enq: HW - contention or

SQL .

为防止多个进程同时修改HWM而提供的锁称为HW锁。想要移动HWM的进程必须获得HW锁。若在获取HW锁过程中发生争用，则等待enq: HW - contention事件。HW锁争用大部分是大量执行insert所引发的。

众所周知，[Oracle](http://www.linuxidc.com/topicnews.aspx?tid=12)高水位线标志着该线以下的block均被Oracle格式过，通俗一点讲就是该高水位线以下的block都被Oracle使用过。 通常在执行insert操作时，当高水位线以下block不够用时，Oracle将会推进高水位线。更进一步讲，当有多个进程在同时进行insert操作时，比较容易引起高水位线争用，主要表现为enq: HW - contention。

-- 查看v$session\_wait，应该会有如下等待事件：

SQL>select event,p1,p2,p3 from v$session\_wait;

EVENT                     P1         P2      P3

---------------------- --------   ------- ----------

enq: HW - contention 1213661190  4 17005691

eg：ash报告

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Event | % Event | P1 Value, P2 Value, P3 Value | % Activity | Parameter 1 | Parameter 2 | Parameter 3 |
| enq: HW - contention | 79.62 | "1213661190","34","209953291" | 79.62 | name|mode | table space # | block |

1》 查找争用的表空间

select \* from dba\_tablespaces where TABLESPACE\_NAME = (select name from v$tablespace where ts#=4);

2》通过P3进行DBMS\_UTILITY转换可以获知发生争用的文件和block

SQL> select DBMS\_UTILITY.DATA\_BLOCK\_ADDRESS\_FILE(17005691) FILE#,

  2  DBMS\_UTILITY.DATA\_BLOCK\_ADDRESS\_BLOCK(17005691) BLOCK#

  3  from dual;

     FILE#     BLOCK#

---------- ----------

         4     228475

 进而通过file#和block#定位对象

SQL> select owner, segment\_type, segment\_name

  2  from dba\_extents

  3  where file\_id = 4

  4  and 228475 between block\_id and block\_id + blocks - 1;

OWNER SEGMENT\_TYPE SEGMENT\_NAME

--------------- --------------- ------------------------------

SCOTT LOBSEGMENT EMP\_DATA\_LOB

3 》将enq: HW - contention等待时间对应的sql抓取出来，或者通过v$active\_session\_history，根据等待enq: HW - contention找到对应的SQL\_ID，再通过v$sqltext查询对应的SQL语句，

SQL> select user\_id,sql\_id,event from v$active\_session\_history where event='enq: HW - contention';

SQL> select sql\_id,command\_type,sql\_text from v$sqltext where sql\_id='6tc1vpf9wz3xs';

我们知道enqueue锁的p2,p3值与v$lock的id1，id2值相同，同样通过id2，也可以知道发生争用的文件和block

SQL> select DBMS\_UTILITY.DATA\_BLOCK\_ADDRESS\_FILE(ID2) FILE#,

  2  DBMS\_UTILITY.DATA\_BLOCK\_ADDRESS\_BLOCK(ID2) BLOCK#

  3  from v$lock

  4  where type = 'HW';

      FILE#     BLOCK#

---------- ----------

         4     228475

通过p1值可以知道锁类型和模式

SQL> select chr(bitand(1213661190,-16777216)/16777215)||

  2  chr(bitand(1213661190,16711680)/65535) "Lock",to\_char( bitand(1213661190, 65535) ) "Mode" from dual;

Lock       Mode

---------- ----------

HW         6

|  |
| --- |
| **How To Analyze the Wait Statistic: 'enq: HW - contention' (文档 ID 419348.1)** |

========================================================================

当知道了，lob对象发生高水位争用时，该怎么办呢？metalink（740075.1）为我们提供了几种解决办法，仅供参考

引用

1. When using Automatic Segment Space Management (ASSM)

   a) As temporary workaround, manually add extra space to the LOB segment

      ALTER TABLE <lob\_table>

      MODIFY LOB (<column\_name>) (allocate extent (size <extent size>));

OR

   b) It may related Bug 6376915.

   Refer to Note 6376915.8 "Bug 6376915 HW enqueue contention for ASSM LOB segments"

   In 10.2.0.4 or above, this fix has been included, and can be enabled by setting event 44951 to a value

   between 1 and 1024.  A higher value would be more beneficial in reducing contention.

   EVENT="44951 TRACE NAME CONTEXT FOREVER, LEVEL < 1 - 1024 >"

OR

  c) Consider partitioning the LOB  in a manner that will evenly distribute concurrent DML across multiple partitions

2. When using Manual Segment Space Management(MSSM)

a) As temporary workaround, manually add extra space to the LOB segment

    ALTER TABLE <lob\_table>

    MODIFY LOB (<column\_name>) (allocate extent (size <extent size>));

OR

     b) Consider partitioning the LOB in a manner that will evenly distribute concurrent DML across multiple partitions