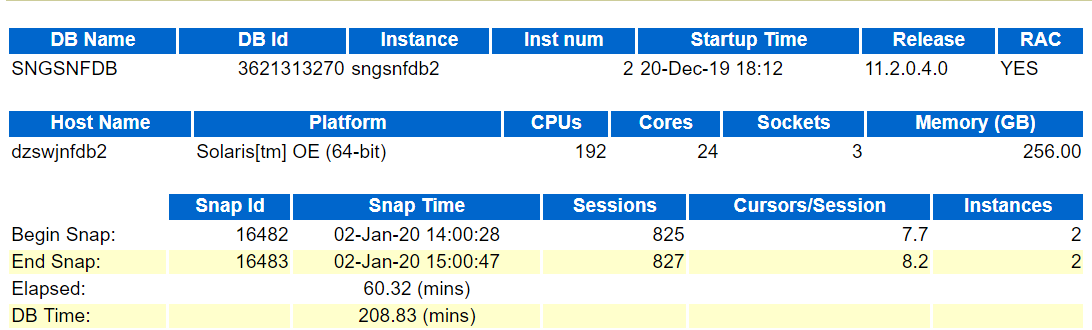
# **WORKLOAD REPOSITORY report for**



192\*60 >>208 说明数据库不忙

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DB Name** | **DB Id** | **Instance** | **Inst num** | **Startup Time** | **Release** | **RAC** |
| SJYYDB | 1864636785 | sjyydb | 1 | 12-Dec-17 20:12 | 11.2.0.4.0 | NO |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Host Name** | **Platform** | **CPUs** | **Cores** | **Sockets** | **Memory (GB)** |
| sjyydb | Linux x86 64-bit | 96 | 48 | 4 | 251.81 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Snap Id** | **Snap Time** | **Sessions** | **Cursors/Session** |
| Begin Snap: | 3162 | 26-Jan-18 11:00:47 | 527 | 5.9 |
| End Snap: | 3163 | 26-Jan-18 12:00:14 | 160 | 13.8 |
| Elapsed: |  | 59.44 (mins) |  |  |
| DB Time: |  | 7,413.87 (mins) |  |  |

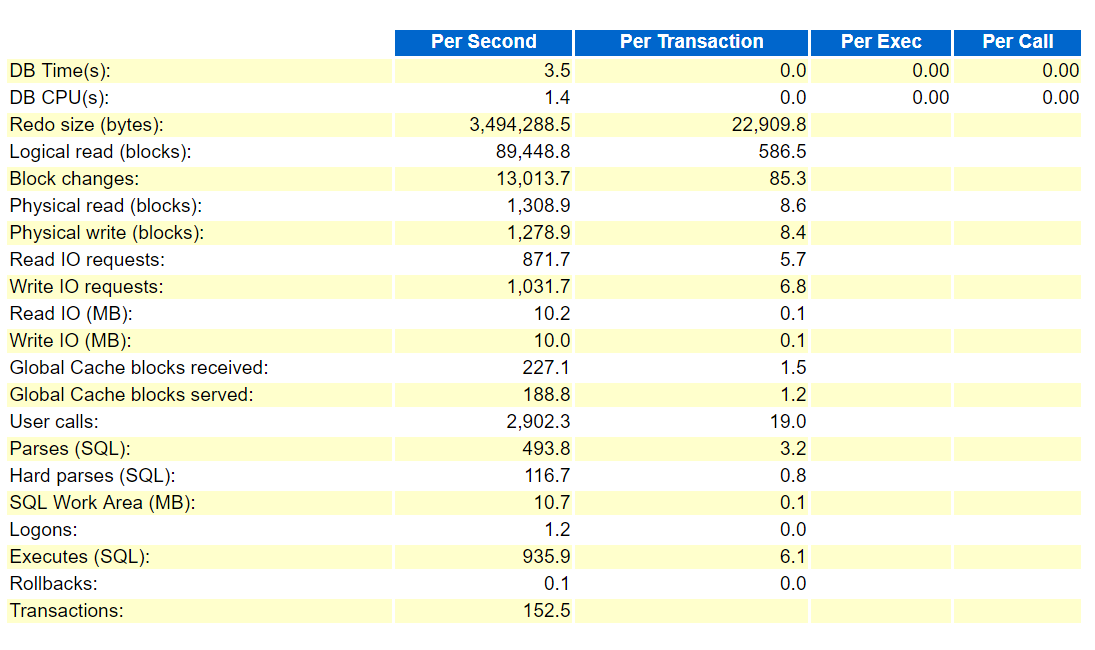
分析：

数据库较忙：Elapsed\*CPUs <DB Time

例：59.44\*96<7,413.87 表示数据库近较忙

## **Report Summary**

Load Profile



--分析

显示数据库负载概况，将之与基线数据比较才具有更多的意义，如果每秒或每事务的负载变化不大，说明应用运行比较稳定。单个的报告数据只说明应用的负载情况，绝大多数据并没有一个所谓“正确”的值，然而Logons大于每秒1~2个、Hard parses大于每秒100、全部parses超过每秒300表明可能有争用问题。

Redo size：每秒/每事务产生的redo大小（单位字节），可标志数据库任务的繁重程序。

Logical reads：每秒/每事务逻辑读的块数

Block changes：每秒/每事务修改的块数

Physical reads：每秒/每事务物理读的块数

Physical writes：每秒/每事务物理写的块数

User calls：每秒/每事务用户call次数

Parses：SQL解析的次数

Hard parses：其中硬解析的次数，硬解析太多，说明SQL重用率不高。

Sorts：每秒/每事务的排序次数

Logons：每秒/每事务登录的次数

Executes：每秒/每事务SQL执行次数

Transactions：每秒事务数

Blocks changed per Read：表示逻辑读用于修改数据块的比例

Recursive Call：递归调用占所有操作的比率

Rollback per transaction：每事务的回滚率

Rows per Sort：每次排序的行数

注:

Oracle的硬解析和软解析

　　提到软解析(soft prase)和硬解析(hard prase)，就不能不说一下Oracle对sql的处理过程。当你发出一条sql语句交付Oracle，在执行和获取结果前，Oracle对此sql将进行几个步骤的处理过程：

　　1、语法检查(syntax check)

　　检查此sql的拼写是否语法。

　　2、语义检查(semantic check)

　　诸如检查sql语句中的访问对象是否存在及该用户是否具备相应的权限。

　　3、对sql语句进行解析(prase)

　　利用内部算法对sql进行解析，生成解析树(parse tree)及执行计划(execution plan)。

　　4、执行sql，返回结果(execute and return)

　　其中，软、硬解析就发生在第三个过程里。

　　Oracle利用内部的hash算法来取得该sql的hash值，然后在library cache里查找是否存在该hash值；

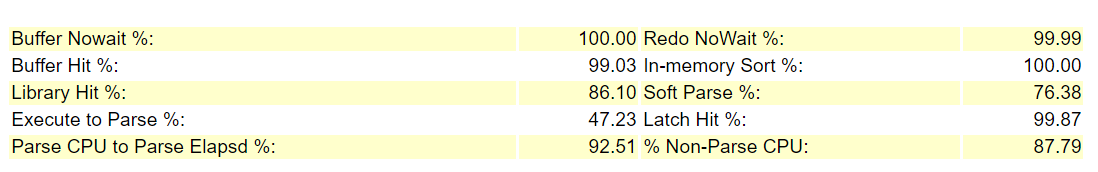
　　假设存在，则将此sql与cache中的进行比较；

　　假设“相同”，就将利用已有的解析树与执行计划，而省略了优化器的相关工作。这也就是软解析的过程。

　　诚然，如果上面的2个假设中任有一个不成立，那么优化器都将进行创建解析树、生成执行计划的动作。这个过程就叫硬解析。

　　创建解析树、生成执行计划对于sql的执行来说是开销昂贵的动作，所以，应当极力避免硬解析，尽量使用软解析。

Instance Efficiency Percentages (Target 100%)



--分析

1. Buffer Hit --缓存命中率，理想应该在90%以上
2. Buffer Nowait --获取缓存的比率，低于80%，则需要分配更多的内存
3. Redo NoWait --redo buffer 从SGA获取的比率，低于90%，考虑增加log buffer。
4. Library Hit --存放sql的解析及执行计划，过多的解析会增加CPU消耗，若低于90%，建议增大shared pool
5. Execute to Parse --语句解析比率，若SQL重用率高，则这个比率会很高，值越高，表示一次解析后被重复执行的次数越多。
6. In-memory Sort --内存中排序的比率，值过低，说明会在临时表空间中进行，需增大PGA。
7. Soft Parse --软解析的百分比，sql在共享区的命中率，太低，则需要使用绑定变量

本节包含了Oracle关键指标的内存命中率及其它数据库实例操作的效率。其中Buffer Hit Ratio 也称Cache Hit Ratio，Library Hit ratio也称Library Cache Hit ratio。同Load Profile一节相同，这一节也没有所谓“正确”的值，而只能根据应用的特点判断是否合适。在一个使用直接读执行大型并行查询的DSS环境，20%的Buffer Hit Ratio是可以接受的，而这个值对于一个OLTP系统是完全不能接受的。根据Oracle的经验，对于OLTPT系统，Buffer Hit Ratio理想应该在90%以上。

Buffer Nowait表示在内存获得数据的未等待比例。

buffer hit表示进程从内存中找到数据块的比率，监视这个值是否发生重大变化比这个值本身更重要。对于一般的OLTP系统，如果此值低于80%，应该给数据库分配更多的内存。

Redo NoWait表示在LOG缓冲区获得BUFFER的未等待比例。如果太低（可参考90%阀值），考虑增加LOG BUFFER。

library hit表示Oracle从Library Cache中检索到一个解析过的SQL或PL/SQL语句的比率，当应用程序调用SQL或存储过程时，Oracle检查Library Cache确定是否存在解析过的版本，如果存在，Oracle立即执行语句；如果不存在，Oracle解析此语句，并在Library Cache中为它分配共享SQL区。低的library hit ratio会导致过多的解析，增加CPU消耗，降低性能。如果library hit ratio低于90%，可能需要调大shared pool区。

Latch Hit：Latch是一种保护内存结构的锁，可以认为是SERVER进程获取访问内存数据结构的许可。要确保Latch Hit>99%，否则意味着Shared Pool latch争用，可能由于未共享的SQL，或者Library Cache太小，可使用绑定变更或调大Shared Pool解决。

Parse CPU to Parse Elapsd：解析实际运行时间/(解析实际运行时间+解析中等待资源时间)，越高越好。

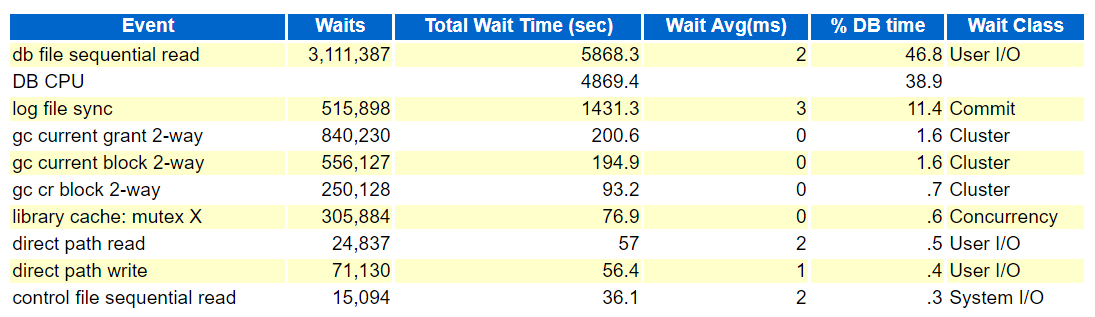
Non-Parse CPU ：SQL实际运行时间/(SQL实际运行时间+SQL解析时间)，太低表示解析消耗时间过多。

Execute to Parse：是语句执行与分析的比例，如果要SQL重用率高，则这个比例会很高。该值越高表示一次解析后被重复执行的次数越多。

In-memory Sort：在内存中排序的比率，如果过低说明有大量的排序在临时表空间中进行。考虑调大PGA。

Soft Parse：软解析的百分比（softs/softs+hards），近似当作sql在共享区的命中率，太低则需要调整应用使用绑定变量。

Top 10 Foreground Events by Total Wait Time

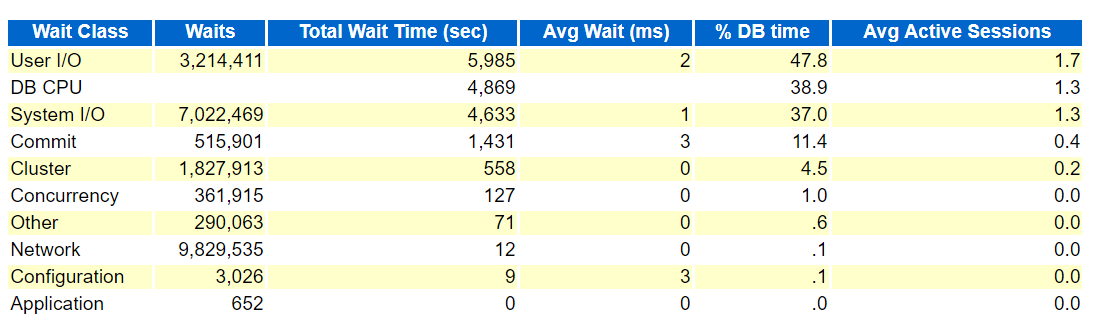


--分析

等待事件

在较好的情况下，CPU TIME 总是在第一个

Wait Classes by Total Wait Time



--分析

等待事件说明：

db file scattered read等待事件是当SESSION等待multi-block I/O时发生的，通过是由于full table scans或 index fast full scans。发生过多读操作的Segments可以在“Segments by Physical Reads”和“SQL ordered by Reads”节中识别（在其它版本的报告中，可能是别的名称）。如果在OLTP应用中，不应该有过多的全扫描操作，而应使用选择性好的索引操作。

DB file sequential read等待意味着发生顺序I/O读等待（通常是单块读取到连续的内存区域中），如果这个等待非常严重，应该使用上一段的方法确定执行读操作的热点SEGMENT，然后通过对大表进行分区以减少I/O量，或者优化执行计划（通过使用存储大纲或执行数据分析）以避免单块读操作引起的sequential read等待。通过在批量应用中，DB file sequential read是很影响性能的事件，总是应当设法避免。

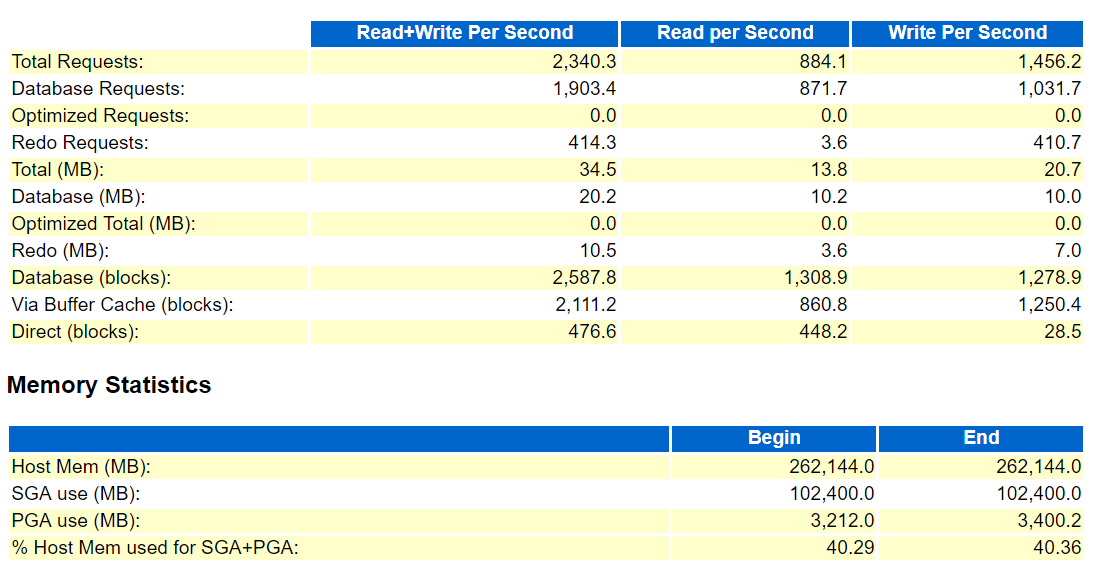
Log File Parallel Write事件是在等待LGWR进程将REDO记录从LOG 缓冲区写到联机日志文件时发生的。虽然写操作可能是并发的，但LGWR需要等待最后的I/O写到磁盘上才能认为并行写的完成，因此等待时间依赖于OS完成所有请求的时间。如果这个等待比较严重，可以通过将LOG文件移到更快的磁盘上或者条带化磁盘（减少争用）而降低这个等待。

Buffer Busy Waits事件是在一个SESSION需要访问BUFFER CACHE中的一个数据库块而又不能访问时发生的。缓冲区“busy”的两个原因是：1）另一个SESSION正在将数据块读进BUFFER。2）另一个SESSION正在以排它模式占用着这块被请求的BUFFER。可以在“Segments by Buffer Busy Waits”一节中找出发生这种等待的SEGMENT，然后通过使用reverse-key indexes并对热表进行分区而减少这种等待事件。

Log File Sync事件，当用户SESSION执行事务操作（COMMIT或ROLLBACK等）后，会通知 LGWR进程将所需要的所有REDO信息从LOG BUFFER写到LOG文件，在用户SESSION等待LGWR返回安全写入磁盘的通知时发生此等待。减少此等待的方法写Log File Parallel Write事件的处理。

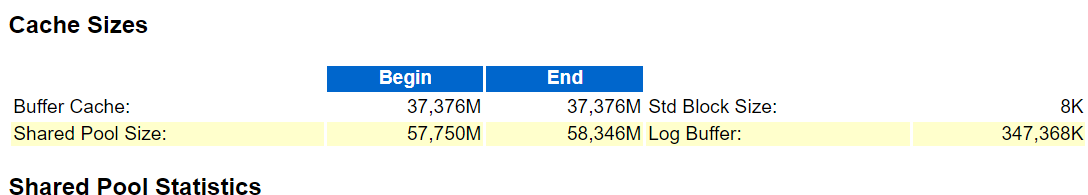
Enqueue Waits是串行访问本地资源的本锁，表明正在等待一个被其它SESSION（一个或多个）以排它模式锁住的资源。减少这种等待的方法依赖于生产等待的锁类型。导致Enqueue等待的主要锁类型有三种：TX（事务锁）, TM D（ML锁）和ST（空间管理锁）。

IO Profile



--观察SGA的使用情况

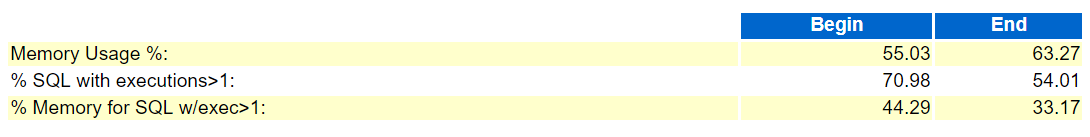
如果在数据库比较忙的情况下，那么SGA可能被耗尽，在[init.ora Parameters](C:/Users/svsky95/Desktop/%E6%A0%B8%E5%BF%83%E5%BE%81%E7%AE%A1/awr_cx0226.html" \l "36)标签中，可以看到当前SGA的大小，若小，需要调整大些。



--分析

显示SGA中每个区域的大小（在AMM改变它们之后），可用来与初始参数值比较。

shared pool主要包括library cache和dictionary cache。library cache用来存储最近解析（或编译）后SQL、PL/SQL和Java classes等。library cache用来存储最近引用的数据字典。发生在library cache或dictionary cache的cache miss代价要比发生在buffer cache的代价高得多。因此shared pool的设置要确保最近使用的数据都能被cache。



--分析

1、Memory Usage --内存使用率，75%-90%之间，值太小浪费，值过高，则说明共享池中有征用，内存不足。

2、SQL with executions>1 --执行次数大于1的SQL的占比率，值过小，则说明需要采用绑定变量

3、Memory for SQL w/exec>1 --执行次数大于1的SQL的内存占比

--TOP SQL 优化

1、一个系统中，CPU资源是系统性能瓶颈所在，那么优化buffer gets最多的SQL语句将获得最大效果。

2、一个I/O等待是最严重事件的系统中，调优的目标应该是physical IOs最多的SQL语句。

在STATSPACK报告中，没有完整的SQL语句，可使用报告中的Hash Value通过下面语句从数据库中查到：

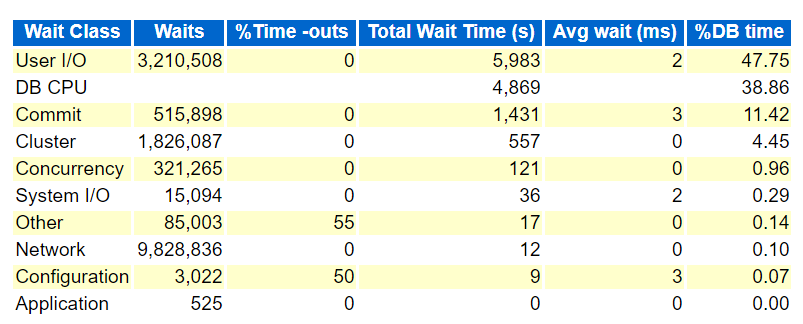
select sql\_text

from stats$sqltext

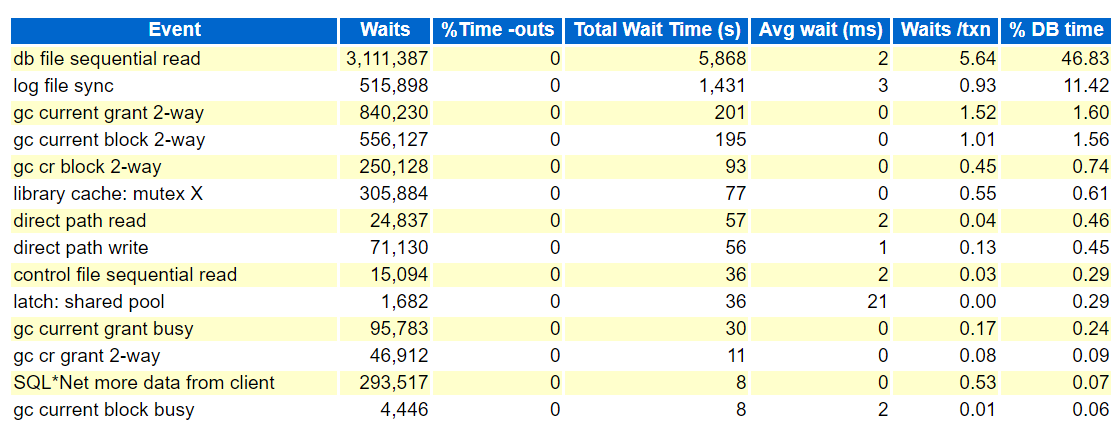
where hash\_value = &hash\_value

1. 如果有相同的SQL排在几个SQL状态的前面，那基本上就是优化的对象，需要重点优化。

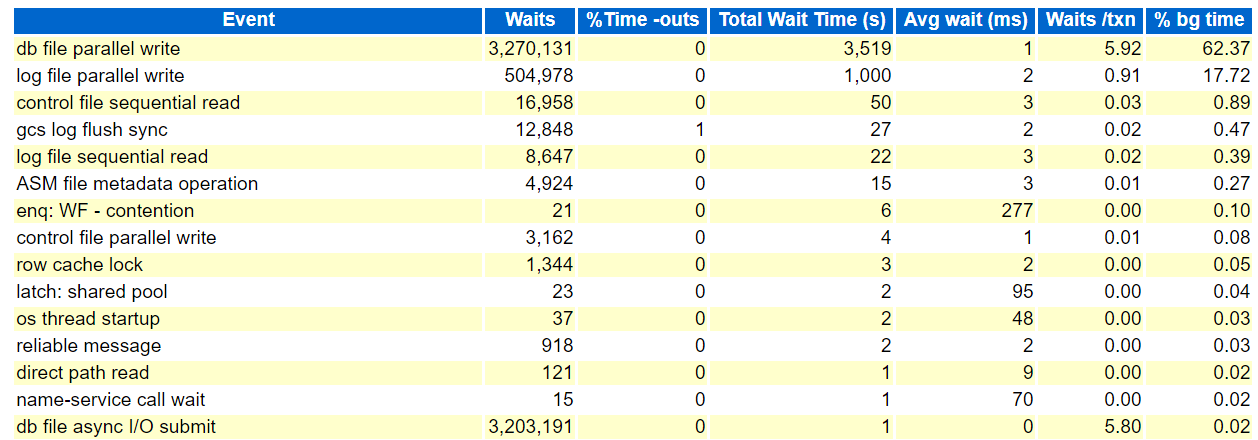
### **Foreground Wait Class**



### **Foreground Wait Events**

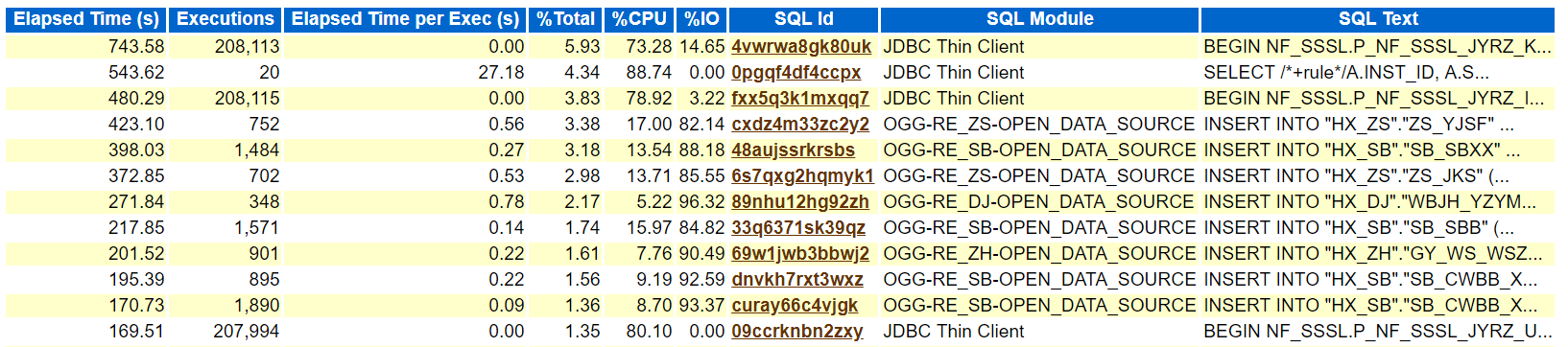


### **Background Wait Events**



## **SQL Statistics**

### **SQL ordered by Elapsed Time**

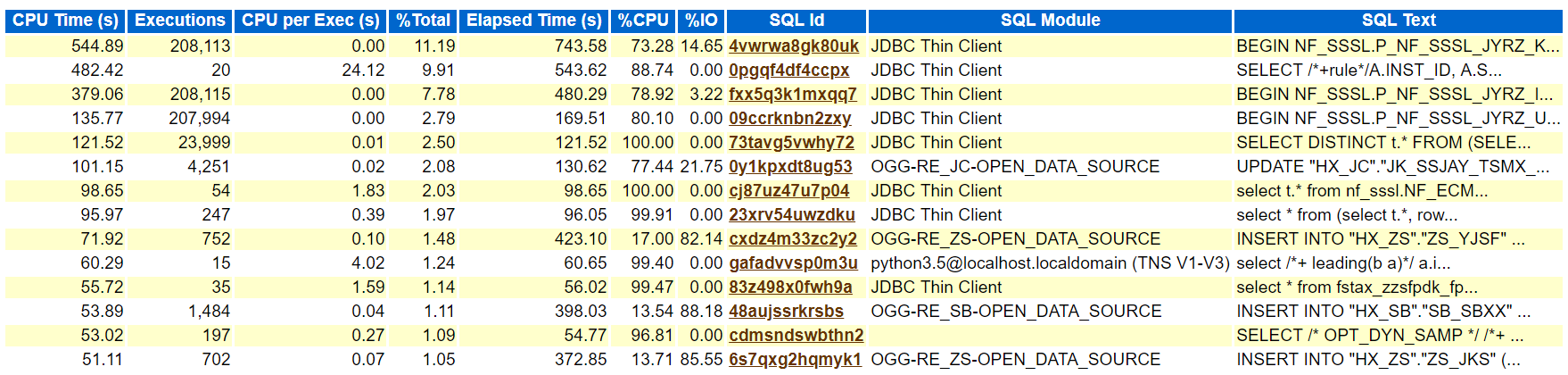


--关于SQL消耗的时间，CPU较高，可能是走了全表扫描或者索引的效率不高导致的。

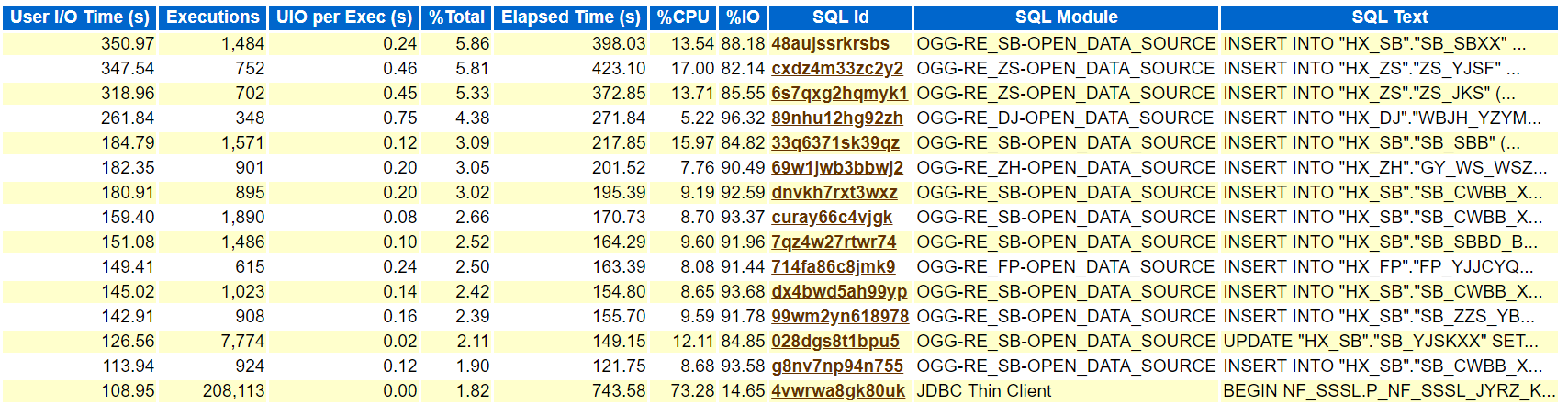
同样也要根据执行的总次数及平均耗时来确定。

IO较高基本上会有磁盘写入等操作。

**SQL ordered by CPU Time**

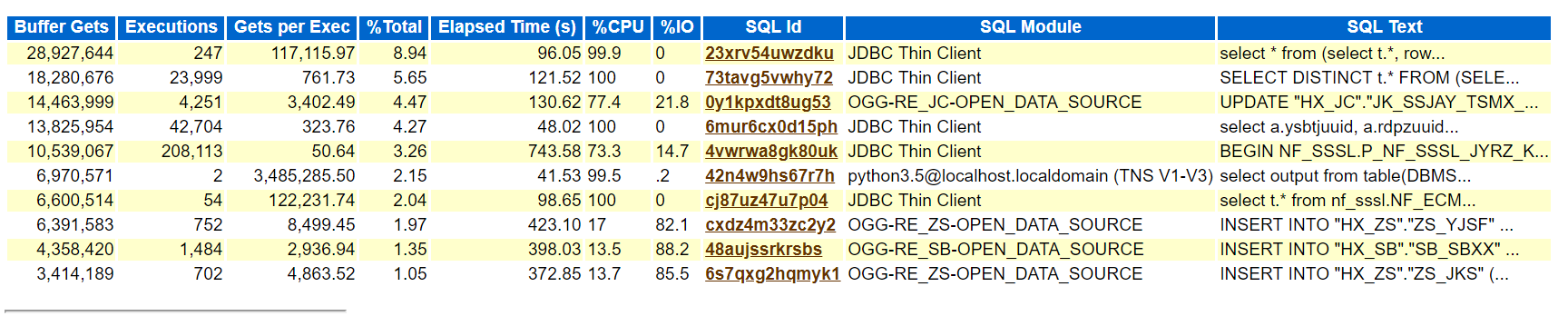


### **SQL ordered by User I/O Wait Time**

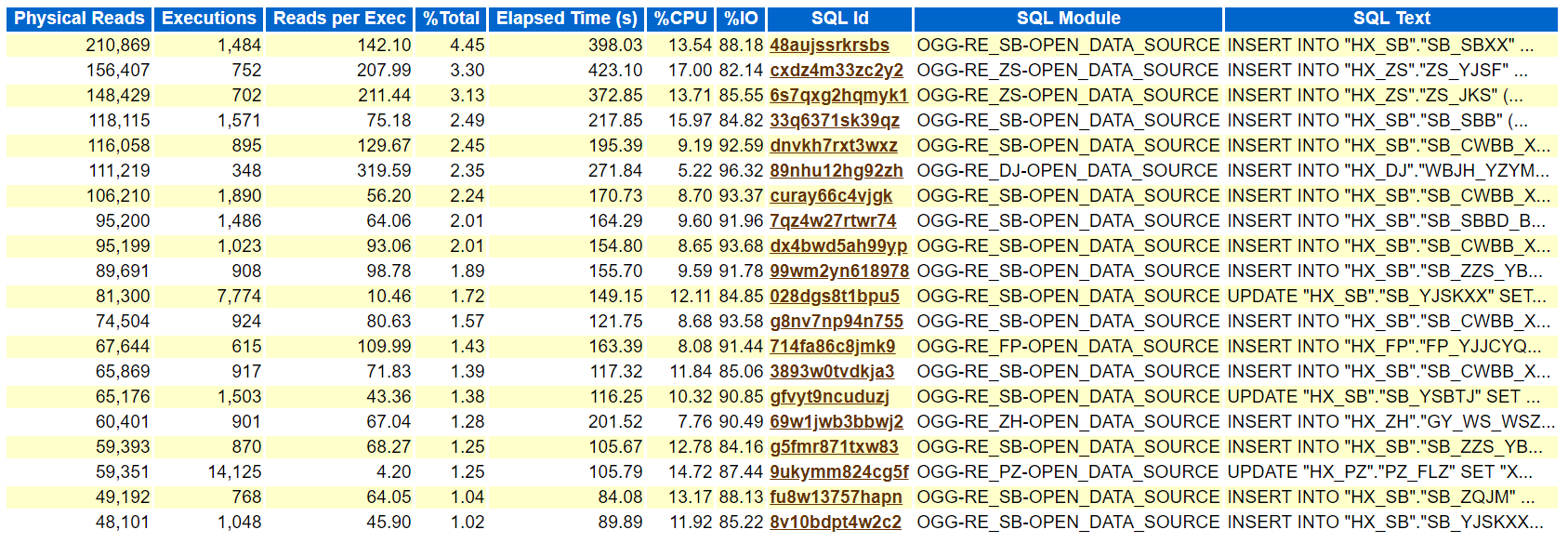


--重点关注IO，从上图可以看出，IO基本上都是insert和update操作。

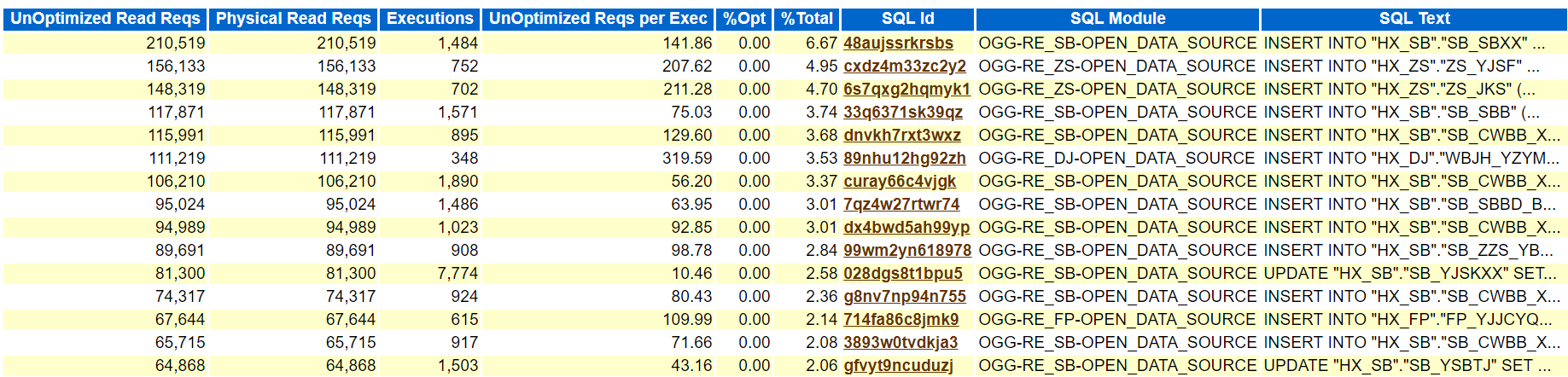
### **SQL ordered by Gets**



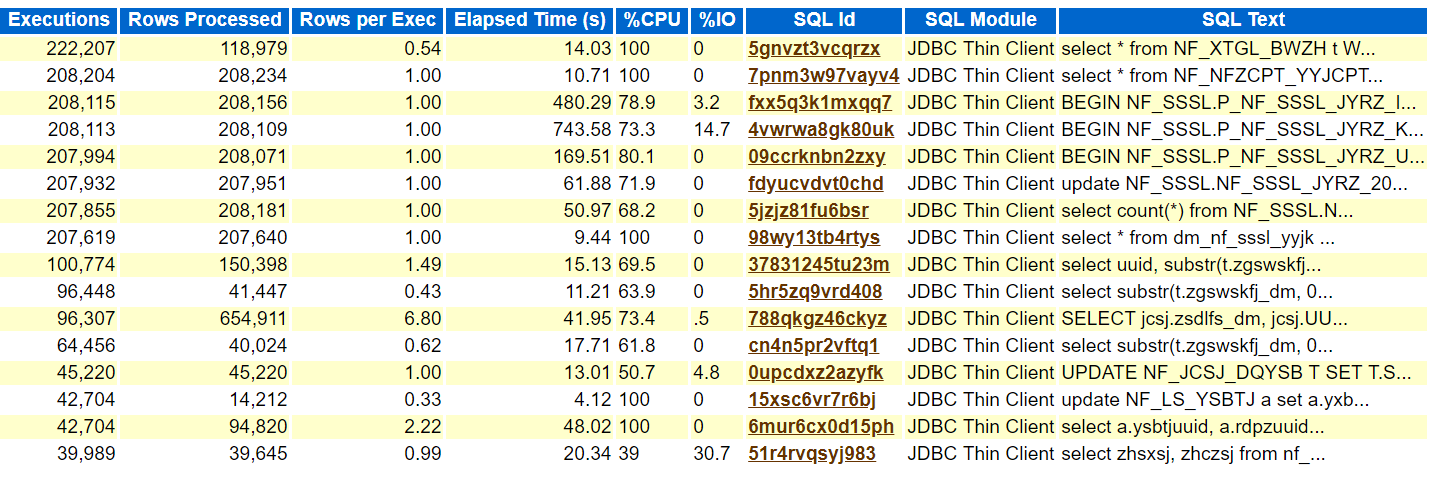
### **SQL ordered by Reads**



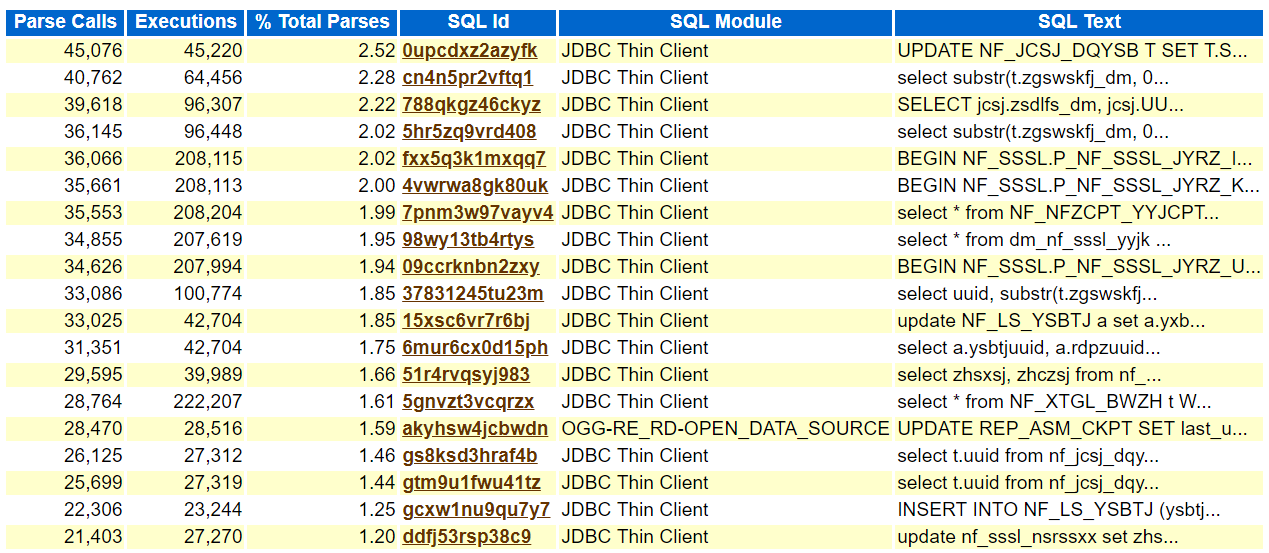
### **SQL ordered by Physical Reads (UnOptimized)**



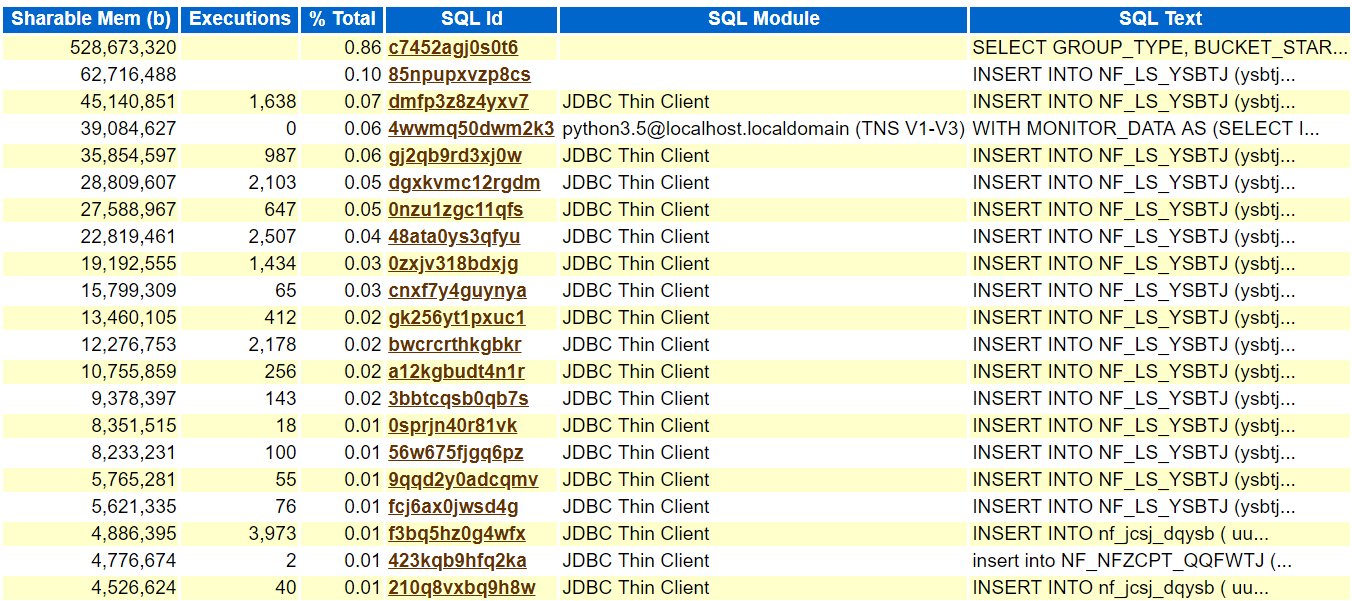
### **SQL ordered by Executions**



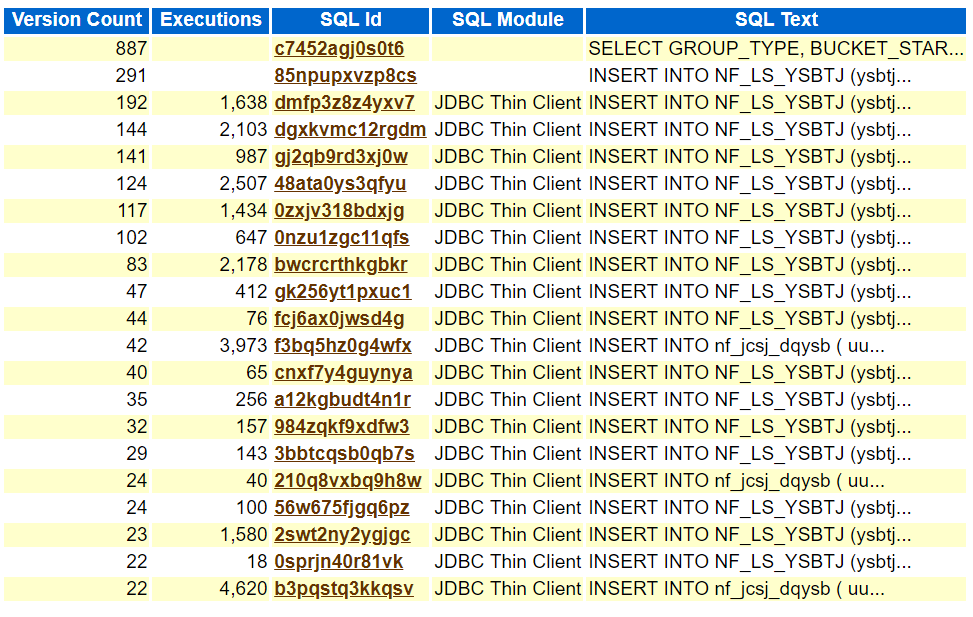
### **SQL ordered by Parse Calls**



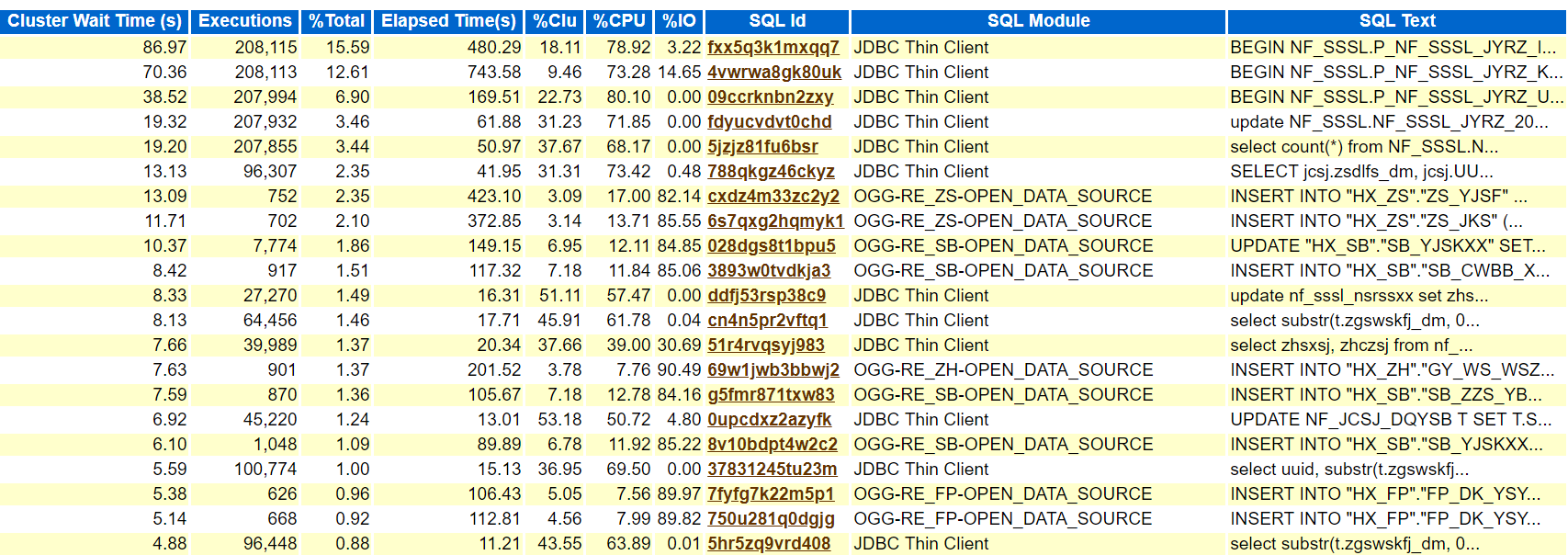
### **SQL ordered by Sharable Memory**



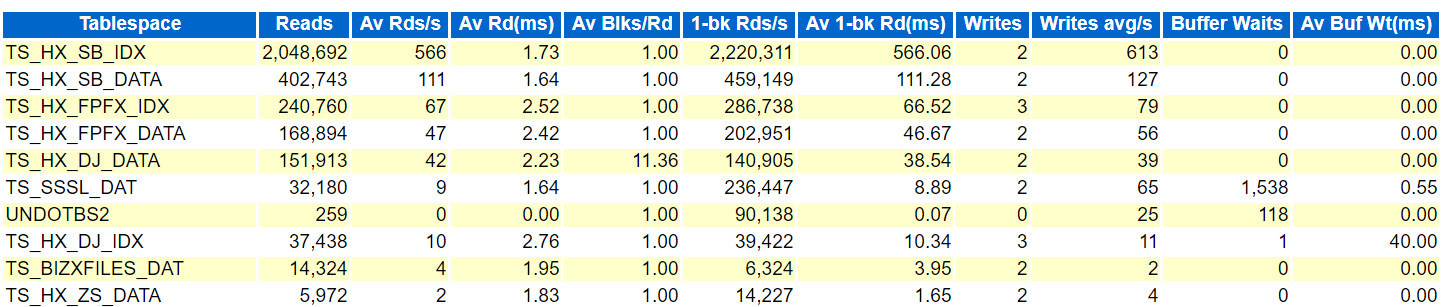
### **SQL ordered by Version Count**



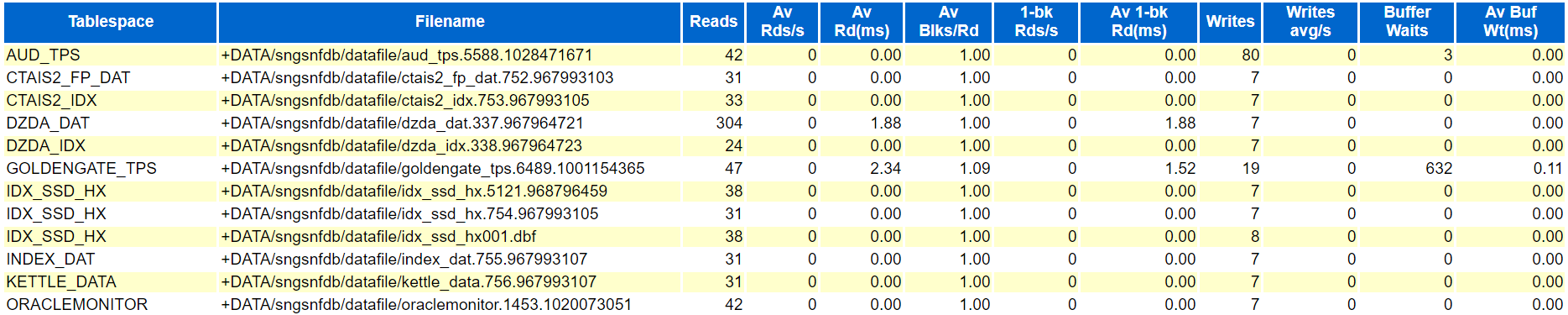
### **SQL ordered by Cluster Wait Time**



## **Tablespace IO Stats**



### **File IO Stats**

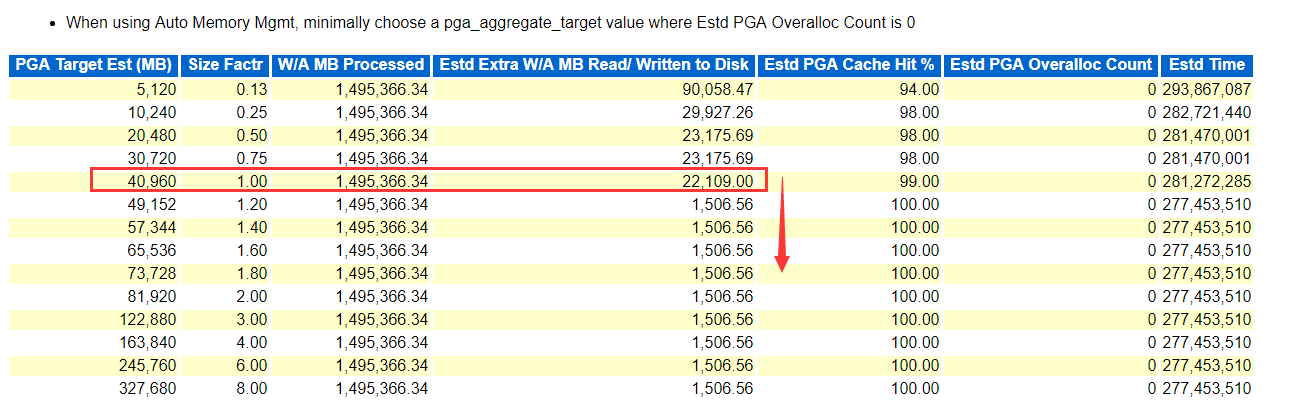


--分析

Av Rd(ms) [Average Reads in milliseconds]不应该超过30，否则认为有I/O争用。

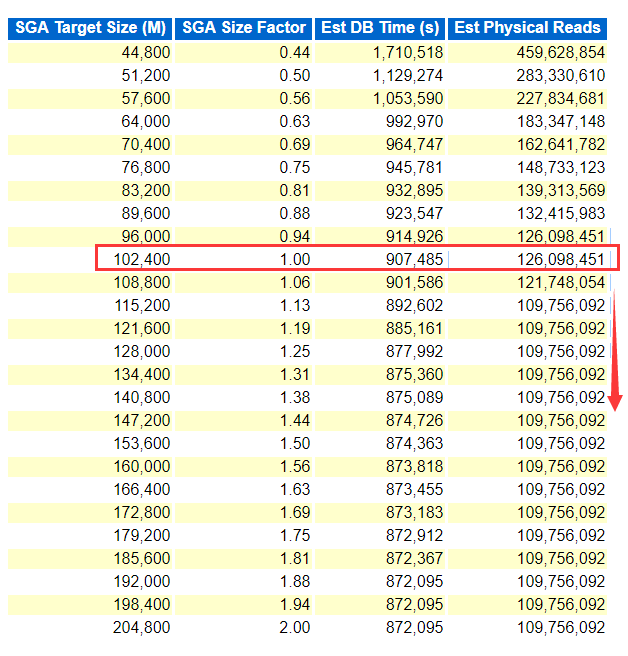
## **Advisory Statistics**

### **PGA Memory Advisory**



从size factr 1.0开始，write to disk，明显减少，从而也就是PGA足够，不再使用临时表空间，所以PGA应从40G调整到50G。

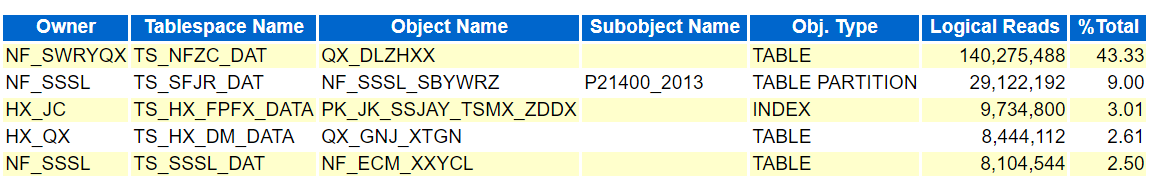
### **SGA Target Advisory**



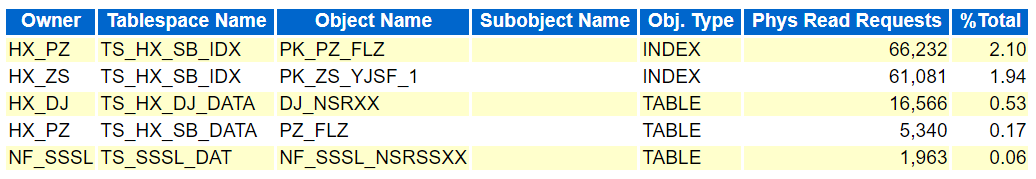
从1.0开始，降低了物理读的指数，并不是很大，如果想调整，可以调制120G

## **Segment Statistics**

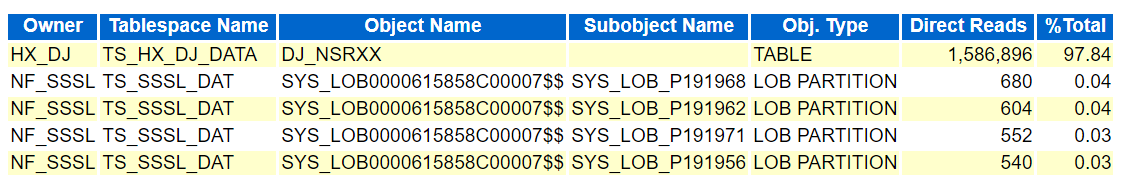
### **Segments by Logical Reads**



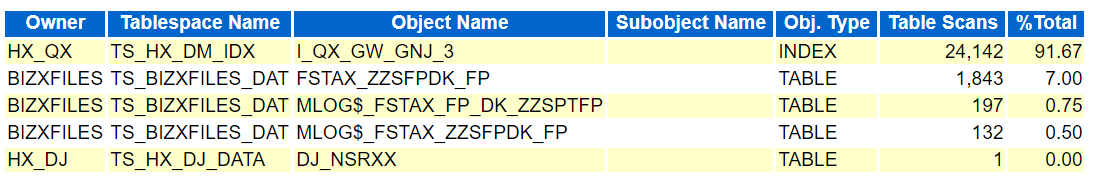
### **Segments by Physical Reads**



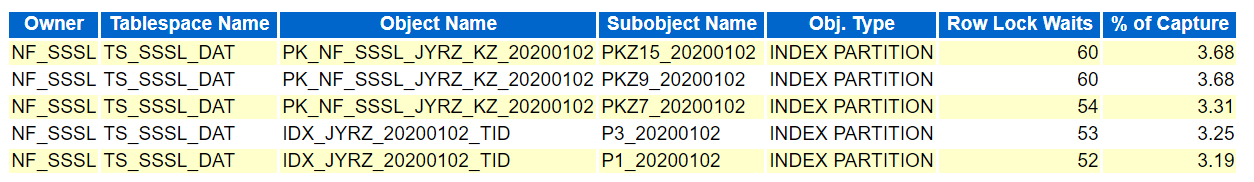
### **Segments by Direct Physical Reads**



### **Segments by Table Scans**

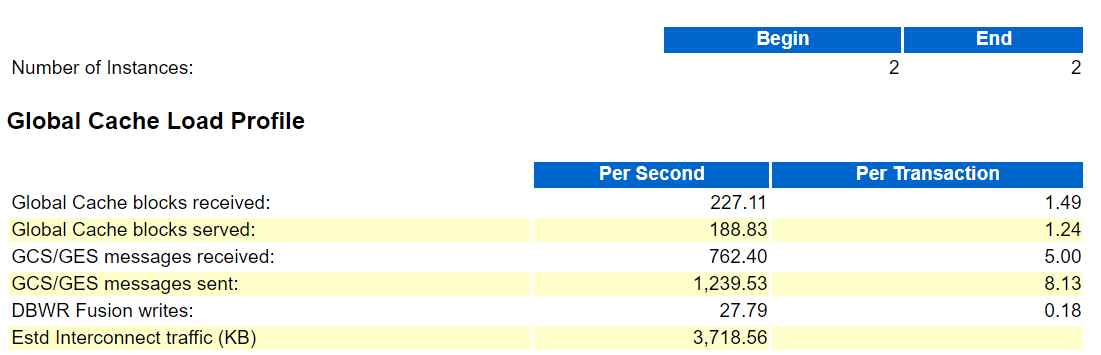


### **Segments by Row Lock Waits**



当一个进程予在正被其它进程锁住的数据行上获得排它锁时发生这种等待。这种等待经常是由于在一个有主键索引的表上做大量INSERT操作。

## **RAC Statistics**



--分析

共享缓存在两个实例之间的数据传输。

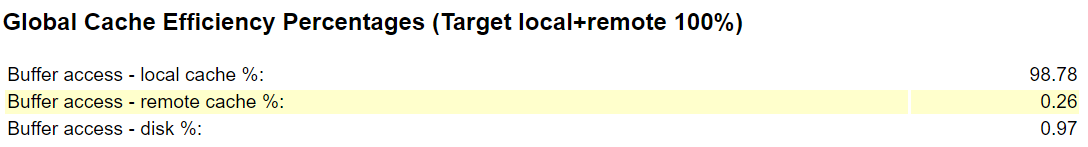
数据量结算方法：

Global Cache blocks received=0.18\*8K/1024/1024 (M)

Estd Interconnect traffic (KB)

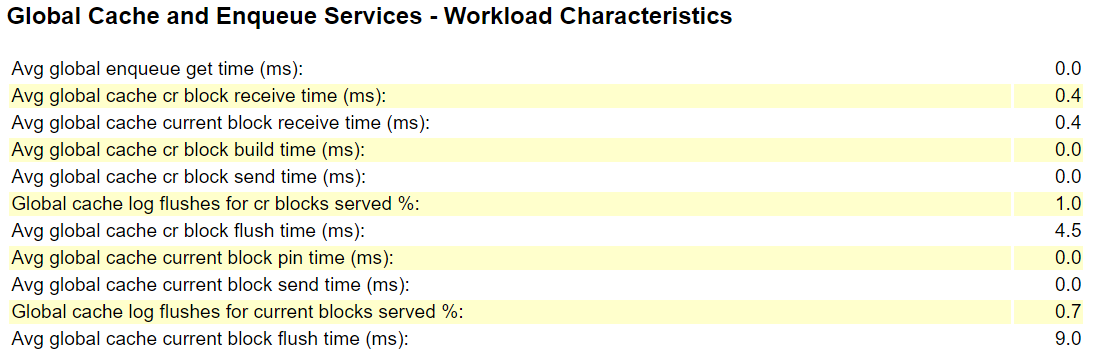
两个节点私有网络之间的流量统计

Global Cache Efficiency Percentages (Target local+remote 100%)



本地缓存的命中率很高，而远程的命中率因该在10%以下，大于则需要进行程序分实例方案。

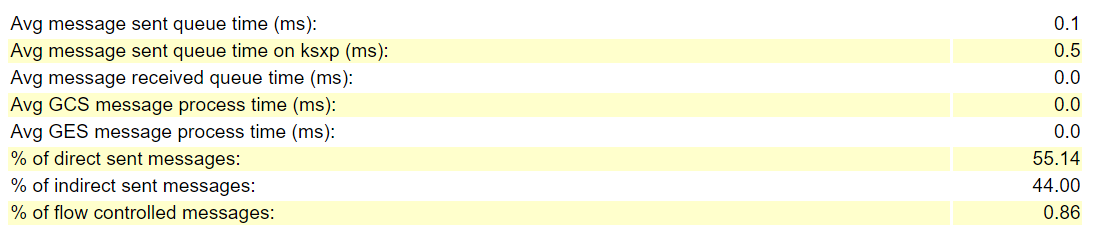
--描述队列和缓存的情况



所有与CR块（consist read 一致读块）相关的参数应该低于5毫秒。

所有与当前块（current block） 相关的参数应当小于10毫秒

Global Cache and Enqueue Services - Messaging Statistics



所有的统计信息应当小于1毫秒

**Segments by Row Lock Waits**

--占比较高，有可能存在争用

