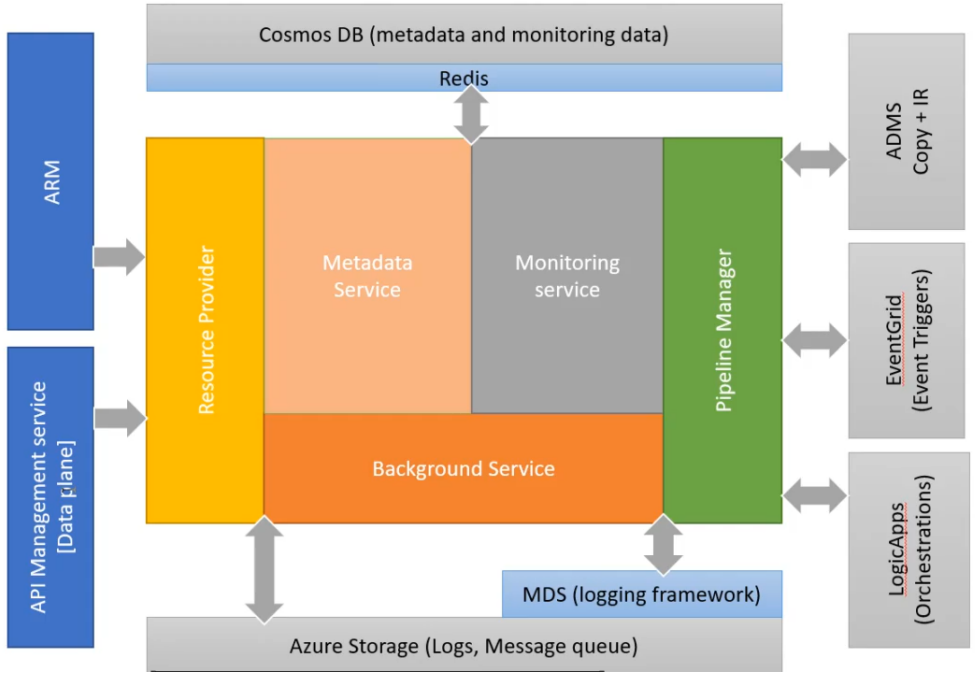
ADMS Kusto Queries and Performance Kusto Functions

Last updated by | Veena Pachauri | Mar 8, 2023 at 11:57 PM PST

ADMS Kusto Queries and Performance Kusto Functions

Thursday, November 28, 2019
10:23 AM



- ARM: HttpIncomingRequests, HttpOutgoingRequests, ARMPR_V2
- ADF RP, MS, PM, MON: ApiOperationEvent
- ADF RP, MS, PM, MON: AdfTraceEvent
- ADMS: ExecutionApiCall
- ADMS: CustomLogEvent

Normal Kusto query

```
ExecutionApiCall
| where PipelineJobId contains "edf8c715-7872-4909-bc72-26541ccbcc3f"
```

ExecutionApiCall can be used to check all the activity based on the Pipeline job id.

After that, you can get the activity ID with CustomLogEvent to get more detail about the activtity status.

Below table is used to check reported event log on the IR Server which included the callstack to help us investigate further.

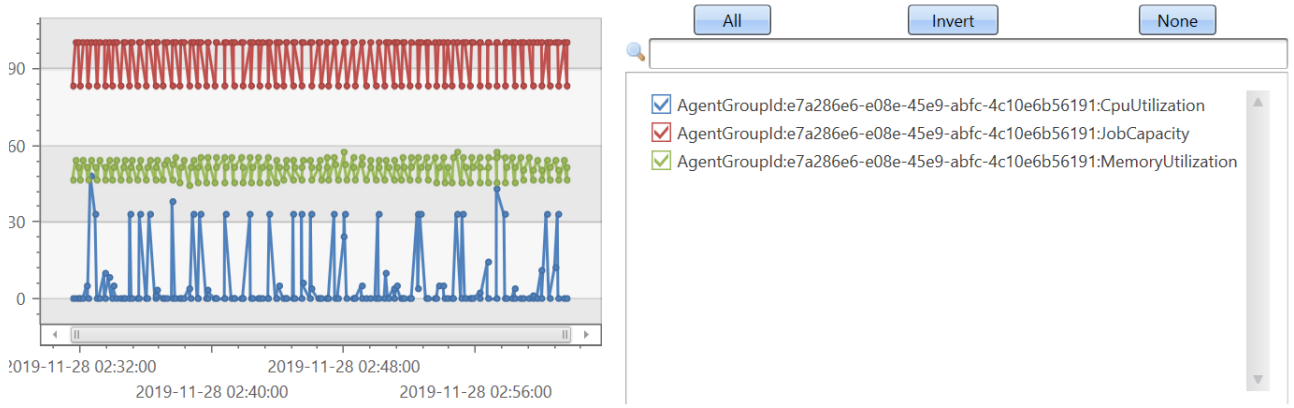
```
TraceGatewayLocalEventLog | where UserReportId contains "dcb92348-d948-4882-b534-90a7e01d98c5"
```

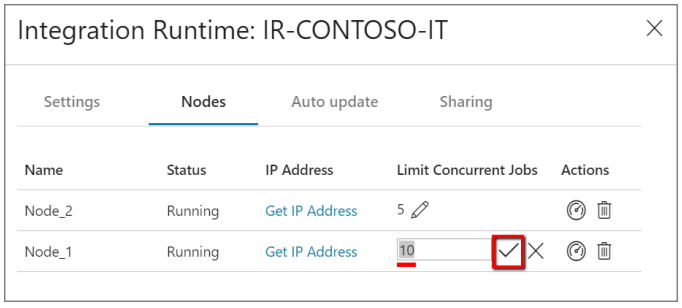
Below Function is used to check perf related issue.

```
202da45b-84d2-4827-8cd0-a8a18cb566d0 is the activity ID.

DiagnosticsSHIRPerfCountersById(@'202da45b-84d2-4827-8cd0-a8a18cb566d0')

DiagnosticsSHIRPerfCountersById kusto query is used to query the IR capacity, if the capacity is quite high than expected such as selfHost-IR, we need to scale up the self-IR node. If jobcapacity is very l
is good, scale up by increasing the number of concurrent jobs that a node can run. You might also want to scale up when activities time out because the self-hosted IR is overloaded. As shown in the follow
increase the maximum capacity for a node:
```





From <https://icm.ad.msft.net/imp/v3/incidents/details/160817085/home>

https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/data-factory/create-self-hosted-integration-runtime.md

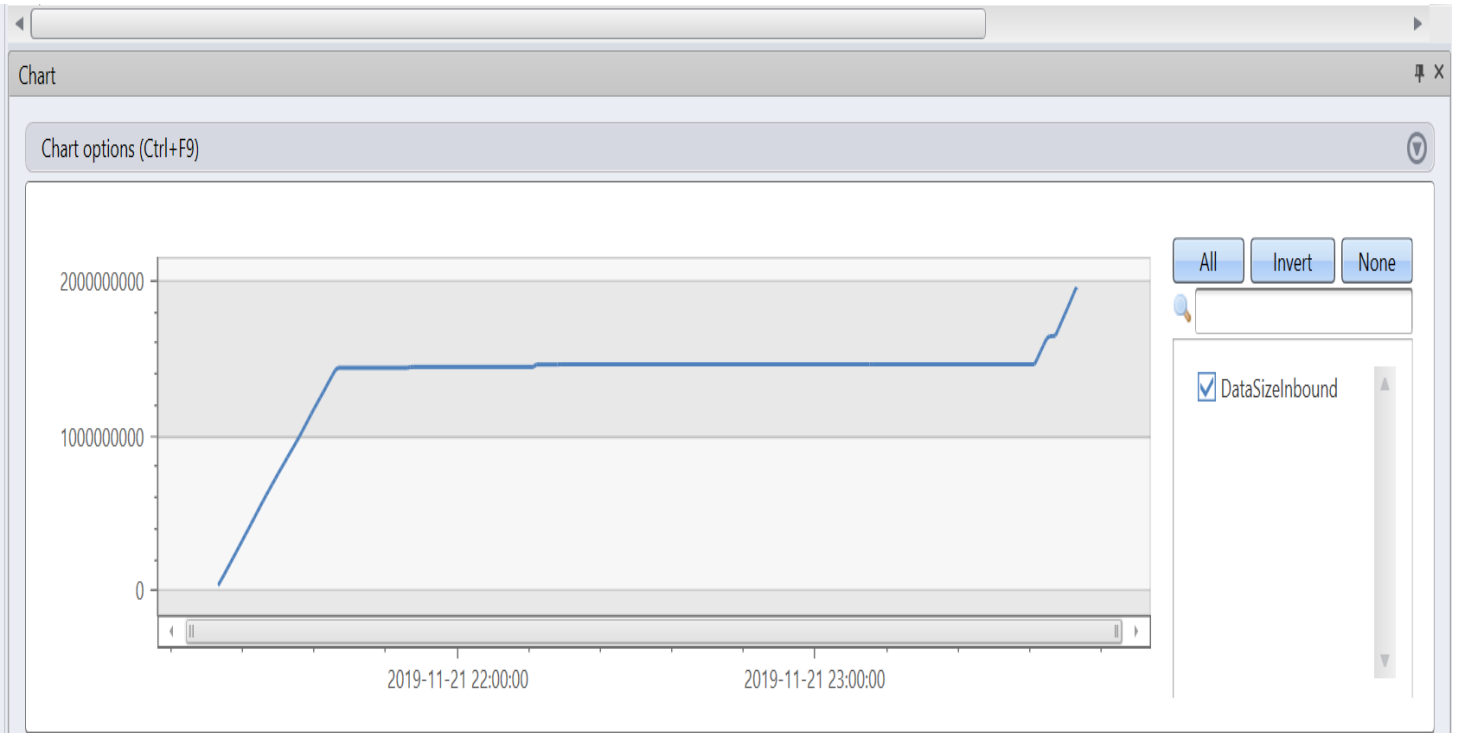
```
ShowCopyTimeLineByActivityId
```

ShowCopyTimeLineByActivityId is used to check the Job status for each stage.

TIMESTAMP	JobCreated	JobPickedUp	CopyInQueue	CopyStarted	CopyEnded
2019-11-21 21:19:32.7457037			1		
2019-11-21 21:19:33.0575854	1				
2019-11-21 21:19:37.6842752		1			
2019-11-21 21:19:37.7390000				1	
2019-11-21 23:44:19.7390000					1
2019-11-21 23:44:20.1575223	1				
2019-11-21 23:44:26.3520084		1			
2019-11-21 23:46:42.1699716	1				
2019-11-21 23:46:43.2623130		1			

```
DiagnosticsCopyThroughputById(@'202da45b-xxxx-xxxx-xxxx-xxxxxxxx')
```

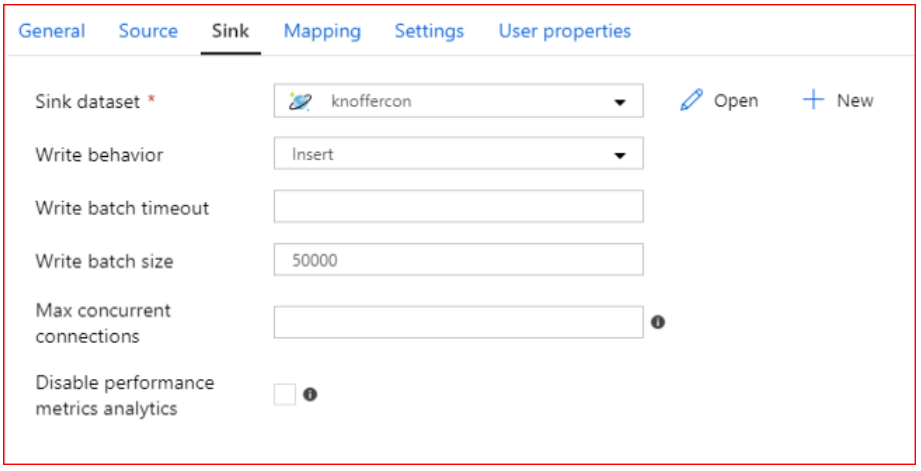
DiagnosticsCopyThroughputById is used to get the copy status throughout, you can see the line without any change which means something stuck there from source as Inbound status. Then we can get dump and driver log to investigate further.



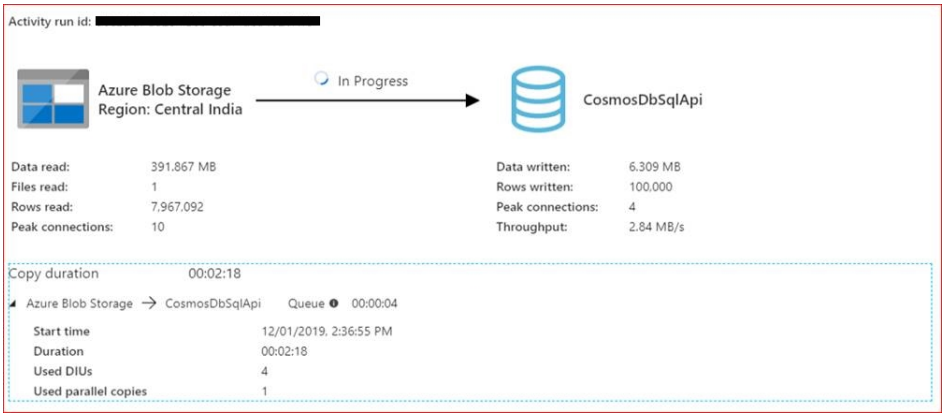
As you can see below as example that the throughout is bigger at beginning since the inbound growing much within minutes with enough buffer, that's why you can see throughout is bigger at beginning, after a while, due to outbounding growing little, so caused the inbouding reached to stable as well not growing too much after buffer full.

The throughput cx see on portal = DataSizeInbound / duration. That's why the value keeps going down.

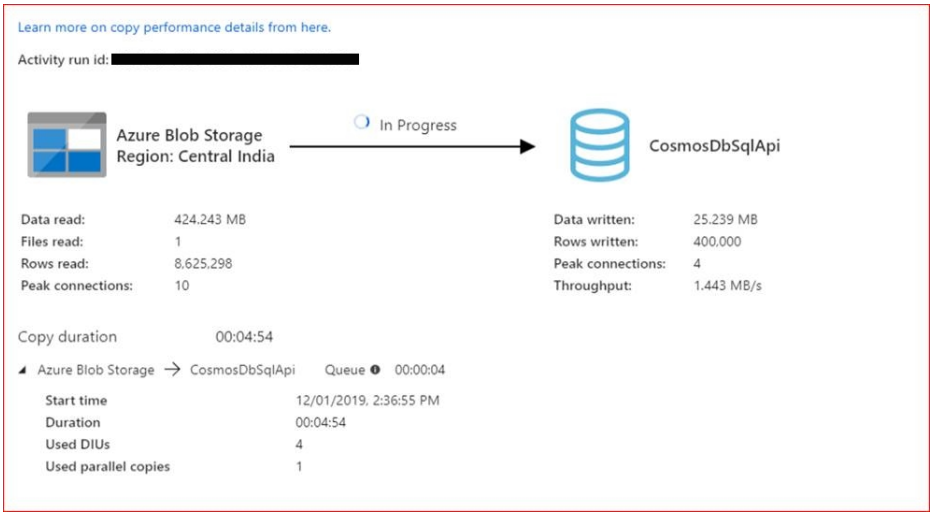
Setting of batch size as 50000



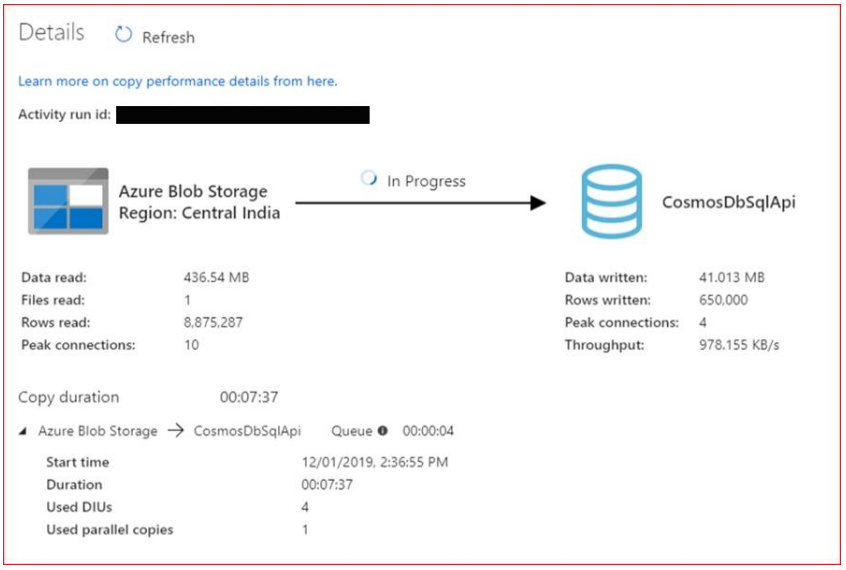
Copy started with 3 MB/s



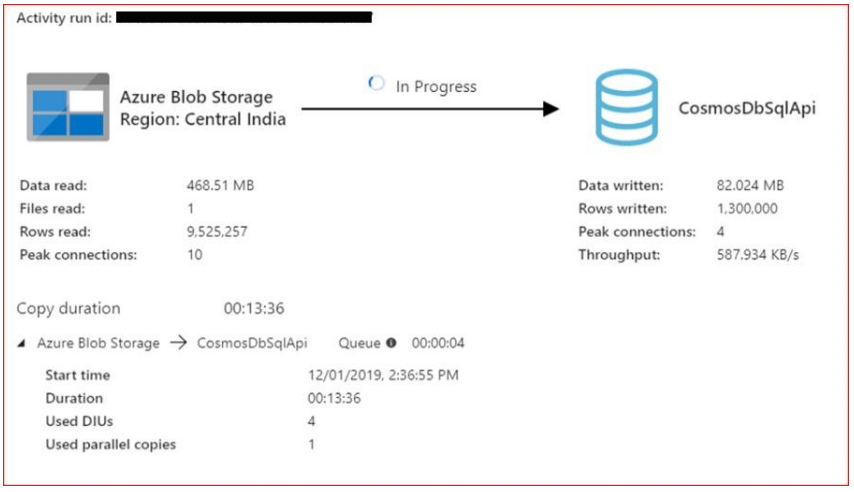
Gradually reduced to 1.5 MB/s

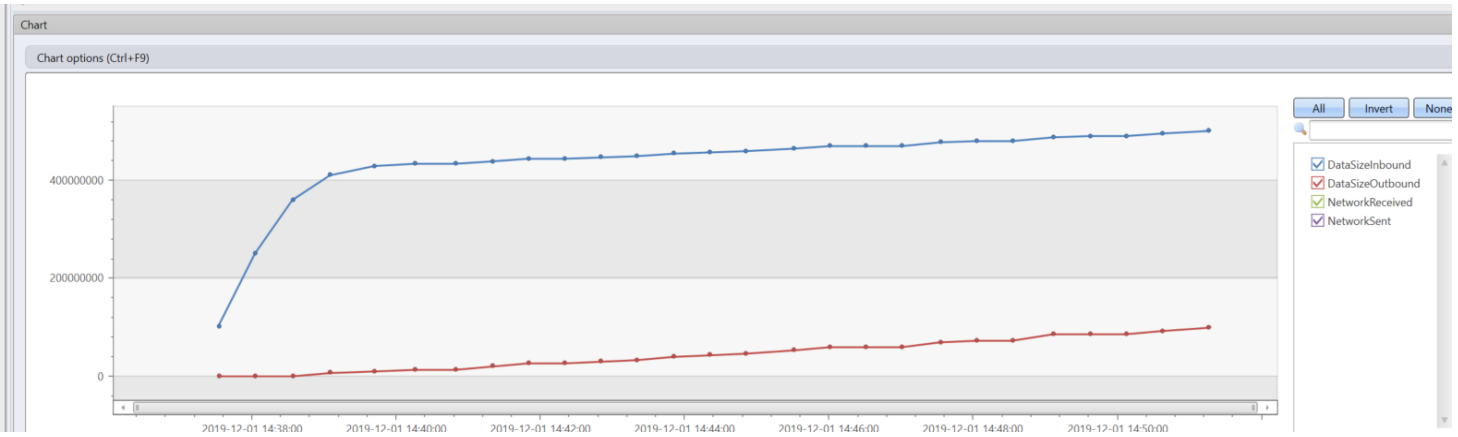


Then to 1 MB/s



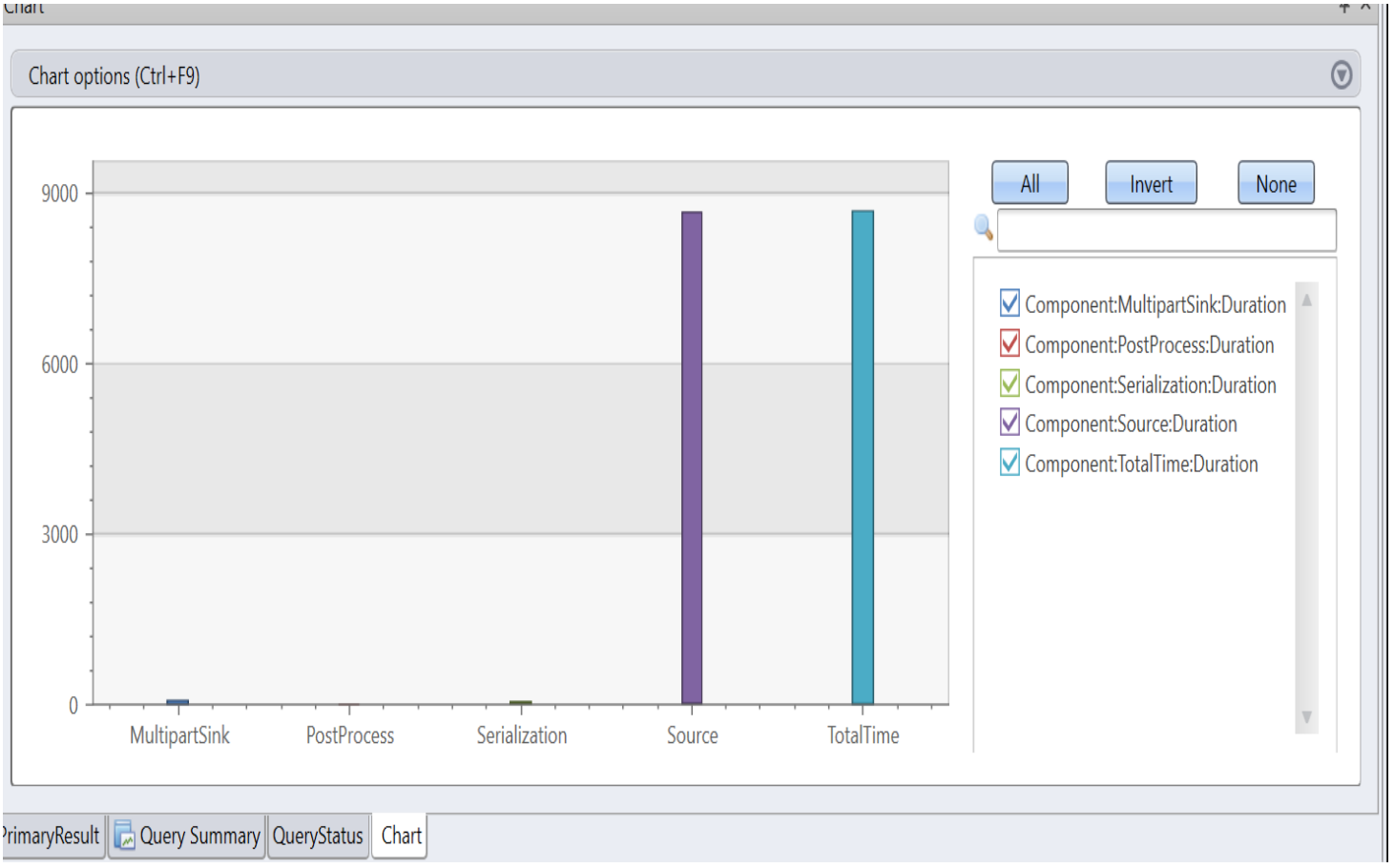
Decreasing less to 500 KB/s






```
ShowCopyPerfBottleneckByActivityId(@'202da45b-xxx-xxxx-xxxx-xxxxxxxx')
```

ShowCopyPerfBottleneckByActivityId is used to check where the bottleneck is, from below chart, you can see it was stuck from Source side, so you need to figure out any issue between IR and Source



Please also refer to the following TSG to further investigate perf issue during copy activity.

[Diagnostic] [V2] Identify performance bottleneck of Copy Activity run

Note: We have merged the kusto query `ShowCopyTimeLineByActivityId`, DiagnosticsSHIRPerfCountersById,DiagnosticsCopyThroughputById and ShowCopyPerfBottleneckByActivityId to ASC, it will support the Chart like below:

Performance: Copy Timeline

Kusto Query

Show checkpoints during copy activity (eg: queued, job picked up, job finished)

Drag a column header and drop it here to group by that column

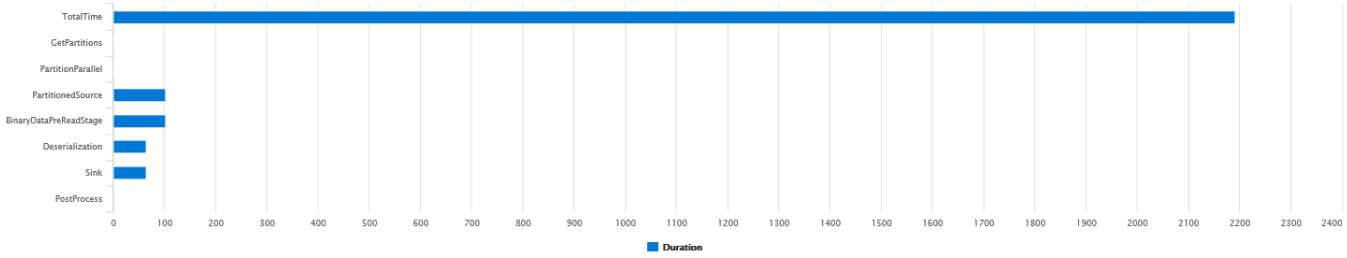
TIMESTAMP	JobCreated	JobPickedUp	CopyInQueue	CopyStarted	CopyEnded
2019-12-02 23:12:45				1	
2019-12-02 23:12:45			1		
2019-12-02 23:12:46	1				
2019-12-02 23:12:46		1			
2019-12-02 23:49:15					1

1 20 items per page 1 - 5 of 5 items

Performance: Copy Bottleneck

Kusto Query

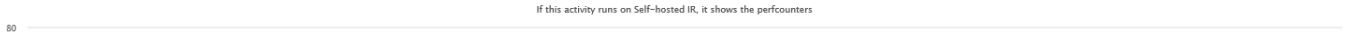
Show the duration of each part during the copy (Missing data in this query was filled with the previous known value)



Performance: SH IR Capacity

Kusto Query

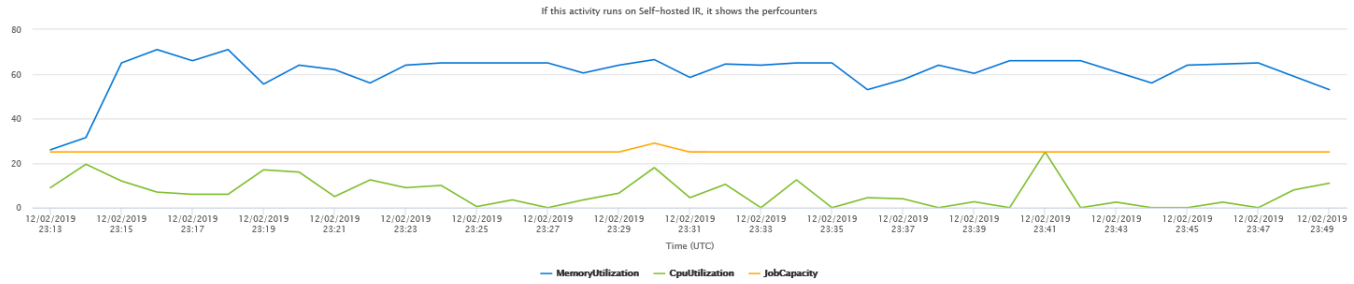
If this activity runs on Self-hosted IR, it shows the perfcounters



Performance: SH IR Capacity

Kusto Query

If this activity runs on Self-hosted IR, it shows the perfcounters



Performance: Copy Throughput

Kusto Query

Show copy throughput during the copy (Missing data in this query was filled with the previous known value)