墨墨导读：在Oracle 11g 版本中可能出现由于JDBC bug导致sql绑定变量无法共享，过期游标过多的情况，此时如果发生大量并发业务，很有可能造成异常library cache lock等待事件，造成数据库突发性能问题。在此，我们分享一次由jdbc bug和绑定变量长度问题共同“作案”引发数据库性能故障的案例。

本文节选自《云和恩墨技术通讯》（12月刊）

下载链接：https://www.modb.pro/doc/1593

library cache lock等待事件是Oracle数据库较为常见的等待事件之一，在之前的几次月刊中，我们也提到过产生library cache lock等待出现的原因有很多，如登录密码错误尝试过多、热表收集统计信息和SQL解析失败等。

在Oracle 11g 版本中可能出现由于JDBC bug导致sql绑定变量无法共享，过期游标过多的情况，此时如果发生大量并发业务，很有可能造成异常library cache lock等待事件，造成数据库突发性能问题。在此，我们分享一次由jdbc bug和绑定变量长度问题共同“作案”引发数据库性能故障的案例，供各位参考。

问题描述

2019年10月11号晚22:00分左右，运维人员对生产系统数据库进行清理历史分区操作，执行近100个分区删除操作后（22:05左右）发现该数据库压力飙升，维护人员紧急停止历史分区清理操作，发现大量业务数据插入（INSERT）缓慢。

查看故障期间数据库发现大量library cache lock等待，数据库活动会话飙升至1000以上，数据库响应非常缓慢，业务受到严重影响。

问题分析

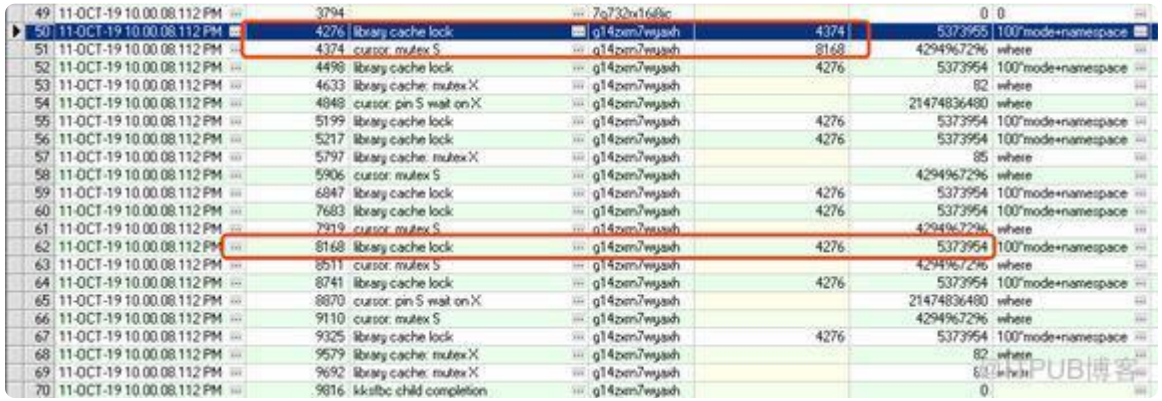
从故障期间ASH的整体运行情况看：

7	2019-10-11 21:59:37	3
8	2019-10-11 22:00:08	39
9	2019-10-11 22:00:18	3
10	2019-10-11 22:00:28	80
11	2019-10-11 22:00:38	18
12	2019-10-11 22:00:48	267
13	2019-10-11 22:00:58	345
14	2019-10-11 22:01:09	436
15	2019-10-11 22:01:19	698
16	2019-10-11 22:01:29	903
17	2019-10-11 22:01:40	1193
18	2019-10-11 22:01:51	1349
19	2019-10-11 22:02:02	1351
20	2019-10-11 22:02:13	1351

从22:00开始，数据库的活动会话飙升，每秒活动会话飙升至1000以上。故障时间段内的TOP EVENT主要表现在library cache lock和library cache: mutex X等待上。

查看故障期间数据库活动会话情况：

	EVENT	COUNT(*)
1	library cache lock	238555
2	library cache: mutex X	66539
3	cursor: mutex S	9796
4		1019
5	Backup: MML write backup piece	266
6	db file sequential read	171
7	direct path read	133
8	cursor: pin S wait on X	60
9	gc cr disk read	31
10	Backup: MML create a backup piece	28
11	log file sync	18
12	cursor: mutex X	15



49	11-OCT-19 10.00.08.112 PM	3794		7g732nx1648c		0	0	
50	11-OCT-19 10.00.08.112 PM	4276	library cache lock	g14zxrn7wyaxh	4374	5373955	100*mode+namespace	
51	11-OCT-19 10.00.08.112 PM	4374	cursor: mutex S	g14zxrn7wyaxh	8168	4294967296	where	
52	11-OCT-19 10.00.08.112 PM	4498	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
53	11-OCT-19 10.00.08.112 PM	4633	library cache: mutex X	g14zxrn7wyaxh		82	where	
54	11-OCT-19 10.00.08.112 PM	4848	cursor: pin S wait on X	g14zxrn7wyaxh		2147483648	where	
55	11-OCT-19 10.00.08.112 PM	5199	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
56	11-OCT-19 10.00.08.112 PM	5217	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
57	11-OCT-19 10.00.08.112 PM	5797	library cache: mutex X	g14zxrn7wyaxh		85	where	
58	11-OCT-19 10.00.08.112 PM	5906	cursor: mutex S	g14zxrn7wyaxh		4294967296	where	
59	11-OCT-19 10.00.08.112 PM	6847	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
60	11-OCT-19 10.00.08.112 PM	7683	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
61	11-OCT-19 10.00.08.112 PM	7919	cursor: mutex S	g14zxrn7wyaxh		4294967296	where	
62	11-OCT-19 10.00.08.112 PM	8168	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
63	11-OCT-19 10.00.08.112 PM	8511	cursor: mutex S	g14zxrn7wyaxh		4294967296	where	
64	11-OCT-19 10.00.08.112 PM	8741	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
65	11-OCT-19 10.00.08.112 PM	8870	cursor: pin S wait on X	g14zxrn7wyaxh		2147483648	where	
66	11-OCT-19 10.00.08.112 PM	9110	cursor: mutex S	g14zxrn7wyaxh		4294967296	where	
67	11-OCT-19 10.00.08.112 PM	9325	library cache lock	g14zxrn7wyaxh	4276	5373954	100*mode+namespace	
68	11-OCT-19 10.00.08.112 PM	9579	library cache: mutex X	g14zxrn7wyaxh		82	where	
69	11-OCT-19 10.00.08.112 PM	9692	library cache: mutex X	g14zxrn7wyaxh		8	where	
70	11-OCT-19 10.00.08.112 PM	9816	kksfbc child completion	g14zxrn7wyaxh		0		

从10:00:08的ash信息来看，多个library cache lock被4276会话阻塞，4276会话被4374会话 “cursor：mutex S”阻塞，同时4374会话被8168“library cache lock”阻塞。从ash分析来看，大量的library cache lock会话的p3值都是5373954和5373955。5373954指的是mode=2，5373955的mode=3，只是持有的方式不同mode=3就是exclusive独占锁。

而4276会话library cache lock的p3值是5373955，对应的namespace HEX:52 —>DEC:82，mode=3。

```
SQL> SELECT indx,kglstdsc FROM x$kglst where indx=82;

      INDX KGLSTDSC
---------- ----------------------------------------------------------------
        82 SQL AREA BUILD                                          @ITPUB博客
```

SQL AREA BUILD说明library cache lock是在SQL解析上或SQL AREA上的问题。

发生等待是会话都是在执行g14zxrn7wyaxh INSERT SQL语句：

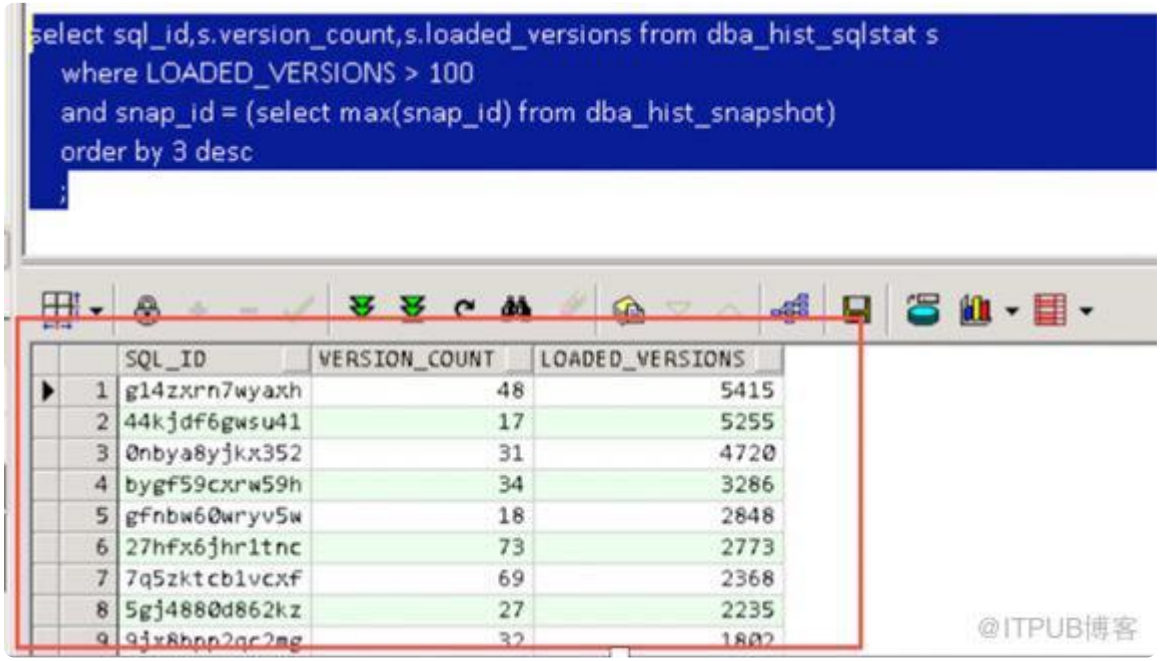
```
/** PayOrderMapper.insert */
INSERT INTO TxxxxxxT T
  (T.ID,……T.SxxxO)
VALUES
  (SEQ_xxx.nextval,

:1,:2,:3,:4,:5,:6,:7,:8,:9,:10,:11,:12,:13,:14,:15,:16,:17,:18,:19,:20,:21,:22,:23,:24,:25,:26,:27,:28,:29,:30,:31,:32
,:33,:34,:35,:36,:37,:38,:39,:40,:41,:42,:43,:44,:45,:46,SYSDATE,:47,SYSDATE,:48,:49,:50,:51
```

该SQL中有51个绑定变量，多个绑定变量可能会导致bind variable graduation问题出现，继而导致cursor无法被shared。

INST	BEGIN_SNAP_ID	BEGIN_SNAP_TIME	EXECS	SQL_ID	PLAN	LOADED_VERSIONS	VERSION_COUNT	LOAD	INVALIDATIONS
1	86796	2019-10-11 19:15	140224	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86797	2019-10-11 19:30	138630	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86798	2019-10-11 19:45	128486	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86799	2019-10-11 20:00	159819	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86800	2019-10-11 20:15	100858	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86801	2019-10-11 20:30	112271	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86802	2019-10-11 20:45	117851	g14zxrn7wyaxh	4052461954	2431	35	2	0
1	86803	2019-10-11 21:00	116778	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86804	2019-10-11 21:15	107542	g14zxrn7wyaxh	4052461954	2431	35	2	0
1	86805	2019-10-11 21:30	99224	g14zxrn7wyaxh	4052461954	2431	35	3	0
1	86806	2019-10-11 21:45	518	g14zxrn7wyaxh	4052461954	2515	20	200	21
1	86807	2019-10-11 22:00	4329	g14zxrn7wyaxh	4052461954	3365	89	685	99
1	86808	2019-10-11 22:15	6929	g14zxrn7wyaxh	4052461954	4416	47	477	0
1	86809	2019-10-11 22:30	8264	g14zxrn7wyaxh	4052461954	5373	5	483	0
1	86810	2019-10-11 22:45	39354	g14zxrn7wyaxh	4052461954	5406	38	31	0
1	86811	2019-10-11 23:00	50470	g14zxrn7wyaxh	4052461954	5409	41	5	0
1	86812	2019-10-11 23:15	44148	g14zxrn7wyaxh	4052461954	5410	42	6	0
1	86813	2019-10-11 23:30	20582	g14zxrn7wyaxh	4052461954	5410	42	3	0
1	86814	2019-10-11 23:45	17041	g14zxrn7wyaxh	4052461954	5410	42	3	0
1	86815	2019-10-12 00:00	54091	g14zxrn7wyaxh	4052461954	5410	42	2	0
1	86816	2019-10-12 00:15	37586	g14zxrn7wyaxh	4052461954	5410	42	2	0
1	86817	2019-10-12 00:30	10009	g14zxrn7wyaxh	4052461954	5410	42	2	0
1	86818	2019-10-12 00:45	10196	g14zxrn7wyaxh	4052461954	5411	44	4	0
1	86819	2019-10-12 01:00	40188	g14zxrn7wyaxh	4052461954	5411	44	4	0
1	86820	2019-10-12 01:15	29665	g14zxrn7wyaxh	4052461954	5411	44	2	0
1	86821	2019-10-12 01:30	32731	g14zxrn7wyaxh	4052461954	5411	44	2	0
1	86822	2019-10-12 01:45	27745	g14zxrn7wyaxh	4052461954	5411	44	1	0

从ASH和DBA_HIST_SQLSTAT中可以看出21:45分之后SQL频繁load到cursor cache中，其中invalidations有120次，这是从DASH中取的数据，实际数值比采样还要大，另外SQL的LOADED_VERSION从原来的2431个在短时间内增长到5411个，实际的version count由于11.2.0.3的隐含参数_cursor_obsolete_threshold的关系，version count超过100会重新开始。

这个时候就怀疑是由于SQL的子游标过多引起SQL解析时遍历library cache object handle链表需要很长时间，造成了library cache: mutex x等待。



在数据库中可以看出大量loaded_version超过1000的SQL语句，并且其中有大量游标是过期的。其中SQL_ID：g14zxrn7wyaxh就是此次library cache lock等待最为严重的SQL。

导致SQL不共享的原因很多，一部分是由于SQL中绑定变量长度不一致导致。

```
select bind_length_upgradeable,count(*)
from v$sql_shared_cursor where sql_id='g14zxrn7wyaxh'
group by bind_length_upgradeable
```

	BIND_LENGTH_UPGRADEABLE	COUNT(*)
1	Y	3292
2	N	1948

采集故障期间的AWR，发现当时DB TIME接近2w，平均活动会话达到1200+。

	Snap Id	Snap Time	Sessions	Cursors/Session
Begin Snap:	86807	11-Oct-19 22:00:01	1578	.9
End Snap:	86808	11-Oct-19 22:15:04	1577	1.0
Elapsed:		15.05 (mins)		
DB Time:		19,168.94 (mins)		

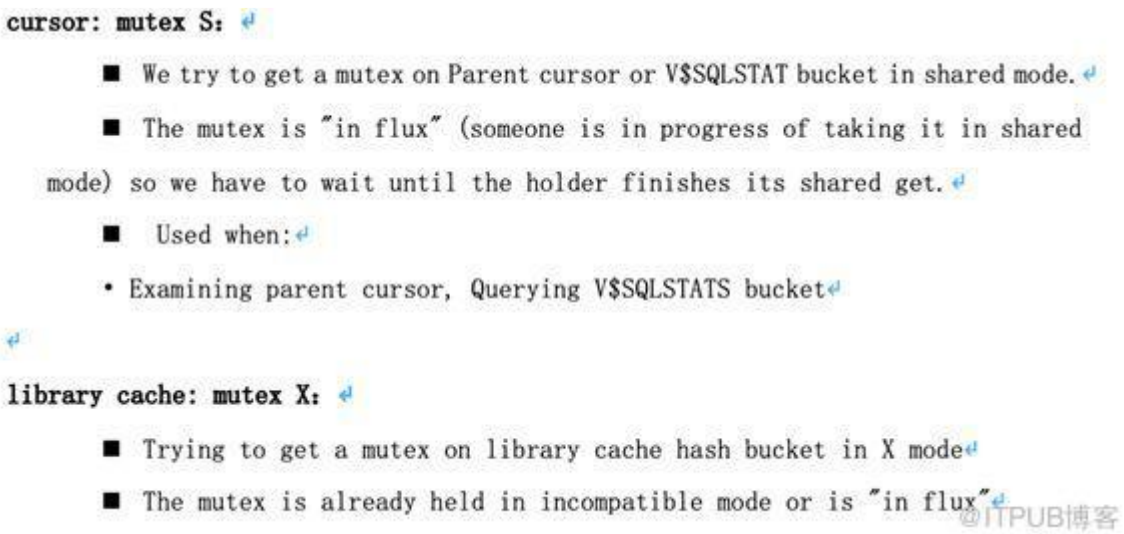
排在前五的等待事件都属于并发类的等待事件，其中cursor：mutex S等待次数最多。

Top 5 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
library cache lock	3,930,384	815,626	208	70.92	Concurrency
library cache: mutex X	8,566,339	257,901	30	22.42	Concurrency
cursor: mutex S	9,066,386	73,529	8	6.39	Concurrency
DB CPU		1,199		0.10	
cursor: pin S wait on X	8,270	264	32	0.02	Concurrency

从ASH中分析library cache lock可以得出，多个会话等待library cache lock主要发生在SQL AREA BUILD的mutex持有争用上。Library cache: mutex X 是10.2.0.2之后library cache latch衍生出来等待。

以下是部分等待事件的含义：



此类等待事件往往都是发生在SQL解析前遍历library cache object handle链表找到shared cursor。

查看AWR中的Mutex Sleep信息发现： Mutex主要有三个函数的sleep是非常高的， kgllkal3 82、 kkshGetNextChild[KKSHBKLOC1]、kglUpgradeLock 119。

函数-kgllkal3 82:kglkal的意思就是kernel generic library cache management library cache lock allocate 82的意思就是SQL AREA BUILD的意思。

函数-kkshGetNextChild [KKSHBKLOC1]:kksh的意思是kernel compile shared objects (cursor) cursor hash table，就是shared cursor的hash链表。持有mutex从library cache 的handle的hash链表上找出可共享的游标。

Mutex Sleep Summary

- ordered by number of sleeps desc

Mutex Type	Location	Sleeps	Wait Time (ms)
Library Cache	kgllkal3 82	29,580,529	0
hash table	kkshGetNextChild [KKSHBKLOC1]	20,237,948	0
Library Cache	kglUpgradeLock 119	10,658,158	0
Library Cache	kgllkdl1 85	154,624	0
Library Cache	kglhdgn2 106	128,465	0
Cursor Pin	kkslce [KKSCHLPIN2]	106,785	0

查看library cache中namespace的命中：

Library Cache Activity

- "Pct Misses" should be very low

Namespace	Get Requests	Pct Miss	Pin Requests	Pct Miss	Reloads	Invali- dations
ACCOUNT_STATUS	2,614	0.00	0		0	0
BODY	25	0.00	64,158	0.00	0	0
CLUSTER	9	0.00	9	0.00	0	0
DBLINK	2,615	0.00	0		0	0
EDITION	1,565	0.00	2,873	0.00	0	0
HINTSET OBJECT	372	0.00	372	0.00	0	0
INDEX	5,335	0.00	5,335	0.04	2	0
QUEUE	182	0.00	197	0.00	0	0
SCHEMA	1,308	0.00	0		0	0
SQL AREA	48,802	0.47	300,694	6.13	10,740	1,170
SQL AREA BUILD	190,333	0.12	0		0	0
SQL AREA STATS	1,797	62.94	1,797	62.94	0	0
TABLE/PROCEDURE	21,460	0.00	271,263	0.06	117	0
TRANSFORMATION	91	0.00	91	0.00	0	0
TRIGGER	997	0.00	997	0.00	0	0

从AWR中可以看出SQL AREA BUILD被请求次数是最多的，这跟ASH中大量library cache lock是吻合的，SQL AREA中cursro reloads次数也达到10740次。Invali_dations达到1170次，说明有很多cursor失效了。

造成library cache lock等一系列严重等待事件的原因是大量的过期游标导致sql解析前花了大量时间去遍历library cache object handle，问题SQL的5415个cursor中有5367个是标记为过期的，查看游标不能被共享的原因：

```
SQL> select count(*) from ssc_reason where reason like '%<reason>Bind mismatch(14)</reason>%';
      3447

SQL> c/14/22
  1* select count(*) from ssc_reason where reason like '%<reason>Bind mismatch(22)</reason>%'
SQL> /
      5257

SQL>
```


造成游标不能被共享的原因中有5257个游标的原因是Bind Mismatch(22)，也就是绑定变量的字符长度发生变化，从32位升级到128位。

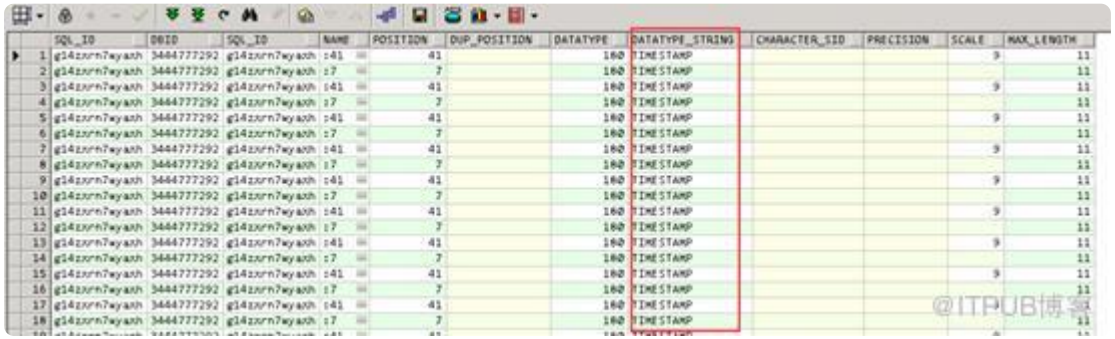
```
<reason>Bind
mismatch(22)</reason><size>4x4</size><bind_position>40</bind_position><original_oacflg>3</original_oacflg><or
iginal_oacmxl>32</original_oacmxl><upgradeable_new_oacmxl>128</upgradeable_new_oacmxl>
```



```
SQL> select count(*) from ssc_reason where reason like '%<reason>Bind mismatch(14)</reason>%' and reason like '%<bind_position>6</bind_position>%';
      3294

SQL>
```


其中Bind mismatch(14)的也有3294个，这个主要是绑定变量TIMESTAMP类型传值到DATE类型导致的问题。Bind mismatch(14)多发生在第6个绑定变量上，对应表中第7个字段，该字段正好的DATE类型。

```
<ChildNode><ChildNumber>39</ChildNumber><ID>34</ID><reason>Rolling Invalidate Window Exceeded(3)</reason><size>2x4</size><invalidation_window>1570825587</inva
gctm>1570835484</ksugctm></ChildNode><ChildNode><ChildNumber>39</ChildNumber><ID>40</ID><reason>Bind mismatch(14)</reason><size>4x4</size><bind_position>40</b
al_oacflg>3</original_oacflg><original_oacscl>0</original_oacscl><new_oacscl>9</new_oacscl></ChildNode>
<ChildNode><ChildNumber>44</ChildNumber><ID>40</ID><reason>Bind mismatch(14)</reason><size>4x4</size><bind_position>40</bind_position><original_oacflg>3</orig
l_oacscl>0</original_oacscl><new_oacscl>9</new_oacscl></ChildNode><ChildNode><ChildNumber>44</ChildNumber><ID>40</ID><reason>Bind mismatch(22)</reason><size>4
ion>25</bind_position><original_oacflg>3</original_oacflg><original_oacmxl>32</original_oacmxl><upgradeable_new_oacmxl>128</upgradeable_new_oacm
xl></ChildNode>


153 rows selected.

SQL> l
  1* select * from ssc_reason where reason like '%<reason>Bind mismatch(14)</reason>%' and reason not like '%<bind_position>6</bind_position>%'
SQL>
```



	SQL_ID	DBID	SQL_ID	NAME	POSITION	DUP_POSITION	DATATYPE	DATATYPE_STRING	CHARACTER_SID	PRECISION	SCALE	MAX_LENGTH
1	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
2	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
3	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
4	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
5	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
6	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
7	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
8	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
9	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
10	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
11	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
12	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
13	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
14	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
15	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP			9	11
16	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11
17	g14zxrn7wyash	3444777292	g14zxrn7wyash	:41	41		180	TIMESTAMP				11
18	g14zxrn7wyash	3444777292	g14zxrn7wyash	:7	7		180	TIMESTAMP				11

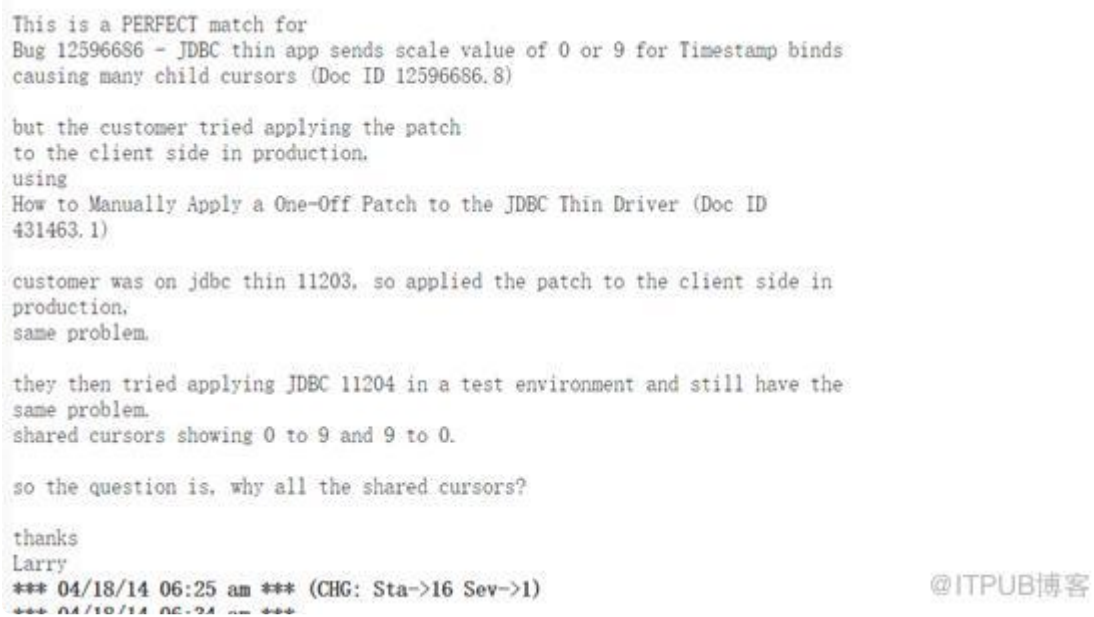
综合以上分析,造成大量游标过期的原因有以下两个:

1、绑定变量长度导致游标无法共享

2、JDBC的bug导致日期类型通过TIMESTAMP传值，继而导致绑定变量无法共享

相关bug:

```
Bug 18617175 : JDBC THIN SENDS SCALE VALUE OF 0 OR 9 FOR BINDS CAUSING MANY CHILD CURSORS
Bug 12596686 : JDBC THIN APP SENDS SCALE VALUE OF 0 OR 9 FOR BINDS CAUSING MANY CHILD CURSORS
Patch 12596686: JDBC THIN APP SENDS SCALE VALUE OF 0 OR 9 FOR BINDS CAUSING MANY CHILD CURSORS
```



从上面截图的MOS文档来看，JDBC版本升级到11203或11204仍有发生此例绑定变量传值问题。

为什么重启应用无法解决?

1、kill session：故障发生后数据库端进行kill session操作，但是因为有连接池，所以连接池会尝试重连数据库，kill 后的重连在连接池上几乎是并发的，因此负载也很高，所以kill session不行；

2、重启应用：重启应用前数据库端的latch竞争一直都有，大量的活动会话并没有释放。如果这个时候重启应用还是会有新的连接进来，这些新进的连接依然会进入到队列中等待，继而加剧争用，因为重启并不会中止数据库上之前的连接，所以重启应用也不行；

3、关闭应用并kill session：应该关闭应用，然后数据库端kill session，再启动应用。

问题解决

建议一：后期进行历史分区清理的操作（DDL操作同类）时，需提前查询表上SQL的游标是否超过200，如超过这个阈值，应主动使用DBMS_SHARED_POOL.PURGE的方式将过期的游标清理出内存，尽可能的减少遍历游标HASH链表时间较长的现象；

查询并清理过期游标的SQL：

```
select q'[exec sys.dbms_shared_pool.purge(']' || address || ',' ||
        hash_value || q'[','C');]' as flush_sql
  from v$sql t
 where t.sql_id = '&sqlid'
   and t.is_obsolete = 'Y'
   group by address,hash_value;
```

建议二：从应用层面，建议将前述同一个SQLID(g14zxrn7wyaxh)的SQL文本，通过在原SQL文本中，加入不同的注释，从而将其变为若干个不同SQLID，但功能相同的SQL。其目的也是业务峰期时，将访问分散到不同的父游标上。

其他建议：

1、将单个SQL游标总数加入到监控告警中，前提是v$sql_shared_cursor中的游标总量在阈值内，目前根据测试和经验总结建议阈值设置为200；

2、数据库分区维护操作属于DDL操作，影响较大，应选择业务最低峰期进行操作；

3、数据库上执行DDL操作时，应实时监控数据库的活动会话等待事件，如果出现mutex或latch等待持续上升，应立即取消DDL操作，并持续监控数据库性能。