高扬制丝SPC分析系统Oracle数据库优化探索

郑　捷　高扬国际烟草有限公司　设备动力部

【摘　要】高扬制丝SPC分析系统长时间运行之后，性能下降，为满足用户需求，必须针对其后台Oracle数据库根据数据采集系统的特点实施合理的优化,一方面对Oracle数据库的缓存区、系统进程进行调整，另一方面对数据的物理模型进行恰当的修改,使应用系统处于良好的运行状态。本文描述了优化的全过程。

【关键字】SPC　数据采集　Oracle　优化　性能调整

前言

高扬制丝统计过程控制(SPC)系统于2004年3月正式投入运行，制丝线全面实施统计过程控制为完善整条制丝线的集中监控，进一步挖掘出历史数据蕴涵的价值，提高制丝线的过程控制水平构建了良好的平台。而SPC系统分析所用到的数据是由位于制丝控制室的上位机负责传送到Oracle数据库中的，每隔六秒就采集一次全线五个工艺段的22个质量特性值。由于采集频率较大，系统运行较长一段时间之后，积累数据已达4GB，数据库负荷较重，响应时间较长，达不到用户需求，必须对数据库实施优化，在满足数据采集速度的基础上尽可能的保证用户查询分析操作的正常进行。

Oracle公司提供了许多优化工具，数据库管理员利用这些工具进行周期性的调整可以防止出现数据瓶颈，使数据库达到最佳性能以满足用户的需要。本文拟对在优化过程中遇到的问题进行分析，并阐述优化方案实施的细节。

一、背景简介

1.1　软硬件环境

高扬SPC系统后台数据库为9.2.0.4版Oracle数据库，运行于DELL POWEREDGE 4600 NT服务器上，服务器操作系统为Win2000 Server、内存4G、双Intel P4 Xeon CPU，存储设备为Dell PowerVault 220S磁盘阵列。

1.2　应用类型

高扬SPC系统作为生产现场数据采集系统，属于比较典型的决策支持系统(DSS),用户对数据基本只查询不修改，负责上传送数据的上位机也基本只添加数据，对历史的数据不会发生修改。

1.3　数据字典

高扬SPC系统数据库包含2张业务表、11张代码表、5个视图，其中，最重要的是GY_AUTOSAMPLE表，它记录了所有自动采集的22个质量特性值的数据，目前共有3120 批次的2362万行数据，该表上建立了5个索引。

表1　业务主表GY_AUTOSAMPLE的字段定义情况

表名		行数	占用空间
gy_autosample		23624029	1243M
字段名	类型	描述	
QualityIndexCode	VarChar2(2)	质量指标代码	
BrandCode	VarChar2(2)	牌号	
ShiftTypeCode	VarChar2(1)	班别代码	
ShiftGroupCode	VarChar2(1)	班次代码	
BatchNo	VarChar2(12)	批次号	
SampleTime	VarChar2(14)	采样时间	
QualityIndexValue	Number(8,2)	质量指标值	
ProductionStatus	VarChar2(1)	生产状态	

表2　业务主表GY_AUTOSAMPLE的索引定义情况

类型	索引名	索引列	唯一	占用空间
主键	PK_AUTOSAMPLE	SampleTime +QualityIndexCode	是	413M
外键	BRAND_AUTOSAMPLE_FK	BrandCode	否	350M
外键	SHIFTGROUP_AUTOSAMPLE_FK	ShiftGroupCode	否	378M
外键	SHIFTTYPE_AUTOSAMPLE_FK	ShiftTypeCode	否	364M
外键	QUAINDEX_AUTOSAMPLE_FK	QualityIndex	否	302M
合计				1807M

1.4　数据模型

用户在使用SPC系统过程中最常用到的功能是数采历史信息菜单下的批次综合查询和详细信息查询 。因这两个查询功能响应时间较慢，而成为此次优化的重点攻关内容。模块内嵌SQL语句涉及到的业务表、代码表的物理模型如图1所示：

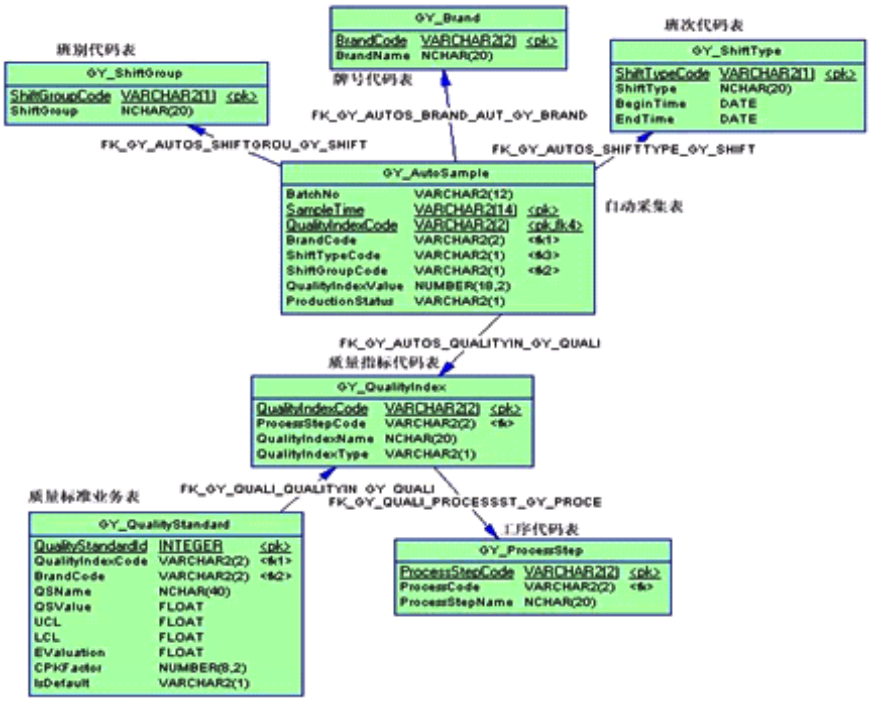

图1　自动数据采集Schema物理模型

1.5　优化前性能

这次针对SPC系统的Oracle数据库优化方案是建立在Oracle提供的状态收集和性能诊断工具StatsPack连续一周采样的数据基础之上的，Statspack除了查找实例中的性能问题外，还可以查找应用程序中高负荷的SQL语句，很容易确定Oracle 数据库的瓶颈所在，并且记录数据库性能状态。这次Statspack监测快照级别选择的是LEVEL 5(一般性能统计+SQL语句)。快照门限根据决策支持系统的特点做了调整，目的是缩小范围，快速定位运行效率较低的SQL语句。

表3　Oracle StatsPack快照门限参数

快照门限参数	参数意义	设置值	默认值
executions_th	sql语句执行的次数	5	10
Disk_reads_th	sql语句执行的磁盘读取数量	3000	1000
parse_calls_th	sql语句执行的解析调用数量	2000	1000
buffers_gets_th	sql语句执行的缓冲区获取的数据量	20000	10000

StatsPack输出的性能分析报告见结束语中优化前后性能比较表。

二、优化方案

根据StatsPack输出的分析报告，对数据库进行了以下四个方面的调整。

2.1　大表分区

由于GY_AUTOSAMPLE表占用空间已达4GB,且StatsPack报告反映出文件离散读取(DB File Scattered Read)等待事件比较显著，分析后得出如下几个主要原因：一是22个关键特性值每隔6秒同时插入到数据库的同一普通表中，这些数据存放在同一个Segment、Extent乃至同一个DB块的可能性极大,不同特性值的数据混杂在一起，不能从物理上加以区分。而用户每次查询时只会关心一种特性值的数据，将质量标准上下限不同的特性值数据放在一起画CPK图是毫无意义的。这就势必造成在查询一个特性值的历史趋势时，Oralce同时读取了用户并不需要的其它21个关键特性值(Oracle读取最小的单位是DB块)。二是init.ora参数db_file_multiblock_read_count设为16，即在读取一个DB块时同时读入邻近的16个DB块，Oracle进一步读取了无用的数据。三是使用CBO优化器模式,在统计更新不及时的情况下， Oracle可能会自动跳过索引，执行全表扫描。这些原因都造成了文件离散读取等待时间过高。

解决的办法是利用Oracle提供的分区技术，将一个很大的表根据一定的规则分别存储到不同的区域，这样可将逻辑上的一张表物理存储在多张小表中，分区的优势在于：一、增强可用性，如果表的一个分区由于系统故障而不能使用，表的其余好的分区仍然可以使用；二、减少关闭时间：如果系统故障只影响表的一部分分区，那么只有这部分分区需要修复，故能比修复整个大表花费的时间更少；三、维护轻松：如果需要重建表，独立管理每个分区比管理单个大表要轻松得多；四、均衡I/O：可以把表的不同分区分配到不同的磁盘来平衡I/O改善性能；五、改善性能：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，可使运行速度更快；六、分区对用户透明，最终用户感觉不到分区的存在，避免了应用程序的修改升级。

Oracle9i中提供了三种分区方式，列表分区(List)、范围分区(Range)、散列分区(Hash)，其中,根据特性值代码进行列表分区方式非常适合SPC系统的实际需求，即先为每个关键特性值建立单独的表空间,然后定义分区规则，将不同的关键特性值移入对应的分区中，最后出于兼容性考虑，建立一个默认分区(GYWEB_P23)存放将来可能出现的新的特性值，创建分区的SQL命令如表4：

表4　建立分区表SQL语句

```
create table GY_AutoSample(
    BatchNo              VARCHAR2(12)                    not null,
    ……
    ProductionStatus VARCHAR2(1)                    not null
 )
PARTITION BY LIST (QualityIndexCode) (
PARTITION GYWEB_P01 VALUES ('01') tablespace GYWEB_TP01,
PARTITION GYWEB_P02 VALUES ('02') tablespace GYWEB_TP02,
……
PARTITION GYWEB_P23 VALUES (DEFAULT)tablespace GYWEB_TP23);
```

在创建分区之后，应根据实际情况调整存储参数使运行效率进一步提升。可有两种选择，一种是改变空间管理方式为自动区段空间管理：原来的本地管理表空间(LMT)是通过把EXTENT MANAGEMENT LOCAL子句添加到tablespace的定义句法而实现的。和旧的字典管理表空间(DMT)不同，LMT实现了扩展管理自动化。而9i第2版中新添加的自动区段空间管理（ASSM）是通过将SEGMENT SPACE MANAGEMENT AUTO子句添加到tablespace的定义句法里而实现的。它使用位图freelist取代传统单向的链接列表freelist，ASSM的表空间会将freelist的管理自动化，并取消为独立的表和索引指定PCTUSED、FREELISTS和FREELIST GROUPS存储参数的能力。换言之，ASSM就是将存储管理的决定权完全交给Oracle控制；另一种选择是仍使用LMT方式，但修改PCTFREE参数由原来的20%改为2%，依据是数据只插入不修改删除，为了使数据存储更加紧密，每一次Disk Read读入的数据行数更多，可将每个DB块写满98%才停止插入。在查阅了Oracle官方文档和在测试环境进行实验之后，发现LMT方式占用空间相对节省，同时也能保证数据采集的速度，是更理想的选择。

表5　修改存储参数

```
ALTER TABLE GYWEB.GY_AUTOSAMPLE
 MODIFY DEFAULT ATTRIBUTES PCTFREE 2
```

在正式环境将数据迁移至新建的分区表并创建本地分区索引之后，追踪详细信息查询模块相关SQL语句的执行计划，发现查询的数据范围已由原来1.2GB范围缩小到60M，提高了运行效率。

2.2　索引调整

索引对于提高检索数据的速度起着至关重要的作用，在分析了StatsPack报表中利用率最高的30句SQL语句及监控所有的索引利用情况之后，发现SPC系统的索引存在着一些性能问题。

表6　监控索引使用情况

```
ALTER INDEX BRAND_AUTOSAMPLE_FK MONITORING USAGE;
select owner,index_name,table_name,used
from all_object_usage where used = "NO";
```

1）GY_AUTOSAMPLE表中四个分别指向牌号,班别代码,班次代码，质量指标值代码表的外键索引在日常的查询中根本用不上，却占用了1.5GB的空间。这些外键唯一的功能就是满足外键约束(FK Constraint)，保证数据的完整性，避免出现无意义的数据。为改变这样的状况可以有两种选择。一种为修改索引类型：修改索引类型即是将四个B*Tree类索引都改为位图索引，系统的实际情况的确也符合位图索引低基数的要求，利用位图索引节省空间的优势可以将1.5GB的索引减至500M，但位图索引不支持行级锁定，数采系统数据插入速度得不到保证。另一种是使用添加CHECK类约束这种手工的方法，即在表中添加CHECK类约束来代替外键的功能，如规定GY_AUTOSAMPLE表的班别列在插入时只接受1、2、3三种数值，其它均拒绝。这样做的好处显而易见，删除四个外键索引后节省了所有的空间，高速插入数据也几乎不受影响，缺点是如增加一个班别必须手工修改后台数据表，而这样的业务需求改动发生的可能性较小。权衡利弊之后，选择了添加CHECK类约束的方法进行索引优化。

表7　添加CHECK类约束的方法

```
ALTER TABLE GYWEB.GY_AUTOSAMPLE ADD CONSTRAINT GY_AUTOS_SHIGR CHECK
(SHIFTGROUPCODE='1'
 or SHIFTGROUPCODE='2' or SHIFTGROUPCODE='3');
```

2）在Top30 SQL中排在前列的几条查询语句中仅包含批次值、关键特性值两个条件，调研了用户需求之后，发现统计员经常以批次号作为条件查询每批烟丝生产过程中的各关键特性值的总CPK值，虽然已经建立了特性值索引，但在批次号列上未建索引，SQL运行时会进行全表扫描，这就大大的影响了这些语句的执行效果，所以必须补建批次号本地分区索引，索引建立之后该查询的响应时间由3分钟降至2秒，效果非常明显。

表8　创建批次号索引

```
CREATE INDEX GYWEB.BATCHNO_AUTOSAMPLE_PART
ON GYWEB.GY_AUTOSAMPLE(BatchNo)  LOCAL;
```

2.3　参数调整

内存的优化历来都是数据库优化的重中之重，优化内存主要是通过优化内存结构来提高系统性能。这里所说的内存结

构主要由专用SQL及PL／SQL区、共享池、日志缓冲区和高速缓冲存储区构成。内存使用的优化原则是要确保内存中的数据被最大限度的利用。由于访问内存比访问磁盘快的多，所以总希望把尽可能多的数据放入内存，但实际内存数量有限。因此调人内存的数据必须是经常使用的数据，有很高的命中率。如果内存中的数据命中率不是很高，就会对系统的性能产生较大的影响。这时，数据库的某些运行参数必须调整。init.ora文件中定义了许多与性能相关的重要参数，根据SPC实际应用情况选择了其中5个比较重要的参数做重点分析。

表9　调整前参数设置情况

参数名称	原值(字节)	原值(兆)
sga_max_size	1326524708	1265
db_cache_size	830472192	792
shared_pool_size	276824064	264
log_buffer	524288	0.5
sort_area_size	2097152	2
parallel_max_servers	5个	n/a

Oracle共享的内存区就是系统全局区(SGA),SGA中存放着数据库中所有用户的共同信息，它由数据库缓冲区、共享缓冲区、重做日志缓冲区组成。

表10　SGA区内存分配情况

```
SQL>  select * from v$sga;
NAME                      VALUE
-------------------- ----------
Fixed Size                456996
Variable Size          494927872
Database Buffers       830472192
Redo Buffers              667648
```

1）要检查数据库缓冲区(Database Buffers)的大小是否合理，需通过公式：命中率=(逻辑读-物理读)/逻辑读*100%。测算得到SPC数据库缓冲区命中率仅为55%左右，大大低于80%理想状态，必须设法增大DB_CACHE_SIZE参数。

2）共享缓冲区由字典高速缓存和SQL共享缓存组成 ,SPC后台数据库中这两个缓存分别的命中率为0.075(理想值为小于1)、0.026(理想值为小于15)，都处于非常好的状态，shared_pool_size可以调的小些，让出部分内存空间划给数据缓冲区。

表11　共享缓存区命中率

```
SQL> select sum(reloads)/sum(pins)*100 from v$librarycache;
SUM(RELOADS)/SUM(PINS)*100
--------------------------
        0.0754677190734068
SQL> select sum(gets) "gets",sum(getmisses) "getmisses",
  2  sum(getmisses)/sum(gets)*100 "rate"
  3  from v$rowcache;
     gets  getmisses       rate
---------- ---------- ----------
```

```
457594963      119932 0.02620920
```

3）重做日志缓冲区相对SGA区来说较小，但当此值设置太小时，LGWR进程会频繁的将LOG BUFFER中的数据写入磁盘，增加I/O的次数，影响系统的性能。理想的Value列应接近于0，SPC系统中为1448，说明日志写操作频繁，重做日志缓冲区设置的偏小,可将其适当增加。

表12　重做日志缓冲区

```
SQL> select name,value from v$sysstat
  2  where name='redo log space requests';
NAME                            VALUE
------------------------- ----------
redo log space requests          1448
```

4）当与Oracle建立起一个会话(session)时，在内存中就会为该session分配一个私有的排序区域。如果该连接是一个专用的连接(dedicated connection)，那么就会根据init.ora中sort_area_size参数的大小在内存中分配一个Program Global Area (PGA) 。对于所有的session，用做排序的内存量都必须是一样的。当排序不能在分配的空间中完成时，就会使用磁盘排序的方式，即在Oracle实例中的临时表空间中进行。磁盘排序的开销是很大的，和内存排序相比较，它们特别慢,而且磁盘排序会消耗临时表空间中的资源,同时Oracle还必须分配缓冲池块来保持临时表空间中的块。无论什么时候，内存排序都比磁盘排序好。Oracle默认将排序区设置为512K ，SPC系统中原排序区大小设为2M，而动态视图中没有出现因为大量排序产生的磁盘I/O操作，测试时将其设为1M，运行一周也没有发生大量的等待事件，可以认为原来的参数设置的过大了。

表13　重做日志缓冲区

```
SQL> select * from v$sysstat where name like 'sort%';
STATISTIC# NAME                 CLASS      VALUE
---------- ------------------ ------- ----------
       242 sorts (memory)         64    9815026
       243 sorts (disk)           64         38
       244 sorts (rows)           64  727462844
```

5）SPC服务器中安装了2块CPU， Oracle也提供了很多面向多CPU的功能。从Oracle8i开始，Oracle在每个数据库函数中都实现了并行性，包括SQL访问（全表检索）、并行数据操作和并行恢复。参数parallel_max_servers_parameter决定了并行查询服务器进程的最大数量，Oracle推荐设置为CPU数量*16,而SPC系统原设置值仅为5，应做适当调整。

6）众所周知，在32位的操作系统如Win2000上，Oracle使用的总内存有2G限制。而SPC系统拥有4G内存，剩余的2G内存无法拨给Oracle使用，造成了系统资源浪费。使用微软提供的4GT(4G Tuning)的技术，使得oracle使用超过2G(不超过3G)的内存成为可能。修改boot.ini文件，在启动windows项中添加 /3G 参数之后,就打开了操作系统3G开关,可以将多出来的1G内存分配给数据缓冲区使用。

根据以上的分析和模拟实验，对原有参数进行了如下调整：

表14　调整后参数设置情况

参数名称	新值(字节)	新值(兆)
sga_max_size	3221225472	3072

db_cache_size	2705326080	2580
shared_pool_size	201326592	192
log_buffer	1048576	1
sort_area_size	1048576	1
parallel_max_servers	32个	n/a

2.4　其它的注意事项

1）由于SPC后台数据库采用了CBO优化器模式(基于成本)，定义和管理好统计数据就显得尤为重要，为了使CBO能够为SQL语句产生一个最好的执行计划，必须要有与SQL语句相关的表和索引统计数据。只有当CBO知道了相关的信息，如表的大小、分布、基数以及列值的可选性等，才能对SQL语句作出正确的判断。定期执行统计更新对于系统的长期良好运行起着至关重要的作用。

表15　定期统计更新

```
exec dbms_stats.GATHER_SCHEMA_STATS('gyweb');
```

2）如果用户反映某个查询操作很慢，可以通过跟踪会话的方式来诊断是否存在瓶颈，研究具体的执行计划，在User_Dump_Dest下根据该会话的进程号或者线程号可以找到一个产生的trace文件。通过使用tkprof格式化文件之后就可以得到详细的统计信息，包括执行计划、Parse/Fetch等步骤消耗CPU时间。通常是观察Query模式下的Consistent Gets值来检查SQL是否使用了索引，然后看执行计划是否正常，是否有调整的余地，这样自下而上优化过程对于保障系统平稳运行，提高用户满意度起着积极的作用。

表16　追逐会话信息

```
tkprof oragy02_ora_656.trc output.txt sys=no
```

三、结束语

这次优化共耗时2周，对SPC系统后台Oracle数据库做了全面的调整，取得了如下的成果：

1）响应时间得到了极大的提升，用户的等待时间显著减少，提高了用户满意度。

2）进行了大表分区，梳理了原先杂乱存放的数据，提高了查询的运行效率。

表16　业务主表GY_AUTOSAMPLE优化后的情况

类型	名称		空间
表	gy_autosample		1179M
类型	名称	索引列	空间
主键	PK_AUTOSAMPLE_PART	SampleTime +QualityIndexCode	413M
外键	BATCHNO_AUTOSAMPLE_PART	BatchNo+QualityIndexCode	303M
合计			1895M

3）删除了4个无用的索引，补 建了1个常用的所用，节省了37％的磁盘空间，减轻了系统后续的存储压力。

4）对内存参数进行了大幅度的调整，数据缓冲区大小增加了近两倍，提高数据缓冲区的命中率。

5）数据库非空闲等待事件进一步降低，文件分散读取、缓冲区释放、缓冲区忙等事件持续的时间同比减少75%。

6）为SPC系统今后长期稳定的运行打下了良好的数据库平台基础。

优化前后性能的比较报表（节选）如下：划红线的表示,通过优化,该性能指标获得了改善,主要是物理读磁盘趋向减少，缓冲区命中率达到了99%，离散读磁盘事件等待时间大幅减少，退出了TOP 5 Timed Events。

Cache Sizes (end)

Buffer Cache:	792M	Std Block Size:	8K
Shared Pool Size:	264M	Log Buffer:	512K

Load Profile

	Per Second	Per Transaction
Redo size:	7,977.53	2,512.72
Logical reads:	1,155.60	363.98
Block changes:	51.92	16.35
Physical reads:	3,008.55	947.62
Physical writes:	2.05	0.65
User calls:	45.36	14.29
Parses:	17.86	5.63
Hard parses:	2.81	0.88
Sorts:	3.95	1.24
Executes:	13.10	4.13
Transactions:	3.17	

% Blocks changed per Read:	4.49	Recursive Call %:	36.33
Rollback per transaction %:	3.31	Rows per Sort:	32.17

Instance Efficiency Percentages (Target 100%)

Buffer Nowait %:	99.99	Redo NoWait %:	100.00
Buffer Hit %:	80.31	In-memory Sort %:	100.00
Library Hit %:	94.36	Soft Parse %:	84.28
Execute to Parse %:	-36.32	Latch Hit %:	99.93
Parse CPU to Parse Elapsd %:	78.67	% Non-Parse CPU:	94.30

Top 5 Timed Events

Event	Waits	Time (s)	% Total Ela Time
direct path read	2,621,301	57,415	58.07
PX Deq: Execute Reply	111,416	30,132	30.48
db file scattered read	208,623	5,686	5.75
CPU time		3,479	3.52
log file sync	44,194	493	.50

图2　优化前Statspack性能分析报表（节选）

Cache Sizes (end)

Buffer Cache:	2,580M	Std Block Size:	8K
Shared Pool Size:	192M	Log Buffer:	1024K

Load Profile

	Per Second	Per Transaction
Redo size:	8,570.66	2,791.94
Logical reads:	2,322.29	758.91
Block changes:	55.82	18.18
Physical reads:	1,957.83	639.28
Physical writes:	3.15	1.03
User calls:	41.65	13.57
Parses:	17.65	5.75
Hard parses:	3.98	1.30
Sorts:	2.81	0.92
Logons:	0.01	0.00
Executes:	12.69	4.14
Transactions:	3.07	

% Blocks changed per Read:	4.22	Recursive Call %:	41.82
Rollback per transaction %:	0.28	Rows per Sort:	30.53

Instance Efficiency Percentages (Target 100%)

Buffer Nowait %:	100.00	Redo NoWait %:	100.00
Buffer Hit %:	99.97	In-memory Sort %:	100.00
Library Hit %:	93.28	Soft Parse %:	77.47
Execute to Parse %:	-39.00	Latch Hit %:	99.89
Parse CPU to Parse Elapsd %:	42.19	% Non-Parse CPU:	94.48

Top 5 Timed Events

Event	Waits	Time (s)	% Total Ela Time
direct path read	3,735,593	67,306	60.85
PX Deq: Execute Reply	160,721	36,171	32.70
CPU time		4,322	3.91
PX Deq: Signal ACK	25,803	1,153	.58
log file sync	25,124		.32

图3　优化后Statspack性能分析报表（节选）

【参考文献】
1. 丁铖，Oracle8/8i数据库系统管理，人民邮电出版社，2001
2. Thomas kyte, Orcale专家高级编程,清华大学出版社 ，2002
3. Richard J.Niemiec,Oracle9i性能调整，清华大学出版社，2004