

第4章 数据利器——SQL

本章通过实例的方法引导读者快速掌握 SQL 的使用，从而能够利用【SQL Plus Worksheet】等工具，使用标准 SQL 语言完成对数据库数据的日常管理工作。

4.1 节介绍 SQL 的概念、主要特点、使用 SQL 的工具、SQL 如何访问数据表以及本书实例所使用的两个数据表。

4.2 节介绍对单个数据表进行的查询操作。

4.3 节介绍如何对多个数据表同时进行查询操作。

4.4 节介绍稍复杂的嵌套查询。

4.5 节介绍了最常用的查询函数。

4.6 节介绍如何使用 SQL 录入数据。

4.7 节介绍了如何使用 SQL 删除数据。

4.8 节介绍如何使用 SQL 更新数据。

4.1 SQL 概述

4.1.1 SQL 是什么

SQL (Structured Query Language, 译为结构化查询语言) 在关系型数据库中的地位就犹如英语在世界上的地位。它是数据库系统的通用语言，利用它，用户可以用几乎同样的语句在不同的数据库系统上执行同样的操作。比如“select * from 数据表名”代表要从某个数据表中取出全部数据，在 Oracle 9i、SQL Server 2000、Foxpro 等关系型数据库中都可以使用这条语句。SQL 已经被 ANSI (美国国家标准化组织) 确定为数据库系统的工业标准。

SQL 语言按照功能可以分为 4 大类。

- ❑ 数据查询语言 DQL: 查询数据。
- ❑ 数据定义语言 DDL: 建立、删除和修改数据对象。
- ❑ 数据操纵语言 DML: 完成数据操作的命令，包括查询。
- ❑ 数据控制语言 DCL: 控制对数据库的访问，服务器的关闭、启动等。

4.1.2 SQL 的主要特点

SQL 语言简单易学、风格统一，利用简单的几个英语单词的组合就可以完成所有的功能。在 SQLPlus Worksheet 环境下可以单独使用的 SQL 语句，几乎可以不加修改地嵌入到如 VB、PB 这样的前端开发平台上，利用前端工具的计算能力和 SQL 的数据库操纵能力，可以快速建立数据库应用程序。

4.1.3 Oracle 9i 使用 SQL 的工具

在 Oracle 9i 中为使用 SQL 语言提供了两个主要的工具。

❑ 【SQL Plus】

❑ 【SQLPlus Worksheet】

两种工具在使用上功能都相同，但在可操作性上，【SQLPlus Worksheet】更适合初学者。因此，本书重点介绍后者的使用。

4.1.4 SQL 中访问数据表的方法

在 SQL 语言中访问数据表是通过“用户名.数据表”的形式来进行的。

比如在 Oracle 9i 数据库服务器安装过程中，默认建立有 scott 用户，该用户对 dept 数据表和 emp 数据表有数据查询的权限，因此访问数据表的语句为 `select * from scott.emp`。当然，如果用户是用 scott 用户本身登录的，则访问数据表的语句可以简化为 `select * from emp`，实质是一样的。

在本章的实例中，我们以数据库系统管理员 system、口令 manager 登录数据库，访问数据必须采用 `select * from scott.emp` 的形式。即使是用户本身登录后访问属于自己的数据表，我们也推荐使用“用户名.数据表”的形式来访问数据表以清楚地反映数据表的有权用户信息。

4.1.5 两个范例数据表

在读者没有学习如何创建数据表，如何创建用户，如何将用户赋予对数据表的访问权限之前，我们以数据库已经建立的两个范例数据表为例来介绍。

(1) 启动【SQLPlus Worksheet】

(2) 在【命令编辑区】输入语句“`desc scott.emp`”，然后单击【执行】按钮，出现如图 4.1 所示的 emp 数据表结构。

【参见光盘文件】：\第 4 章\4.1\415-1.sql。



图 4.1 scott.emp 数据表结构



desc, describe 命令的简化形式，作用是显示数据表的结构。使用形式：“desc 数据表名”。

(3) 在【命令编辑区】输入“desc scott.dept”，然后单击【执行】按钮，出现如图 4.2 所示的 scott.dept 数据表结构。

【参见光盘文件】：\第 4 章\4.1\415-2.sql。

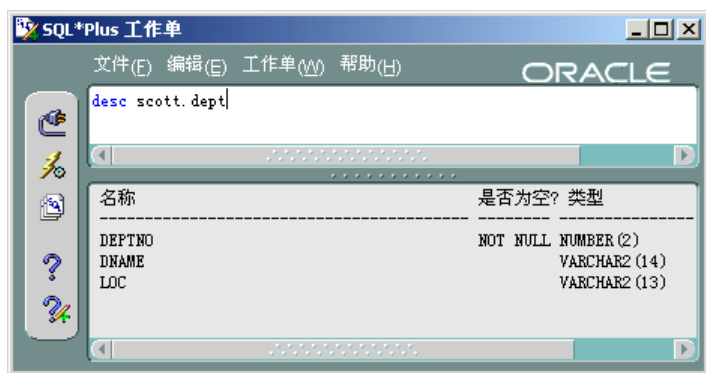


图 4.2 scott.dept 数据表结构

接下来我们以实际查询例子来介绍数据查询的语法，读者可以参照配套光盘的实例跟随本书执行同样的操作。

4.2 用 SQL 进行单表查询

单表查询是相对多表查询而言的，指从一个数据表中查询数据。

4.2.1 查询所有的记录

在【命令编辑区】执行输入“select * from scott.emp”，然后单击【执行】按钮，出现如图 4.3 所示的 emp 数据表所有记录。

【参见光盘文件】：\第 4 章\4.2\421.sql。

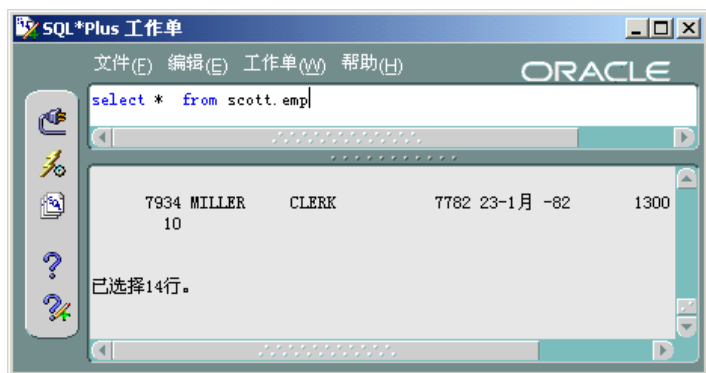


图 4.3 查询 emp 数据表所有记录



select * from 数据表，这里的“*”代表数据表中所有的字段。

4.2.2 查询所有记录的某些字段

在【命令编辑区】输入“select empno,ename,job from scott.emp”，然后单击【执行】按钮，将显示 emp 数据表的 empno、ename 和 job 字段，如图 4.4 所示。

【参见光盘文件】：\第 4 章\4.2\422.sql。



The screenshot shows the SQL*Plus interface with the query 'select empno, ename, job from scott.emp' entered in the command editor. The result is displayed in a table with three columns: EMPNO, ENAME, and JOB. The data is as follows:

EMPNO	ENAME	JOB
7369	SMITH	CLERK
7499	ALLEN	SALESMAN
7521	WARD	SALESMAN
7566	JONES	MANAGER
7654	MARTIN	SALESMAN
7698	BLAKE	MANAGER
7782	CLARK	MANAGER
7788	SCOTT	ANALYST
7839	KING	PRESIDENT

图 4.4 查询 emp 数据表的 empno、ename 和 job 字段

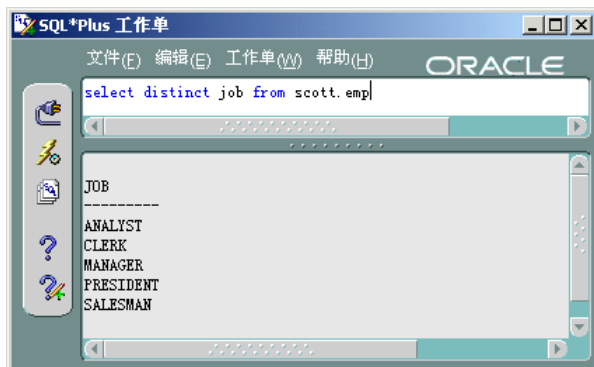


select 字段名 1, 字段名 2,..... from 数据表, 将显示某些特定的字段, 注意这里的字段名之间的逗号是英文状态下的逗号。

4.2.3 查询某些字段不同记录

在图 4.4 所示的 job 字段中, 可以发现相同的数据, 为了查询有多少种不同的 job, 在【命令编辑区】输入“select distinct job from scott.emp”, 然后单击【执行】按钮, 出现如图 4.5 所示的结果。

【参见光盘文件】：\第 4 章\4.2\423.sql。



The screenshot shows the SQL*Plus interface with the query 'select distinct job from scott.emp' entered in the command editor. The result is displayed in a table with one column: JOB. The data is as follows:

JOB
ANALYST
CLERK
MANAGER
PRESIDENT
SALESMAN

图 4.5 查询不同的 job 字段



select distinct 字段名 from 数据表, 这里的“distinct”保留字指在显示时去除相同的记录, 与之对应的是“all”将保留相同的记录, 默认为“all”。

4.2.4 单条件的查询

(1) 在【命令编辑区】输入“select empno,ename,job from scott.emp where job='MANAGER'”，然后单击【执行】按钮，出现如图 4.6 所示的字符型字段条件查询的结果，查询的是 job 为 MANAGER 的记录。

【参见光盘文件】：\第 4 章\4.2\424-1.sql。

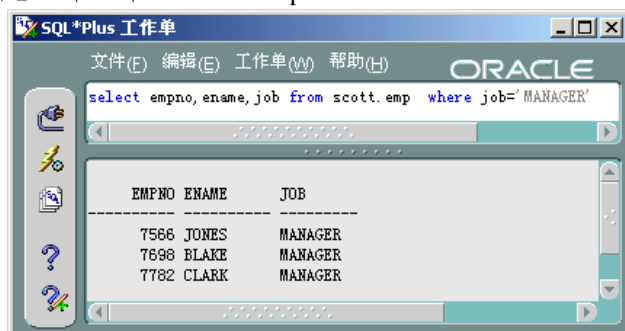


图 4.6 查询满足字符型字段条件的记录

(2) 在【命令编辑区】输入“select empno,ename,sal from scott.emp where sal<=2500”，然后单击【执行】按钮，出现如图 4.7 所示的数字型字段条件查询的结果，查询的是满足 sal 小于等于 2500 的记录。

【参见光盘文件】：\第 4 章\4.2\424-2.sql。

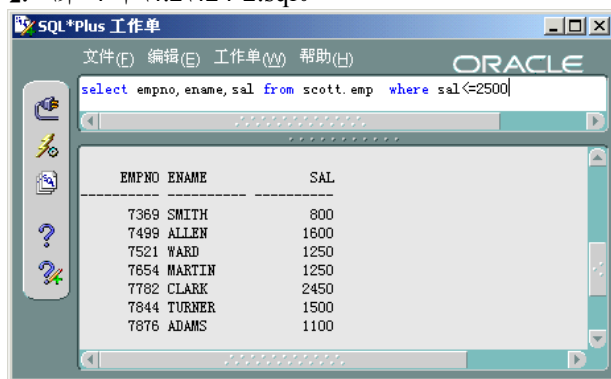


图 4.7 查询满足数字型字段条件的记录



where 可以指定查询条件，如果是指定字符型字段查询条件，形式为字段名 运算符 '字符串'；如果是指定数字型字段查询条件，形式为字段名 运算符 '字符串'。单条件查询使用的比较运算符如表 4.1 所示。

【参见光盘文件】：\第 4 章\4.2\table41.sql。

表 4.1

比较运算符

名称	实例
=(等于)	select * from scott.emp where job='MANAGER';
	select * from scott.emp where sal=1100;

续表

名称	实例
!= (不等于)	select * from scott.emp where job!='MANAGER';
	select * from scott.emp where sal!=1100;
^=(不等于)	select * from scott.emp where job^='MANAGER';
	select * from scott.emp where sal^=1100;
◇(不等于)	select * from scott.emp where job◇'MANAGER';
	select * from scott.emp where sal◇1100;
<(小于)	select * from scott.emp where sal<2000;
	select * from scott.emp where job<'MANAGER';
>(大于)	select * from scott.emp where sal>2000;
	select * from scott.emp where job>'MANAGER';
<=(小于等于)	select * from scott.emp where sal<=2000;
	select * from scott.emp where job<='MANAGER';
>=(大于等于)	select * from scott.emp where sal>=2000;
	select * from scott.emp where job>='MANAGER';
in(列表)	select * from scott.emp where sal in (2000,1000,3000);
	select * from scott.emp where job in ('MANAGER','CLERK');
not in(不在列表)	select * from scott.emp where sal not in (2000,1000,3000);
	select * from scott.emp where job not in ('MANAGER','CLERK');
between(介于之间)	select * from scott.emp where sal between 2000 and 3000;
	select * from scott.emp where job between 'MANAGER' and 'CLERK';
not between (不介于之间)	select * from scott.emp where sal not between 2000 and 3000;
	select * from scott.emp where job not between 'MANAGER' and 'CLERK';
like(模式匹配)	select * from scott.emp where job like 'M%';
	select * from scott.emp where job like 'M__';
not like (模式不匹配)	select * from scott.emp where job not like 'M%';
	select * from scott.emp where job not like 'M__';
Is null (是否为空)	select * from scott.emp where sal is null;
	select * from scott.emp where job is null;
is not null(是否不为空)	select * from scott.emp where sal is not null;
	select * from scott.emp where job is not null;



like 和 not like 适合字符型字段的查询，%代表任意长度的字符串，_下划线代表一个任意的字符。like ‘m%’ 代表 m 开头的任意长度的字符串，like ‘m__’ 代表 m 开头的长度为 3 的字符串。

4.2.5 组合条件的查询

(1) 在【命令编辑区】输入 “select empno,ename,job from scott.emp where job>=‘CLERK’ and sal<=2000”，然后单击【执行】按钮，出现如图 4.8 所示的逻辑与组合查询的结果。

【参见光盘文件】：\第 4 章\4.2\425-1.sql。

SQL*Plus 工作单

文件(E) 编辑(E) 工作单(W) 帮助(H) ORACLE

```
select empno,ename,job,sal from scott.emp
where job>= 'CLERK' and sal<=2000
```

7499	ALLEN	SALESMAN	1800
7521	WARD	SALESMAN	1250
7654	MARTIN	SALESMAN	1250
7844	TURNER	SALESMAN	1500
7876	ADAMS	CLERK	1100
7900	JAMES	CLERK	950
7934	MILLER	CLERK	1300

已选择8行。

图 4.8 逻辑与组合查询

(2) 在【命令编辑区】输入 “select empno,ename,job from scott.emp where job>=‘CLERK’ or sal<=2000”，然后单击【执行】按钮，出现如图 4.9 所示的逻辑或组合查询的结果。

【参见光盘文件】：\第 4 章\4.2\425-2.sql。

SQL*Plus 工作单

文件(E) 编辑(E) 工作单(W) 帮助(H) ORACLE

```
select empno,ename,job,sal from scott.emp
where job>= 'CLERK' or sal<=2000
```

EMPNO	ENAME	JOB	SAL
7369	SMITH	CLERK	800
7499	ALLEN	SALESMAN	1800
7521	WARD	SALESMAN	1250
7566	JONES	MANAGER	2975
7654	MARTIN	SALESMAN	1250
7698	BLAKE	MANAGER	2850

图 4.9 逻辑或组合查询

(3) 在【命令编辑区】输入 “select empno,ename,job from scott.emp where not job=‘CLERK’”，然后单击【执行】按钮，出现如图 4.10 所示的逻辑非组合查询的结果。

【参见光盘文件】：\第 4 章\4.2\425-3.sql。

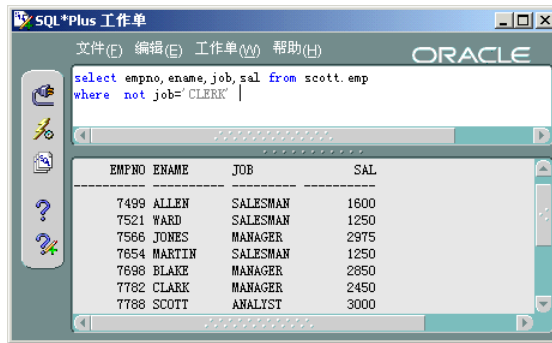


图 4.10 逻辑非组合查询



“not job='CLERK'” 等价于 “job<>'CLERK'”。

组合条件中使用的逻辑比较符如表 4.2 所示。

【参见光盘文件】: \第 4 章\4.2\table42.sql。

表 4.2 逻辑比较符

名称	实例
and(与)	select * from scott.emp where job='MANAGER' and sal<>2000;
or(或)	select * from scott.emp where job!='MANAGER' or sal<>2000;
not(非)	select * from scott.emp where not job>='MANAGER';

4.2.6 排序查询

在【命令编辑区】输入 “select empno,ename,job from scott.emp where job<='CLERK' order by job asc,sal desc”，然后单击【执行】按钮，出现如图 4.11 所示的排序查询的结果。

【参见光盘文件】: \第 4 章\4.2\426.sql。

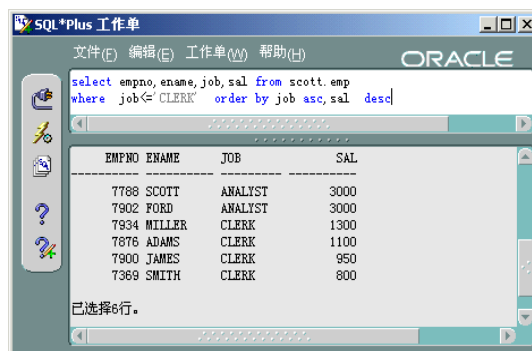


图 4.11 排序查询



order by 可以指定查询结果如何排序，形式为字段名 排序关键词；asc 代表升序排列，desc 代表降序排列，多个排序字段之间通过逗号分割。若有 where 查询条件，order

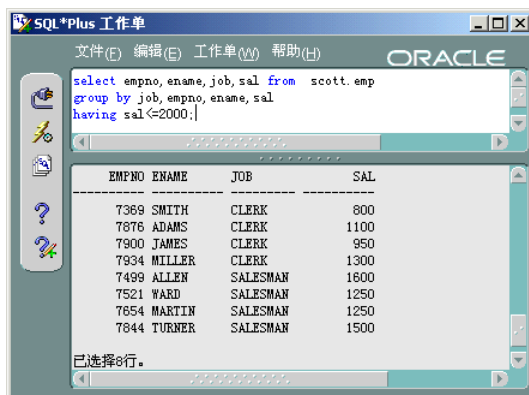
by 要放在 where 语句后面。

4.2.7 分组查询

分组查询是指将查询结果按照字段分组。

(1) 在【命令编辑区】输入“select empno,ename,job,sal from scott.emp group by job,empno,ename,sal having sal<=2000”，然后单击【执行】按钮，出现如图 4.12 所示的分组查询的结果。

【参见光盘文件】：\第4章\4.2\427-1.sql。



SQL*Plus 工作单

文件(F) 编辑(E) 工作单(W) 帮助(H) ORACLE

```
select empno,ename,job,sal from scott.emp
group by job,empno,ename,sal
having sal<=2000.
```

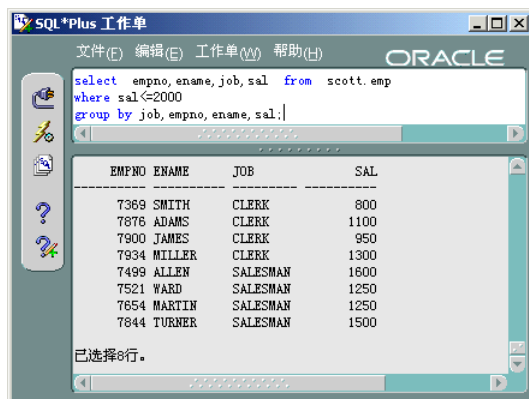
EMPNO	ENAME	JOB	SAL
7369	SMITH	CLERK	800
7876	ADAMS	CLERK	1100
7900	JAMES	CLERK	950
7934	MILLER	CLERK	1300
7499	ALLEN	SALESMAN	1600
7521	WARD	SALESMAN	1250
7654	MARTIN	SALESMAN	1250
7844	TURNER	SALESMAN	1500

已选择8行。

图 4.12 使用 having 子句的分组查询

(2) 在【命令编辑区】输入“select empno,ename,job,sal from scott.emp where sal<=2000 group by job,empno,ename,sal”，然后单击【执行】按钮，出现如图 4.13 所示的分组查询的结果。

【参见光盘文件】：\第4章\4.2\427-2.sql。



SQL*Plus 工作单

文件(F) 编辑(E) 工作单(W) 帮助(H) ORACLE

```
select empno,ename,job,sal from scott.emp
where sal<=2000
group by job,empno,ename,sal.
```

EMPNO	ENAME	JOB	SAL
7369	SMITH	CLERK	800
7876	ADAMS	CLERK	1100
7900	JAMES	CLERK	950
7934	MILLER	CLERK	1300
7499	ALLEN	SALESMAN	1600
7521	WARD	SALESMAN	1250
7654	MARTIN	SALESMAN	1250
7844	TURNER	SALESMAN	1500

已选择8行。

图 4.13 使用 where 子句的分组查询



where 检查每条记录是否符合条件，having 是检查分组后的各组是否满足条件。having 语句只能配合 group by 语句使用，没有 group by 时不能使用 having，但可以使用 where。

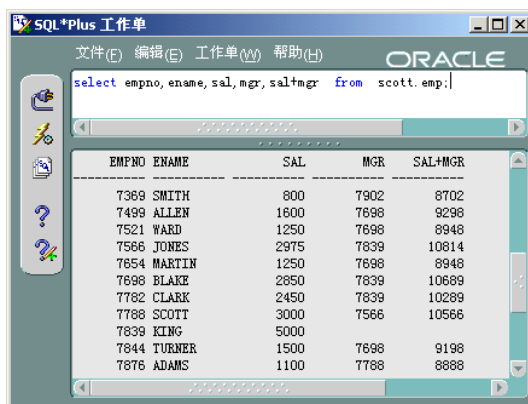
4.2.8 字段运算查询

可以利用几种基本的算术运算符来查询数据。

常见的+（加）、-（减）、*（乘）、/（除）4种算术运算都可以用来查询数据。

在【命令编辑区】输入“select empno,ename,sal,mgr,sal+mgr from scott.emp”，然后单击【执行】按钮，出现如图 4.14 所示的结果。

【参见光盘文件】：\第 4 章\4.2\428.sql。



SQL*Plus 工作单

文件(F) 编辑(E) 工作单(W) 帮助(H) ORACLE

select empno,ename,sal,mgr,sal+mgr from scott.emp;

EMPNO	ENAME	SAL	MGR	SAL+MGR
7369	SMITH	800	7902	8702
7499	ALLEN	1600	7698	9298
7521	WARD	1250	7698	8948
7566	JONES	2975	7839	10814
7654	MARTIN	1250	7698	8948
7698	BLAKE	2850	7839	10689
7782	CLARK	2450	7839	10289
7788	SCOTT	3000	7566	10566
7839	KING	5000		
7844	TURNER	1500	7698	9198
7876	ADAMS	1100	7788	8888

图 4.14 字段运算查询结果



利用算术运算符仅仅适合多个数值型字段或字段与数字之间的运算。

4.2.9 变换查询显示

在【命令编辑区】输入“select empno 编号,ename 姓名,job 工作,sal 薪水 from scott.emp”，然后单击【执行】按钮，出现如图 4.15 所示的结果，可以将默认的字段名以设定的名称显示。

【参见光盘文件】：\第 4 章\4.2\429.sql。



SQL*Plus 工作单

文件(F) 编辑(E) 工作单(W) 帮助(H) ORACLE

select empno 编号,ename 姓名,job 工作,sal 薪水
from scott.emp;

编号	姓名	工作	薪水
7369	SMITH	CLERK	800
7499	ALLEN	SALESMAN	1600
7521	WARD	SALESMAN	1250
7566	JONES	MANAGER	2975
7654	MARTIN	SALESMAN	1250
7698	BLAKE	MANAGER	2850
7782	CLARK	MANAGER	2450
7788	SCOTT	ANALYST	3000
7839	KING	PRESIDENT	5000

图 4.15 变换查询显示

以上我们学习了对单个数据表的查询语句。将上面这些基本的实例经过组合，就可以完

成基本的日常数据查询任务，接下来进一步学习多表查询。

4.3 用 SQL 进行多表查询

所谓多表查询是相对单表而言的，指从多个数据表中查询数据，这里我们主要学习从两个数据表中如何查询数据的方法。

4.3.1 无条件多表查询

无条件多表查询是将各表的记录以“笛卡尔”积的方式组合起来。

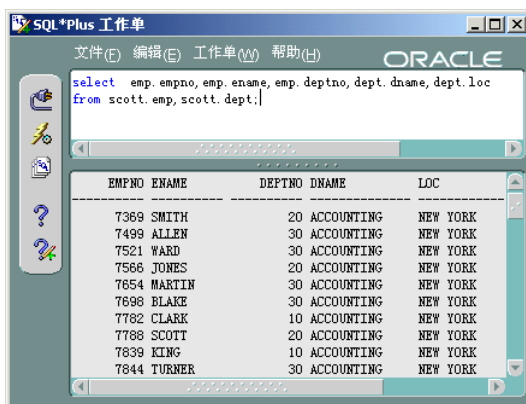
如 scott.dept 表共有 4 条记录，scott.emp 表共有 14 条记录，其“笛卡尔”积将有 $4 \times 14 = 56$ 条记录。

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.deptno, dept.dname, dept.loc
from scott.emp, scott.dept;
```

单击【执行】按钮，出现如图 4.16 所示的结果。

【参见光盘文件】：\第 4 章\4.3\431.sql。



EMPNO	ENAME	DEPTNO	DNAME	LOC
7369	SMITH	20	ACCOUNTING	NEW YORK
7499	ALLEN	30	ACCOUNTING	NEW YORK
7521	WARD	30	ACCOUNTING	NEW YORK
7566	JONES	20	ACCOUNTING	NEW YORK
7654	MARTIN	30	ACCOUNTING	NEW YORK
7698	BLAKE	30	ACCOUNTING	NEW YORK
7782	CLARK	10	ACCOUNTING	NEW YORK
7788	SCOTT	20	ACCOUNTING	NEW YORK
7839	KING	10	ACCOUNTING	NEW YORK
7844	TURNER	30	ACCOUNTING	NEW YORK

图 4.16 无条件多表查询

4.3.2 等值多表查询

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.deptno, dept.dname, dept.loc
from scott.emp, scott.dept
where scott.emp.deptno=scott.dept.deptno;
```

单击【执行】按钮，出现如图 4.17 所示的结果。

【参见光盘文件】: \第 4 章\4.3\432.sql。

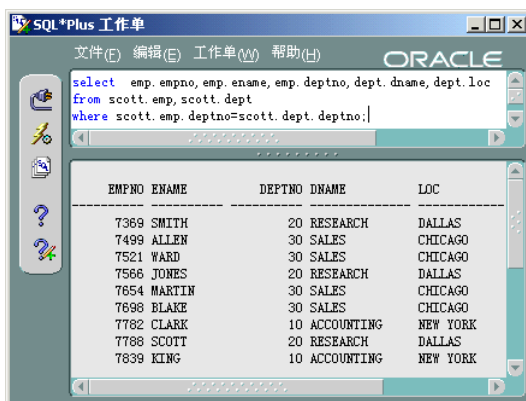


图 4.17 等值多表查询



等值多表查询将按照等值的条件查询多个数据表中关联的数据。要求关联的多个数据表的某些字段具有相同的属性，即具有相同的数据类型、宽度和取值范围。

4.3.3 非等值多表查询

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.deptno, dept.dname, dept.loc
from scott.emp, scott.dept
where scott.emp.deptno!=scott.dept.deptno and scott.emp.deptno=10;
```

单击【执行】按钮，出现如图 4.18 所示的结果。

【参见光盘文件】: \第 4 章\4.3\433.sql。



图 4.18 非等值多表查询

在非等值多表查询中，读者可以使用表 4.1 所示的比较运算符来组合查询条件。

4.4 用 SQL 进行嵌套查询

在 select 查询语句里可以嵌入 select 查询语句，称为嵌套查询。有些书上将内嵌的 select 语句称为子查询，子查询形成的结果又成为父查询的条件。



子查询可以嵌套多层，子查询操作的数据表可以是父查询不操作的数据表。子查询中不能有 order by 分组语句。

4.4.1 简单嵌套查询

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal >= (select sal from scott.emp where ename='WARD');
```

单击【执行】按钮，出现如图 4.19 所示的结果。

【参见光盘文件】：\第 4 章\4.4\441.sql。

EMPNO	ENAME	JOB	SAL
7499	ALLEN	SALESMAN	1600
7521	WARD	SALESMAN	1250
7566	JONES	MANAGER	2975
7654	MARTIN	SALESMAN	1250
7698	BLAKE	MANAGER	2850
7782	CLARK	MANAGER	2450
7788	SCOTT	ANALYST	3000
7839	KING	PRESIDENT	5000
7844	TURNER	SALESMAN	1500
7902	FORD	ANALYST	3000
7934	MILLER	CLERK	1300

图 4.19 简单嵌套查询

在这段代码中，子查询 select sal from scott.emp where ename='WARD' 的含义是从 emp 数据表中查询姓名为 WARD 的员工的薪水，父查询的含义是要找出 emp 数据表中薪水大于等于 WARD 的薪水的员工。上面的查询过程等价于两步的执行过程。

(1) 执行 “select sal from scott.emp where ename='WARD'”，得出 sal=1250；

(2) 执行 “select emp.empno, emp.ename, emp.job, emp.sal from scott.emp where sal >= 1250;”

4.4.2 带【in】的嵌套查询

在【命令编辑区】执行下列语句。

```

select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal in (select sal from scott.emp where ename='WARD');

```

单击【执行】按钮，出现如图 4.20 所示的结果。

【参见光盘文件】：\第 4 章\4.4\442.sql。

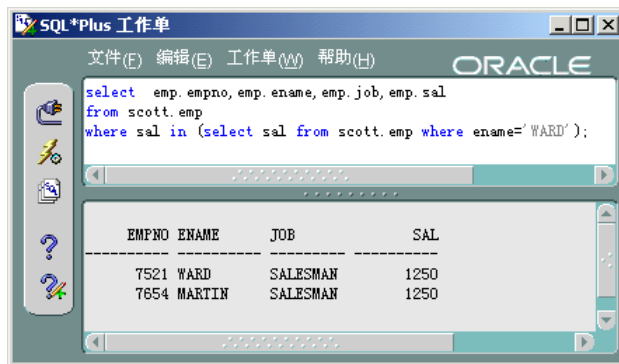


图 4.20 带【in】的嵌套查询

上述语句完成的是查询薪水和 WARD 相等的员工，也可以使用【not in】来进行查询。

4.4.3 带【any】的嵌套查询

在【命令编辑区】执行下列语句。

```

select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal > any(select sal from scott.emp where job='MANAGER');

```

单击【执行】按钮，出现如图 4.21 所示的结果。

【参见光盘文件】：\第 4 章\4.4\443.sql。

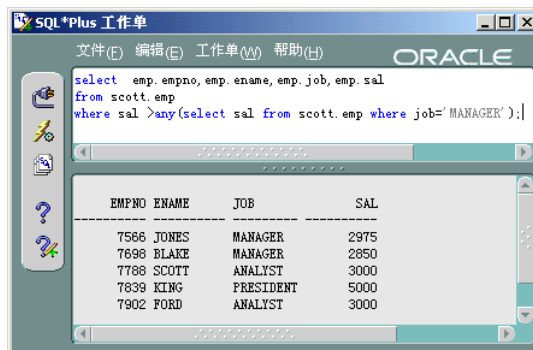


图 4.21 带【any】的嵌套查询

带 any 的查询过程等价于两步的执行过程。

(1) 执行 “select sal from scott.emp where job='MANAGER'”, 其结果如图 4.22 所示。

【参见光盘文件】: \第4章\4.4\443-1.sql。

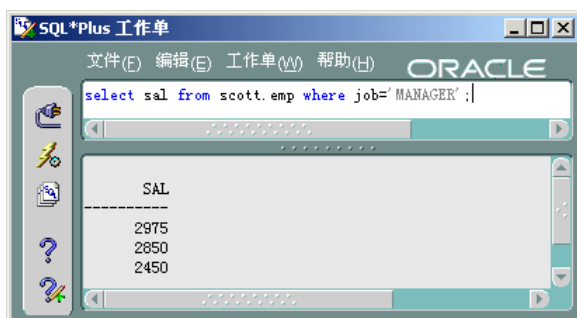


图 4.22 查询工作为 MANAGER 的员工的薪水

(2) 查询到 3 个薪水值 2975、2850 和 2450, 父查询执行下列语句。

【参见光盘文件】: \第4章\4.4\443-2.sql。

```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal >2975 or sal>2850 or sal>2450;
```

4.4.4 带【some】的嵌套查询

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal =some(select sal from scott.emp where job='MANAGER');
```

单击【执行】按钮, 出现如图 4.23 所示的结果。

【参见光盘文件】: \第4章\4.4\444.sql。

带 some 的嵌套查询与 any 的步骤相同。

(1) 子查询, 执行 “select sal from scott.emp where job='MANAGER'”, 其结果如图 4.22 所示。

(2) 父查询执行下列语句。

```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal =2975 or sal=2850 or sal=2450;
```

【参见光盘文件】: \第4章\4.4\444-2.sql。

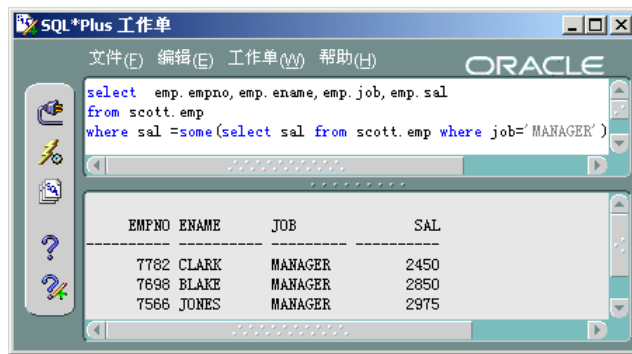


图 4.23 带【some】的嵌套查询



带【any】的嵌套查询和【some】的嵌套查询功能是一样的。早期的 SQL 仅仅允许使用【any】，后来的版本为了和英语的【any】相区分，引入了【some】，同时还保留了【any】关键词。

4.4.5 带【all】的嵌套查询

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal >all(select sal from scott.emp where job='MANAGER');
```

单击【执行】按钮，出现如图 4.24 所示的结果。

【参见光盘文件】：\第 4 章\4.4\445.sql。

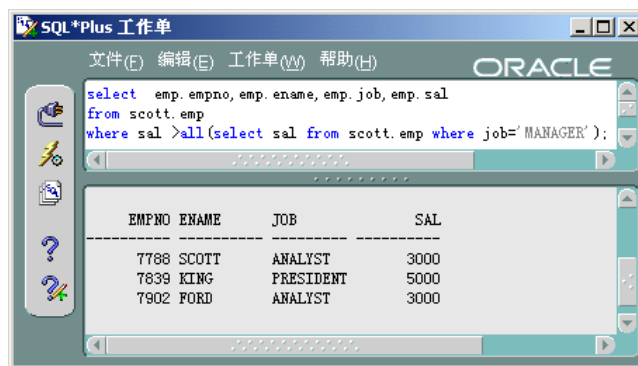


图 4.24 带【all】的嵌套查询

带 all 的嵌套查询与【some】的步骤相同。

- (1) 子查询，结果如图 4.22 所示。
- (2) 父查询执行下列语句。


```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp
where sal >2975 and sal>2850 and sal>2450;
```

【参见光盘文件】：\第4章\4.4\445-2.sql。

4.4.6 带【exists】的嵌套查询

在【命令编辑区】执行下列语句。

```
select emp.empno, emp.ename, emp.job, emp.sal
from scott.emp, scott.dept
where exists
(select * from scott.emp where scott.emp.deptno=scott.dept.deptno);
```

单击【执行】按钮，出现如图 4.25 所示的结果。

【参见光盘文件】：\第4章\4.4\446.sql。

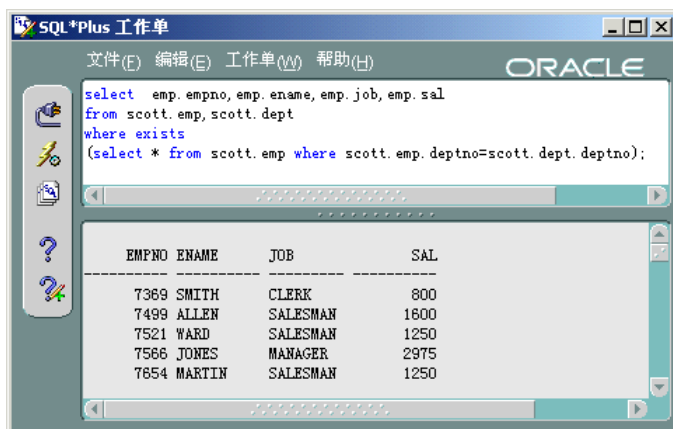


图 4.25 带【exists】的嵌套查询

4.4.7 并操作的嵌套查询

并操作就是集合中并集的概念。属于集合 A 或集合 B 的元素总和就是并集。

在【命令编辑区】执行下列语句。

```
(select deptno from scott.emp)
union
(select deptno from scott.dept);
```

单击【执行】按钮，出现如图 4.26 所示的结果。

【参见光盘文件】：\第4章\4.4\447.sql。

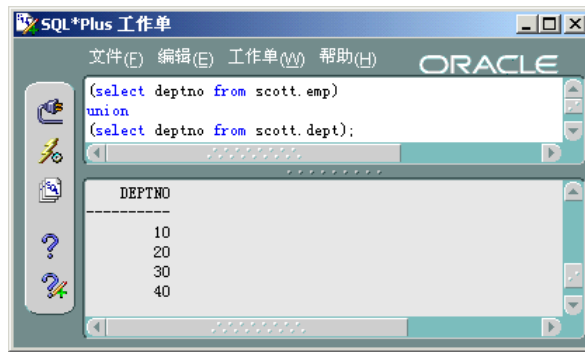


图 4.26 并操作的嵌套查询

4.4.8 交操作的嵌套查询

交操作就是集合中交集的概念。属于集合 A 且属于集合 B 的元素总和就是交集。
在【命令编辑区】执行下列语句。

```
(select deptno from scott.emp)
intersect
(select deptno from scott.dept);
```

单击【执行】按钮，出现如图 4.27 所示的结果。

【参见光盘文件】：\第 4 章\4.4\448.sql。



图 4.27 交操作的嵌套查询

4.4.9 差操作的嵌套查询

差操作就是集合中差集的概念。属于集合 A 且不属于集合 B 的元素总和就是差集。
在【命令编辑区】执行下列语句。

```
(select deptno from scott.dept)
minus
```

```
(select deptno from scott.emp);
```

单击【执行】按钮，出现如图 4.28 所示的结果。

【参见光盘文件】：\第 4 章\4.4\449.sql。



图 4.28 差操作的嵌套查询



并、交和差操作的嵌套查询要求属性具有相同的定义，包括类型和取值范围。

4.5 用 SQL 进行函数查询

Oracle 9i 提供了很多函数可以用来辅助数据查询。接下来我们介绍常用的函数功能及使用方法。

4.5.1 【ceil】函数

在【命令编辑区】输入“select mgr, mgr/100,ceil(mgr/100) from scott.emp;”，然后单击【执行】按钮，出现如图 4.29 所示的结果。

【参见光盘文件】：\第 4 章\4.5\451.sql。

MGR	MGR/100	CEIL(MGR/100)
7902	79.02	80
7698	76.98	77
7698	76.98	77
7839	78.39	79
7698	76.98	77
7839	78.39	79
7839	78.39	79
7566	75.66	76

图 4.29 【ceil】函数的使用



【ceil】函数用法：ceil (n)，取大于等于数值 n 的最小整数。

4.5.2 【floor】函数

在【命令编辑区】输入“select mgr, mgr/100,floor(mgr/100) from scott.emp;”，然后单击【执行】按钮，出现如图 4.30 所示的结果。

【参见光盘文件】：\第 4 章\4.5\452.sql。

SQL*Plus 工作单

文件(E) 编辑(E) 工作单(W) 帮助(H) ORACLE

select mgr, mgr/100,floor(mgr/100) from scott.emp;

MGR	MGR/100	FLOOR(MGR/100)
7902	79.02	79
7698	76.98	76
7698	76.98	76
7839	78.39	78
7698	76.98	76
7839	78.39	78
7839	78.39	78
7566	75.66	75

图 4.30 【floor】函数的使用



【floor】函数用法：floor (n)，取小于等于数值 n 的最大整数。

4.5.3 【mod】函数

在【命令编辑区】输入“select mgr, mod(mgr,1000), mod(mgr,100), mod(mgr,10) from scott.emp;”，然后单击【执行】按钮，出现如图 4.31 所示的结果。

【参见光盘文件】：\第 4 章\4.5\453.sql。

SQL*Plus 工作单

文件(E) 编辑(E) 工作单(W) 帮助(H) ORACLE

select mgr, mod(mgr,1000), mod(mgr,100), mod(mgr,10) from scott.emp;

MGR	MOD(MGR, 1000)	MOD(MGR, 100)	MOD(MGR, 10)
7902	902	2	2
7698	698	98	8
7698	698	98	8
7839	839	39	9
7698	698	98	8
7839	839	39	9
7839	839	39	9

图 4.31 【mod】函数的使用



【mod】函数用法：mod (m,n)，取 m 整除 n 后的余数。

4.5.4 【power】函数

在【命令编辑区】输入“select mgr, power(mgr,2),power(mgr,3) from scott.emp;”，然后单击【执行】按钮，出现如图 4.32 所示的结果。

【参见光盘文件】：\第4章\4.5\454.sql。

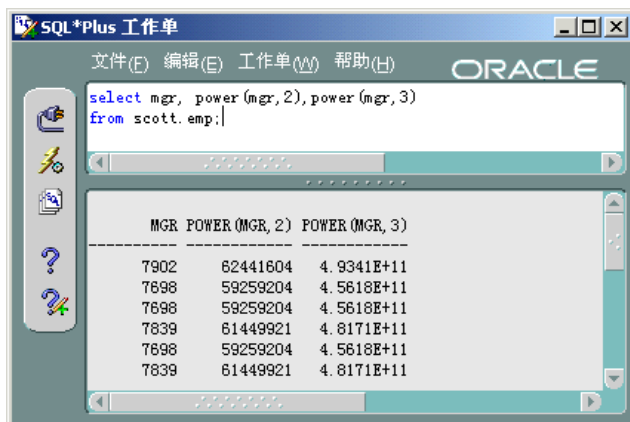


图 4.32 【power】函数的使用



【power】函数用法：power (m,n)，取 m 的 n 次方。

4.5.5 【round】函数

在【命令编辑区】输入“select mgr, round(mgr/100,2),round(mgr/1000,2) from scott.emp;”，然后单击【执行】按钮，出现如图 4.33 所示的结果。

【参见光盘文件】：\第4章\4.5\455.sql。

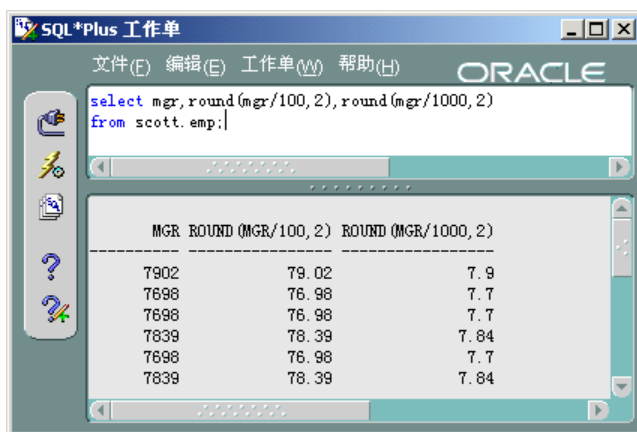


图 4.33 【round】函数的使用

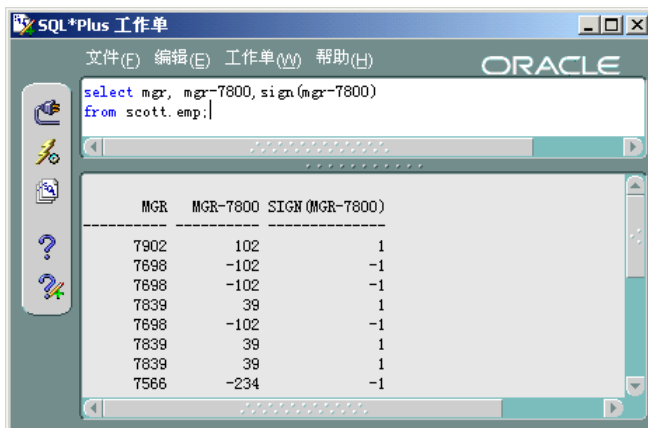


【round】函数用法：round (m,n)，四舍五入，保留 n 位。

4.5.6 【sign】函数

在【命令编辑区】输入“select mgr, mgr-7800, sign(mgr-7800) from scott.emp;”，然后单击【执行】按钮，出现如图 4.34 所示的结果。

【参见光盘文件】：\第 4 章\4.5\456.sql。



The screenshot shows the SQL*Plus interface with the query 'select mgr, mgr-7800, sign(mgr-7800) from scott.emp;' entered in the command editor. The results are displayed in a table with three columns: MGR, MGR-7800, and SIGN(MGR-7800). The data rows are as follows:

MGR	MGR-7800	SIGN(MGR-7800)
7902	102	1
7698	-102	-1
7698	-102	-1
7839	39	1
7698	-102	-1
7839	39	1
7839	39	1
7566	-234	-1

图 4.34 【sign】函数的使用

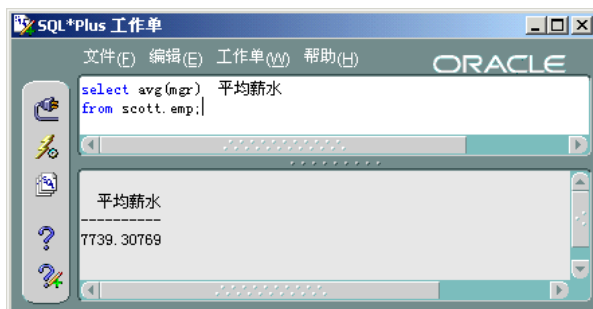


【sign】函数用法：sign (n)。n>0，取 1；n=0，取 0；n<0，取-1。

4.5.7 【avg】函数

在【命令编辑区】输入“select avg(mgr) 平均薪水 from scott.emp;”，然后单击【执行】按钮，出现如图 4.35 所示的结果。

【参见光盘文件】：\第 4 章\4.5\457.sql。



The screenshot shows the SQL*Plus interface with the query 'select avg(mgr) 平均薪水 from scott.emp;' entered in the command editor. The results are displayed in a table with one column: 平均薪水. The data row is as follows:

平均薪水
7739.30769

图 4.35 【avg】函数的使用



【avg】函数用法：avg (字段名)，求平均值。要求字段为数值型。

4.5.8 【count】函数

(1) 在【命令编辑区】输入“select count(*) 记录总数 from scott.emp;”，然后单击【执行】按钮，出现如图 4.36 所示的结果。

【参见光盘文件】: \第4章\4.5\458-1.sql。

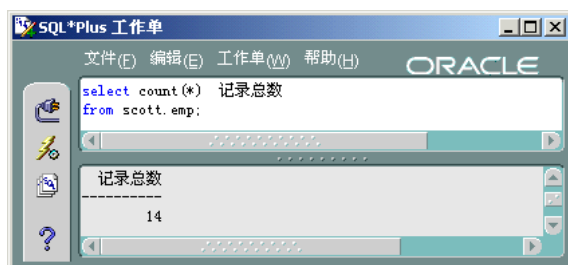


图 4.36 【count(*)】函数的使用

(2) 在【命令编辑区】输入“select count(distinct job) 工作类别总数 from scott.emp;”，然后单击【执行】按钮，出现如图 4.37 所示的结果。

【参见光盘文件】: \第4章\4.5\458-2.sql。



图 4.37 【count(字段名)】函数的使用



【count】函数用法：count（字段名）或 count（*），统计总数。

4.5.9 【min】函数

在【命令编辑区】输入“select min(sal) 最少薪水 from scott.emp;”，然后单击【执行】按钮，出现如图 4.38 所示的结果。

【参见光盘文件】: \第4章\4.5\459.sql。

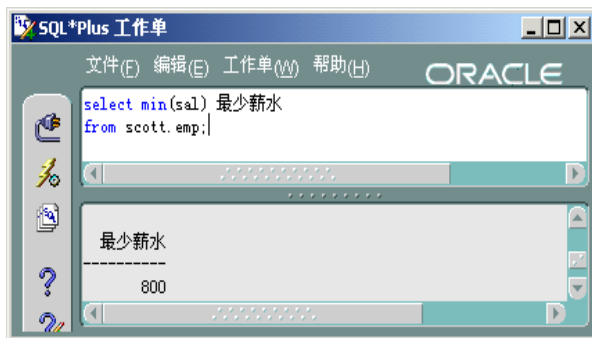


图 4.38 【min】函数的使用



【min】函数用法：min（字段名），计算数值型字段最小数。

4.5.10 【max】函数

在【命令编辑区】输入“select max(sal) 最高薪水 from scott.emp;”，然后单击【执行】按钮，出现如图 4.39 所示的结果。

【参见光盘文件】：\第 4 章\4.5\4510.sql。

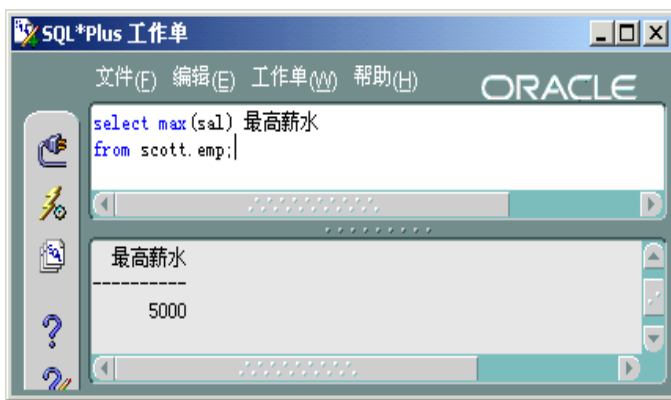


图 4.39 【max】函数的使用



【max】函数用法：max（字段名），计算数值型字段最大数。

4.5.11 【sum】函数

在【命令编辑区】输入“select sum(sal) 薪水总和 from scott.emp;”，然后单击【执行】按钮，出现如图 4.40 所示的结果。

【参见光盘文件】：\第 4 章\4.5\4511.sql。

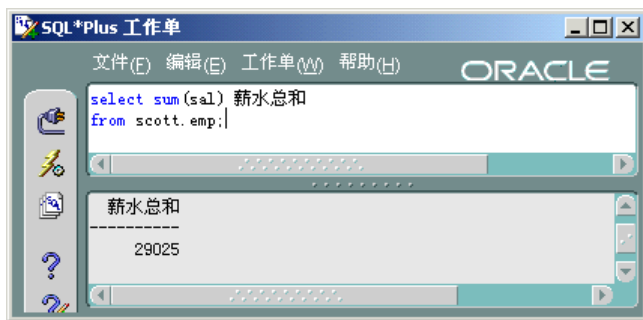


图 4.40 【sum】函数的使用



【sum】函数用法：sum（字段名），计算数值型字段总和。

通过上面 4 类查询实例的学习，读者可以举一反三，灵活运用。用 SQL 进行数据的查询就介绍到这里，下面学习如何录入数据。

4.6 用 SQL 录入数据

数据的录入采用【insert】语句。对应不同的录入方式，【insert】语句的语法会有所变化。

4.6.1 单行记录的录入

1. 语法

insert into 数据表(字段名 1, 字段名 2, ……) values(字段名 1 的值, 字段名 2 的值, ……)。

由于字段的类型不同，在书写字段值的时候要注意格式。

数值型字段，可以直接写值。

字符型字段，其值上要加上单引号。

日期型字段，其值上要加上单引号，同时还要注意年、月、日的排列次序。



在数据的插入语句中，插入列排序和插入值要一一对应。字符型和日期型字段要加上单引号，非空列必须有值。

2. 实例

在 scott.emp 数据表里共包含了 3 种类型的字段。

empno, number (4), NOT NULL, 数值型，长度为 4，不能为空。

ename, varchar2(10), 字符型，长度为 10。

hiredate, date, 日期型。

我们以在这 3 个字段中插入记录为例进行说明。

对于日期型的数据，读者往往会感觉为难，因为不知道年、月、日的排列顺序和格式，这里教大家几个方法。首先查询范例数据表中的数据，然后“依葫芦画瓢”就可以了。

(1) 在【命令编辑区】输入“select empno, ename, hiredate from scott.emp;”，然后单击【执行】按钮，出现如图 4.41 所示的结果。因此，笔者的计算机系统默认的日期型数据格式应该为“日一月一年”。

【参见光盘文件】：\第 4 章\4.6\461-1.sql。

EMPNO	ENAME	HIREDATE
7369	SMITH	17-12月-80
7499	ALLEN	20-2月-81
7521	WARD	22-2月-81
7566	JONES	02-4月-81
7654	MARTIN	28-9月-81
7698	BLAKE	01-5月-81

图 4.41 查询数据表中的数据

(2) 在【命令编辑区】输入“insert into scott.emp(empno, ename, hiredate) values (7999, 'JONE', '25-11月-2002');”，然后单击【执行】按钮，出现如图 4.42 所示的结果。

【参见光盘文件】: \第 4 章\4.6\461-2.sql。



图 4.42 成功插入数据

(3) 在【命令编辑区】输入 “select * from scott.emp where empno=7999;”, 然后单击【执行】按钮, 出现如图 4.43 所示的结果。

【参见光盘文件】: \第 4 章\4.6\461-3.sql。



图 4.43 查询录入的数据

4.6.2 多行记录的录入

在数据的录入中, 经常需要将数据表中查询到的数据稍做修改成批录入的情况, 这就是多行数据的录入。

1. 语法

```
insert into 数据表(字段名 1, 字段名 2, .....)  
(select(字段名 1 或运算, 字段名 2 或运算, .....)) from 数据表  
where 条件)
```



实际上, 首先利用子查询语句查询结果, 然后再利用 insert 语句将结果插入数据表。子查询和 insert 中的数据表既可以相同, 也可以不同, 但要求查询结果的字段和 insert 插入的数据表中字段属性完全一致。

2. 实例

在【命令编辑区】执行以下语句。

```
insert into scott.emp(empno, ename, hiredate)  
(select empno+100, ename, hiredate from scott.emp
```

```
where empno>=6999
);
```

【参见光盘文件】：\第4章\4.6\462.sql。

单击【执行】按钮，出现如图4.44所示的结果。

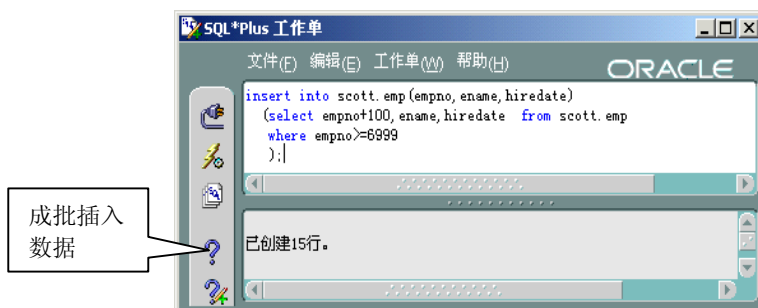


图 4.44 成批插入数据

4.6.3 表间数据复制

可以从一个数据表中选择需要的数据插入到全新的数据表中。

(1) 在【命令编辑区】执行以下语句。

```
create table scott.test
as
(
  select distinct empno,ename,hiredate
  from scott.emp
  where empno>=7000
);
```

【参见光盘文件】：\第4章\4.6\463.sql。



图 4.45 成批创建数据表并复制数据

然后单击【执行】按钮，出现如图 4.45 所示的结果。

上述语句的功能是创建一个名为 `scott.test` 的数据表，表结构包含 3 个字段。并将 `scott.emp` 中具有不同的 `empno` 字段，且 `empno >= 7000` 的数据复制到 `scott.test` 数据表中。

(2) 在【命令编辑区】输入“`select * from scott.test;`”语句，然后单击【执行】按钮，出现如图 4.46 所示的结果。

The screenshot shows the SQL*Plus interface with the command 'select * from scott.test;' entered in the command editor. Below the editor, the query results are displayed in a table format with three columns: EMPNO, ENAME, and HIREDATE. The results list seven employees with their respective IDs, names, and hire dates.

EMPNO	ENAME	HIREDATE
7369	SMITH	17-12月-80
7469	SMITH	17-12月-80
7499	ALLEN	20-2月-81
7521	WARD	22-2月-81
7566	JONES	02-4月-81
7599	ALLEN	20-2月-81
7621	WARD	22-2月-81

图 4.46 查询新数据表 test 的数据



这里的 `create table` 语句的功能是创建新的数据表，上述过程实际是分 3 步执行的。首先查询符合要求的数据，其次建立 3 个字段的名为 `test` 的数据空表，最后是将查询的数据插入到 `test` 数据表中。

4.7 用 SQL 删除数据

使用【`delete`】命令可以删除数据，使用【`truncate`】命令可以删除整表数据但保留结构。

4.7.1 删除记录

在【命令编辑区】输入“`delete from scott.test where empno >= 7500 and empno <= 8000;`”，然后单击【执行】按钮，出现如图 4.47 所示的结果。

【参见光盘文件】：\第 4 章\4.7\471.sql。



图 4.47 删除数据表 test 中的数据



删除记录的语法：`delete from 数据表 where 条件。`

4.7.2 整表数据删除

在【命令编辑区】输入“truncate table scott.test ;”，然后单击【执行】按钮，出现如图 4.48 所示的结果。

【参见光盘文件】：\第 4 章\4.7\472.sql。



图 4.48 删除数据表 test 中的所有数据



truncate table 命令将快速删除数据表中的所有记录，但保留数据表结构。这种快速删除与 delete from 数据表的删除全部数据表记录不一样，delete 命令删除的数据将存储在系统回滚段中，需要的时候，数据可以回滚恢复，而 truncate 命令删除的数据是不可恢复的。

4.8 用 SQL 更新数据

更新数据使用的是【update】命令。

4.8.1 直接赋值更新

1. 语法

update 数据表

set 字段名 1=新的赋值,字段名 2=新的赋值,.....

where 条件

2. 实例

在【命令编辑区】执行以下语句。

```
update scott.emp
```

```
set empno=8888,ename='TOM',hiredate='03-9月 -2002 '
```

```
where empno=7566;
```

【参见光盘文件】：\第 4 章\4.8\481.sql。

单击【执行】按钮，出现如图 4.49 所示的结果。

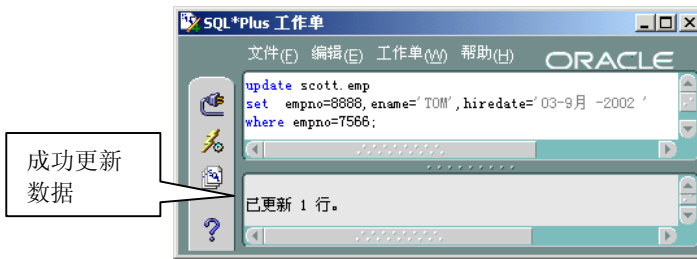


图 4.49 更新数据表中的数据

4.8.2 嵌套更新

1. 语法

update 数据表

set 字段名 1=(select 字段列表 from 数据表 where 条件), 字段名 2=(select 字段列表 from 数据表 where 条件),.....

2. 实例

在【命令编辑区】执行以下语句。

```
update scott.emp  
set sal=  
(  
    select sal+300 from scott.emp  
    where empno=7599  
)  
where empno=7599;
```

【参见光盘文件】: \第 4 章\4.8\482.sql。

单击【执行】按钮，出现如图 4.50 所示的结果。



图 4.50 嵌套更新数据表中的数据

以上我们学习了如何利用 SQL 对数据表中的数据进行录入、删除、更新和查询操作。读者利用这些实例可以结合自己的实际快速掌握这些基本的数据管理语句。

4.9 习题

- (1) SQL 有哪些主要特点，SQL 的用途有哪些？
- (2) 在 Oracle 9i 中，SQL 如何访问数据表？
- (3) 结合光盘配套文件练习掌握 SQL 查询语句的使用方法。
- (4) 结合光盘配套文件练习掌握 SQL 录入语句的使用方法。
- (5) 结合光盘配套文件练习掌握 SQL 删除语句的使用方法。
- (6) 结合光盘配套文件练习掌握 SQL 更新语句的使用方法。