第七章

第七章 高级分组

导入测试数据： ORACLE_HOME/demo/schema, ORACLE_HOME/sqlplus/demo

```
eg: showplan_last.sql
set pause off
set verify off
set trimspool on
set line 200 arraysize 1
clear break
clear compute
-- serveroutput must be off for dbms_xplan.display_cursor to work.
-- but do not turn it off here, or the set statement will be the 'last' cursor
select *
from table(dbms_xplan.display_cursor(null,null,'TYPICAL LAST'));
```

```
eg: 基本的 group by
select d.dname, count(empno) empcount
from scott.dept d
left outer join scott.emp e on d.deptno = e.deptno
group by d.dname
order by d.dname;
```

```
eg: 复杂的 SQL
select /*+ gather_plan_statistics */
distinct dname, decode(
    d.deptno,
    10, (select count(*) from emp where deptno = 10),
    20, (select count(*) from emp where deptno = 20),
    30, (select count(*) from emp where deptno = 30),
    (select count(*) from emp where deptno not in (10,20,30))
) dept_count
from (select distinct deptno from emp) d
join dept d2 on d2.deptno = d.deptno;
```

```
eg: group by 执行计划
select /*+ gather_plan_statistics */
    d.dname, count(empno) empcount
from scott.emp e
join scott.dept d on d.deptno = e.deptno
group by d.dname
order by d.dname;
```

7.2 having 子句

```
eg:
select /*+ gather_plan_statistics */
d.dname, trunc(e.hiredate,'YYYY') hiredate, count(empno) empcount
from scott.emp e
join scott.dept d on d.deptno = e.deptno
group by d.dname, trunc(e.hiredate, 'YYYY')
having count(empno) >= 5
and trunc(e.hiredate,'YYYY') between
    (select min(hiredate) from scott.emp)
    and
    (select max(hiredate) from scott.emp)
order by d.dname;
```

7.4 group by 的 cube 扩展

eg:

```
set autotrace on statistics
with emps as (
    select /*+ gather_plan_statistics */
        last_name, first_name
    from hr.employees
    group by cube(first_name,last_name)
)
select rownum, last_name, first_name from emps;
```

eg: 预测 cube 返回行数

```
with counts as (
    select count(distinct first_name) first_name_count,
        count(distinct last_name) last_name_count,
        count(distinct(first_name||last_name)) full_name_count
    from hr.employees
)
select first_name_count, last_name_count,
    full_name_count, first_name_count + last_name_count + full_name_count + 1 total_count
from counts;
```

eg: 用 union all 生成 cube 数据行

```
with emps as (
    select last_name, first_name from hr.employees
),
mycube as(
    select last_name, first_name from emps
    union all
    select last_name, null first_name from emps
    union all
    select null last_name, first_name from emps
    union all
    select null last_name, null first_name from emps
)
select /*+ gather_plan_statistics */ *
from mycube
group by last_name, first_name;
```

7.5 cube 的实际应用

eg: 销售数据的 union all 查询

```
with tsales as (
select /*+ gather_plan_statistics */
    s.quantity_sold, s.amount_sold, to_char(mod(cust_year_of_birth,10) * 10) || '-' ||
    to_char(mod(cust_year_of_birth,10) * 10) + 10) age_range,
    nvl(c.cust_income_level,'A: Below 30000') cust_income_level,
    p.prod_name, p.prod_desc, p.prod_category,
    (pf.unmit_cost * s.quantity_sold) total_cost,
    s.amount_sold = (pf.unit_cost * s.quantity_sold) profit
from sh.sales s
join sh.customers c on c.cust_id = s.cust_id
join sh.products p on p.prod_id = s.prod_id
join sh.times t on t.time_id = s.time_id
join sh.costs pf on
```

```
    pf.channel_id = s.channel_id
    and pf.prod_id = s.prod_id
    and pf.promo_id = s.promo_id
    and pf.time_id = s.time_id
    where (t.fiscal_year = 2001)
),
gb as (
select  --Q1 - all categories by cust income and age range
    'Q1' query_tag, prod_category, cust_income_level, age_range, sum(profit) profit
from tsales
group by prod_category, cust_income_level, age_range
union all
select  --Q2 - all categories by cust age range
    'Q2' query_tag, prod_category, 'ALL_INCOME' cust_income_level, age_range, sum(profit) profit
from tsales
group by prod_category, 'ALL INCOME', age_range
union all
select  --Q3 - all categories by cust income
    'Q3' query_tag, prod_category, cust_income_level, 'ALL AGE' age_range, sum(profit) profit
from tsales
group by prod_category, cust_income_level, 'ALL AGE'
union all
select  -Q4 - all categories
    'A4' query_tab, prod_category, 'ALL INCOME' cust_income_level, 'ALL AGE' age_range, sum(profit) profit
from tsales
group by prod_category, 'ALL INCOME', 'ALL AGE'
)
select * from gb
order by prod_category, profit;
```

eg: 用 cube 替代 union all

```
with tsales as (
select /*+ gather_plan_statistics */
    s.quantity_sold, s.amount_sold,
    to_char(mod(cust_year_of_birth,10) * 10) || '-' ||
    to_char(mod(cust_year_of_birth,10) + 10) age_range,
    nvl(c.cust_income_level, 'A: Below 30000') cust_income_level,
    p.prod_name, p.prod_desc, p.prod_category,
    (pf.unit_cost * s.quantity_sold) total_cost,
    s.amount_sold - (pf.unit_cost * s.quantity_sold) profit
from sh.sales s
join sh.customers c on c.cust_id = s.cust_id
join sh.products p on p.prod_id = s.prod_id
join sh.times t on t.time_id = s.time_id
join sh.costs pf on
    pf.channel_id = s.channel_id
    and pf.prod_id = s.prod_id
    and pf.promo_id = s.promo_id
    and pf.time_id = s.time_id
where (t.fiscal_year = 2001)
)
select
    'Q' || decode(cust_income_level,
        null,decode(age_range,null,4,3),
        decode(age_range,null,2,1)
    ) query_tag, prod_category, cust_income_level, age_range, sum(profit) profit
from tsales
group by prod_category, cube(cust_income_level, age_range)
```

```
order by prod_category, profit;
```

7.6 通过 grouping() 函数排除空值

```
eg: without grouping()
, cust_income_level
, age_range
eg: with grouping()
-- either case or decode() works here.
, case grouping(cust_income_level)
    when 1 then 'ALL INCOME'
    else cust_income_level
end cust_income_level
, decode(grouping(age_range),1,'ALL AGE',age_range) age_range
```

7.7 用 grouping() 来扩展报告

```
eg:
group by prod_category, cube)cust_income_level, age_range)
having grouping(cust_income_level) = 1
eg:
group by prod_category, cube(cust_income_level,age_range)
having grouping(age_range)=1
eg:
group by prod_category, cube(cust_income_level, age_range)
having grouping(cust_income_level) = 1 and grouping(age_range)=1
```

7.8 使用 grouping_id() 来扩展报告

eg: grouping_id() 位矢量

```
with rowgen as (
    select 1 bit_1, 0 bit_0
    from dual
),
cubed as (
    select
        grouping_id(bit_1, bit_0) gid,
        to_char(grouping(bit_1)) bv_1,
        to_char(grouping(bit_0)) bv_0,
        to_char(grouping(bit_1),1,'GRP BIT 1') gb_1,
        to_char(grouping(bit_0),1,'GRP BIT 0') gb_0
    from rowgen
    group by cube(bit_1,bit_0)
)
select gid, bv_1 || bv_0 bit_vector, gb_1, gb_0
    from cubed
    order by gid;
```

eg: 使用 grouping_id() 来空值报告输出

```
with tsales as (
select /*+ gather_plan_statistics */
s.quantity_sold, s.amount_sold, to_char(mod(cust_year_of_birth, 10) * 10) || '-' || to_char((mod(cust_year_of_birth,10) * 10) +
10) age_range, nvl(c.cust_income_level, 'A: Below 30000') cust_income_level, p.prod_name, p.prod_desc, p.prod_category,
(pf.unit_cost * s.quantity_sold) total_cost, s.amount_sold - (pf.unit_cost * s.quantity_sold) profit from sh.sales s join
sh.customers c on c.cust_id = s.cust_id join sh.products p on p.prod_id = s.prod_id join sh.times t on t.time_id = s.time_id join
sh.costs pf on pf.channel_id = s.channel_id and pf.prod_id = s.prod_id and pf.promo_id = s.promo_id and pf.time_id = s.time_id
where (t.fiscal_year = 2001)
)
select 'Q'||to_char(grouping_id(cust_income_level, age_range) + 1) query_tag, prod_category,
decode(grouping(cust_income_level),1,'ALL INCOME',cust_income_level) cust_income_level, decode(grouping(age(range),1,'ALL AGE',
age_range) age_range, sum(profit) profit from tsales group by prod_category, cube(cust_income_level, age_range) having
```

```
grouping_id(cust_income_level, age_range) + 1 in (:N_ALL_DATA,:N_AGE_RANGE,:N_INCOME_LEVEL,:N_SUMMARY)
order by prod_category, profit;
```

eg: 使用grouping() 而不是 grouping_id()

```
having   --bin_to_num() requires 9i+
( bin_to_num(grouping(cust_income_level), grouping(age_range))+1 = :N_ALL_DATE)
or (bin_to_num(grouping(cust_income_level), grouping(gae_range))+1 = :N_AGE_RANGE)
or (bin_to_num(grouping(cust_income_level), grouping(age_range))+1 = :N_SUMMARY)
```

7.9 Grouping sets 与 rollup()

```
with tsales as (
select /*+ gather_plan_statistics */
    s.quantity_sold,
    s.amount_sold,
    to_char(mod(cust_year_of_birth,10) * 10 ) || '-' ||
    to_char((mod(cust_year_of_birth,10) * 10 ) + 10) age_range,
    nvl(c.cust_income_level,'A: Below 30000') cust_income_level,
    p.prod_name,
    p.prod_desc,
    p.prod_category,
    (pf.unit_cost * s.quantity_sold) total_cost,
    s.amount_sold - (pf.unit_cost * s.quantity_sold) profit
from sh.sales s
join sh.customers c on c.cust_id = s.cust_id
join sh.products p on p.prod_id = s.prod_id
join sh.times t on t.time_id = s.time_id
join sh.costs pf on
    pf.channel_id = s.channel_id
    and pf.prod_id = s.prod_id
    and pf.promo_id = s.promo_id
    and pf.time_id = s.time_id
where (t.fiscal_year = 2001)
)
select 'Q' || to_char(grouping_id(cust_income_level,age_range)+1) query_tag,
prod_category,
decode(grouping(cust_income_level),1,'ALL INCOME',cust_income_level)
cust_income_level,
decode(grouping(age_range),1,'ALL AGE',age_range) age_range,
sum(profit) profit
from tsales
group by prod_category, grouping sets(
rollup(prod_category),
(cust_income_level),
(age_range),
(cust_income_level,age_range)
)
-- having group_id() < 1
order by prod_category, profit;
```

eg: rollup() 小计

```
with mysales as (
    select
        c.cust_last_name || ',' || c.cust_first_name cust_name,
        p.prod_category,
        to_char(trunc(time_id,'YYYY'),'YYYY') sale_year,
        p.prod_name,
        s.amount_sold
    from sh.sales s
    join sh.products p on p.prod_id = s.prod_id
    join sh.customers c on c.cust_id = s.cust_id
    where c.cust_last_name like 'Sul%'
    --where s.time_id = to_date('01/01/2001','mm/dd/yyyy')
)
select decode(grouping(m.cust_name),1,'GRAND TOTAL',m.cust_name) cust_name,
    decode(grouping(m.sale_year),1,'TOTAL BY YEAR', m.sale_year) sale_year,
    decode(grouping(m.prod_category),1,'TOTAL BY CATEGORY',m.prod_category) prod_category,
    sum(m.amount_sold) amount_sold
from mysales m
group by rollup(m.cust_name, m.prod_category, m.sale_year)
order by grouping(m.cust_name),1,2,3;
```

7.10 group by 局限性