --当前（to_date='9999-01-01'）
ORDER BY salary desc  --逆序
LIMIT 1,1;  --需要注意的就是limit 1,1表示从第二个数开始取，取一个，即文中要求的第二个人
```