

1 问题的提出

随着ORACLE数据库在越来越多领域的应用,许多应用的共享数据量变得越来越大,而大部分的数据空间是被少数大数据量表占用。通常这些表都有很强的时效性及区域性,对不同时期的数据处理要求各异。例如:本年度数据的访问、操作频繁,十分活跃、访问量较大,对时间响应要求高;而隔年数据在做年转结后就变为只读,仅供查询,访问量较少、不活跃,对时间响应要求不高,但要求数据管理简单。对业务系统中存在的这种大数据量表进行有效的管理和维护,提高有大数据量表的应用系统性能、简化数据管理,变得越来越重要。

2 解决策略

对大数据量表进行处理,提高应用系统性能有两个方案:(1)分区处理;(2)分表处理。

分表处理就是将大数据量表分解为工作表和 history 表来处理。工作表:用于存放本年度数据;历史表:用于存放历史数据。每年将转结后的工作表数据导入历史表中,并对工作表进行清空。分表处理在应用上,实现难度较大、管理复杂;应用系统的开发量较大;不利于系统维护。

分区就是将大数据量表分割成相对较小、可独立管理的部分。分区的表与未分区的表在执行查询语句或其它DML语句时没有任何区别。表的各分区与定义它们的表的逻辑属性是相同的,但各分区的物理属性可以不同,即各分区可以有自己的存储参数和表空间。由于表空间数据存放的物理位置是由数据文件的存放位置决定的;因此,通过合理地选择数据文件的存放位置、设置正确的存储参数、将并发访问的数据文件分别存储在不同的硬盘中。当用户访问数据时,可以同时从不同的硬盘读取所需的数据。减少了处理的数据量、避免硬盘I/O冲突,提高了应用系统的性能。

ORACLE中对大数据量表的处理方法

A Processing Method of Mass Data Table in Oracle

黄杰圣 李传目(福建厦门集美大学网络中心 361021)

摘要: 本文阐述了在ORACLE数据库应用系统中使用分区技术处理大数据量表的原理和方法,并给出实现的例子。

关键词: ORACLE 表 大数据量 分区

由于分区在ORACLE中是被当作段来对待的,所以我们在管理和维护数据库时有很大的灵活性、对分区有很多的控制和管理方法。既可以对分区表直接管理,也可以对各分区进行单独管理和维护,且分区的单独操作互相不会影响。例如:对存放历史数据的区可以设为只读、访问频繁的区可以优化系统访问参数等等。这样就简化了数据的管理。

ORACLE对分区的管理十分简便,它提供了一套SQL命令来管理分区对象,主要有以下几类操作:①添加新分区②合并已有分区③分割已有分区④删除已有分区⑤移动分区⑥更改分区属性⑦重命名分区。ORACLE主要有四类分区:①范围分区②列表分区③散列分区④复合分区。

对大数据量表采用分区技术可以较好地解决大数据量表对应用系统性能造成的影响、简化用户对大数据量的管理。

3 应用实例

以厦门医保管理系统为例,有一个医保收费明细数据表(yb_fee),用于保存厦门市所有医院每天每笔业务的数据。由于每天全市各家医院时刻不停的对表进行插入、更

新、删除查询等操作,使该共享表逐日增大,十分明显地影响了系统的执行效率、增加了数据管理的难度。即使是使用了高配置、先进的硬件,还是明显感得系统执行日益缓慢、管理困难。因此,我们对yb_fee表应用范围分区进行管理和维护。以yb_fee表中2001年-2003年数据为例;我们把yb_fee表以医保收费时间(fee_date)为分区字段。按年度分成三个区,每个区各自拥有独立的表空间,见下表:

yb_fee表分区(2001--2003)

表名	yb_fee		
分区名	fee_2001	fee_2002	fee_2003
表空间名	Yb_2001	yb_2002	yb_2003

以下是简化的yb_fee表分区的建立方法:

```

① 建立表yb_fee的各分区表空间
CREATE TABLESPACE yb_2001
DATAFILE '/ybdata/fee/yb_fee2001.
dat'
SIZE 500M
EXTENT MANAGEMENT LOCAL
UNIFORM SIZE 10M;
CREATE TABLESPACE yb_2002
DATAFILE 'yb_data/fee/yb_fee2002.

```

```

dat'
    SIZE 500M
    EXTENT MANAGEMENT LOCAL
    UNIFORM SIZE 10M;
CREATE TABLESPACE yb_2003
DATAFILE 'yb_data/fee/yb_fee2003.
dat'
    SIZE 500M
    EXTENT MANAGEMENT LOCAL
    UNIFORM SIZE 10M;
② 建立基于分区的表yb_fee
CREATE TABLE yb_fee
(card_no char(18),
fee_date date,
total_fee number(10,2),
.....)
STORAGE (INITIAL 100M NEXT 30M)
LOGGING
PARTITION BY RANGE(fee_date)
(PARTITION fee_2001 VALUES LESS THAN
(TO_DATE( '01-01-2002' , 'dd-mm-
yyyy' ))
TABLESPACE yb_2001 NOLOGGING,
PARTITION fee_2002 VALUES LESS THAN
(TO_DATE( '01-01-2003' , 'dd-mm-
yyyy' ))
TABLESPACE yb_2003 NOLOGGING,
PARTITION fee_2003 VALUES LESS THAN
(TO_DATE( '01-01-2004' , 'dd-mm-
yyyy' ))
TABLESPACE yb_2003 NOLOGGING
);
    在PARTITION BY RANGE 子句后的括号
    中列出分区的字段名( fee_date ),若有多个
    分区字段,则各字段之间需要用逗号分隔。
    每个分区的定义由PARTITION关键字开
    头,其后分区名为可选,若不指定ORACLE会
    自动对分区命名。
    VALUES LESS THAN用于设置一个分区中
    分区字段值的范围。
    
```

通过使用TABLESPACE子句,yb_fee表的四个分区分别存储在不同的表空间中,而且每个表空间如有要求还可根据需要使用不同的存储参数。

以后ORACLE的DBA可根据需要每年或几年添加一个新分区进行扩容。新分区只能添加在已有分区的最后一个分区之后。

如要增加一个2004年分区,则使用:

```

① 建立表空间
CREATE TABLESPACE yb_2004
DATAFILE 'ybdata/fee/yb_fee2004.
dat'
    SIZE 500M
    EXTENT MANAGEMENT LOCAL
    UNIFORM SIZE 10M;
② 增加分区
ALTER TABLE yb_fee
ADD PARTITION yb_2004 VALUES LESS
THAN
(TO_DATE( '01-01-2005' , 'dd-mm-
yyyy' ))
TABLESPACE yb_2004;
③ 将表空间yb_2003设为只读
ALTER TABLESPACE yb_2003 READ ONLY;
若要删除2001年数据则使用
ALTER TABLE yb_fee
DROP PARTITION fee_2001;
若要导出2001年数据则使用
EXP username/password TABLES=yb_fee:
fee_2001
ROWS=Y
FILE=yb_2001.dmp;
若要导入2001年数据则使用
IMP username/password FILES=yb_2001.
dmp
TABLES=(yb_fee;fee_2001)IGNORE=y
    系统运行证明:由于使用分区,较好地
    解决了yb_fee表对系统的不良影响,有效地
    提高了系统的性能;同时也简化了yb_fee表的数
    据管理。
    
```

4 结论

当表的数据很大,对数据的管理、存储、建立索引及进行查询都很困难时,可采用分区处理。其好处:

- (1) 均衡I/O:把表的不同分区分配到不同的磁盘上,可以平衡I/O、改善性能;
- (2) 改善系统操作性能:对大表的查询、增加、修改等操作可以分解到表的不同分区来并行执行,使运行速度加快;
- (3) 数据管理变得简单:由于表的数据的导入、导出、索引重建等操作可以在不同分区上独立进行,因此更快更简单;
- (4) 快速恢复:若要恢复表的某部分数据,对分区操作比对单个大表操作要简单快速,而且可以并行处理;
- (5) 增加有效性:若由于某种原因导致表的某几个分区故障而不能使用时,表的其余好的分区仍然可以使用。

ORACLE数据库以高可靠性、安全性、兼容性,越来越广泛应用于各领域,如何针对不同应用使ORACLE保持优良性能,对我们提出了考验;需要我们在实际应用中不断摸索和改进。

参考文献

[1] [美] David Austin Meghraj Thakkar Kurt Lysy
升级到ORACLE8i,人民邮电出版社。