故障排除："WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK! " (Doc ID 2016422.1)

文档内容

适用于:

Oracle Database Cloud Schema Service - 版本 N/A 和更高版本
Gen 1 Exadata Cloud at Customer (Oracle Exadata Database Cloud Machine) - 版本 N/A 和更高版本
Oracle Cloud Infrastructure - Database Service - 版本 N/A 和更高版本
Oracle Database Cloud Exadata Service - 版本 N/A 和更高版本
Oracle Database Exadata Express Cloud Service - 版本 N/A 和更高版本
本文档所含信息适用于所有平台

用途

本文档的目的是帮助排查原因"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK! "

排错步骤

什么是 row cache enqueue 锁（Row Cache Enqueue Lock）？

行缓存（Row Cache）或数据字典缓存（Data Dictionary Cache）是保存数据字典信息的共享池的内存区域。row cache 保存数据时并不是以数据块的形式，而是以行的形式。row cache enqueue 锁是在数据字典行的锁。此 enqueue 是关于特定数据字典对象的。这就是所谓的 enqueue 类型，可以在视图 V$rowcache 中找到。

对于每个版本 row cache 类型的列表，请参阅：

> Document 468334.1 How to Match a Row Cache Object Child Latch to its Row Cache

"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!" 警告信息是什么意思?

当我们试图获得 row cache 锁，这种等待事件将被使用。
当 row cache 冲突发生时，如果不能在一个预定的时间周期内得到 enqueue，将在 USER_DUMP_DEST 或 background_dump_dest 目录下生成一个跟踪文件，这取决于是用户还是后台进程创建的跟踪文件。alert.log 通常会相应的更新警告消息和跟踪文件的位置。

数据库检测到核心资源被持有太久并通知管理员，从而让这种情况可以得到解决。这也可能伴随着数据库挂起或变慢。
alert.log 的消息和生成的跟踪文件趋向于包含消息：

```
> WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK! <<<
```

如果不能立即获取 rowcache entry 锁，那么进入一个循环，先释放 row cache 对象闩锁，等待（等待上述等待事件），重新获得闩锁，然后再次尝试获取 rowcache 锁。在单实例模式，会重复 1000次直到进程报错"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK"。在 RAC 环境会一直重复，直到不能获得实例锁或者被中断。

Systemstate dump 可以提供一些有用的信息诊断争用的原因。

> 请注意：The "WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!" 当达到阈值会引发这个消息，因此，如果未达到阈值它不会被引发。这意味着，不太严重的问题，即使具有相同的原因，也可以不输出该消息。

"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!" 可能的原因

SGA收缩（shrink）/调整大小的操作（resize）

如果 SGA 动态地改变大小，需要持有各种 latches 来避免其它进程同时操作，直到操作完成。如果调整大小需要一段时间，或者是经常发生，你会看到"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!"的发生。定位这种情况的方法是，有很多'SGA: allocation forcing component growth'等待事件，或 AWR 的 TOP 列表有类似等待,以及阻塞等待"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!" 的会话在等待'SGA: allocation forcing component growth'（或类似）。

常见的解决方案是增大SGA或者关闭 AMM。

有一些可用的代码修复，请参阅：

> Document 7189722.8 Bug 7189722 - Frequent grow/shrink SGA resize operations
> Document 9267837.8 Bug 9267837 - Auto-SGA policy may see larger resizes than needed

row cache enqueue 类型

对于每一个 enqueue 类型，都有对应的一些操作会需要获取这类 enqueue。队列的类型,可能给出由于操作可能导致的问题的指示。一些常见的原因如下：

DC_TABLESPACES

最可能的原因是新 extent 的分配。如果 extent 大小设置过小，那么应用程序可能会不断地要求新的 extent，这可能导致争用。你有很小的 extent 尺寸，正在迅速增长的对象吗？（通过查找具有大量 extents 的对象可以定位它们）。检查 insert/update 活动的 trace，查找那些就有很多 extents 的对象。

DC_SEQUENCES

检查应用程序用到的 sequence 的 cache 的大小：

> Document 853652.1 RAC and Sequences
> Document 395314.1 RAC Hangs due to small cache size on SYS.AUDSES$ - fixed in 10.2.0.3
> Document 6027068.8 Bug 6027068 - Contention on ORA_TQ_BASE sequence -fixed in 10.2.0.5 and 11.2.0.1

DC_USERS

一个会话正在对一个用户执行 GRANT，与此同时此用户正在登录到数据库中，此时可能会发生死锁或导致"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!" 。

> Document 4604972.8 Bug 4604972 - Deadlock on dc_users by Concurrent Grant/Revoke - fixed in 11.1.0.6
> Document 6143420.8 Bug 6143420 - Deadlock involving "ROW CACHE LOCK" on dc_users AND "CURSOR: PIN S WAIT ON X"- fixed in 10.2.0.5 and 11.1.0.6 DC_OBJECTS
> Document 12772404.8 Bug 12772404 - Significant "row cache objects" latch contention when using VPD

DC_OBJECT_IDS

> Document 11693365.8 Bug 11693365 - Concurrent Drop table and Select on Reference constraint table hangs(deadlock) - fixed in 12.1

DC_SEGMENTS

这很可能是 segment 的分配导致的。确定持有锁的用户正在做什么并使用 errorstacks 进行诊断。

DC_ROLLBACK_SEGMENTS

这可能是由于 rollback 段的分配导致的。正如 dc_segments，确定谁持有锁并收集 errorstack 来进行诊断。请记住，在多节点系统（RAC）上，持有者可能在另一节点上，因此需要所有节点的 systemstate。

DC_TABLE_SCNS

Document 5756769.8 Bug 5756769 - Deadlock between Create MVIEW and DML - fixed in 10.2.0.5 ,11.1.07 and 11.2.0.1

DC_AWR_CONTROL

此 enqueue 关系到 AWR（Automatic Workload Repository）的控制权。任何操纵 AWR 资料库的操作将持有它. 要分析这个问题,需要查找是那些进程阻塞了它们。

RAC 相关的 Bugs

Document 6004916.8 Bug 6004916 - Hang involving row cache enqueues in RAC (ORA-4021) - fixed in 102.0.5 and 11.1.0.6
Document 8666117.8 Bug 8666117 - High row cache latch contention in RAC - fixed in 11.2.0.2 and 12.1
Document 9866045.8 Bug 9866045 - Long wait on 'wait for master scn' in LCK causing long row cache lock waits - fixed in 12.1

我可以收集哪些信息，以确定原因?

Systemstate dump

当问题发生时，错误会记入 alert.log，并自动产生一个 systemstate dump 文件。

```
Wed Sep 21 13:39:19 2011
> WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK! pid=37
System State dumped to trace file /oracle/diag/rdbms/..../.trc
```

AWR，ADDM 和 ASH 报告

收集两份 AWR 报告，一份有问题时间段的,另一个是没有问题时间段的，因为这些可以帮助我们理解问题发生时数据库的状况 AWR，ADDM，ASH 报告，可以相互取长补短，从而更完整地理解整个问题。

取决于 AWR 快照生成的时间间隔，收集最小时间间隔的报告。默认的快照是一个小时的时间间隔。

```
SQL>@$ORACLE_HOME/rdbms/admin/awrrpt.sql
SQL>@$ORACLE_HOME/rdbms/admin/addmrpt.sql
SQL>@$ORACLE_HOME/rdbms/admin/ashrpt.sql
```

鉴于分析 systemstate 是一件很复杂的事情，您可以创建一个服务请求，并上传 alert.log，systemstate dump，以及问题发生前和问题发生时的 AWR 报告请 Oracle 技术支持来分析。

如何分析收集到的诊断信息?

Systemstate dump

通常情况下， row cache enqueue 是一系列事件的一部分，阻塞了申请 row cache enqueue 的进程的进程很可能被另一个进程阻塞。Row cache enqueue 经常是问题的表象。

Systemstate dump 可以帮助查找申请的是哪一个 row cache ，并可能有助于发现阻塞进程。

示例1：

```
Oracle process number: 77
Unix process pid: 10846, image: oracle@xxxxxxx
*** 2011-05-13 08:08:58.775
*** SERVICE NAME:(xxxxxx_SERVICE) 2011-05-13 08:08:58.775
*** SESSION ID:(1076.796) 2011-05-13 08:08:58.775
> WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK! <<<
row cache enqueue: session: 0x1df57ade8, mode: N, request: S
```

trace 的标题显示下列内容：

- 等待 row cache enqueue 锁的 Oracle 进程号（PID）（在这个案例，进程 77）。
- 正在申请的 row cache enqueue 的模式（请求：S）。

因此，在上述例子中，进程 77 是在请求以共享模式获得 row cache（请求：S）。

Systemstate 包含数据库中每一个进程的状态信息，因此可以在 systemstate 中查找这个进程：

```
PROCESS 77
----------------------------------------
.
.
----------------------------------------
SO: 0x1cdf11958, type: 50, owner: 0x17d198288, flag: INIT/-/-/0x00
row cache enqueue: count=1 session=0x1df57ade8 object=0x1dc9a5d30, request=S
savepoint=0x87b70d
row cache parent object: address=0x1dc9a5d30 cid=7(dc_users)
.
.
```

从上面我们看到，进程 77 请求共享模式获得 row cache dc_users。
进程 77 处于等待状态，意味着被其它进程阻塞，我们现在需要检查 systemstate 判断谁持有资源并阻塞了这个进程。

搜索引用的对象，在这个例子中，是 object=0x1dc9a5d30。
这样做完以后，我们发现，进程 218 正以独占模式持有这个对象：

```
PROCESS 218:
----------------------------------------
.
.
SO: 0x1cdf118f8, type: 50, owner: 0x1ceb0f178, flag: INIT/-/-/0x00
row cache enqueue: count=1 session=0x1da54cf68 object=0x1dc9a5d30, request=X
savepoint=0x11e
row cache parent object: address=0x1dc9a5d30 cid=7(dc_users)
```

独占模式的请求，将会一直阻塞共享模式的请求，直到该进程独占模式的请求被满足并稍后释放了这个资源。因此，这将阻止其他共享模式请求。请注意，这是请求独占而不是独占持有，所以这个请求也一定被阻塞了。查看其他进程中，我们看到进程 164 在以共享模式（mode=s）持有这个对象。

```
PROCESS 164:
----------------------------------------
.
.
O/S info: user: u1m, term: , ospid: 1234, machine: cpc44711
program:
last wait for 'SQL*Net message from client' blocking sess=0x(nil) seq=36289 wait_time=6943 seconds since wait
started=2539
driver id=54435000, #bytes=1, =0
.
.
SO: 0x1cdf11418, type: 50, owner: 0x1ccc26120, flag: INIT/-/-/0x00
row cache enqueue: count=2 session=0x1df578318 object=0x1dc9a5d30, mode=S
savepoint=0xb1bd8e
row cache parent object: address=0x1dc9a5d30 cid=7(dc_users)
hash=fc968070 typ=11 transaction=(nil) flags=00000002
own=0x1dc9a5e00[0x1cdf11448,0x1cdf11448] wat=0x1dc9a5e10[0x1cdf11928,0x17d5192e0] mode=S
```

因此，进程 164 以共享模式持有 row cache enqueue（mode= S），从而防止了进程 218 以独占模式获得该 row cache enqueue。此外，我们看到，进程 164 在 ON CPU（systemstate 显示最后一个等待是'SQL*Net message from client'，而不是等待'SQL*Net message from client'）。为了进一步诊断，技术支持需要检查堆栈调用，以确定为什么这个进程在 ON CPU 并持有该队列这么久（从开始已经等待了2539秒）。

示例2：

在这个例子中，进程 18（MMON）等待以共享模式获得类型为 dc_awr_control 的 row cache。

```
Oracle Database 10g Enterprise Edition Release 10.2.0.5.0 - 64bit Production
With the Partitioning, Oracle Label Security, OLAP, Data Mining
and Real Application Testing options
ORACLE_HOME = /opt/oracle10/product/10.2.0
System name: SunOS
Node name: xxxxxx
Release: 5.10
Version: Generic_144488-04
Machine: xxxxx
Instance name: xxxx
Redo thread mounted by this instance: 1
Oracle process number: 18
Unix process pid: 6196, image: oracle@xxxxx (MMON)
.
.

PROCESS 18:
```

```
----------------------------------------
.
.
last wait for 'ksdxexeotherwait' wait_time=0.000013 sec, seconds since wait started=6
.
.
SO: 39bf1f0e8, type: 50, owner: 3980783a0, flag: INIT/-/-/0x00
row cache enqueue: count=1 session=3be37ea80 object=39a79f090, request=S
savepoint=0x41f0ae
row cache parent object: address=39a79f090 cid=22(dc_awr_control)
hash=6f60197e typ=9 transaction=3bc39f560 flags=0000002a
own=39a79f160[39bf1f178,39bf1f178] wat=39a79f170[39bf1f118,39bf1f118] mode=X
.
.
```

对象（object=39a79f090）的 row cache lock 被进程 269 以独占模式（mode=x）持有。进程在等待'SGA: allocation forcing component growth'。

```
PROCESS 269:
----------------------------------------
.
.

waiting for 'SGA: allocation forcing component growth' wait_time=0, seconds since wait started=3

.
.
SO: 39bf1f148, type: 50, owner: 3bc39f560, flag: INIT/-/-/0x00
row cache enqueue: count=1 session=3be1b7c98 object=39a79f090, mode=X
savepoint=0x41efe8
row cache parent object: address=39a79f090 cid=22(dc_awr_control)
hash=6f60197e typ=9 transaction=3bc39f560 flags=0000002a
own=39a79f160[39bf1f178,39bf1f178] wat=39a79f170[39bf1f118,39bf1f118] mode=X
.
.
```

因此根本原因就是 SGA 的大小调整，等待 row cache 是次要结果。
我们使用该期间的 AWR 报告看相关信息：

AWR 报告

```
Top 5 Timed Events                                         Avg %Total
~~~~~~~~~~~~~~~~~~                                        wait   Call
Event                                 Waits    Time (s)   (ms)   Time Wait Class
----------------------------- ------------ ----------- ------ ------ ----------
SGA: allocation forcing compon   42,067,317      38,469      1    7.6      Other
CPU time                                          2,796           0.6
db file sequential read             132,906         929      7    0.2   User I/O
latch free                        4,282,858         704      0    0.1      Other
log file switch (checkpoint in          904         560    620    0.1 Configurat
          -------------------------------------------------------------
```

我们可以清楚地看到，在 Top 5 等待事件中，整个系统中有针对此事件的一个显著等待；并且'SGA: allocation forcing component growth' 是这一时间点的一个主要问题。"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!" 消息的根本原因就是内存调整活动，TOP 5 的等待事件甚至不显示等待"row cache"症状。

注意：
如果调整内存大小没有那么严重，有些时候没有"WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!"消息。 - 主要是因为没有达到在此之前提到的阈值。但是，您可能会看到其他等待事件。在以下文献中概述了一个常见的例子：

Document 742599.1 High 'cursor: pin S wait on X' and/or 'library cache lock' Waits Generated by Frequent Shared Pool/Buffer Cache Resize Activity

对于频繁的内存调整，有几个潜在的可用修复，请参阅：

Document 7189722.8 Bug 7189722 - Frequent grow/shrink SGA resize operations
Document 9267837.8 Bug 9267837 - Auto-SGA policy may see larger resizes than needed

10g 以前的版本可能存在的问题

10g 之前的版本，检测 row cache 级别死锁的方法有限。为了尽量减少发生死锁的可能性，可能的解决方法：

- 设置 TIMED_STATISTICS=FALSE
- 设置 _row_cache_cursors=20 以上（默认值 10）
- 不要做任何 tracing

Document 30802.1 Init.ora Parameter "ROW_CACHE_CURSORS" Reference Note

其他问题疑难解答

对于其他性能问题的故障排除,请参阅:

> Document 1377446.1 Troubleshooting Performance Issues

Bug 12772404 - Significant "row cache objects" latch contention when using VPD (Doc ID 12772404.8)

参考

BUG:8666117 - LCK0 PROCESS STUCK AT WAITING FOR "LATCH: ROW CACHE OBJECTS"
NOTE:1377446.1 - * Troubleshooting Performance Issues
NOTE:853652.1 - RAC and Sequences
NOTE:11693365.8 - Bug 11693365 - Concurrent Drop table and Select on Reference constraint table hangs (deadlock)
NOTE:8666117.8 - Bug 8666117 - High row cache latch contention in RAC

NOTE:9866045.8 - Bug 9866045 - Long wait on 'wait for master scn' in LCK causing long row cache lock waits
NOTE:6027068.8 - Bug 6027068 - Contention on ORA_TQ_BASE$ sequence
NOTE:6143420.8 - Bug 6143420 - Deadlock involving "ROW CACHE LOCK" on dc_users AND "CURSOR: PIN S WAIT ON X"
NOTE:742599.1 - High 'Cursor: Pin S Wait On X', 'Library Cache Lock' And "Latch: Shared Pool" Waits due to Shared Pool/Buffer Cache Resize Activity
NOTE:4604972.8 - Bug 4604972 - Deadlock on dc_users by Concurrent Grant/Revoke
NOTE:468334.1 - How to Match a Row Cache Object Child Latch to its Row Cache
NOTE:395314.1 - RAC Hangs due to small cache size on SYS.AUDSES$
NOTE:5756769.8 - Bug 5756769 - Deadlock between Create MVIEW and DML
NOTE:6004916.8 - Bug 6004916 - Hang involving row cache enqueues in RAC (ORA-4021)
NOTE:30802.1 - Init.ora Parameter "ROW_CACHE_CURSORS" Reference Note

BUG:5756769 - ROW CACHE DEADLOCK "WAITED TOO LONG FOR A ROW CACHE ENQUEUE LOCK!"