ORACLE物化视图

一、----------

物化视图是包括一个查询结果的数据库对像，它是远程数据的的本地副本，或者用来生成基于数据表求和的汇总表。物化视图存储基于远程表的数据，也可以称为快照。

物化视图可以查询表，视图和其它的物化视图。

通常情况下，物化视图被称为主表（在复制期间）或明细表（在数据仓库中）。

对于复制，物化视图允许你在本地维护远程数据的副本,这些副本是只读的。如果你想修改本地副本，必须用高级复制的功能。当你想从一个表或视图中抽取数据时，你可以用从物化视图中抽取。

对于数据仓库，创建的物化视图通常情况下是聚合视图，单一表聚合视图和连接视图。

本篇我们将会看到怎样创建物化视图并且讨论它的刷新选项。

在复制环境下，创建的物化视图通常情况下主键，rowid,和子查询视图。

1.主键物化视图:

下面的语法在远程数据库表emp上创建主键物化视图

```
SQL> CREATE MATERIALIZED VIEW mv_emp_pk

 REFRESH FAST START WITH SYSDATE

NEXT  SYSDATE + 1/48

WITH PRIMARY KEY

AS SELECT * FROM emp@remote_db;
```

2010年01月(2)
2009年11月(1)
2009年10月(1)
2009年09月(1)
2009年07月(9)
2009年06月(26)
2009年05月(14)
2009年02月(6)
2009年01月(15)
2008年12月(1)
2008年11月(2)
2008年10月(1)
2008年08月(1)
2008年07月(5)
2007年12月(5)
2007年08月(20)
2007年07月(4)
2007年04月(1)

```
Materialized view created.
```

注意：当用FAST选项创建物化视图，必须创建基于主表的视图日志,如下：

```
SQL> CREATE MATERIALIZED VIEW LOG ON emp;

Materialized view log created.
```

2.Rowid物化视图

下面的语法在远程数据库表emp上创建Rowid物化视图

```
SQL> CREATE MATERIALIZED VIEW mv_emp_rowid
  REFRESH WITH ROWID
   AS SELECT * FROM emp@remote_db;
  Materialized view log created.
```

3.子查询物化视图

下面的语法在远程数据库表emp上创建基于emp和dept表的子查询物化视图

```
SQL> CREATE MATERIALIZED VIEW  mv_empdept
AS SELECT * FROM emp@remote_db e
WHERE EXISTS
  (SELECT * FROM dept@remote_db d
 WHERE e.dept_no = d.dept_no)
Materialized view log created.
```

REFRESH 子句

```
      [refresh [fast|complete|force]
[on demand | commit]
[start with date] [next date]
```

```
[with {primary key|rowid}]]
```

Refresh选项说明:

a. oracle用刷新方法在物化视图中刷新数据.

b. 是基于主键还是基于rowid的物化视图

c. 物化视图的刷新时间和间隔刷新时间

Refresh方法-FAST子句

增量刷新用物化视图日志（参照上面所述）来发送主表已经修改的数据行到物化视图中.如果指定REFRESH FAST子句，那么应该对主表创建物化视图日志

```
SQL> CREATE MATERIALIZED VIEW LOG ON emp;

Materialized view log created.
```

对于增量刷新选项，如果在子查询中存在分析函数，则物化视图不起作用。

Refresh方法- COMPLETE子句

完全刷新重新生成整个视图，如果请求完全刷新，oracle会完成 完全刷新即使增量刷新可用。

Refresh Method – FORCE 子句

当指定FORCE子句，如果增量刷新可用Oracle将完成增量刷新，否则将完成完全刷新,如果不指定刷新方法(FAST, COMPLETE, or FORCE),Force选项是默认选项

主键和ROWD子句

WITH PRIMARY KEY选项生成主键物化视图,也就是说物化视图是基于主表的主键，而不是ROWID(对应于ROWID子句). PRIMARY KEY是默认选项,为了生成PRIMARY KEY子句，应该在主表上定义主键，否则应该用基

于ROWID的物化视图.

主键物化视图允许识别物化视图主表而不影响物化视图增量刷新的可用性。

Rowid物化视图只有一个单一的主表，不能包括下面任何一项:

n Distinct 或者聚合函数.

n Group by，子查询，连接和SET操作

刷新时间

START WITH子句通知数据库完成从主表到本地表第一次复制的时间,应该及时估计下一次运行的时间点，NEXT 子句说明了刷新的间隔时间.

```
SQL> CREATE MATERIALIZED VIEW mv_emp_pk
        REFRESH FAST
     START WITH SYSDATE
     NEXT  SYSDATE + 2
     WITH PRIMARY KEY
     AS SELECT * FROM emp@remote_db;
     Materialized view created.
```

在上面的例子中，物化视图数据的第一个副本在创建时生成，以后每两天刷新一次.

总结

物化视图提供了可伸缩的基于主键或ROWID的视图,指定了刷新方法和自动刷新的时间。

二、----------

Oracle的物化视图提供了强大的功能，可以用于预先计算并保存表连接或聚集等耗时较多的操作的结果，这样，在执行查询时，就可以避免进行这些耗时的操作，而从快速的得到结果。物化视图有很多方面和索引很相似：使用物化视图的目的是为了提高查询性能；物化视图对应用透明，增加和删除物化视图不会影响应用程序中SQL语句的正

确性和有效性；物化视图需要占用存储空间；当基表发生变化时，物化视图也应当刷新。

物化视图可以分为以下三种类型：包含聚集的物化视图；只包含连接的物化视图；嵌套物化视图。三种物化视图的快速刷新的限制条件有很大区别，而对于其他方面则区别不大。创建物化视图时可以指定多种选项，下面对几种主要的选择进行简单说明：

创建方式（Build Methods）：包括BUILD IMMEDIATE和BUILD DEFERRED两种。BUILD IMMEDIATE是在创建物化视图的时候就生成数据，而BUILD DEFERRED则在创建时不生成数据，以后根据需要在生成数据。默认为BUILD IMMEDIATE。

查询重写（Query Rewrite）：包括ENABLE QUERY REWRITE和DISABLE QUERY REWRITE两种。分别指出创建的物化视图是否支持查询重写。查询重写是指当对物化视图的基表进行查询时，Oracle会自动判断能否通过查询物化视图来得到结果，如果可以，则避免了聚集或连接操作，而直接从已经计算好的物化视图中读取数据。默认为DISABLE QUERY REWRITE。

刷新（Refresh）：指当基表发生了DML操作后，物化视图何时采用哪种方式和基表进行同步。刷新的模式有两种：ON DEMAND和ON COMMIT。ON DEMAND指物化视图在用户需要的时候进行刷新，可以手工通过DBMS_MVIEW.REFRESH等方法来进行刷新，也可以通过JOB定时进行刷新。ON COMMIT指出物化视图在对基表的DML操作提交的同时进行刷新。刷新的方法有四种：FAST、COMPLETE、FORCE和NEVE*。**ST刷新采用增量刷新，只刷新自上次刷新以后进行的修改。COMPLETE刷新对整个物化视图进行完全的刷新。如果选择FORCE方式，则Oracle在刷新时会去判断是否可以进行快速刷新，如果可以则采用FAST方式，否则采用COMPLETE的方式。NEVER指物化视图不进行任何刷新。默认值是FORCE ON DEMAND。

在建立物化视图的时候可以指定ORDER BY语句，使生成的数据按照一定的顺序进行保存。不过这个语句不会写入物化视图的定义中，而且对以后的刷新也无效。

物化视图日志：如果需要进行快速刷新，则需要建立物化视图日志。物化视图日志根据不同物化视图的快速刷新的需要，可以建立为ROWID或PRIMARY KEY类型的。还可以选择是否包括SEQUENCE、INCLUDING NEW VALUES以及指定列的列表。

可以指明ON PREBUILD TABLE语句将物化视图建立在一个已经存在的表上。这种情况下，物化视图和表必须同名。当删除物化视图时，不会删除同名的表。这种物化视图的查询重写要求参数QUERY_REWRITE_INTEGERITY必须设置为trusted或者stale_tolerated。

物化视图可以进行分区。而且基于分区的物化视图可以支持分区变化跟踪（PCT）。具有这种特性的物化视图，当基表进行了分区维护操作后，仍然可以进行快速刷新操作。对于聚集物化视图，可以在GROUP BY列表中使用CUBE或ROLLUP，来建立不同等级的聚集物化视图。

物化视图的基本操作和使用可以查看网址：http://blog.itpub.net/post/468/13318 相关的东东。我主要说明一下使用物化视图的基本东东。如如何建立在特定的表空间上，这些在其他的物化视图上面几乎都没有任何介绍的。主要以我做的一个例子来操作，
如果对物化视图的基本概念清楚了就比较明白在那里写特定的表空间存储了。

创建物化视图时应先创建存储的日志空间

```
CREATE MATERIALIZED VIEW LOG ON mv_lvy_levytaxbgtdiv
tablespace ZGMV_DATA --日志保存在特定的表空间
WITH ROWID ;
CREATE MATERIALIZED VIEW LOG ON tb_lvy_levydetaildata
tablespace ZGMV_DATA --日志保存在特定的表空间
WITH ROWID,sequence(LEVYDETAILDATAID);
CREATE MATERIALIZED VIEW LOG ON tb_lvy_levydata
tablespace ZGMV_DATA --日志保存在特定的表空间
WITH rowid,sequence(LEVYDATAID);
```

然后创建物化视图

```
--创建物化视图
create materialized view MV_LVY_LEVYDETAILDATA
TABLESPACE ZGMV_DATA --保存表空间
BUILD DEFERRED --延迟刷新不立即刷新
refresh force --如果可以快速刷新则进行快速刷新，否则完全刷新
on demand --按照指定方式刷新
start with to_date('24-11-2005 18:00:10', 'dd-mm-yyyy hh24:mi:ss') --第一次刷新时间
next TRUNC(SYSDATE+1)+18/24 --刷新时间间隔
as
SELECT levydetaildataid, detaildatano, taxtermbegin, taxtermend,
......
ROUND(taxdeduct * taxpercent1, 2) - ROUND(taxdeduct * taxpercent2, 2) -
ROUND(taxdeduct * taxpercent3, 2) - ROUND(taxdeduct * taxpercent4, 2) -
ROUND(taxdeduct * taxpercent5, 2) taxdeduct, ROUND(taxfinal * taxpercent1, 2) -
ROUND(taxfinal * taxpercent2, 2) - ROUND(taxfinal * taxpercent3, 2) -
ROUND(taxfinal * taxpercent4, 2) - ROUND(taxfinal * taxpercent5, 2) taxfinal,
a.levydataid, a.budgetitemcode, taxtypecode,
......
FROM tb_lvy_levydetaildata a, tb_lvy_levydata c, MV_LVY_LEVYTAXBGTDIV b
```

```
WHERE a.levydataid = c.levydataid
AND a.budgetdistrscalecode = b.budgetdistrscalecode
AND a.budgetitemcode = b.budgetitemcode
AND c.incomeresidecode = b.rcvfisccode
AND C.TAXSTATUSCODE='08'
AND C.NEGATIVEFLAG!='9'
```

删除物化视图日志

```
--删除物化视图:
--删除日志:  DROP materialized view log on mv_lvy_levytaxbgtdiv;
DROP materialized view log on tb_lvy_levydetaildata;
DROP materialized view log on tb_lvy_levydata;
--删除物化视图 drop materialized view MV_LVY_LEVYDETAILDATA;
```

--基本和对表的操作一致 --物化视图由于是物理真实存在的，故可以创建索引。

创建方式和对普通表创建方式相同，就不在重复写了。

三、----------

物化视图对于前台数据库使用者来说如同一个实际的表,具有和一般表相通的如select等操作,而其实际上是一个视图,一个由系统实现定期刷新其数据的视图(具体刷新时间在定义物化视图的时候已有定义),使用物化视图更可以实现视图的所有功能,而物化视图确不是在使用时才读取,大大提高了读取速度,特别适用抽取大数据量表某些信息以及数据链连接表使用.具体语法如下:

物化视图对于前台数据库使用者来说如同一个实际的表,具有和表相通的一般select操作,而其实际上是一个视图,一个定期刷新数据的视图(具体刷新时间在定义物化视图的时候已有定义),使用物化视图可以实现视图的所有功能,而物化视图确不是在使用时才读取,大大提高了读取速度,特别适用抽取大数据量表某些信息以及数据链连接表使用.具体语法如下:

```
CREATE MATERIALIZED VIEW an_user_base_file_no_charge
           REFRESH COMPLETE START WITH SYSDATE
          NEXT TRUNC(SYSDATE+29)+5.5/24
  as
select distinct user_no
from cw_arrearage t
where (t.mon = dbms_tianjin.getLastMonth or
      t.mon = add_months(dbms_tianjin.getLastMonth, -1))
drop materialized view an_user_base_file_no_charge;
```

第 13 章 物化视图

8.1.5企业版/个人版开始支持

需要权限：GRANT CREATE MATERIALIZED VIEW，还必须直接赋予GRANT QUERY REWRITE.为实现查询重写，必须使用CBO.
13.1 物化视图如何工作
设置
COMPATIBLE参数必须高于8.1.0
QUERY_REWRITE_ENABLED = TRUE
QUERY_REWRITE_INTEGRETY =
ENFORCED - 查询仅用Oracle强制与保证的约束、规则重写;
TRUSTED – 查询除用Oracle强制与保证的约束、规则，也可用用户设定的数据间的任何关系来重写;
STALE_TOLERATED – 即便Oracle知道物化视图中数据过期（与事实表等不同步），也重写查询。
创建物化视图的用户必须具有直接赋予的GRANT QUERY REWRITE权限，不能通过角色继承。
内部机制
全文匹配
部分匹配：从FROM子句开始，优化器比较之后的文本，然后比较SELECT列表
一般重写方法：
数据充分
关联兼容
分组兼容
聚集兼容
13.2 确保使用物化视图
约束
考虑到现实环境的数据量，可以将主键、外键、非空等约束置为NOVALIDATE，并调整QUERY_REWRITE_INTEGRITY为TRUSTED，这样可以达到“欺骗”数据库的目的，但必须注意如果无法保证此类约束的真实有效，查询改写后可能造成结果不精确。
维度
实际就是指明已存在的表中各列的归并关系，从而关联事实表后形成的物化视图可用于向“上”归并（相当于用表中代表更高归并关系的列关联事实表）。标准语法：
```
CREATE DIMENSION time_hierarchy_dim
LEVEL day      IS time_hierarchy.day
LEVEL mmyyyy   IS time_hierarchy.mmyyyy
LEVEL yyyy     IS time_hierarchy.yyyy
HIERARCHY time_rollup
(day CHILD OF mmyyyy CHILD OF yyyy)
ATTRIBUTE mmyyyy
DETERMINES mon_yyyy;
```
13.3 DBMS_OLAP
估计（物化视图）大小

DBMS_OLAP.ESTIMATE_SUMMARY_SIZE（视图名， 视图定义， 估计行数， 估计字节数）；
其中后两个参数为NUMBER型输出参数。
维度有效性检查
DBMS_OLAP.VALIDATE_DIMENSION（视图名， 用户名， FALSE， FALSE）；
SELECT * FROM 维度表名
WHERE ROWIN IN （SEELCT bad_rowid FROM MVIEW$_EXCEPTION）；
所选出行即为不符合维度定义的行。
推荐物化视图
首先必须添加合适的外键，包通过外键来判定表之间的关系而不是维度。
DBMS_OLAP.RECOMMEND_MV（事实表名， 1000000000， ‘’）；
第二个参数表示物化视图可用的空间大小，可传入一个较大的数。第三个参数传入需要保留的特定物化视图，传入空即为不考虑其他物化视图。
执行C：\oracle\RDBMS\demo\sadvdemo后执行：
DEMO_SUMADV.PRETTYPRINT_RECOMMENDATIONS
13.4 最后说明
物化视图不为OLTP系统设计
在事实表等更新时会导致物化视图行锁，从而影响系统并发性。

四、----------

定位导致物化视图无法快速刷新的原因
==========
物化视图的快速刷新采用了增量的机制，在刷新时，只针对基表上发生变化的数据进行刷新。因此快速刷新是物化视图刷新方式的首选。

但是快速刷新具有较多的约束，而且对于采用ON COMMIT模式进行快速刷新的物化视图更是如此。对于包含聚集和包含连接的物化视图的快速刷新机制并不相同，而且对于多层嵌套的物化视图的快速刷新更是有额外的要求。

如此多的限制一般很难记全，当建立物化视图失败时，Oracle给出的错误信息又过于简单，有时无法使你准确定位到问题的原因。

Oracle提供的DBMS_MVIEW.EXPLAIN_MVIEW过程可以帮助你快速定位问题的原因。下面通过一个例子来说明，如果通过这个过程来解决问题。

建立一个快速刷新的嵌套物化视图：

```
SQL> CREATE TABLE B (ID NUMBER PRIMARY KEY, NAME VARCHAR2(30));

表已创建。

SQL> CREATE TABLE C (ID NUMBER PRIMARY KEY, NAME VARCHAR2(30));

表已创建。

SQL> CREATE TABLE A (ID NUMBER, BID NUMBER, CID NUMBER, NUM NUMBER,
  2  CONSTRAINT FK_A_B_BID FOREIGN KEY (BID) REFERENCES B(ID),
  3  CONSTRAINT FK_A_C_BID FOREIGN KEY (CID) REFERENCES C(ID));

表已创建。

SQL> INSERT INTO B SELECT ROWNUM, 'B'||ROWNUM FROM USER_TABLES WHERE ROWNUM <
= 6;

已创建6行。

SQL> INSERT INTO C SELECT ROWNUM, 'C'||ROWNUM FROM USER_TABLES WHERE ROWNUM <
= 4;

已创建4行。

SQL> INSERT INTO A SELECT ROWNUM, TRUNC((ROWNUM - 1)/2) + 1, TRUNC((ROWNUM - 1)/3)
+ 1, ROWNUM
  2  FROM USER_TABLES
  3  WHERE ROWNUM <= 12;

已创建12行。

SQL> COMMIT;

提交完成。
```

上面建立好基表，下面建立第一层物化视图。

```
SQL> CREATE MATERIALIZED VIEW LOG ON A WITH ROWID;

实体化视图日志已创建。

SQL> CREATE MATERIALIZED VIEW LOG ON B WITH ROWID;

实体化视图日志已创建。

SQL> CREATE MATERIALIZED VIEW LOG ON C WITH ROWID;

实体化视图日志已创建。

SQL> CREATE MATERIALIZED VIEW MV_ABC REFRESH FAST ON COMMIT ENABLE QUERY REWRIT
E AS
  2  SELECT C.ID CID, C.NAME CNAME, B.ID BID, B.NAME BNAME, A.NUM,
  3  A.ROWID AROWID, B.ROWID BROWID, C.ROWID CROWID
  4  FROM A, B, C WHERE A.BID = B.ID AND A.CID = C.ID;

实体化视图已创建。

第一次物化视图已经建立成功，下面建立嵌套物化视图：

SQL> CREATE MATERIALIZED VIEW LOG ON MV_ABC WITH ROWID (BNAME, CNAME, NUM) INCL
UDING NEW VALUES;

实体化视图日志已创建。

SQL> CREATE MATERIALIZED VIEW MV_MV_ABC REFRESH FAST ON COMMIT ENABLE QUERY RE
WRITE AS
  2  SELECT CNAME, BNAME, COUNT(*) COUNT, SUM(NUM) SUM_NUM FROM MV_ABC
  3  GROUP BY CNAME, BNAME;
SELECT CNAME, BNAME, COUNT(*) COUNT, SUM(NUM) SUM_NUM FROM MV_ABC
                                                    *
ERROR 位于第 2 行:
ORA-12053: 这不是一个有效的嵌套实体化视图
```

错误出现了，不过错误的描述包含的信息量并不大。我们看看Oracle的文档上是如何描述这个错误的。

ORA-12053 this is not a valid nested materialized view

Cause: The list of objects in the FROM clause of the definition of this materialized view had some dependencies upon each other.

Action: Refer to the documentation to see which types of nesting are valid.

文档上的描述也是十分笼统的，并没有指出具体问题所在。

接下来，我们通过使用DBMS_MVIEW.EXPLAIN_MVIEW过程来定位错误。

使用EXPLAIN_MVIEW过程首先要建立MV_CAPABILITIES_TABLE表，建表的脚步是$ORACLE_HOME/rdbms/admin/utlxmv.sql。（EXPLAIN_MVIEW过程是两个过程的重载，一个输出到MV_CAPABILITIES_TABLE表，另一个以PL/SQL的VARRAY格式输出，为了简单起见，我们建立MV_CAPABILITIES_TABLE表）。

```
SQL> @?rdbmsadminutlxmv.sql
```

表已创建。

下面简单研究一下EXPLAIN_MVIEW过程。

```
DBMS_MVIEW.EXPLAIN_MVIEW(mv IN VARCHAR2, Statement_id IN VARCHAR2:= NULL);
```

该过程可以输入已经存在的物化视图名称（或USER_NAME.MV_NAME），也可输入建立物化视图的查询语句。另外一个参数STATEMENT_ID输入一个语句ID，为了标识出表中对应的记录。

```
SQL> BEGIN
  2  DBMS_MVIEW.EXPLAIN_MVIEW('SELECT CNAME, BNAME, COUNT(*) COUNT, SUM(NUM) SUM
_NUM FROM MV_ABC
  3  GROUP BY CNAME, BNAME', 'MV_MV_ABC');
  4  END;
  5  /

PL/SQL 过程已成功完成。

SQL> SELECT CAPABILITY_NAME, RELATED_TEXT, MSGTXT FROM MV_CAPABILITIES_TABLE
```

```
  2  WHERE STATEMENT_ID = 'MV_MV_ABC' AND POSSIBLE = 'N' AND CAPABILITY_NAME NOT LI
KE '%PCT%';

CAPABILITY_NAME                RELATED_TEXT     MSGTXT
------------------------------ -------------- --------------------------------------------------
REFRESH_FAST_AFTER_ONETAB_DML  SUM_NUM          使用 SUM(expr) 时, 未提供 COUNT(expr)
REFRESH_FAST_AFTER_ANY_DML     YANGTK.MV_ABC    mv 日志没有序列号
REFRESH_FAST_AFTER_ANY_DML                      查看禁用 REFRESH_FAST_AFTER_ONETAB_DML 的
原因
```

根据上面的信息，已经可以确定问题的原因了，对于聚集物化视图，使用了SUM(COLUMN)，但是没有包括COUNT(COLUMN)。

修改物化视图，重新建立：

```
SQL> CREATE MATERIALIZED VIEW MV_MV_ABC REFRESH FAST ON COMMIT ENABLE QUERY RE
WRITE AS
  2  SELECT CNAME, BNAME, COUNT(*) COUNT, COUNT(NUM) NUM_COUNT, SUM(NUM) SUM_NU
M FROM MV_ABC
  3  GROUP BY CNAME, BNAME;
```

实体化视图已创建。

用 户 名: 匿名用户 登录 注册

验 证 码:  重新获得验证码

发表评论