Dedup Analysis

# DEDUP.R

DDB\_Analysis()

It is the fundamental function for analysing the ddb performance over time.

DEDUPREAD()

It is the fundamental reader for the DDB csv file.

DEDUPERF()

Similar to the DDB\_Analysis() function but more orientated to the long term analysis.

On the file there are many test and experiment on first analysis on deduplication analysis.

DedupOvertime.R

HeatMap Analysis

# heatMAp\_SingleFilejobs()

Function to analyse a single matrix jobs.

heatMAp\_collect()

Function to aggregate the previous matrix by H==21 for each Storage Policy.

heatMAp\_jobs()

Similar to headMap\_SingleFielJobs() but without the graphics.

It is used to create matrix for each Storage Policy for heatMap\_Collect()

AUXCOPY analysis

AUX\_Logs.R

AUX\_Read\_Logs()

It reads the output from the readAUXLogs.py and summarise and graph them.

readAUXLogs.py

Read the auxcopy logs from the Commcell.

Identify two entry patterns:

The first:

Identify:

[Time][JobId][Storage Policy]

Looking for the pattern:

AuxCopyReserve::getJobParameters Copy properties: Policy

It gives the storage policy name and basic configurations:

2920 1c34 05/21 11:03:32 5288412 AuxCopyReserve::getJobParameters Copy properties: Policy [z4WeekOffSite (DiskE-MA8) Inactive], Copy [primary][200], Source copy [198], StartTime [-1363737600], CopyFlags [6], DedupeFlags [0], DeferDays [0], ArchCheckValidMonths [0], MaxMultiplex [15].

The second logs the progress:

Identify:

[jobid][source streams...][target streams][size] [time ]

Looking for the pattern:

AuxCopyManager::updateProgressToJM <Copy/Stream>

8628 2ff0 03/13 15:45:19 6568991 AuxCopyManager::updateProgressToJM <Copy/Stream> Source <226/26> Target <228/2>: Application Size, Stream Throughput parameters: [2198773421] bytes read in [16814] seconds