Vertica Knowledge Base Article

Tuple Mover Best Practices

Document Release Date: 11/15/2018

MICRO FOCUS | VERTICA

Legal Notices

Warranty

The only warranties for Micro Focus International plc products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from Micro Focus required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

Contents

Tuple Mover Best Practices

Click here for a PDF version of this document.

Applies to Vertica 7.2.x and earlier

Tuple Mover Overview

The Vertica analytics platform provides storage options to trickle load small data files in memory, known as WOS, or to bulk load large data files directly into a file system, known as ROS. Data that is loaded into the WOS is stored as unsorted data, whereas data that is loaded into ROS is stored as sorted, encoded, and compressed data, based on projection design.

The Tuple Mover is a Vertica service that runs in the background and performs two operations:

- **Moveout**: The Tuple Mover moveout operation periodically moves data from a WOS container into a new ROS container, preventing WOS from filling up and spilling to ROS. Moveout runs on a single projection at a time, on a specific set of WOS containers. When the moveout operation picks projections to move into ROS, it combines projection data loaded from all previously committed transactions and writes them into a single ROS container.
- **Mergeout**: The Tuple Mover mergeout operation consolidates ROS containers and purges deleted records.

In most use cases, the Tuple Mover requires little or no configuration beyond the default parameters. However, some workloads may require some tuning of configuration parameters. This document:

- Answers frequently asked questions
- Provides troubleshooting tips
- Describes best practices with respect to WOS, data loading, and schema design

Following the information given in this document can help improve the Tuple Mover operations and system efficiency.

Tuple Mover Moveout Operation

Detect WOS Spillover

WOS memory is controlled by a built-in resource pool named WOSDATA, whose default maximum memory size is 2GB per node. If you load data into the WOS faster than the Tuple Mover can move the data out, the data can spill into ROS until space in the WOS becomes available. Data loss does not occur with a spillover, but a spillover can create ROS containers much faster than anticipated and slows the moveout operation.

You can detect whether your database is running into a WOS spillover by running the following query:

```
=>SELECT node_name, count(*) from dc_execution_engine_events
 WHERE event_type = 'WOS_SPILL'
 group by node_name;
```

Moveout Best Practices

Follow these best practices to verify the Tuple Mover moveout operation runs smoothly and correctly:

Use COPY DIRECT for Loading Large Data Files

If you load large data files (more than 100 MB per node), convert the large COPY statement to a COPY DIRECT statement. This allows you to bypass loading to the WOS and instead, load files directly to ROS.

Configuration Parameter: `MoveOutInterval`

Set this parameter to a value that is less than the time it takes to fill half of the WOS. This parameter determines the time in seconds that the Tuple Mover moveout operation sleeps when it detects there is nothing to move out from the WOS. The default value of this parameter is 300 seconds and decreasing the value can move data more frequently from the WOS into ROS.

Uncommitted Data in WOS

Do not leave uncommitted data in the WOS for longer than necessary because the Tuple Mover moves out only committed data. If the WOS is filled with data from uncommitted transactions, moveout cannot move data out. This can lead to a WOS spillover into ROS.

Do not use the WOS for Large Temporary Tables

Do not use the WOS to load temporary tables with a large data set (more than 50 MB per node). The moveout operation does not move out temporary table data and the data is dropped when the transaction or session ends.

`maxMemorySize` of WOSDATA Resource Pool

If you concurrently trickle load into more than 10 tables, and experience a WOS spillover, one or more of the previous best practices should help resolve your issue. You can increase the size of WOS memory by increasing the `maxMemorySize` of the WOS resource pool. **Do not** increase the size of WOS memory unless you aggressively load data into many tables and the data in WOS per projection is not more than a few hundred megabytes. If you have more than a few hundred MB in the WOS per projection, you could encounter inconsistent query performance.

Configuration Parameter: `MoveOutSizePct`

Set this parameter to set space-utilization based eligibility criteria to trigger a moveout. Setting this parameter is helpful if you trickle load data into a few tables. In these cases, the moveout

operation may aggressively move out data from the WOS, which creates frequent, small ROS containers. For example, if you set this value to 40, then the moveout operation does not move out projections until the WOS is 40% full and can help batch data before moving the data out.

Configuration Parameter: `MoveOutMaxAgeTime`

Set this parameter to set the amount of time data resides in WOS before the Tuple Mover triggers a moveout based on age (in seconds). The default value of this parameter is 1800. If you change the value of the `MoveOutSizePct` parameter, you can change the value of this parameter to 600.

Moveout: Frequently Asked Questions

Can I increase the number of moveout threads?

No. The default number of moveout threads is 1 and you cannot change this value.

How can I change the `MoveOutInterval` parameter and when does it go into effect?

You can change this configuration parameter by running the `set_config_parameter` command in vsql. Changing this parameter affects future moveout sleep states. The following example changes the value of this parameter to 120 seconds:

```
=>SELECT set_config_parameter('MoveOutInterval',120);
```

How can I check if moveout is running on a node?

You can check the progress of your Tuple Mover operations by running the following SQL statement:

```
=>SELECT * from tuple_mover_operations where is_executing;
```

Which locks does moveout take?

When moving data out for projections, moveout takes a U lock on a table. When managing delete marks or replay deletes, moveout takes a T lock on a table.

What can cause moveout to fail?

- The moveout operation fails or is blocked if it cannot obtain a T lock. This can happen because of a long running transaction holding an X lock on the same table.
- The moveout operation fails or is blocked if it is trying to move out deleted vectors that point to ROS containers participating in a long-running mergeout.
- The moveout operation also fails if you load more than 1024 partitions on a table into WOS.

Tuple Mover Mergeout Operation

The Tuple Mover mergeout operation is a Vertica service that runs in the background and consolidates ROS containers. The maximum number of ROS containers per projection per node is 1024. If you encounter an error during a data load that states “TOO MANY ROS CONTAINERS”, you have reached the maximum number of ROS containers per projection per

node. Mergeout consolidates ROS containers into fewer containers, verifying that under normal usage the database does not hit this limit.

Best Practices for Mergeout

Follow these best practices to verify the Tuple Mover mergeout operation runs smoothly and to reduce your ROS container buildup:

Use WOS for Trickle Loads or Batch Load with DIRECT HINT

Trickle loading into WOS helps batch data before moving into disk, which reduces the number of ROS containers created. For larger files, batch load directly into ROS using DIRECT HINT.

MemorySize of TM Resource Pool

If your database has wide tables (more than 100 columns), increase the `MemorySize` of the TM resource pool from the default of 200MB to 6GB and change the `PLANNEDCONCURRENCY` parameter to 3. Reserve at least 2GB per Tuple Mover mergeout thread to speed up operations on wide tables.

Partitions per Table

Create 50 or fewer partitions per table because Vertica does not merge ROS containers across partitions. Thus, tables with hundreds of partitions can hit ROS pushback quickly. You can alter the partition scheme for a table using the `ALTER TABLE ALTER PARTITION` command.

> Important If you have older partitions that you no longer access, you can move them to archived tables using the `MOVE_PARTITION_TO_TABLE` function.

Configuration Parameter: `ActivePartitionCount`

If your tables receive frequent data into the current and the most recent inactive partition, change the `ActivePartitionCount` parameter to 2 from the default of 1. Vertica expects partitions to be time-based, with one active partition receiving data and other inactive partitions rarely or never receiving data. The Tuple Mover merges ROS containers for inactive partitions into a single ROS container.

Limit Projection Sets Anchored on the Same Table

Do not have more than two projection sets anchored on the same table. Having more than two sets of projections per table can lead to wasted system resources.

Projection Sort Order Guidelines

Include fewer than 10 columns in the sort order and avoid having wide VARCHAR columns in the sort order. To control the number of columns in the sort order, replace table projections with new projections that contain fewer columns in the sort order and avoid having wide VARCHAR columns in the sort order. This helps decrease the time it takes for the mergeout operation to run. Long running operations can block mergeout threads, which increases the number of ROS containers.

Optimize Projections for Delete and Replay Delete

Optimize your projections for deletes by using a high-cardinality column as your last column in the sort order. This helps you avoid a long running mergeout due to replay delete. If mergeout is stuck performing a replay delete, cancel the operation using the `close_session (<session_id>)` function. Then run the `make_ahm_now` function to advance the Advanced History Mark (AHM) epoch. After the AHM advances, deletes do not replay and the mergeout operation should run faster. Long running mergeouts performing replay delete can cause ROS container accumulation over time.

Verify that the Reorganize Operation Completed Successfully

You can check the status of the reorganize operation by checking the `PARTITION_STATUS` system table. If a partition expression is altered but the reorganize expression does not start or fails, the ROS containers of the projections anchored on the table that existed before the alteration cannot qualify for mergeout, until the table is reorganized. You can resolve this issue by initiating:

```
=>ALTER TABLE <TABLE_NAME> REORGANIZE;
```

Batch Deletes and Updates Whenever Possible

Batch deletes and updates whenever possible. If you must perform frequent deletes or updates, use WOS to consolidate deletes into fewer delete vectors. This prevents having too many delete vectors per projection, which can lead to ROS container buildup.

Configuration Parameter: `MergeOutInterval`

Adjust this parameter from its default value of 600 to verify that the number of ROS containers created per projection per interval does not exceed the ROS pushback limit.

MaxConcurrency and PlannedConcurrency of TM Resource Pool

If you considered the previously listed best practices but your workload requires an additional mergeout thread, increase the `MaxConcurrency` and `PlannedConcurrency` parameters of the TM resource pool from 3 to 4. The default value is 3: 1 moveout and 2 mergeouts. Do not increase this value to more than 6.

Mergeout Improvement with MergeOutCache

In Vertica 7.1.2-6, a job analysis tool called MergeOutCache was added to improve the efficiency of the Tuple Mover algorithm. With MergeOutCache, the Tuple Mover can more quickly identify mergeout jobs that need to be run. If you have large catalogs with hundreds of tables, consider upgrading to Vertica 7.1.2-6 or the latest Vertica patch.

Mergeout: Frequently Asked Questions

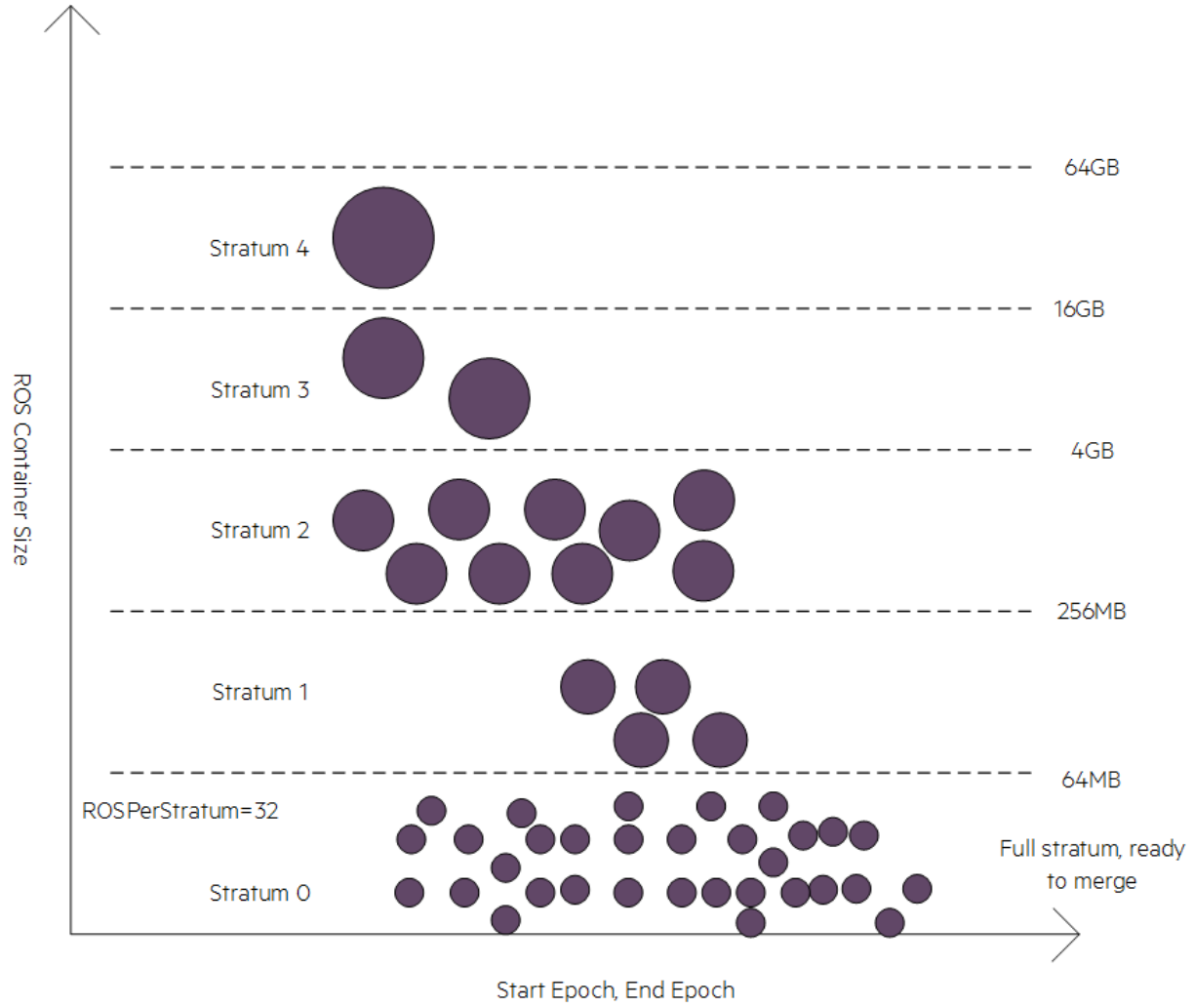
How does the Tuple Mover mergeout STRATA algorithm work?

The Tuple Mover mergeout operation uses a strata-based algorithm. This algorithm is designed to verify that each tuple is subjected to mergeout a small constant number of times despite the load process that is used to load the data. Using this algorithm, the mergeout operation

chooses which ROS containers to merge for tables without partitions and for active partitions in partitioned tables.

Vertica builds strata for each active partition and for projections anchored to non-partitioned tables. The number of stratum, size of each stratum, and maximum number of ROS containers in a stratum is computed based on disk size, memory, and the number of columns in a projection.

The following graphic shows how the algorithm categorizes ROS containers into strata based on their size. Any ROS containers in stratum 0 are of negligible ROS size (1 MB per column or less). The purple dots represent the ROS containers:



Merging small ROS containers before merging larger ones provides a maximum benefit in the mergeout algorithm. The algorithm begins at stratum 0 with 1 byte to the negligible ROS chunk size and moves upward. It checks to see if the number of ROS containers in a stratum has reached a value equal or greater to the maximum ROS containers allowed per stratum. (The

default value is 32). If the algorithm finds that a strata is full, it marks the projections and the stratum as eligible for mergeout.

The mergeout operation combines ROS containers from full strata and produces a new ROS container that is usually assigned to the next stratum. With the exception of stratum 0, the mergeout operation merges only those ROS containers equal to the value of `ROSPerStratum`. For stratum 0, the mergeout operation merges all eligible ROS containers present within the stratum into one ROS container.

By default, the mergeout operation has two threads. Typically, the mergeout of large ROS containers in higher stratum takes longer than the mergeout of ROS containers in lower stratum. Only mergeout thread 0 can work on higher stratum and inactive partitions. This restriction prevents the accumulation of ROS containers in lower stratum, because mergeout thread 0 takes more time to perform mergeouts in higher stratum. Mergeout thread 1 operates only on the lower strata.

> Note If you add more mergeout threads, these additional threads operate only on the lower part of the strata.

How many mergeout threads work on non-active partitions?

Mergeout thread 0 operates only on non-active partitions and merges ROS containers from inactive partitions into a single ROS container. Thread 0 works on non-active partitions only when no other projections qualify for mergeout based on the strata algorithm.

Does the mergeout operation purge deleted data?

The Tuple Mover purges only data that was deleted prior to the AHM in a ROS container that qualified for mergeout based on the strata algorithm. Data that is deleted in inactive partitions can also be purged. However, to qualify for purge, the percentage of deleted data in a ROS container with inactive partitions must exceed the value of the `PurgeMergeoutPercent` parameter. The default value of this parameter is 20.

When and how can I use the following configuration parameters?

Parameter	Function
`ROSPerStratum`	The number of ROS containers in a stratum. When a stratum reaches the default value of 32, the projection qualifies for mergeout. Decreasing this value increases I/O because records participate in more mergeout operations but it decreases the number of ROS containers.
`MaxDVROSPerContainer`	When the number of delete vectors attached to a single ROS container reaches the default value of 10, the Tuple Mover mergeout operation merges delete vectors.

How can I find active partitions for a given projection?

You can find the active partitions for a given projection using the following command:

```
=>SELECT DISTINCT partition_key FROM strata;
 WHERE projection_name <projection_name>
 AND schema_name <schema>
```

How can I find the container count per projection per node?

You can find the container count per projection per node using the following commands:

```
=>SELECT node_name, schema_name, projection_name,
 sum(delete_vector_count) delete_vector_count, count(*) ROS_container_
count, sum(delete_vector_count) + count(*) total_container_count
 FROM storage_containers
 GROUP BY 1,2,3 ORDER BY 6 DESC;
```

Tuple Mover Best Practices