Parameter Tuning | Hive - For Tez Engine

Last updated by | Prashant12 Singh < Prashant12.Singh@ril.com > | Thu, 20 Jun 2024 06:00:56 GMT

Author **@Bharath Ellanti**

Working on tuning Parameter for CM WO KPI.

1. The Constraint here is that, this is an hourly job which needs to be completed within 25 mins of time.
2. Initially this job is requesting around 43TB memory and using only 22TB. Almost 50% of memory is being unused.
3. This KPI has around 400GB of data to be processed every time the job runs.

Inital Parameters:

```
SET hive.exec.reducers.max=50;
SET hive.exec.reducers.bytes.per.reducer=134217728; -- 128 mb;
SET hive.tez.container.size=12288;
SET tez.runtime.io.sort.mb=3278;
SET tez.task.resource.memory.mb=6144;
SET tez.am.resource.memory.mb=12288;
SET tez.grouping.min-size=8589934592;
SET tez.grouping.max-size=8589934592;
--Common settings
SET tez.am.container.reuse.enabled=true;
SET hive.auto.convert.join=true;
SET hive.exec.parallel=true;
SET hive.exec.parallel.thread.number=12;
SET hive.vectorized.execution.reduce.enabled=true;
SET hive.support.quoted.identifiers=none;
--Avoids small files
set mapred.reduce.tasks = 50;
SET hive.merge.tezfiles=true;
SET hive.merge.mapfiles=true;
SET hive.merge.mapredfiles=true;
SET hive.merge.size.per.task=128000000;
SET hive.merge.smallfiles.avgsize=128000000;
```

The job with above parameters is requesting 43TB of memory and job is completing in around 15 mins every time.

To optimise this we have reduced the parameters.

```
SET hive.exec.reducers.max=25;
SET hive.exec.reducers.bytes.per.reducer=134217728; -- 128 mb;
SET hive.tez.container.size=4096;
SET tez.runtime.io.sort.mb=1638;
SET tez.task.resource.memory.mb=2048;
SET tez.am.resource.memory.mb=4096;
SET tez.grouping.min-size=1073741824;
SET tez.grouping.max-size=1073741824;
--Common settings
SET tez.am.container.reuse.enabled=true;
SET hive.auto.convert.join=true;
SET hive.exec.parallel=true;
SET hive.exec.parallel.thread.number=12;
SET hive.vectorized.execution.reduce.enabled=true;
SET hive.support.quoted.identifiers=none;
--Avoids small files
set mapred.reduce.tasks = 100;
SET hive.merge.tezfiles=true;
SET hive.merge.mapfiles=true;
SET hive.merge.mapredfiles=true;
SET hive.merge.size.per.task=128000000;
SET hive.merge.smallfiles.avgsize=128000000;

SET hive.optimize.skewjoin.compiletime=true;
SET hive.optimize.skewjoin=true;
-- adjust these according to your dataset
SET hive.skewjoin.key=1000000;
SET hive.skewjoin.mapjoin.map.tasks=60;
SET hive.skewjoin.mapjoin.min.split=134217728;
```

Container size has been reduced to 4GB from 12GB, Number of reducers are decreased to half, Each task memory has been reduced to 2GB from 6GB and group size has been reduced to 1GB from 8GB.

The Job with above parameters is requesting around 10.7 TB out of which 6.4TB is been used. Even here 36% of resource wastage is present. Also 10.7TB is too much memory for processing 400GB data. Even with this the job is completing around 17 mins.

Grouping decides the number of mappers. If we put max grouping a large value, then the input data is split into large chunks and number of mappers to process them decreases. Decreasing this max grouping value increases number of mappers. As the number of mappers increases the time to process same data decreases. Processing speed increases. Also, after some time if number of mappers increases the resources requested also increases. Needs to be careful while increasing number of mappers.

```
SET hive.optimize.skewjoin.compiletime=true;
SET tez.grouping.max-size=473741824;
SET hive.tez.container.size=3072;
SET tez.runtime.io.sort.mb=1024;
SET mapred.reduce.tasks=15;
SET tez.task.resource.memory.mb=2048;
SET hive.exec.reducers.max=25;
```

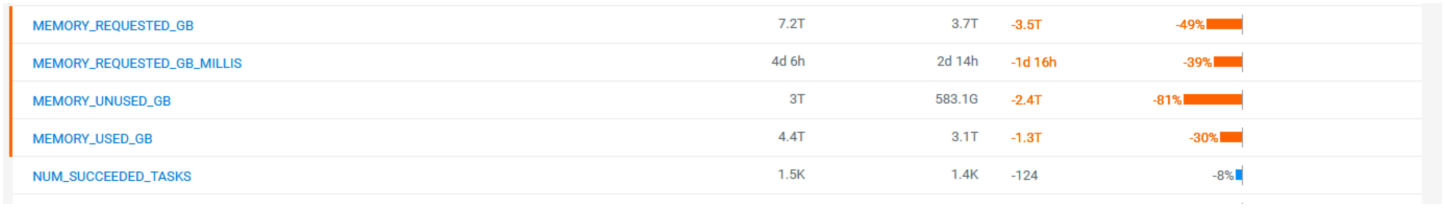
In the above parameters the container size is reduced to 3GB and max split size is reduced to 400MB which increased the number of mappers, hence job is completing fast even with less container size provided than before. This took around 25 mins to complete the job and requested around 6TB and used 4.4 TB memory. Here I tried to reduce the container size 2GB which is giving vertex failure error every time in my case. Reducing container size here will decrease the amount of memory requested.

I also tried to reduce the number of mapred.reduce.tasks to 10 which decreased the memory requested to 3.5TB, but the job is taking more time to complete, around 35mins. Increasing this number is reducing the time taken to process but increasing the memory requested. This parameter is crucial in determining the time taken and memory requested by a job.

Final Parameters

```
SET tez.grouping.min-size=268435456;
SET tez.grouping.max-size=536870912;
SET hive.tez.container.size=2560;
SET tez.runtime.io.sort.mb=512;
SET mapred.reduce.tasks=15;
SET hive.execution.engine=tez;
```

This took around 25mins to complete the Job and requesting around 3.7TB of memory and using 3.1TB of memory. Wastage has been reduced to 16% of requested. In the above the container size is the minimum that I require for my job. If I go less than that it is giving vertex failure error. Number mapred.reduce.tasks can't be increased, if increased requesting for resource, if decreased then taking more time to complete the job.



MEMORY_REQUESTED_GB	7.2T	3.7T	-3.5T	-49%
MEMORY_REQUESTED_GB_MILLIS	4d 6h	2d 14h	-1d 16h	-39%
MEMORY_UNUSED_GB	3T	583.1G	-2.4T	-81%
MEMORY_USED_GB	4.4T	3.1T	-1.3T	-30%
NUM_SUCCEEDED_TASKS	1.5K	1.4K	-124	-8%

We have different execution engines available in Yarn Queue. Mainly Apache TEZ and Apache Spark. Default in our Hadoop Ecosystem, Apache Tez is configured.

Tez: Generally provides good performance for most Hive queries and is more resource-efficient for traditional ETL and batch processing jobs.
Spark: May require more memory and resources, but delivers superior performance for in-memory processing, complex analytics, and interactive queries.

```
SET hive.execution.engine=tez;
```

```
SET hive.execution.engine=spark;
```

Use Apache Tez for traditional Hive ETL and batch processing workloads that benefit from optimized DAG execution and lower resource overhead.
Use Apache Spark for interactive queries, complex analytics, iterative algorithms, and scenarios where in-memory processing provides a significant performance boost.

In general I observed that if there is more resource wastage increasing number of mappers will help to decrease the resource wastage. To increase number mappers we need to decrease the Grouping.min-size and grouping.max-size. As number of mappers are increased the parallel processing also increases which increases the processing speed and correct utilisation of resouces.

Mapred.reduce.tasks plays a vital role in deciding resource allocation.

Setting too few reduce tasks may lead to underutilization of cluster resources. This can result in longer job completion times because fewer reducers are handling larger amounts of data.

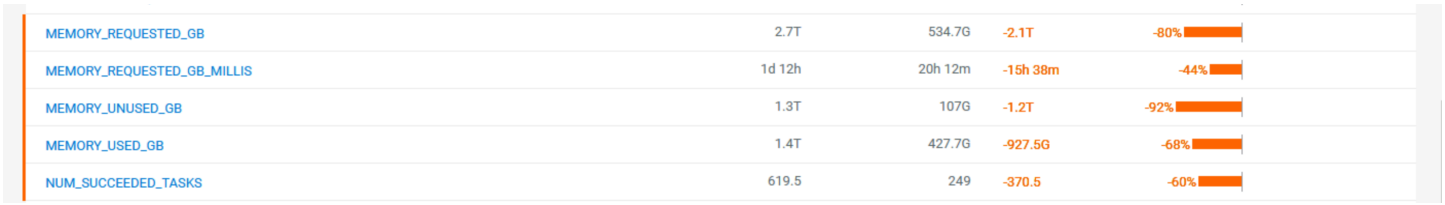
Setting too many reduce tasks can lead to overhead in managing too many tasks, each processing a smaller amount of data. This can cause unnecessary scheduling overhead and reduce efficiency.

By tuning the number of reduce tasks, you can ensure that data processing is adequately parallelized. This helps to fully utilize the available cluster resources, leading to faster job completion.

There is one more ingestion job which has around 160GB of data to be optimised. Even this job has a time constraint to complete in less 30 mins.

Started with following Parameters

```
set tez.grouping.min-size=74217728;
set tez.grouping.max-size=134217728;
SET hive.tez.container.size=2260;
SET tez.runtime.io.sort.mb=512;
SET mapred.reduce.tasks=15;
SET hive.exec.parallel=true;
SET hive.execution.engine=tez;
```



MEMORY_REQUESTED_GB	2.7T	534.7G	-2.1T	-80%
MEMORY_REQUESTED_GB_MILLIS	1d 12h	20h 12m	-15h 38m	-44%
MEMORY_UNUSED_GB	1.3T	107G	-1.2T	-92%
MEMORY_USED_GB	1.4T	427.7G	-927.5G	-68%
NUM_SUCCEEDED_TASKS	619.5	249	-370.5	-60%

It has reduced the memory requested to 500GB but Unused memory is still high at around 20 percentage. Job took around 37 mins to complete.

To decrease the time, I have increased the container size to 4GB and number of tasks.

```
set tez.grouping.min-size=1073741824;
set tez.grouping.max-size=1073741824;
SET hive.tez.container.size=4096;
SET tez.runtime.io.sort.mb=1710;
SET mapred.reduce.tasks=50;
SET hive.exec.parallel=true;
SET hive.execution.engine=tez;
```

MEMORY_REQUESTED_GB	1.9T	5.9T	+4T	210%
MEMORY_REQUESTED_GB_MILLIS	5d 14h	6d 6h	+15h 56m	12%
MEMORY_UNUSED_GB	333G	376.1G	+43.1G	13%
MEMORY_USED_GB	1.5T	5.6T	+4T	262%
NUM_SUCCEEDED_TASKS	806.5	3.2K	+2.4K	303%

This has reduced the unused memory to less than 10 percentage and the job completes in around 20 mins. The requested memory is too high to process 160GB data.

Hence reduced the size of container to 1.5GB and increased the mapred tasks to 100 for fast processing.

```
set tez.grouping.min-size=33554432;
SET tez.grouping.max-size=67108864;
SET hive.tez.container.size=1741;
SET tez.runtime.io.sort.mb=512;
SET mapred.reduce.tasks=100;
SET hive.execution.engine=tez;
SET hive.exec.parallel=true;
```

Data read (Local)	811.8 GiB	835.9 GiB	+24.1 GiB	3%
Data read (S3)	0 B	0 B	0 B	
Data written (ADL)	0 B	0 B	0 B	
Data written (HDFS)	169.5 GiB	169.3 GiB	-151.8 MiB	
Data written (Local)	1.2 TiB	1.2 TiB	+20.9 GiB	2%
Data written (S3)	0 B	0 B	0 B	
Failed Task attempts	0	0	0	
Killed Task attempts	0	0	0	
MEMORY_REQUESTED_GB	1.6T	1.5T	-67.5G	-4%
MEMORY_REQUESTED_GB_MILLIS	4d 6h	4d 15h	+8h 49m	9%
MEMORY_UNUSED_GB	319.1G	89G	-230.1G	-72%
MEMORY_USED_GB	1.5T	1.4T	-108G	-7%
NUM_SUCCEEDED_TASKS	726	824	+98	13%

This has reduced the requested memory, unused memory and job completes in around 28 mins.

Decreasing container size will help always to reduce the requested memory. But we also need to make sure we don't get **Vertex Failure because of less container size**.

Increasing number of mappers will reduce resource wastage and increase processing speed.

Increasing number of reducers will increase processing speed but also increase resources requested.

```
set hive.vectorized.execution.enabled=true;
set hive.vectorized.execution.reduce.enabled=true;
set hive.stats.autogather=true;
set hive.exec.parallel=true;
set hive.auto.convert.join=true;
set hive.cbo.enable=true;
set hive.merge.mapfiles=true;
set tez.task.resource.cpu.vcores=1;
set hive.compute.query.using.stats=true;
set hive.stats.fetch.column.stats=true;
set hive.stats.fetch.partition.stats=true;
set hive.exec.compress.output=true;
set hive.exec.compress.intermediate=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
SET hive.exec.reducers.max=40;
```

The above parameters will also help in optimised and fast query processing. I haven't used these parameters as tweaking these didn't make any difference in my stats

---

Author **@Prashant12 Singh @Roshan Patidar**
Parameter Tuning For ISOC Usecase tas summary.
Job_tuning_ISOC_13Jun24_19Jun24.docx