第八章

分析函数

8.1 示例数据

eg:

```
drop table sales_fact;
create table sales_fact as
select country_name country, country_subRegion region, prod_name product, calendar_year year, calendar_week_number week,
sum(amount_sold) sale, sum(amount_sold*(case when mod(rownum,10)=0 then 1.4 when mod(rownum, 5)=0 then 0.6 when mod(rownum,2)=0
then 0.9 when mod(rownum,2)=1 then 1.2 else 1 end)) receipts
from sales, times, customers, countries, products
where sales.time_id=times.time_id and sales.prod_id=products.prod_id and sales.cust_id = customers.cust_id and
customers.country_id = countries.country_id group by country_name, country_subRegion, prod_name, calendar_year,
calendar_week_number;
```

8.2 分析函数剖析

function (argument1, argument2, ...argumentN) over ([partition-by-clause] [order-by-clause] [windowing-clause])

分析函数3个基本组成部分：分区子句，排序子句，开窗子句

Note:

nulls first : 将空值放到数据分区的最上面

nulls last : 将空值放到数据分区的最小面

8.3 函数列表

lag : 访问一个分区或结果集中之前的一行

lead : 访问一个分区或结果集中之后的一行

first_value : 访问一个分区或结果集中第一行

last_value : 访问一个分区或结果集中最后一行

nth_value : 访问一个分区或结果集中第n行

rank : 将数据行值按照排序后的顺序进行排名，在有并列的情况下排名值将被跳过

dense_rank : 将数据航值按照排序后的顺序进行排名，在有并列的情况下也不跳过排名值

row_number : 对行进行排序并未每一行增加一个唯一编号。这是一个非确定性行数

ratio_to_report : 计算报告中值的比例

percent_rank : 将计算得到的排名值标准化为0到1之间的值

percentile_cont : 取出与指定的排名百分比相匹配的值。是percent_rank函数的反函数

percentile_dist : 取出与指定的排名百分比想匹配的值。采用谨慎分布模型

ntile : 将数据行分组为单元

listagg : 将来自不同行的列值转化为列表格式

8.4 聚合函数

• 聚合函数可以在分析模式或传统的非分析模式下来进行运算。

分析模式的聚合函数提供了不需要任何自联结就可以聚合不同层级数据的能力。

分析函数在写需要在不同层级上对数据进行聚合的复杂查询报表时是非常有用的。

eg: sale 列的动态求和

```
select year, week, sale, sum(sale) over (partition by product, country, region, year order by week rows between unbounded
preceding and current row) running_sum_ytd
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;
```

8.4.1 跨越整个分区的聚合函数

eg: sale 列的最大值

```
select year, week, sale, max(sale) over (partition by product, country, region, year order by week rows between unbounded
preceding and unbounded following) max_sale
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;
```

8.4.2 细粒度窗口声明

eg ： 计算本周前两周到本周后两周共梧州的时间窗口内sale列的最大值

rows between 2 preceding and 2 following

eg: 5周时间跨度窗口内sale列的最大值

```
select year, week, sale, max(sale) over (partition by product, country, region, year order by week rows between 2 preceding and 2 following) max_week_5
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;
```

8.4.3 默认窗口声明

默认的窗口子句是 rows between unbounded preceding and current row.

8.5 Lead 和 Lag

Lag 和 Lead 函数提供了跨行引用的能力。

Lag 提供了访问结果集中前面的行的能力，

Lead 允许访问结果集中后面的行。

8.5.1

语法：

lag (expression, offset, default) over (partition-clause order-by-clause)

Lead 和 Lag 函数不支持开窗子句。仅支持partition-by 子句和 order by 子句

8.5.2 例1：从前一行中返回一个值

eg:

```
select year, week, sale, lag(sale,1,sale) over (partition by product, country, region order by year, week) prior_wk_sales from sales_fact where country in ('Australia') and product='Xtend Memory' order by product, country,year,week;
```

8.5.3 理解数据行的位移

eg: 位移值为10的Lag函数

```
select year, week, sale, lag(sale,10,sale) over (partition by product, country, region order by year, week) prior_wk_sales_10
from sales_fact where country in ('Australia') and product = 'Xtend Memory' order by product, country, year, week;
```

8.5.4 例2：从下一行中返回一个值

eg: Lead 函数

```
select year, week, sale, lead(sale,1,sale) over (partition by product, country, region order by year, week) prior_wk_sales from sales_fact where country in ('Australia') and product = 'Xtend Memory' order by product, country, year, week;
```

8.6 First_value 和 Last_value

本质上来讲，任何计算最大值和最小值的报表都可以使用first_value和last_value函数

语法：

first_value(expression) over (partition-clause order-by-clause windowing-clause)

8.6.1 例子：使用First_value 来计算最大值

eg:

```
select year, week, sale,
first_value (sale) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) top_sale_value,
first_value (week) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) top_sale_week
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;
```

8.6.2 例子：使用 Last_value 来计算最小值

eg:

```
select year, week, sale,
```

last_value (sale) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) low_sale
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;
Note: 可以有效地使用窗口声明的粒度控制来生成复杂报表
eg: 指定了再一个21行数据的窗口中求最大值或最小值。
rows between 10 preceding and 10 following 子句

8.7 其他分析函数
8.7.1 Nth_value (11gR2)
nth_value 是 first_value 和 last_value 函数的通用化版本，可以获取排过序的结果集中的任意一行。
first_value 函数可以写为 nth_value (column_name,1)
语法：
nth_value (measure, n) [from first | from last] [respect nulls | ignore nulls] over (partitioning-clause order-by-clause windowing-clause)
eg:
select year, week, sale,
nth_value (sale, 2) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) sale_2nd_top
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;

8.7.2 rank
rank 函数已数值形式返回一个数据行在排序后的结果集中的位置。
Note: rank 排名可能有重复的排名，且排名可能不是连续的。
语法：
rank() over (partition-clause order-by-clause)
Note: 开窗子句在rank函数中不适用
eg: rank 函数的使用：销售额前10位的周
select * from (
select year, week, sale,
rank() over (partition by product, country, region, year order by sale desc) sales_rank
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week
) where sales_rank <=10
order by 1,4;

8.7.3 dense_rank
dense_rank 是 rank 函数的变体。区别在于当存在并列值的时候，dense_rank函数不会跳过排名值
eg:
select * from (
select year, week, sale,
dense_rank() over (partition by product, country, region, year order by sale desc) sales_rank
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week
) where sales_rank <=10
order by 1,4;

8.7.4 row_number
row_number 函数为有序结果集中的每一行分配唯一的行编号。
Note: row_number 函数不支持开窗子句
语法：
row_number() over (partition-clause order-by-clause)
row_number 函数是一个非确定性函数。即如果两行具有同样的值，row_number函数的值是不确定的，多次执行这个查询可能会得到不同的结果。而

rank 和 dense_rank 函数是确定性函数，如果重复执行查询将会返回一致的数据。

eg:

```
select year, week, sale,
row_number() over (partition by product, country, region, year order by sale desc) sales_rn,
rank() over (partition by product, country, region, year order by sale desc) sales_rank
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, sales_rank;
```

8.7.5 ratio_to_report

ratio_to_report 函数计算数据分区中某个值与和值的比率。

eg:

```
select year, week, sale,
trunc(100*ratio_to_report(sale) over (partition by product, country, region, year),2) sales_yr,
trunc(100*ratio_to_report(sale)  over (partition by product, country, region),2) sales_prod
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by product, country, year, week;
```

8.7.6 percent_rank

percent_rank 函数以0到1之间的份数形式返回某个值在数据分区中的排名。

计算公式为 (rank - 1) / (n - 1)

eg:

```
select * from (
select year, week, sale,
100 * percent_rank() over (partition by product, country, region, year order by sale desc) pr
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
) where pr < 50
order by year, sale desc;
```

8.7.7 percentile_cont

percentile_cont函数接收一个0到1之间的几率值并返回与声明了排序的percent_rank函数计算值相等的百分位值。percentile_cont函数是percent_rank函数的反函数，对于计算百分位数，中位数很有用。

eg:

percentile_cont(0.25) 子句获取 percent_rank 为0.25的值。

percentile_cont(0.5) 子句返回中位值。

median 函数是 percentile_cont 函数的一个默认值为0.5的特例。

语法：

```
percentile_cont(expr) within group (sort-clause) over (partition-clause order-by-clause)
```

Note: 该函数不支持开窗子句

eg:

```
select year, week, sale,
percentile_cont (0.5) within group (order by sale desc) over (partition by product, country, region, year) pc,
percent_rank() over (partition by product, country, region, year order by sale desc) pr
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
```

8.7.8 Percentile_disc

percentile_disc 函数在功能上类似于 percentile_cont 函数，区别在于 percentile_cont 函数使用了连续分布模型，而 percentile_disc 函数使用了离散分布模型。

eg:

```
select year, week, sale,
percentile_disc(0.5) within group (order by sale desc) over (partition by product, country, region, year) pd_desc,
percentile_disc (0.5) within group (order by sale) over (partition by product, country, region, year) pd_asc,
percent_rank () over (partition by product, country, region, year order by sale desc) pr
from sales_fact
where country in ('Australia') and product = 'Xtend Memory';
```

8.7.9 ntile

ntile 函数对一个数据分区中的有序结果集进行划分，将其分组为各个桶，并为每个小组 分配一个唯一的组编号。在统计学术语中，ntile 函数创建等宽直方图信息。

eg:

ntile (100) 将数据行分组为100个桶，并为每个桶分配唯一编号。

eg:

```
select year, week, sale
ntile (10) over (partition by product, country, region, year order by sale desc) group#
from sales_fact
where country in ('Australia') and product = 'Xtend Memory';
```

8.7.10 stddev

stddev 函数可以计算标准偏差，即方差的平方根

eg:

```
select year, week, sale,
stddev (sale) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) stddv
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
order by year, week;
```

8.7.11 listagg

listagg 函数可将列转行。

语法：

```
listagg (string, separator) within group (order-by-clause) over (partition-by-clause)
```

Note: listagg 函数不支持开窗子句

eg:

```
select listagg (country, ',') within group (order by country desc)
from (
select distinct country from sales_fact
order by country
);
```

8.8 性能调优

8.8.1 执行计划

关键字 window sort 表明使用了分析函数。

8.9 高级话题

8.9.1 动态SQL

eg: 动态SQL语句

```
create or replace procedure
analytic_dynamic_prc (part_col_string varchar2, v_country varchar2, v_product varchar2)
is
    type numtab is table of number(18,2) index by binary_integer;
    l_year numtab;
    l_week numtab;
    l_sale numtab;
    l_rank numtab;
    l_sql_string varchar2(521);
begin
    l_sql_string := 'select * from (select year, week, sale, rank() over (partition by '||part_col_string||' order by sale desc) slaes_rank from sales_fact where country in (' ||chr(39) || v_country ||chr(39) || ') and product = ' || chr(39) || v_product || chr(39) ||' order by product, country, year, week) where sales_rank <= 10 order by 1,4';
    execute immediate l_sql_string bulk collect into l_year, l_week, l_sale, l_rank;
    for i in 1 .. l_year.count
    loop
        dbms_output.put_line(l_year(i)||' ' || l_week(i) ||'|'|| l_sale(i) ||'|' || l_rank(i) );
```

```
    end loop;
    end;
/
exec analytic_dynamic_prc('product, country, region','Australia','Xtend Memory');
exec analytic_dynamic_prc('product, country, region, year','Australia','Xtend Memory');
```

8.9.2 嵌套分析函数

分析函数不能进行嵌套，但可以通过子查询来实现嵌套的效果。

eg: 嵌套分析函数

```
select year, week, top_sale_year,
lag (top_sale_year) over (order by year desc) prev_top_sale_yer
from (
select distinct
    first_value(year) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) year,
    first_value(week) over (partition by product, country, region, year order by sale desc rows between unbounded preceding and unbounded following) week,
    first_value(sale) over (partition by prouct, country, region, year order by sale desc rows between unbounded preceding and unbounded following) top_sale_year
from sales_fact
where country in ('Australia') and product = 'Xtend Memory'
)
order by year, week;
```

8.9.3 并行

分析函数也可以并行

8.9.4 PGA 大小

大多数与分析函数相关的运算都是在进程的程序共享区PGA中进行的。

有一个足够大的内存区以便程序能够不必使用硬盘来执行分析函数很重要。

8.10 组织行为

8.11 小结

使用分析函数来很简洁地写出复杂SQL语句。