Constraint based Performance tuning of Oracle Database

S.K.Srivastava
DBA
Indorama Synthetics Tbk.

Abstract
Many times DBA faces problem of sudden performance drop, sometimes the full Database faces this problem and sometimes some selected sessions face the same. Normally it happens during peak load period so immediate rectification is needed. How to rectify these performance issues immediately? The most difficult issue is to identify the cause of the performance degradation. Sometimes it happens in the cases where no recent change in parameters of database is being done.

In Order to identify these problems oracle has provided a very good source of information i.e. v$ views. There are few views in v$ views family which gives information on immediate bottleneck in the database, they can tell exact reason of performance degradation. It is very helpful for immediate identification and thus rectification of the problem. In addition to these views, Operating system can also provide very useful information as sometimes these types of problems are caused by operating system.

Introduction

In a Production database many times DBA receives complaints from users that Database is running slow, DBA checks the database and found no recent change in configuration so where is the problem? Many times we receives this type of issues and we check hit ratio, hit ration is perfectly correct then how to identify the problem.

All the performance related issues are linked with resource availability i.e. whenever sessions are waiting for resources then each activity will take more time and hence database will become slow. This could be due to insufficient Hardware resource or due to excessive utilization of resources by some of the sessions. If a system is not having adequate Hardware resources then performance problem should be there during most of the peak load period and thus will not be sudden drop in performance, whereas the problem is of sudden excessive resource utilization then performance drop will be sudden i.e. whenever such excessive utilization will be there server's performance will degrade.

For both type of problems it is extremely important to identify the reason of the problem, once reason is known then rectification can be done accordingly. How to find out exact reason of the performance degradation? The Answer of this type of problem is Wait Events, Oracle Database is having very powerful feature of recording any waits encountered by any session.

In order to solve sudden drop in performance issue followings are the main area, where one can investigate –

1.V$ wait views
2. Problematic SQL identification and Tuning by DBA
3. O.S related issues

V$ Views

These are the following V$ views, which are having wait related information
(As we know V$ views are views which are based on the information stored in control file and Memory, so all the information shown by these views will go away during shutdown and startup of the database)

V$SYSTEM_EVENT – This view shows cumulative wait information of all the sessions since instance startup, this view gives listing of events which are the main areas where server is waiting.

V$SESSION_EVENT – This view shows cumulative wait information of the sessions since the session creation.

V$SESSION_WAIT - This view shows the wait events faced by each session, this is an online information so anytime all the waits in database faced by sessions can be viewed from this view.
This is very important information as it directly indicates the reason of sudden performance drop.
This view shows online non-cumulative information so it directly indicates current situation of the database.

SQL> DESC V$SESSION_WAIT
SID - Session Identifier
SEQ# - Session Identifier
EVENT - Name of Wait event
P1TEXT - Parameter1 Name
P1 - Parameter1 value
P1RAW - Parameter 1 raw value (Hexadecimal)
P2TEXT - Parameter2 Name
P2 - Parameter2 Value
P2RAW - Parameter2 raw value (Hexadecimal)
P3TEXT - Parameter3 Name
P3 - Parameter3 Value
P3RAW - Parameter3 raw value (Hexadecimal)
WAIT_TIME -
SECONDS_IN_WAIT -
STATE -

The last three columns i.e. STATE, WAIT_TIME and SECONDS_IN_WAIT are very important to find out the actual wait time for each session's event. Value of WAIT_TIME and SECONDS_IN_WAIT is dependent of value of STATE.

If value of STATE field is WAITING, it indicates that session is currently waiting and it is waiting since n seconds, where n is the value of column SECONDS_IN_WAIT.

If the value of STATE is WAITED_KNOWN_TIME then it indicates that session has got the resource for which it was waiting. Total wait time for this will be m seconds where m is the value of column WAIT_TIME.

There are two types of events, one is idle event i.e. these events do not create any performance related problems. Where as few events are non-idle events, these events directly impacts on performance.

In normal case when database is not facing any performance problem then v$session_wait will show very few sessions where as when the database is facing performance problem then v$session_wait will start showing many sessions waiting for non-idle events. If a sudden bottleneck is there then you will notice that many sessions are waiting for one or two types of non-idle event. ***The reason of the problem is the event for which most of the sessions are waiting.***

Now on once event or root cause is identified then rectification needs to be done, for e.g. if the event is *db file scattered read* or *db file sequential read* then try to find out the sessions which are doing large amount of I/O. The selected sessions should be killed and they should be scheduled for off peak period. There could be *latch free* event of type Shared Pool latch, try to find out the blocker session and then clean the blocked session. Similarly different action can be taken for different types of event.

In order to go in depth of these events, it is important to understand the information stored in different parameters. These parameters store different values for different events, the exact description of each parameter for a specific event can be taken out from V$EVENT_NAME view.

In order to explain these parameters, we can take event *db file scattered read*, in order to find description of different parameters following sql can be used

```
select PARAMETER1,PARAMETER2,PARAMETER3 from
v$event_name
Where name like 'db file scattered read';
```

PARAMETER1 PARAMETER2 PARAMETER3
---
file# block# blocks

So Parameter1 indicates file number where as Parameter 2 indicates starting block number where as blocks indicates number of requested block for this IO event. In case of full table scan number of requested block should be equal to *db_file_multiblock_read_count* . So from this event details one can find out how effectively system is picking blocks during full table scan.

There are many wait events, which are there in database, but only few of them are critical ones.
Few of them are listed below-

db file sequential read – This event indicates about I/O contention faced during Index scan
db file scattered read– This event indicates about I/O contention faced during full table scan
db file parallel write – This wait is related to database writer. Sometimes during sorting data is written on temporary file if there is some contention or wait in this, then this event will be show as waiting event.
latch free – session is waiting to acquire a latch , details of latch can be derived from p1,p1raw, p2 and p3.
log file sync – when a session commits then log buffer needs to be flushed and data should gets written on redo log files, if there is any delay in this operation then session will show *log file sync* event as waiting.

SQL*Net message to client, SQL*Net more data to client, SQL*Net message from client,
rdbms ipc message block, rdbms ipc reply, rdbms ipc message etc. are not very important event from tuning point of view so they can be ignored.

In order to capture all these wait data into v$ views it is necessary to enable TIMED_STATISTICS parameter in init.ora file.

The above view will keep on changing very frequently, so if someone wanted to know summary of waits for a particular session then V$SESSION_EVENT will give the needed information.

SQL> DESC V$SESSION_EVENT

SID -- Session Identifier
EVENT -- Name of the wait event
TOTAL_WAITS -- Total number of waits for this event by this session
TOTAL_TIMEOUTS --Total number of timeouts
TIME_WAITED -- Total amount of time waited in centi seconds
AVERAGE_WAIT -- Average wait time in centi seconds
MAX_WAIT -- Maximum time waited in centi seconds

SQL identification which are creating problem-

From monitoring of v$session_wait one can find out the sqls which are creating performance problem in database. We can take example of db file scattered read, one can find the sql text, which is actually doing the full table scan. For e.g. from v$session_wait sid – 505 is doing full table scan
Then –

```
select sql_text from
v$sqlarea where
address in
(select sql_address from v$session
where sid=505)
```

Will return sql text, which is doing full table scan.

Once sqltext is retrieved then sql*trace with tkprof can be used for further analysis. Hence the wait information can find out sqls , which may need tuning.

There are multiple tricks to tune sqls but for a DBA there are very limited options of tuning, as normally DBA do not understand functionality of the SQL and any change in SQL may result wrong data. Important tools for DBA to tune the overall cost of sql is indexing, optimiser mode and hints. With use of these tricks without changing core code performance can be increased.

In order to use these tools one must understand sql*trace and tkprof, because from tkprof's output one can get clue that for which table and column indexing is required. Creation of Bit map indexes, function based indexes are also very useful as sometimes they improves sql 's performance drastically.

Operating System Related critical issues: -

Swapping-

Swapping is one of the very critical OS event on the server, which may slowdown the whole database drastically.

On operating system memory is divided into small units called page, whenever memory requirement increases from available physical memory then OS transfers few pages from memory to Hard Disk, this process is called Paging. This process is a common process, impact of this activity on system's performance is not very adverse. But if memory requirement further increases then after a limit OS transfers some full processes instead of transferring the pages, this process is called swapping. Swapping is all together is totally unaccepted process from system performance point of view. Drastic performance degradation will be there in case of swapping.

How to rectify this problem, one solution is to add memory and second one is to check parameter setting. Normally whenever some major changes is done in system's design which increments the memory requirement and physical memory is not available then swapping may happen. Be careful during migration or upgradation because if memory requirement is not calculated properly then swapping may happen.

Regarding memory requirement, it is important to check following parameter as memory allocation for these parameters are done on session basis.

sort_area_size
sort_area_retained_size
hash_area_size etc.

For unix based system **vmstat** is very useful command, swapping information can be found out from this command.
e.g.-

```
>vmstat 5
procs     memory            page            disk          faults      cpu
 r b w   swap  free  re  mf pi po fr de sr s0 s6 s3 sd   in   sy   cs us sy id
0 0 0 8034552 384976 0   0  0  0 251 0 38  0  0  0  0 1575  838  501  1  4 95
 0 0 0 8034552 384960 0   0  0  0 124 0 21  0  0  0  0 1558  866  539  1  2 97
 0 0 0 8034608 384992 0   0  0  0  0  0  0  9  0  9  0 1551  471  274  0  1 99
```

Numbers appearing under column with heading **w** is actually number of current processes, which are being swapped.

vmstat –s directly gives overall picture of swapping event since OS startup.

Similarly there are multiple commands for OS, which can be used very effectively to find out swapping and cpu utilization etc. information.

Hanged Process-

There could be one more reason of overall performance problem degradation, hanged process on operating system. They consume a lot of CPU and memory. There should be regular monitoring of such processes and it should be removed from database and OS.

Following command works for Solaris

ps –e –o pcpu,pmem,pid|sort –k 1 – r|more

This will show process listing along with cpu and memory utilization information.
From pid corresponding sid and serial# can be retrieved from the database. These processes are the processes, which are outcome of improper way of logging out from system or due to some bugs. All such processes should be cleaned automatically but sometimes they are not removed automatically then manual intervention is needed.
(There should be similar command for other operating system also, DBA must find out this command as they are very useful during immediate rectification of server performance issue.)

I have found that in few version of 11i, form60run processes takes large amount of CPU.
Sometimes database becomes very slow due to heavy consumption of this type of process.

Last but not the least, one solution is very common during performance degradation, it is shutting down and startup the database. It happens with all of us, whenever no tricks are working then first thing comes into mind is shutdown and startup. Whenever database is shutdown and restarted then OS should also shut down such that hanged process gets cleaned.

Especially in Oracle Financials 10.7 and 11i, I have noticed that sometimes concurrent processes also remain there even though database is down. This happens due to improper way of shutting down the concurrent managers, i.e. sometimes we shutdown the database without waiting for complete stoppage of the process related with Concurrent managers. For Oracle financials DBA one more important point related with concurrent programs, there is one setting of restart the program in case of system failure. This setting is ***Restart on System failure,*** defined at the time of Program registration. Some time if some concurrent program is creating locking problem and database is shutdown and started then these programs will start again and will create locking problem once again. So whenever shutdown and restart operation is done in the case of performance degradation, the running concurrent program should be checked before shutting down.

I have tried my best to put points on immediate database tuning but there could be many more points, which can be very effective for immediate tuning.

About the Author

S. K. Srivastava Bachelor of Engineering from Indian school of Mines (1995).

Oracle Certified Professional (9i), 3.5 years Oracle Application DBA (along with Solaris Sys Admin), and 3 years on Oracle Apps Techno functional. I have been involved in the following projects:

- Oracle apps Migration (10.7SC to 11.5.3 (11i)), role Oracle apps DBA (along with Solaris Administration), Company - Indorama Synthetics Tbk, Indonesia
    - Oracle Application Migration from 10.7 SC to 11.5.3 (11i) on Solaris Platform
    - Server Sizing for 11i
    - Oracle Database Tuning
    - Solaris Installation and Patch Application
    - Veritas Reconfiguration for optimum I/O
    - Database backup (Hot, Cold and Export Backup) and testing recovery

- Oracle apps 10.7SC Administration, role Oracle apps DBA, Company - Indorama Synthetics Tbk, Indonesia
    - Oracle Apps 10.7 SC Maintenance
    - Database Tuning (Wait Events, Hit Ratios and tkprof) – optimization of init.ora parameters
    - Veritas Installation and Raid (0+1) implementation
    - Database backup (Hot, Cold and Export) and testing Recovery
    - Oracle Multi Threaded Server Implementation

- Oracle apps 10.7SC (AR – OE implementation), role–Oracle Apps Techno-functional Team Leader, Company - Indorama Synthetics Tbk, Indonesia
    - Oracle Apps (Order Entry and Oracle Receivable) 10.7 SC Implementation
    - Customized New Modules in Oracle Apps (It Included Export Documentation, Packing, LC and Quality Control Module)
    - Developed multiple programs to use Oracle Open Interfaces (Order open Interface, Customer open Interface, Inventory Open Interface and Receivable open Interface)

- Oracle apps 10.6 (Inventory) and Maximo implementation, role – Programmer, Company – Punj Lloyd Limited, India
    - Oracle Apps Implementation (Oracle Inventory) 10.6
    - Maximo Implementation
    - Open Interface implementation between Maximo and Oracle

Email: - sksrivastava@indorama.com