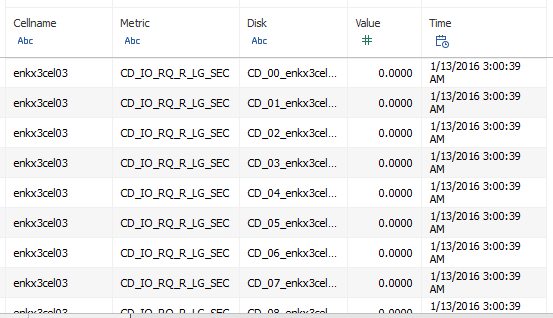
Exadata Storage Cells Performance – Flash vs Hard Disk

By Karl Arao

# The data set

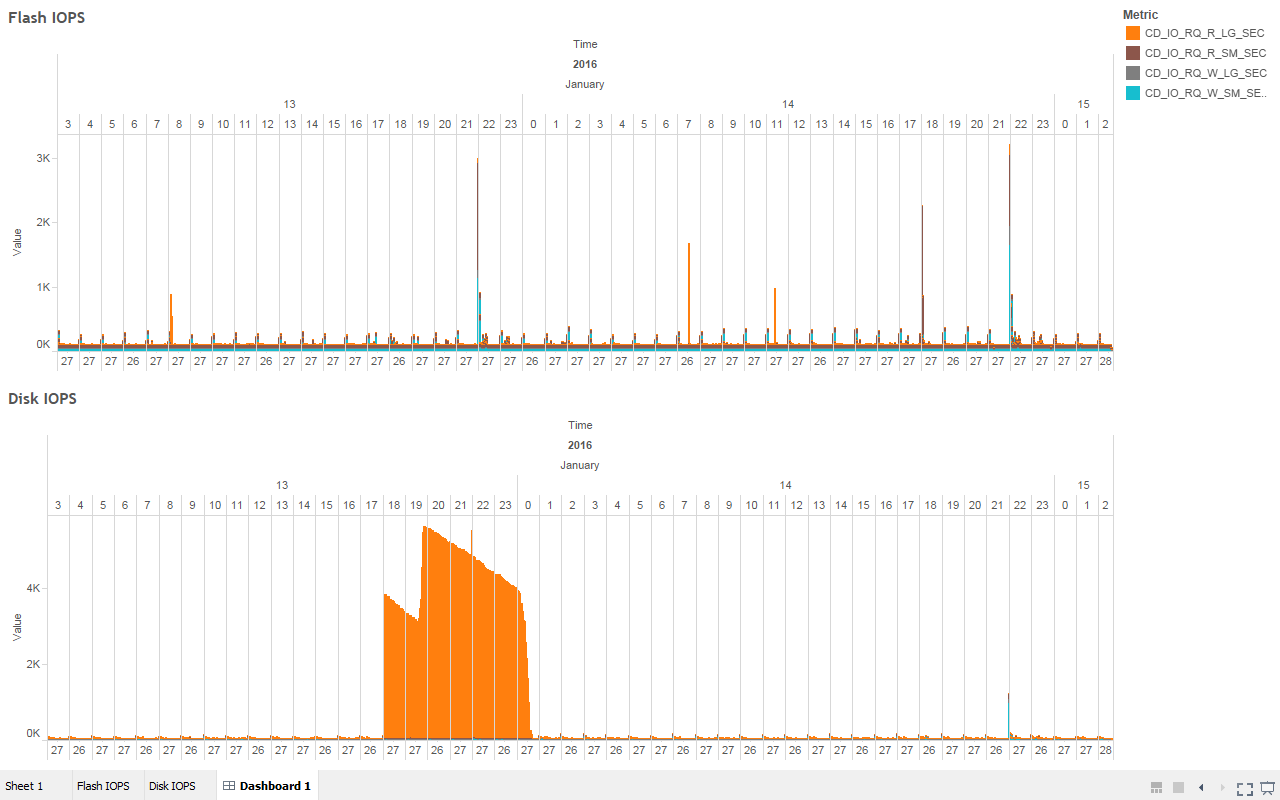
The data set used is coming from the storage cell metrics with the following variables (columns).



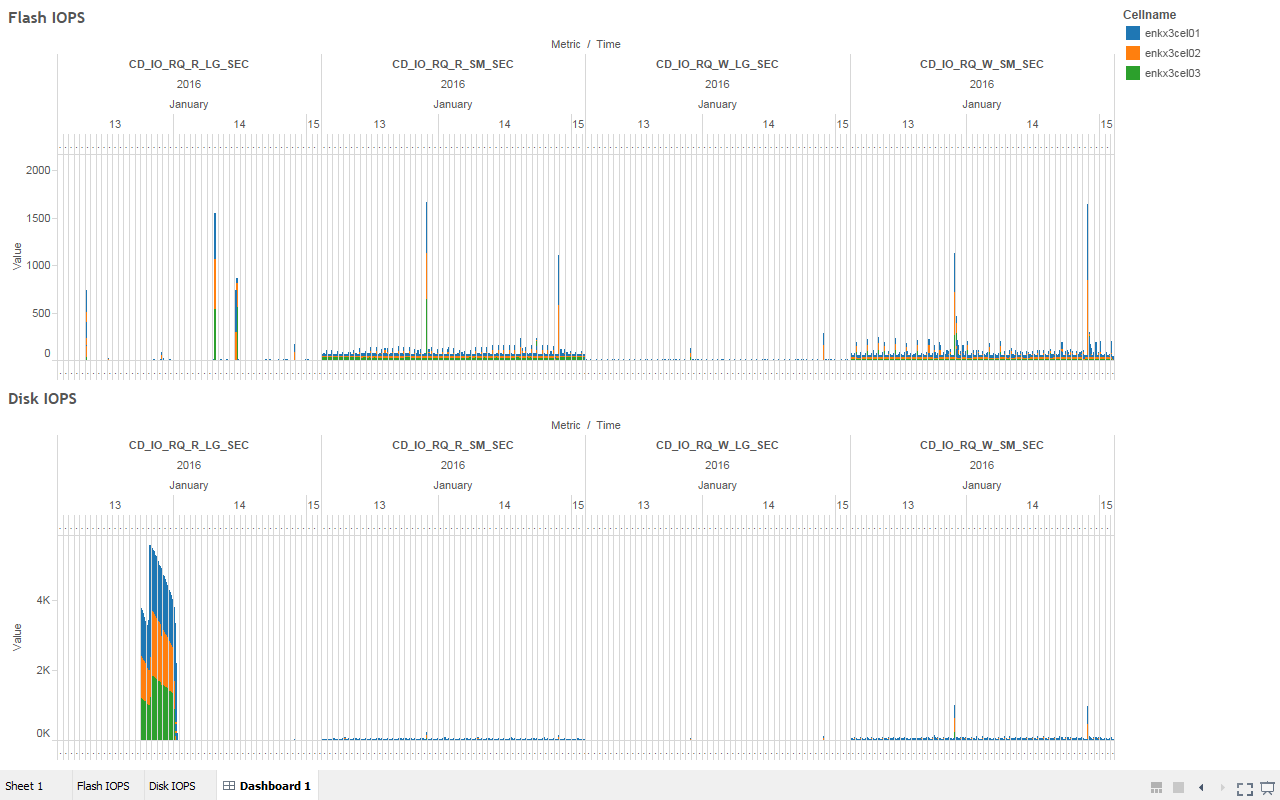
# Example reports

The objective is to generate performance reports for an Exadata Cluster. The analysis can be done by IOPS, Bandwidth, and Latency across dimensions (Flash/HD, IO type, Storage Cell, Grid Disk, etc.). Also this new version has additional metrics about flash cache, flash log, and storage cell health metrics (CPU, memory, filesystem space, etc.). Below are some example graphs:

## Flash vs HD



## Flash vs HD IO type breakdown

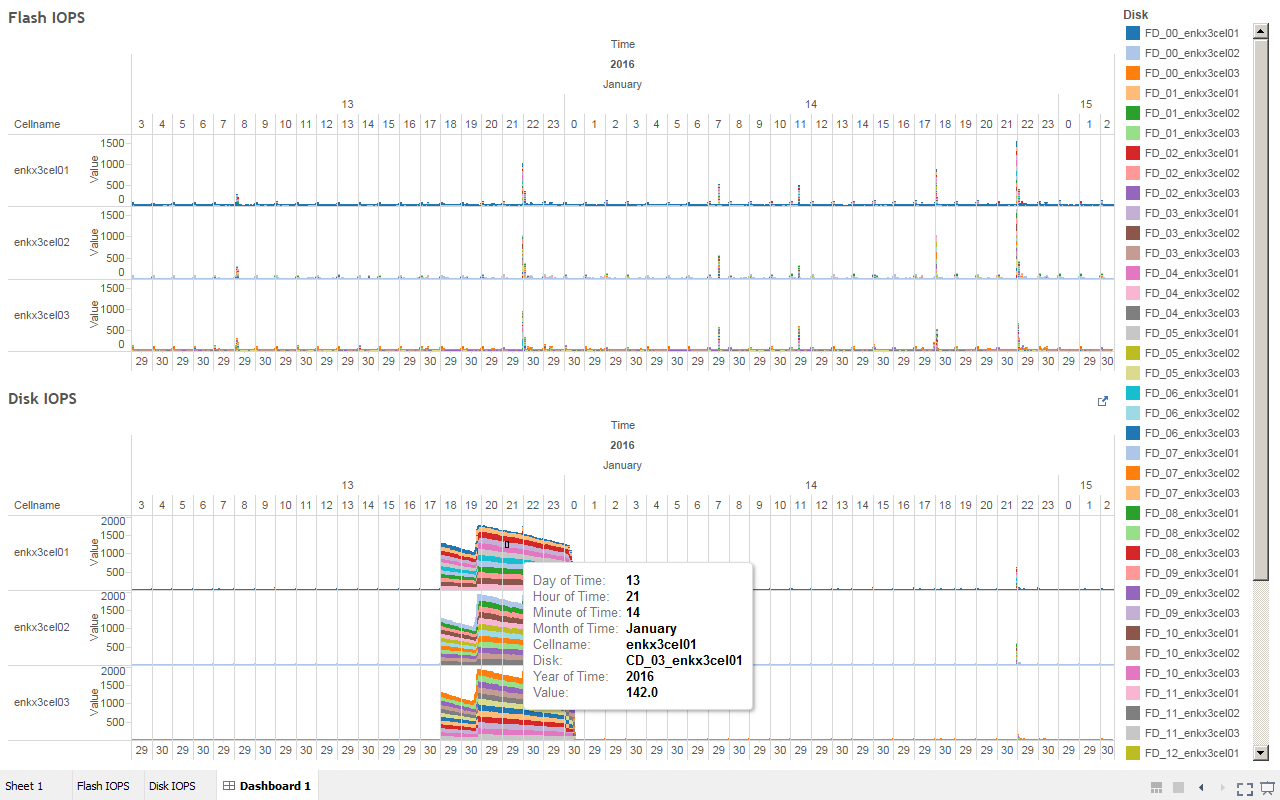


## Flash vs HD by Storage Cell



## Flash vs HD by Grid Disk

Here the CD\_03 HD is shown to be pushing 142 IOPS which is on the range of the limit of the device, meaning at that time it is 100% busy



## New graphs

### Storage cell health metrics

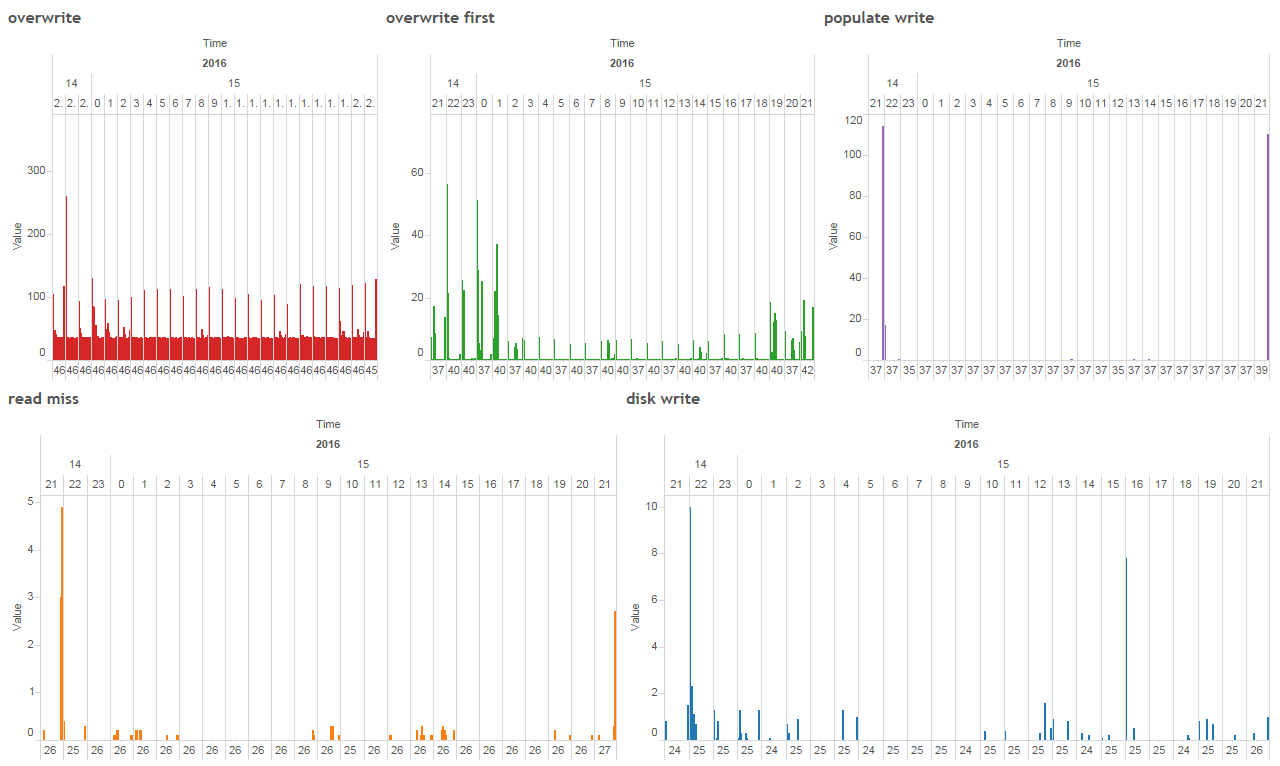
The main metrics are MB/s reverse offload and passthru, CPU%, memory%, filesystem space%, IORM mode

The cell metric (MB/s) SIO\_IO\_RV\_OF\_SEC is "reverse offload" (when there is high CPU on the cells) and SIO\_IO\_PA\_TH\_SEC is passthru (row chaining & migration among them)



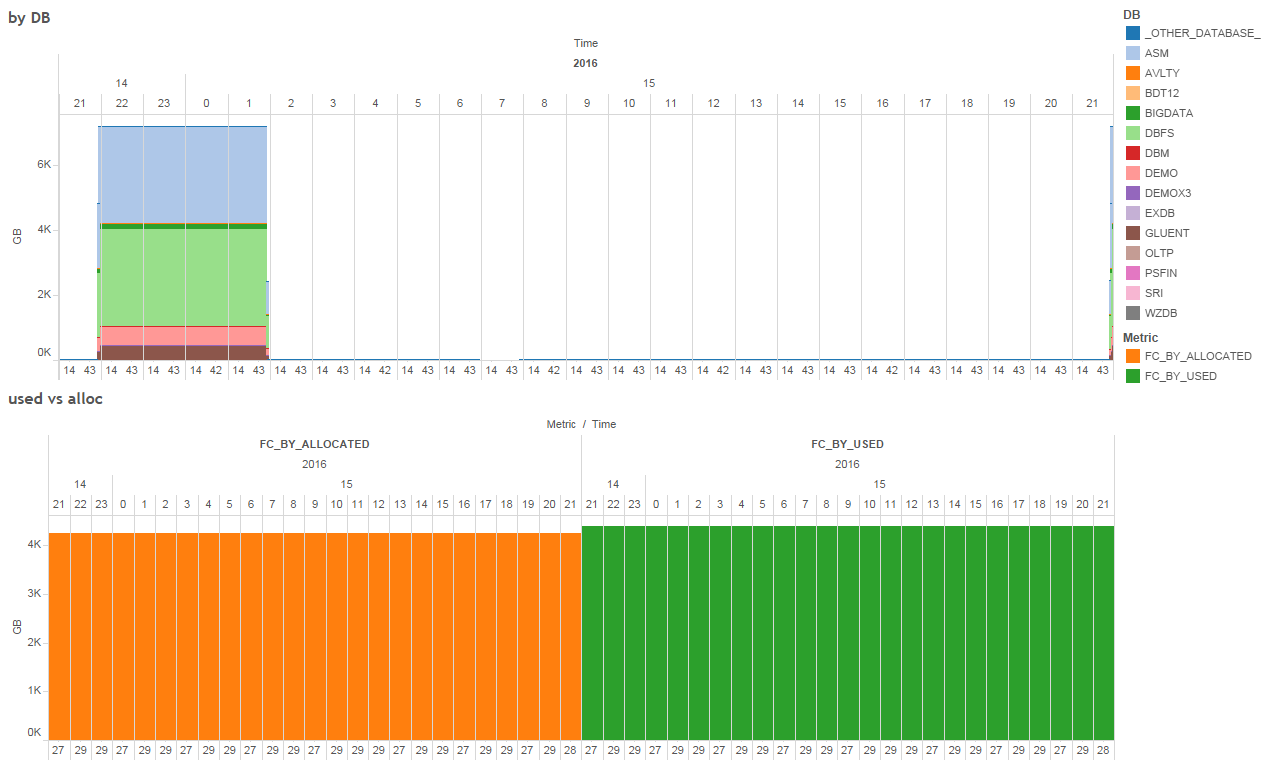
### Flash cache destage

Measures how much destage is happening



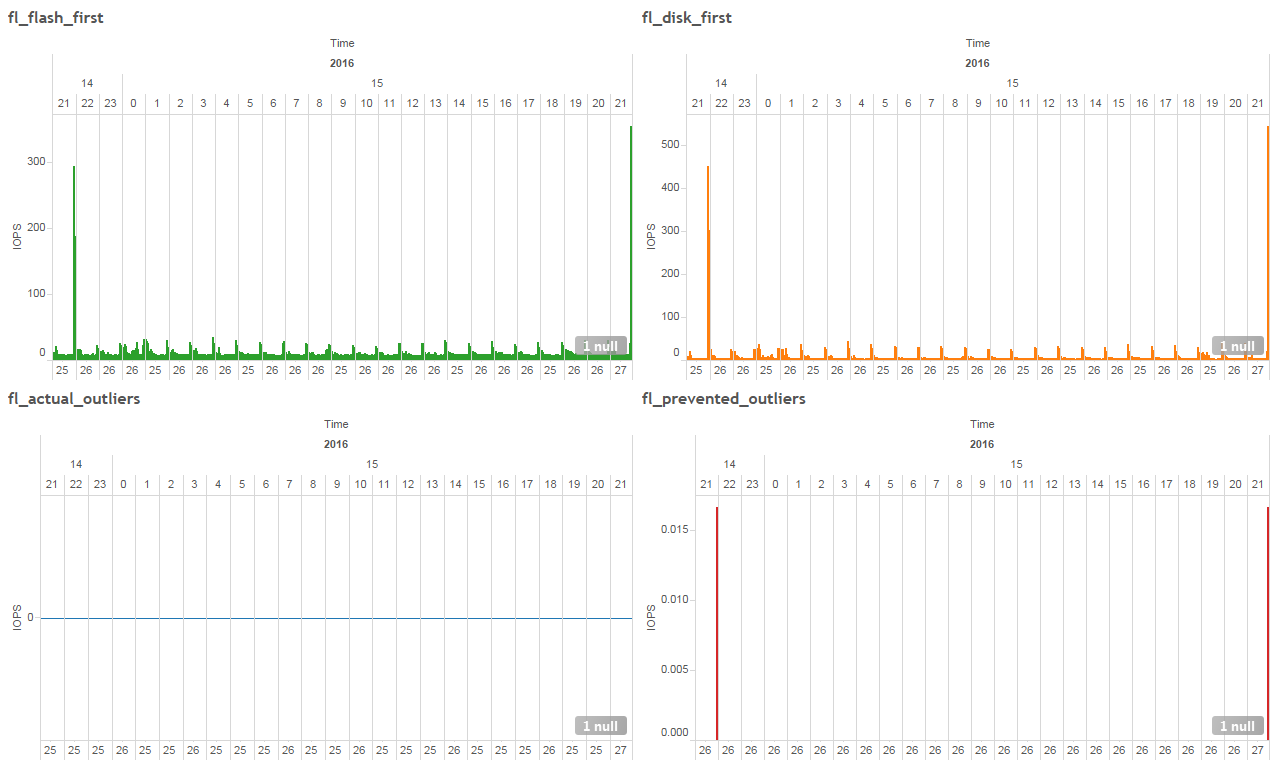
### Flash cache space

The space usage in flash cache



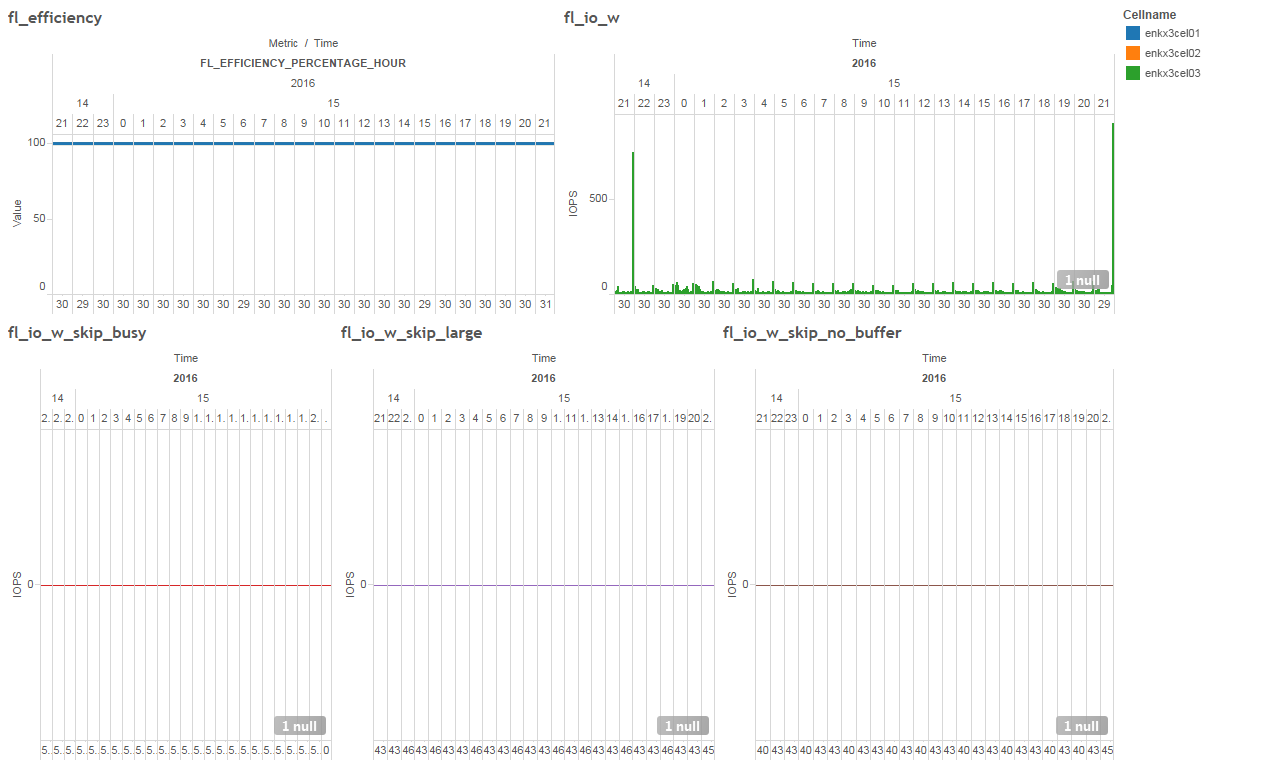
### Flash log outliers

Show occurrence of outliers on the flash log IOs



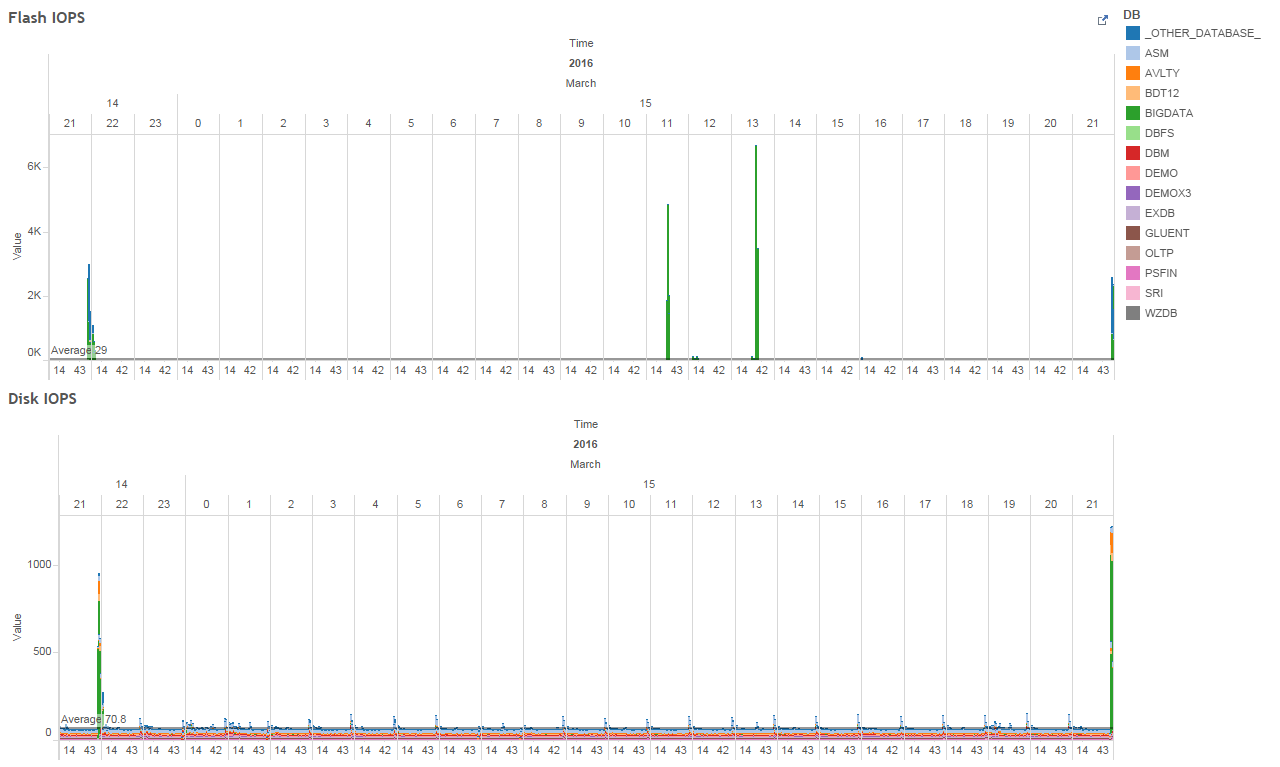
### Flash log efficiency and skip statistics

Flash log health metrics, useful for sizing the flash log



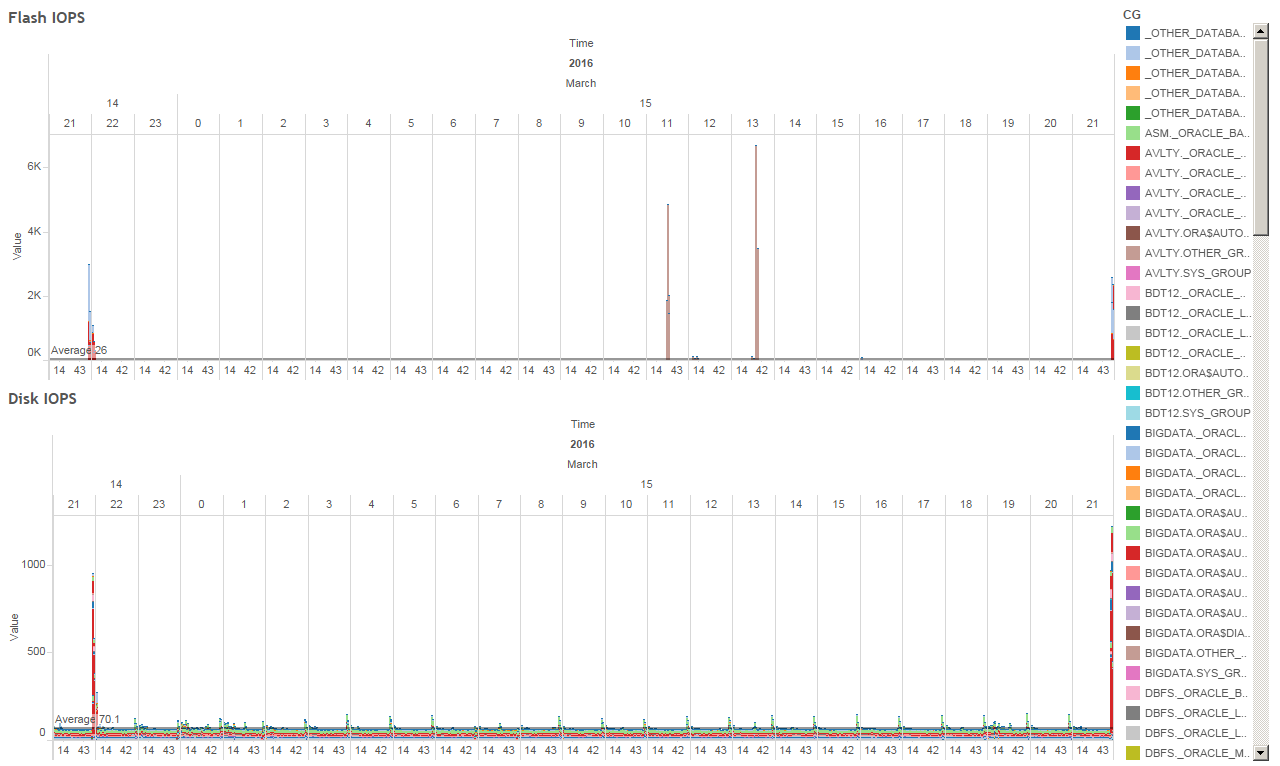
### DB level IOPS, MB/s, latency

Database level IO performance



### CG level IOPS, MB/s, latency

Consumer group level IO performance



# HOWTO

## Login on the 1st storage cell

Make sure the cell\_group file is created and passwordless SSH is working across storage cells

|  |
| --- |
| [root@enkx3cel01 ~]# cat cell\_group  enkx3cel01  enkx3cel02  enkx3cel03  [root@enkx3cel01 ~]# pwd  /root |

## Create the gen\_scl.sh script

|  |
| --- |
| [root@enkx3cel01 ~]# vi gen\_scl.sh  echo "enter start time (format: 2016-03-14T01:00:00-06:00) ->"  read start\_time  echo "enter end time (format: 2016-03-15T21:00:00-06:00) ->"  read end\_time  # cell additional stats  cell\_addtl\_stats="CL\_FSUT,CL\_CPUT,CL\_MEMUT,IORM\_MODE,SIO\_IO\_RV\_OF\_SEC,SIO\_IO\_PA\_TH\_SEC"  # cell iops  cell\_iops="CD\_IO\_RQ\_R\_LG\_SEC,CD\_IO\_RQ\_R\_SM\_SEC,CD\_IO\_RQ\_W\_LG\_SEC,CD\_IO\_RQ\_W\_SM\_SEC"  # cell mbs  cell\_mbs="CD\_IO\_BY\_R\_LG\_SEC,CD\_IO\_BY\_R\_SM\_SEC,CD\_IO\_BY\_W\_LG\_SEC,CD\_IO\_BY\_W\_SM\_SEC"  # cell latency  cell\_latency="CD\_IO\_TM\_R\_LG\_RQ,CD\_IO\_TM\_R\_SM\_RQ,CD\_IO\_TM\_W\_LG\_RQ,CD\_IO\_TM\_W\_SM\_RQ"  # flash space  cell\_flash\_space="FC\_BY\_ALLOCATED,FC\_BY\_USED,DB\_FC\_BY\_ALLOCATED"  # flash destage  cell\_flash\_destage="FC\_IO\_RQ\_W\_OVERWRITE\_SEC,FC\_IO\_RQ\_W\_FIRST\_SEC,FC\_IO\_RQ\_R\_MISS\_SEC,FC\_IO\_RQ\_W\_POPULATE\_SEC,FC\_IO\_RQ\_DISK\_WRITE\_SEC"  # flashlog skip  cell\_flashlog\_skip="FL\_EFFICIENCY\_PERCENTAGE\_HOUR,FL\_IO\_W,FL\_IO\_W\_SKIP\_BUSY,FL\_IO\_W\_SKIP\_LARGE,FL\_IO\_W\_SKIP\_NO\_BUFFER"  # flashlog outliers  cell\_flashlog\_outliers="FL\_FLASH\_FIRST,FL\_DISK\_FIRST,FL\_PREVENTED\_OUTLIERS,FL\_ACTUAL\_OUTLIERS"  # db iops  cell\_db\_iops="DB\_IO\_RQ\_LG\_SEC,DB\_IO\_RQ\_SM\_SEC,DB\_FD\_IO\_RQ\_LG\_SEC,DB\_FD\_IO\_RQ\_SM\_SEC"  # db mbs  cell\_db\_mbs="DB\_IO\_BY\_SEC,DB\_FC\_IO\_BY\_SEC,DB\_FD\_IO\_BY\_SEC,DB\_FL\_IO\_BY\_SEC"  # db latency  cell\_db\_latency="DB\_FD\_IO\_WT\_LG\_RQ,DB\_FD\_IO\_WT\_SM\_RQ,DB\_IO\_WT\_LG\_RQ,DB\_IO\_WT\_SM\_RQ"  # cg iops  cell\_cg\_iops="CG\_IO\_RQ\_LG\_SEC,CG\_IO\_RQ\_SM\_SEC,CG\_FD\_IO\_RQ\_LG\_SEC,CG\_FD\_IO\_RQ\_SM\_SEC"  # cg mbs  cell\_cg\_mbs="CG\_IO\_BY\_SEC,CG\_FC\_IO\_BY\_SEC,CG\_FD\_IO\_BY\_SEC"  # cg latency  cell\_cg\_latency="CG\_FD\_IO\_WT\_LG\_RQ,CG\_FD\_IO\_WT\_SM\_RQ,CG\_IO\_WT\_LG\_RQ,CG\_IO\_WT\_SM\_RQ"  for i in ${!cell\_\*}; do  echo "set dateformat local" >> $i.scl  echo "LIST METRICHISTORY ${!i} where collectionTime > '$start\_time' and collectionTime < '$end\_time'" >> $i.scl  echo "/usr/local/bin/dcli -l root -g /root/cell\_group -f ~/$i.scl" >> distribute\_scl.sh  echo "/usr/local/bin/dcli --serial -l root -g /root/cell\_group "cellcli -e start $i.scl" | bzip2 > $i.txt.bz2" >> run\_scl.sh  done |

## Execute the gen\_scl.sh script

This script will generate the following:

* .scl files
  + The LIST METRICHISTORY commands
* distribute\_scl.sh
  + Distributes the .scl files across the storage cells
* run\_scl.sh
  + Executes all the .scl files sequentially across the storage cells and generates a .bz2 file for each .scl

On prompt, start and end date filters will be asked. Input the proper date filters and time zone accordingly. **The time zone for PST is -8, CST is -6, and EST is -5.**

|  |
| --- |
| sh gen\_scl.sh |

## Execute the distribute\_scl.sh script

Distribute the .scl files across the storage cells

|  |
| --- |
| sh distribute\_scl.sh |

## Execute the run\_scl.sh script

Execute all the .scl files sequentially across the storage cells.

This is a low footprint data collection. The --serial allows running serially for each of the storage cells and the bzip2 is for compressing the data set on the fly which makes the space usage footprint on the filesystem very small.

|  |
| --- |
| sh run\_scl.sh |

Below shows the load average on the storage cell during collection period

[root@enkx3cel01 ~]# top -c

top - 02:48:52 up 15 days, 1:50, 2 users, load average: 1.59, 1.75, 1.62

Tasks: 257 total, 2 running, 255 sleeping, 0 stopped, 0 zombie

Cpu(s): 3.4%us, 2.0%sy, 0.0%ni, 94.6%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st

Mem: 65661560k total, 23323212k used, 42338348k free, 430256k buffers

Swap: 2097084k total, 0k used, 2097084k free, 3984632k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND

18809 root 20 0 13552 6936 484 R 55.4 0.0 0:06.71 bzip2

18808 root 20 0 389m 40m 3396 S 46.8 0.1 0:17.36 python /usr/local/bin/dcli --serial -l root -g /root/cell\_group cellcli -e

## Copy the .bz2 files

Stage the .bz2 files **to a filesystem w/ enough space which is at least 5GB**

## Process the files – create and run the process\_files.sh

The following commands will do the following:

* Extract all the .bz2 files
* Massage the file for visualization
* Zip the final data set

|  |
| --- |
| vi process\_files.sh  bunzip2 \*bz2  cell\_addtl\_stats.txt  sed -n -i '2,$ p' cell\_addtl\_stats.txt  sed -i 's/%/ /g' cell\_addtl\_stats.txt  sed -i 's/MB\/sec/ /g' cell\_addtl\_stats.txt  sed -i 's/: /\t/g' cell\_addtl\_stats.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_addtl\_stats.txt  zip cell\_addtl\_stats.zip cell\_addtl\_stats.txt  cell\_cg\_iops.txt  sed -n -i '2,$ p' cell\_cg\_iops.txt  sed -i 's/IO\/sec/ /g' cell\_cg\_iops.txt  sed -i 's/: /\t/g' cell\_cg\_iops.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_cg\_iops.txt  zip cell\_cg\_iops.zip cell\_cg\_iops.txt  cell\_cg\_latency.txt  sed -n -i '2,$ p' cell\_cg\_latency.txt  sed -i 's/ms\/request/ /g' cell\_cg\_latency.txt  sed -i 's/: /\t/g' cell\_cg\_latency.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_cg\_latency.txt  zip cell\_cg\_latency.zip cell\_cg\_latency.txt  cell\_cg\_mbs.txt  sed -n -i '2,$ p' cell\_cg\_mbs.txt  sed -i 's/MB\/sec/ /g' cell\_cg\_mbs.txt  sed -i 's/: /\t/g' cell\_cg\_mbs.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_cg\_mbs.txt  zip cell\_cg\_mbs.zip cell\_cg\_mbs.txt  cell\_db\_iops.txt  sed -n -i '2,$ p' cell\_db\_iops.txt  sed -i 's/IO\/sec/ /g' cell\_db\_iops.txt  sed -i 's/: /\t/g' cell\_db\_iops.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_db\_iops.txt  zip cell\_db\_iops.zip cell\_db\_iops.txt  cell\_db\_latency.txt  sed -n -i '2,$ p' cell\_db\_latency.txt  sed -i 's/ms\/request/ /g' cell\_db\_latency.txt  sed -i 's/: /\t/g' cell\_db\_latency.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_db\_latency.txt  zip cell\_db\_latency.zip cell\_db\_latency.txt  cell\_db\_mbs.txt  sed -n -i '2,$ p' cell\_db\_mbs.txt  sed -i 's/MB\/sec/ /g' cell\_db\_mbs.txt  sed -i 's/: /\t/g' cell\_db\_mbs.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_db\_mbs.txt  zip cell\_db\_mbs.zip cell\_db\_mbs.txt  cell\_flash\_destage.txt  sed -n -i '2,$ p' cell\_flash\_destage.txt  sed -i 's/IO\/sec/ /g' cell\_flash\_destage.txt  sed -i 's/: /\t/g' cell\_flash\_destage.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_flash\_destage.txt  zip cell\_flash\_destage.zip cell\_flash\_destage.txt  cell\_flash\_space.txt  sed -n -i '2,$ p' cell\_flash\_space.txt  sed -i 's/MB/ /g' cell\_flash\_space.txt  sed -i 's/: /\t/g' cell\_flash\_space.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_flash\_space.txt  zip cell\_flash\_space.zip cell\_flash\_space.txt  cell\_flashlog\_outliers.txt  sed -n -i '2,$ p' cell\_flashlog\_outliers.txt  sed -i 's/IO requests/ /g' cell\_flashlog\_outliers.txt  sed -i 's/: /\t/g' cell\_flashlog\_outliers.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_flashlog\_outliers.txt  zip cell\_flashlog\_outliers.zip cell\_flashlog\_outliers.txt  cell\_flashlog\_skip.txt  sed -n -i '2,$ p' cell\_flashlog\_skip.txt  sed -i 's/%/ /g' cell\_flashlog\_skip.txt  sed -i 's/IO requests/ /g' cell\_flashlog\_skip.txt  sed -i 's/: /\t/g' cell\_flashlog\_skip.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_flashlog\_skip.txt  zip cell\_flashlog\_skip.zip cell\_flashlog\_skip.txt  cell\_iops.txt  sed -n -i '2,$ p' cell\_iops.txt  sed -i 's/IO\/sec/ /g' cell\_iops.txt  sed -i 's/: /\t/g' cell\_iops.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_iops.txt  zip cell\_iops.zip cell\_iops.txt  cell\_latency.txt  sed -n -i '2,$ p' cell\_latency.txt  sed -i 's/us\/request/ /g' cell\_latency.txt  sed -i 's/: /\t/g' cell\_latency.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_latency.txt  zip cell\_latency.zip cell\_latency.txt  cell\_mbs.txt  sed -n -i '2,$ p' cell\_mbs.txt  sed -i 's/MB\/sec/ /g' cell\_mbs.txt  sed -i 's/: /\t/g' cell\_mbs.txt  sed -i '1 i\cellname\tmetric\tdisk\tvalue\ttime' cell\_mbs.txt  zip cell\_mbs.zip cell\_mbs.txt |

Execute the process\_files.sh

|  |
| --- |
| sh process\_files.sh |

## Email the 14 zip files