Database Performance Monitor Utility

In the past five years, I am managing the world's biggest database system for online payment service (AliPay of Alibaba Group), it handles 100 million trades on 2012/11/11, totally 4 billion database transaction, 28.5 billion SQL executions, 193 billion memory data block touches, 15 terabytes database log files generated at that day. So for every database (including Oracle and MySQL), I have to know how exactly the database load changes according to the business volume, so I need to gather a lot of database running performance data and the business volume data, and do a further dig of the relation between the business complexity and the database capacity.

I wrote an Oracle performance monitor (I called it oramon) to record the database performance every 10 seconds, and then summarize it to one minute duration, because I think a sudden spike (just take few seconds) is ok for our system, but minute is not ok, which can erase some accident cases to make the performance data more suitable for capacity planning. However oramon is dedicated designed for Oracle only, and the output format is coded in the program, so it need change for different Oracle version, I will not recommend it to you now.

I want to introduce the new program to you, you can create your own database performance monitor quickly.

KEY VALUE DATA

Every performance data is a key value pair, the key is the performance point name, and the value is the performance point value. And the value has two different types, first type is a difference data, which mean the valid value should be the different between two different timestamp, such as SQL execute count in Oracle system statistics, second type is a current data, such as the active sessions in Oracle database at specific timestamp, the total session count at specific timestamp.

THE NEW UTILITY

"orastats" is the new utility for Oracle database performance monitor, "mysqlstats" is the new utility for MySQL database performance monitor. It contains three types of performance data (key value pairs).

- The timestamp

  Tell you the current timestamp of the performance data. The key name is “orastats.timestamp”.

- The OS performance data

  Some OS performance data include process, CPU usage, memory usage, network traffic etc, every value is calculated as current time value (valid on Linux x86 or Linux x86_64 only).

Key Name	Description
os.load	1 minute load multiply by 100
os.run	CPU run queue length
os.process	Total process count
os.cpu.usr	User CPU usage percent
os.cpu.sys	Kernel CPU usage percent
os.cpu.wio	IO wait CPU usage percent
os.fork	New process fork count since last timestamp
os.irq	Total interrupt times
os.softirq	Software interrupt times
os.context	Context switch count since last timestamp
os.uptime	System up time by hours
os.pagein	Memory page in operation
os.pageout	Memory page out operation
os.swapin	Page read from swap area, should very closed to 0
os.swapout	Page write to swap area, should very closed to 0
os.mem.free	System free memory
os.mem.swap	Swap space used totally, should very closed to 0
os.mem.cache	File cache memory used
os.mem.page	Page cache memory used
os.net.ibytes	Network Input Traffic
os.net.obytes	Network Output Trafic
os.net.ierrors	Network Input Errors
os.net.oerrors	Network Output Errors
os.file.nr	File description used percent

  The OS performance data comes from where you run the utility, so usually you need to run it on the database server.

- The database performance data

You need to tell the new utility a SQL query (which should return two columns, first column as the key name with character type, second column as key value with number type) to get the performance data from database performance views. In Oracle, there are a lots of dynamic performance data, such as “V$SYSSTAT”, “V$SYSTEM_EVENT”, “V$SESSION” etc. In MySQL, you should use “show global status like ‘%innodb%’” command to get the performance data. The performance data is based on what SQL query you pass to the utility, which is under your control.

COMMAND LINE OPTIONS

The new utility accepts eight command line options.

- user
  Database connection information, a string value with “user/pass@host:port:db” pattern. If you don’t specify this value, Oracle will be “sys”, and MySQL will be “/@::test”, both try to make a local database connection.
- query
  The SQL query to get the performance data from database, such as:

  select name, value from v$sysstat

  You can use a UNION all to query performance data from multiple tables.
- format
  Control the output format of the performance data, if you don’t specify it will print all the performance data with “key=value” format, such as:

  total number of times SMON posted=1131.000000
  SMON posted for dropping temp segment=27.000000
  SMON posted for undo segment shrink=112.000000
  redo size for lost write detection=0.000000
  gc claim blocks lost=0.000000
  HSC OLTP positive compression=0.000000
  SMON posted for undo segment recovery=0.000000
  redo write broadcast lgwr post count=0.000000
  SMON posted for instance recovery=0.000000

  If you specify this option with correct value, the output will be formated as

following, which make it more readable.

```
2013-04-09 13:31:00 Ctxt SY WI US Exec Read Cget
2013-04-09 13:31:00   239  0  0  1    1    0    0
2013-04-09 13:31:01   382  0  0  0    1    0    0
2013-04-09 13:31:02   152  0  0  0    1    0    0
2013-04-09 13:31:03   256  0  0  0    6    0   16
2013-04-09 13:31:04   188  0  0  0    1    0    0
2013-04-09 13:31:05   183  0  0  0    1    0    0
2013-04-09 13:31:06   159  0  0  0    1    0    0
2013-04-09 13:31:07   187  0  0  0    1    0    0
2013-04-09 13:31:08   138  0  0  0    1    0    0
2013-04-09 13:31:09   242  0  0  0    1    0    0
```

For each performance data, you need to specify the key name, the value type (either delta or curr), and a short label, every attributes separated by a vertical line (“|”), assume the vertical line character will not appear in the key name. An example format option will be looked as following.

```
format=os.context|delta|Ctxt|
    os.cpu.sys|curr|SY|
    os.cpu.wio|curr|WI|
    os.cpu.usr|curr|US|
    execute count|delta|Exec|
    physical reads|delta|Read|
    consistent gets|delta|Cget
```

You can create multiple parameter file for different performance data list, and refer them with “parfile” option.

- wait

  The interval time between two timestamp, default is 10 seconds.
- log

  The log file name for formatted data output, by default, it will write to the screen. Each value will be formatted to the same width. If the value was too large, it will be suffixed by “k” or “m”, as following:

2013-04-09 13:43:37 Ctxt SY WI US Exec Read Cget
2013-04-09 13:43:37 3073 0 0 7 47 0 159k
2013-04-09 13:43:38 2887 0 0 2 47 0 166k
2013-04-09 13:43:39 2859 0 0 5 53 0 160k
2013-04-09 13:43:40 2920 0 0 3 46 0 154k
2013-04-09 13:43:41 2898 0 0 4 48 0 165k
2013-04-09 13:43:42 2912 0 0 2 46 0 156k
2013-04-09 13:43:43 2761 0 0 5 48 0 159k
2013-04-09 13:43:44 2973 0 0 2 46 0 161k

So you can use a “tail –f” command to watch the real database performance.

- Data

  The raw data file name for data output, by default, it will not write the raw data file. In raw data file, every performance value is not formatted, and every values is separated by comma character(“,”) as following.

  2013-04-09 13:40:56,229,0,0,0,1,0,0
  2013-04-09 13:40:57,234,0,0,0,6,0,16
  2013-04-09 13:40:58,153,0,0,0,1,0,0
  2013-04-09 13:40:59,229,0,0,0,1,0,0
  2013-04-09 13:41:00,209,0,0,0,1,0,0
  2013-04-09 13:41:01,283,0,0,0,1,0,0
  2013-04-09 13:41:02,136,0,0,0,1,0,0
  2013-04-09 13:41:03,182,0,0,0,1,0,0
  2013-04-09 13:41:04,190,0,0,0,1,0,0
  2013-04-09 13:41:06,177,0,0,0,1,0,0

  You can load the data into database for further analyze operations.

- loop

  The count of the performance data to be displayed, default is 0 which means forever until any errors occurred.

- parfile

  You can write all the above options to a text file, to avoid write them in command line every time you want to run it, let’s check a full parameter file’s content.

  query=select name, value from v$sysstat
  format=os.context|delta|Ctxt|
  os.cpu.sys|curr|SY|

os.cpu.wio|curr|WI|

os.cpu.usr|curr|US|

execute count|delta|Exec|

physical reads|delta|Read|

consistent gets|delta|Cget

wait=10

loop=50

You should create different parameter files for quick reference.

RUN THE UTILITY

If I have created a parameter file (“orastats.txt”), then you can run it with the follow command.

```
$ ./orastats_linux64.bin parfile=orastats.txt wait=1
2013-04-09 13:50:45 Ctxt SY WI US Exec Read Cget
2013-04-09 13:50:45   244   0   0   0     1     0     0
2013-04-09 13:50:46   210   0   0   0     1     0     0
2013-04-09 13:50:47   171   0   0   0     1     0     0
2013-04-09 13:50:48   171   0   0   0     1     0     0
2013-04-09 13:50:49   234   0   0   0     1     0     0
2013-04-09 13:50:50   181   0   0   0     1     0     0
2013-04-09 13:50:51   233   0   0   0    10     0    32
2013-04-09 13:50:52   249   0   0   0     2     0    42
2013-04-09 13:50:53   167   0   0   0     1     0     0
```

You can know more about your database, know more about your business with the new performance monitor program.

GET SOFTWARE

For MySQL Database

http://www.mydul.net/software/mysqlstats_linux32.zip

http://www.mydul.net/software/mysqlstats_linux64.zip

For Oracle Database

http://www.mydul.net/software/orastats.zip

ABOUT ME

My name is Fangxin Lou, Oracle ACE, about 15 years Oracle DBA career life. The author of Oracle data recovery utiltiy (AUL, also named MyDUL). You can get touch with me by skype (anysql) or gmail (anysql@gmail.com).

Thanks!