awr报告绝非是要用户每次都从头到尾读一遍，而是要根据用户根据自己的实际情况从报告中需要有用的信息

比如对于oltp系统：

         \*Library Hit

         \*Buffer Hit

          这两项要非常关注，应为oltp是一个sql非常密集的系统，共享池命中低说明很多sql不能被重用，需要重新解析，这会大大降低系统性能和sql执行效率。

          Buffer Hit也非常重要，oltp系统要求sql执行效率高，sql需要数据块能保持在内存中，那么sql执行效率自然比从磁盘读取数据块要高很多，当这个值接近100时，说明内存中sql访问的数据块越多，也就是磁盘读取的越少

但是如果你是一个olap系统，则基本上可以忽略这两个参数。

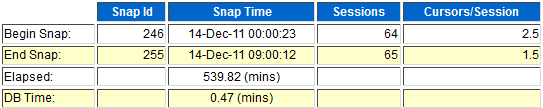
**第一部分AWR报告反映数据库的信息**

让我们来分析一个awr报告

          这是报告的第一部分，它包含了数据和实例的一个基本信息，如果是一个rac结构，RAC选项为YES，最好对每个实例做性能分析。

          这部分是采集周期里系统的一个概述，要注意下面三个列的含义：

﻿﻿



1、  sessions

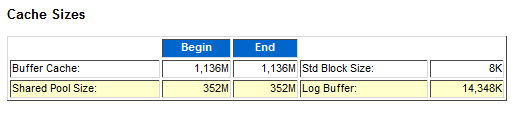
表示采集是实例连接的会话数，这个数可以让我们了解数据库并发用户的大概情况。如果是新接手的数据库，对判断数据库的类型可以做参考

2、  Cursors/Session，平均每个会话卡开的游标数。

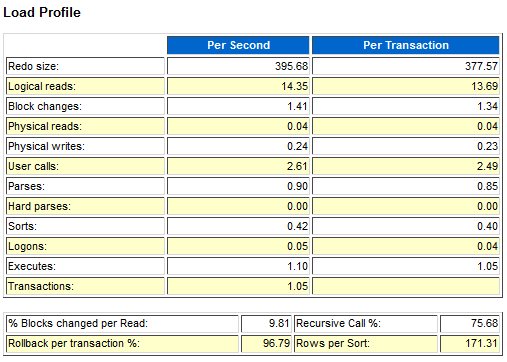
3、  DB Time

4、  这个数值比较重要，它表示用户操作花费的时间，包括cpu和等待事件。有时候DB Time会比Elapsed时间要长。因为AWR是一个数据的合集，比如说1分钟内一个用户等待10秒钟，那么10个用户是300秒（5分钟）；cpu的时间也是一样一分钟之内，一个cpu处理30秒，那么4个cpu就是1.2分钟，8个就是2.4分钟，这些都以累计的方式记录在awr报告当中的。

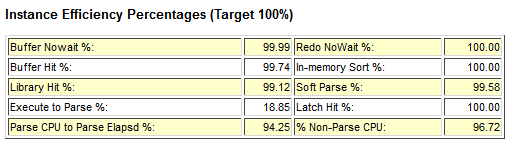
**Report Summary**



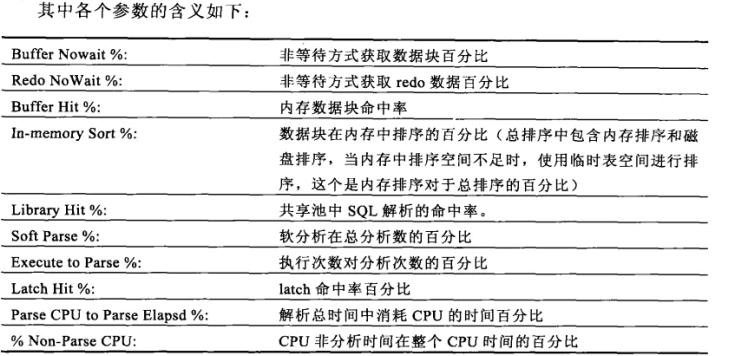
         这个表列出从开始采集到结束的时候数据缓冲池（Bubber Cache）和共享池（Shared Pool Size）的大小。



         这个是共享池的一个明细列表，分割为每秒钟资源负载和一个事物资源负载情况



         这一部分是内存效率的统计信息，我个人觉得对于oltp系统来说，这个意义比较重大，这些值应该尽可能的解决100，execute to parse这里很低是不是数据库有问题了呢看下面的一段话no,这不是绑定变量的问题,从楼主提供的比例来看,变量绑定的还是比较好的。“这个比例是由于太多的soft parse导致的.对于比如web数据库,这个值很小是正常的,因为用户要反复链接,每一次链接,及时相同的sql,都会有一次soft parse.这个参数是解决所谓的softer soft parse的”

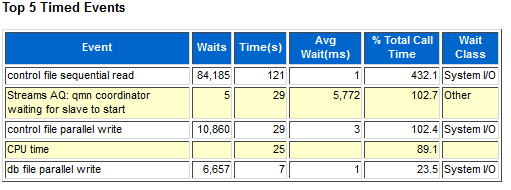


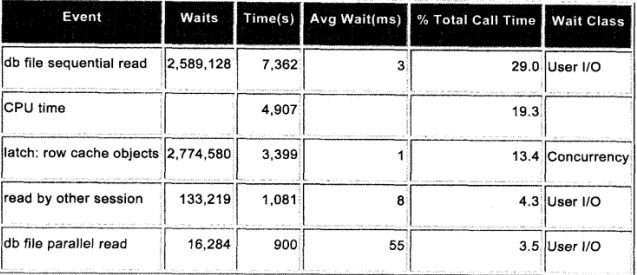
        其他的，比如说Soft Parse比例太低，说明sql没有重用，可能没有采用绑定变量，再比如说Buffer Hit 太低，说明很多数据块没有缓存到内存中。要考虑增大sga尺寸，再比如Buffer Nowait的值太小，就说明发生的sql访问数据块时正在被别的会话读入内存，需要等待这个操作完成，发生这样的事情通常是某些数据块变成了热快（hot block）。

**Top 5 Timed Events**

         这一部分的内容为awr报告最重要的一部分，基本上每次看awr报告，首先看到这里。

这一部分前5个时间一共也没有等待多长时间，这个awr报告就没必要看下去了，因为系统状况非常好—几乎没有太长的等待操作，所以性能上不需要做性能优化。这里从网上找了了一个等待事件，来一起看一下。

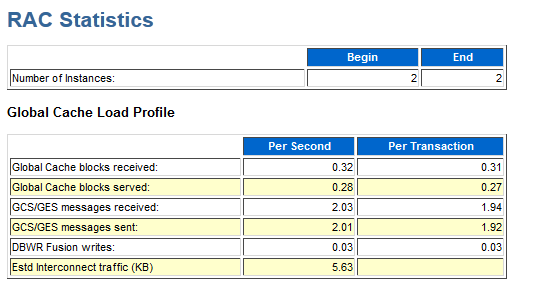




         我们看到排在第一位的dbfile sequential read为7362秒（超过2小时），如果查看对这个等待事件的解释就可以知道，它通常是指oracle在访问索引数据块会以db file sequential read方式来把数据块读入内存。让我们看看%TotalCall Time事件为29.0%，说明db file sequential read占整个dbtime的29.0%，我们可以猜到数据块中一定运行非常多的大查询，才导致时间远远超过了采集时间（1个小时）这个报告中snap time为一个小时（在报告的刚开始可以看到）。

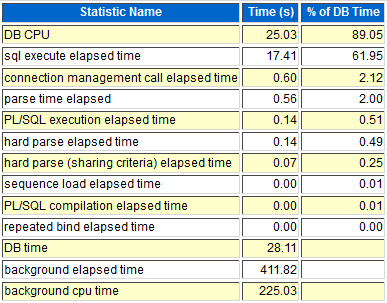
Cputime也比较高。

另外一个等待事件：Latch:rowcache objects，这是是共享池争用。看看这个等待的类型是Concurrency（并发），也就是说多个会话可能存在争用相同的资源。



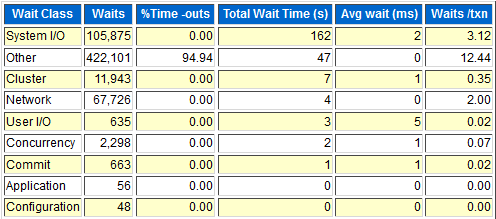
这一部分只有rac环境才会出现，是一些全局内存中发送数据、接受方面性能指标，还有一些全局锁的信息，除非这个数据库在运行正常时设定了一个基准线作为参照，否则这一部分性能指标很难说是否有性能问题。

**Time Model Statistics**



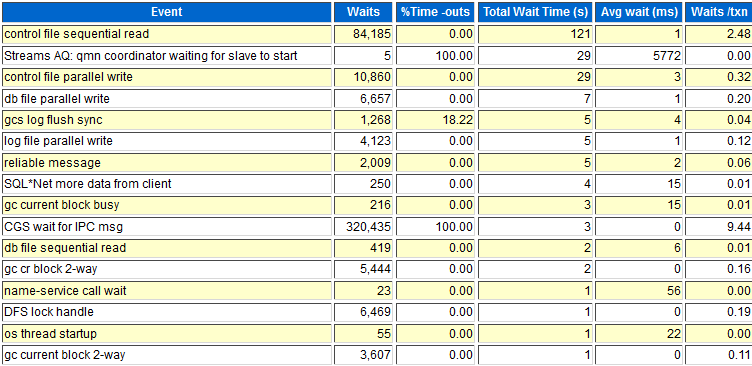
         这部分信息列出了各种操作占用的数据库时间比例，在整个db time28.11秒中，sql执行的时间为61.95秒。

**Wait class**



         这一部分又从另外一个角度帮助我们分析和确定等待的事件。其实在一个awr报告中，其实性能可能只是由一个原因所引起的，在这里比如说磁盘的io不够，再次说明awr报告完全不需要顺序的从头读到尾（当时每个表大概内容讲的什么还是要知道的）。

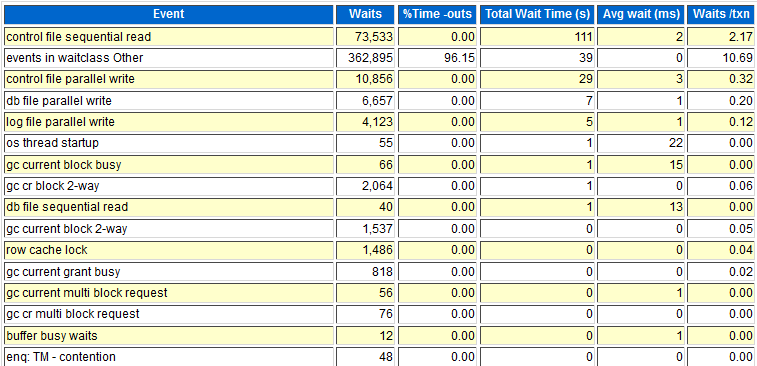
**Wait events**



。。。。

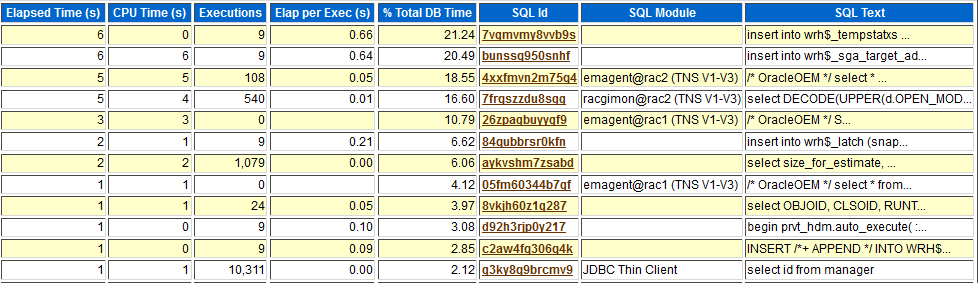
         截取了一部分，这一部分是整个实例等待事件的明细，它包括了top5等待时间的信息，这部分只是比如说top5提供的信息依然不足以说明问题所在的情况下使用的。

**Background wait events**



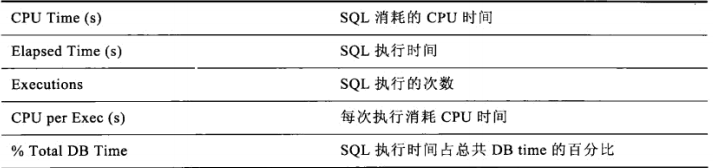
         后台等待事件，这一部分也是只有需要才会用到，比如说我们怀疑一年过后的操作可能是由于后台某个进程（比如dbwr）无法及时响应导致，那么需要到这里来确认是否有后台进程等待时间太长的事件存在。

**Sql statistics**



         这部分内容要结合I/O方面的信息。以及top5里面的等待事件

内容含义



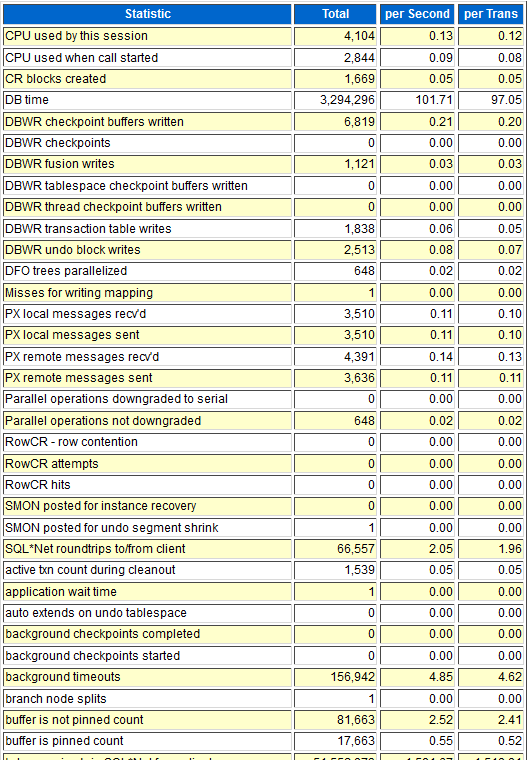
这里面分别正对了elapsed time、cpu time、gets、reads、executions、parse calls、Version Count、Cluster Wait Time排列

如果是oltp系统，可以多关注executions（执行次数）、parse calls（解析次数）进行排列的sql。

Complete List ofSQL Text

这一部分列出了上面各个部分设计sql的完整信息，如果awr报告是html格式，可以从上面各部分的sql\_id例通过超级链接直接找到对应的sql完整语句，非常方便，索引awr报告选择一般选择html格式（默认）。

**Instance Activity Statistics**



这一部分是实例统计，项目非常多，要重点讨论下

**CPU used by this session**

http://hi.csdn.net/attachment/201112/15/0_13239519075vXZ.gif

这里还要结合cpus的个数，让我们看看Operating System Statistics里面num\_cpus

         这里cpu的个数为24个，现在我们来计算cpu的相关情况

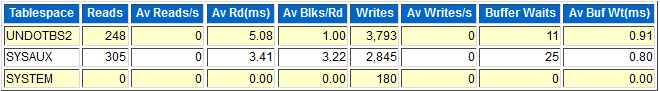
这一部分是以cpu单位（cpu units）为单位的显示方式，整个过程消耗了4104个cpu单位，

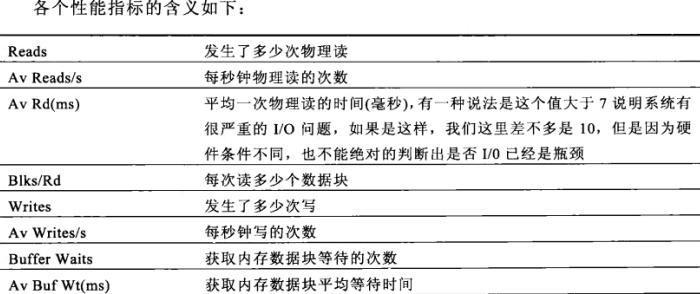
每秒钟的cpu单位为：0.13，对应的时间为：0.13/100=0.0013秒，也就是每秒钟cpu的处理时间是0.0013秒，我们系统有24个cpu，每秒钟cpu消耗的cpu单位为0.0013/24=5.416666666666667e-5，这样看来cpu资源是非常丰富的，远远没有出现瓶颈。

         但是这里是rac结构，我们要去每个实例上面做awr报告，进行综合分析。

**IO Stats**

**Tablespace IO Stats**





**File IO Stats**

这部分和上面类似。

**第二部分是oracle给出一些关于各个内存组建大小的建议**

这些建议是oracle通过自身设置一个模拟环境，把内存组建设置不同的大小的建议，对于这种改变造成的相关方面的性能影响进行估算，最后将这个估算清单提交给我们。

**Advisory Statistics**

Buffer Pool Advisory

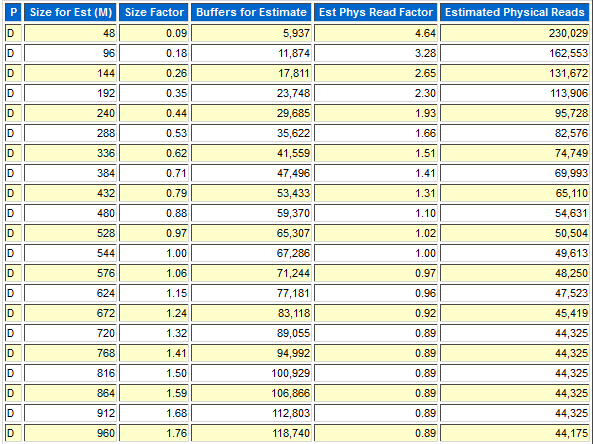
PGA Memory Advisory

Shared Pool Advisory

SGA Target Advisory

这几部分并不能帮助我们直观的定位系统问题，但是它会给我们一些关于oracle内存大小的建议，所以我们应该关注一下这里，以便知道当前数据库在这方面的设置是否合理。

**Buffer Pool Advisory**

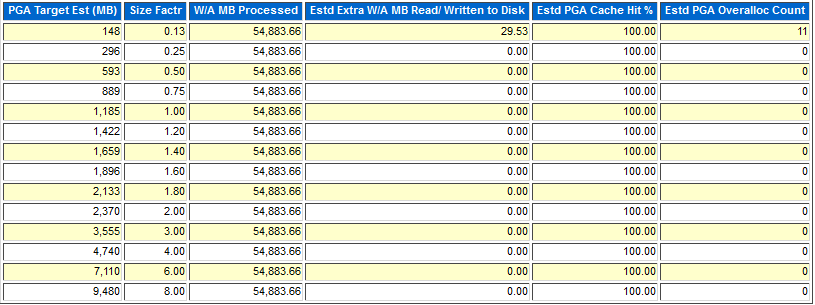


         第一部分是关于BufferPool Advisory的大小建议，为了能看懂这个建议清单，我们来先了解各个列的含义

|  |  |
| --- | --- |
| Size for Est(M) | Oracle估算Buffer pool的大小 |
| Size Factor | 估算值和实际值的一个比例，比如0.9就是估算值是实际大小的90%，1.0表示buffer pool的实际大小 |
| Buffers for Estimate | 估算的buffer的大小（数量） |
| Est Phys Read Factor | 估算的物理读的影响因子，是估算物理读和实际物理读的一个比例，1.0表示实际的物理读 |
| Estimated Physical Reads | 估算的物理读次数 |

         我们看到bufferpool的实际大小为1136M（size=1.0）此时物理读是49613次，当oracle尝试增大1.06倍时，物理读减少为48250，如果我们继续增加到1.41倍，尽管我们buffer pool增加一半，物理读减少还是有限的，这个就要一个度控制文件和成本。

**PGA Aggr Summary**



这一部是建议关于PGA内存非大小的建议，他们的每个列大小含义如下：

|  |  |
| --- | --- |
| PGA Target Est (MB) | PGA的估算大小 |
| Size Factr | 影响因子，作用和buffer pool相同 |
| W/A MB Processed | Oracle为了产生估算处理的数据量 |
| Estd Extra W/A MB | 处理数据中需要物理读写的数据量 |
| Estd PGA Cache Hit % | 估算的PGA命中率 |
| Estd PGA Overalloc Count | 需要在估算的PGA大小额外分配内存的次数 |

这个和Buffer Pool Advisory类似，这个我们不需要调整pga，如果pga调整为原来的0.13倍的时候，系统的性能会有很大的影响，即修改参数PGA\_AGGREAGTE\_TARGET的大小设置。

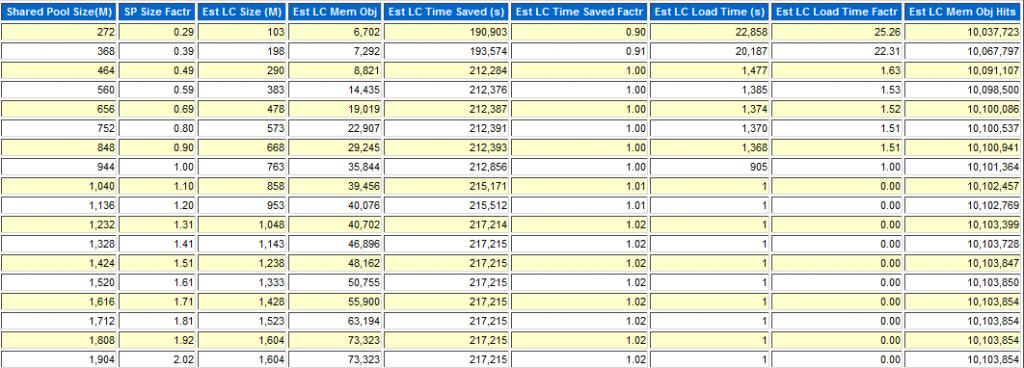
**Shared Pool Advisory**

这一部分是建议器对共享次大小的建议清单，各个列的含义如下

|  |  |
| --- | --- |
| Shared Pool Size(M) | 估算共享池的大小 |
| SP Size Factr | 估算共享池的影响因子 |
| Est LC Size (M) | 估算的库高速缓存占用的大小（LC，library cache） |
| Est LC Mem Obj | 高速缓冲区命中的对象数 |
| Est LC Time Saved (s) | 需要额外将对象读入共享池的时间 |
| Est LC Time Saved Factr | 影响因子 |
| Est LC Load Time (s) | 分析所花费的时间 |
| Est LC Load Time Factr | 分析花费时间事件的影响因子 |
| Est LC Mem Obj Hits | 内存中对象被发现的次数 |

这里我们主要Est LC Time Saved Factr，它表示每模拟一次sharedpool大小将对对象读入共享池的影响情况，当这个值变化很小或者不变的时候，增加shared pool就没有多大意义比如Est LC Time Saved Factr     =1，随着内存的增大，它的值变化显得非常小，所以我们认为当前共享池大小事合适的。

**SGA Target Advisory**



|  |  |
| --- | --- |
| SGA Target Size (M) | 估算SGA大小 |
| SGA Size Factor | SGA大小的影响因子 |
| Est DB Time (s) | 估算的SGA大小计算出的DB Time |
| Est Physical Reads | 物理读次数 |

         这部分比较简单，我们看到当前的SGA大小基本上是合适的，当影响因子从0.75到1时，物理读减少了很多，下面的影响程度就没有那么大了。

以上部分就是oracle建议器给出内存各个组件大小的建议，这些信息非常直观，很容易理解。

接下来的几个部分是关于latch信息。

Latch Activity

Latch Sleep Breakdown

Latch Miss Sources

Parent Latch Statistics

Child Latch Statistics

这里就不具体讨论了。

**Segment Statistics**

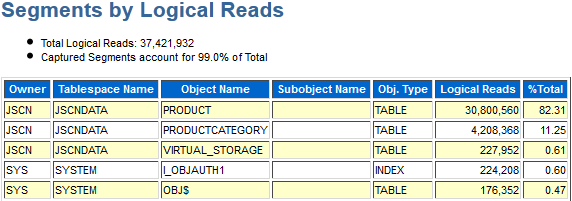
Segments by Logical Reads

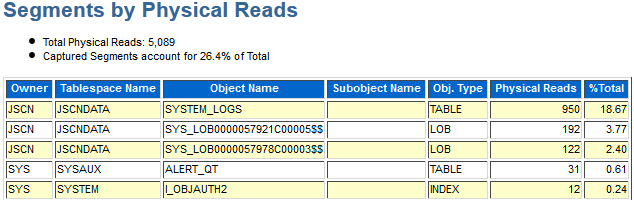
Segments by Physical Reads

Segments by Row Lock Waits

Segments by ITL Waits

Segments by Buffer Busy Waits





这两个部分是从对象角度来展现io的情况，之前的部分我们是从物理层面（表空间，数据文件）来分析I/O性能的，分析者两部分信息，可以具体地知道哪些对象访问导致了I/O性能的下降，这些信息对我们最终定位和解决问题是有一定帮助的。

让我们来分析了一下生产数据库awr报告，简单地回顾一下我们的思路：

首先，我们在报告load profile部分发现了数据的I/O比较大：

让我们怀疑系统是否有I/O瓶颈。

在top5的等待时间里，我们看到数据库的I/O的确比较大，但是更应该关注一个并发的等待事件latch：row cache objects，它很有可能

影响到数据库的性能，在sql部分，我们看到1条执行事件很长的sql，cpu消耗比较少，更让我们相信sql都处于一种I/O等待状态：

在表空间I/O的部分，我们看到一些表空间的I/O延时很大：

综合这些信息，我们大概可以得出下面的结论：

数据库的I/O等待有些长，这是由一些大的查询导致的，对于I/O是否真的出现瓶颈，我们需要在操作系统级使用一些工具进行进一步判断，

并不能认为数据库I/O有瓶颈，因为对于olap数据库，这样的I/O是很正常的。

共享池的等待对sql操作有比较大的影响，要考虑做优化。

cpu资源消耗不太大。

整体看来，数据库中资源只用情况比较正常，没有出现大的阻塞和等待问题。

小结

awr报告的信息量非常大，基本上涉及了所有的性能指标，在看一个数据库的awr报告时候，一定要由清晰的思路，如果你了解数据库的业务

情况，那么就应该针对性地看一些可能存在性能问题的部分，并结合业务的实际情况来判断性能问题是否严重，是否有优化的余地，

比如说内存大小的建议部分，我们要结合实际机器的业务情况，是否客观的对他的取值做出评价。

如果对数据库运行业务不太了解，那么可以从top5的等待事件出发，按照等待事件的类型，到相应的部分获取详细信息来对系统性能问题

做出判断。

另外，我们需要可逛地对待这些新能指标，不要觉得有些地方稍微有些性能问题，就把问题扩大化，只要客户能容忍系统的性能，没有

理由主动地优化它，毕竟系统的稳定性才是第一位的。

整理之网络