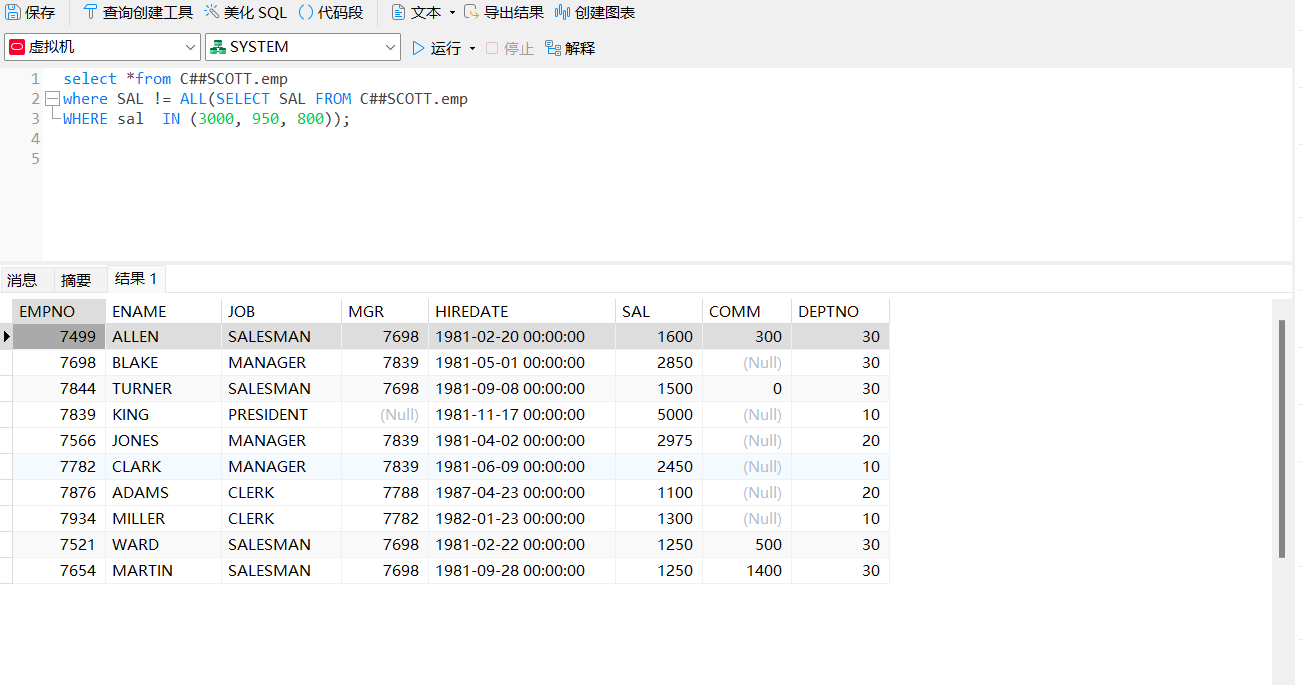
前提：scott.sql文件中的数据导入到了数据库中。

# **练习1：在scott用户下(12c以上版本用户名为c##scott)，执行下述操作。**

1 在emp中，使用all关键字过滤工资（sal）同时不等于3000、950和800的员工记录。



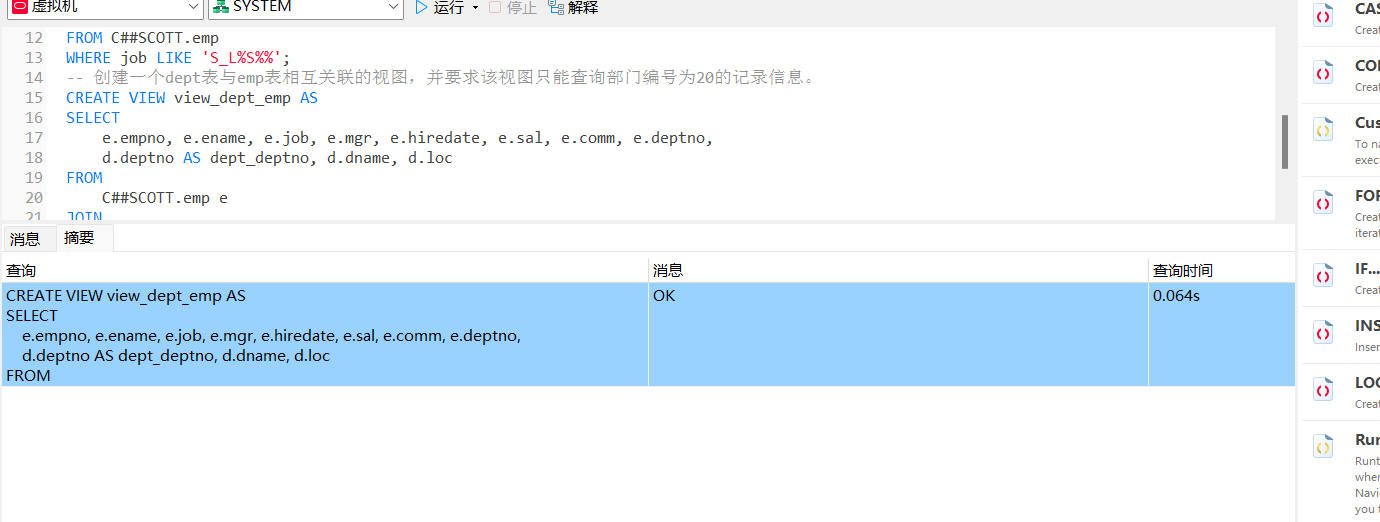
2 使用LIKE关键字，在emp表中，要显示在1981年雇佣的所有员工的信息。



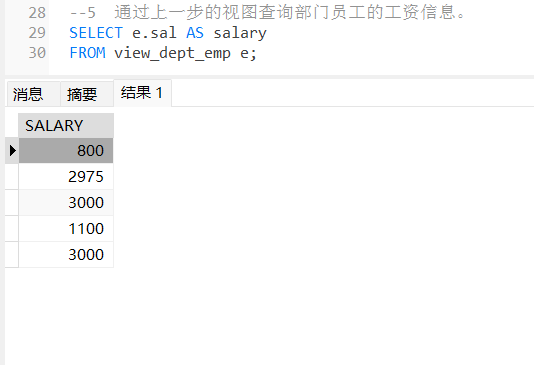
3 在emp表中，查询工作是SALESMAN的员工姓名，但是不记得SALESMAN的准确拼写，但还记得它的第1个字符是S，第3个字符是L，第5个字符为S。



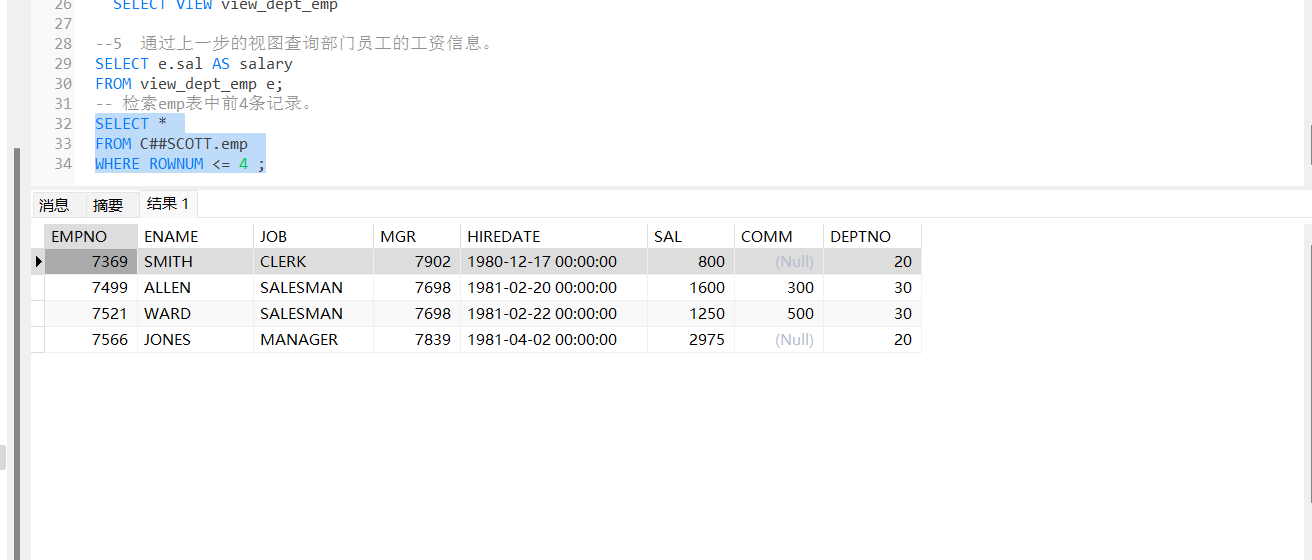
4 创建一个dept表与emp表相互关联的视图，并要求该视图只能查询部门编号为20的记录信息。



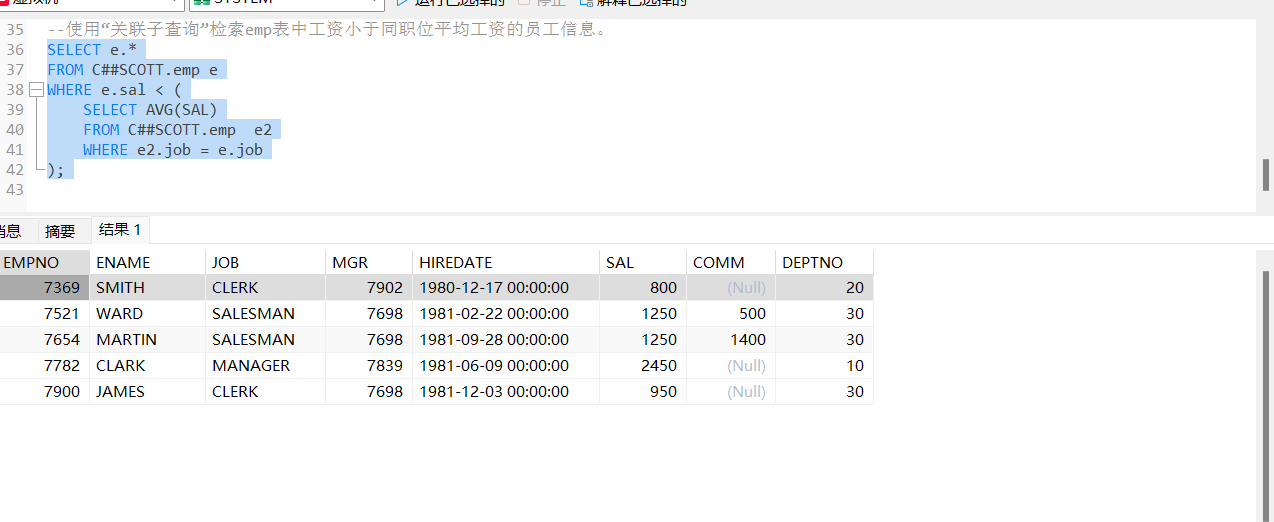
5 通过上一步的视图查询部门员工的工资信息。



6 检索emp表中前4条记录。



7 使用“关联子查询”检索emp表中工资小于同职位平均工资的员工信息。

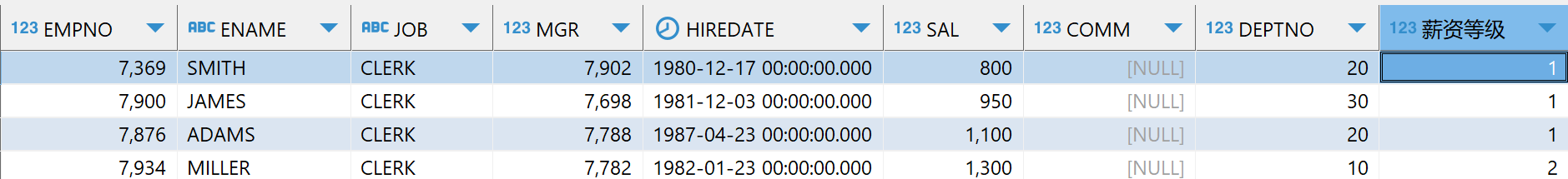


# **练习2：多个表连接查询主要有内连接、外连接、交叉连接。**

|  |  |
| --- | --- |
| **使用scott用户登录系统。(12c以上版本用户名为c##scott)**  **insert into scott.emp(empno,ename,job) values(9527,'EAST','SALESMAN');**    **执行上述语句向员工表中插入一条数据，若失败请自行调整字段值。** | |
| SELECT e.\*,d.\* FROM emp e  INNER JOIN dept d ON e.DEPTNO = d.DEPTNO; | 语句的作用：  表示将 emp 表与 dept 表通过 DEPTNO 字段进行内连接。即只返回两个表中 DEPTNO 相等的记录。  执行截图： |
| SELECT e.\*,d.\* FROM emp e  LEFT JOIN dept d ON e.DEPTNO = d.DEPTNO; | 语句的作用：  这种类型的联接确保了来自左侧表（在这里是C##SCOTT.emp表）的所有记录都会出现在结果集中，即使右侧表中没有匹配的数据也是如此。  执行截图： |
| SELECT e.\*,d.\* FROM emp e  RIGHT JOIN dept d ON e.DEPTNO = d.DEPTNO; | 语句的作用：  返回所有在C##SCOTT.dept表中存在的部门记录，以及与这些部门相关的员工信息。如果某个部门没有对应的员工，则C##SCOTT.emp表中的字段将显示为NULL。  执行截图： |
| SELECT e.\*,d.\* FROM emp e  FULL OUTER JOIN dept d ON e.DEPTNO = d.DEPTNO; | 语句的作用：   1. 获取所有记录：FULL OUTER JOIN会返回两个表中所有的记录，包括那些在一方有数据而另一方没有对应数据的行。 2. 合并员工与部门信息：该查询将显示所有员工的信息（由前缀e.\*表示），以及对应的部门信息（由前缀d.\*表示）。如果某个员工所属的部门编号(DEPTNO)在dept表中有相应的条目，则这两部分信息会被关联起来；如果没有找到匹配项，则缺失的部分将以NULL填充。   执行截图： |
| SELECT e.\*,d.\* FROM emp e  CROSS JOIN dept d; | 语句的作用：   1. 生成笛卡尔积：对于emp表中的每一行记录，都会与dept表中的每一条记录相结合产生一个新的结果行。例如，如果emp表有N条记录，而dept表有M条记录，则最终产生的结果集会有N \* M条记录。 2. 不考虑任何关联条件：不同于其他类型的JOIN操作（如INNER JOIN、LEFT JOIN等），这里并没有指定任何用于匹配的条件。因此，无论两表之间是否存在实际关系或是否有必要将它们结合在一起，查询都将执行并生成上述所述的所有可能组合。   执行截图： |
| 比较上述5条语句执行完毕后，执行结果有什么区别？  这五种类型的连接（内连接、左外连接、右外连接、全外连接和交叉连接）在处理两个表之间的关系时有不同的行为：  内连接 (INNER JOIN)：这种连接方式只返回两个表中满足连接条件的记录。例如，在您的示例中，只有当 emp 表中的 DEPTNO 与 dept 表中的 DEPTNO 相匹配时，才会显示这些行。  左外连接 (LEFT JOIN)：此类型连接会返回所有左边表 (emp) 的记录，并且如果右边表 (dept) 中有匹配项，则也会包括那些匹配项；如果没有匹配项，则用 NULL 填充右边表的所有字段。这意味着即使部门没有员工，也会列出该部门的信息。  右外连接 (RIGHT JOIN)：类似地，右外连接将返回所有右边表 (dept) 的记录以及任何匹配的左边表 (emp) 记录。如果不匹配则使用 NULL 来填充左边表的所有字段。也就是说，即便某个部门没有任何员工信息，其部门详情仍然会被列出。  全外连接 (FULL OUTER JOIN)：这种连接将会结合左外连结和右外连结的结果集，即同时包含左右两边所有的数据。对于不匹配的部分同样以 NULL 进行填充。因此，无论哪个表中有无对应的数据都会被展示出来。  交叉连接 (CROSS JOIN)：不同于上述基于特定列进行比对的方式，交叉连接会产生第一个表中每一行与其他第二个表中每一行组合而成的新行。最终结果将是两表行数相乘后的数量级增长。具体到这个例子来说，每个雇员都将与每一个部门生成一条新的记录，不论实际是否有关联。  综上所述，不同类型的连接根据需求选择了不同的数据呈现方式，从仅选择共同部分到全面覆盖双方数据均有涉及。 | |

**练习3：高级查询(12c以上版本用户名为c##scott)**

**1.SCOTT.SALGRADE表中记录了员工的薪资等级。SCOTT.EMP表和SCOTT.SALGRADE 联合起来，并且展示出每个人的基本信息和薪资等级(如下图所示)。**



提示：

|  |  |
| --- | --- |
| CASE expression  WHEN value1 THEN result1  WHEN value2 THEN result2  ...  ELSE default\_result  END | CASE语句是一种条件逻辑结构，类似于其他编程语言中的switch或if-else语句。它可以用于查询中根据不同条件返回不同的值。  两张表联合查询时，连接条件除了可以用等于e.DEPTNO = d.DEPTNO，还可以用e.sal BETWEEN s.losal AND s.hisal |



**2.在员工表emp中，按照deptno分组，找到薪资大于2000的员工，人员大于两个的部门。输出部门编号deptno和人员数量。输出信息示例：**

****

