VMware Tanzu Greenplum Command Center v4.14 Documentation

VMware Tanzu Greenplum Command Center 4.14

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

Pivotal Greenplum Command Center 4.14 Documentation

Greenplum® Command Center 4.14 Release Notes

About This Release

This document contains release information about Pivotal Greenplum Command Center 4.14. Greenplum Command Center 4.14 adds new features to Greenplum Command Center and resolves some issues.

Supported Platforms

Greenplum Command Center 4.14 is compatible with the following platforms.

- Pivotal Greenplum Database 5.29.2 - 5.29.5. (Greenplum Database 6 requires Command Center 6.x.) See Pivotal Greenplum Command Center Supported Platforms for the most current Greenplum Command Center and Greenplum Database compatibility information.
- Red Hat Enterprise Linux 6.x[1] and 7.x
- CentOS 6.x[1] and 7.x
- SUSE Enterprise Linux 11 SP4[2]

[1]If you use resource groups and workload management on Red Hat or CentOS 6.x, upgrade your kernel to 2.6.32-696 or higher to benefit from improvements in the Linux cgroups service.

[2]Greenplum Command Center workload management features are not supported on SUSE Enterprise Linux 11.

Enhancements and Changes in Greenplum Command Center 4.14.0

New and Changed Features

- The **realtime Query details** and **Query History details** pages now display:
    - the top 5 slices that consume the most CPU time
    - the top 5 slices that consume the most memory
    - the top 5 slices that use disk I/O

  For more information, see the Query Details and Query History Details documentation pages.
- A user can now exclude "idle in transaction" sessions when adding or editing an idle session

killing rule. For more information, see the Workload Management documentation page.

- The `gpmetrics` schema includes new and changed tables:
    - `gpmetrics.gpcc_resgroup_history`, is a new table that stores the history of the resource consumption of each resource group on each segment
    - `gpmetrics.gpcc_department`, is a new table that stores department information
    - `gpcc_role_department`, is a new table that stores information about roles within department
    - The `gpmetrics.gpcc_queries_history` table includes three new columns: `slices_metrics`, `peak_memory` and `node_sliceid`
    - The `gpmetrics.gpcc_plannode_history` table's `sliceid` column has been deprecated.

  For more information, see the gpmetrics Schema Reference documentation page.
- The `gpmetrics` schema now includes user-defined functions:
    - `gpcc_delete_department` deletes a department name from the `gpperfmon` database.
    - `gpcc_queries_per_hour` returns a variety of details about query activity per hour.
    - `gpcc_queries_per_user` returns, for each user, the number of queries whose runtime is longer than the input interval, per hour, in the specified time range.
    - `gpcc_queries_per_user_max_and_total_spill_size` returns, for each user, the total spill_size and maximum spill_size per query per hour.
    - `gpcc_queries_per_user_max_cpu` returns, for each user, the query with the maximum segment and master cpu usage per hour, along with details about the query.
        - `gpcc_queries_per_user_max_run_time` returns, for each user, the longest running query per hour, along with details about the query.
        - `gpcc_queries_per_user_max_skew` returns, for each user, the query with the maximum amount of processing skew in the system (skew_cpu) per hour, along with details about the query.
        - `gpcc_queries_per_user_rows_out` returns, for each user, the query with the maximum rows_out per hour, along with details about the query.
        - `gpcc_system_per_hour` returns a variety of system information.
        - `gpcc_update_department` renames a department.

    For more information, see the gpmetrics Schema Reference documentation page.
- A user can now monitor resource group system information, including:
    - CPU utilization of selected resource groups
    - CPU utilization of a specific resource group
    - Memory utilization of selected resource groups
    - Memory utilization of a specific resource group
    - Concurrency status of selected resource groups

For more information, see the Managing Resource Groups documentation page.

- Through a new user management interface, a user can now view information about departments and roles, manage and view permissions for roles, export usage information by role, rename a department name using a user-defined function, and delete unwanted departments from the `gpperfmon` database using a user-defined function. For more information, see the Managing Users documentation page.
- Administrators now allow guest users to view the **Query Monitor** without logging in via the **Settings** view, rather than the now obsolete **Permissions** view. For more information, see the Settings documentation page.
- Administrators now manage permissions via the new User Management view rather than the now obsolete **Permissions** view.
- A user can now now determine the TLS cipher suites that Command Center accepts. For more information, see the Command Center Console Parameters documentation page.

Resolved Issues

- [31132] - Command Center was failing to set up an email alert when the SMTP certificate was invalid, rather than issuing a message explaining to users how to proceed without a valid certificate. This has been resolved.
- [31508] - Resolved an issue where Command Center was failing to save queries to the query history when there were CTAS queries inside a function.
- [31531] - The Command Center was using a hard coded initial password for the gpmon role if the password was not created before the installer ran. It now uses a random string generated during installation instead.
- [31781] - Resolved an issue where the `accuracy` and `last_accuracy_ts` fields in the `gpmetrics.gpcc_table_info_history` table were not being correctly populated when the `planrows` field in the `gpmetrics.gpcc_plannode_history` table had a very large value.
- [31782] - Resolved a memory leak in the Command Center session monitor.
- [31890] - Resolved an issue where, when users executed a query with an extremely long query text it could cause the query to fail, due to an invalid `memory alloc` request.
- [31867] - Resolved an issue where the `cpu_segment_percent` column of the `gpmetrics.gpcc_queries_now` table was not matching the `Host CPU Total`.
- [178957820] - Resolved an issue where the **Query Monitor** was refreshing a query list after returning from checking query details when paused.
- [179418127] - Resolved an issue where Command Center was failing to send alert emails to all recipients if there was just one invalid recipient address.
- [179666482] - Resolved an issue where Command Center was recording an incorrect spill file size in the `gpmetrics.gpcc_queries_history` table.

Enhancements and Changes in Greenplum Command Center 4.13.0

New and Changed Features

- The **Query Monitor** page now filters displayed queries based on the query Status: **Running**, **Queued**, or **Blocked**. For more information, see the Query Monitor documentation page.
- The **Query Monitor** page now contains an **Advanced Search** tool. The tool filters displayed queries based on query metrics such as query ID, database name, resource group, and more. For more information, see the Query Monitor documentation page.
- A user can now pause the **Query Monitor** in order to view a snapshot of queries. For more information, see the Query Monitor documentation page.
- The **Query Monitor** now displays session information, such as session status, associated users and databases, idle time, and associated queries. A user with Admin or Operator permission can see and cancel all users' sessions, as well as export session details to a CSV file. For more information, see the Query Monitor documentation page.
- A user can create a workload rule to terminate idle sessions and can examine rule logs of terminated sessions. For more information, see the Workload Management documentation page.
- A user can create rules by spill file size. For more information, see the Workload Management documentation page.
- The `gpmetrics` schema contains a new `gpmetrics.gpcc_queries_now` table that stores real-time query metrics data.
  The `gpcc_wlm_rule` table has two additional rows: one for storing parameters for idle session kill rules, and one for storing spill file size.
  For more information , see the gpmetrics Schema Reference documentation page.
- Microsoft IE is no longer a supported browser for Command Center.

Resolved Issues

- [176183074] - Resolved CORS security issues.
- [31245] - The `gpcc_queries_now` table is now available as part of the `gpmetrics` schema. For more information , see the gpmetrics Schema Reference documentation page.
- [31229] - Resolved an issue in which queries with long query text were blocking.
- [31473] - Resolved an issue in which Command Center was sending alert emails too frequently.
- [31443] - Resolved an issue in which Command Center reported blocked DDL queries as running.

Enhancements and Changes in Greenplum Command Center 4.12.0

New and Changed Features

- The **History** page now contains a new **Advanced Search** tool. The tool allows users to search based on query metrics such as custom time range, query text, database name, user name, and more. Selecting "Custom" from the Time Range dropdown also displays the **Advanced**

Search pop-up window. For more information see the Query History documentation page.

- The **Query details** page now displays the query cost when clicking on the "+" sign next to the query Planner name. The submitted time now inludes a tooltip which displays the query's submitted date. See Query History Metrics for more details.
- Greenplum Command Center now supports IPV6. To install an IPV6-compatible version of Greenplum Command Center, pass the new `-ipv6` parameter to the `gpccinstall` command.
- The `stats_check_interval` parameter is now set in the `gpcc.conf` file rather than the `app.conf` file. If you currently have it set in `app.conf`, you must reset it in `gpcc.conf`; the value in `app.conf` will be ignored.
- Command Centre now displays a **Notifications** label at login. Clicking on the label displays the Alerts window with alert history from the last 7 days. For more information, see the Alerts documentation page.
- The **Table Browser** page has a new **Advanced** search link to allow administrators to filter their search based on table name, table size, or table owners. The results can be exported to a local file. See the Table Browser documentation page for more details.
- The `gpmetrics` **schema** contains a new `gpcc_export_log` table that records the log notifications from the "EXPORT ALL" dropdown in the Query History or the Table Browser page.
  The `gpcc_alert_log` table has been renamed to `gpcc_alert_history` and includes a new `config` column that records the alert history from the Notification Center or from email alerts.
  The table `gpcc_alert_rule` has been discontinued.
  For more information , see the gpmetrics Schema Reference documentation page.
- The Command Center dashboard now displays a **Feedback** link, which displays a customer feedback form when clicked. Administrators can enter feedback for current Command Center features and experience. The form requires internet connectivity for the pop-up to appear.
- Command Center now aggregates and displays a single alert notification when multiple segment failures occur at the same time.
- Users can now customize security parameters in the `app.conf` file. For more details, see Setting Security Parameters.

Fixed Issues

- [31026] - Resolved an issue where Greenplum Command Center version 6.3.0 and 6.3.1 would not show a visual query plan in the query monitor in certain cases. Upgrade the `metrics_collector` if your Greenplum Database version is below 6.12.1.
- [31061] - When a query ran less than 5 seconds, the CPU and Disk I/O were not captured. This issue has been resolved.

Enhancements and Changes in Greenplum Command Center 4.11.1

Changes

- The GPCC web server configuration has been altered to reject unsafe DES ciphers.

Fixed Issues

- [30954] - During security scans, some SSL ciphers were highlighted as "medium strength". This issue has been resolved. GPCC will now reject the unsafe ciphers.
- [30939] - When a Greeplum Database query run a function with 0 arguments, the database crashed. This issue has been resolved.
- [30956] - When exporting the queries history page to a CSV file, the `Submitted` field contained the time of the query but not the date. For example, in the GPCC UI, the `Submitted` field would show 2020-09-13 10:25:11 but the CSV file would only show 10:25:11. This issue has been resolved.
- [174798424] - When creating a rule condition containing more than one slice, the rule condition was ignored, and the query ran normally. This issue has been resolved.
- [174796107] - When updating the status of a workload management rule to inactive, when the rule condition was met, the rule remained active . This issue also appeared when deactivating multiple rules in a set, and when one rule of the set had been deleted but the rest remained in the inactive status. This issue has been resolved.
- [174795058] - On the Query History Detail page, a query's Submitted Time did not match the one displayed on the Query List page. This issue has been resolved.
- [174983144] - On the Query Monitor page, when the query list extended further than the window size, the Workload dropdown column did not display correctly. This issue has been resolved.
- [174922355] - When a partition's root table appeared in the `Accuracy % Rank` list, the row count column incorrectly showed a 0 (nil) value. This issue has been resolved.
- [174767109] - When creating a rule using the Planner Cost condition, and subsequently deleting that condition by reducing the planner cost to "0", the updated rule still showed: "planner cost is greater than [0,0]". This issue has been resolved.

Enhancements and Changes in Greenplum Command Center 4.11.0

Workload Management

- The Command Center interface for managing workload assignment rules has moved to a new page, **Workload> Workload Mgmt**. Resource group configuration and idle session management controls remain on the **Workload> Resource Groups** page. See Workload Management.
- Workload management rules can now be assigned using any combination of resource group name (or all resource groups, by default), database role, and query tag identifiers.
- Workload management rules can include one or more conditions that must be met before the rule action is performed. Conditions are triggered after configured maximum values for CPU time, Planner Cost, Running time, Slices, or Total Disk I/O are exceeded by a query.

- For Greenplum 6.8 or later, you can configure workload management rules to automatically move queries to a specified resource group. These rules can be created on earlier Greenplum versions, but are immediately placed in the **Inactive** state.
- Command Center automatically attempts to retry applying a failed rule action 2 times, after waiting a minimum of 15 seconds between retries. You can configure the time interval using the new configuration parameter, `wlm_query_cooldown_time`. See Greenplum Command Center Parameters.
- A new configuration parameter, `wlm_short_query_threshold` , can be used to ensure that Command Center only applies workload management rules after a query has run for at least the specified number of seconds. See Greenplum Command Center Parameters.
- Programmatically managing workload rules using the JSON object is no longer supported.
- The new `gpmetrics` tables, `gpcc_wlm_rule` and `gpcc_wlm_log_history`, were introduced to store workload rule definitions and log history. See the gpmetrics Schema Reference.

Query Monitor

- The Query Monitor page includes a new column, **CPU Time**, to show the amount of system CPU consumed by each query.
- The **Blocked By** column is no longer displayed for active queries. To view information about blocking transactions, use the tooltip that is displayed when the query status is **Blocked**. See Query Monitor.

Permissions Changes

- Only Operator and Admin users can move queries from the Query Monitor page.
- Only Admin users can make changes to the Recommendations and Workload pages.
- Basic users can now view the Table Browser page.

Fixed Issues

- [30545] The metrics collection code was updated to resolve a buffer overflow condition that could cause Greenplum Database to crash when `gp_enable_query_metrics` was set to "on."
- [30812] Resolved a problem where the `rows_out` value displayed an incorrect number for certain queries.
- [173978192] Resolved a problem where the web socket connection was not rebuilt after a user attempted to login to Command Center after a previous session timed out.
- [174275398] Command Center will now fail to start if the web server port (28080) is being used by another program.
- [174665588] Command Center now displays the correct value for **Statement Memory** for resource group entries.

Enhancements and Changes in Greenplum Command Center 4.10.0

Command Center Installation Changes

Command Center directory names have changed to omit `-web`.

- The Command Center installation directory name changed from `greenplum-cc-web-<version>` to `greenplum-cc-<version>`, for example `/usr/local/greenplum-cc-4.9.0`.
- The Command Center installer creates a symbolic link `greenplum-cc` to the Command Center home directory if gpadmin has write permission in the installation directory. If the directory already exists, the link is recreated to link to the new Command Center installation directory. Use the link to access the `gpcc_path.sh` file in your shell startup script to ensure that you access the most recent installation. For example, add this line to your `.bashrc` or `.bash_profile` file:

  ```
  source /usr/local/greenplum-cc/gpcc_path.sh
  ```
- If the installation directory is not writable by the gpadmin user, you can create the directory and symbolic link and set the owner to gpadmin before you run `gpccinstall`.
- You can run more than one Command Center instance on the same host. For example, if you run a Greenplum Database 5.x system and a Greenplum Database 6.x system on the same cluster, you can install Command Center instances for each system on the same master host. This feature is supported on Greenplum Database 5.28.0 or higher.

  Before you run the installer, choose the Greenplum Database instance by sourcing the environment file (`greenplum_path.sh`). You must manually edit the `$GPCC_HOME/app.conf` file for each additional Command Center instance you want to run to choose different port numbers for the `httpport`, `httpsport`, `rpcport`, `ws_perf_port`, and `agent_perf_port` parameters.

Recommendations Feature

The new **Recommendations** feature helps to identify Greenplum Database tables that require maintenance to reduce bloat, improve accuracy of query plans, and reduce skew that affects query performance.

- Schedule a table scan to run at a designated time and for a specified period of time. The scan can be scheduled to repeat daily or weekly on selected days of the week.
- Command Center runs queries to collect information about tables, processing as many tables as it can during the scheduled period. The next scan resumes where the previous scan left off. Collected data is saved in the new `gpmetrics.gpcc_table_info` and [`gpmetrics_table_info_history`](../topics/ref-gpmetrics.html#gpcc_table_info_history) tables in the gpperfmon database.
- The **Recommendations** page lists scanned tables in ranked order with recommendations to `VACUUM`, `VACUUM FULL`, `ANALYZE`, or redistribute tables.

See Recommendations for more information about the Recommendations feature.

I/O Wait Metrics

Command Center now collects CPU IOWait metrics. You can view IOWait metrics on the **Host**

Metrics, **Cluster Metrics**, and **History** pages.

- On the **Host Metrics** page, the **CPU** column value is the total of System, User, and IOWait percentages. The chart and pop-up box break out the System, User, IOWait, and Idle percentages separately.
- On the **Cluster Metrics** and **History** pages, the **CPU** chart shows System and User percentages. The IOWait percentage is included in the System percentage, as in previous Command Center versions. The new **IOWait** chart shows just the percentage of CPU time spent in IOWait.

Workload Management Improvements

You can now set **Memory Spill Ratio %** for resource groups on the **Workload Management** page. Transactions spill to disk when they reach this threshold.

While editing resource group attributes, Command Center recalculates and displays **Statement Memory**, the amount of memory allocated to a query for each resource group.

See Workload Management for more information about these features.

gpcc stop Utility

- If the Greenplum system is down, running the `gpcc stop` command will stop the Command Center agent and web server processes.

Fixed Issues

- When a query called a UDF that runs an inner query, the top level query could be missing in the **Query Monitor** view and in query history. This caused some columns in the **Query Monitor** view to display `-` instead of the correct value. This issue is fixed.
- The disk IO read/write calculations in `gpcc_system_history` and the **Cluster Metrics** page included R/W rates for multiple devices on the same physical disk. This is fixed. The disk IO read/write rate calculations now exclude:
    - partitions of disk
    - dm (device-mapper)
    - loop device
    - md (multi-device such as raid)

  The reported disk IO rate is now the same as the actual IO rate of the physical disk.
- On the **System** **Storage** page, values displayed as terrabytes (TB) were incorrectly converted from gigabytes (GB) by dividing by 1000. This is fixed. Values greater than 1024GB are correctly converted to TB by dividing by 1024.
- Security vulnerability fixes have been added to prevent ClickJacking Attacks and to deny use of risky HTTP request methods.
- The **Running**, **Queued**, and **Blocked** query counts on the **Dashboard** were misleading because they were not the current status at the time you were viewing the **Dashboard**, but the status about 15 seconds earlier. The numbers have been removed to avoid confusion. To view current numbers, go to the real-time **Query Monitor** by clicking the query graph from

the **Dashboard**.

- A **Disk Full** alert was raised when the total disk space for all hosts in the cluster exceeded the threshold. Now an alert is raised if disk usage for any host exceeds the threshold. The alert email includes the name of the host that triggered the alert.
- The backend scan for the table browser periodically connected to the template1 database. This could prevent a user from creating a new database, because `CREATE DATABASE` is not allowed when there are any connections to the template1 database. This issue has been fixed. The template1 database is omitted from the backend scan.

Enhancements and Changes in Greenplum Command Center 4.9.0

Command Center Installation Changes

- The Command Center release download file names have changed to include the Greenplum Database major version, for example `greenplum-cc-web-4.9.0-gp5-rhel7-x86_64.zip`.
- The Command Center installer creates four entries in the `pg_hba.conf` authentication file for the `gpmon` role if there are no existing `gpmon` entries.

```
local      gpperfmon     gpmon                  md5
host       all           gpmon   127.0.0.1/28   md5
host       all           gpmon   ::1/128        md5
host       all           gpmon   samenet        md5
```

  The `samenet` entry is new in this release, and the installer will add it to the `pg_hba.conf` file even when there are existing entries for the `gpmon` role.

  Note that the Table Browser feature requires `all` in the database field of the `host` entries so that `gpmon` can retrieve table information from each database.

  If you use an authentication method other than `md5` for the `gpmon` user, such as LDAP or Kerberos, edit the `pg_hba.conf` file to enable connections from all hosts and access to all databases.

Command Center History Data Collection Changes

- The Command Center version 4.9.0 installer enables History Data Collection.
- If the `gp_enable_gpperfmon` Greenplum Database server configuration parameter is on, the Command Center installer sets it to off. You must restart Greenplum Database for this change to take effect.

  Note: Pivotal recommends that you do not use the Greenplum Database gpperfmon service while running Greenplum Command Center because of its effects on Greenplum Database performance.
- The `history_enabled` property in the `$MASTER_DATA_DIRECTORY/gpmetrics/gpcc.conf` file is deprecated and the installer will comment it out if it is present. A new property `enable_history` is added to the `$GPCC_HOME/conf/app.conf` configuration file. The default is true. If you disable history data collection on the Command Center **Admin> Settings** page,

`enable_history = false` is written in the `app.conf` file.

New and Changed Features

- A new **Table Browser** is added to allow administrators to see information about tables in Greenplum Databases. Tables displayed can be filtered by database, schema, owner, and size. The Command Center interface provides details about tables, including table storage types, distribution policies, partitions, sizes, and last access times. It lists recent queries that have accessed the table, with links into the Command Center query history. Notably, the Table Browser displays only metadata for tables; it does not provide access to data stored in tables.

  Note: When you first start Command Center after installing this version, Command Center loads data into the new `gpmetrics.gpcc_table_info` table in the gpperfmon database. For databases with large numbers of tables, the initial load could take five minutes or longer. Table data will not be available in the Table Browser until it has been loaded.

- The table browser uses two new tables in the `gpmetrics` schema: `gpcc_table_info` and `gpcc_table_info_history`. See gpmetrics Schema Reference for information about the contents of these tables.

- The Command Center web server (`gpccws`) memory consumption is much improved compared to earlier Command Center versions.

- On the **History** page, the count of running queries at any point on the **Queries** graph now includes queries that started and finished in the interval since the previous metrics sample. A new `queries_finished` column is added to the `gpmetrics.gpcc_database_history` table to record this count.

- The metrics collector extension adds a new `gpcc.enable_query_profiling` server configuration parameter that can be enabled to help with performance troubleshooting. When off, the default, the metrics collector does not collect queries executed by the gpmon user in the gpperfmon database or plan node history for queries that run in less than ten seconds (or `min_query_time`, if it is set and greater than ten seconds). If you enable `gpcc.enable_query_profiling` in a session the metrics collector collects those queries. This parameter is introduced in the metrics collector extension included with Greenplum Database 5.24.0 and is not available in earlier Greenplum Database releases.

- Each night, Command Center archives files in the `$GPCC_HOME/ccdata` directory that are more than two weeks old. The files in this directory include saved query text files (`q*.txt`), plan node files (`gpccexec*.txt`), and table size and statistics information files (`gpcc_size_info.dat` and `gpcc_stat_info.dat`). Archive files have names in the format `archive_YYYYMMDD_YYYYMMDD.tar.gz`, where the dates are the beginning and end dates of the week included in the archive. The archived files are no longer needed by Command Center but may be useful for troubleshooting; you can remove them manually if you do not want to save them.

Resolved Issues

- [MPP-30259] The `gpmetrics.gpcc_plannode_history` table definition changed in Command Center 4.7. When upgrading from Command Center 4.4.x, 4.5.x, or 4.6.x to Command

Center 4.7.x or 4.8.x, the installer did not upgrade the definition of the existing table. This resulted in a failure to harvest data to the `gpcc_plannode_history` table. This is fixed. Upgrading from a Command Center version earlier than 4.7 to Command Center 4.9 updates the `gpcc_plannode_history` table definition in the `gpmetrics` schema.

- When a Command Center user signs out of Command Center, the Welcome page is displayed, but the message "Checking DB status..." is displayed for several seconds before the login form is presented. This is fixed. If there are no errors the login form is now displayed immediately.
- The rightmost edge of system metrics graphs could drop to zero if the metrics data for that period was not yet available to display. This is fixed by cutting the time axis on the right edge of metrics graphs to the last period with available data. As a result of this fix, the time period displayed at the right edge of a graph can be slightly earlier than the last sync time displayed at the top of the page.
- Users with self-only permissions could see other users' query history. In Command Center 4.9 users with self-only permission have no access to the query history page, so they cannot see their own or others' query history. Self-only users will be allowed access to their own query history in a future Command Center release.
- The `send_alert.sh` script in the `$MASTER_DATA_DIRECTORY/gpmetrics` directory had a bug that caused some values to be incorrectly expanded when substituted into the template. This is fixed by replacing all occurrences of `$ARGUMENT` with `${ARGUMENT}` in the script. You can implement the same fix if you have created a custom `send_alert.sh` script.
- The Command Center **Storage Status** page did not display tablespaces with locations other than `pg_default` or `pg_global`. This is fixed. Expanding a hostname now shows each tablespace location. Hovering over the tablespace location displays a pop-up with a list of the data directories at that location. Data directories may not be displayed until Command Center has refreshed the **Storage Status** page, which occurs every four hours.
- Transaction performance issues could occur when the workload manager `gp_wlm` extension was loaded in Greenplum Database and the Command Center workload management feature was not enabled. This is fixed in the `gp_wlm` extension included with Greenplum Database 5.24.0. For earlier versions of Greenplum Database, you can use one of the following methods to work around the problem:
    - If you are using Greenplum Database resource groups, enable Workload Management in the Greenplum Command Center.
    - If you are using Greenplum Database resource queues for workload management, remove the `gp_wlm` shared library from the `shared_preload_libraries` server configuration variable.

```
$ gpconfig -s shared_preload_libraries
```

```
Values on all segments are consistent
GUC           : shared_preload_libraries
Master  value: metrics_collector,$libdir/gp_wlm
Segment value: metrics_collector,$libdir/gp_wlm
```

```
$ gpconfig -c shared_preload_libraries -v 'metrics_collector'
$ gpstop -ar
```

Enhancements and Changes in Greenplum Command Center 4.8.0

New Features and Changes

- The **Enable GPCC history data collection** option on the **Admin> Settings** page is now on by default. If you turned off this option in your current Command Center installation and you upgrade to Command Center 4.8.0, the option will remain off. Pivotal recommends that you turn it on.
- Command Center can collect all metrics without the Greenplum Database gpperfmon service. For lower overhead, stop the gpperfmon service if it is running. Set the `gp_enable_gpperfmon` server configuration parameter to off and restart Greenplum Database.
- If you include the `-W` option with the `gpccinstall` command, the installer prompts you to enter the password for the gpmon role. If the gpmon role does not already exist, the installer creates it with the password you specify. Without the `-W` option, the installer creates the gpmon role with the default password "changeme".
- The installer has a new `-u` (upgrade) option that installs Command Center using the configuration parameters from the current Command Center installation. You can use this option to reinstall the same version of Command Center or to upgrade to a newer version.
- Japanese and Russian language options have been added to the Command Center installer.
- When canceling one or more queries in the Command Center UI, a pop-up box prompts you to enter a message of up to 128 characters to add to the error message that is displayed to the user.
- On the **System> Storage Status** view, when you expand a **Hostname**, the **Data Directory** column now lists the mount points of partitions on the segment host file system that contain Greenplum segment data directories. When you move your cursor over a mount point, a tooltip lists the data directories the partition contains.
- History query search performance is greatly improved.
- Newly created partitions for history tables in the `gpmetrics` schema will be created with RLE compression and the `COMPRESSLEVEL=2` option, for improved compression/performance. Existing partitions are not changed.
- The Command Center web server now supports only the TLSv1.2 protocol and above for secured connections.
- Logging improvements. A new `log_level` parameter in the `$GPCC_HOME/conf/app.conf` configuration file determines the level of messages that will be logged. The parameter specifies which messages are added to these log files: `gpccinstall.log`, `webserver.log`, `agent.log`, and `cli.log`.

  The parameter can be set to `Debug`, `Info`, or `Error`, where `Debug` is the most verbose and `Error` is the least verbose. The parameter values are not case-sensitive. The default is `Info`.

Fixed Issues

- [#166658749] - If the `MASTER_DATA_DIRECTORY` environment variable is not set when `gpcc start` is run, the `gpmetrics` directory is created in the current directory. This issue is resolved. If the `MASTER_DATA_DIRECTORY` is not set, the `gpcc start` command will print a message and exit.
- [#166346339] - If CPUSET is enabled for a resource group, the CPU% value is changed to `-1`. This value prevented editing resource groups in Command Center. Command Center now accepts a `-1` value.
- [#167156751] - Query text files in the `$MASTER_DATA_DIRECTORY/gpmetrics/query_text` directory were not cleaned up automatically. This issue is fixed.
- [MPP-29944] - When Command Center is configured to use Kerberos authentication with the user id as the Command Center user, a query cancelled by the user was logged as cancelled by the gpmon user. This is fixed. If the optimizer is set to GPORCA, an extra message, "Feature not supported: SIRV functions", is logged. This is a known issue.

Enhancements and Changes in Greenplum Command Center 4.7.0

Greenplum Command Center 4.7.0 contains the following new changes and features.

Command Center Installation New Features and Changes

- The `gpccinstall` installation utility now writes details of the installation to the installation log file, `gpccinstall.<timestamp>.log`. Previously, the log contained a success message or any errors that prevented a successful message. The log now contains the values for the parameters used during the Command Center installation.
- The `gpccinstall` utility must be run by the gpadmin user and the installation directory must be writable by the gpadmin user. The option to install Command Center as root and change ownership of the installed files to gpadmin is not supported because the root user cannot connect to Greenplum Database.
- The new `gpccinstall -auto` option performs a non-interactive installation, using default values for all of the installation parameters.
- Added the Korean language option for the Command Center User Interface.
- It is now recommended to run Command Center on the master host, instead of the standby master host. Saving plan node history, a new feature in this Comand Center release, can generate a large number of messages between the agent running on the master host and the Command Center backend. Running Command Center on the master host keeps this traffic off of the LAN.

gpmetrics Schema Changes

- When real-time metrics collection is enabled, Command Center saves execution plan node details to the `gpmetrics.gpcc_plannode_history` table in the gpperfmon database. Details are only saved for queries that execute for at least 10 seconds. The data in this table enable you to view visual query execution plan details when you select a query in the **History** view.

Query Monitor and History Changes

- A `Planner` field is added to the **Query Monitor Details** view to identify the planner used to plan the query, either GPORCA or Legacy.
- The detail for a query in the **History** view includes a visual query execution view that shows the actual rows returned for each plan node. History
- The visual plan for a query in the **Query Monitor** shows the **Estimated Progress** for the query, a calculated percentage using the query plan's cost estimate for each plan node and the actual rows vs. estimated rows for each node.

Alerts

- When sending alert emails, Command Center can authenticate with the SMTP server using NONE, PLAIN, LOGIN, or CRAM-MD5 authentication.
- The **Username** and **Password** fields in the **Manage email configuration** section of the **Alerts** page are now optional. These fields are only needed if the SMTP server requires them for authentication.

Logging Changes

- Every hour, Command Center logs the number of queries that were not saved to the gpmetrics history in the previous hour because they completed in less time than than the threshhold time specified on the **Admin> Settings** page. The message is logged in the `logs/webserver.log` file in the Command Center installation directory.

Fixed Issues

- 165236914 - The `gpcc` command-line utility did not work with the `-W` option. This has been fixed. The `gpcc` command will use the gpmon password entered by the user when the `-W` option is supplied. If the `-W` option is not supplied, `gpcc` will use the `PGPASSWORD` environment variable, if set, or the `.pgpass` file in the gpadmin user's home directory.
- 165111508 - LDAP authentication fails because Command Center improperly encodes the username in the Greenplum connection string. This is resolved. Command Center correctly escapes the username in the connection string.
- 164620455 - The Command Center agent memory usage and CPU usage grew very high when executing a query with an inner query. The inner query generated many small queries, sending a large number of packets to the metrics collector. This is fixed. Inner queries are marked so that metrics are not collected for them.
- 164818798 - When Workload Manager terminates an idle connection, a FATAL error and stack trace were logged. Now Workload Manager logs a WARNING message with no stack trace when an idle connection is terminated.
- 164263538, 164718926 - Command Center does not show query information for a COPY command, such as `COPY <table> FROM STDIN;`. The Command Center Query Monitor now displays available information for some `COPY` commands. A `COPY` command run by a non-

superuser role can be controlled by a resource queue if the resource queue's `ACTIVE_STATEMENTS` parameter specifies a maximum limit for the number of queries that can be executed by roles assigned to the queue. (Greenplum Database does not assign cost or memory values to `COPY` commands, so a `COPY` command with no `ACTIVE_STATEMENTS` parameter is not managed by a resource queue.) In Command Center, `COPY` commands managed by resource queues will now appear in the Query Monitor with available activity information. Command Center caches roles' resource queue names and, for these commands, the resource queue name is displayed from the cache. If a role has been recently assigned a different resource queue, the cache may be out of date. See Known Issues for more information.

- 164346696 - Command Center does not display more than one filespace per host. This has been fixed in Command Center.
- 165589279 - The Command Center Dashboard displayed UTC time when the Europe/Moscow timezone was set in the OS and Greenplum Database. The root cause of this issue is that Greenplum Database 5 does not handle the `MSK` time zone properly. This is fixed in Command Center and issue 7543 has been opened for Greenplum Database.

Enhancements and Changes in Greenplum Command Center 4.6.0

Greenplum Command Center 4.6.0 contains the following new features:

Display Language Selection

The Greenplum Command Center user interface has been updated with localization support. At installion time, you can choose the language to use in the Command Center user interface. Command Center currently supports English (default) and Chinese languages.

gpmetrics History Schema (Beta)

The Command Center history views currently draw data from the Greenplum Database gpperfmon database, which has a data collection process that is separate from the Command Center real-time metrics collection system.

In this release, you can enable saving data from the real-time metrics collection system to history tables in the gpmetrics schema. When you enable this feature, Command Center uses the gpmetrics history tables to display history data in the user interface. This is a new feature, currently considered experimental. In a future release, the gpmetrics tables will be the only source for history data and Command Center will no longer depend upon the gpperfmon database.

To enable gpmetrics history, visit the **Admin> Settings** view in Command Center. In addition to enabling history collection, you can set the minimum run-time for queries to be saved in history.

See the gpmetrics Schema Reference for information about the tables in the gpmetrics schema.

Enhancements and Changes in Greenplum Command Center 4.5.0

Greenplum Command Center 4.5.0 contains the following new features:

Alert Rules

On the new **Admin> Alerts** page, Command Center users with superuser permission can configure alert rules. When an alert rule is matched, a record is logged in the `gpmetrics.gpcc_alert_log` table and, optionally, an email is sent to a list of addresses you specify. You can also write a custom shell script to send alerts to other messaging systems.

Alert rules can detect the following events:

- Segment failure
- Out of memory errors
- Average memory usage percentage of segment hosts exceeds a specified percentage for a specified number of minutes
- Memory usage on the master host exceeds a specified percentage for a specified number of minutes
- Total disk space used on all hosts exceeds a specified percentage
- Number of Greenplum Database connections exceeds a specified percentage
- Average CPU usage of segment hosts exceeds a specified percentage for a specified number of minutes
- Spill files for an active query exceed a specified number of gigabytes
- Runtime for a query exceeds a specified number of minutes
- A query is blocked for more than a specified number of minutes

Query Text Download for Long Queries

On the **Query Details** page, if the query text is longer than 100K characters, Command Center only shows the first 100K characters and, when you click **COPY**, copies only the first 100K characters to the clipboard. When the query text is longer than 100K characters, Command Center adds a button, **Retrieve full query text**, which you can click to download a text file containing the entire text of the query. The file is available to download for 24 hours, or until the query is saved to history, when history collection is enabled.

Fixed Issues

- 161436520 - Panels displaying help text can now be scrolled vertically when the text does not fit in the window.
- 158278655 - The displayed metrics for a query could be inaccurate if Command Center received the query "done" status before the final metrics arrived. Now the final metrics are updated even when they arrive after the "done" status.
- 161074984 - Repeated assertion errors logged in the `agent.log` file caused log files to grow quickly and consume too much disk space. This is fixed. Each type of assertion error will be logged no more than once every 10 minutes.
- 162114428 - When a user visits Command Center with a browser, the Command Center web server (gpccws process) establishes a websocket connection and runs two goroutines to service the websocket. When the websocket closes, one of the goroutines does not exit.

After many connections the gpccws process can occupy gigabytes of memory. This memory leak is fixed.

Enhancements and Changes in Greenplum Command Center 4.4.2

- The Greenplum Database metrics collector extension is now enabled only when Command Center agents are running. Previously, if an agent process terminated, the metrics collector continued to collect and send data.

Greenplum Command Center 4.4.2 contains the following resolved issues:

- 160195564 - When Command Center is restarted after a user has submitted a query, and the user submits another query in the same session, the database name and user name are missing from the second query in the Command Center interface. This is fixed.
- 161077294 - The Authentication view now prevents the user from saving an undefined entry to the `pg_hba.conf` file.
- 161274227 - The Command Center installer, `gpccinstall`, no longer prints errors to the `pg_log` log file about missing `iterators` and `emc_connect_history` tables.
- 161273865 - The Query Monitor no longer shows runtimes flashing to 0 seconds when a query is cancelled through the user interface.

Enhancements and Changes in Greenplum Command Center 4.4.1

- The **Admin> Authorization** view allows entering host names in the **Address** field.
- On the visual query plan, a data motion showed a finished status when the send data operation completed. This is changed so that the node completes only after the corresponding data receive operation has also completed.
- Enabled cross-site request forgery prevention during login.

Resolved Issues in Greenplum Command Center 4.4.1

- 160679694 - If Command Center is restarted while on the Query Monitor view and the browser is then refreshed, an extra web socket connection is created. This is fixed.
- MPP-29539 - Command Center agent (ccagent) logging is disabled for a known issue with PL/pgSQL queries.

Enhancements and Changes in Greenplum Command Center 4.4.0

Greenplum Command Center 4.4.0 contains the following features and enhancements.

Workload Management

- Command Center has a new user interface to assist administrators in enabling resource groups in Greenplum Database, importing existing resource queues to resource groups, and enabling workload management with Command Center. The option to import resource queues to resource groups is presented if no resource groups have been created (other than `default_group` and `admin_group`) and Greenplum Database has resource queues to convert (other than `pg_default`). Once the administrator has imported resource queues, or chosen to skip importing resource queues, the option to import queues is no longer presented.

- The resource group list on the Workload Management view has a new column to show the minimum (fixed) amount of memory that will be allocated to a query for each resource group. This value is recalculated when you enter new values while editing resource groups.
- Administrators can now define resource group assignment rules and idle session kill rules with an interactive interface. It is no longer necessary to edit the JSON document for workload management rules. The JSON text field is removed.
- The Workload Management view changed to a light theme.

Permissions

- Command Center users with the Self Only permission level can:
    - see all queries on the query monitor, including queries owned by other users
    - cancel their own queries
    - access the query details view for their own queries
    - hover on a query to see query text for their own queries
    - hover on locking/blocking queries and access details of locking/blocking queries that do not belong to other users

  Users with Self Only permission level cannot see query text or access the details views of others' queries.
- The Greenplum Database roles `gpcc_basic`, `gpcc_operator`, and `gpcc_operator_basic` are created during Command Center installation if they do not already exist.

Query Monitor

- Fixed a bug where the database name and role name were missing from query details when queries are executed in a session after restarting Command Center.

Enhancements and Changes in Greenplum Command Center 4.3.0

Greenplum Command Center 4.3.0 contains the following enhancements.

Resource Group Management

The **Admin>Workload Mgmt** view has a new user interface you can use to add and remove resource groups and to change the Concurrency, CPU %, and Memory % attributes of resource groups.

Resource Group Role Assignments

The **Admin>Workload Mgmt** view has a new user interface you can use to view and change Greenplum Database roles' default resource groups.

Details Added to Visual Query Plan Steps

The metrics collector extension in Greenplum Database release 5.9 is updated to submit additional information about each step in the query plan to the Command Center backend. Command Center displays this information in the visual query plan when you expand a step in the query plan. The new information displayed depends on the operation the step performs and includes details such as hash key, merge key, join condition, or filter condition. Previously, you could only see this information by generating the textual query plan.

Enhancements and Changes in Greenplum Command Center 4.2.0

Greenplum Command Center 4.2.0 contains the following enhancements.

Visual Query Plan

The Command Center Query Details view now includes a visual query plan.

Idle Session Kill Rules

Idle session kill rules can include optional `exemptedRoles` and `message` parameters.

- The value of the `exemptedRoles` parameter is a list of Greenplum Database roles that are exempted from the rule. The list can include Posix regular expressions to match Greenplum Database role names.
- The value of the `message` parameter is a string to include in the message that is displayed when a session is killed.

Command Center Can Run on the Master Host or Standby Master Host

The Greenplum Command Center web server and backend may now be executed on the master host or on the standby master host. Running GPCC on the standby master host is recommended to avoid adding load to the master server, but it is no longer a requirement. After the GPCC software is installed, log in to the host where you want to run GPCC, source the `gpcc_path.sh` file in the GPCC installation directory, and run the `gpcc start` command.

Enhancements and Changes in Greenplum Command Center 4.1.0

Greenplum Command Center 4.1.0 contains the following enhancements.

- Command Center administrators can set permission levels for Command Center users. Permissions are enforced as described in the documentation.
- On the **Query Detail** view, clicking **Copy** in the query text or query plan panel copies the text

in the panel to the clipboard.

- A help icon and in-app help have been added on the **Query Monitor** and **Query Detail** views.

The following workload management features, improvements, and bug fixes have been added in the workload management extension included with Greenplum Database 5.8.0.

- On the **Admin>Workload Mgmt** view, you can add idle session kill rules for each resource group. When you add these rules, the Greenplum Database workload management extension kills a session after it has been idle for the number of seconds you specify. See Workload Management for syntax and examples.
- Optimizations have been implemented to reduce the impact on Greenplum Database when the workload management extension is disabled.
- The workload management extension takes advantage of resource group name-to-id caching added in Greenplum Database.
- Fixed a bug in the workload management extension that caused errors to print to the `psql` prompt, even when the extension was disabled.

Greenplum Command Center 4.1.0 contains the following bug fixes.

- In the **History Detail** view, when either of the **Disk R** or **Disk W** metrics is 0, both are reported to be 0. This is fixed.
- When using the Kerberos gpmon-only authentication mode, generating an explain plan failed. This is fixed.
- The Command Center agent failed with the message "Error: can't find gpcc.query_metrics_port, metrics_collector is not correctly installed." This occurs when running Command Center on a Greenplum Database system that was upgraded from an earlier Greenplum Database 5.x release. The `metrics_collector` and `gp_wlm` extensions are installed with the upgrade, but the upgrade process does not perform the required configuration changes in the `postgresql.conf` configuration file. The Command Center installation instructions now include steps to manually configure and restart an upgraded Greenplum Database system.
- When Command Center starts, an error message is written in the Greenplum Database log file: "function gpcc_schema.read_pghba(unknown) does not exist." This is fixed.
- A change to time zone handling in Greenplum Database 5.7 can cause Greenplum Command Center 4.0.0 to display an incorrect time if the master host operating system time and Greenplum Database use different time zones. Now Command Center times are displayed using the time zone of the Greenplum Database master host operating system. The current time, last sync time, and timestamps in alert logs, cluster metrics, and query history are all displayed using the master host's system time zone.
- On the **Admin> Authorization** view, when the authorization method is `gss` and the options field contains text, changing the method to `trust` does not clear the options field and it is not possible to save changes to the `pg_hba.conf` file.
- The Command Center web server, `gpccws`, spawns ssh processes but does not reap them in a timely manner, leading to many zombie processes. This is fixed.

- On the **Admin> Authentication** view, if the number of users listed in the user column is longer than can be displayed, Command Center truncates the list and adds ellipsis (`...`) to the end. Only administrators can view the complete value, by editing the field. Now any user with access to the view can see the full list of users by hovering over the field.

Enhancements and Changes in Greenplum Command Center 4.0.0

Command Center Installation Changes

Greenplum Command Center 4.x software, unlike previous releases, is installed on every host in the Greenplum Database cluster. The Command Center web server and backend run on the standby master, if your Greenplum Database cluster has a standby master. If there is no standby master, Command Center runs on the master host instead.

To modify the Command Center installation—for example to enable or disable SSL or install Command Center on new or replaced hosts—just re-execute the installer and restart Command Center. It is not necessary to uninstall Command Center before reinstalling.

There is one Command Center installation per Greenplum Database cluster. It is no longer necessary to create Command Center instances after installing the software.

The `gpcmdr` command-line utility is replaced with the new `gpcc` utility. Use the `gpcc` utility to start and stop Command Center and metrics collection agents, check Command Center status, and enable or disable Kerberos authentication.

In previous releases, the gpmon role required only local connections to databases on the Greenplum master host. In Greenplum Command Center 4.x, the gpmon user must be able to connect to databases from the host running the Command Center web server and backend, which requires adding a host entry to the `pg_hba.conf` authentication configuration file.

Real-time Query Metrics

Greenplum Command Center 4.0 introduces real-time query metrics for Pivotal Greenplum Database 5.7 and above. This new feature combines the following new features in Greenplum Database and Greenplum Command Center:

- Greenplum Database saves query execution metrics in shared memory while queries execute.
- A new Greenplum Database metrics collection extension, included with Pivotal Greenplum Database, emits the saved metrics as UDP datagrams.
- A new Greenplum Command Center metrics collection agent running on each Greenplum Database host receives the datagrams and posts metrics to the Greenplum Command Center backend. The Command Center backend starts and manages the metrics collection agents.

The Command Center Query monitor view updates in real time so you can see queries that are waiting to execute and the current status and resource usage for queries that are running.

Metrics collection now includes lock and spill file information. On the Query Monitor, you can see which queries are blocked and which queries hold the locks blocking them. The Query Monitor shows the total size of spill files created on all segments for each query.

Installing the gpperfmon database remains a prerequisite for Command Center. The gpperfmon database is the source for query history displayed in the Command Center user interface. The new real-time metrics are not persisted and are not directly related to the metrics collected and persisted in the gpperfmon database.

Workload Management

Workload management is now an integrated Command Center feature rather than a separate product. Workload management is available in Command Center only after resource groups have been enabled in Greenplum Database by changing the `gp_resource_manager` server configuration parameter from `'queue'` to `'group'` and enabling Linux control groups (cgroups).

In Command Center 4.0, workload management allows you to assign transactions to Greenplum Database resource groups at execution time by evaluating the current database role and *query tags* against workload assignment filters you define in Command Center. Query tags are user-defined `name=value` parameters that you define in the `gpcc.query_tags` database session parameter. You can define multiple query tags separated by semicolons. Set query tags in a Greenplum Database session either as a parameter in the database connection URL or by executing `SET gpcc.query_tags TO '<tag1>=<val1>;<tag2>=<val2>;...'` in the database session.

When a transaction is about to execute, the current database role and query tags are compared to the workload assignment filters that you have created in Command Center. If a match is found, the transaction is assigned to a resource group according to the workload management filter. Otherwise, the transaction is assigned to the database user's resource group, which is the default behavior when Command Center workload management is not enabled.

Workload management uses the `gp_wlm` database extension included with Pivotal Greenplum Database.

Unimplemented Features

Some features available in previous Greenplum Command Center releases have been removed or are not yet implemented in Command Center 4.x.

- The ability for a Command Center admin to post a message to the Query Monitor view is not yet implemented.
- The multi-cluster view has been removed.

Known Issues

The following are known issues in the current Greenplum Command Center release.

Performance issues with a large query string

Submitting a large query string with `psql -c` or libraries causes Command Center performance issues.

<a id=179572065">Incorrect query ID displayed on "Query is blocked for xxx minutes alert"

When the "Query is blocked for xxx minutes" alert is triggered, the query ID shown on the notification feed may be incorrect.

<a id=178295227">Query tags rule not enforced when query tags are surrounded by single quotes

Resource group assignment by a query tags rule has no effect if the query tags are defined by `PGOPTIONS` with single quotes.

<a id=mpp31132">Requirement for a trusted certificate, even when insecure authentication is enabled

When the "Allow insecure authentication" setting is enabled, but the email protocol is STARTTLS, Command Centre ignores the setting and still expects a trusted certificate.

Command Centre requires restart after Daylight Savings Time

When Daylight Saving Time occurs, Command Center requires a restart to repopulate certain data metrics.

Last Accessed column affected by persistent table rebuild

When using the `gppersistentrebuild` utility, the **Last Accessed** column in the Table Browser view might be refreshed, although no queries accessed the tables.

Discrepancy in the Query Submitted Time

On the Query History Detail page, a query's Submitted Time does not match the one displayed on the Query List page.

Limitation in Displaying Workload Group Drop Down Options

On the Query Monitor page, the last row (or the only row) of the query list doesn't display the Workload column menu options correctly. As a workaround, scroll the page to show the Workload dropdown menu content.

Limitation in the Accuracy % Rank list

When a partition's root table appears in the `Accuracy % Rank` list, the row count column incorrectly shows 0 (nil) value.

Limitation when a rule condition contains more than one slice

When you create a rule that contains more than one slice, the rule is ignored, the condition is not met, and the query runs normally.

Workload Management rule still effective after status set to inactive

When you update the status of a workload management rule to inactive, the rule remains effective when the condition is met. The issue also appears when a set of multiple rules have been set to inactive, or when one rule of the set has been deleted but the rest remain in the inactive status.

Planner Cost condition can not be removed from a rule

When you create a rule that uses the Planner Cost condition, and subsequently delete that condition by reducing the planner cost to "0", the updated rule still shows: "planner cost is greater than [0,0]".

Limitation for Defining Planner Cost in Workload Rules

When you define a workload management rule that uses the Planner Cost condition, the input field transforms your entry to a double datatype value. This can limit the ability to accurately define large planner costs that may be necessary for the GPORCA optimizer. For example, if you enter the GPORCA maximum cost value as 1457494496834852608, the actual value is converted to 1457494496834852600 for the rule. The value shown in the completed rule definition is the value that the rule enforces, and it may not be the exact value that you entered. Increase the cost value as necessary to cover the cost you want to set as the maximum.

Limitation for Defining Planner Cost in Workload Rules

When you define a workload management rule that uses the Planner Cost condition, the input field transforms your entry to a double datatype value. This can limit the ability to accurately define large planner costs that may be necessary for the GPORCA optimizer. For example, if you enter the GPORCA maximum cost value as 1457494496834852608, the actual value is converted to 1457494496834852600 for the rule. The value shown in the completed rule definition is the value that the rule enforces, and it may not be the exact value that you entered. Increase the cost value as necessary to cover the cost you want to set as the maximum.

External Updates to Workload Management Rules are Delayed

If you recreate the `gp_wlm` extension from outside of Command Center after you have already created workload management rules using the Command Center interface, the rules engine may not run for a period of roughly 1 hour. This behavior occurs because Command Center checks for the availability of the extension every hour. Any changes you make outside of Command Center in this situation will not be visible until Command Center checks for the extension, or until you login to Command Center and access the **Workload> Workload Mgmt** page.

Unrecognized Configuration Parameter Error Message

When starting Command Center GPCC 4.10.0 with a Greenplum Database version between 5.19.0 and 5.27.1, the message `unrecognized configuration parameter gpcc.metrics_collector_pkt_version` is logged. This message is expected and can be ignored. It will be suppressed in the next Greenplum Command Center release.

Customized SSH Path Not Supported with the Upgrade (-u) Option

If you upgrade your Command Center installation using the `gpccinstall -u` option and you also specify an SSH program using the `-ssh-path <path>` option, the customized SSH path will not be used during the installation and the `ssh_path` parameter will not be set in the `app.conf` file.

Command Center Installer Truncates pg_hba.conf LDAP Entries

When the Command Center `gpccinstall` installer updates the `pg_hba.conf` file it truncates LDAP information on entries that use the `ldap` authentication method. The truncated information must be reinserted manually after the Command Center installation is complete. This issue exists in Command Center versions 4.7 through 4.9, 6.0, and 6.1.

Loading `gp_wlm` Extension Without Enabling Resource Groups Can Cause Performance Issues

- A performance issue can occur if Workload Management is not enabled in Command Center and the `gp_wlm.so` library is included in the `shared_reload_libraries` system configuration parameter when Greenplum Database starts. The fix for this issue is in Greenplum Database 5.24.0.

  If you cannot upgrade to Greenplum Database 5.24, you can prevent problems with one of these workarounds:

  - If you are using resource group workload management in Greenplum Database, ensure that Workload Management is enabled in Command Center.
  - If you are using resource queue workload management in Greenplum Database, remove `gp_wlm` from the `shared_preload_libaries` configuration parameter and restart Greenplum Database.

Calculated Root Table Size Inaccurate for Some Partitioned Tables

When viewing the **Table Browser** view, the calculated size for a partitioned table is incorrect if some child partitions are in different schemas than the root table.

Sorting Tables by Size on Partition List Page is Slow

If there are a large number of tables in a schema in a database, sorting the partition table list by size can take a long time to display.

Failure to Auto-Create Monthly Partitions in gpmetrics Schema

In time zones with daylight savings time (DST) ending in the month of October, Greenplum Command Center fails to create new October partitions for tables in the gpmetrics schema because it specifies the same start and end date for the partitions. The error does not occur in time zones that do not have DST, or in time zones with DST ending in November.

Workaround

Manually create partitions for October 2019.

Log in to the Greenplum master server as the gpadmin user and connect to the gpperfmon database as the gpmon user:

```
psql -U gpmon gpperfmon
```

Enter the following commands to create the October 2019 partitions:

```
ALTER TABLE gpmetrics.gpcc_disk_history
      ADD PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-01') EXCLUS
IVE
```

```
      WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=RLE_TYPE, COMPRESSLEVEL=
2);

ALTER TABLE gpmetrics.gpcc_system_history
      ADD PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-01') EXCLUS
IVE
      WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=RLE_TYPE, COMPRESSLEVEL=
2);

ALTER TABLE gpmetrics.gpcc_queries_history
      ADD PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-01') EXCLUS
IVE
      WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=RLE_TYPE, COMPRESSLEVEL=
2);

ALTER TABLE gpmetrics.gpcc_plannode_history
      ADD PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-01') EXCLUS
IVE
      WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=RLE_TYPE, COMPRESSLEVEL=
2);

ALTER TABLE gpmetrics.gpcc_database_history
      ADD PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-01') EXCLUS
IVE
      WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=RLE_TYPE, COMPRESSLEVEL=
2);

ALTER TABLE gpmetrics.gpcc_alert_log
      ADD PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-01') EXCLUS
IVE
      WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=RLE_TYPE, COMPRESSLEVEL=
2);

ALTER TABLE gpmetrics.gpcc_pg_log_history
      SPLIT DEFAULT PARTITION START (date '2019-10-01') INCLUSIVE END (date '2019-11-0
1') EXCLUSIVE
      INTO(PARTITION t2019Oct, DEFAULT PARTITION);
```

Query Monitor is Empty When Using HTTP Proxy

When accessing the Command Center frontend or backend with an HTTP proxy in place, no query activity is displayed in the Query Monitor. To work around this issue, the proxy must be disabled for Command Center client browser connections and for `ccagent` processes connecting to the Command Center backend from the Greenplum master and segment hosts. See Using an HTTP Proxy Server with Command Center for more information about this issue.

Query Monitor is Empty When Using HTTP Proxy

When accessing the Command Center frontend or backend with an HTTP proxy in place, no query activity is displayed in the Query Monitor. To work around this issue, the proxy must be disabled for Command Center client browser connections and for `ccagent` processes connecting to the Command Center backend from the Greenplum master and segment hosts. See Using an HTTP Proxy Server with Command Center for more information about this issue.

Cyrillic Font Support in the Pivotal UI

The Source Sans Pro font in the Pivotal UI toolkit does not contain Cyrillic symbols, so the Command Center UI falls back to the default sans-serif font. This may cause some cosmetic rendering problems for Russian users.

Unable to View Real Time Queries After Upgrading From a Previous Command Center Release

If you install a new version of Greenplum Command Center using the same port number as the previous version, and you use the Chrome web browser, you may be unable to view real-time queries until after you clear the browser's cache. See the note in Connecting to the Command Center Console for steps to clear the browser cache.

Resource Queue Name Incorrect for Some Queries

To display the resource queue name for queries in the Query Monitor, Command Center caches user IDs with resource queue names. The cache is updated when the Command Center backend receives a planned query from the metrics collector. Some utility commands are not planned, such as `COPY` commands that have no `SELECT` clause. These commands use a resource queue slot, but are not passed through an optimizer. If a user is assigned a different resource queue and then executes an unplanned command, the Command Center Query Monitor will display the resource queue name that was previously cached.

Cannot Set SSH Path When Upgrading Command Center

The Command Center installer `-ssh_path <path>` command-line option specifies the path to a custom SSH command and also saves the path to the `ssh_path` parameter of the `app.conf` configuration file. If both the `-u` (upgrade) and `-ssh_path` options are specified on the `gpccinstall` command line, however, the `-ssh_path` option is ignored. The installer does not use the custom SSH command and the path is not saved in the `app.conf` file.

About Pivotal Greenplum Command Center

Pivotal Greenplum Command Center is a management tool for the Pivotal Greenplum Database Big Data Platform. This topic introduces key concepts about Greenplum Command Center and its components.

Greenplum Command Center Features

Greenplum Command Center monitors system performance metrics, analyzes cluster health, and enables database administrators to perform management tasks in a Greenplum Database environment.

Greenplum Command Center provides a browser-native HTML5 graphical console for viewing Greenplum Database system metrics and performing certain database administrative tasks. The Command Center application provides the following functionality:

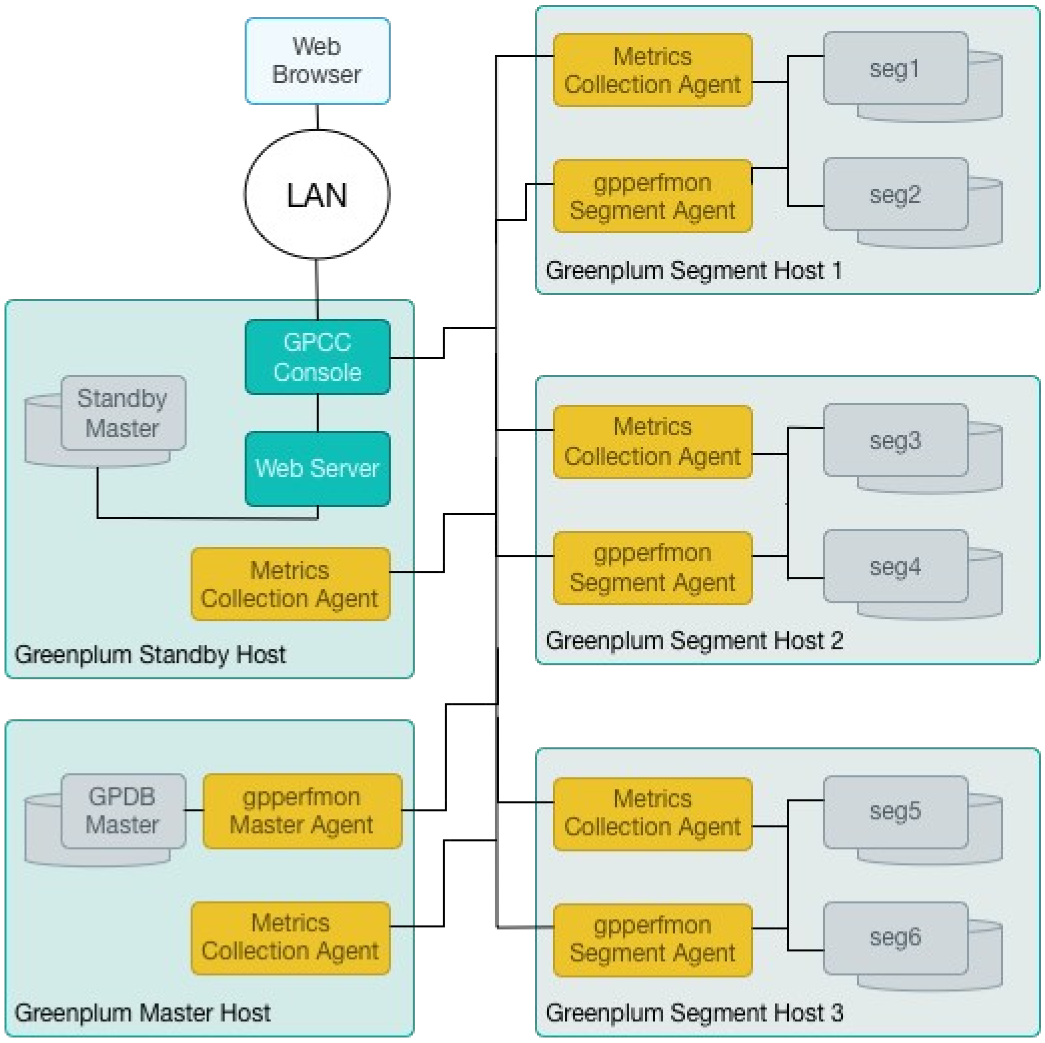
- Interactive overview of realtime system metrics. Drill down to see details for individual cluster hosts and segments.
- Detailed realtime statistics for the cluster and by server.
- Query Monitor view lists queries executing, waiting to execute, and blocked by locks held by other queries.
- Query Detail view shows query metrics, query text, and the execution plan for the query.
- Workload Management view allows administrators to:
    - Create and manage workloads to manage concurrency and allocate CPU and memory resources.
    - Change default resource groups for Greenplum Database roles.
    - Create assignment rules to assign transactions to resource groups.
    - Create idle session timeout rules to set the amount of time before an idle session is killed.
- Four permission levels allow users to view or cancel their own or others' queries, and to view or manage administrative information.
- Cluster Metrics view shows synchronized charts of historical system metrics.
- History view lists completed queries and system metrics plotted over a selected time period.
- Permissions view to see or manage Command Center permission levels.
- Authentication view to see or edit the `pg_hba.conf` host-based authentication configuration file.
- Segment Status view with summaries and details by segment.

- Storage Status view with summaries and details by segment data directory.

Note: Command Center versions 6.4/4.12.0 and later do not run on the Microsoft IE browser.

Greenplum Command Center Architecture

The following figure illustrates the Greenplum Command Center architecture.



Greenplum Command Center Web Server and Web Application

The Greenplum Command Center web server and backend application can run on the master or standby master host—the master host is recommended. The web server, gpccws, is a custom HTTP server designed for Command Center. The web application is an HTML5 and Go language application.

The Command Center web server authenticates users with the Greenplum Database authentication system. Administrators can edit the Greenplum Database host-based authentication file, `pg_hba.conf`, in the Command Center Console. Command Center can also be configured to authenticate users in a Kerberos environment.

Command Center defines four user authorization levels to manage users' access to the Query Monitor, and to administrative information and operations. User authorization is managed in the Administrative area of the Command Center user interface.

Greenplum Command Center displays information derived from several sources:

- Greenplum Database performance monitoring database (gpperfmon)
- Operating system process accounting
- Greenplum Database system catalog tables
- Real-time query metrics collection extension
- Workload management extension

Greenplum Database is instrumented to enable capturing performance metrics and tracking query execution. The performance monitoring database and the query metrics collection extension deploy agents—processes running on each host to collect metrics. The gpperfmon agents forward collected data to an agent on the Greenplum Database master. The real-time query metrics agents submit collected data directly to the Command Center rpc port. The agents also collect data from the host operating system so that query performance can be correlated with CPU and memory utilization and disk space can be monitored in Command Center.

Real-Time Query Metrics Collection

The real-time query metrics collection collects detailed and current statistics. Command Center users can observe queries as they execute and, with sufficient permissions, cancel problem queries to allow other queries to complete.

The Greenplum Database query metrics extension and the metrics collection agent work together to collect real-time metrics and update the Command Center application.

Greenplum Database calls the query metrics extension when a query is first submitted, when a query's status changes, and when a node in the query execution plan initializes, starts, or finishes. The query metrics extension sends metrics to the metrics collection agent running on each segment host. The extension also collects information about the locks queries hold so that you can see which queries hold locks that block other queries. The agent posts the metrics to the Greenplum Command Center rpc port.

The `metrics_collection` extension is included with Pivotal Greenplum Database. The extension is enabled by setting the `gp_enable_query_metrics` server configuration parameter to on and restarting the Greenplum Database cluster. The metrics collection agent is installed on each host when you install Greenplum Command Center. The Command Center application monitors the agent and restarts it if needed.

Command Center Workload Management

Workloads set concurrency, memory, and CPU resource limits for database transactions they manage. A Greenplum Command Center workload corresponds to a Greenplum Database resource group, but adds additional capabilities that are not available with resource groups.

Command Center allows administrators greater flexibility in assigning transactions to workloads. Every Greenplum Database role is assigned to a single resource group and, by default, transactions are managed by the role's resource group. With Command Center workload management, administrators can define criteria to assign transactions to workloads based on attributes other than the role submitting the transaction. Currently, assignment criteria can evaluate query tags and roles in combination with query tags.

A *query tag* is a key-value pair defined in the `gpcc.query_tags` parameter of a database session. The parameter has the format `<tag1>=<value1>;<tag2>=<value2>`, where tags and values are user-defined values. For example, if you want to run ETL operations in a workload named "etl", you could define a tag named "xact-type" and set it to "etl": `xact-type=etl`. The `gpcc.query_tags` parameter can be set as a connection parameter on Greenplum Database clients that allow it, or with a `SET` command inside the session after the connection has been established, for example `SET gpcc.query_tags='xact-type=etl'`.

The `gp_wlm` extension in Pivotal Greenplum Database provides support for Command Center workloads. The extension is included with Pivotal Greenplum Database, but is not enabled by default. Initially, Greenplum Database uses resource queues to manage resources. Using Command Center workloads requires enabling resource groups in Greenplum Database. Resource groups are based on the Linux control groups (cgroups) service, which must first be enabled in the operating system.

Installing Greenplum Command Center

Perform these tasks to install Greenplum Command Center on your Greeplum Database system:

- Installing Greenplum Command Center
- Securing the gpmon Database User
- Setting the Command Center Environment

Additional topics:

- Upgrading Greenplum Command Center
- Uninstalling Greenplum Command Center

Installing VMware Tanzu Greenplum Command Center

The VMware Tanzu Greenplum Command Center installation utility installs the Command Center files on all hosts in the Greenplum Database cluster, creates the gpperfmon database and gpmon user, if needed, and creates the `gp_metrics` schema in the gpperfmon database.

Note: Run the Greenplum Command Center installer on the Greenplum Database master host. The installer installs the Command Center software on every host in your Greenplum Database cluster. It retrieves the list of hosts in your Greenplum Database cluster from the `gp_segment_configuration` system table.

After you have run the installer you can start Greenplum Command Center on the master host (recommended) or on the standby master host.

Prerequisites

Before installing Greenplum Command Center, ensure the following requirements are met:

- Greenplum Database must be installed and running. See the VMware Tanzu Greenplum Command Center release notes for compatible Greenplum Database versions.
- The Greenplum Database `MASTER_DATA_DIRECTORY` environment variable must be set.
- The directory where Greenplum Command Center will be installed, `/usr/local/` by default, must be writable by the gpadmin user on all Greenplum Database hosts. See Selecting and Preparing an Installation Directory for Command Center.
- Port 28080 (default) must be open to TCP connections from Web clients to the HTTP server on the master and standby master hosts. Greenplum Command Center web browser clients connect to this port to access the Command Center Console. Browser connections use HTTP/HTTPS and WebSocket (WS)/Secure WebSocket (WSS) protocols. A different port number can be specified when Command Center is installed. To access the Command

Center web server through a proxy, the proxy must have WebSocket support.

- Port 8899 must be open on all hosts in the Greenplum Database cluster for TCP connections. This is an RPC port, used by the metrics collection agents on the segment hosts to send metrics to the backend.
- If you choose to have Command Center server support SSL/TLS encryption for browser connections -- the default behavior -- you need a combined SSL certificate file containing the server certificate and private key. See SSL/TLS Encryption for more information.

Important! If you upgraded to VMware Tanzu Greenplum Database release 5.7 or later from an earlier Greenplum Database 5.x release, you must follow steps at Set Up the Metrics Collection and Workload Management Extensions before you start Greenplum Command Center.

Selecting and Preparing an Installation Directory for Command Center

The Command Center installation directory (default `/usr/local`) must exist and be writable by the gpadmin user on every host in the Greenplum Database cluster. The Command Center installer creates a directory named `greenplum-cc-<version>` in this directory on every host. When Command Center installation is complete the `greenplum-cc-<version>` directory and all of its contents must be owned by the gpadmin user. If Command Center creates the `greenplum-cc-<version>` directory, it also creates a `greenplum-cc` symbolic link to it, providing the gpadmin user has permission to create the link. You can use the `greenplum-cc` link in your shell startup script to source the `gpcc_path.sh` file.

In a standard Linux system, the `/usr/local` directory is owned by root and is only writable by root. If you choose the default installation directory or another directory where gpadmin does not have write permission, the root user can create the Command Center directory and the symbolic link in the installation directory on each host and make gpadmin the owner. Then the gpadmin user can run the installer without error.

If the gpadmin user has sudo access, you can use the `gpssh` utility to create the directory and set the owner to gpadmin on all hosts with these commands:

```
$ source /usr/local/greenplum-db-<version>/greenplum_path.sh
$ gpssh -f <hostfile> 'sudo mkdir -p /usr/local/greenplum-cc-4.11.0; chown -R gpadmin:
gpadmin /usr/local/greenplum-cc-4.11.0'
$ gpssh -f <hostfile> 'sudo ln -s /usr/local/greenplum-cc-4.11.0 /usr/local/greenplum-
cc'
```

The `<hostfile>` text file contains a list of all Greenplum host names, including the master, standby master, and segment hosts.

See Installing the Greenplum Database Software for information about setting up passwordless SSH.

Install the Greenplum Command Center Software

Run the Greenplum Command Center installer on the Greenplum Database master host. The installer copies the software to all other hosts in the cluster.

1. Download the Greenplum Command Center distribution file for your Greenplum Database

   version from Tanzu Network and copy it to the gpadmin user's home directory on the master host.

2. Extract the installer from the zip file.

   ```
   $ unzip greenplum-cc-web-gp6-<version>-<platform>.zip
   ```

   Extracting the installer creates a `greenplum-web-<version>` directory containing the `gpccinstall-<version>` installation utility.

There are four ways to run the Greenplum Command Center installer:

- Interactive – the installer prompts you for the installation parameters.
- Scripted – you run the installer with a configuration file containing installation parameters.
- Upgrade – the installer uses the installation parameters from the current Command Center installation.
- Auto – the installer uses default installation parameters.

Note: The Command Center installer uses HTTPS as the default protocol for the Command Center web server. If you choose HTTP instead, Command Center issues a warning message. The only exception is if you install with the `-auto` option, for which the default protocol is HTTP.

Interactive Installation

1. Source the `greenplum_path.sh` script in the Greenplum Database installation directory to ensure the `GPHOME` environment variable is set.

   ```
   $ source /usr/local/greenplum-db/greenplum_path.sh
   ```

2. Run the Greenplum Command Center installer.

   ```
   $ cd greenplum-cc-<version>
   $ ./gpccinstall-<version>
   ```

   You can add the following options to the `gpccinstall` command-line.

   - The `-W` option instructs the installer to prompt for the gpmon database user's password.
     - If the gpmon user does not yet exist, the installer creates it using the password you enter at the prompt. The password is not saved anywhere, so be sure to remember the password you enter.
     - If the gpmon user already exists, the installer uses the password you enter at the prompt for the gpmon user. It ignores any password set in the `PGPASSWORD` environment variable or in the `.pgpass` file (`~gpadmin/.pgpass`, or the file specified in the `PGPASSFILE` environment variable).
     - If the gpmon user already exists, but you do not specify the `-W` option, the installer uses the password set in the `PGPASSWORD` environment variable or in the `.pgpass` file.
     - If the gpmon user does not yet exist and you do not specify `-W` (or if you

specify the `-W` option but enter no password at the prompt) the installer creates the gpmon user with a default password. See the `.pgpass` file to find the password.

- The `-ssh_path` option allows you to specify the full path to a custom SSH program. If you do not supply this option, the installer uses the `ssh` command on the path. Example:

```
$ ./gpccinstall-<version> --ssh_path /usr/local/bin/ssh -W
```

- The `-ipv6` option installs a version of Command Center with support for running on IPv6 networks. To enable IPv6, you must set the `HTTPAddr` configuration parameter in the `$GPCC_HOME/conf/app.conf` file to the IPv6 address of the host that runs Command Center.
- The `-krbenable` option includes prompts to configure Command Center Kerberos authentication. The Kerberos prompts are omitted from the installation dialog by default.

3. **Where would you like to install Greenplum Command Center? (Default=/usr/local)**

   Press Enter to accept the default or enter the desired path. The directory must exist on all hosts in the Greenplum Database cluster and must be writable by the gpadmin user. See Selecting and Preparing an Installation Directory for Command Center for more information.

4. **What would you like to name this installation of Greenplum Command Center? (Default=gpcc)**

   Enter a name to display on Command Center web pages to identify this Greenplum Command Center installation.

5. **What port would you like the gpcc webserver to use? (Default=28080)**

   The default Command Center listen port is 28080. Press Enter to accept the default or enter another port number.

6. **Would you like to enable Kerberos?**

   Enter `y` if you want to enable client authentication with Kerberos. Kerberos must already be enabled for Greenplum Database. (If you enter `n`, you can set up Kerberos authentication later using the `gpcc krbenable` command.) The installer prompts for information about your Kerberos installation.

   Choose Kerberos mode (1.normal/2.strict/3.gpmon_only)

   Greenplum Command Center supports three different Kerberos authentication schemes.

   1 - normal mode (default) – The Command Center Kerberos keytab file contains the Command Center principal and may contain principals for Command Center users. If the principal in the client's connection request is in the Command Center's keytab file, Command Center uses the client's principal for database connections. Otherwise, Command Center uses the gpmon user for database connections.
   2 - strict mode – Command Center has a Kerberos keytab file containing the Command Center service principal and a principal for every Command Center user. If the principal in

the client's connection request is in the keytab file, the web server grants the client access and the web server connects to Greenplum Database using the client's principal name. If the principal is not in the keytab file, the connection request fails.
3 - gpmon_only mode – Command Center uses the gpmon database role for all Greenplum Database connections. No client principals are needed in the Command Center's keytab file.

Enter path to the keytab file

Provide the path to the keytab file

Enter the path to the keytab file containing the Kerberos principal for the Command Center web server and, optionally, Command Center user principals.

What is the name of the GPDB Kerberos service?

The Kerberos service name for Greenplum Database is `postgres`.

With Greenplum Database versions earlier than 6.0, a different Kerberos service name can be specified by setting the `krb_srvname` server configuration parameter. You can find the service name for your pre-6.0 Greenplum Database cluster using the `gpconfig` utility:

```
$ gpconfig -s krb_srvname
```

What is the URL of the Command Center web server?

The Kerberos keytab file must contain a principal for the Command Center web server. The principal name is of the format `HTTP/<gpcc-host>@<realm>`, where `<gpcc-host>` is the host name clients use in URLs when connecting to the Command Center web server.

7. **Would you like to enable SSL?**

   Enter `y` if you want to enable SSL/TLS (HTTPS) encryption for client connections to the Command Center web server. The installation utility prompts for the location of the SSL certificate.

   Please enter the full path for the SSL certificate file, including file name

   Enter the path to the combined SSL certificate file installed on the Command Center host. This file contains a certificate and a private key for the web server. The file must be readable by the gpadmin user. See SSL/TLS Encryption for information about creating this file and installing it on your server.

8. **Choose a display language (Default=English)**
   1. English
   2. Chinese
   3. Korean
   4. Russian
   5. Japanese

   Enter a number to choose a language for the Command Center user interface.

The installer saves a log of the installation session in the current directory in a file named `gpccinstall.<timestamp>.log`.

Updating the Metrics Collector Extension

If a new Greenplum Database `metrics_collector` extension is required, the installer displays the following message:

```
*******************************************************************************
*                                                                             *
* INSTALLATION IS ALMOST COMPLETED                                            *
*                                                                             *
* The last step is to update the metrics_collector extension, which needs to  *
* restart the Greenplum Database cluster. Please proceed to $GPCC_HOME and    *
* follow the instructions in this file:                                       *
*                                                                             *
*                           update-extension.txt                              *
*                                                                             *
* Some new features may not be available before the update is done.          *
*                                                                             *
* To use GPCC with the old metrics_collector extension, source gpcc_path.sh   *
* and run 'gpcc start'.                                                       *
*                                                                             *
*******************************************************************************
```

If you are unable to restart Greenplum Database at this time, you can source the `gpcc_path.sh` file in the newly installed Command Center directory and then restart Command Center. New Command Center features may not be available until you update the metrics collector extension and restart Greenplum Database.

Follow these steps to upgrade the `metrics_collector` extension.

1. Source the `gpcc_path.sh` file in the new Command Center installation directory.

   ```
   $ source <install-dir>/greenplum-cc/gpcc_path.sh
   ```

2. Change to the new Command Center home directory.

   ```
   $ cd $GPCC_HOME
   ```

3. View and follow the instructions in the `update-extension.txt` file.

   ```
   $ cat update-extension.txt
   ```

   ```
   Please update metrics_collector extension with the following steps.
   Skip if you just reinstalled GPCC without any changes to GPDB binary.
   For more information please check the documentation.

   ######################################
   #                                    #
   #   I M P O R T A N T  N O T I C E   #
   #                                    #
   ######################################

   When upgrading GPDB to a newer version,
   DO NOT use "gppkg --migrate" to migrate the previous metrics_collector.

   If you have to use "gppkg -- migrate" for some other packages,
   first run "gppkg -r" to remove metrics_collector from the old GPDB
   ```

```
installation.

Before you proceed with updating the metrics_collector extension,
be sure to back up these original files, in case you decide to roll back:

$GPHOME/lib/postgresql/metrics_collector.so
$GPHOME/share/postgresql/extension/metrics_collector*
$GPHOME/share/postgresql/extension/gp_wlm*

1. cd /usr/local/greenplum-cc-4.11.0/gppkg
2. Stop GPCC: gpcc stop
3. Drop extension: psql gpperfmon -c 'drop extension metrics_collector'
4. Use "gppkg -q --all" to list packages installed.
5. Remove the old version of the metrics_collector: "gppkg -r <name>-<version>"
6. Install the new version: gppkg -i MetricsCollector-6.2.0_gp_6.7.0-<OS>-<ARCH
>.gppkg
7. Restart GPDB
8. Restart GPCC: gpcc start
```

Install With a Configuration File

You can provide a configuration file to the Greenplum Command Center installer to perform a non-interactive Command Center installation.

```
$ cd greenplum-cc-<version>
$ ./gpccinstall-<version> -c <config-file>
```

The following table contains the names of the parameters corresponding to the interactive installation prompts and their default values. Define parameters in a configuration file for any parameters that have no default value or to override default values.

Installer Prompt	Default	Parameter
Where would you like to install Greenplum Command Center?	/usr/local	path
What would you like to name this installation of Greenplum Command Center?	gpcc	display_name
On which port would you like to install the Greenplum Command Center web server?	28080	web_port
Would you like to enable SSL?	false	enable_ssl
Please provide the file path for the SSL certificate:	none	ssl_cert_file
Would you like to enable Kerberos?	false	enable_kerberos
Choose Kerberos mode (1.normal/2.strict/3.gpmon_only):	1	krb_mode
Please provide the path to the keytab file:		keytab
What is the name of the GPDB Kerberos service?	postgres	krb_service_name
What is the URL of the Command Center web server?	gpcc	webserver_url
Would you like to enable SSL?	true	enable_ssl
Please provide the file path for the SSL certificate:	/etc/ssl/certs/cert.pem	ssl_cert_file

Installer Prompt	Default	Parameter
Please choose a display language (1.English2.Chinese/3.Korean/4.Russian/5.Japanese)	1	language

If the `enable_kerberos` parameter is true, the `keytab`, `webserver_url`, `krb_mode`, and `krb_service_name` parameters must also be set.

If the `enable_ssl` parameter is true, the `ssl_cert_file` parameter is required.

The following installation configuration file example sets all parameters to their default values.

```
path = /usr/local
# Set the display_name param to the string to display in the GPCC UI.
# The default is "gpcc"
# display_name = gpcc

master_port = 5432
web_port = 28080
rpc_port = 8899
enable_ssl = false
# Uncomment and set the ssl_cert_file if you set enable_ssl to true.
# ssl_cert_file = /etc/certs/mycert
enable_kerberos = false
# Uncomment and set the following parameters if you set enable_kerberos to true.
# webserver_url = <webserver_service_url>
# krb_mode = 1
# keytab = <path_to_keytab>
# krb_service_name = postgres
# User interface language: 1 = English, 2 = Chinese, 3 = Korean, 4 = Russian, 5 = Japa
nese
language = 1
```

Non-Interactive Installation with Defaults

The non-interative installation is useful when installing Command Center in a cloud environment.

1. Source the `greenplum_path.sh` script in the Greenplum Database installation directory to ensure the `GPHOME` environment variable is set.

   ```
   $ source /usr/local/greenplum-db/greenplum_path.sh
   ```

2. Run the Greenplum Command Center installer with the `-auto` option.

   ```
   $ cd greenplum-cc-<version>
   $ ./gpccinstall-<version> -auto
   ```

Note: When you run the installer with the `-auto` option, the Command Center installer uses HTTP as the default protocol for the Command Center web server. To override this, you must use the `-c` option to pass in a configuration file that has `enable_ssl` set to true and `ssl_cert_file` set to the cert file path.

Upgrade

Running `gpccinstall` with the `-u` option installs a Greenplum Command Center release using the configuration parameters from the Command Center installation. You can install a new Command Center release, or reinstall the current release. This option is also useful after you have added new hosts to the Greenplum Database cluster or replaced failed hosts.

The configuration parameters are read from the `$GPCC_HOME/conf/app.conf` file.

1. Source the `greenplum_path.sh` script in the Greenplum Database installation directory to ensure the `GPHOME` environment variable is set.

   ```
   $ source /usr/local/greenplum-db/greenplum_path.sh
   ```

2. Source the `gpcc_path.sh` script in the Greenplum Command Center installation directory.

   ```
   $ source /usr/local/greenplum-cc-<version>/gpcc_path.sh
   ```

3. Run the Greenplum Command Center installer with the `-u` option.

   ```
   $ cd greenplum-cc-<version>
   $ ./gpccinstall-<version> -u
   ```

Set Up Command Center and Workload Management Extensions

You must follow the steps in this section only if you have upgraded your VMware Tanzu Greenplum Database system from a 5.x release earlier than 5.7.0.

The Greenplum Database metrics collection and workload management extensions are installed when you upgrade to VMware Tanzu Greenplum Database 5.7.0 or later. However, the upgrade procedure preserves your previous `postgresql.conf` configuration file, so you must manually set the server configuration parameters that enable the extensions. You must restart Greenplum Database if you change any configuration parameters.

To set up the Command Center and workload management extensions, log in to the master host as gpadmin and follow these steps.

1. Add the metrics collector and workload management shared libraries to the `shared_preload_libraries` configuration parameter.

   Check the current value of the `shared_preload_libraries` configuration parameter.

   ```
   $ gpconfig -s shared_preload_libraries
   Values on all segments are consistent
   GUC          : shared_preload_libraries
   Master  value:
   Segment value:
   ```

   Add the Command Center and workload management libraries to the parameter. (If there were existing libraries in the parameter, append the new libraries, separated with a comma.)

   ```
   $ gpconfig -c shared_preload_libraries -v '\$libdir/metrics_collector,\$libdir/
   gp_wlm'
   ```

2. Make sure the `gp_enable_query_metrics` configuration parameter is on.

```
gpconfig -s gp_enable_query_metrics
gpconfig -c gp_enable_query_metrics -v on
```

3. If you changed any configuration parameters, restart Greenplum Database.

```
gpstop -r
```

Installing Multiple Command Center Instances

You can install a Command Center instance for each Greenplum Database system installed on your cluster. This feature requires Greenplum Database 5.28.0 or above.

Follow these steps when installing an additional Command Center instance.

1. Source the `greenplum_path.sh` file in the target Greenplum Database system before you start the Command Center installer.
2. Run the latest Greenplum Command Center installer for the target Greenplum Database system.
    - Choose an installation path that is different from any existing Command Center instances.
    - Choose a name for the Command Center installation that is different from any existing Command Center instances.
    - Choose a Command Center web server port number that is different from any existing Command Center instances.
3. When the installation has finished source the `gpcc_path.sh` file in the new Command Center installation directory.
4. Edit the `$GPCC_HOME/conf/app.conf` file.
    - Change the `rpc_port` parameter from the default value, 8899, to a different, available port number.
    - Change the `ws_perf_port` parameter from the default value, 6162, to a different, available port number.
    - Change the `agent_perf_port` parameter from the default value, 6163, to a different, available port number.

Before you start or stop a Command Center instance, be sure to source the `greenplum_path.sh` file in the Greenplum Database installation directory and the `gpcc_path.sh` file in the corresponding Command Center installation directory. Then you can run the `gpcc start`, `gpcc stop`, or other `gpcc` commands.

Prepare the Standby Master Host

After the Command Center software is installed, you start the Command Center web server and backend on the master host or the standby master host. Running Command Center on the master host is recommended, but preparing the standby host ensures the standby can run Command Center if the master fails.

1. Copy the `.pgpass` file from the master host to the standby master host. Run these commands from the master:

   ```
   $ ssh gpadmin@<standby_host>
   $ scp gpadmin@<master_host>:~/.pgpass ~
   $ chmod 600 ~/.pgpass
   ```

   Note: There are two alternative methods you can use to supply the gpmon user's password so that you do not have to put the `.pgpass` file on the host. Command Center only requires the gpmon password when you run `gpcc start`, `gpcc stop`, or `gpcc status`.

   1. Set the `PGPASSWORD` environment variable before you run `gpcc` commands. Example:

      ```
      $ PGPASSWORD=changeme gpcc status
      ```

   2. Add the `-W` option to `gpcc` commands to have the command prompt you for the password. Example:

      ```
      $ gpcc start -W
      ```

2. If the Greenplum Command Center web server is to support TLS/SSL, a server certificate must be obtained and installed on the Command Center host in a location readable by the gpadmin user.

3. If Greenplum Command Center is to support Kerberos authentication, Greenplum Database must have Kerberos authentication set up and the required principals and keytabs must be installed on the Command Center host. See Enabling Authentication with Kerberos for Kerberos setup instructions. You can install Command Center without Kerberos authentication initially and then enable Kerberos later by running the `gpcc krbenable` command.

Next Steps

- Setting the Greenplum Command Center Environment
- Starting and Stopping Greenplum Command Center
- Connecting to Greenplum Command Center

Securing the gpmon Database User

The Greenplum Database gpmon role is a superuser role used to manage the gpperfmon database and to retrieve other information from Greenplum databases used in the Command Center interface. The Command Center installer creates the gpperfmon database and the gpmon database role if they do not already exist.

Authentication for the gpmon Role

If the `$MASTER_DATA_DIRECTORY/pg_hba.conf` authentication file contains no entries for the gpmon role, the installer adds these entries:

```
local       gpperfmon     gpmon                     md5
host        all           gpmon   127.0.0.1/28      md5
host        all           gpmon   ::1/128           md5
host        all           gpmon   samenet           md5
```

If the `pg_hba.conf` file already has one or more gpmon entries, the installer will add the `samenet` entry if it is not already present. If `samenet` does not work with your network configuration, you must manually edit the `pg_hba.conf` file to allow gpmon access from all hosts in the cluster.

The Command Center Table Browser requires the `all` entry in the database field to allow gpmon to retrieve table metadata and statistics from each database.

If you use a method other than `md5` to authenticate the gpmon role, such as LDAP or Kerberos, edit the `pg_hba.conf` file manually to enable gpmon connections from all hosts and access to all databases.

If any of these entries are missing from `pg_hba.conf` Command Center may fail to start, or some features may not work properly.

Administrators can view and update the `pg_hba.conf` file in Command Center on the **Admin> Authentication** page. If you update the `pg_hba.conf` outside of Command Center, be sure to use the `gpstop -u` command to have Greenplum Database reload the file.

```
$ gpstop -u
```

Setting the gpmon Password

If you run the `gpccinstall` command with the `-W` option, it prompts you to enter a password for the gpmon role. If you do not supply the `-W` option, `gpccinstall` uses a default password. It adds the password to the `.pgpass` file in the gpadmin user's home directory. The entry in the `.pgpass` file is similar to the following:

```
*:5432:gpperfmon:gpmon:changeme
```

See The Password File in the PostgreSQL documentation for details about the `.pgpass` file.

The `.pgpass` file is required on the Greenplum Database master host to start the gpperfmon data collection agents. If you run Greenplum Command Center on the standby master host, you can copy the `.pgpass` file to that host, or you can run the Command Center `gpcc` management utility with the `-W` option to request password entry each time you start or stop Command Center or request status.

To change the gpmon password, follow these steps:

1. Log in to Greenplum Database as a superuser and change the gpmon password with the `ALTER ROLE` command:

   ```
   # ALTER ROLE gpmon WITH ENCRYPTED PASSWORD 'new_password';
   ```

2. On the Greenplum master host, update the password in the `.pgpass` file in the gpadmin home directory (`~/.pgpass`). Replace the existing password in the line or lines for gpmon with the new password.

```
*:5432:gpperfmon:gpmon:new_password
```

3. Ensure that the `.pgpass` file is owned by gpadmin and RW-accessible by gpadmin only.

   ```
   $ chown gpadmin:gpadmin ~/.pgpass
   $ chmod 600 ~/.pgpass
   ```

4. Restart Greenplum Command Center with the `gpcc` utility.

   ```
   $ gpcc stop
   $ gpcc start
   ```

Note: Be sure to also update the `.pgpass` file on the standby master host.

Authenticating gpmon with Kerberos

If you authenticate Greenplum Database and Command Center users with Kerberos, you can also authenticate the gpmon user with Kerberos.

To prepare for installing Command Center with Kerberos authentication, follow these steps:

1. Create the gpperfmon database using the Greenplum Database `gpperfmon-install` management utility.
2. On the KDC, create a keytab file containing the Kerberos principal for the gpmon user, just as you would for any Kerberos-authenticated client. Install the file on the Greenplum master and standby hosts.
3. Update the entries for gpmon in the `$MASTER_DATA_DIRECTORY/pg_hba.conf` file to use the `gss` authentication method.

   ```
   host all gpmon 0.0.0.0/0 gss include_realm=0 krb_realm=GPDB.EXAMPLE.COM
   ```

   Note that `local` entries in `pg_hba.conf` cannot be authenticated with Kerberos. If there is a `local` entry for the gpmon user, it will use the `.pgpass` file to authenticate with the database. See The pg_hba.conf file in the PostgreSQL documentation for complete `pg_hba.conf` file documentation.

4. Log in to the master host as gpadmin and authenticate the gpmon user.

   ```
   $ kinit gpmon
   ```

5. Install Greenplum Command Center to set up the Kerberos-enabled Command Center.

Setting the Greenplum Command Center Environment

To enable the gpadmin user to execute Command Center utilities such as `gpcc` at the command line, source the `gpcc_path.sh` file in the Greenplum Command Center installation directory. For example:

```
$ source /usr/local/greenplum-cc-<version>/gpcc_path.sh
```

The `gpcc_path.sh` script sets the `GPCC_HOME` environment variable to the Command Center installation directory and adds the `$GPCC_HOME/bin` directory to the path.

To automatically source the `gpcc_path.sh` each time you log in, add the above source command to your start-up script, for example `~/.bashrc` or `~/.bash_profile`.

Upgrading Greenplum Command Center

To upgrade Greenplum Command Center, you install the new Command Center software release, stop the old version, and start the new version. You can then remove the older Command Center release from your Greenplum Database hosts.

Notes: Upgrading From Greenplum Command Center 3.x to 4.x

The Greenplum Command Center architecture changed between Command Center 3.x and 4.x.

With Command Center 3.x, you installed the Greenplum Command Center software one time on the Command Center host. You then created a Command Center instance for each Greenplum Database cluster you monitored with Command Center.

Command Center 4.x does not have instances; you install the Command Center software on the master or standby master of the Greenplum Database cluster you want to monitor with Command Center. The installer copies the software to every host in the Greenplum Database cluster. To monitor additional Greenplum Database clusters you must install the Command Center software again, on a different master or standby host. Monitoring multiple Greenplum Database clusters running on the same hardware cluster is not supported.

To upgrade to a new release of Greenplum Command Center 4.x:

1. Download and install the new Command Center release by following the instructions in Installing Greenplum Command Center.
2. Stop the current Command Center release.

   Command Center 3.x:

   ```
   $ gpcmdr --stop <instance_name>
   ```

   Command Center 4.x:

   ```
   $ gpcc stop
   ```
3. Source the `gpcc_path.sh` script in the new Command Center installation directory. This example uses the symbolic link created by the installer to access the script.

   ```
   $ source /usr/local/greenplum-cc/gpcc_path.sh
   ```

   Note: Also update the source command in your shell start-up script, for example `~/.bashrc.sh` or `~/.bash_profile.sh`.
4. Start the new Command Center release.

   ```
   $ gpcc start
   ```
5. Uninstall the older Command Center release. See "Uninstalling Greenplum Command Center" in the Greenplum Command Center documentation for the release you are uninstalling.

Uninstalling Greenplum Command Center

To uninstall Greenplum Command Center, you must stop both the Command Center Console and disable the data collection agents. Optionally, you may also remove any data associated with Greenplum Command Center by removing your Command Center Console installation and the gpperfmon database.

1. Stop Command Center Console if it is currently running. For example:

   ```
   $ gpcc --stop
   ```

2. Remove the Command Center installation directory from all hosts. For example:

   ```
   $ rm /usr/local/greenplum-cc  ## removes the symbolic link
   $ rm -rf /usr/local/greenplum-cc-version
   ```

3. Disable the data collection agents.

   1. Log in to the master host as the Greenplum administrative user (gpadmin):

      ```
      $ su - gpadmin
      ```

   2. Disable the data colleciton agents by setting the `gp_enable_gpperfmon` server configuration parameter off:

      ```
      $ gpconfig -c gp_enable_gpperfmon -v off
      ```

   3. Remove or comment out the gpmon entries in `pg_hba.conf`. For example:

      ```
      #local     gpperfmon     gpmon      md5
      #host      gpperfmon     gpmon     0.0.0.0/0    md5
      ```

   4. Drop the Command Center superuser role from the database. For example:

      ```
      $ psql template1 -c 'DROP ROLE gpmon;'
      ```

   5. Restart Greenplum Database:

      ```
      $ gpstop -r
      ```

   6. Clean up any uncommitted Command Center data and log files that reside on the master file system:

      ```
      $ rm -rf $MASTER_DATA_DIRECTORY/gpperfmon/data/*
      $ rm -rf $MASTER_DATA_DIRECTORY/gpperfmon/logs/*
      ```

   7. If you do not want to keep your historical Command Center data, drop the gpperfmon database:

      ```
      $ dropdb gpperfmon
      ```

About the Command Center Installation

The installation procedure creates a software installation directory for Greenplum Command Center. This directory is copied to all hosts in the Greenplum Cluster. Versions of Greenplum Database that are compatible with Greenplum Command Center include pre-packaged files that support the Command Center real-time metrics and workload management features.

Software Installation Directory

The following files and first-level subdirectories are copied into the installation directory you specify when you install Greenplum Command Center. This location can be referenced with the `$GPCC_HOME` environment variable when you have set the Command Center environment.

- `gpcc_path.sh` – file containing environment variables for Greenplum Command Center
- `bin/` – program files for Greenplum Command Center
    - `gpcc-agent` - real-time query metrics collection agent
    - `gpccws` - the Greenplum Command Center web server
    - `static/` - static files for the Command Center application
- `ccdata/` - temporary work files
- `conf/`
    - `app.conf` - configuration file for the Command Center web server
- `logs/` - log files for each of the Command Center components
- `open_source_licenses_GPCC.txt` – licenses for open source components used by Greenplum Command Center

Using an HTTP Proxy Server with Command Center

Using an HTTP proxy between client browsers connecting to Greenplum Command Center, or between Command Center agents on Greenplum hosts and the Command Center backend, can cause problems updating data in the Command Center Query Monitor. This topic explains how to configure client connections and the Command Center agents to prevent this problem.

Client Browser Configuration

With the exception of the Query Monitor, Command Center supports browser access through an HTTP proxy. The Query Monitor, however, uses websocket connections on the front end. The websocket connection will fail if the user's browser is configured to access the Command Center server through an HTTP proxy. Command Center will be unable to update data on the Query Monitor, and the display will be empty with no reported error.

To enable the Query Monitor to work with a client browser configured through an HTTP proxy, either the Command Center hostname must be added to the proxy's exception list, or the HTTP proxy must be disabled for the Command Center browser session.

On some platforms, the proxy configuration is accessed through the browser's settings. On others, MacOS for example, the proxy is configured in the operating system network preferences. See the documentation for your web browser for help configuring the HTTP proxy.

Command Center Backend Configuration

On the backend, `ccagent` agent processes running on the Greenplum master and segment hosts connect to the Command Center backend. The agents gather the data, insert it into the database and send the information to the Command Center backend through an HTTP connection. If the `ccagent` process is started in a shell session that has `http_proxy` or `https_proxy` environment variables set, the `ccagent` connection is redirected to the specified proxy server instead of the Greenplum master host. This prevents the agent from delivering the data that updates the Query Monitor display, and the Query Monitor will be empty.

To prevent the proxy settings from interfering with Command Center backend connections, you can disable the proxy environment variables using one of these methods:

- Remove any `http_proxy` or `https_proxy` environment variable settings from shell startup scripts on the master and all segment hosts. For example, delete commands like the following from `/etc/profile`, `/home/gpadmin/.bashrc`, `/home/gpadmin/.bash_profile`, and any other scripts executed for the gpadmin user.

  ```
  export http_proxy=http://<proxy-server>:<port>/
  ```

```
export https_proxy=https://<proxy-server>:<port>/
```

- Unset the `http_proxy` or `https_proxy` environment variables in the `$GPCC_HOME/bin/start_agent.sh` files on the master and all segment hosts.

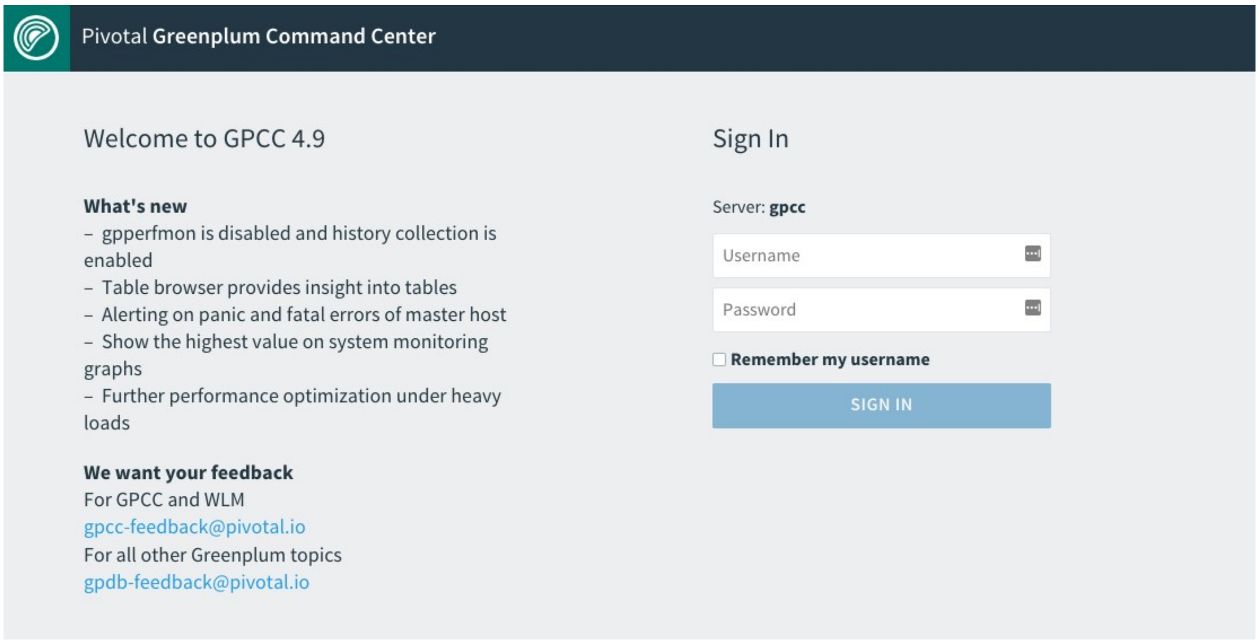
  ```
  unset http_proxy
  unset https_proxy
  ```

Connecting to the Greenplum Command Center Console

Sign in to the Command Center Console with a name and password. If the Guest Access to Query Monitor feature is enabled, you can sign in anonymously to see just the **Query Monitor** view.

Open the Command Center Console in a supported browser using the host name and port configured for the Command Center web server. For example, to open a secure Command Center connection on a host named `smdw` at port 28080, enter this URL into your browser:

```
https://smdw:28080
```



Pivotal Greenplum Command Center

Welcome to GPCC 4.9

What's new
– gpperfmon is disabled and history collection is enabled
– Table browser provides insight into tables
– Alerting on panic and fatal errors of master host
– Show the highest value on system monitoring graphs
– Further performance optimization under heavy loads

We want your feedback
For GPCC and WLM
gpcc-feedback@pivotal.io
For all other Greenplum topics
gpdb-feedback@pivotal.io

Sign In

Server: **gpcc**

- If the **View Query Monitor** link is present, you can click it to view the **Query Monitor** page without signing in. This takes you immediately to the Query Monitor view. To access additional Command Center features, click **Sign In** on the **Query Monitor** view and sign in with a valid Command Center user name and password. If the link is not present on the sign-in page, a Command Center administrator has disabled the anonymous query monitor feature.
- To sign in as a Command Center user, enter the user name and password of a Greenplum role that has been configured to allow authentication to Greenplum Command Center, then click **Sign In**. This opens the **Dashboard** page of the Command Center Console, which provides a graphical system snapshot and a summary view of active queries. See the Dashboard for information about the Dashboard view.

Note to Chrome Browser Users

If you install a new version of Greenplum Command Center using the same port number as the previous version, and you use the Chrome web browser, you may be unable to view real-time queries until after you clear the browser's cache. Follow these steps.

1. Choose **Settings** from the Chrome menu.
2. Scroll to the bottom and click **Advanced**.
3. Under **Privacy and security**, click **Clear browsing data**.
4. Click the **Basic** tab and select **Cached images and files**. You do not have to clear **Browsing history** or **Cookies and other site data**.
5. Click **CLEAR DATA** and then log in to Command Center.

Administering the Command Center Web Server

The gpccws web server binary and web application files are installed in the `bin` directory of your Greenplum Command Center installation.

Starting and Stopping the Web Server

Starting the Command Center Web Server runs the gpccws web server, starts the metrics collection agents on the segment servers, and starts a listener on the Command Center rpc port.

You can run the `gpcc` command as the gpadmin user on the master host (recommended) or on the standby host. Starting Command Center on the standby host is not recommended because it can cause heavy network traffic between the Command Center agent on the master host and the backend on the standby host.

To ensure the `gpcc` command is on your path, source the `gpcc_path.sh` file in the Command Center installation directory or add it to the startup script for your command shell. See Setting the Greenplum Command Center Environment for instructions. The `MASTER_DATA_DIRECTORY` environment variable must be set to the location of the Greenplum Database master data directory.

Note: The `gpcc` command uses the gpmon role to connect to Greenplum Database. It looks for the gpmon password in the `PGPASSWORD` environment variable or in the `.pgpass` file in the gpadmin user's home directory. You can instead append the `-W` flag to the `gpcc` commands below to have `gpcc` prompt you to enter the password.

To start Greenplum Command Center

Log on to the master host or the standby host.

To log on to the standby from the master host:

```
$ ssh <standby-host>
```

Source the Command Center environmental script.

```
$ source /usr/local/greenplum-cc-<version>/gpcc_path.sh
```

Start the Command Center web server and the metrics collection agents.

```
$ gpcc start
Starting the gpcc agents and webserver…
2019/05/07 01:51:03 Agent successfully started on 5/5 hosts
2019/05/07 01:51:03 View Greenplum Command Center at http://mdw:28090
```

To stop Greenplum Command Center

```
$ gpcc stop
2019/05/07 01:51:55 GPCC webserver and metrics collection agents have been stopped. Us
e gpcc start to start them again
```

To check the Greenplum Command Center status

```
$ gpcc status
2019/05/07 01:50:13 GPCC webserver: running
2019/05/07 01:50:14 GPCC agents: 5/5 agents running
```

See the `gpcc` reference page for full syntax for the `gpcc` command.

Configuring the Command Center Web Server

The web server configuration file is stored in `$GPCC_HOME/conf/app.conf`. The parameters in this configuration file are set when you install Greenplum Command Center. The installer copies the Command Center installation directory, including this configuration file, to every Greenplum Database host.

See the *Web Server Parameters* section of Configuration File Reference for a description of the parameters in this file.

You can see a summary of the current configuration using the `gpcc --settings` command.

```
$ gpcc --settings
Install path:                /usr/local
Display Name:                gpcc
GPCC port:                   28080
Kerberos:                    enabled
Kerberos webserver name:     mdw
Kerberos gpdb name:          gpadmin
Kerberos mode:               gpmon_only
Kerberos path:               /home/gpadmin/gpcc-kerberos.keytab
SSL:                         disabled
```

If you modify the file on one host you should copy it to every other host. Be sure to restart the web server after you change the configuration. Rather than modifying the configuration directly, you can just stop Command Center and re-run the `gpccinstall-<version>` installation command. This ensures the configuration is consistent on all hosts.

You can use the `gpcc krbenable` command to add Kerberos authentication to the Command Center configuration. See Enabling Authentication with Kerberos for details about setting up Kerberos on the Command Center host. The `gpcc krbenable` command prompts for the Kerberos principal names and artifacts and updates the configuration.

The `gpcc krbdisable` command removes Kerberos parameters from the Command Center configuration.

Restarting Command Center After Master Failure

The Greenplum Database standby master is a *warm* standby—if the primary master fails, the administrator must activate the standby master by running the `gpactivatestandby` management utility.

After Greenplum Database has restarted on the standby master, you can start Command Center on the standby master.

Follow these steps to activate Greenplum Command Center on the standby master host.

1. On the standby master, edit the `app.conf` file in the `$GPCC_HOME` directory.
    - Change the `master_host` and `master_port` parameters to the standby master host name and port number.
    - Change the `httpport` and `httpsport` parameters to the Command Center web server port number.
2. Edit the `pg_hba.conf` file on the standby master to allow the `gpmon` user to access all databases and then run `gpstop -u` on the standby host to reload the configuration files.
3. Copy the `$MASTER_DATA_DIRECTORY/gpmetrics` directory from the old master host to the standby master host.
4. Copy the `~gpadmin/.pgpass` file from the master to the standby master and make sure the port number is correct in the entry for the `gpmon` user.
5. Restart Command Center on the standby master.

    ```
    $ source <command-center-install-dir>/gpcc_path.sh
    $ gpcc start
    ```

If you perform the initial four steps ahead of time and keep the standby master configuration files up to date when configurations change on the primary master, you can just restart Command Center on the standby master host after Greenplum Database has restarted on the standby master.

Viewing and Maintaining Web Server Log Files

Web server access and error log messages are written to `$GPCC_HOME/logs/gpccws.log`.

If you experience errors viewing the Greenplum Command Center Console, refer to this file for more information.

To prevent the web server log from growing to excessive size, you can set up log file rotation using `logrotate` or `cronolog`.

Administering Command Center Agents

The Greenplum Command Center metrics collection agent, `ccagent`, runs on segment hosts and receives real-time metrics emitted by the metrics collection database extension. Each segment host has one `ccagent` process. The metrics collection extension connects to `ccagent` using Unix Domain Sockets (UDS) to transfer metrics from Greenplum Database. Starting Greenplum Command Center with the `gpcc start` command starts the Command Center agent on each segment host. Stopping Command Center with `gpcc stop` ends these processes. The Command Center backend monitors these agents and restarts them when necessary.

If you expand the Greenplum Database cluster with new segment hosts or replace segment hosts, you must reinstall Command Center so that the agent is installed onto the new hosts.

See Managing Log Files and Data Files for information about managing Command Center agent log files.

Note: For information about the gpperfmon service agents `gpmmon` and `gpsmon`, see The gpperfmon Database.

Settings

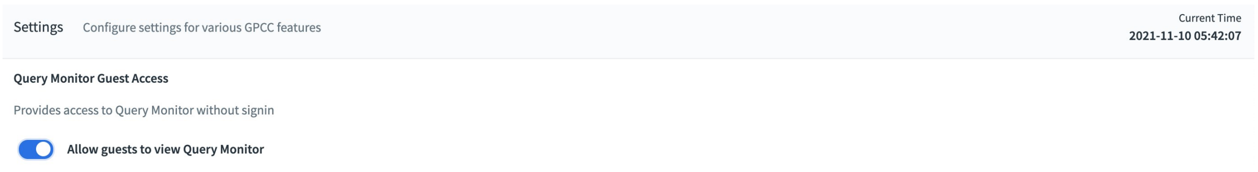
The **Admin> Settings** view enables Command Center administrators to configure settings for Command Center features.

Query Monitor guest access settings

Turn on **Allow guests to view Query Monitor** to allow users access to the **Query Monitor** without logging in. When enabled, the **Allow guests to view Query Monitor** feature allows anyone with access to the Greenplum Command Center web server to click **View Query Monitor** on the Command Center sign-in screen and see the **Query Monitor** page without logging in. These anonymous users cannot cancel queries and cannot access any other Command Center features.

When this feature is off, the **View Query Monitor** link does not appear on the sign-in screen and anonymous users cannot see the **Query Monitor** page.

Command Center users with Admin permission can toggle the feature on an off with a mouse click. Users with Operator or Operator Basic permission see a message reporting if the feature is on or off.


Settings Configure settings for various GPCC features

Current Time
2021-11-10 05:42:07

Query Monitor Guest Access

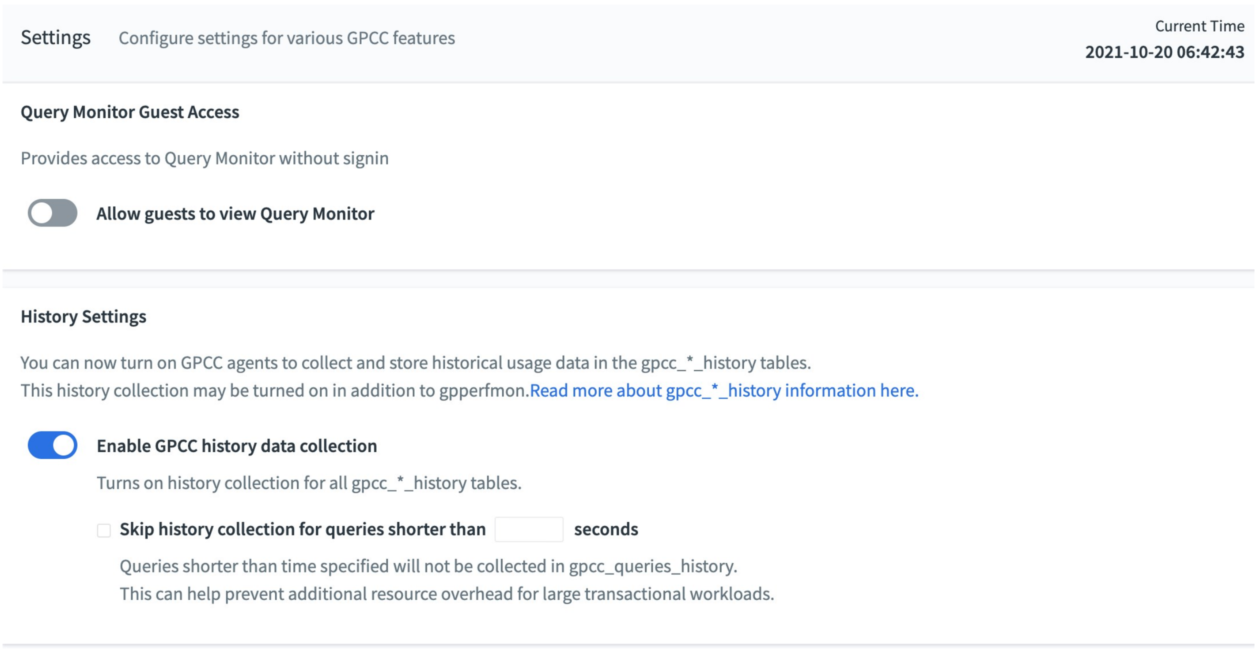
Provides access to Query Monitor without signin

Allow guests to view Query Monitor

History settings

Turn on **Enable GPCC history data collection** to save query history, host metrics, and disk usage metrics to tables in the gpperfmon database gmetrics schema.

Enter a number of seconds to set the minimum runtime for a query to be saved in history. The default is to save all queries. Set this threshold to prevent Command Center from filling history with trivial queries.


Settings Configure settings for various GPCC features

Current Time
2021-10-20 06:42:43

Query Monitor Guest Access

Provides access to Query Monitor without signin

Allow guests to view Query Monitor

History Settings

You can now turn on GPCC agents to collect and store historical usage data in the gpcc_*_history tables.
This history collection may be turned on in addition to gpperfmon.Read more about gpcc_*_history information here.

Enable GPCC history data collection

Turns on history collection for all gpcc_*_history tables.

Skip history collection for queries shorter than seconds

Queries shorter than time specified will not be collected in gpcc_queries_history.
This can help prevent additional resource overhead for large transactional workloads.

This option is enabled by default.

When **GPCC history data collection** is enabled:

- Command Center saves query and metrics history in the gpmetrics schema tables in the gpperfmon database. Only queries that execute for at least the number of seconds you specify are saved. Query plan node history is only saved for queries that run for at least 10 seconds, or the number of seconds you specify, if greater than 10.
- The Command Center **History** view displays saved history and metrics from the gpmetrics tables. See gpmetrics Schema Reference for information about the history tables.
- For best system performance, disable the Greenplum Database gpperfmon service if it is running. To disable the service, set the Greenplum Database `gp_enable_gpperfmon` server configuration parameter to off and restart Greenplum Database.

When **GPCC history data collection** is disabled:

- The Greenplum Databse gpperfmon service saves query and metrics history to the "legacy" history tables, in the gpperfmon database public schema. To use the gpperfmon service, you must create the gpperfmon database with the Greenplum Database `gpperfmon_install` management utility and enable the service.
- The **Command Center History** view uses history data collected by the Greenplum Database gpperfmon service. See The gpperfmon Database for information about the gpperfmon service and tables.

Administering Log Files and Work Files

The Command Center components—the backend server, agents, HTTP server, and the `gpcc` command-line utility—write messages to log files in the `$GPCC_HOME/logs` directory.

Log files rotate once per day or whenever the component starts or restarts. The current log file is saved to a file with a timestamp in its name and a new log file is started.

The log files can be useful for monitoring and troubleshooting, but you can delete them when you determine they are no longer useful. Remove older log files regularly to conserve disk space.

You can use a utility such as the Linux `logrotate` utility to manage archived log and data files.

webserver.log

The Command Center backend web server writes to the `webserver.log` log file on the Command Center host in the `$GPCC_HOME/logs` directory on the host running Command Center. This is usually the master host, but could also be the standby master host. Messages in this log file describe start-up events, rpc request/response events with agents running on the Greenplum Database segment hosts, and data loading from external tables to the gpmetrics tables in the gpperfmon database.

agent.log

A Command Center agent running on each Greenplum Database host (including the segment and master hosts) logs messages to the `agent.log` log file in the `$GPCC_HOME/logs` directory of the host machine. Agents log messages about connections to the Command Center rpc server, periodic data collection events, and metrics transmissions to the rpc port on the backend.

gpccws.log

The Command Center HTTP server writes messages to the `gpccws.log` file on the Command Center host in the `$GPCC_HOME/logs` directory. The messages in this log file include server start up messages and Websocket connection events with Command Center clients.

cli.log

The `gpcc` command-line utility writes messages to the `cli.log` file in the `$GPCC_HOME/logs` directory. Whenever the `gpcc` utility executes, it writes its output to the log file as well as the standard output. The log file is useful for auditing Command Center start and stop events. The `cli.log` file is not rotated and continues to grow until you archive or delete it manually.

Command Center Work Files

Command Center saves temporary data into files in the `$GPCC_HOME/ccdata` directory. These files include saved query text files (`q*.txt`), plan node files (`gpccexec*.txt`), and table size and statistics information files (`gpcc_size_info.dat` and `gpcc_stat_info.dat`). Each night, Command Center archives files in the `$GPCC_HOME/ccdata` directory that are more than two weeks old. The archive files have names in the format `archive_YYYYMMDD_YYYYMMDD.tar.gz`, where the dates are the beginning and end dates of the week included in the archive. The archived files are no longer needed by Command Center but may be useful for troubleshooting; you can remove them manually if you do not want to save them.

gpcc

Manages the Greenplum Command Center web service and metrics collection agents.

```
gpcc <action> [-W]

gpcc [--version | -v ]

gpcc [--help | -h]

gpcc [--settings]
```

Actions

`start`

Starts the Command Center web service and metrics collection agents. Add the `-W` flag to force a prompt for the gpmon user password.

`stop`

Stops the Command Center web service and metrics collection agents. Add the `-W` flag to force a prompt for the gpmon user password.

`status`

Displays the status, either `Running` or `Stopped`, of the web server and metrics collection agents. Add the `-W` flag to force a prompt for the gpmon user password.

`krbenable`

Enables Kerberos authentication for Command Center.

Use the `gpcc krbenable` command to set up Kerberos authentication for Command Center users if Command Center was initially installed without enabling Kerberos. When you run `gpcc krbenable`, `gpcc` prompts for:

- the web server name
- the name of the Greenplum Database Kerberos service
- the Command Center Kerberos authentication mode
- the path to the keytab file on the Command Center host.

Before you run `gpcc krbenable`, see Enabling Authentication with Kerberos to check prerequisites and for help preparing the Command Center host to allow Kerberos authentication.

`krbdisable`

Disables Kerberos authentication for Command Center.

`help`

Displays syntax and help text for the `gpcc` command.

Options

`--settings`

Displays the current values of the Command Center configuration parameters. See Command Center Console Parameters for a list of the configuration parameters.

`--version` or `-v`

Displays the Greenplum Command Center version.

`-W <password>`

The optional `-W` option specifies the password for the gpmon user. The `gpcc` command normally gets the password from the `$PGPASSWD` environment variable or the `.pgpass` file in the gpadmin user's home directory. If the password is not available with either of these methods, the `-W` option must be included to specify the password whenever you run `gpcc`.

Description

The Greenplum Database `MASTER_DATA_DIRECTORY` environment variable must be set when you run the `gpcc` command. This environment variable is usually set in the gpadmin user's shell initialization script (`/home/gpadmin/.bashrc`, for example). If `MASTER_DATA_DIRECTORY` is not set when you run `gpcc start`, `gpcc` prints an error message and exists.

Once started, the Command Center backend monitors the metrics agents with a heartbeat. If a failed agent is detected, the backend spawns a new agent process.

Examples

Start Command Center and the metrics agents, prompting for the gpmon password.

```
$ gpcc start -W
Password for GPDB user gpmon:
Starting the gpcc agents and webserver…
2018/03/22 17:51:51 Agent successfully started on 7/8 hosts
2018/03/22 17:51:51 View Greenplum Command Center at http://smdw:28080
```

gpmetrics Schema Reference

Greenplum Command Center creates the `gpmetrics` schema in the Greenplum Database gpperfmon to save alert rules and logs, and historical metrics collected by the Greenplum Database metrics collection system. The `gpmetrics` schema contains the following tables and user-defined functions:

Tables

- `gpcc_alert_rule` — saves alert rules configured on the Command Center **Admin> Alerts** page.
- `gpcc_alert_history` — records an event when an alert rule is triggered.
- `gpcc_database_history` — saves summary query activity information.
- `gpcc_department` — saves department information.
- `gpcc_role_department` — saves information about roles within a department.
- `gpcc_disk_history` — saves disk usage statistics for each Greenplum Database host file system.
- `gpcc_export_log` — saves the log notifications from every "EXPORT ALL" user action.
- `gpcc_pg_log_history` — saves history from the Greenplum Database `pg_log` log file.
- `gpcc_plannode_history` — saves plan node execution statistics for completed queries.
- `gpcc_queries_history` — saves execution metrics for completed Greenplum Database queries.
- `gpcc_queries_now` — saves real-time query metrics data.
- `gpcc_resgroup_history` — saves the history of the resource consumption of each resource group on each segment.
- `gpcc_scan_history` — saves history for Recommendations scans.
- `gpcc_schedule` — saves schedule for Recommendations scans.
- `gpcc_system_history` — saves system metrics sampled from Greenplum Database segments hosts.
- `gpcc_table_info` — saves current statistics and size information for tables.
- `gpcc_table_info_history` — saves daily statistics and size information for tables.
- `gpcc_wlm_rule` — saves workload management rules.
- `gpcc_wlm_log_history` — saves the log history of workload management rule actions.

User-Defined Functions

- `gpcc_delete_department` — deletes a department.
- `gpcc_queries_per_hour` — returns a variety of details about query activity per hour.
- `gpcc_queries_per_user` — returns, for each user, the number of queries whose runtime is longer than the input interval, per hour, in the specified time range.
- `gpcc_queries_per_user_max_and_total_spill_size` — returns, for each user, the total `spill_size` and maximum `spill_size` per query per hour.
- `gpcc_queries_per_user_max_cpu` — returns, for each user, the query with the maximum segment and master cpu usage per hour, along with details about the query.
- `gpcc_queries_per_user_max_run_time` — returns, for each user, the longest running query per hour, along with details about the query.
- `gpcc_queries_per_user_max_skew` — returns, for each user, the query with the maximum amount of processing skew in the system (`skew_cpu`) per hour, along with details about the query.
- `gpcc_queries_per_user_rows_out` — returns, for each user, the query with the maximum `rows_out` per hour, along with details about the query.
- `gpcc_system_per_hour` — returns a variety of system information.
- `gpcc_update_department` — renames a department.

If you set the schema search path to include the `gpmetrics` schema, you do not have to qualify table and user-defined function names with the `gpmetrics` schema name. To set the default search path for the gpperfmon database enter this SQL command.

```
=# ALTER DATABASE gpperfmon SET search_path TO public,gpmetrics;
```

You must exit the current session and start a new session for the new search path to take effect.

Alert Tables

Command Center uses the `gpcc_alert_rule` and `gpcc_alert_log` tables to store the alert rules you set up in the user interface and to log messages when the alert rules are triggered.

gpcc_alert_rule

Note: Deprecated in Command Center 6.4/4.12.

The `gpcc_alert_rule` table records the alert rules configured in the Command Center user interface. It has the columns shown in the following table.

Column	Type	Description
rule_id	integer	Unique id for the rule.
rule_type	integer	Reserved for future use.
rule_description	character varying(512)	Text of the rule.
rule_config	json	JSON string containing parameters for user-specified values.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the rule was created.
etime	timestamp(0) without time zone	Time the rule became inactive, or null if rule is active.

The `gpcc_alert_rule` table keeps a history of alert rule configurations. When a rule becomes active, a new row is inserted and the `ctime` timestamp column is set to the current time; the `etime` timestamp is null, indicating that the rule is still active. When a rule is either disabled or superceded by a new rule, the `etime` timestamp column is set to the current time. Thus, the set of currently active rules is all rows where the `etime` column is null. A row that has timestamps in both `ctime` and `etime` columns is an historical record of the period of time during which the rule was active.

The `rule_id` column, a unique integer, is the distribution key for the table and is used to identify a single alert rule configuration. This column can be joined with the `rule_id` column in the `gpcc_alert_log` table to identify the rule that triggered each recorded alert event.

The `rule_description` column contains a string that describes the event that matches the rule. It is the text displayed in the Command Center UI for the rule, with user-specified values inserted.

The `rule_config` column contains a JSON string with parameters for the values entered for each of the rule's fields in the Command Center UI.

gpcc_alert_history

The `gpcc_alert_history` table (was`gpcc_alert_log` before Command Center 6.4) has the following columns:

Column	Type	Description
id	integer	Unique ID for the alert.
rule_id	integer	The ID of the rule that triggered this alert.
transaction_time	timestamp(0) without time zone	Time the alert was raised.
content	json	Contains parameters specifying values that triggered the alert.
config	json	Stores alert history from emails or notifications.

Where:

- The `id` column, a unique integer, is the distribution key for the table.
- The `transaction_time` column is set to the current time when a row is created.
- The `rule_id` column can be joined with the `rule_id` column in the `gpcc_alert_rule` table to access details of the rule that triggered the alert.
- The `content` column contains a JSON string with parameters specifying details about the event that triggered the alert. The JSON parameters vary with the type of the alert.
- The `config` column stores the alert history received from the notification center or from emails.

The `gpcc_alert_history` table is an append-only, column-oriented table, partitioned by month on the `transaction_time` column. Command Center creates new partitions as needed and removes partitions over 12 months old.

A row is added to the `gpcc_alert_history` table whenever an alert rule is matched.

Greenplum Database Metrics History Tables

The `gpmetrics` query history saves information collected by the Greenplum Database metrics collection system and forwarded to Greenplum Command Center.

The distribution key for each table is a `ctime` timestamp column, which is the time when the row is added to the datbase. The tables are partitioned by year and month. Greenplum Command Center creates new partitions automatically as needed.

The history tables use append-optimized, column-oriented storage.

Command Center only saves queries with runtimes greater than the value of the `min_query_time` configuration parameter, found in the `$MASTER_DATA_DIRECTORY/gpmetrics/gpcc.conf` configuration file on the host executing Command Center. The default, 0, saves all queries in the history table. This parameter can be configured on the Command Center **Admin> Settings** page.

gpcc_database_history

The `gpcc_database_history` table saves summary query activity metrics collected by the Greenplum Database metrics collector. This data can be used to review the Greenplum Database query load over time.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the record was created.
queries_total	integer	Total number of queries running and queued to run.
queries_running	integer	Number of queries currently running.
queries_queued	integer	Number of queries queued, but not yet running.
queries_blocked	integer	The number of queries started, but blocked by other transactions.
queries_finished	integer	The number of queries that completed since the previous sampling interval.

gpcc_disk_history

The `gpcc_disk_history` table saves historical disk usage statistics for each Greenplum Database segment host file system.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the row was created.
hostname	character varying(64)	Name of the segment host.
filesystem	text	Path to the segment's data directory.
total_bytes	bigint	Total size of the file system storage in bytes.

Column	Type	Description
bytes_used	bigint	Number of storage bytes in
bytes_available	bigint	Number of storage bytes available.

gpcc_export_log

The `gpcc_export_log` table saves the log notifications from every "EXPORT ALL" user action. Whenever the user exports search results from the History or the Table Browser pages, this table gets updated.

Column	Type	Modifiers
id	integer	not null default nextval('gpcc_export_log_id_seq'::regclass)
ctime	timestamp(0) with time zone	not null default now()
source	character varying(64)	not null
filename	character varying(64)	not null
role	name	not null
req_params	json	not null
etime	timestamp(0) with time zone	
status	character varying(64)	
fail_msg	text	

gpcc_pg_log_history

The `gpcc_pg_log_history` table stores `pg_log` errors and warnings from the Greenplum Server log files.

Column	Type	Description
logtime	timestamp without time zone	Timestamp for this log.
loguser	text	Name of the role executing the query.
logdatabase	text	The database accessed.
logpid	text	Process id.
logthread	text	Thread number.
loghost	text	Host name or IP address of the host.
logport	text	Port number.
logsessiontime	timestamp without time zone	Session timestamp.
logtransaction	integer	Transaction id.
logsession	text	Session id.
logcmdcount	text	Command count.

Column	Type	Description
logsegment	text	Segment number.
logslice	text	Slice number.
logdistxact	text	Distributed transation id.
loglocalxact	text	Location transacton id.
logsubxact	text	Subtransaction id.
logseverity	text	Log severity.
logstate	text	SQL State code associated with this log message.
logmessage	text	Log or error message text.
logdetail	text	Detail message text associated with an error message.
loghint	text	Hint message text associated with an error message.
logquery	text	Internally-generated query text.
logquerypos	integer	Index into the internally-generated query text.
logcontext	text	Context in which this message gets generated.
logdebug	text	Query string with full detail for debugging.
logcursorpos	integer	Cursor index into the query string.
logfunction	text	Function in which this message is generated.
logfile	text	Log file in which this message is generated.
logline	integer	Line in the log file in which this message is generated.
logstack	text	Full text of the stack trace associated with this message.

gpcc_plannode_history

The `gpcc_plannode_history` table saves detailed metrics for each operation (node) in a completed query plan. Each row contains metrics for one operation that executed on one Greenplum Database segment. This information allows reconstructing the plan and execution metrics for a completed query.

Plan node history is only saved for queries that execute for 10 seconds or more.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the row was created.
tmid	integer	A time identifier for the query. All records associated with a query will have the same `tmid`.
ssid	integer	Session id for the database connection. All records associated with the query will have the same `ssid`.

Column	Type	Description
ccnt	integer	Command number within the session. All records associated with the query will have the same `ccnt`.
segid	integer	Id (`dbid`) of the segment for this plan node.
procid	integer	The postgres process ID for this plan node.
sliceid	integer	DEPRECATED. ~~Id of the slice the operation belongs to. Operations that belong to the same slice execute in parallel.~~
nodeid	integer	The query plan node ID for this operation.
parent_nodeid	integer	The parent query plan node ID from the query plan.
node_type	character varying(64)	Name of the operation type.
tinit	timestamp(6) without time zone	Time the operation was initialized.
tstart	timestamp(6) without time zone	Time the operation started.
tfinish	timestamp(6) without time zone	Time the operation finished.
status	character varying(16)	Status of the operation: `initialize`, `executing`, or `finished`.
planrows	double precision	The number of output rows estimated for the operation.
planwidth	integer	Width of output rows estimated for the operation.
start_cost	double precision	Number of page reads expected to produce first output row.
total_cost	double precision	Number of page reads expected to produce all output rows.
tuple_count	bigint	
ntuples	bigint	
nloops	bigint	
first_tuple	timestamp(6) without time zone	Time the operation produced the first output row.
rel_oid	oid	Object ID of the output rows produced by the operation.
relation_name	character varying(64)	Name of the table this operation processed, if any.
index_name	character varying(64)	Name of the index used by this operation, if any.
alias_name	character varying(64)	Alias for the relation declared in the SQL command.
node_seq	integer	Node sequence
condition	text	Condition expression used for a filter or join operation.

gpcc_queries_history

The `gpcc_queries_history` table saves metrics for completed queries.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the row was created.
tmid	integer	A time identifier for the query. All records associated with the query will have the same `tmid`.
ssid	integer	Session id for the database connection. All records associated with the query will have the same `ssid`.
ccnt	integer	Command number within this session. All records associated with the query will have the same `ccnt`.
username	character varying(64)	Role name that issued the query.
db	character varying(64)	Name of the database queried.
cost	double precision	Estimated cost to execute query, computed by the legacy planner or GPORCA.
tsubmit	timestamp(6) without time zone	Time the query was submitted.
tstart	timestamp(6) without time zone	Time the query was started.
tfinish	timestamp(6) without time zone	Time the query finished.
status	character varying(16)	Status of the query – `abort`, `error`, or `done`.
rows_out	bigint	Number of rows returned by the query.
error_msg	text	Error message, if the query failed.
plan_gen	character varying(16)	`PLANNER` if query plan was generated by the legacy planner; `OPTIMIZER` if the plan was generated by GPORCA, the Pivotal query optimizer.
query_hash	character varying(64)	Hash code generated from the text of the query.
query_text	text	Complete text of the query. Some queries may be reformatted before storing in the history database.
application_name	character varying(64)	Name of the client application that established the database connection.
rsqname	character varying(64)	If the `gp_resource_manager` configuration parameter is `queue`, the name of the resource queue managing the query.

Column	Type	Description
rsgname	character varying(64)	If the `gp_resource_manager` configuration parameter is `group`, the name of the resource group managing the query.
cpu_master	bigint	Total CPU usage for this query on the Greenplum Database master instance.
cpu_segs	bigint	Total CPU usage for this query across all segments, measured in seconds. This is the sum of the CPU usage values taken from all active primary segments in the database array.
cpu_master_percent	double precision	Average CPU percent usage on the master host during execution of this query.
cpu_segs_percent	double precision	Average CPU percent usage on the segment hosts during the execution of this query.
skew_cpu	double precision	Displays the amount of processing skew in the system for this query. Processing/CPU skew occurs when one segment performs a disproportionate amount of processing for a query. The skew is calculated from total CPU seconds used on all segments during the execution of the query.
skew_rows	double precision	Displays the amount of row skew in the system. Row skew occurs when one segment produces a disproportionate number of rows for a query.
memory	bigint	Total size of memory, in bytes, used by all segments to execute this query.
disk_read_bytes	bigint	Number of bytes read from disk.
disk_write_bytes	bigint	Number of bytes written to disk.
spill_size	bigint	Total size, in bytes, of spill files used by all segments to execute this query.
rqpriority	character varying(16)	Priority setting for the resource queue managing this query. Blank if resource group management is enabled.
query_tag	text	A key-value pair describing a query.
slices_metrics	JSON	The cpu/memory/disk metrics for a slice.
peak_memory	bigint	Maximum memory usage across all segments during the execution of the query, measured in KB.
node_sliceid	json	A map of node ID to slice ID.

gpcc_resgroup_history

The `gpcc_resgroup_history` table saves the history of the resource consumption of each resource group on each segment.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the row was created.
rsgname	name	Resource group name.
segid	integer	Database ID of the segment for this resource group info.
groupid	oid	Object ID of the resource group.

Column	Type	Description
concurrency_limit	integer	The concurrency (`CONCURRENCY`) value specified for the resource group.
cpu_rate_limit	integer	The CPU limit (`CPU_RATE_LIMIT`) value specified for the resource group, or -1.
num_running	integer	The number of transactions currently executing in the resource group.
num_queueing	integer	The number of currently queued transactions for the resource group.
cpu_usage_percent	double precision	The sum of the percentages of CPU cores that are used by the resource group.
mem_used_mb	integer	Memory used, in MB.
mem_available_mb	integer	Available memory, in MB.
mem_quota_used_mb	integer	Memory quota used, in MB.
mem_quota_available_mb	integer	Available memory quota, in MB.
mem_quota_granted_mb	integer	Granted memory quota, in MB.
mem_quota_proposed_mb	integer	Proposed memory quota, in MB.
mem_shared_used_mb	integer	Shared memory used, in MB.
mem_shared_available_mb	integer	Available shared memory, in MB.
mem_shared_granted_mb	integer	Granted shared memory, in MB.
mem_shared_proposed_mb	integer	Proposed shared memory, in MB.

gpcc_scan_history

Column	Type	Description
ctime	timestamp(0) without time zone	Time the row was created.
scan_id	integer	Scan id for this scan. See `schedule_id` column in the `gpcc_schedule`.
status	integer	Scan completion status. 0=Running, 1=Success, 2=Failed, 3=Cancelled, 4=Expired.
tables_scanned	bigint	The number of tables that were scanned.
start_dt	timestamp(0) without time zone	Time the scan began.
end_dt	timestamp(0) without time zone	Time the scan ended.

gpcc_schedule

The `gpcc_schedule` table contains details for scheduled events, such as the table scan for Recommendations. Each row describes a schedule in a JSON value. The `ctime` and `etime` columns together specify the period of time during which the schedule is enabled.

Column	Type	Description
schedule_id	integer	Id number for this schedule.
description	character varying(512)	Description of schedule.
schedule_config	json	JSON string with schedule details.
ctime	timestamp(0) without time zone	Time the schedule was added and enabled.
etime	timestamp(0) without time zone	Time the schedule was disabled.

gpcc_system_history

The `gpcc_system_history` table saves historical system metrics for each Greenplum Database host, including the master, standby master, and segment hosts. The metrics include information about memory, CPU, disk, and network utilitization.

Column	Type	Description
cpu_iowait	double precision	The percentage of CPU used waiting on IO requests.
ctime	timestamp(0) without time zone	Time the row was created.
hostname	character varying(64)	Segment or master hostname associated with these system metrics.
mem_total	bigint	Total system memory in Bytes for this host.
mem_used	bigint	System memory used, in Bytes, for this host.
mem_actual_used	bigint	Actual memory used, in Bytes, for this host (not including the memory reserved for cache and buffers).
mem_actual_free	bigint	Free actual memory, in Bytes, for this host (not including the memory reserved for cache and buffers).
swap_total	bigint	Total swap space in Bytes for this host.
swap_used	bigint	Swap space used, in Bytes, for this host.
swap_page_in	bigint	Number of swap pages in.
swap_page_out	bigint	Number of swap pages out.
cpu_user	double precision	Percentage of time CPU processes execute in user mode.
cpu_sys	double precision	Percentage of time CPU processes execute in system (kernel) mode.
cpu_idle	double precision	Percentage idle CPU.
load0	double precision	CPU one-minute load average.
load1	double precision	CPU five-minute load average.
load2	double precision	CPU fifteen-minute load average.
quantum	seconds	Interval between metrics collections.
disk_ro_rate	bigint	Disk read operations per second.

Column	Type	Description
disk_wo_rate	bigint	Disk write operations per second.
disk_rb_rate	bigint	Bytes per second for disk read operations.
disk_wb_rate	bigint	Bytes per second for disk write operations.
net_rp_rate	bigint	Packets per second on the system network for read operations.
net_wp_rate	bigint	Packets per second on the system network for write operations.
net_rb_rate	bigint	Bytes per second on the system network for read operations.
net_wb_rate	bigint	Bytes per second on the system network for write operations.

gpcc_table_info

The `gpcc_table_info` table stores a current snapshot of statistics for tables. There is one row for each table and partition.

Column	Type	Description
ctime	timestamp without time zone	Time the row was created.
dbid	oid	Object ID of the database.
relid	oid	Object ID of the table.
paroid	oid	Object ID of a partition.
reltablespace	oid	Object ID of the table's tablespace.
seq_scan	bigint	Number of sequential scans initiated on this table.
idx_scan	bigint	Number of index scans initiated on this table.
n_tup_ins	bigint	Number of rows inserted.
n_tup_del	bigint	Number of rows deleted.
n_tup_upd	bigint	Number of rows updated (includes hot updated rows).
n_tup_hot_upd	bigint	Number of rows HOT updated (no separate index update required).
last_seq_scan	timestamp with time zone	Time of the last sequential scan on this table. An `ANALYZE` on a table can cause a sequential scan and update the count in the `seq_scan` column. However, the time of that scan is not saved in this column.
last_idx_scan	timestamp with time zone	Time of the last index scan on this table.
last_ins	timestamp with time zone	Time of the last insert on this table.

Column	Type	Description
last_del	timestamp with time zone	Time of the last delete on this table.
last_upd	timestamp with time zone	Time of the last update on this table.
last_analyze	timestamp with time zone	Time of the last analyze on this table.
last_vacuum	timestamp with time zone	Time of the last vacuum of this table (excludes vacuum full).
last_vacuum_full	timestamp with time zone	Time of the last vacuum full of this table.
size	bigint	Combined size of all of this table's files.
row_cnt	real	Number of rows in the table.
children	integer	Number of partitions, including middle-level and child partitions.
schema	name	Name of the schema this table belongs to.
table_name	name	Name of the table.
owner	name	Name of the database role that owns this table.
relstorage	character(1)	Storage mode of this table. a=append-optimized, c=column-oriented, h =heap, v = virtual, x= external table.
relkind	character(1)	The type of object: r = heap or append-optimized table.
bloat	real	Calculated bloat for the table when last scanned.
scan_size	bigint	Size of table when last scanned.
unused	real	Unused.
skew	real	Calculated skew for the table when last scanned.
last_scan_rowcnt	real	Number of rows in the table when last scanned.
last_scan_ts	timestamp with time zone	Time the table was last scanned for Recommendations.
accuracy	real	Accuracy ratio calculated from query history when the table was last scanned.
last_accuracy_ts	timestamp with time zone	Time the accuracy ratio was last set.
distributed_by	text	Distribution policy for the table.

gpcc_table_info_history

The `gpcc_table_info_history` table stores a daily snapshot of statistics about tables. Command Center saves statistics from the `gpcc_table_info` table just before midnight each night. There is one row for each table per day.

Column	Type	Description
ctime	timestamp(0) without time zone	Time the record was created.
dbid	oid	Object ID of the database.
relid	oid	Object ID of the table.
paroid	oid	Object ID of a partition.
reltablespace	oid	Object ID of the table's tablespace.
seq_scan	bigint	Number of sequential scans initiated on this table.
idx_scan	bigint	Number of index scans initiated on this table.
n_tup_ins	bigint	Number of rows inserted.
n_tup_del	bigint	Number of rows deleted.
n_tup_upd	bigint	Number of rows updated (includes hot updated rows).
n_tup_hot_upd	bigint	Number of rows HOT updated (no separate index update required).
last_seq_scan	timestamp with time zone	Time of the last sequential scan on this table.
last_idx_scan	timestamp with time zone	Time of the last index scan on this table.
last_ins	timestamp with time zone	Time of the last insert on this table.
last_del	timestamp with time zone	Time of the last delete on this table.
last_upd	timestamp with time zone	Time of the last update on this table.
last_analyze	timestamp with time zone	Time of the last analyze on this table.
last_vacuum	timestamp with time zone	Time of the last vacuum of this table (excludes vacuum full).
last_vacuum_full	timestamp with time zone	Time of the last vacuum full of this table.
size	bigint	Combined size of all of this table's files.
row_cnt	real	Number of rows in the table.
children	integer	Number of partitions, including middle-level and child partitions.
schema	name	Name of the schema this table belongs to.
table_name	name	Name of the table.

Column	Type	Description
owner	name	Name of the database role that owns this table.
relstorage	character(1)	Storage mode of this table. a=append-optimized, c=column-oriented, h = heap, v = virtual, x= external table.
relkind	character(1)	The type of object: r = heap or append-optimized table.
bloat	real	Calculated bloat for the table when last scanned.
scan_size	bigint	Size of the table when last scanned.
unused	real	
skew	real	Calculated skew for the table when last scanned.
last_scan_rowcnt	real	Number of rows in the table when last scanned.
last_scan_ts	timestamp with time zone	Time the table was last scanned for Recommendations.
accuracy	real	Accuracy ratio calculated from query history when the table was last scanned.
last_accuracy_ts	timestamp with time zone	Time the accuracy ratio was last set.
distributed_by	text	Distribution policy for the table.

Real-Time Monitoring Tables

gpcc_queries_now

The `gpcc_queries_now` table saves real-time query metrics data.

Column	Type	Description
tmid	integer	A part of the query ID.
ssid	integer	Session ID of the query.
ccnt	integer	A part of the query ID.
username	character varying(64)	User name.
db	character varying(64)	Database.
status	character varying(16)	Query status.
rsgname	character varying(64)	Resource group name.
rsqname	character varying(64)	Resource queue name.
rsqpriority	character varying(16)	Resource queue priority.
tsubmit	timestamp(0) without time zone	Submit time.
tstart	timestamp(0) without time zone	Start time.
cpu_time	double precision	CPU time accumulated (unit: seconds).

Column	Type	Description
cpu_master	double precision	CPU time on master.
cpu_segment	double precision	CPU time on all segemnts.
cpu_master_percent	double precision	CPU percent on master.
cpu_segment_percent	double precision	CPU percent on all segments.
spill_size	bigint	Spill file size (bytes)
memory	bigint	Memory (KB)
disk_read_rate	double precision	Current disk read rate (KB/s).
disk_write_rate	double precision	Current disk write rate (KB/s).
disk_read_bytes	bigint	Current disk read bytes (bytes).
disk_write_bytes	bigint	Current disk write bytes (bytes).
skew_cpu	double precision	CPU skew.
plan_gen	character varying(16)	Plan generator for the query.
cost	double precision	Query cost.
query_text	text	Query text.

Workload Management Tables

gpcc_wlm_rule

The `gpcc_wlm_rule` table stores the definition of workload management rules (both assignment rules and workload rules).

Column	Type	Description
serial_number	integer	Serial number of the rule.
rule_id	integer	Unique ID of the rule shown in the Command Center interface.
rsgname	name	Resource group name used to assign rule.
role	name	Database role name used to assign rule.
query_tag	text	Query tag string used to assign rule.
dest_rsg	name	Destination resource group name for moving a query.
cpu_time	integer	Maximum CPU time consumed by the query.
running_time	integer	Maximum running time for the query.
disk_io_mb	integer	Maximum total disk I/O used by the query.
planner_cost	float8	Maximum Postgres Planner cost for the query.
orca_cost	float8	Maximum GPORCA optimizer ost for the query.

Column	Type	Description
slice_num	integer	Maximum number of slices assigned for executing the query.
action	integer	Action to performed when conditions are met.
active	boolean	Whether the rule is active or inactive.
ctime	timestamp without time zone	Time the rule was created.
etime	timestamp without time zone	Time that the rule was deleted.
idle session	json	JSON string containing parameters for idle session kill rules.
spill_file_mb	int	Spill file size, in MB.

gpcc_wlm_log_history

The `gpcc_wlm_log_history` table stores the log history of workload rule actions that are triggered. Note that log history for assignment rules is not collected.

Column	Type	Description
ctime	timestamp without time zone	Time the log entry was created.
tstart	timestamp without time zone	Time the operation started.
tfinish	timestamp without time zone	Time the operation finished.
rule_serial_number	integer	Serial number of the rule.
rule_id	integer	Unique ID of the rule shown in the Command Center interface.
tmid	integer	A time identifier for the query. All records associated with a query will have the same `tmid`.
ssid	integer	Session id for the database connection. All records associated with the query will have the same `ssid`.
ccnt	integer	Command number within the session. All records associated with the query will have the same `ccnt`.
action	integer	Workload rule action that was initiated.
rsgname	name	Resource group name used to assign rule.
role	name	Database role name used to assign rule.
status	text	Indication of whether the action succeeded or failed.
fail_msg	text	Message associated with a failed action.

Other Tables

Command Center uses the `gpcc_department` and `gpcc_role_department` tables to store information about departments and roles.

gpcc_department

Column	Type	Description
dept_id	integer	Department ID.
dept_name	character varying(64)	Department name. Limited to 64 characters, can only contain [A-Z][a-z][0-9][_].
dept_status	smallint	1 or 0, where 1 indicates active and 0 indicates deleted.

gpcc_role_department

Column	Type	Description
rolname	name	Role name.
dept_id	integer	Department ID.

User-Defined Functions

The `gpmetrics` schema includes a number of user-defined functions that provide users with information about queries and about the hosts on which the queries run.

gpcc_delete_department

The `gpcc_delete_department` user-defined function deletes a department name from the `gpperfmon` database. It takes one parameter: the name of the department you want to delete. You may only delete department names that have no roles associated with them.

gpcc_queries_per_hour

The `gpcc_queries_per_hour` user-defined function returns the average and maximum number of total queries, running queries, queued queries, blocked queries, and finished queries per hour. Here is an example query:

```
select * from gpcc_queries_per_hour();
```

gpcc_queries_per_user

The`gpcc_queries_per_user` user-defined function returns, for each user, the number of queries whose runtime is longer than the input interval, per hour, in the specified time range. The function takes an interval, a start time, and an end time. The default start time is 00:00:00 today. The default end time is the current time. The interval can not be omitted. It accepts PostgreSQL's `INTERVAL` value.

This example returns, by user, all queries running longer than 5 minutes for the interval between 00:00:00 05/01/2018 and 23:59:59 06/01/2018:

```
select * from gpmetrics.gpcc_queries_per_user('5min','2018-05-01','2018-06-01');
```

gpcc_queries_per_user_max_and_total_spill_size

The `gpcc_queries_per_user__max_and_total_spill_size` user-defined function returns, for each user, the total `spill_size` and maximum `spill_size` per query per hour. The function takes a start time and an end time. The default start time is 00:00:00 today. The default end time is the current time.

This example returns, by user, the total `spill_size` and maximum `spill_size` per query per hour for the interval between 00:00:00 05/20/2018 and 23:59:59 05/21/2018.

```
select * from gpmetrics.gpcc_queries_per_user_max_and_total_spill_size('2018-05-20','2
018-05-21');
```

gpcc_queries_per_user_max_cpu

The `gpcc_queries_per_user_max_cpu` user-defined function returns, for each user, the query with the maximum segment and master cpu usage per hour, along with details about the query. The function takes a start time and an end time. The default start time is 00:00:00 today. The default end time is the current time.

This example returns, by user, the query with the maximum segment and master cpu usage per hour for the interval between 00:00:00 05/20/2018 and 23:59:59 05/21/2018.

```
select * from gpmetrics.gpcc_queries_per_user_max_cpu('2018-05-20','2018-05-21');
```

gpcc_queries_per_user_max_run_time

The `gpcc_queries_per_user_max_run_time` user-defined function returns, for each user, the longest running query per hour, along with details about the query. The function takes a start time and an end time. The default start time is 00:00:00 today. The default end time is the current time.

This example returns, by user, the longest running query per hour for the interval between 00:00:00 05/20/2018 and 23:59:59 05/21/2018:

```
select * from gpmetrics.gpcc_queries_per_user_max_run_time('2018-05-20','2018-05-21');
```

gpcc_queries_per_user_max_skew

The `gpcc_queries_per_user_max_skew` user-defined function returns, for each user, the query with the maximum amount of processing skew in the system (`skew_cpu`) per hour, along with details about the query. The function takes a start time and an end time. The default start time is 00:00:00 today. The default end time is the current time.

This example returns, by user, the query with the maximum amount of processing skew in the system for the interval between 00:00:00 05/20/2018 and 23:59:59 05/21/2018.

```
select * from gpmetrics.gpcc_queries_per_user_max_skew('2018-05-20','2018-05-21');
```

gpcc_queries_per_user_rows_out

The `gpcc_queries_per_user_rows_out` user-defined function returns, for each user, the query with the maximum `rows_out` per hour, along with details about the query. The function takes a start time and an end time. The default start time is 00:00:00 today. The default end time is the current time.

This example returns, by user, the query with the maximum `rows_out` per hour for the interval between 00:00:00 05/20/2018 and 23:59:59 05/21/2018.

```
select * from gpmetrics.gpcc_queries_per_user_rows_out('2018-05-20','2018-05-21');
```

gpcc_system_per_hour

The `gpcc_system_per_hour` user-defined function returns a variety of system information, including: the average `cpu_user` and `cpu_sys` for all hosts, aggregated by hour; the average `load0`, `load1`, and `load2` per hour for all hosts, aggregated by hour; and the average disk write and disk read bytes for all hosts, aggregated by hour. The function takes a start time and and time. The default start time is 00:00:00 today. The default end time is the current time.

This example returns the system information summarized in the previous paragraph for the interval between 00:00:00 08/30/2021 and 23:59:59 08/31/2021.

```
gpperfmon=# select * from gpmetrics.gpcc_system_per_hour('2021-08-30','2021-08-31');
```

gpcc_update_department

The `gpcc_update_department` user-defined function renames a department name. It takes two parameters: `from_name` -- which specifies the old name -- and `to_name` -- which specifies the new name. Once it has run, all the roles associated with the old name will now be associated with the new name.

Configuration Files Reference

Configuration parameters for Greenplum Command Center are stored in the following files.

`$MASTER_DATA_DIRECTORY/gpperfmon/conf/gpperfmon.conf`, on Greenplum Database master host

Stores configuration parameters for the Greenplum Command Center agents.

`$GPCC_HOME/conf/app.conf`, on Command Center host.

Stores configuration parameters for the Command Center web application and web server.

`$MASTER_DATA_DIRECTORY/gpmetrics/gpcc.conf`

Stores configuration parameters for Command Center interface options and alert emails.

`$MASTER_DATA_DIRECTORY/postgresql.conf`

Stores configuration parameters to enable the Greenplum Command Center features for Greenplum Database server. These parameters are normally set using the `gpconfig` Greenplum Database management utility.

Command Center Agent Parameters

The `$MASTER_DATA_DIRECTORY/gpperfmon/conf/gpperfmon.conf` file on the Greenplum Database master host stores configuration parameters for the Command Center agents. For configuration changes to these options to take effect, you must save `gpperfmon.conf` and then restart Greenplum Database server (`gpstop -r`).

To enable the Command Center agents within Greenplum Database server, you must also set the Greenplum Database Server Configuration Parameters. See the `gpperfmon_install` reference in the *Greenplum Database Utility Guide* for details.

log_location

Specifies a directory location for Command Center log files. Default is `$MASTER_DATA_DIRECTORY/gpperfmon/logs`.

min_query_time

Specifies the minimum query run time in seconds for statistics collection. Command Center logs all queries that run longer than this value in the queries_history table. For queries with shorter run times, no historical data is collected. Defaults to 20 seconds.

If you know that you want to collect data for all queries, you can set this parameter to a low value. Setting the minimum query run time to zero, however, collects data even for the numerous queries run by Command Center itself, creating a large amount of data that may not be useful.

min_detailed_query_time

Specifies the minimum iterator run time in seconds for statistics collection. Command Center logs all iterators that run longer than this value in the iterators_history table. For iterators with shorter run times, no data is collected. Minimum value is 10 seconds.

This parameter’ s value must always be equal to, or greater than, the value of `min_query_time`. Setting `min_detailed_query_time` higher than `min_query_time` allows you to log detailed query plan iterator data only for especially complex, long-running queries, while still logging basic query data for shorter queries.

Given the complexity and size of iterator data, you may want to adjust this parameter according to the size of data collected. If the `iterators_*` tables are growing to excessive size without providing useful information, you can raise the value of this parameter to log iterator detail for fewer queries.

max_log_size

This parameter is not included in gpperfmon.conf, but it may be added to this file for use with Greenplum Command Center.

To prevent the log files from growing to excessive size, you can add the `max_log_size` parameter to `gpperfmon.conf`. The value of this parameter is measured in bytes. For example:

```
max_log_size = 10485760
```

With this setting, the log files will grow to 10MB before the system rolls over to a new log file.

partition_age

The number of months that Greenplum Command Center statistics data will be retained. The default it is 0, which means we won't drop any data.

quantum

Specifies the time in seconds between updates from Command Center agents on all segments. Valid values are 10, 15, 20, 30, and 60. Defaults to 15 seconds.

If you prefer a less granular view of performance, or want to collect and analyze minimal amounts of data for system metrics, choose a higher quantum. To collect data more frequently, choose a lower value.

ignore_qexec_packet

When set to true, Greenplum Command Center agents do not collect performance data in the gpperfmon database `queries_*` tables: `rows_out`, `cpu_elapsed`, `cpu_currpct`, `skew_cpu`, and `skew_rows`. The default setting, true, reduces the amount of memory consumed by the gpmmon process. Set this parameter to false if you require this additional performance data.

smdw_aliases

This parameter allows you to specify additional host names for the standby master. For example, if the standby master has two NICs, you can enter:

```
smdw_aliases= smdw-1,smdw-2
```

This optional fault tolerance parameter is useful if the Greenplum Command Center loses connectivity with the standby master. Instead of continuously retrying to connect to host smdw, it will try to connect to the NIC-based aliases of `smdw-1` and/or `smdw-2`. This ensures that the Command

Center Console can continuously poll and monitor the standby master.

Command Center Console Parameters

The Command Center Console configuration file is on the Command Center host at `$GPCC_HOME/conf/app.conf`. Some parameters in this file are set by the Command Center installer.

You can add security settings in `app.conf` to suit your environment. See Security Parameters.

After editing this file, reload the configuration by restarting the Command Center Console.

```
$ gpcc --start
```

`appname = gpccws`

The web server binary file. Do not change.

`httpport = <port>`

The web server port when `EnableHTTP` is true. The default is 28080.

`httpsport = <port>`

The web server port when `EnableHTTPS` is true. The default is 28080.

`rpcport = <port>`

The port on which the Command Center backend receives data from metrics collector agents. The default is 8899.

`listentcp4 = [true | false]`

When `true`, the address type is tcp4. The default is `true`.

`runmode = [prod | dev | test]`

The application mode, which can be `dev`, `prod` or `test`. The default, `prod`, is the recommended setting. In `dev` and `test` modes Command Center prints more verbose log messages. These are different logs than the logs affected by the `log_level` parameter.

`session = [true | false]`

Use sessions to manage user experience. The default is `true`. Sessions are stored in memory.

`enablexsrf = [true | false]`

Enable CSRF protection.

`xsrfkey = <token_string>`

The CSRF token.

`xsrfexpire = <seconds>`

CSRF expire time. The default is `2592000` seconds.

`rendertype = json`

The render type of the web server. Do not change.

`printallsqls = [true | false]`

Print all backend gpperfmon SQL to the web server console. The default is `false`.

`log_level`

The level of messages to log: `Debug`, `Info`, or `Error`. `Debug` is the most verbose and `Error` is the least verbose. The default is `Info`.

`master_host = <hostname>`

The Greenplum Database host name. The default is `localhost`.

`master_port = <port>`

The Greenplum Database master port. The default is `5432`.

`path = /usr/local`

Path to the directory where Greenplum Command Center is installed.

`display_name = <display_name>`

The display name for the console.

`enable_kerberos = [true | false]`

True if Kerberos authentication is enabled for Command Center. The default is `false`.

`enable_history = [true | false]`

True if history data collection is enabled for Command Center. The default is `true`. This parameter is managed in Command Center by setting **Enable GPCC history data collection** on or off on the **Admin> Settings** page.

`HTTPSCertFile = </path/to/cert.pem>`

`HTTPSKeyFile = </path/to/cert.pem>`

Set both of these properties to the full path to a .pem file containing the certificate and private key for the Command Center web server.

`EnableHTTPS = [true | false]`

Enable listening on the secure SSL port. The default is `true`. True if SSL is enabled. Only one of `EnableHTTPS` or `EnableHTTP` can be true. <

`EnableHTTP = [true | false]`

Enable listening on the HTTP port. True if SSL is not enabled. Only one of `EnableHTTP` or `EnableHTTPS` can be true.

`HTTPAddr = <ipaddress>`

The IPv6 address of the host that runs Command Center. It is only necessary to set this parameter if Command Center is running in an IPv6 environment.

`stats_check_interval = <seconds>`

How often the statistics in the Command Center Table Browser are refreshed. The default is `300`. New tables and changed values such as file size and last access time may not be seen until `stats_check_interval` seconds have elapsed.

`ws_perf_port = <port>`

Port to access the Command Center web server Go profiling data. (See pprof for more information.)

The default is `6162`. Choose another port if there is a port conflict or if you are setting up another Command Center instance on the same host.

`agent_perf_port = <port>`

Port to use to access agent Go profiling data. The default is `6163`. Choose another port if there is a port conflict on segment hosts, or if you are setting up another Command Center instance on the same cluster.

Setting Security Parameters

You may customize the following security headers:

```
  "Cache-Control",             // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/Cache-Control
  "Content-Security-Policy",   // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/Content-Security-Policy
  "Permissions-Policy",        // See https://www.w3.org/TR/permissions-policy-1/
  "Referrer-Policy",           // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/Referrer-Policy
  "Strict-Transport-Security", // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/Strict-Transport-Security
  "X-Content-Type-Options",    // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/X-Content-Type-Options
  "X-Frame-Options",           // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/X-Frame-Options
  "X-XSS-Protection",          // See https://developer.mozilla.org/en-US/docs/Web/HTT
P/Headers/X-XSS-Protection
```

The following headers are configured by default and are set to these values:

```
 "Cache-Control":             "no-store",
 "Referrer-Policy":           "same-origin",
 "Strict-Transport-Security": "max-age=31536000",
 "X-Content-Type-Options":    "nosniff",
 "X-Frame-Options":           "SAMEORIGIN",
 "X-XSS-Protection":          "1; mode=block",
```

You may customize the following security headers:

Where:

- `"Cache-Control": "no-store"` indicates that the response may not be stored in any cache.
- `"Referrer-Policy": "same-origin"` indicates a referrer will be sent for same-site origins, but cross-origin requests will send no referrer information.
- `"Strict-Transport-Security": "max-age=31536000"` indicates the time, in seconds, that the browser should remember that a site is only to be accessed using HTTPS.
- `"X-Content-Type-Options": "nosniff"` blocks or allows requests depending on type.
- `"X-Frame-Options": "SAMEORIGIN"` indicates that the page can only be displayed in a frame on the same origin as the page itself.
- `"X-XSS-Protection": "1; mode=block"` enables XSS filtering and the browser will block page rendering if it detects an attack.

To customize any of these headers, enter your values in the `app.conf` file and restart Command Center. For example, to customize `Content-Security-Policy`, `Permissions-Policy`, and `X-Frame-Options` use a `app.conf` entry similar to:

```
[security_headers]
    Content-Security-Policy = default-src 'self' http://example.com;
    Permissions-Policy = fullscreen=(), geolocation=()
    X-Frame-Options = DENY
```

Setting TLS Cipher Suites

By default, Command Center supports the following cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

To use cipher suites other than the default four, add them to `$GPCC_HOME/conf/app.conf` in a section labeled `[tls_cipher_suites]`, as in the following example:

```
[tls_cipher_suites]
Enable_TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 = true
Enable_TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 = true
Enable_TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 = true
Enable_TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 = true
Enable_TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 = true
Enable_TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA = true
Enable_TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 = true
Enable_TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA = true
Enable_TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA = true
Enable_TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA = true
Enable_TLS_RSA_WITH_AES_128_GCM_SHA256 = true
Enable_TLS_RSA_WITH_AES_256_GCM_SHA384 = true
Enable_TLS_RSA_WITH_AES_128_CBC_SHA256 = true
Enable_TLS_RSA_WITH_AES_128_CBC_SHA = true
Enable_TLS_RSA_WITH_AES_256_CBC_SHA = true
Enable_TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305 = true
Enable_TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305 = true
```

Warning: Cipher suites that are not among the four default cipher suites may have potential security risks and are not recommended.

When there are one or more entries under `[tls_cipher_suites]` in `app.conf`, Command Center will not use any default cipher suites, unless they are also declared in the `[tls_cipher_suites]` section.

gpmetrics Configuration File Reference

Greenplum Command Center uses the `gpcc.conf` configuration file to save configuration information entered in the Command Center user interface.

You should not normally edit the `gpcc.conf` file directly. Instead, modify configuration information in the Command Center user interface. The threshold properties for the **Recommendations** page reports are an exception and must be added manually. If you edit the file, you must restart Command Center for the new values to take effect.

The `gpcc.conf` file is created in the `$MASTER_DATA_DIRECTORY/gpmetrics/` directory on the Greenplum Database master or standby host where you start Command Center. The file is an INI-format configuration file, containing properties defined as `key = value` entries, one property per line.

Property	Description	
min_query_time	Queries shorter than number of seconds specified will not be captured in Query History.	[history]
wlm_short_query_threshold	Specifies the minimum number of seconds that a query must run before Command Center applies the action of a matching workload management rule. Use this parameter to prevent Command Center from applying rule actions to short-running queries. The default is 45 seconds.	[wlm]
wlm_query_cooldown_time	Specifies the wait time, in seconds, for retrying a failed workload management rule (canceling a query or moving a query to another resource group). Command Center makes 2 retry attempts for failed rules, after waiting the configured amount of time. The default is 15 seconds.	[wlm]
allow_anonymous	If `true`, Command Center users can access the Query Monitor view without logging into Command Center. You can change this setting on the Command Center **Admin> Permissions** page.	
resource_queue_import_status	Command Center uses this property to determine whether to offer to import Greenplum Database resource queues to resource groups when you access the **Workload Mgmt** view. The default is `false`.	
emailFrom	The email address to set on the "From:" line of alert emails. The default is `noreply-gpcc-alerts@pivotal.io`. Note: Set the email and smtp properties on the Command Center **Admin> Alerts** page.	
emailTo	A comma-separated list of email addresses to send alert emails.	
smtpUsername	The account name to use when authenticating with the SMTP server.	
smtpServer	The address and port of the SMTP server to use for alert emails.	
smtpPassword	The password used to authenticate the SMTP user with the SMTP server, base 64-encoded.	
bloat_threshold	Specifies the minimum percentage of bloat (space occupied by dead tuples) required to include a table with a `VACUUM` recommendation in the **Bloat Rank** report on the **Recommendations** page. The default is `0.1` (10%).	
bloat_vacuum_full_threshold	Specifies the minimum percentage of bloat (combined dead tuples and unused space) required to include a table with a `VACUUM FULL` recommendation in the **Bloat Rank** report on the **Recommendations** page. The default is `0.5` (50%).	

Property	Description
accuracy_threshold	Specifies the minimum inaccuracy percentage required to include a table in the **Accuracy% Rank** table on the **Recommendations** page. The default is `0.1` (10%).
skew_threshold	Specifies the minimum skew percentage required to include a table in the **Skew% Rank** report on the **Recommendations** page. The default is `0.1` (10%).
skew_tuple_per_segment	Specifies a minimum number of tuples per segment to include a table in the **Skew% Rank** report on the **Recommendations** page. This parameter helps to avoid reporting tables that have high skew due to small numbers of tuples. The default is `128`, which means a table with fewer tuples than 128 times the number of segments will be excluded from the report.
stats_check_interval	Specifies how often to refresh the statistics in the Command Center Table Browser. The default is `300`. New tables and changed values such as the file size and the last access time may not be seen until `stats_check_interval` seconds have elapsed.

Setup Configuration File

A setup configuration file contains properties used to configure Greenplum Command Center when you perform a non-interactive Command Center installation. The file is passed to the `gpccinstall` command with the `-c` option:

```
$ ./gpccinstall-<version> -c <config_file>
```

The configuration file contains `name: value` or `name=value` entries, one per line. Comments begin with a `#` or `;` character and continue through the end of the line.

See Installing Pivotal Greenplum Command Center for more information about installing Command Center with a configuration file.

Parameters

`path`

The path to the directory where Greenplum Command Center software will be installed. The directory must be writable by the gpadmin user on all hosts in the Greenplum Cluster.

`display_name`

The name to display in the Command Center user interface. The default display name is `gpcc`.

`master_port`

The Greenplum Database master port. Default: `5432`.

`web_port`

The listen port for the Command Center web server. The default is `28080`.

`enable_ssl`

`True` if client connections to the Command Center web server are to be secured with SSL. The default is `false`. If `true` the `ssl_cert_file` parameter must be set and the SSL certificate must be installed on the host where you run Command Center.

`ssl_cert_file`

If `enable_ssl` is `true`, set this parameter to the full path to a valid certificate in PEM file format. The certificate must be installed on the host where you run Command Center.

`enable_kerberos`

Set to `true` to enable Kerberos authentication.

`krb_mode`

The Kerberos authentication scheme to use. The default is `1`.

- **1 - normal mode (default)** - The Command Center Kerberos keytab file contains the Command Center principal and may contain principals for Command Center users. If the principal in the client's connection request is in the Command Center's keytab file, Command Center uses the client's principal for database connections. Otherwise, Command Center uses the gpmon user for database connections.
- **2 - strict mode** - Command Center has a Kerberos keytab file containing the Command Center service principal and a principal for every Command Center user. If the principal in the client's connection request is in the keytab file, the web server grants the client access and the web server connects to Greenplum Database using the client's principal name. If the principal is not in the keytab file, the connection request fails.
- **3 - gpmon_only mode** - Command Center uses the gpmon database role for all Greenplum Database connections. No client principals are needed in the Command Center's keytab file.

`webserver_url`

The web server hostname, from the Kerberos HTTP service principal.

`keytab`

Path to the keytab file containing Kerberos principals for the Command Center web server and users.

Examples

```
#####
# GPCC 4.0 setup configuration file
#####
path = /opt
display_name = Greenplum Database Production Cluster
master_port = 5432
webserver_port = 28081
EnableHTTP = true      ; allow both HTTP and HTTPS
EnableHTTPS = true
ssl_cert_file = /etc/ssl/cert.pem
enable_kerberos = false
```

Server Configuration Parameters

The metrics collector extension in Greenplum Database defines custom server configuration parameters to configure metrics collection options. If the metrics collector extension is installed and the value of the `shared_preload_libraries` configuration parameter includes `metrics_collector` when Greenplum Database starts, these parameters can be viewed and set using the Greenplum Database `gpconfig` utility.

`gpcc.enable_send_query_info`

Enables collection of query metrics in Greenplum Database. When `on`, the metrics collector sends query metrics to the Greenplum Command Center agent.

Type: `boolean`

Default: `on`

`gpcc.enable_query_profiling` When `off` the metrics collector does not collect queries executed by the gpmon user in the gpperfmon database or plan node history for queries that run in less than ten seconds (or `min_query_time`, if it is set and greater than 10 seconds). If enabled in a session, the metrics collector collects those queries in the session.

Type: `boolean`

Default: `off`

`gpcc.query_metrics_port`

The port number to send query metrics. The metrics collector sends query metrics to the Greenplum Command Center agent process on this port.

Type: `integer`

Default: `9898`

gp_external_enable_exec

This parameter is enabled by default and must remain enabled. It allows the use of external tables that execute OS commands or scripts on the segment hosts. The Command Center agents use this type of external tables to collect current system metrics from the segments.

gp_enable_query_metrics

When on, enables query metrics collection. The default is off. After setting this configuration parameter, Greenplum Database must be restarted for the change to take effect.

gp_instrument_shmem_size

The amount of shared memory, in kilobytes, allocated for query metrics. The default is 5120 and the maximum is 131072. At startup, if `gp_enable_query_metrics` is set to on, Greenplum Database allocates space in shared memory to save query metrics. This memory is organized as a header and a list of slots. The number of slots needed depends on the number of concurrent queries and the number of execution plan nodes per query. The default value, 5120, is based on a Greenplum Database system that executes a maximum of about 250 concurrent queries with 120 nodes per query. If the `gp_enable_query_metrics` configuration parameter is off, or if the slots are exhausted, the metrics are maintained in local memory instead of in shared memory.

shared_preload_libraries

A comma-separated list of shared libraries that are to be preloaded when Greenplum Database starts. The workload management and query metrics extension libraries must be included in this configuration parameter to use Greenplum Command Center.

track_activities

Enables the collection of information on the currently executing command of each session, along with the time when that command began execution. The default value is true. Only superusers can change this setting.

Securing Greenplum Command Center

Greenplum Command Center Console can be secured by encrypting network traffic between the web server and users' browsers, authenticating Command Center users, and managing users' permissions to access Command Center features.

SSL/TLS Encryption

Greenplum Command Center supports SSL/TLS encryption to secure connections between browsers and the Command Center web server. Command Center supports TLS 1.2 protocol and higher. When enabled, Command Center uses the Secure WebSockets API, enabling long-lived, full-duplex connections, in addition to encryption.

To enable SSL/TLS encryption, you need a combined certificate/key file for the Command Center web server in place when you install Command Center. The file contains a private key and a server certificate.

You can request a key pair and certificate from your organization's internal certificate authority or from an external certificate authority. You can also create a self-signed certificate with a cryptography suite such as OpenSSL. If you create a self-signed certificate, however, clients will have to override a security warning when they first connect to the Command Center web server.

To create the combined certificate/key file, create a text file, for example `server.pem`, and copy the entire body of the private key and certificate into it. Make sure to include the beginning and end tags:

```
-----BEGIN RSA PRIVATE KEY-----
   < private key >
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
   < certificate >
-----END CERTIFICATE-----
```

You can concatenate additional certificates to the end of the file if a certificate chain is required to authenticate the server certificate.

Place the file on the server where Command Center will execute, for example in the `/etc/ssl/certs` directory of the Greenplum Database master host. When you install Command Center the installer prompts you to enter the full path to this file. See Command Center Console Parameters for details.

Authentication Options

Users logging in to Greenplum Command Center are authenticated with the Greenplum Database host-based authentication system. Users can enter credentials as a user name and password or, if Kerberos authentication is configured, by authenticating with Kerberos on their workstation before browsing to the Command Center web server.

Note: Greenplum Command Center does not accept logins from the gpadmin user, or from users configured with trust authentication in the `pg_hba.conf` file.

Database users must first be added to the Greenplum Database by using commands such as `CREATE ROLE` or `CREATE USER`. The `LOGIN` privilege is required. This example creates a login user with an encrypted password:

```
CREATE ROLE cc_user WITH LOGIN ENCRYPTED PASSWORD 'changeme';
```

The `pg_hba.conf` configuration file determines how authentication will proceed. This file contains a list of entries that are compared to attributes of the user's connection request, including the type of connection, network location of the originating host, database name, and login user name. When a match is found, the authentication method specified in the entry is applied.

The `pg_hba.conf` file can be viewed by Operators and edited by Admins in the Command Center console on the Admin>Authentication page.

The `md5` and `password` authentication methods authenticate the user name and password with the Greenplum Database `pg_roles` system table. The `md5` method requires the password to be MD5-encoded when sent over the network, so it is preferred over the `password` method, which sends the password in clear text.

The `ldap` authentication method authenticates the user name and password with an LDAP server. The LDAP server and parameters are specified in the options field of the `pg_hba.conf` entry. See the PostgreSQL LDAP authentication documentation for the format of the LDAP options.

The `gss` authentication method is used for Kerberos authentication. To use Kerberos with Command Center, Kerberos authentication must be enabled for the Greenplum Database system and Command Center must also be configured. Users authenticate with the Kerberos KDC on their workstations (using `kinit`, for example) before connecting to the Command Center web server. The role name in Command Center is the user's Kerberos principal name.

For details about setting up Kerberos authentication, see Enabling Kerberos Authentication with Greenplum Command Center.

See the PostgreSQL Authentication methods documentation for additional details of the authentication options.

Authorization

Note: The functionality described in this section has not been fully implemented in Greenplum Command Center 4.0.0. Only Admin and Self Only permission levels are available.

Command Center manages permission levels using Greenplum Database roles and groups. The Basic, Operator Basic, and Operator permission levels correspond to the `gpcc_basic`, `gpcc_operator_basic`, and `gpcc_operator` group roles in the database. The Admin permission level is conferred to roles that have the `SUPERUSER` privilege. A user who has not been added to any of the groups and does not have `SUPERUSER` privilege has the most restrictive permission level, Self Only.

Greenplum Database superusers can manage permission levels on the Command Center User Management page. Superusers can also directly assign users roles in the database by using the

`ALTER USER`, `ALTER GROUP`, and related commands to add or remove users from groups and add or remove the `SUPERUSER` privilege. If a role is configured for more than one permission level, Command Center uses the highest permission level.

Command Center users have the following capabilities, according to their permission levels:

Self Only

Users can view metrics and view and cancel their own queries.

Any Greenplum Database user successfully authenticated through the Greenplum Database authentication system can access Greenplum Command Center with Self Only permission. Higher permission levels are required to view and cancel other's queries and to access the System and Admin Control Center screens.

Basic

Allows users to view metrics, view all queries, and cancel their own queries.

Users with Basic permission are members of the Greenplum Database `gpcc_basic` group.

Operator Basic

Allows users to view metrics, view their own and others' queries, cancel their own queries, and view the System and Admin screens.

Users with Operator Read-only permission are members of the Greenplum Database `gpcc_operator_basic` group.

Operator

Allows users to view their own and others' queries, cancel their own and other's queries, and view the System and Admin screens.

Users with Operator permission are members of the Greenplum Database `gpcc_operator` group.

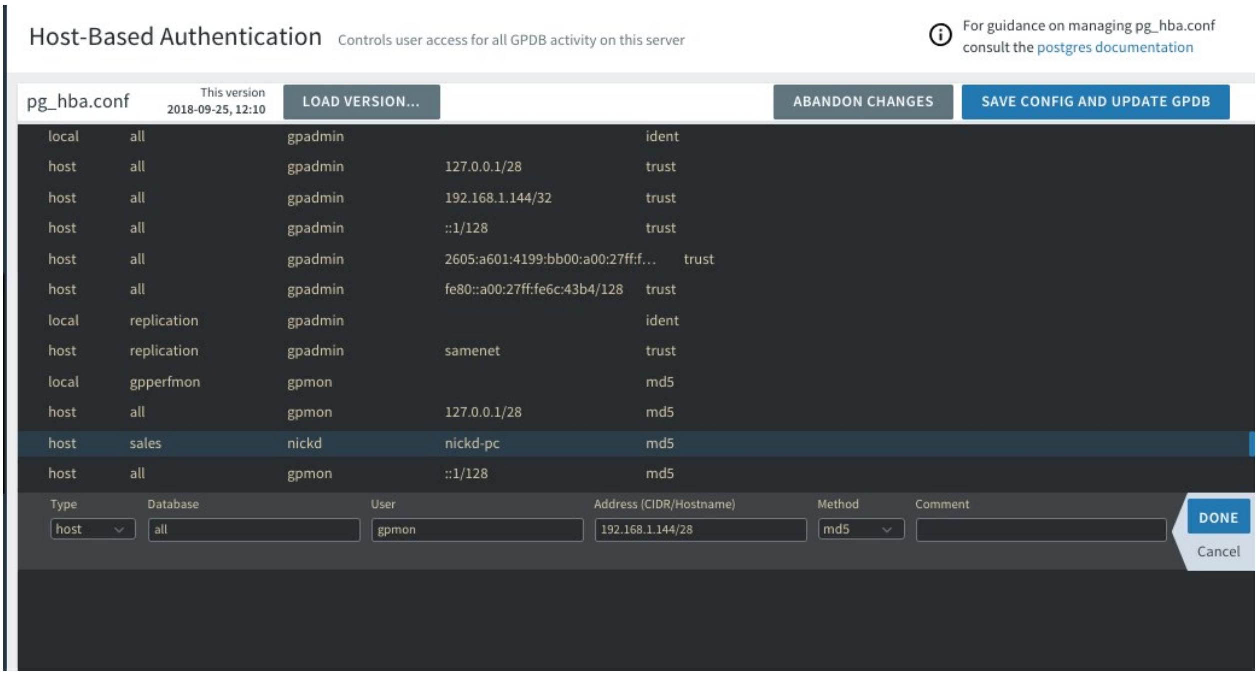
Admin

Allows users access to all views and capabilities in the Command Center.

Greenplum Database users with the `SUPERUSER` privilege in Greenplum Database have Superuser permissions in Command Center.

Managing Greenplum Command Center Authentication

The **Admin> Authentication** screen allows users with Operator Basic, Operator, and Admin permission to view the Greenplum Database host-based authentication file, `pg_hba.conf`.


Host-Based Authentication
Controls user access for all GPDB activity on this server
For guidance on managing pg_hba.conf consult the postgres documentation
pg_hba.conf
This version 2018-09-25, 12:10
LOAD VERSION...
ABANDON CHANGES
SAVE CONFIG AND UPDATE GPDB
DONE
Cancel

local	all	gpadmin		ident
host	all	gpadmin	127.0.0.1/28	trust
host	all	gpadmin	192.168.1.144/32	trust
host	all	gpadmin	::1/128	trust
host	all	gpadmin	2605:a601:4199:bb00:a00:27ff:f...	trust
host	all	gpadmin	fe80::a00:27ff:fe6c:43b4/128	trust
local	replication	gpadmin		ident
host	replication	gpadmin	samenet	trust
local	gpperfmon	gpmon		md5
host	all	gpmon	127.0.0.1/28	md5
host	sales	nickd	nickd-pc	md5
host	all	gpmon	::1/128	md5

Type	Database	User	Address (CIDR/Hostname)	Method	Comment
host	all	gpmon	192.168.1.144/28	md5	

Users with Admin permission can add, remove, change, and move entries in the file. The Command Center UI validates entries to ensure correct syntax. Previous versions of the file are archived so that you can restore an earlier version or audit changes.

See Authentication Options for an overview of user authentication options for Greenplum Database and Greenplum Command Server.

See Configuring Client Authentication in the *Greenplum Database Administrator Guide* for a detailed description of the contents of the `pg_hba.conf` file.

Viewing the Host-Based Authentication File

Choose **Admin>Authentication** to display the content of the Greenplum Database `pg_hba.conf` file.

The `pg_hba.conf` file contains a list of entries that specify the characteristics of database connection requests and authentication methods. When Greenplum Database receives a connection request from a client, it compares the request to each entry in the `pg_hba.conf` entry in turn until a match is found. The request is authenticated using the specified authentication method and, if successful, the connection is accepted.

Editing the Host-Based Authentication File

Command Center users with the *Admin* permission can edit the `pg_hba.conf` file. Note that any changes you make are lost if you move to another screen before you save them.

- To change an existing entry, click anywhere on the entry. Edit the fields and click **Save** to save your changes, or **Cancel** to revert changes.
- To move an entry up or down in the list, click on the ▦ symbol, drag the line to the desired location, and release.
- To add a new entry to the end of the file, click **Add New Entry** at the bottom of the screen. Edit the fields and click **Save** to save your changes, or **Cancel** to abandon the new entry.

- To add a new entry after an existing entry, highlight the existing entry and click ➕. Edit the fields and click **Save** to save your changes, or **Cancel** to abandon the new entry.
- To copy an entry, select the entry and click 🗐. A copy of the selected entry is added below the selected entry and displayed for editing. Edit the fields and click **Save** to save your changes, or **Cancel** to abandon the copy.
- To add a comment to the file, add an entry by clicking **Add New Entry** or ➕ and then choose `#` from the `Type` list.
- To toggle an entry between active and inactive, select the line and click the **active/inactive** toggle control to the right. This action adds or removes a comment character (`#`) at the beginning of the entry.
- To remove an entry, highlight the line and click ⊖. The entry is displayed with strikethrough text. You can restore the entry by highlighting it and clicking **undelete**. The entry is permanently removed when you click **Save config and update GPDB**.
- To finish editing, click **Save config and update GPDB**. Then click **Save and Update** to save your changes or click **Cancel** to return with your edits intact.

When you select **Save and Update**, the `pg_hba.conf` file is saved and refreshed in Greenplum Database. Note that existing client connections are unaffected.

Loading a Previous Version of the Host-Based Authentication File

When you save a new version of the `pg_hba.conf` file, a copy is saved in the Greenplum Database `$MASTER_DATA_DIRECTORY/pg_hba_archive` directory as `pg_hba.conf-<timestamp>`.

To view an archived version of the `pg_hba.conf` file, click **Load versions...** and click the timestamp for the version to display.

To revert to a previous version of the file, load the previous version and then click **Save config and update GPDB**. The configuration is refreshed in Greenplum Database and saved as a new version in the archive directory.

Enabling Authentication with Kerberos

If you have enabled Kerberos authentication for Greenplum Database, you can set up Greenplum Command Center to accept connections from Kerberos-authenticated users.

Greenplum Database and Command Center include support for the Generic Security Service Applications Program Interface (GSS-API) standard. A related standard, Simple and Protected GSS-API Negotiation Mechanism (SPNEGO), describes the protocol GSS-API clients and servers use to agree on the method of authentication.

With a SPNEGO-compliant web application such as Command Center, the client and server agree on the authentication method on the client's initial HTTP request. If Kerberos authentication is not supported on both ends of the connection the server falls back to basic authentication, and displays a login form requesting a user name and password. If a user has authenticated on the workstation with Kerberos and has a valid ticket granting ticket, the web browser offers the user's credential to the Command Center web server. A Kerberos-enabled Command Center web server is configured to handle the authenticated user's connection request in one of three modes, called strict, normal, or gpmon-only.

Strict

Command Center has a Kerberos keytab file containing the Command Center service principal and a principal for every Command Center user. If the principal in the client's connection request is in the keytab file, the web server grants the client access and the web server connects to Greenplum Database using the client's principal name. If the principal is not in the keytab file, the connection request fails.

Normal

The Command Center Kerberos keytab file contains the Command Center principal and may contain principals for Command Center users. If the principal in the client's connection request is in Command Center's keytab file, it uses the client's principal for database connections. Otherwise, Command Center uses the gpmon user for database connections.

gpmon-only

Command Center uses the gpmon database role for all Greenplum Database connections. No client principals are required in the Command Center' s keytab file. This option can be used, for example, if Command Center users authenticate with Active Directory and you do not want to maintain client principals in the keytab file.

If you have set up Kerberos authentication for Greenplum Database, most of the configuration required to enable Command Center Kerberos authentication has been done. The Command Center Kerberos configuration builds upon the Greenplum Database Kerberos setup.

Kerberos authentication can be enabled by responding to prompts when you install Command Center, or you can use the `gpcc --krbenable` command to enable Kerberos after Command Center has been installed.

Before You Begin

Kerberos authentication must be enabled for Greenplum Database. See Using Kerberos Authentication for instructions. Make sure the following prerequisites are met before you continue:

- The `krb5-workstation` package and associated libraries (`libkrb5*`) must be installed on the Greenplum master host and each client workstation.
- The date and time on the Greenplum master host and all client workstations must be synchronized with the KDC.
- The `krb5.conf` configuration file must be the same on the KDC host, the Greenplum Database master host, and client workstations.
- The KDC database must have a service principal for Greenplum Database. The default service name for Greenplum Database is `postgres/<master-host>@<realm>`. You can choose a service name other than `postgres`, but it must match the value of the `krb_srvname` parameter in the `$MASTER_DATA_DIRECTORY/postgresql.conf` file.
- A keytab file with the Greenplum Database principal must be installed on the Greenplum master host and identified by the `krb_server_keyfile` parameter in the `$MASTER_DATA_DIRECTORY/postgresql.conf` file.

- Each client workstation can have a keytab file containing their Kerberos principal, `<username>@<realm>`.

Add Command Center Principals to the KDC Database

Before you configure Command Center for Kerberos authentication, you must create the required Kerberos principals. All of the principals used with Command Center are created in the Greenplum Database Kerberos realm. Command Center users can use the same Kerberos principal to authenticate with Command Center and Greenplum Database.

Command Center Service Principal

A service principal is needed for the Command Center web server. This principal has the format `HTTP/<host>@<realm>`. For example, if users access Command Center at the URL `http://mdw.example.com:28080`, the `<host>` part of the service key is `mdw.example.com` and the `<realm>` part is the Greenplum Database Kerberos realm, for example `GPDB.KRB`.

Note that Kerberos authentication only works if Command Center users enter the host in the same format specified in the Kerberos service principal. If the principal specifies the FQDN, for example, using the host's IP address in the browser URL will not work; the web server will fall back to basic authentication.

Greenplum Database gpmon User

Command Center uses the gpmon Greenplum role to access the gpperfmon database, which contains data presented in the Command Center UI.

You can choose to authenticate the gpmon user with Kerberos or with basic authentication. To use Kerberos, you must create a principal for the gpmon user.

If you choose to use basic authentication you do not need a Kerberos principal for the gpmon user. The gpmon user will authenticate with Greenplum Database using the password saved in the `.pgpass` file in the gpadmin user's home directory on the host running Command Center. If you run Command Center on a host other than the Greenplum Database master host, you must copy the `.pgpass` file from the master host to the Command Center host. See Changing the gpmon Password for instructions to manage the gpmon password.

Command Center Users

Add Kerberos principals for any Command Center users who do not already have principals in the KDC for Greenplum Database.

Adding Kerberos Principals

To add the required principals, perform the following steps as root on the KDC server.

1. Add a principal for the Command Center web service. Be sure to specify the `<gpcc-host>` in the same format that users should enter the host in their browsers.

   ```
   # kadmin.local -q "addprinc -randkey HTTP/<gpcc-host>@<realm>"
   ```

2. If you want the gpmon database user to use Kerberos authentication, add a gpmon principal.

   ```
   # kadmin.local -q "addprinc gpmon@<realm>"
   ```

3. Add principals for any new Command Center users.

   ```
   # kadmin.local -q "addprinc cc_user1@<realm>"
   ```

   Repeat for each new Command Center user.

Set Up Keytab Files

After you have created all of the Kerberos principals needed, you create and distribute keytab files. Keytab files contain Kerberos principals and encrypted keys based on the principals' Kerberos passwords. Keytab files are needed for the Greenplum Database master and standby hosts and the Command Center host.

You can also create a keytab file for each Greenplum Database or Command Center user containing just the user's principal. This keytab file is installed on the user's workstation to enable the user to authenticate to Kerberos. Note that all keytab files must contain the most recent versions of the principals' keys.

Command Center Running on the Greenplum Master Host

If the Greenplum Command Center web server is running on the Greenplum Database master host, Command Center can share the Greenplum Database keytab file. You need to create a keytab file that contains the following principals:

- Service key for the `postgres` process on the Greenplum Database master host, for example `postgres/mdw.example.com@GPDB.EXAMPLE`.
- Service key created for Command Center in the previous section, for example `HTTP/mdw.example.com@GPDB.KRB.`
- A principal for every Kerberos-authenticated Greenplum Database or Command Center user.

All service keys and principals should be in the Greenplum Database realm.

To create a keytab file for Greenplum Database and Command Center, perform the following steps as root on the KDC server.

2. Create a keytab file containing the Greeplum Database service key, the command center service key, and all database and Command Center users.

   ```
   kadmin.local -q "ktadd -k gpdb-kerberos.keytab postgres/mdw.example.com@GPDB.KR
   B HTTP/mdw.example.com@GPDB.KRB"
   ```

   You can enter one or more principals with each `ktadd` command. You can specify a wildcard using the `-glob` option. For example this command adds all principals in the `GPDB.KRB` realm, including service principals and admin users.

   ```
    kadmin.local -q "ktadd -k gpdb-kerberos.keytab -glob *@GPDB.KRB"
   ```

3. Copy the keytab you created to the Greenplum Database master host, replacing the old keytab file. The location of the file is given by the `krb_server_keyfile` parameter in the `$MASTER_DATA_FILE/postgresql.conf` file. Set the permissions on the file so that it can be

read only by the gpadmin user.

4. Update any entries required for new Greenplum Database principals in the `pg_hba.conf` file and `pg_ident.conf` files. See Update the Greenplum Database `pg_hba.conf` File for details.

Command Center Running on the Standby Master

If the Command Center web server is on a different host than the Greenplum Database master, you need separate keytab files for Greenplum Database and Command Center. The keytab file for Greenplum Database may not require any updates, but you will need to create a keytab file for Command Center.

- The Greenplum Database keytab file must contain the Greenplum Database service key and all principals for users with database access.
- The Command Center keytab file contains the Command Center service key and principals for users that have Command Center access. Users with Command Center access must also have Greenplum Database access, so user principals in the Command Center keytab file must also be in the Greenplum Database keytab file.

Update the Greenplum Database keytab if you created new database roles and principals for Command Center. For example, if you want to use Kerberos authentication for the gpmon user, you must create a principal and add it to both the Greenplum Database and Command Center keytab files.

To create the keytab file for Command Center, perform the following steps as root on the KDC host.

1. Create a keytab file and add the Command Center service key.

   ```
   kadmin.local -q "ktadd -k gpcc-kerberos.keytab HTTP/smdw.example.com@GPDB.KRB"
   ```

2. If you want to authenticate the gpmon user with Kerberos, add the gpmon principal.

   ```
   kadmin.local -q "ktadd -k gpcc-kerberos.keytab gpmon@GPDB.KRB"
   ```

3. Add principals for all Command Center users:

   ```
   kadmin.local -q "ktadd -k gpcc-kerberos.keytab cc_user1@GPDB.KRB cc_user2@GPDB.
   KRB"
   ```

   You can enter one or more principals with each `ktadd` command.

4. Enter `quit` to exit `kadmin.local`.
5. Copy the keytab you created to the the host running Command Center, for example:

   ```
   $ scp gpcc-kerberos.keytab gpadmin@<host-name>:/home/gpadmin
   ```

6. Update any entries required for new principals in the `pg_hba.conf` file and `pg_ident.conf` files on the Greenplum master. See Update the Greenplum Database pg_hba.conf File.

Update the Greenplum Database pg_hba.conf File

The Greenplum Database `$MASTER_DATA_DIRECTORY/pg_hba.conf` configuration file determines which authentication methods to use to allow database access.

If you created new Command Center users, you may need to add an entry to allow access via Command Center. The entry for an individual user has this format:

```
host database <user-name> <gpcc CIDR> gss [options]
```

Authentication for the gpmon user needs to be set up in the `pg_hba.conf` file in one of the following ways.

Basic authentication

The `/home/gpadmin/.pgpass` file contains the password for gpmon to use. See Changing the gpmon Password for details. An entry in the `pg_hba.conf` file specifies the md5 authentication method for gpmon:

```
local all gpmon md5
```

Trust authentication

On the Greenplum Database master host only, the gpmon user can access databases without authentication:

```
local all gpmon trust
```

The `/home/gpadmin/.pgpass` file is not needed.

Kerberos authentication

A Kerberos principal has been created for the gpmon user and added to the Greenplum Database and Command Center keytab files.

```
host all gpmon <gpcc CIDR> gss [options]
```

Remove any existing reject rules for gpmon:

```
host all gpmon <auth-method> reject
```

See Using Kerberos Authentication for more information about the `pg_hba.conf` file.

Enable Kerberos for Command Center

Set up Command Center to use the Command Center keytab file you created.

If you are adding Kerberos authentication to an existing Command Center, use the `gpcc` command. For example:

```
$ gpcc --krbenable
```

Enter the Command Center host name and path to the keytab file at the prompts. See the gpcc Reference for more information.

Authenticating With Kerberos on the Client Workstation

To use Kerberos Command Center authentication, the user must have authenticated with Kerberos using the `kinit` command-line tool.

The user then accesses the Command Center web server with a URL containing the host name in the format specified in the Command Center service principal and the port number, for example `http://mdw.example.com:28080`.

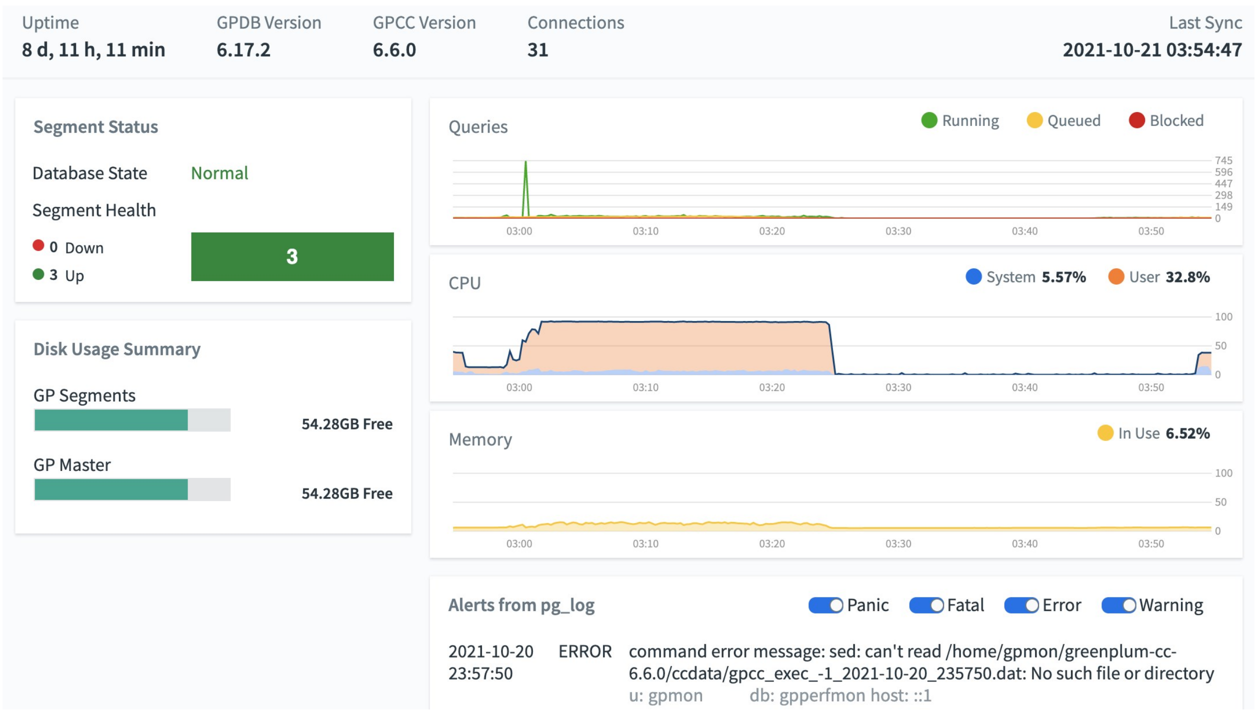
The user's web browser must be configured to use the SPNEGO protocol so that it offers the user's Kerberos principal to the web browser. The method for configuring web browsers varies with different browsers and operating systems. Search online to find instructions to enable SPNEGO with your browser and OS.

Monitoring and Managing the Greenplum Database System

Dashboard

The **Dashboard** displays when you first sign in to Pivotal Greenplum Command Center. The **Dashboard** provides a quick view of the current system status, Segment Health, Queries, CPU, Memory, and Disk usage.

Clicking on a panel provides more detailed information about the metric. The Alerts panel shows the most recent messages from the Greenplum Database log file. Some information is available only to Command Center users with Admin or Operator permission level.



Uptime	GPDB Version	GPCC Version	Connections	Last Sync
8 d, 11 h, 11 min	6.17.2	6.6.0	31	2021-10-21 03:54:47

Segment Status

Database State	Normal
Segment Health	
● 0 Down	3
● 3 Up	

Disk Usage Summary

GP Segments	54.28GB Free
GP Master	54.28GB Free

Queries ● Running ● Queued ● Blocked

CPU ● System 5.57% ● User 32.8%

Memory ● In Use 6.52%

Alerts from pg_log Panic Fatal Error Warning

```
2021-10-20    ERROR    command error message: sed: can't read /home/gpmon/greenplum-cc-
23:57:50               6.6.0/ccdata/gpcc_exec_-1_2021-10-20_235750.dat: No such file or directory
                       u: gpmon         db: gpperfmon host: ::1
```

System Information

The following system information is displayed at the top of the page.

Uptime

The elapsed time since the Greenplum Database system was last started.

GPDB Version

The version of the Greenplum Database software the monitored cluster is running.

GPCC Version

The version of the Greenplum Command Center software.

Connections

The number of active Greenplum Database sessions (client connections).

Server

The display name for this Greenplum Command Center.

Last Sync

Date and time the data was last synchronized. The Command Center user interface updates views with live data every 15 seconds.

Segment Health

The **Segment Health** section of the Dashboard provides a quick overview of the status of the database system and segments this Command Center monitors.

Database State

Database State is the current state of the Greenplum Database system. Following are some possible database states:

- **Normal**: The database is functioning with no major errors or performance issues.
- **Segment(s) Down**: The database is in change-tracking mode or resync mode. Overall performance and system reliability is greatly reduced. See the *Pivotal Greenplum Database System Administrator Guide* for information about resolving this condition.
- **Standby Down**: The master standby instance cannot be reached.
- **Standby Not Syncing**: The master standby instance is not synchronizing with the master instance.
- **Database Unreachable**: The Greenplum Performance Monitor agent cannot connect to the database. The database is likely down. See the *Pivotal Greenplum Database System Administrator Guide* for troubleshooting information.
- **Unbalanced**: Some segments are not running in their preferred roles. That is, primaries are running as mirrors and mirrors are running as primaries, resulting in unbalanced processing.
- **Resyncing**: The database is performing a recovery or rebalance operation.

Note: An error message or state other than the above may be an indication of a network connectivity problem, or of another undetected problem in the Greenplum Database system. Run the `gpstate` utility on the master host to determine if there are issues to troubleshoot in the Greenplum Database system.

Segment Status

The bar graph in the **Segment Status** section shows the up or down status of all database segments in your Pivotal Greenplum Database system. A color indicator and associated number indicate the number of database segments that are currently in that particular state. Segments can have the following states:

- **Up** (Green)
- **Down** (Red)

Clicking the **Segment Status** panel displays the Segment Status Command Center page.

Disk Usage Summary

This chart displays total disk usage and disk available for the Greenplum master host and segment hosts at the last synchronization. Hover over the chart to see the amount of disk used, free, and total.

Queries

This graph displays a summary view of active and queued queries for the last 60 minutes. Click on the colored dot next to the **Running** or **Queued** label to toggle the line on or off. At least one line must be visible at all times. Hover over the graph to display the number of queries for each visible line at that point in time.

CPU

This graph displays average CPU usage across the entire cluster, for the last 60 minutes. The graph displays separate lines for system processes and user processes. The user CPU usage includes the Greenplum database master, standby, and segment processes. Click on the colored dot next to the **System** or **User** label to toggle that line on or off. At least one line must be visible at all times.

Hovering the cursor over a line in the graph displays a small window with the percentage of CPU used at that point in time for the visible lines and the total if both the system and user lines are visible.

Memory

This graph displays the average percent of memory used across the entire cluster over the last 60 minutes. Hover over the line to display the percent of memory used at that point in time.

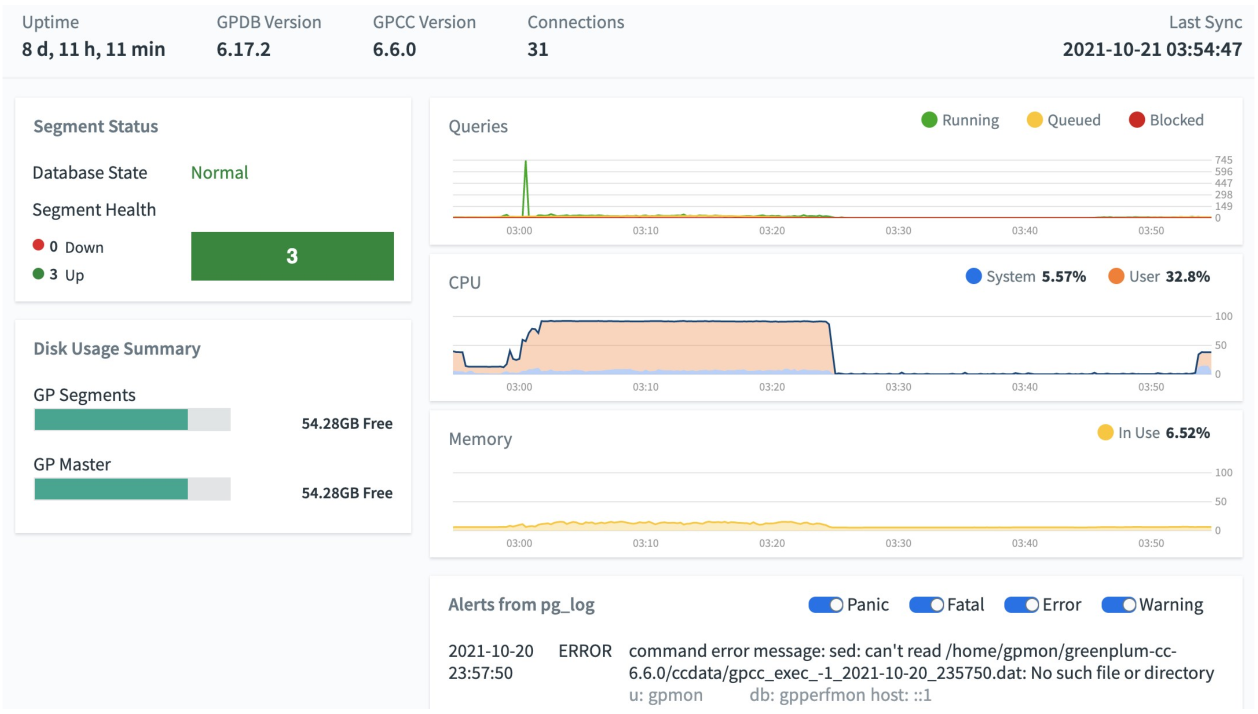
Alerts

Admin and Operator permission levels only

The **Alerts** panel displays recent messages from the Greenplum Database `pg_log` log file. The panel is updated at each synchronization. Filter the messages by severity level using the controls at the top right of the panel.

Greenplum Database Cluster State

The Greenplum Command Center **Dashboard** provides a quick view of the current system status, Segment Health, Queries, CPU, Memory, and Disk usage. Clicking on a panel provides more detailed information about that metric. The **Alerts** panel shows the most recent messages from the Greenplum Database log file. Some information is available only to Command Center users with Admin or Operator permission level.



```
2021-10-20    ERROR   command error message: sed: can't read /home/gpmon/greenplum-cc-
23:57:50              6.6.0/ccdata/gpcc_exec_-1_2021-10-20_235750.dat: No such file or directory
                      u: gpmon       db: gpperfmon host: ::1
```

System Information

The following system information is displayed at the top of the page.

Uptime

The elapsed time since the Greenplum Database system was last started.

GPDB Version

The version of the Greenplum Database software the monitored cluster is running.

GPCC Version

The version of the Greenplum Command Center software.

Connections

The number of active Greenplum Database sessions (client connections).

Server

The display name for this Greenplum Command Center.

Last Sync

Date and time the data was last synchronized. The Command Center user interface updates views with live data every 15 seconds.

Segment Health

The **Segment Health** section of the Dashboard provides a quick overview of the status of the database system and segments this Command Center monitors.

Database State

Database State is the current state of the Greenplum Database system. The state can be one of the following:

- **Normal**: The database is functioning with no major errors or performance issues.
- **Segment(s) Down**: The database is in change-tracking mode or resync mode. Overall performance and system reliability is greatly reduced. See the *Pivotal Greenplum Database System Administrator Guide* for information about resolving this condition.
- **Database Unreachable**: The Greenplum Performance Monitor agent cannot connect to the database. The database is likely down. See the *Pivotal Greenplum Database System Administrator Guide* for troubleshooting information.
- **Unbalanced**: Some segments are not running in their preferred roles. That is, primaries are running as mirrors and mirrors are running as primaries, resulting in unbalanced processing.
- **Resyncing**: The database is performing a recovery or rebalance operation.

Segment Status

The bar graph in the **Segment Status** section shows the up or down status of all database segments in your Pivotal Greenplum Database system. A color indicator and associated number indicate the number of database segments that are currently in that particular state. Segments can have the following states:

- **Up** (Green)
- **Down** (Red)

Clicking the **Segment Status** panel displays the Segment Status Command Center page.

Disk Usage Summary

This chart displays total disk usage and disk available for the Greenplum master host and segment hosts at the last synchronization. Hover over the chart to see the amount of disk used, free, and total.

Queries

This graph displays a summary view of active and queued queries for the last 60 minutes. Click on the colored dot next to the **Running** or **Queued** label to toggle the line on or off. At least one line must be visible at all times. Hover over the graph to display the number of queries for each visible line at that point in time.

CPU

This graph displays average CPU usage across the entire cluster, for the last 60 minutes. The graph displays separate lines for system processes and user processes. The user CPU usage includes the Greenplum database master, standby, and segment processes. Click on the colored dot next to the **System** or **User** label to toggle that line on or off. At least one line must be visible at all times.

Hovering the cursor over a line in the graph displays a small window with the percentage of CPU used at that point in time for the visible lines and the total if both the system and user lines are visible.

Memory

This graph displays the average percent of memory used across the entire cluster over the last 60 minutes. Hover over the line to display the percent of memory used at that point in time.

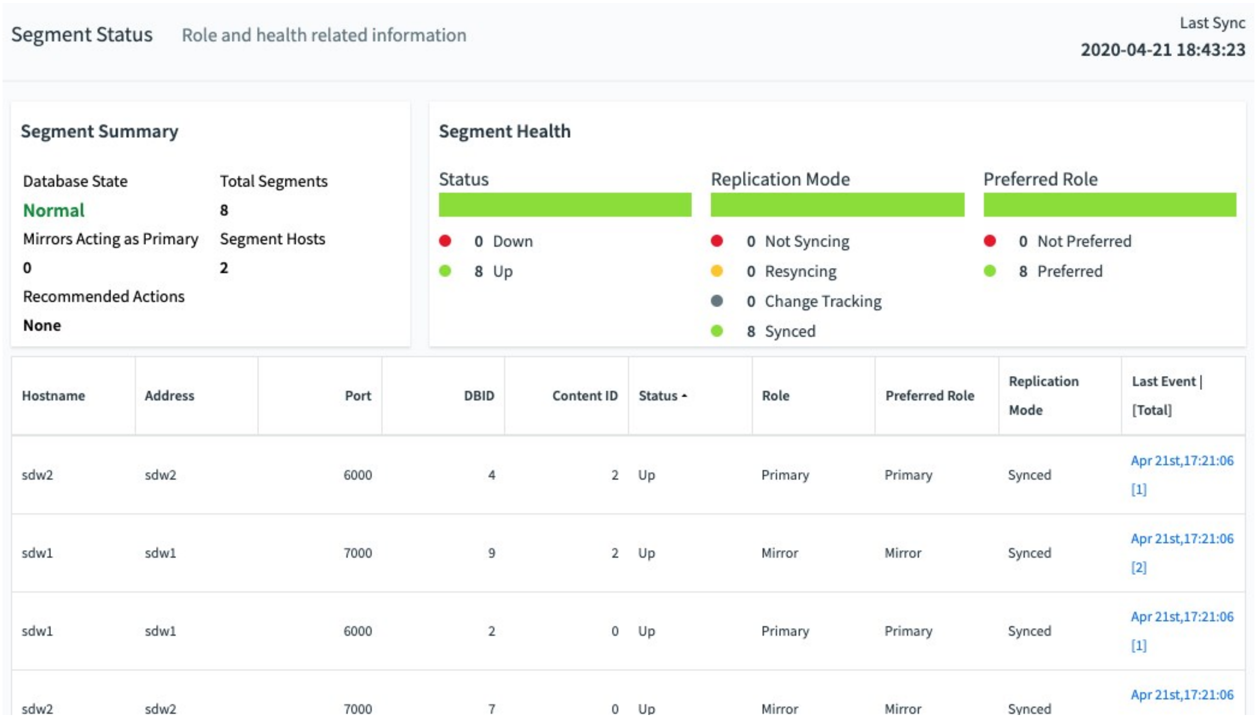
Alerts

Admin and Operator permission levels only

The **Alerts** panel displays recent messages from the Greenplum Database `pg_log` log file. The panel is updated at each synchronization. Filter the messages by severity level using the controls at the top right of the panel.

Segment Status

The **Segment Status** page provides a health overview for the Greenplum Database segments and details for each primary and mirror segment.



Segment Status — Role and health related information

Last Sync
2020-04-21 18:43:23

Segment Summary

Database State	Total Segments
Normal	8
Mirrors Acting as Primary	Segment Hosts
0	2
Recommended Actions	
None	

Segment Health

Status	Replication Mode	Preferred Role
● 0 Down	● 0 Not Syncing	● 0 Not Preferred
● 8 Up	● 0 Resyncing	● 8 Preferred
	● 0 Change Tracking	
	● 8 Synced	

Hostname	Address	Port	DBID	Content ID	Status	Role	Preferred Role	Replication Mode	Last Event | [Total]
sdw2	sdw2	6000	4	2	Up	Primary	Primary	Synced	Apr 21st,17:21:06 [1]
sdw1	sdw1	7000	9	2	Up	Mirror	Mirror	Synced	Apr 21st,17:21:06 [2]
sdw1	sdw1	6000	2	0	Up	Primary	Primary	Synced	Apr 21st,17:21:06 [1]
sdw2	sdw2	7000	7	0	Up	Mirror	Mirror	Synced	Apr 21st,17:21:06

Segment Summary

Greenplum Database is most efficient when all segments are operating in their preferred roles. The

Segment Summary panel tells you the overall segment status and if any mirrors are acting as primaries.

The **Segment Summary** panel provides the following information:

Database State

The database state can be one of the following:

- **Normal**: The database is functioning with no major errors or performance issues.
- **Segment(s) Down**: The database is in change-tracking mode or resync mode. Overall performance and system reliability is greatly reduced. See the *Pivotal Greenplum Database System Administrator Guide* for information about resolving this condition.
- **Database Unreachable**: The Greenplum Performance Monitor agent cannot connect to the database. The database is likely down. See the *Pivotal Greenplum Database System Administrator Guide* for troubleshooting information.
- **Unbalanced**: Some segments are not running in their preferred roles. That is, primaries are running as mirrors and mirrors are running as primaries, resulting in unbalanced processing.
- **Resyncing**: The database is performing a recoverty or rebalance operation.

Mirrors Acting as Primary

The number of mirror segments acting as primary segments.

Recommended Actions

Suggests actions to perform to restore the cluster to balance. These include:

- Recover and Rebalance
- Rebalance

These actions are executed from the command line using the `gprecoverseg` Greenplum management utility. See `gprecoverseg` in the *Pivotal Greenplum Database Utility Reference* for more information.

Total Segments

The total number of primary and mirror segments in the Greenplum cluster.

Segment Hosts

The total number of segment hosts in the Greenplum cluster.

Segment Health

The **Segment Health** panel contains charts for Greenplum Database segments' status, replication mode, and preferred roles.

Status

Numbers of segments that are down and up.

Replication Mode

A chart that shows the number of segments in each of the possible replication modes.

- Not Syncing: The primary segment and mirror segment are active and all changes to the primary segment have been copied to the mirror using a file block replication process.

- Change Tracking: If a primary segment is unable to copy changes to its mirror segment using the file replication process, it logs the unsent changes locally so they can be replicated when the mirror again becomes available. This can happen if a mirror segment goes down or if a primary segment goes down and its mirror segment automatically assumes the primary role.
- Resyncing: When a down segment is brought back up, administrators initiate a recovery process to return it to operation. The recovery process synchronizes the segment with the active primary and copies the changes missed while the segment was down.
- Synced: Once all mirrors and their primaries are synchronized, the system state becomes synchronized.

Preferred Roles

The red portion of the Preferred Role chart shows the numbers of segments that not operating in their preferred primary or mirror roles. If the chart is not solid green, the performance of the Greenplum cluster is not optimal.

Primary and mirror segments are distributed evenly among the segment hosts to ensure that each host performs an equivalent share of the work and primary segments and their mirror segments reside on different segment hosts. When a primary segment goes down, its mirror on another host in the cluster automatically assumes the primary role, increasing the number of primary segments running on that host. This uneven distribution of the workload will affect query performance until the down segment is restored and the segments are returned to their original, preferred, roles.

Segment Table

The table at the bottom of the **Segment Status** page contains a detailed row for every primary and mirror segment in the Greenplum Cluster. The table has the following columns for each segment:

Hostname

The name of the segment host where the segment is running.

Address

The network interface on the segment host for the segment.

Port

The port number assigned to the segment.

DBID

The unique identifier for the segment instance.

ContentID

The content identifier for the segment, from 0 to the number of segments minus 1. A primary segment and its mirror have the same ContentID. The master and standby master, which have ContentID −1, are excluded from the table.

Status

"UP" if the segment is running, "DOWN" if the segment has failed or is unreachable.

Role

The segment's current role, either "primary" or "mirror".

Preferred Role

The segment's intended role, either "primary" or "mirror".

Replication Mode

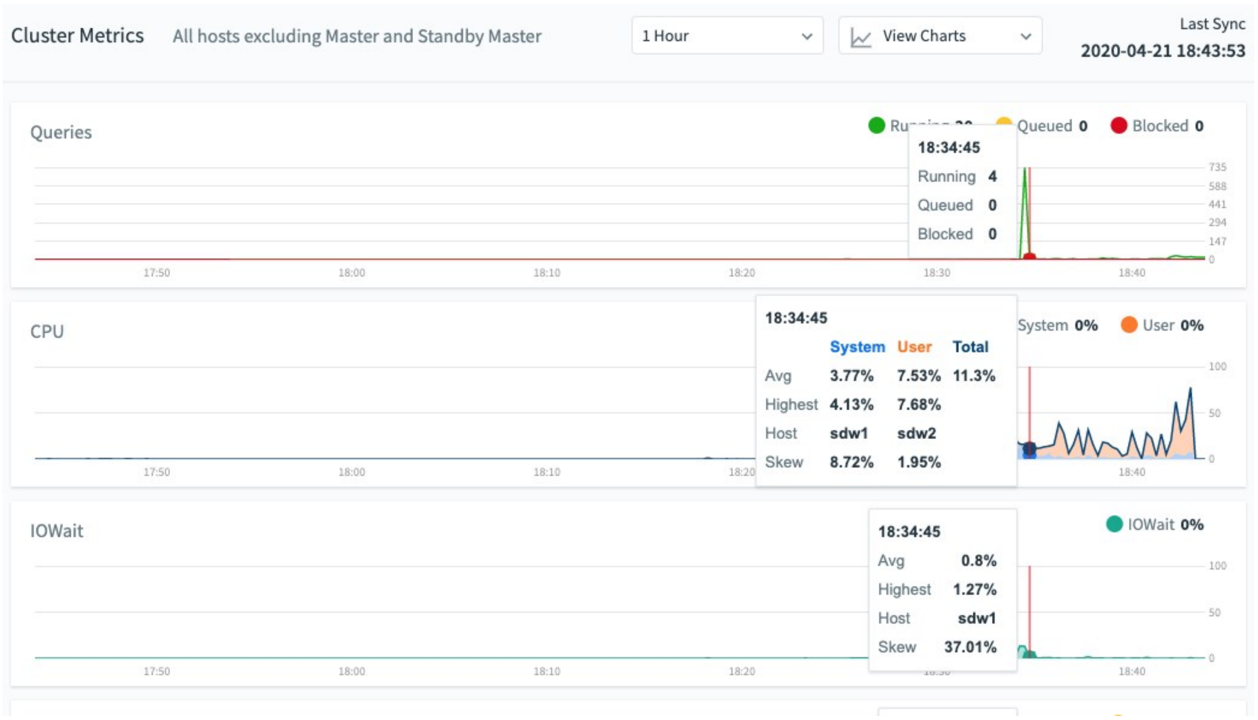
The replication status for the segment. See Segment Health for possible values.

Last Event|[Total]

The date and time of last segment health-related activity. Click to display a list of recent events.

Cluster Metrics

The **Cluster Metrics** page shows consolidated statistics for all segment hosts in the Greenplum cluster. Master and standby master hosts are excluded from the metrics.



18:34:45

Running	4
Queued	0
Blocked	0

18:34:45

	System	User	Total
Avg	3.77%	7.53%	11.3%
Highest	4.13%	7.68%	
Host	sdw1	sdw2	
Skew	8.72%	1.95%	

18:34:45

Avg	0.8%
Highest	1.27%
Host	sdw1
Skew	37.01%

Use the **Show/hide Charts** control to choose which charts to display.

Choose the time period to display using the control in the top right corner of the screen: 1 hour, 4 hours, 1 day, or 3 days. The right side of the chart corresponds to the **Last Sync** time shown in the top right of the page.

Hover over any of the charts to see values for that point in time in pop-up boxes. The charts are synchronized so that hovering over any chart shows the same point in time in all charts.

- The master and standby master hosts are excluded from the metrics.
- When the metric is an average of values collected from all segment hosts, the host with the highest value is identified in the pop-up box.
- Solid lines trace the average value for all segment hosts. The stacked graphs represent the highest values for the metric at each point in time.

- The current value of a metric is shown in the upper right corner of its chart.

The page has charts for the following metrics:

Queries

The number of queries running and the number of queries queued to run.

CPU

The percentage CPU used by system processes and the percentage CPU used by user processes. The chart represents the total of system (including IOWait) and user CPU usage across the hosts. The system and user average CPU values that are shown in the pop-up box are not seperately represented on the chart.

IOWait

The percentage of CPU used waiting on IO requests.

Memory

Percentage of memory in use.

Memory is calculated as follows:

Total = MemTotal
Free = MemFree + Buffers + Cached
Used = MemTotal - Free

Disk I/O

Disk read and write rates in megabytes per second.

Network

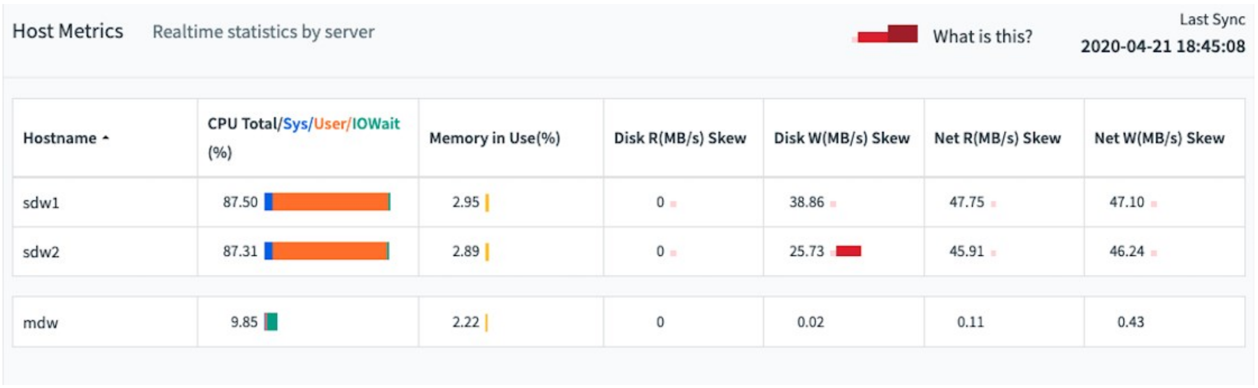
Network I/O read and write rates in megabytes per second. Network metrics include traffic over all NICs (network interface cards), including internal interconnect and administrative traffic. Load

System load average for 1-minute, 5-minute, and 15-minute periods.
Swap

Percentage of swap space used.

Host Metrics

The **Host Metrics** page displays a table of the hosts in the cluster with statistics collected at the most recent quantum interval.



Host Metrics Realtime statistics by server

What is this?

Last Sync
2020-04-21 18:45:08

Hostname	CPU Total/Sys/User/IOWait (%)	Memory in Use(%)	Disk R(MB/s) Skew	Disk W(MB/s) Skew	Net R(MB/s) Skew	Net W(MB/s) Skew
sdw1	87.50	2.95	0	38.86	47.75	47.10
sdw2	87.31	2.89	0	25.73	45.91	46.24
mdw	9.85	2.22	0	0.02	0.11	0.43

At the top right, **Last Sync** displays the time the statistics were last updated.

Click a column header to sort the table by that column. Click again to toggle between ascending and descending sort. Master and standby hosts are not included in the sort and are always displayed following the sorted list of segment hosts.

For each server, the following columns are displayed:

Hostname

The hostname name of the server. CPU Total/Sys/User/IOWait (%)

The CPU column shows the total percentage of CPU used for system processes, user processes, and IOWait. Hover over the table cell to show the percentages for each of these CPU modes, as well as the idle time.
When comparing Command Center CPU values with other unix utilities like `top`, there are slight differences.
To compare Command Center CPU user time with `top`, adjust the `top` CPU user time to be: CPU user time + `nice` time. To compare CPU sys time, before Command Center 4.9/6.1, adjust the `top` utility CPU sys time to be: sys time + iowait time. After 4.10/6.2, the values are similar, for example CPU sys = sys and CPU IOWait = iowait.

Memory In Use (%)

The percentage of host memory in use is displayed next to a graph illustrating the memory in use and available. Hover over the table cell to see memory used and available in gigabytes.

Memory is calculated as follows:
Total = MemTotal
Free = MemFree + Buffers + Cached
Used = Total - Free

Disk R (MB/s) | Skew

Disk read rate in megabytes per second is displayed next to a graph of calculated disk read skew. Hover over the table cell to see a Low/Medium/High rating for disk skew.

Disk W (MB/s) | Skew

Disk write rate in megabytes per second is displayed next to a graph of calculated disk write skew. Hover over the table cell to see a Low/Medium/High rating for disk write skew.

Net R (MB/s) | Skew

Network read rate in megabytes per second is displayed next to a graph of calculated network read skew. Hover over the table cell to see a Low/Medium/High rating for network read skew.

Net W (MB/s) | Skew

Network write rate in megabytes per second is displayed next to a graph of calculated network write skew. Hover over the table cell to see a Low/Medium/High rating for network write skew.

About Skew Calculations

Disk and Network skew ratings are calculated as each server's standard deviation from the mean calculated from all segment hosts.

Low

Value is within 1 standard deviation from the mean. (Note: if the variance of the set is less than 3, skew is considered low regardless of deviation from mean.)

Moderate

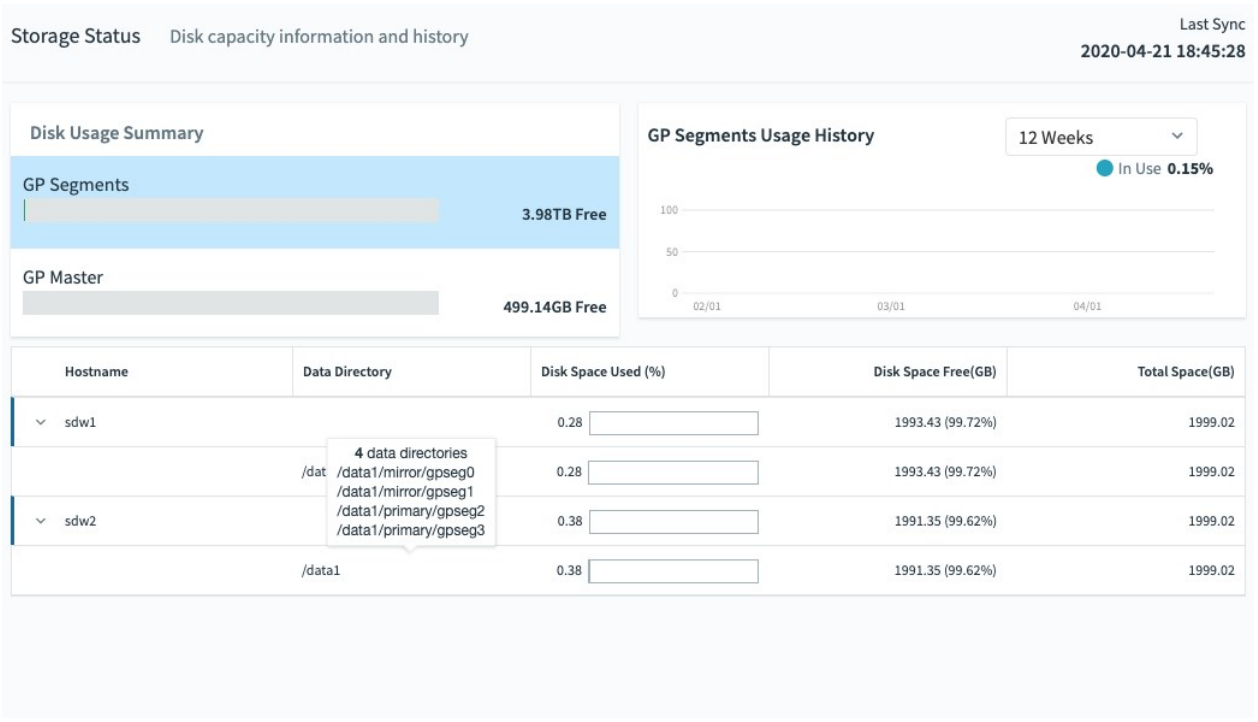
Value is between 1 and 2 standard deviations from the mean.

Very High

Value is greater than 3 standard deviations from the mean.

Storage Status

The **Storage Status** page shows current historical disk usage for Greenplum master and segment hosts.


Storage Status Disk capacity information and history

Last Sync 2020-04-21 18:45:28

Disk Usage Summary

GP Segments 3.98TB Free

GP Master 499.14GB Free

GP Segments Usage History 12 Weeks

In Use 0.15%

Hostname	Data Directory	Disk Space Used (%)	Disk Space Free(GB)	Total Space(GB)
sdw1		0.28	1993.43 (99.72%)	1999.02
	/dat	0.28	1993.43 (99.72%)	1999.02
sdw2		0.38	1991.35 (99.62%)	1999.02
	/data1	0.38	1991.35 (99.62%)	1999.02

4 data directories
/data1/mirror/gpseg0
/data1/mirror/gpseg1
/data1/primary/gpseg2
/data1/primary/gpseg3

Disk Usage Summary

You can see current disk space in use, space free, and total space in the **Disk Usage Summary** panel. Disk space metrics for the segment hosts (**GP Segments**) and the master (**GP Master**) are shown in separate bar charts.

The **GP Segments** bar chart shows combined disk space for all segments.

The **GP Masters** bar chart shows combined disk space for master and standby master.

Click either of the summary charts to see details by host: space used, free, and total in gigabytes and as a percentage of the total.

Click a host name to display a list of directories on the host file system where partitions containing one or more Greenplum Database data directories are mounted. Hover over the directory name to see a list of the Greenplum data directories the partition contains.

Note: Newly added data directories (a newly created tablespace, for example) are not immediately updated, but will be refreshed within four hours.

GP Segments Usage History

The **GP Segments Usage History** panel presents a chart of percentage of disk space in use for the time period set by the control in the panel header.

Hover over the chart to see the percentage disk in use by all Greenplum Database segments at any given point in time.

GP Masters Usage History

The **GP Masters Usage History** panel presents a chart of percentage of disk space in use by the master and standby masters for the time period set by the control in the panel header.

Hover over the chart to see the percentage disk in use at any given point in time.

Recommendations

Use the **Recommendations** page to find tables that require maintenance. Command Center users with Admin permission can make changes to this page.

Scheduling Scans

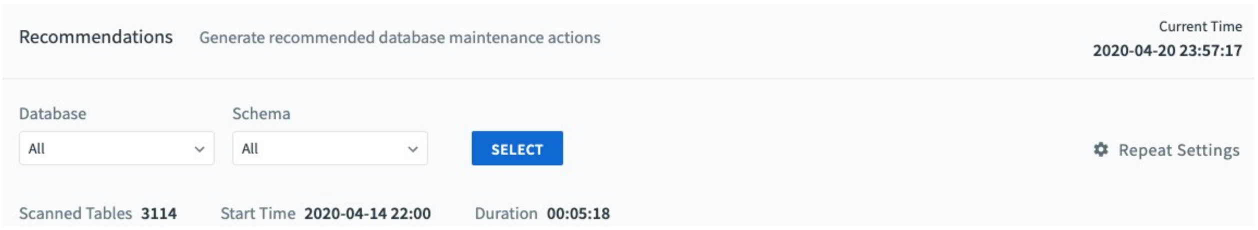
Schedule a period of time during which Command Center scans databases for tables that need to be vacuumed, analyzed, or redistributed.

Command Center scans tables of type heap, append-optimized (AO), and append-optimized/column-oriented (AO/CO) that have size greater than zero. Views and external tables are not scanned.

Tables that have been updated recently or have never been scanned are scanned first.

It is best to schedule the scan during a down-time or a period of low activity to minimize the effect of the scan on database performance.

You can set the date and time the scan begins and the duration of the scan. Check **Repeat** to set the scan to repeat daily, or weekly on selected days of the week.


Recommendations Generate recommended database maintenance actions

Current Time 2020-04-20 23:57:17

Database: All

Schema: All

Scanned Tables 3114   Start Time 2020-04-14 22:00   Duration 00:05:18

Command Center starts scanning automatically at the scheduled time. A message is displayed on every Command Center page while the scan is in progress. It may not be possible to scan all tables in the scheduled period, but when the next scan begins, Command Center resumes scanning where the previous scan left off, until all tables have been scanned. Then the scan starts over.

The recommendation tables always list results from the most recent scan.

Click **Repeat Settings** to view a history of scans or to update the scan schedule.

Choose a database and schema and click **SELECT** to view recommendations. Recommendations are reported in three reports:

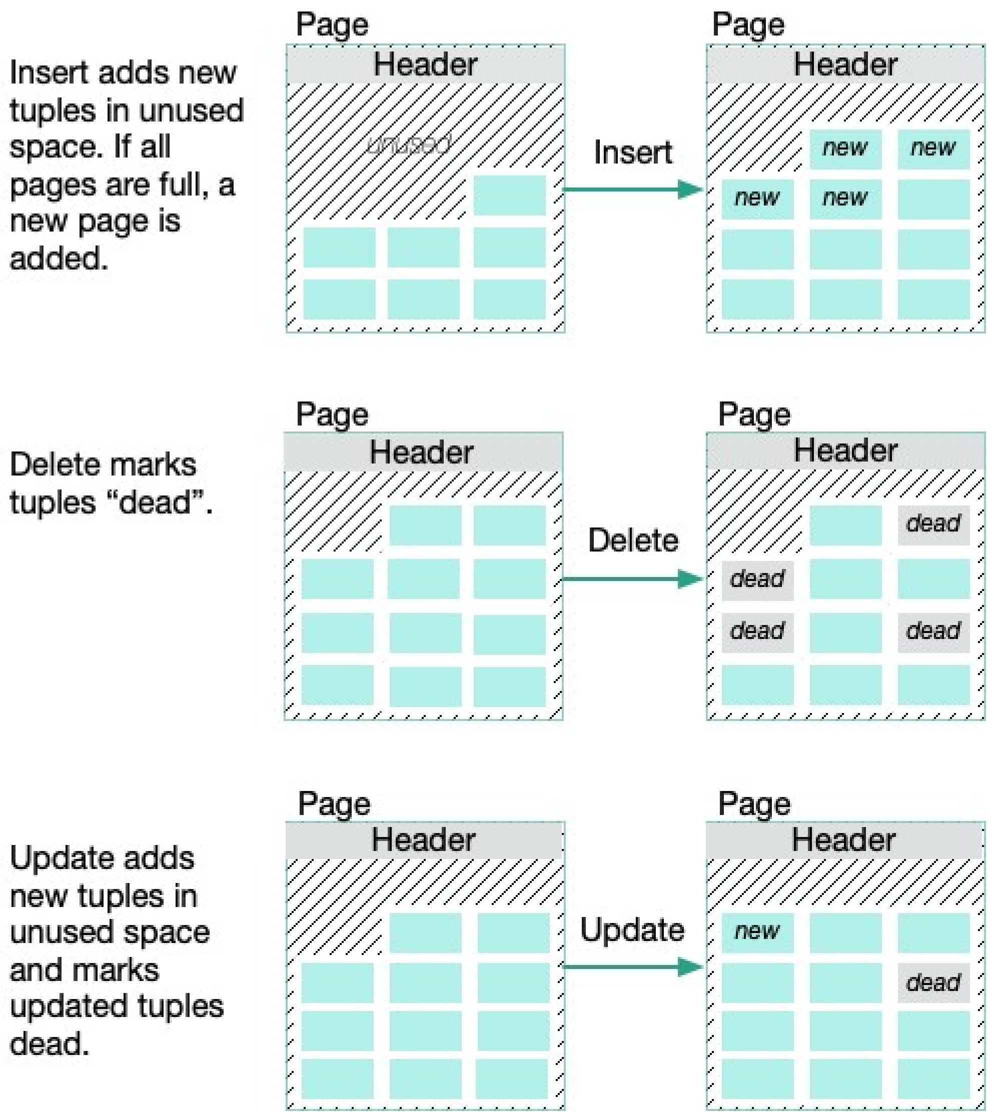
- Bloat
- Accuracy
- Skew

Bloat

Bloat is a measure of the amount of unused space and space occupied by dead tuples in a table.

Greenplum Database uses PostgreSQL multi-version concurrency control (MVCC) to manage concurrent transactions. Table storage is organized into fixed size pages containing a header, an array of item pointers, unused space, and tuples (rows).

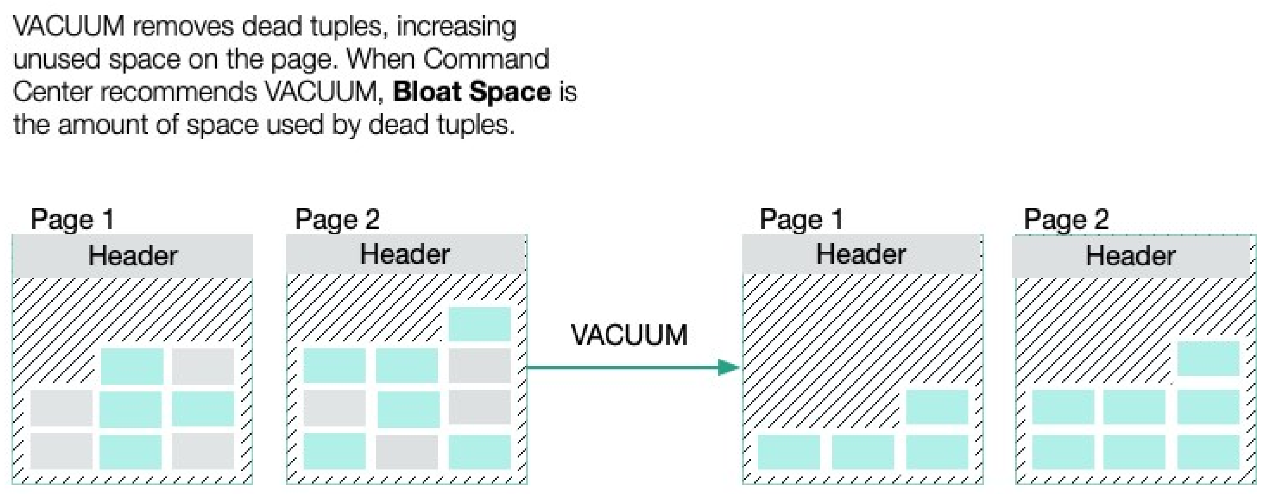
- Inserting a tuple into a table adds a new tuple and reduces the amount of unused space on the page. Tuples are allocated from the end of the unused space.
- Deleting tuples marks them dead, but does not remove them from the page.
- Updating a tuple inserts a new live tuple and marks the original tuple dead.
- New pages are added to the table when there is insufficient unused space to add new tuples.

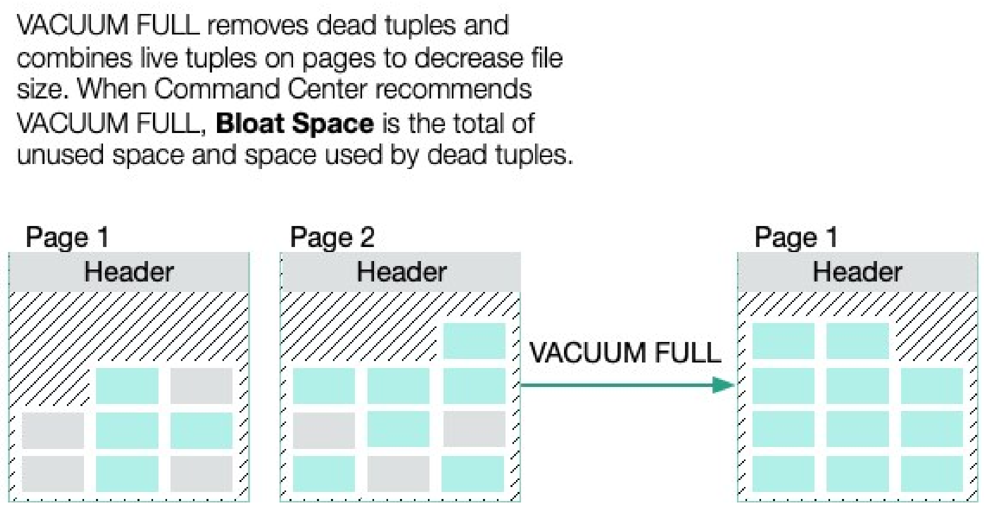

Insert adds new tuples in unused space. If all pages are full, a new page is added.
Page
Header
unused
Insert
Page
Header
new
new
new
new
Delete marks tuples "dead".
Page
Header
Delete
Page
Header
dead
dead
dead
dead
Update adds new tuples in unused space and marks updated tuples dead.
Page
Header
Update
Page
Header
new
dead

Over time, the dead tuples accumulate, consuming a greater proportion of the table. New pages are added when unused space is depleted, increasing the file size of the table. Scanning a table with a large amount of bloat requires additional I/O, and has a detrimental effect on query performance.

The `VACUUM` command removes the dead tuples, increasing the amount of unused space available for new live tuples. `VACUUM` only reduces the size of the table on disk if the unused space is at the end of the table. Command Center by default recommends vacuuming a table when dead tuples occupy more than 10% of the space. You can specify a different threshold by setting the `bloat_threshold` parameter in the gpmetrics configuration file. See the gpmetrics Configuration File Reference for details.

The `VACUUM FULL` command removes dead tuples and compacts the table to the mininum number of pages by packing live tuples on pages. By default, Command Center recommends `VACUUM FULL` for heap tables when dead tuples and unused space occupy greater than 50% of the space. You can specify a different threshold for the `VACUUM FULL` recommendation by setting the `bloat_vacuum_full_threshold` parameter in the gpmetrics configuration file. See the gpmetrics Configuration File Reference for details.


VACUUM removes dead tuples, increasing unused space on the page. When Command Center recommends VACUUM, **Bloat Space** is the amount of space used by dead tuples.

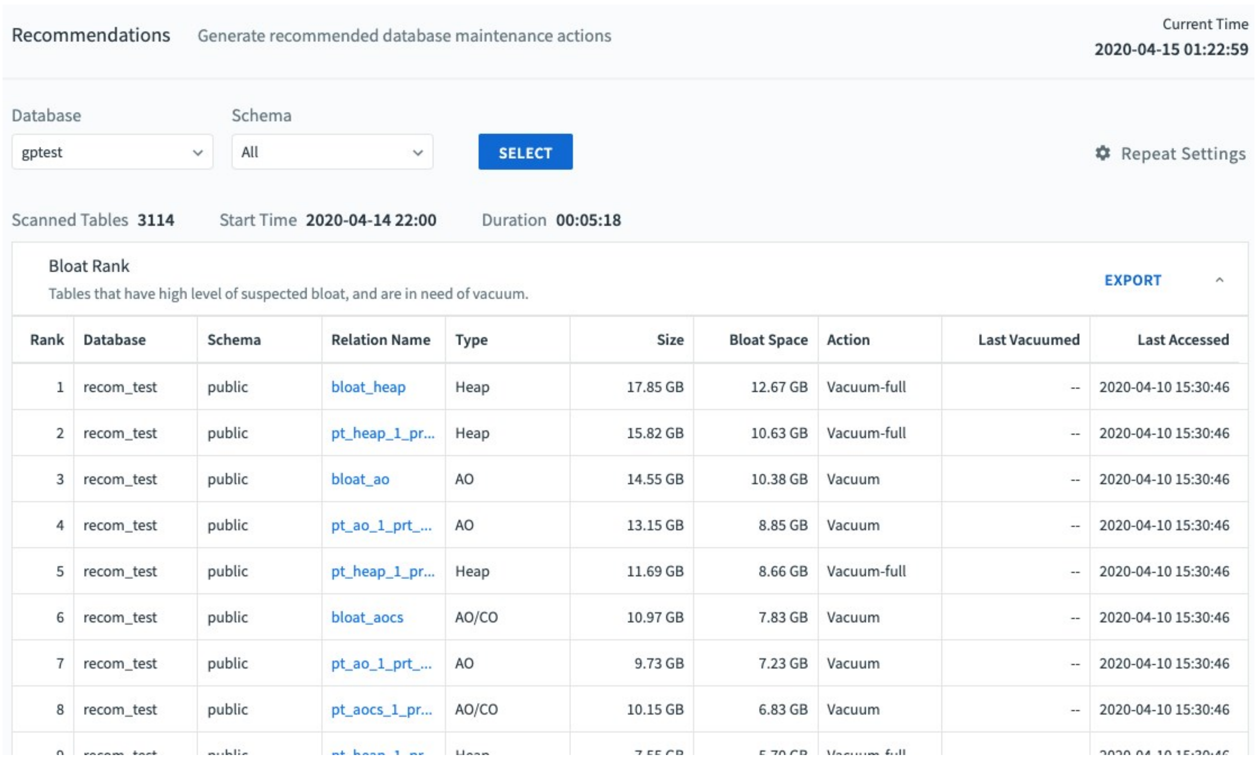

VACUUM FULL removes dead tuples and combines live tuples on pages to decrease file size. When Command Center recommends VACUUM FULL, **Bloat Space** is the total of unused space and space used by dead tuples.

For append-optimized (AO) and append-optimized/column-oriented (AO/CO) tables, Command Center recommends `VACUUM`. Vacuuming AO and AO/CO tables has the same effect as `VACUUM FULL` for heap tables.

`VACUUM FULL` is an expensive operation, requiring an `ACCESS EXCLUSIVE` lock and additional disk space, and it should be avoided, especially for very large tables. Running `VACUUM` frequently can prevent the need to run `VACUUM FULL`.

For more information about `VACUUM` and `VACUUM FULL`, see VACUUM in the Greenplum Database SQL Commands Reference.

The **Bloat Rank** table lists tables by the estimated amount of bloat space, from greatest to least.


Recommendations Generate recommended database maintenance actions

Current Time 2020-04-15 01:22:59

Database: gptest Schema: All SELECT Repeat Settings

Scanned Tables 3114 Start Time 2020-04-14 22:00 Duration 00:05:18

Bloat Rank

Tables that have high level of suspected bloat, and are in need of vacuum.

EXPORT

Rank	Database	Schema	Relation Name	Type	Size	Bloat Space	Action	Last Vacuumed	Last Accessed
1	recom_test	public	bloat_heap	Heap	17.85 GB	12.67 GB	Vacuum-full	--	2020-04-10 15:30:46
2	recom_test	public	pt_heap_1_pr...	Heap	15.82 GB	10.63 GB	Vacuum-full	--	2020-04-10 15:30:46
3	recom_test	public	bloat_ao	AO	14.55 GB	10.38 GB	Vacuum	--	2020-04-10 15:30:46
4	recom_test	public	pt_ao_1_prt_...	AO	13.15 GB	8.85 GB	Vacuum	--	2020-04-10 15:30:46
5	recom_test	public	pt_heap_1_pr...	Heap	11.69 GB	8.66 GB	Vacuum-full	--	2020-04-10 15:30:46
6	recom_test	public	bloat_aocs	AO/CO	10.97 GB	7.83 GB	Vacuum	--	2020-04-10 15:30:46
7	recom_test	public	pt_ao_1_prt_...	AO	9.73 GB	7.23 GB	Vacuum	--	2020-04-10 15:30:46
8	recom_test	public	pt_aocs_1_pr...	AO/CO	10.15 GB	6.83 GB	Vacuum	--	2020-04-10 15:30:46
9	recom_test	public	pt_heap_1_pr...	Heap	[illegible]	[illegible]	Vacuum-full		[illegible]

The **Bloat** table has these columns:

Rank

Tables in the list are ranked by the estimated amount of bloat

Database

Name of the database containing the table

Schema

Name of the schema to which the table belongs

Relation Name

Name of the table

Type

Storage type for the table, for example heap or AO (append-optimized) Size

Total size of the table on all segments

Bloat Space

The amount of space that can be freed with the recommended action. If the recommended action is `VACUUM`, **Bloat Space** is the amount of space occupied by dead rows. If `VACUUM FULL` is recommended, **Bloat Space** is the total unused space and space occupied by dead rows.

Action

Running `VACUUM FULL` decreases the size of the table and is recommended only when the table size would be reduced by at least 50%. Running `VACUUM` removes only dead tuples, increasing unused space.

Last Vacuumed

Time the table was last vacuumed. This is the end of the Command Center sampling period during which the vacuum completed.

Last Accessed

Time the table was last updated (`DELETE`, `INSERT`, or `UPDATE` operation).

Accuracy

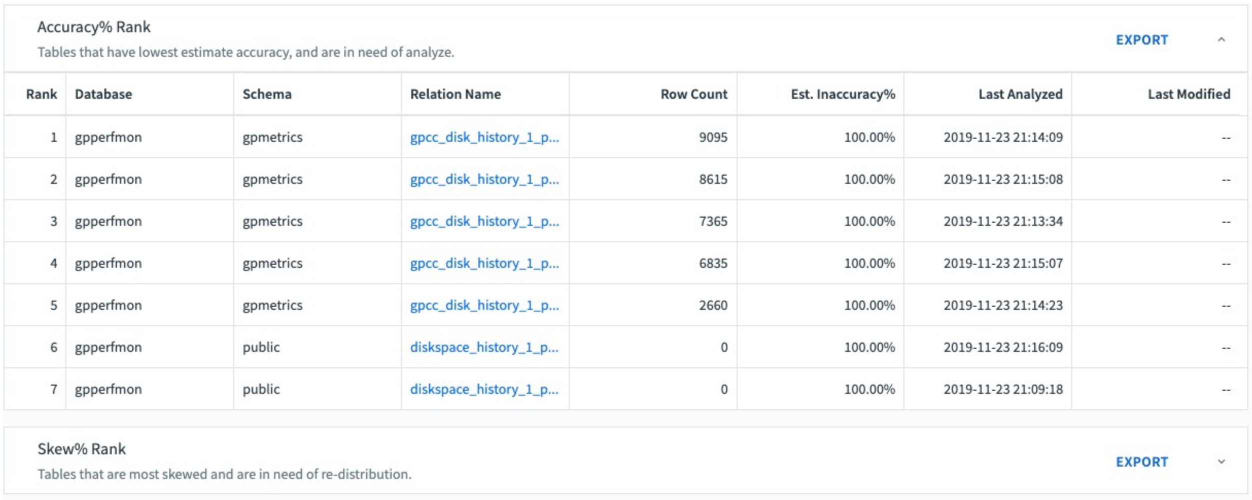
The **Accuracy% Rank** table lists tables ranked by **Est.Inaccuracy**. A higher **Est.Inaccuracy** is an indication that you should run `ANALYZE` on the table to update the table statistics. When statistics are inaccurate, the optimizer can generate an inefficient query plan.

The **Est.Inaccuracy** metric is calculated from query history by comparing the query plan estimate of rows and the actual number of rows returned by table scans. The calculation includes scan operations (query plan scan nodes) for queries that completed in the last 30 days and that started after the scanned table was last analyzed. The formula for inaccuracy is:

```
inaccuracy = (MAX(plan_rows, actual_rows) - MIN(plan_rows, actual_rows)) / MAX(plan_ro
ws, actual_rows)
```

Scans without conditions (`WHERE` clauses) are considered more predictive of row count accuracy, so if the query history has both scans without conditions and scans with conditions, then scans without conditions are weighted more heavily.

By default, the **Accuracy%** report includes tables with a calculated inaccuracy of at least 10%. You can specify a different threshold by setting the `accuracy_threshold` parameter in the gpmetrics configuration file. See the gpmetrics Configuration File Reference for details.


Accuracy% Rank

Tables that have lowest estimate accuracy, and are in need of analyze.

EXPORT

Rank	Database	Schema	Relation Name	Row Count	Est. Inaccuracy%	Last Analyzed	Last Modified
1	gpperfmon	gpmetrics	gpcc_disk_history_1_p...	9095	100.00%	2019-11-23 21:14:09	--
2	gpperfmon	gpmetrics	gpcc_disk_history_1_p...	8615	100.00%	2019-11-23 21:15:08	--
3	gpperfmon	gpmetrics	gpcc_disk_history_1_p...	7365	100.00%	2019-11-23 21:13:34	--
4	gpperfmon	gpmetrics	gpcc_disk_history_1_p...	6835	100.00%	2019-11-23 21:15:07	--
5	gpperfmon	gpmetrics	gpcc_disk_history_1_p...	2660	100.00%	2019-11-23 21:14:23	--
6	gpperfmon	public	diskspace_history_1_p...	0	100.00%	2019-11-23 21:16:09	--
7	gpperfmon	public	diskspace_history_1_p...	0	100.00%	2019-11-23 21:09:18	--

Skew% Rank

Tables that are most skewed and are in need of re-distribution.

EXPORT

The **Accuracy%** table has these columns:

Rank

Tables in the list are ranked by **Est.Accuracy%**, from least to greatest.

Database

Name of the database containing the table

Schema

Name of the schema to which the table belongs

Relation Name

Name of the table

Row Count

Number of rows in the table

Est.Accuracy%

Estimate of statistics accuracy from 0.0 to 1.0 Last Analyzed

Time the table was last analyzed. This is the end of the Command Center sampling period during which the analyze operation was completed.

Last Modified

The most recent timestamp among `last_ins`, `last_del`, and `last_upd`.

Skew

Skew is a measure of how evenly table rows are distributed to Greenplum Database segments. When some tables are distributed unevenly, queries can take longer to complete.

Skew% is a value between between 0% and 100%, inclusive. The higher the value, the more skew is present in the distribution of the table. Given a table T distributed among *n* segments, S_1_ is the number of tuples on segment *1* and S_n_ is the number of tuples on segment *n*:

```
Avg = (S1 + S2 + ... Sn) / n
Max = MAX(S1, S2, ... Sn)
Skew% = (1 - Avg / Max) / (1 - 1/n) * 100%
```

A high **Skew%** for a table is usually caused by a poor choice of distribution key. The distribution key should be a column (or columns) with high cardinality, such as a unique ID. Columns that have a limited number of possible values make poor distribution keys and will result in skew. You can distribute the table `RANDOMLY` to ensure an even distribution if there is no suitable column.

If Commmand Center recommends redistributing a table, use the `ALTER TABLE` command:

```
ALTER TABLE <table-name> SET DISTRIBUTED BY (<column>, ...);
```

The table will be redistributed using the new distribution key.

Use this command to change the distribution policy for a table to `RANDOMLY`:

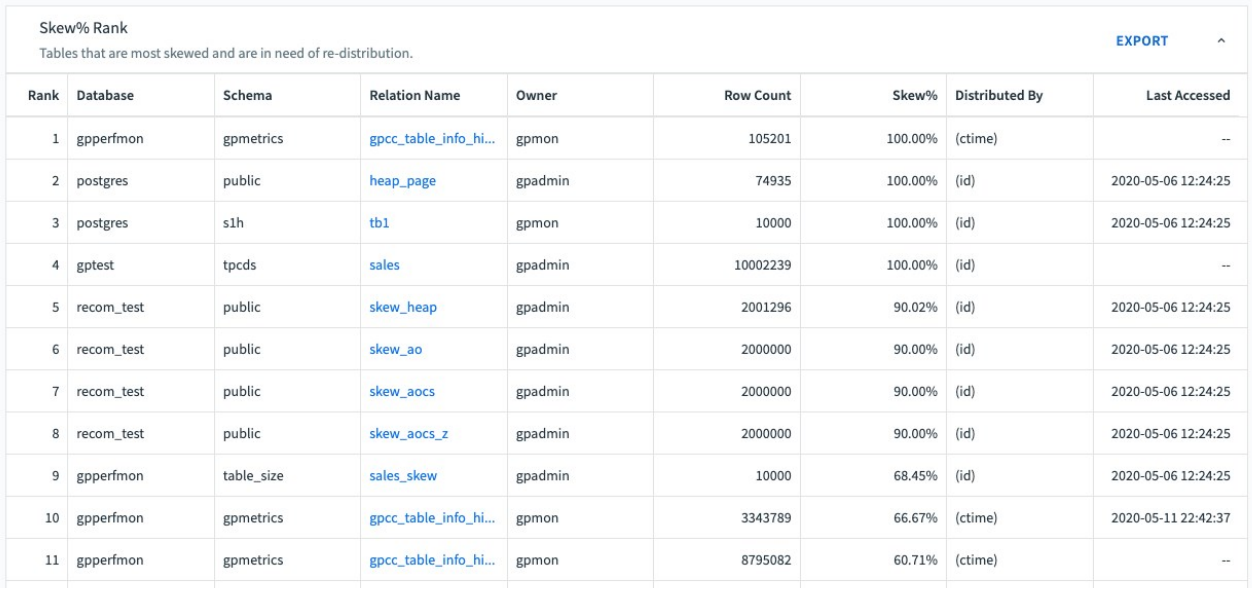
```
ALTER TABLE <table-name> SET DISTRIBUTED RANDOMLY;
ALTER TABLE <table-name> SET WITH (REORGANIZE=TRUE);
```

The second `ALTER TABLE` command is needed to redistribute the table. Without it, only newly added rows will be distributed randomly.

By default, the **Skew%** report includes tables with a calculated skew of at least 10%. You can specify a different skew threshold by setting the `skew_threshold` parameter in the gpmetrics configuration file.

A table with a small number of rows can have a high **Skew** value because there are insufficient rows to achieve an even distribution. Command Center excludes such tables from the **Skew%** report by filtering for tables that have at least `128 * NSeg` rows, where `NSeg` is the number of segments in the Greenplum system. You can change this default by setting the `skew_tuple_per_segment` value in the gpmetrics configuration file.

See the gpmetrics Configuration File Reference for more information about setting the `skew_threshold` and `skew_tuple_per_segment` parameters.



Skew% Rank
Tables that are most skewed and are in need of re-distribution.

Rank	Database	Schema	Relation Name	Owner	Row Count	Skew%	Distributed By	Last Accessed
1	gpperfmon	gpmetrics	gpcc_table_info_hi...	gpmon	105201	100.00%	(ctime)	--
2	postgres	public	heap_page	gpadmin	74935	100.00%	(id)	2020-05-06 12:24:25
3	postgres	s1h	tb1	gpmon	10000	100.00%	(id)	2020-05-06 12:24:25
4	gptest	tpcds	sales	gpadmin	10002239	100.00%	(id)	--
5	recom_test	public	skew_heap	gpadmin	2001296	90.02%	(id)	2020-05-06 12:24:25
6	recom_test	public	skew_ao	gpadmin	2000000	90.00%	(id)	2020-05-06 12:24:25
7	recom_test	public	skew_aocs	gpadmin	2000000	90.00%	(id)	2020-05-06 12:24:25
8	recom_test	public	skew_aocs_z	gpadmin	2000000	90.00%	(id)	2020-05-06 12:24:25
9	gpperfmon	table_size	sales_skew	gpadmin	10000	68.45%	(id)	2020-05-06 12:24:25
10	gpperfmon	gpmetrics	gpcc_table_info_hi...	gpmon	3343789	66.67%	(ctime)	2020-05-11 22:42:37
11	gpperfmon	gpmetrics	gpcc_table_info_hi...	gpmon	8795082	60.71%	(ctime)	--

The **Skew%** table has these columns:

Rank

Tables in the list are ranked by Skew%, from greatest to least

Database

Name of the database containing the table.

Schema

Name of the schema to which the table belongs

Relation Name

Name of the table

Row Count

Number of rows in the table

Skew%

Amount of skew, between 0% and 100%. A lower percentage is best for Greenplum Database query performance.

Distributed By

The current distribution key for the table.

Monitoring and Managing Greenplum Database Queries

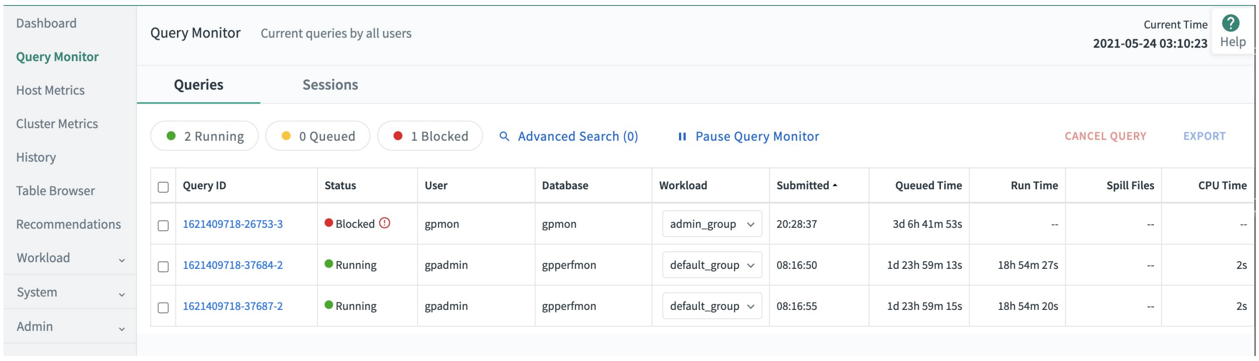
- Query Monitor
- Query Details
- Query History

Query Monitor

The **Query Monitor** view allows you to view information for all Greenplum Database server queries, including details about queries running, queued to run, and blocked by other queries.

In addition, the **Query Monitor** displays a variety of information about sessions, such as session status, associated users and databases, idle time, and associated queries.

Users with Admin or Operator privileges can see and cancel all users' queries, or can move queries. They can also see and cancel all users' sessions, and export session details to a CSV file.



Query ID	Status	User	Database	Workload	Submitted	Queued Time	Run Time	Spill Files	CPU Time
1621409718-26753-3	● Blocked	gpmon	gpmon	admin_group	20:28:37	3d 6h 41m 53s	--	--	--
1621409718-37684-2	● Running	gpadmin	gpperfmon	default_group	08:16:50	1d 23h 59m 13s	18h 54m 27s	--	2s
1621409718-37687-2	● Running	gpadmin	gpperfmon	default_group	08:16:55	1d 23h 59m 15s	18h 54m 20s	--	2s

If a Command Center administrator has enabled Query Monitor Guest Access, anyone able to access the Command Center web server can view the system status and query list on this page without signing in to Command Center. Anonymous users, however, cannot cancel queries or access any other Command Center features.

With the information available in this view, Greenplum Database administrators can easily:

- Understand how the system is being used — both in real-time and trending over time.
- Identify and diagnose problem queries while they are running, detect skew, find runaway queries, and so on.
- Review and balance the query load on the system by better optimizing and scheduling the query load.
- Cancel queries that disrupt system performance.
- Cancel idle sessions.

- View and isolate queries with high CPU consumption.

Note: The Query Monitor does not display queries executed by the `gpmon` user in the gpperfmon database.
Important: Connecting through an HTTP proxy to the Command Center frontend (client browser), or backend (Command Center agent processes) can prevent the display of any query data in the Query Monitor. See Using an HTTP Proxy Server with Command Center for instructions to disable proxied Command Center connections.

Query Metrics

To display query metrics, click the Queries tab. The Query Monitor table displays the following columns for queries.

Query ID

An identification string for the query. If the column is blank, no query ID has been assigned yet. In the Console, this looks like "1295397846-56415-2". Command Center generates this ID by combining the query record's `tmid`, `ssid`, and `ccnt` fields.

- `tmid` is a time identifier for the query.
- `ssid` is the session id.
- `ccnt` is the number of the command within the session.

Status

The status of the query. This can be one of the following:

- Queued: the query has not yet started to execute
- Running: execution has started, but is not yet complete
- Blocked: the query is waiting for one or more other queries to release locks
- Done: completed successfully
- Cancelling: cancel request sent, cancel pending
- Cancelled: terminated, no longer running
- Idle Transaction: the transaction is open, but idle, for example, waiting while a user in an interactive session enters a statement

User

The Greenplum Database role that submitted the query.

Database

The name of the database that was queried.

Workload

The resource group or resource queue that is managing the query.

Submitted

The time the query was submitted to the query planner.

Queued Time

The amount of time the query has been (or was) in queue awaiting execution.

Run Time

The amount of time since query execution began.

Spill Files

The total size of spill files created for the query. Greenplum Database creates spill files when there is insufficient memory to to execute the query in memory. See Managing Spill Files Generated by Queries for information about spill files.

CPU Time

Shows the amount of system CPU consumed by the individual query.

Session Information

To display session information, click the Sessions tab. The Query Monitor table displays the following columns for sessions.

Session ID

An identification string for the session.

User

The Greenplum Database role that established the session.

Database

The name of the database connected to in the session.

Status

The status of the session. This can be one of the following:

- active: the connection is working
- idle: the connection is idle
- idle in transaction: a transaction is running but not active, potentially waiting for user input
- idle in transaction (aborted): similar to idle in transaction except that one of the statements in the transaction caused an error

Start Time

The time the session was established.

Application Name

The name of the application that established the connection, for example, psql.

Idle Time

The amount of time since activity was last detected in the session.

Note: Command Center calculates the idle time from the last query's completion time. However, if this information is not available, idle time is calculated from the time that the idle status is detected. In such situations, the displayed idle time may be shorter than the actual idle time.

Last Query ID

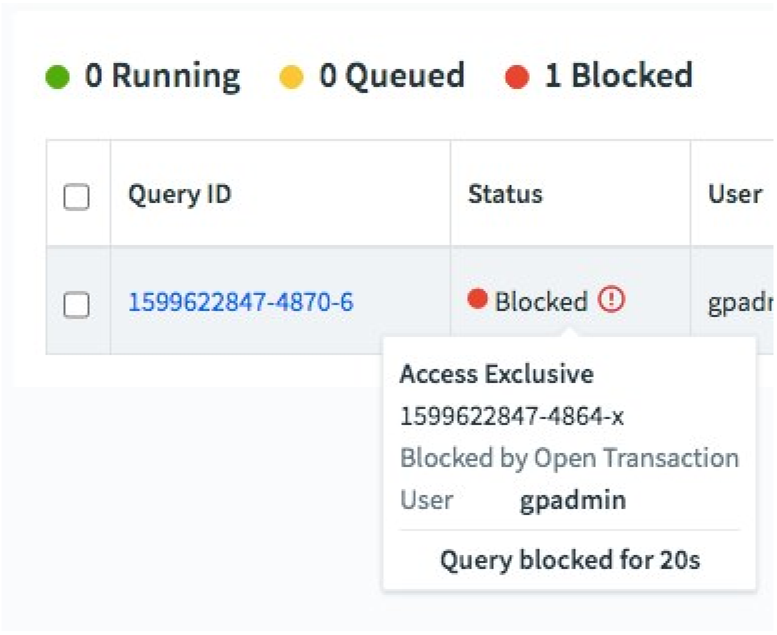
The ID of the last query that was running before the session became idle. If the session is active, this displays the ID of the running query. To view the query text, hover your mouse over the ID.

Using the Query Monitor Controls for Queries

You can use the Query Monitor to display basic information about queries, cancel or export queries, see a query's details, and reassign the query to a different resource group.

In addition, you can pause the query monitor to see a snapshot of all running queries, filter queries by their status, and perform an advanced search of queries.

General Tasks

- If a query shows the **Blocked** status, use the tooltip to display the transaction that is blocking 

	Query ID	Status	User
☐	1599622847-4870-6	● Blocked	gpadr

Access Exclusive
1599622847-4864-x
Blocked by Open Transaction
User gpadmin
Query blocked for 20s

  the query.
- Click a column heading to sort the rows on that column in ascending or descending order.
- Click the checkbox at the left of a row to choose a query to cancel or export. Click the checkbox in the heading row to choose all queries.
- Click **CANCEL QUERY** to cancel selected queries. A pop-up box prompts you to enter a reason. Enter the error message (max 128 characters) displayed to users whose sessions are cancelled.
- Click **EXPORT** to download a comma-separated values (CSV) text file containing rows for the selected queries. When no queries are selected, all rows are exported. The default file name is `spreadsheet.csv`.
- Click any query ID to see the Query Details, including metrics, the text of the query, and the query plan.

Pausing and Resuming the Query Monitor

You can pause the Query Monitor to view a snapshot of running queries, and then resume the Query Monitor. This allows you to see the status of a query before it disappears from the query list.

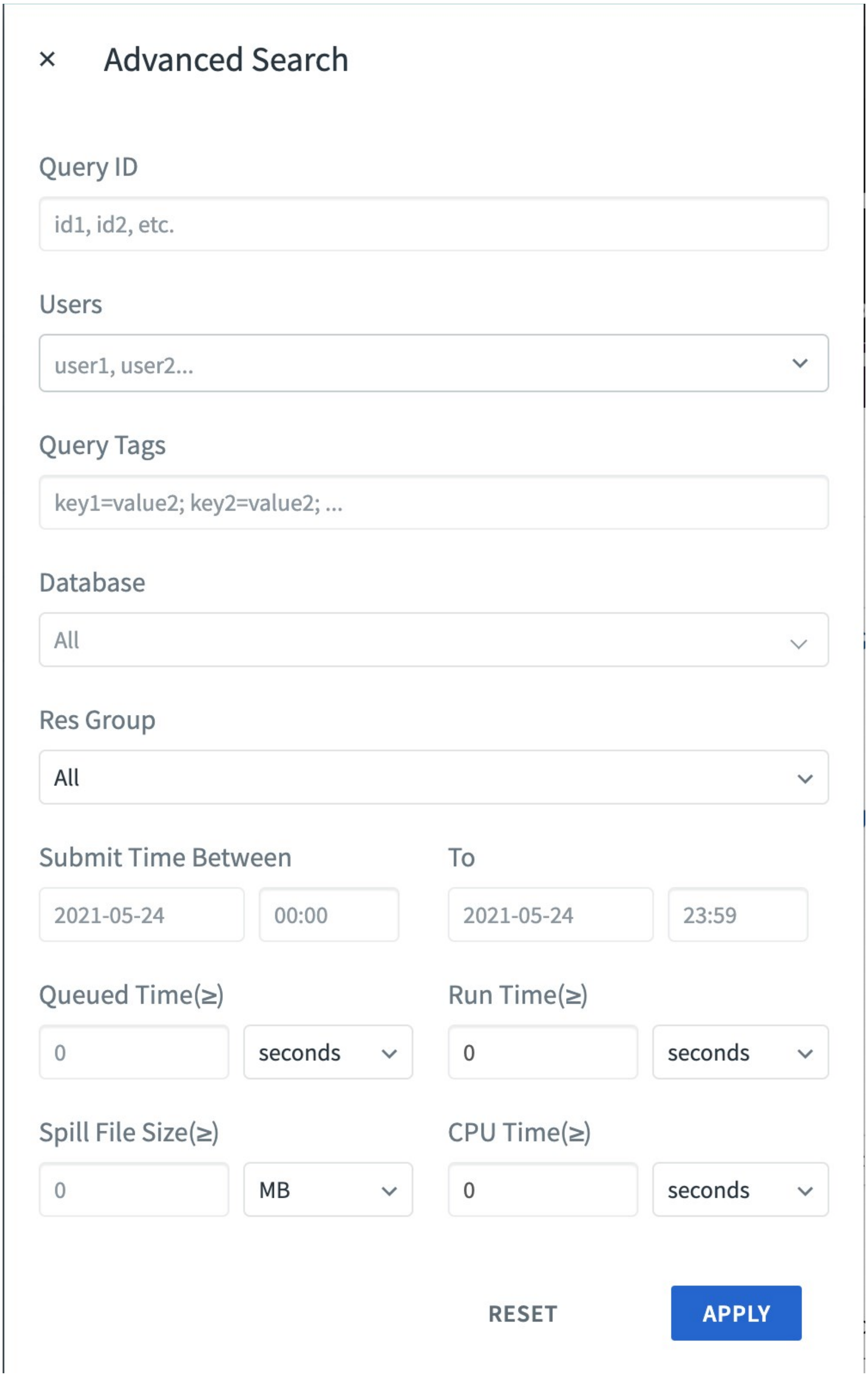
- To pause the Query Monitor, click the Pause Query Monitor button. When you pause the Query Monitor, Command Center displays a warning message that the queries you are viewing are not up to date.
- When you pause the Query Monitor, the query list stops refreshing. Pausing the Query Monitor merely suspends the UI data stream, it does not pause query execution. When you resume monitoring, you will see the updated state, which means some queries may disappear immediately if they are already done.
- You can click a Query ID to display the Query details page. The information on this page continues to be updated in real time.
- When you are viewing the details of a query that completed after pausing the Query Monitor, you will see two possible pages:
    - For queries already saved to Query History, the displayed information matches that of the Query History page.
    - If the query is not already saved to query history, you will see a temporary page containing some outdated information, as well as a warning message requesting that you reload the page.
- While the Query Monitor is paused, if you click on a Query ID to view that query's details, when you return to the Query Monitor some data will have been updated, including Status, Spill File Size, and CPU Time. If a query has completed, it will no longer appear on the list.
- To resume the Query Monitor, click the Resume Query Monitor button.

Filtering Queries

You can filter queries to show any combination of running, queued, and blocked queries. In addition, using the Advanced Search button, you can filter queries by a variety of other attributes.

- To filter by Status, select or deselect the **Running** (green), **Queued** (yellow) or **Blocked** (red) buttons. When the colored dot is solid, the Query Monitor displays queries with that status. When empty, the Query Monitor hides queries with that status.

  For example, if the **Running** and **Queued** dots are solid but the **Blocked** dot is empty, the Query Monitor displays running and queued queries but not blocked queries.
- To filter query results using Advanced Search, click the Advanced Search button and then select the criteria for the search.



Advanced Search

Query ID

id1, id2, etc.

Users

user1, user2...

Query Tags

key1=value2; key2=value2; ...

Database

All

Res Group

All

Submit Time Between		To	
2021-05-24	00:00	2021-05-24	23:59
Queued Time(≥)		Run Time(≥)	
0	seconds	0	seconds
Spill File Size(≥)		CPU Time(≥)	
0	MB	0	seconds

RESET APPLY

Query ID

Returns the query or queries identified by the string or strings entered. Separate multiple query ids with commas.

Users

Returns queries owned by the specified users.

Query Tags

Returns queries whose query tags match the values entered. Enter the tags as key-value pairs. Separate multiple query tags with semicolons. Not available in Command Center 4.x.

Database

Returns queries that are running against the specified database.

Res Group

Returns queries executed by the specified resource queue or resource group.

Submit Time (>=)

Returns queries that were submitted in the window of time specified.

Queued Time (>=)

Returns queries that have been in queue awaiting execution for at least the specified amount of time. You may specify in seconds, minutes, or hours.

Run Time (>=)

Returns queries that have been running for at least the specified amount of time. You may specify in seconds, minutes, or hours.

Spill File Size (>=)

Returns queries that generated spill files of the specified size. You may specify in MB, GB, or TB.

CPU Time (>=)

Returns queries that have consumed the specified amount of CPU time. You may specify in seconds, minutes, or hours.

Click **APPLY** to display results that match your criteria. The number of conditions will be retained until you click **RESET**.

Using the Query Monitor Controls for Sessions

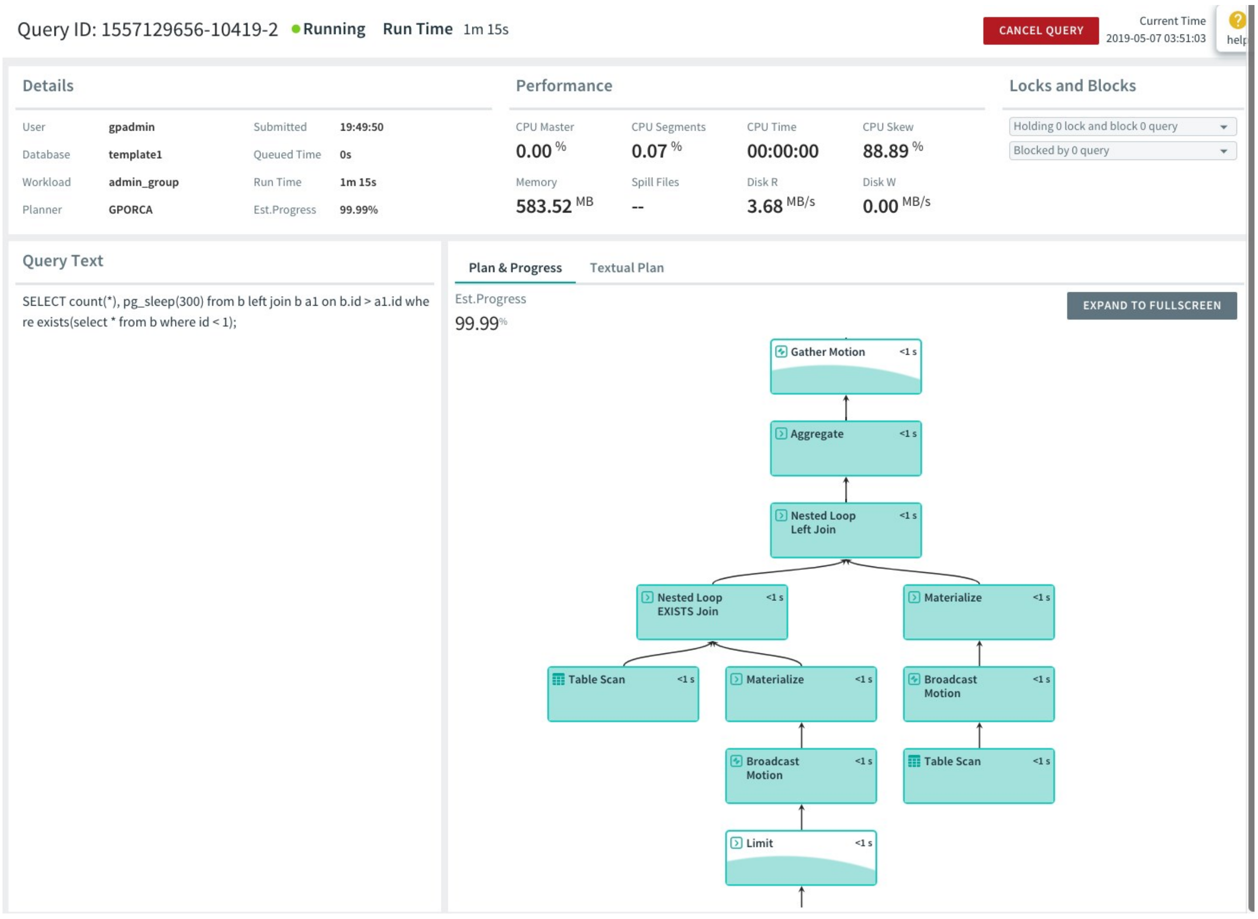
- Click a column heading to sort the rows on that column in ascending or descending order.
- Click the checkbox to the left of a row to choose a session to cancel or export. Click the checkbox in the heading row to choose all sessions.
- Click the checkbox just above the Session ID column labeled "Show idle in transaction Only" to display only the sessions whose status is idle in transaction.
- Click **CANCEL SESSION** to cancel selected sessions. A pop-up box prompts you to enter a reason. You can enter a message of up to 128 characters to display with the error message that is received by users whose sessions are cancelled.

  Note: You cannot cancel sessions whose Status is active.

- Click **EXPORT** to download a comma-separated values (CSV) text file containing rows for the selected sessions. When no sessions are selected, all rows are exported. The default file name is `spreadsheet.csv`.

Query Details

The **Query Details** view displays query metrics, the text of the query, and the query plan and progress for a single query selected from the Query Monitor view.



Query ID: 1557129656-10419-2 ●Running Run Time 1m 15s

CANCEL QUERY Current Time 2019-05-07 03:51:03 help

Details

User	gpadmin	Submitted	19:49:50
Database	template1	Queued Time	0s
Workload	admin_group	Run Time	1m 15s
Planner	GPORCA	Est.Progress	99.99%

Performance

CPU Master	CPU Segments	CPU Time	CPU Skew
0.00 %	0.07 %	00:00:00	88.89 %
Memory	Spill Files	Disk R	Disk W
583.52 MB	--	3.68 MB/s	0.00 MB/s

Locks and Blocks

Holding 0 lock and block 0 query

Blocked by 0 query

Query Text

```
SELECT count(*), pg_sleep(300) from b left join b a1 on b.id > a1.id whe
re exists(select * from b where id < 1);
```

Plan & Progress | Textual Plan

Est.Progress 99.99%

Query Execution Metrics

The Query ID, execution status, and run time of the query are displayed at the top.

The following metrics are displayed for the query.

User

The Greenplum Database role that submitted the query.

Database

The name of the database that was queried.

Workload

The name of the resource group or resource queue that is managing the query.

Planner

The name of the query planner used for this query, GPORCA or Legacy.

Submitted

The time the query was submitted to the query planner.

Queued Time

The amount of time the query has been (or was) in queue awaiting execution.

Run Time

The amount of time since query execution began.

Est. Progress

An estimate of the percentage of the query execution completed. The estimate is calculated from row count and cost estimates generated by either the GPORCA or legacy planner for the particular query and the available statistics. The estimate does not account for the many other factors that can affect query execution so it should not be seen as a reliable predictor of query completion time.

The progress for each plan node is calculated as the ratio of actual rows produced to the planner's estimate of the total number of rows the node will produce:

$$NodeProgress = \frac{ActualRows}{EstimatedRows}$$

The overall progress for the query is estimated using the calculated node progress and the planner's cost estimates:

$$OverallProgress = \frac{\sum NodeProgress(x) * Est.Cost(x)}{\sum Est.Cost(x)}$$

If the estimate is greater than 100% and the query has not yet completed, 99.9% completion is reported. 100% is reported if the formula produces an estimated percentage greater than 100%.

CPU Master

Current CPU percent on the Greenplum Database master host for this query.

CPU Segments

(Active queries only.) Current CPU percent average for all segment processes executing this query. The percentages for all processes running on each segment are averaged, and then the average of all those values is calculated to render this metric. Current CPU percent average is always zero in historical and tail data. The master and standby master are excluded from the calculation.

CPU Time

Total CPU time consumed by all processes on all segments executing this query.

CPU Skew

The amount of CPU skew. CPU skew occurs when query executor processes for one segment use a disproportionate amount of CPU compared to processes for other segments executing the query. This value is calculated as

```
1 - (average_segment_CPU / maximum_segment_CPU)
```

Memory

Memory consumed by all segment processes executing the query.

Spill Files</dt`> The total size of spill files created for the query. Greenplum Database creates spill files when there is insufficient memory to execute the query in memory. See Managing Spill Files Generated by Queries for information about spill files.

Disk R

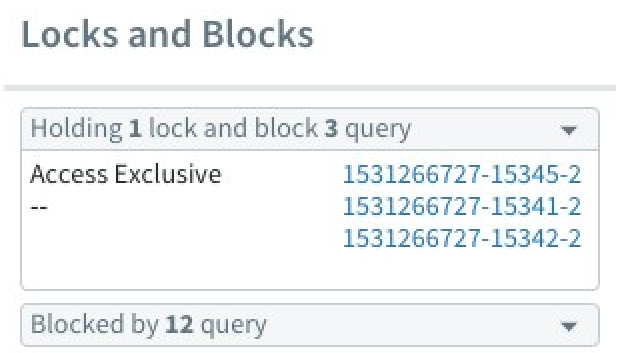
The current average disk read rate for all segment hosts.

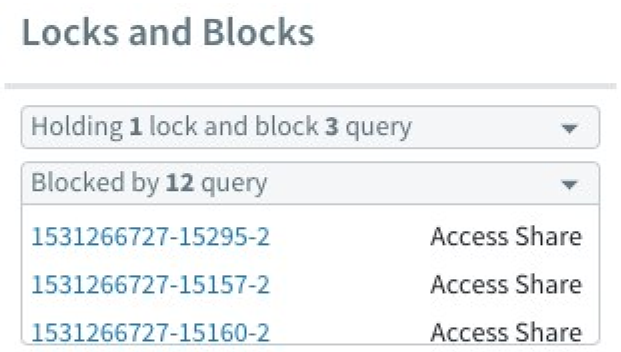
Disk W

The current average disk write rate for all segment hosts.

Locks and Blocks

Contains two lists of locks currently blocking transactions. Click a list to expand and view the contents.

- A list of locks held by this query, including the type of each lock and the queries blocked by 
Locks and Blocks
Holding **1** lock and block **3** query
Access Exclusive 1531266727-15345-2
-- 1531266727-15341-2
1531266727-15342-2
Blocked by **12** query

  that lock.
- A list of queries that hold locks that block this query and the lock type.

Locks and Blocks
Holding **1** lock and block **3** query
Blocked by **12** query
1531266727-15295-2 Access Share
1531266727-15157-2 Access Share
1531266727-15160-2 Access Share

Query Text and Execution Plan

The query text and the query's plan and execution progress are shown in the lower panels of the Query Details view. The text of the query is displayed in the left panel, and the plan and progress is displayed in the right panel.

Query Text

The **Query Text** panel displays the text of the query as it was submitted to Greenplum Database.

Command Center can display up to 100K characters. If you click **COPY**, up to 100K characters of the query text are copied to the clipboard.

If the query text is longer than 100K characters, a message is displayed with a link you can use to download the full text of the query. The name of the text file is the ID of the query with a `.txt` extension. The file is available to download for 24 hours after the query completes, or until the query has been saved to history, once history collection is enabled.

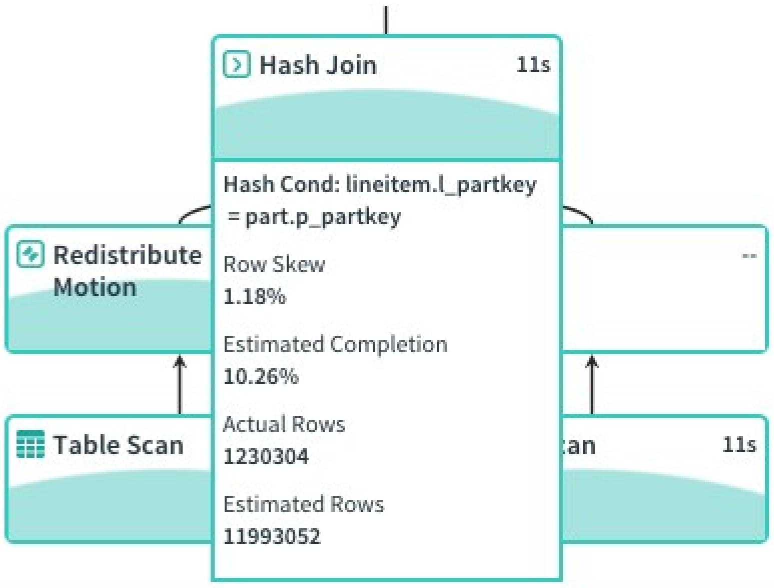
Query Plan and Progress

The **Plan & Progress** tab in the lower right panel is a graphical representation of the query plan with animation and real-time metrics to show execution progress. Each box in the tree represents a step in the query execution plan. The boxes are labeled with the operation they represent and have a CPU usage metric. Query execution begins at the bottom of the tree and ends at the top.

Before a step begins to execute, the box has a solid white fill. When the step becomes active, the box is animated with a green and white fill to indicate that the operator is working. When the step has completed, the box has a solid green fill.

Query execution plans are executed in "slices," portions of the query plan that segments can work on independently in parallel. The plan is sliced wherever a data motion operator occurs. The time displayed in the upper right corner of each box is the amount of CPU time used for all steps in the slice to which the step belongs. Each slice is displayed in its own color, making it easy to distinguish the slices from each other. The visual query plan does not illustrate slices, but you can find information about slices in the textual plan.

If you click a step, the box expands to show additional details.



Hash Cond: lineitem.l_partkey = part.p_partkey

Row Skew
1.18%

Estimated Completion
10.26%

Actual Rows
1230304

Estimated Rows
11993052

The expanded plan box contains the following metrics.

- The type of operator. When the operator is a table scan, the name of the scanned table is included. See Query Plan Execution for descriptions of the operators.
- Information related to the current operation, such as the hash key, merge key, join condition, or filter condition.
- Row Skew - the amount of row skew for the current operator, a value from 0% to 100%. Row skew occurs when some segments process more rows than other segments. The percentage is calculated as `(1 - (average_segment_rows / maximum_segment_rows)) * 100`. Nodes with a row skew of greater than 30% are marked with an exclamation point.

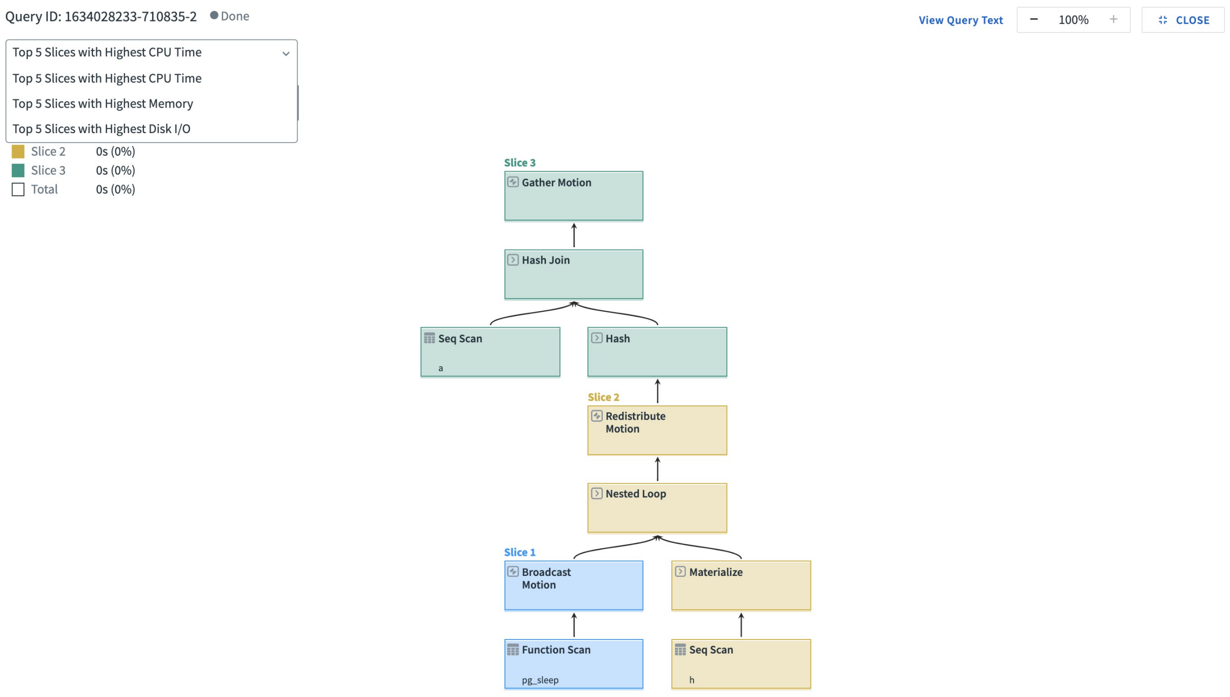
- Estimated Completion - the current percentage of actual rows to estimated rows for this plan step. The percentage can exceed 100% if the operation produces more rows than the optimizer's estimate. The percentage changes to "Completed" when the operation completes.
- Actual Rows - The current number of rows produced by this step. Note that for nested join operators, the Actual Rows is estimated since the actual row counts are not available while the join is executing.
- Estimated Rows - The estimated number of rows the operator will produce.

Display Slice Metrics

From within the **Plan and Progress** tab you may display key metrics about slices, including:

- the top 5 slices that consume the most CPU time
- the top 5 slices that consume the most memory
- the top 5 slices that use the most disk I/O

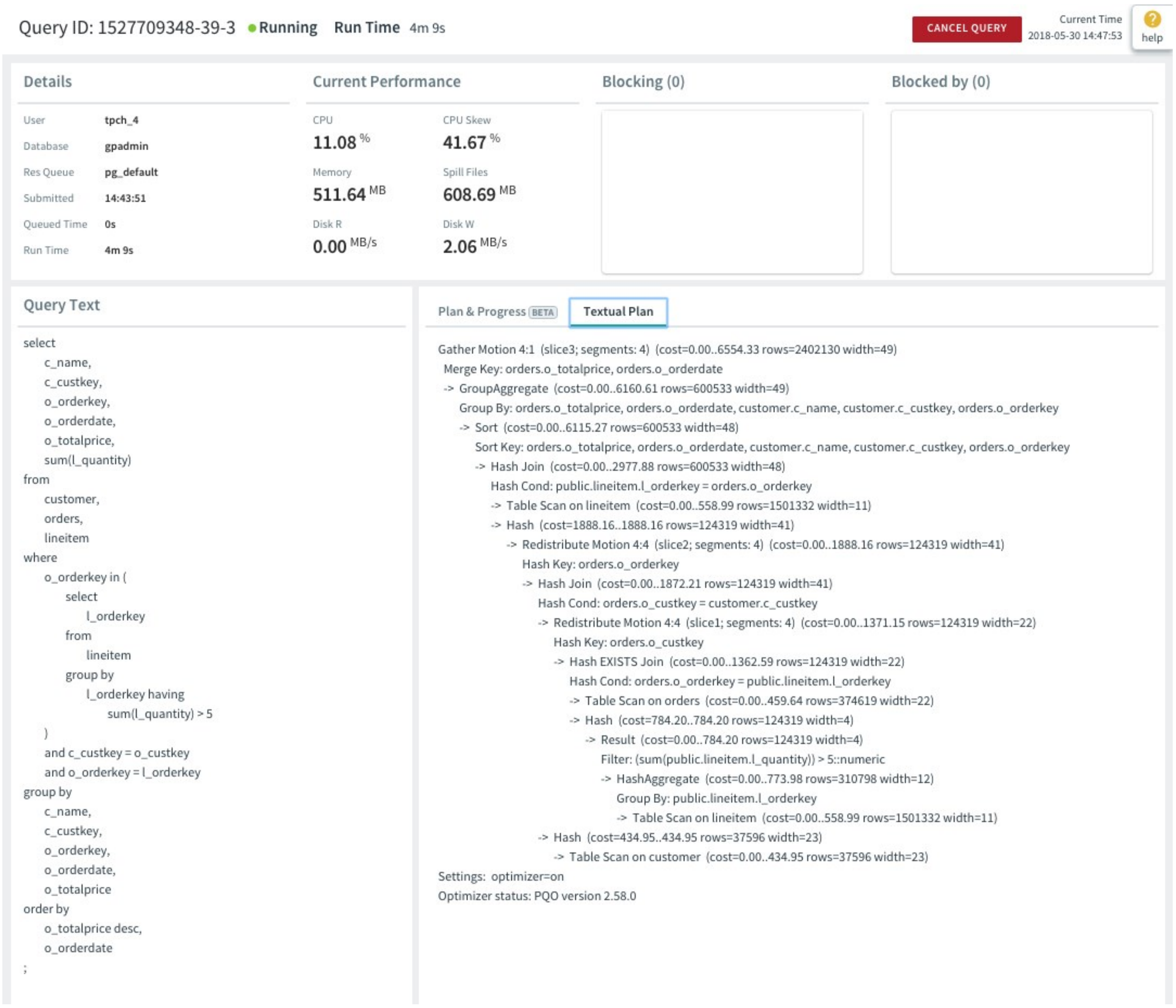
To display this information, click the **Expand** button on the far right. This displays a screen with a dropdown menu from which you can choose which slice metrics you want to display:



Textual Plan

Select the **Textual Plan** tab and click **RUN EXPLAIN** to generate the text representation of the explain plan.

Note: The **RUN EXPLAIN** button is dimmed if Command Center is unable to generate the explain plan. Command Center is unable to generate the explain plan if the size of the query text is greater than 100K characters or if the query text contains multiple statements.



Details

User	tpch_4
Database	gpadmin
Res Queue	pg_default
Submitted	14:43:51
Queued Time	0s
Run Time	4m 9s

Current Performance

CPU 11.08 %	CPU Skew 41.67 %
Memory 511.64 MB	Spill Files 608.69 MB
Disk R 0.00 MB/s	Disk W 2.06 MB/s

Blocking (0)

Blocked by (0)

Query Text

```
select
    c_name,
    c_custkey,
    o_orderkey,
    o_orderdate,
    o_totalprice,
    sum(l_quantity)
from
    customer,
    orders,
    lineitem
where
    o_orderkey in (
        select
            l_orderkey
        from
            lineitem
        group by
            l_orderkey having
                sum(l_quantity) > 5
    )
    and c_custkey = o_custkey
    and o_orderkey = l_orderkey
group by
    c_name,
    c_custkey,
    o_orderkey,
    o_orderdate,
    o_totalprice
order by
    o_totalprice desc,
    o_orderdate
;
```

```
Gather Motion 4:1  (slice3; segments: 4)  (cost=0.00..6554.33 rows=2402130 width=49)
 Merge Key: orders.o_totalprice, orders.o_orderdate
 -> GroupAggregate  (cost=0.00..6160.61 rows=600533 width=49)
     Group By: orders.o_totalprice, orders.o_orderdate, customer.c_name, customer.c_custkey, orders.o_orderkey
     -> Sort  (cost=0.00..6115.27 rows=600533 width=48)
         Sort Key: orders.o_totalprice, orders.o_orderdate, customer.c_name, customer.c_custkey, orders.o_orderkey
         -> Hash Join  (cost=0.00..2977.88 rows=600533 width=48)
             Hash Cond: public.lineitem.l_orderkey = orders.o_orderkey
             -> Table Scan on lineitem  (cost=0.00..558.99 rows=1501332 width=11)
             -> Hash  (cost=1888.16..1888.16 rows=124319 width=41)
                 -> Redistribute Motion 4:4  (slice2; segments: 4)  (cost=0.00..1888.16 rows=124319 width=41)
                     Hash Key: orders.o_orderkey
                     -> Hash Join  (cost=0.00..1872.21 rows=124319 width=41)
                         Hash Cond: orders.o_custkey = customer.c_custkey
                         -> Redistribute Motion 4:4  (slice1; segments: 4)  (cost=0.00..1371.15 rows=124319 width=22)
                             Hash Key: orders.o_custkey
                             -> Hash EXISTS Join  (cost=0.00..1362.59 rows=124319 width=22)
                                 Hash Cond: orders.o_orderkey = public.lineitem.l_orderkey
                                 -> Table Scan on orders  (cost=0.00..459.64 rows=374619 width=22)
                                 -> Hash  (cost=784.20..784.20 rows=124319 width=4)
                                     -> Result  (cost=0.00..784.20 rows=124319 width=4)
                                         Filter: (sum(public.lineitem.l_quantity)) > 5::numeric
                                         -> HashAggregate  (cost=0.00..773.98 rows=310798 width=12)
                                             Group By: public.lineitem.l_orderkey
                                             -> Table Scan on lineitem  (cost=0.00..558.99 rows=1501332 width=11)
                         -> Hash  (cost=434.95..434.95 rows=37596 width=23)
                             -> Table Scan on customer  (cost=0.00..434.95 rows=37596 width=23)
Settings:  optimizer=on
Optimizer status: PQO version 2.58.0
```

The textual plan is the output of the Greenplum Database `EXPLAIN` command for the query. The query plan steps are labeled with arrows (`->`) and the structure of the query plan tree is indicated with indentation.

The `Optimizer status:` line at the bottom of the textual plan reports whether the explain plan was generated using the GPORCA optimizer (PQO) or the legacy query optimizer.

For help reading the textual explain plan see the `EXPLAIN` command in the *Greenplum Database Reference Guide* and Query Profiling in the *Greenplum Database Administrator Guide*. See Query Execution for descriptions of the query operators.

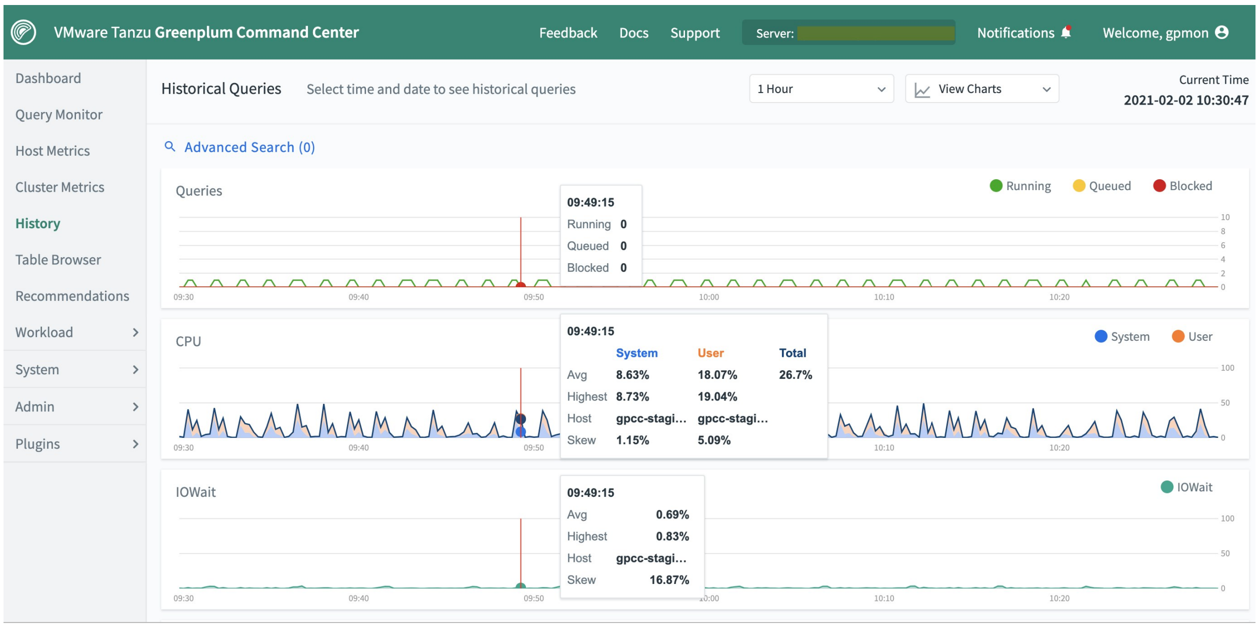
History

Note: On the Admin> Settings page you can enable saving the real-time metrics collected by the Greenplum Database metrics collector extension to history in the gpmetrics schema of the gpperfmon database. When you enable collecting this history, the Command Center History, Query Monitor, and Query Detail views all use data derived from the same data collection method.

The **History** page allows you to display system metrics charts and and details about queries executed during a specified time period.

Use the Advanced Search tool to refine the query search results. The Queries Table shows the results and details of the search.

By default, the dashboard shows the last 1 hour of query history.



09:49:15
Running 0
Queued 0
Blocked 0

09:49:15

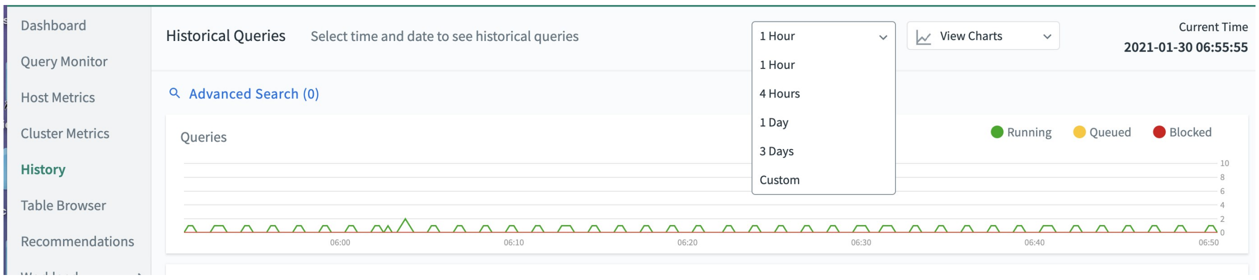
	System	User	Total
Avg	8.63%	18.07%	26.7%
Highest	8.73%	19.04%	
Host	gpcc-stagi...	gpcc-stagi...	
Skew	1.15%	5.09%	

09:49:15
Avg 0.69%
Highest 0.83%
Host gpcc-stagi...
Skew 16.87%

Note: On the Admin> Settings page, you can enable saving the real-time metrics collected by the Greenplum Database metrics collector extension to history in the gpmetrics schema of the gpperfmon database. When you enable collecting this history, the Command Center History, Query Monitor, and Query Detail views all use data derived from the same data collection method.

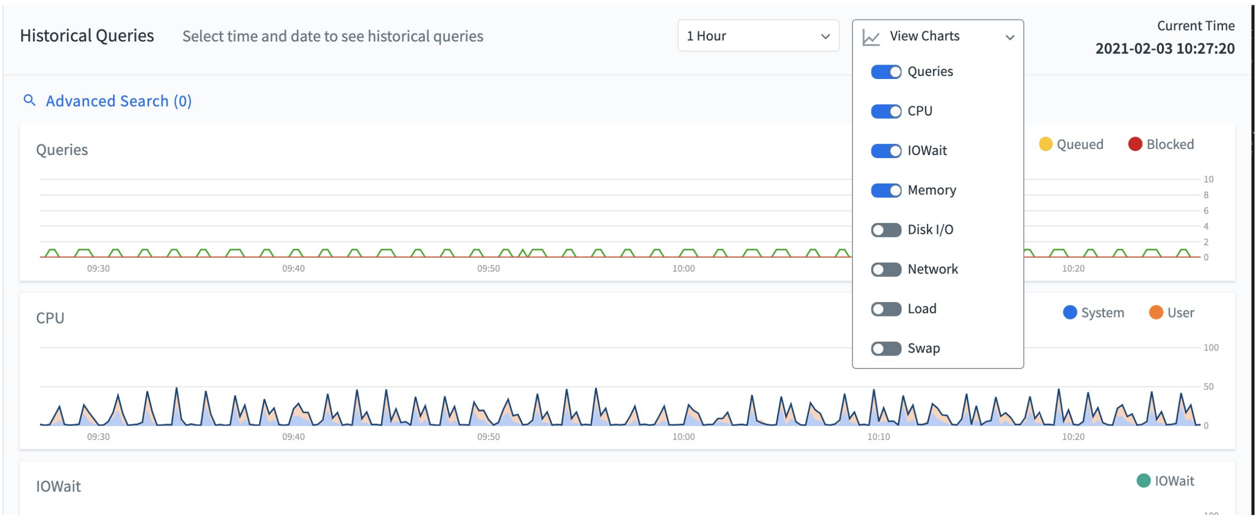
If you disable GPCC query history collection on the **Admin> Settings** page, the history view displays data collected and saved in the gpperfmon database by the `gpmmon` and `gpsmon` agents. That data is not derived from the real-time metrics displayed in the Query Monitor and Query Detail view. Variations should be expected due to the different data collection methods.

To select a different time period (1 hour, 4 hours, 1 day, 3 days, and custom), use the time range dropdown:



Selecting "custom" from the time period dropdown leads you to the **Advanced Search** popup window. See Advanced Search for more details.

To select which charts to view for the latest queries, use the **View Charts** dropdown. By default, the following charts are selected:



The following metrics charts are available:

Queries

The number of queries running and the number of queries queued to run.

CPU

The percentage of CPU used by system processes and user processes.

IOWait

The time waiting for disk I/O.

Memory

The percentage of memory in use.

Disk I/O

Disk read and write rates in megabytes per second.

Network

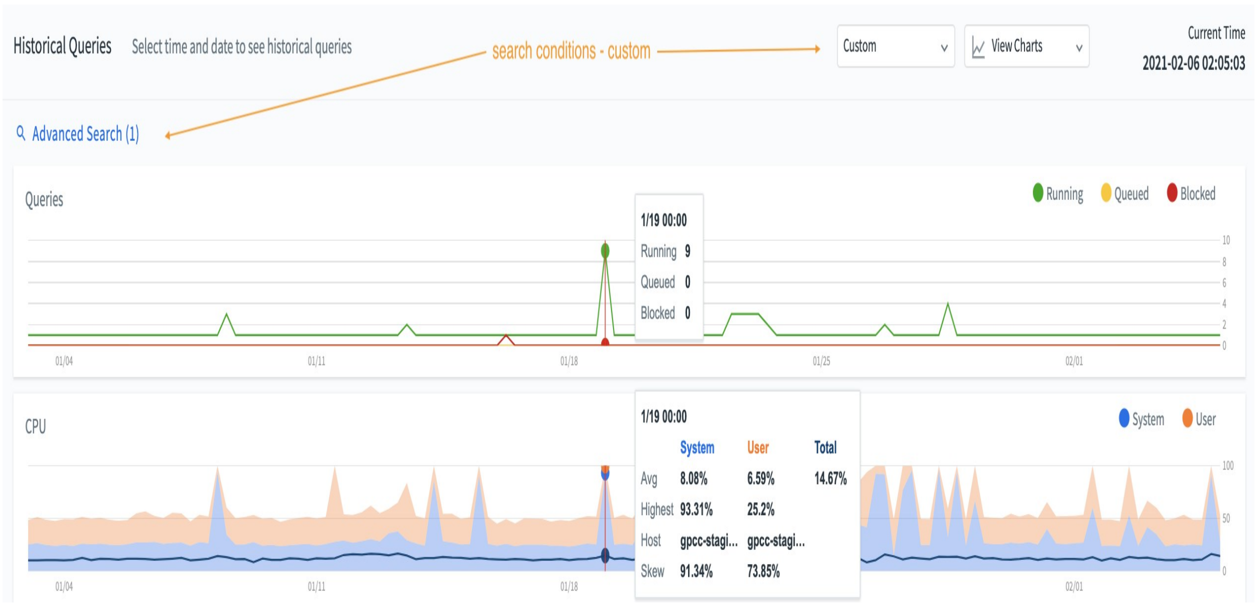
Network I/O read and write rates in megabytes per second. Network metrics include traffic over all NICs (network interface cards), including internal interconnect and administrative traffic. Load

System load average for 1-minute, 5-minute, and 15-minute periods.
Swap

The percentage of swap space used.

Advanced Search

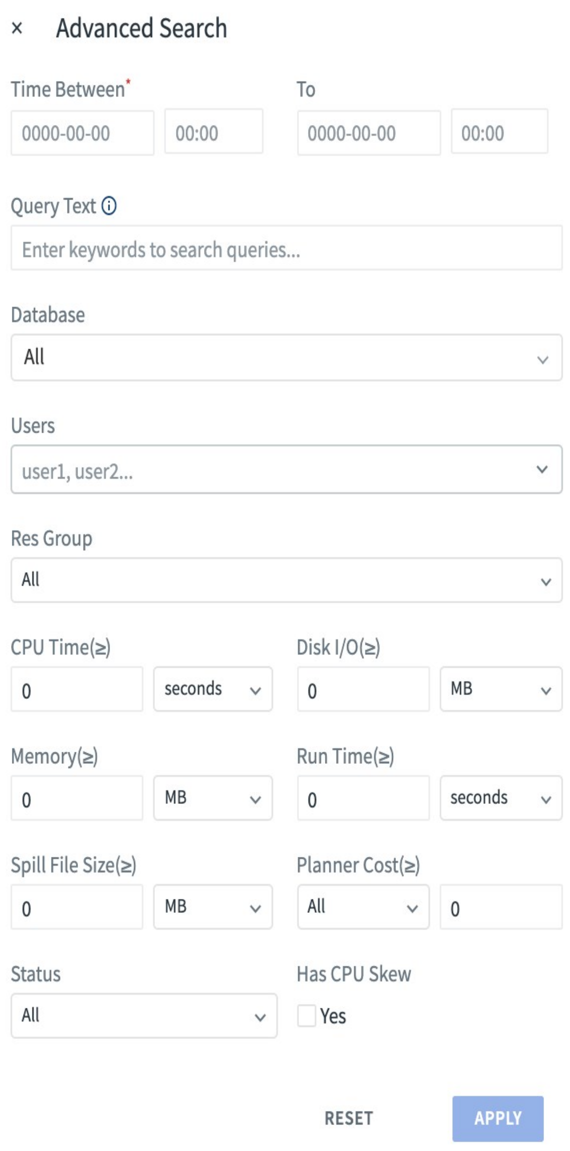
Use the **Advanced Search** tool to restrict the query results displayed at the bottom of the history dashboard.



1/19 00:00

	System	User	Total
Avg	8.08%	6.59%	14.67%
Highest	93.31%	25.2%	
Host	gpcc-stagi...	gpcc-stagi...	
Skew	91.34%	73.85%	

The number of search conditions is displayed beside the **Advanced Search** link. If the search conditions are greater than (0), the Time Range dropdown defaults to **Custom**. The search conditions number is retained between different Command Center dashboards. To reset it, select an option from the Time Range dropdown, or click the **RESET** button in the **Advanced Search** popup window. The search conditions go back to (0) if the user logs out.



Note: In the **Advanced Search** popup window, you must set the **Time Between** start field in order for the **APPLY** button to be highlighted.

Time Between

(Required) Select a valid time range in the HH:MM, HHMM, H:MM, HMM format. If you enter start time but no end time, the results will be all the queries that are running at that specific time. If you select a date but do not specify a time, the default is 00:00.

Query Text

Enter the keywords for your query search, separated by spaces, and within a limit of 256 characters. You may enter a `%` wildcard to match any characters, for example `SELECT%ABC%`.

Database

Select a database from the dropdown. Filtering by database affects only the queries displayed in query results. The metrics displayed in charts include all activity during the selected time period.

Users

Enter a user, or a list of users, from the dropdown. Filtering by users shows the queries owned by those users. The metrics displayed in charts include all activity during the selected time period.

Res Group

Display queries executed by a specified resource queue or resource group.

CPU Time (>=)

Select an integer for the seconds, minutes, or hours of CPU time the queries consume.

Disk I/O (>=)

Select queries with disk I/O above the specified Megabytes (MB), Gigabytes (GB), or Terabytes (TB).

Memory (>=)

Select queries that consume the specified memory number, in MB, GB, or TB.

Run Time (>=)

Select queries that consumed a specific CPU time in seconds, minutes, or hours.

Spill File Size (>=)

Display queries that generated spill files of a certain size.

Planner Cost (>=)

Select a planner, and a cost.

Status

Display queries that completed with the specified status: `Done`, `Cancelled`, or `Error`.

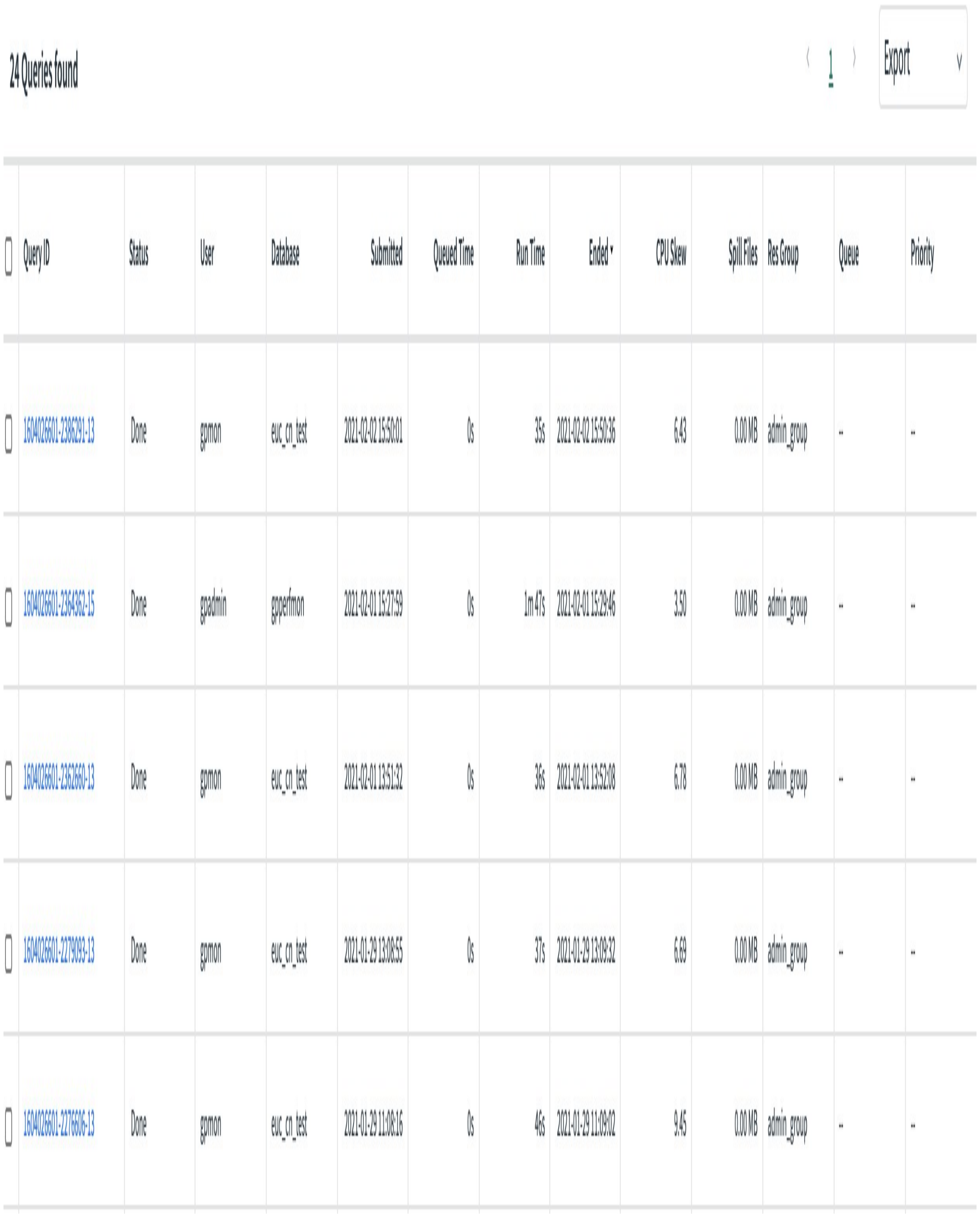
Has CPU Skew

Display queries that had CPU skew greater than zero.

Click **APPLY** to display results that match your criteria. The number of conditions will be retained until your click **RESET** or select a time period in the History dashboard.

Queries Table

The Queries results table displays queries that were active during the specified Time Range or the **Advanced Search** conditions. The results can include queries that started before or finished after the specified time. However, queries that are still active are not included in the table; these queries can be viewed on the Query Monitor page.

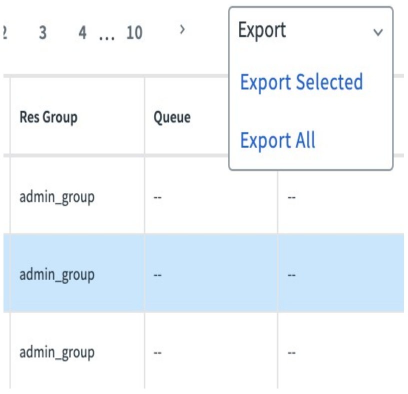


24 Queries found

1 | Export

	Query ID	Status	User	Database	Submitted	Queued Time	Run Time	Ended	CPU Skew	Spill Files	Res Group	Queue	Priority
☐	1604026601-2386291-13	Done	gpmon	euc_cn_test	2021-02-02 15:50:01	0s	35s	2021-02-02 15:50:36	6.43	0.00 MB	admin_group	--	--
☐	1604026601-2364362-15	Done	gpadmin	gpperfmon	2021-02-01 15:27:59	0s	1m 47s	2021-02-01 15:29:46	3.50	0.00 MB	admin_group	--	--
☐	1604026601-2362660-13	Done	gpmon	euc_cn_test	2021-02-01 13:51:32	0s	36s	2021-02-01 13:52:08	6.78	0.00 MB	admin_group	--	--
☐	1604026601-2279093-13	Done	gpmon	euc_cn_test	2021-01-29 13:08:55	0s	37s	2021-01-29 13:09:32	6.69	0.00 MB	admin_group	--	--
☐	1604026601-2276606-13	Done	gpmon	euc_cn_test	2021-01-29 11:08:16	0s	46s	2021-01-29 11:09:02	9.45	0.00 MB	admin_group	--	--

To export any selected query information, or the whole table, use the Export dropdown at the top right of the table:



The Queries results table has the following columns:

Query ID

An identification string for the query. In the Console, this looks like "1295397846-56415-2".

Status

The final status of the query: Done, Cancelled, or Error.

User

The Greenplum Database user who submitted the query.

Database

The name of the database that was queried.

Submitted

The time the query was submitted to the query planner.

Queued Time

The time the query waited before it was executed. In addition to time in the queue, this includes other time such as time in the optimizer.

Run Time

The amount of time the query required to produce a result.

Ended

The time the query completed or was cancelled.

CPU Skew

The amount of CPU skew. CPU skew occurs when query executor processes for one segment use a disproportionate amount of CPU compared to processes for other segments executing the query. This value is the coefficient of variation for the CPU used by processes running this query on each segment, multiplied by 100. For example, a value of .95 is shown as 95. **Spill Files**

The total size of spill files created for the query. Greenplum Database creates spill files when there is insufficient memory to execute the query in memory. See Managing Spill Files Generated by Queries for information about spill files.

Res Group or Res Queue

The name of the resource group or resource queue for the query. For more information about Resource Queues, Resource Groups, and Query Plans, refer to the Greenplum Database

Administrator Guide.

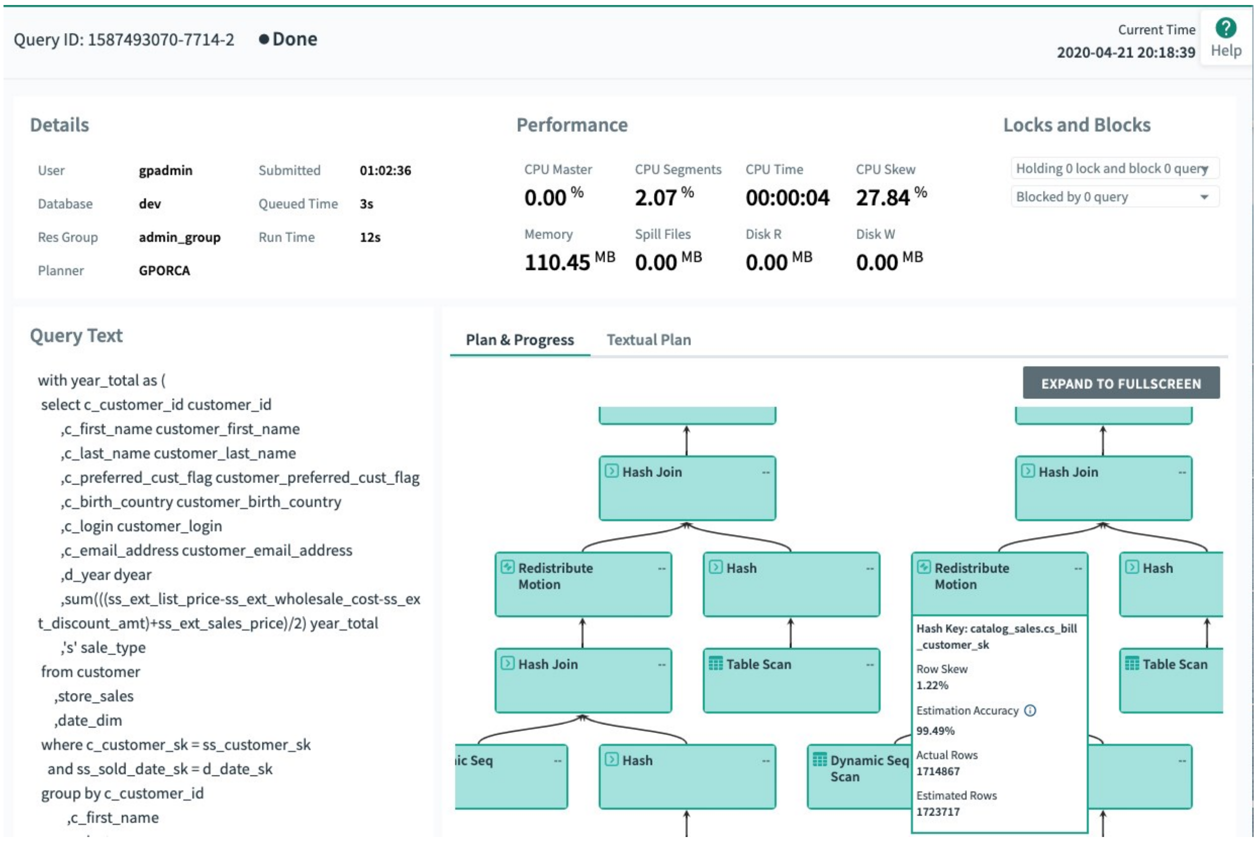
Priority

(Res Queue only) A query executes with the CPU priority assigned to its resource queue.

Select one of the query links to move to the Query History Details dashboard, which includes details about the query metrics, the query text, and the query plan and execution statistics for that query.

Query History Details

The **Query History Details** view displays query metrics, the query text, and the query plan and execution statistics for a single query selected from the Query History view.



Query ID: 1587493070-7714-2 ● Done

Current Time 2020-04-21 20:18:39 Help

Details

User	gpadmin	Submitted	01:02:36
Database	dev	Queued Time	3s
Res Group	admin_group	Run Time	12s
Planner	GPORCA		

Performance

CPU Master	CPU Segments	CPU Time	CPU Skew
0.00 %	2.07 %	00:00:04	27.84 %
Memory	**Spill Files**	**Disk R**	**Disk W**
110.45 MB	0.00 MB	0.00 MB	0.00 MB

Locks and Blocks

Holding 0 lock and block 0 query

Blocked by 0 query

Query Text

```
with year_total as (
 select c_customer_id customer_id
       ,c_first_name customer_first_name
       ,c_last_name customer_last_name
       ,c_preferred_cust_flag customer_preferred_cust_flag
       ,c_birth_country customer_birth_country
       ,c_login customer_login
       ,c_email_address customer_email_address
       ,d_year dyear
       ,sum(((ss_ext_list_price-ss_ext_wholesale_cost-ss_ex
t_discount_amt)+ss_ext_sales_price)/2) year_total
       ,'s' sale_type
 from customer
     ,store_sales
     ,date_dim
 where c_customer_sk = ss_customer_sk
   and ss_sold_date_sk = d_date_sk
 group by c_customer_id
         ,c_first_name
```

Hash Key: catalog_sales.cs_bill_customer_sk

Row Skew
1.22%

Estimation Accuracy
99.49%

Actual Rows
1714867

Estimated Rows
1723717

Query History Metrics

The Query ID, execution status, and run time of the query are displayed at the top.

The following metrics are displayed for the query. Under the **Details** heading:

User

The Greenplum Database role that submitted the query.

Database

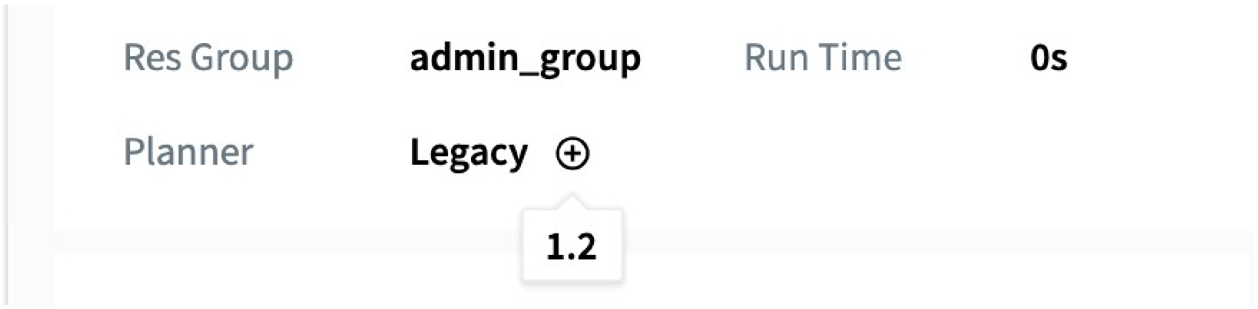
The name of the database that was queried.

Res Group / Res Queue

The name of the resource group or resource queue that is managing the query.

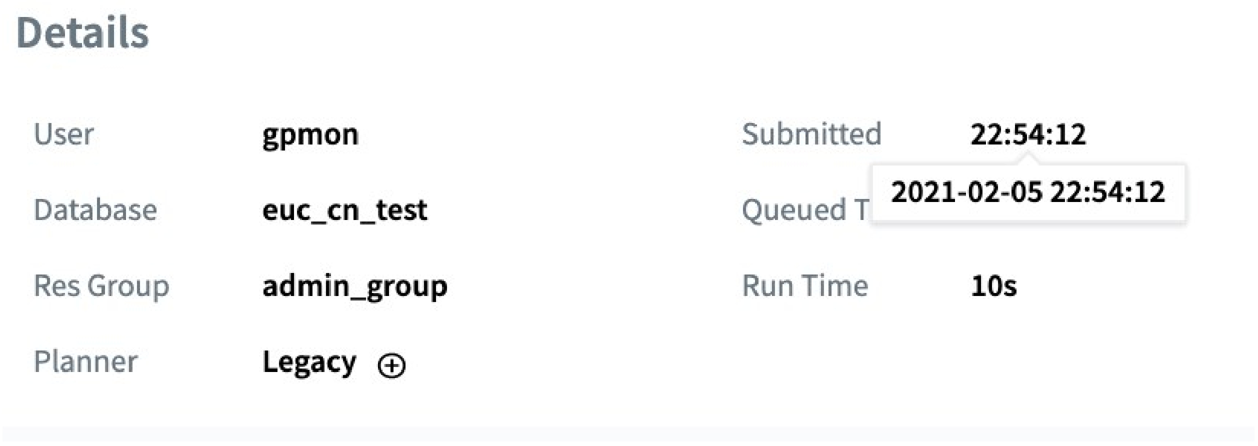
Planner

The name of the query planner used for this query, GPORCA or Legacy. Press the "+" to see the cost of the query, and copy using the mouse.


Res Group admin_group Run Time 0s
Planner Legacy ⊕
1.2

Submitted

The time the query was submitted to the query planner. Hover over the submitted time to view the submitted date.


Details

User	gpmon	Submitted	22:54:12
Database	euc_cn_test	Queued T	2021-02-05 22:54:12
Res Group	admin_group	Run Time	10s
Planner	Legacy ⊕		

Queued Time

The time the query waited before it was executed. In addition to time in the queue, this includes other time such as time in the optimizer.

Run Time

The amount of time the query executed.

```
Under the **Performance** heading, the metrics indicate the final state of the query b
efore it ends:
```

CPU Master

The CPU percent on the Greenplum Database master host for this query.

CPU Segments

CPU percent average for all segment processes executing this query. The percentages for all processes running on each segment are averaged, and then the average of all those values is calculated to render this metric. The master and standby master are excluded from the calculation.

CPU Time

Total CPU time consumed by all processes on all segments executing this query.

CPU Skew

The amount of CPU skew. CPU skew occurs when query executor processes for one segment use a disproportionate amount of CPU compared to processes for other segments executing the query. This value is calculated as `1 - (average_segment_CPU / maximum_segment_CPU)`

Memory

Memory consumed by all segment processes executing the query.

Spill Files</dt`> The total size of spill files created for the query. Greenplum Database creates spill files when there is insufficient memory to execute the query in memory. See Managing Spill Files Generated by Queries for information about spill files.

Disk R

The average disk read rate for all segment hosts.

Disk W

The average disk write rate for all segment hosts.

Locks and Blocks

Contains lists of queries blocked by locks this query holds, and queries held by other transactions that block this transaction. Click a list to expand and view the contents.

Query Text and Execution Plan

The query text and the query's plan and execution progress are shown in the lower panels of the Query Details view. The text of the query is displayed in the left panel, and the plan and progress is displayed in the right panel. The plan is available only for queries that ran for at least ten seconds.

Query Text

The **Query Text** panel displays the text of the query as it was submitted to Greenplum Database.

Command Center can display up to 100K characters. If you click **COPY**, up to 100K characters of the query text are copied to the clipboard.

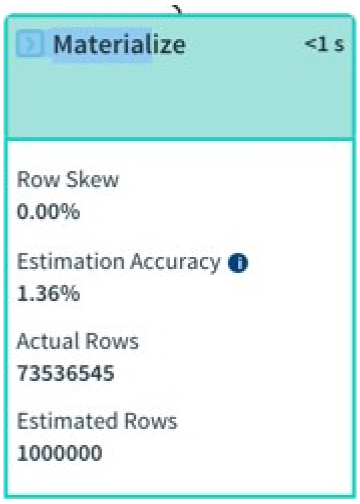
If the query text is longer than 100K characters, a message is displayed with a link you can use to download the full text of the query. The name of the text file is the ID of the query with a `.txt` extension. The file is available to download for 24 hours.

Query Plan and Progress

The **Plan & Progress** tab in the lower right panel is a graphical representation of the query plan with metrics to show the planned and actual query execution. Each box in the tree represents a step in the query execution plan. The boxes are labeled with the operation they represent. Query execution begins at the bottom of the tree and ends at the top.

Query execution plans are executed in "slices," portions of the query plan that segments can work on independently in parallel. The plan is sliced wherever a data motion operator occurs. The time displayed in the upper right corner of each box is the amount of CPU time used for all steps in the slice to which the step belongs. Each slice is displayed in its own color, making it easy to distinguish the slices from each other. The visual query plan does not illustrate slices, but you can find information about slices in the textual plan.

If you click a step, the box expands to show additional details.


Materialize

<1 s

Row Skew
0.00%

Estimation Accuracy
1.36%

Actual Rows
73536545

Estimated Rows
1000000

The expanded plan box contains the following metrics.

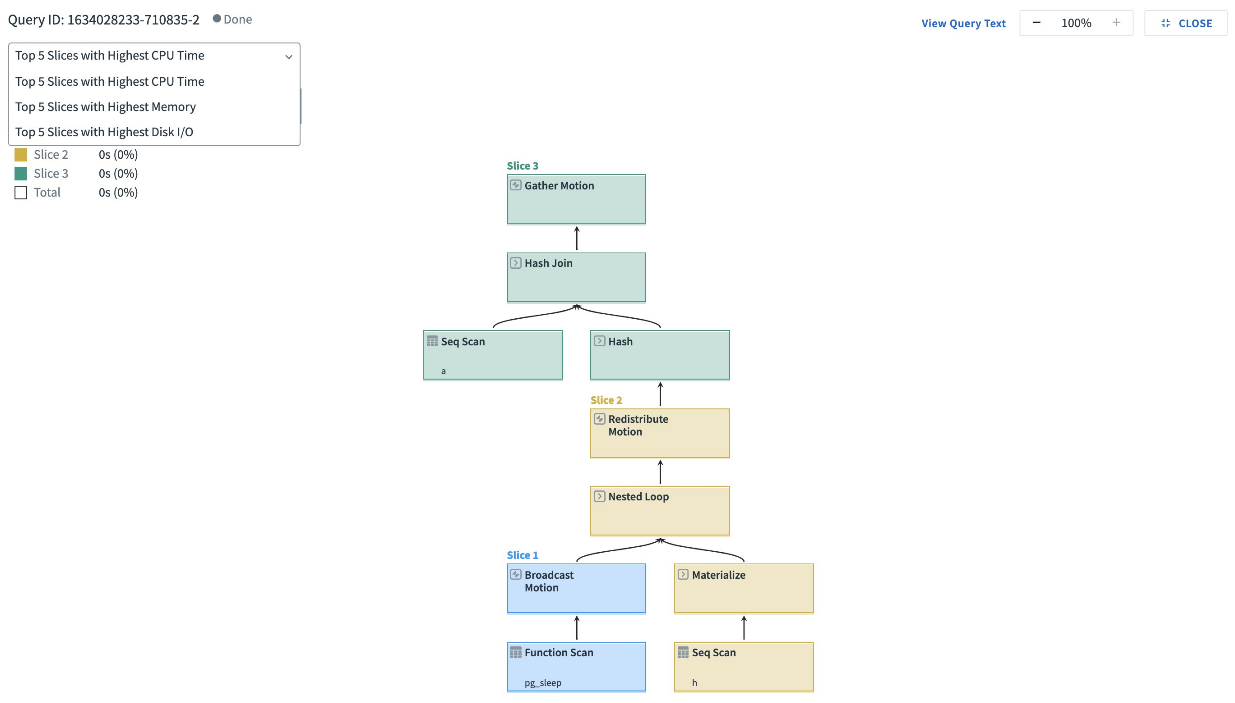
- The type of operator. When the operator is a table scan, the name of the scanned table is included. See Query Plan Execution for descriptions of the operators.
- Information related to the current operation, such as the hash key, merge key, join condition, or filter condition.
- Row Skew - the amount of row skew for the current operator, a value from 0% to 100%. Row skew occurs when some segments process more rows than other segments. The percentage is calculated as `(1 - (average_segment_rows / maximum_segment_rows)) * 100`. Nodes with a row skew of greater than 30% are marked with an exclamation point.
- Estimation Accuracy - a percentage calculated from the estimated rows the node would produce and the actual rows the node produced when the query executed. The percentage is calculated as `min(estimated_rows, actual_rows) / max(estimated_rows, actual_rows) * 100`
- Actual Rows - The number of rows produced by this step.
- Estimated Rows - The estimated number of rows the operator will produce.

Display Slice Metrics

From within the **Plan and Progress** tab you may display key metrics about slices, including:

- the top 5 slices that consume the most CPU time
- the top 5 slices that consume the most memory
- the top 5 slices that use the most disk I/O

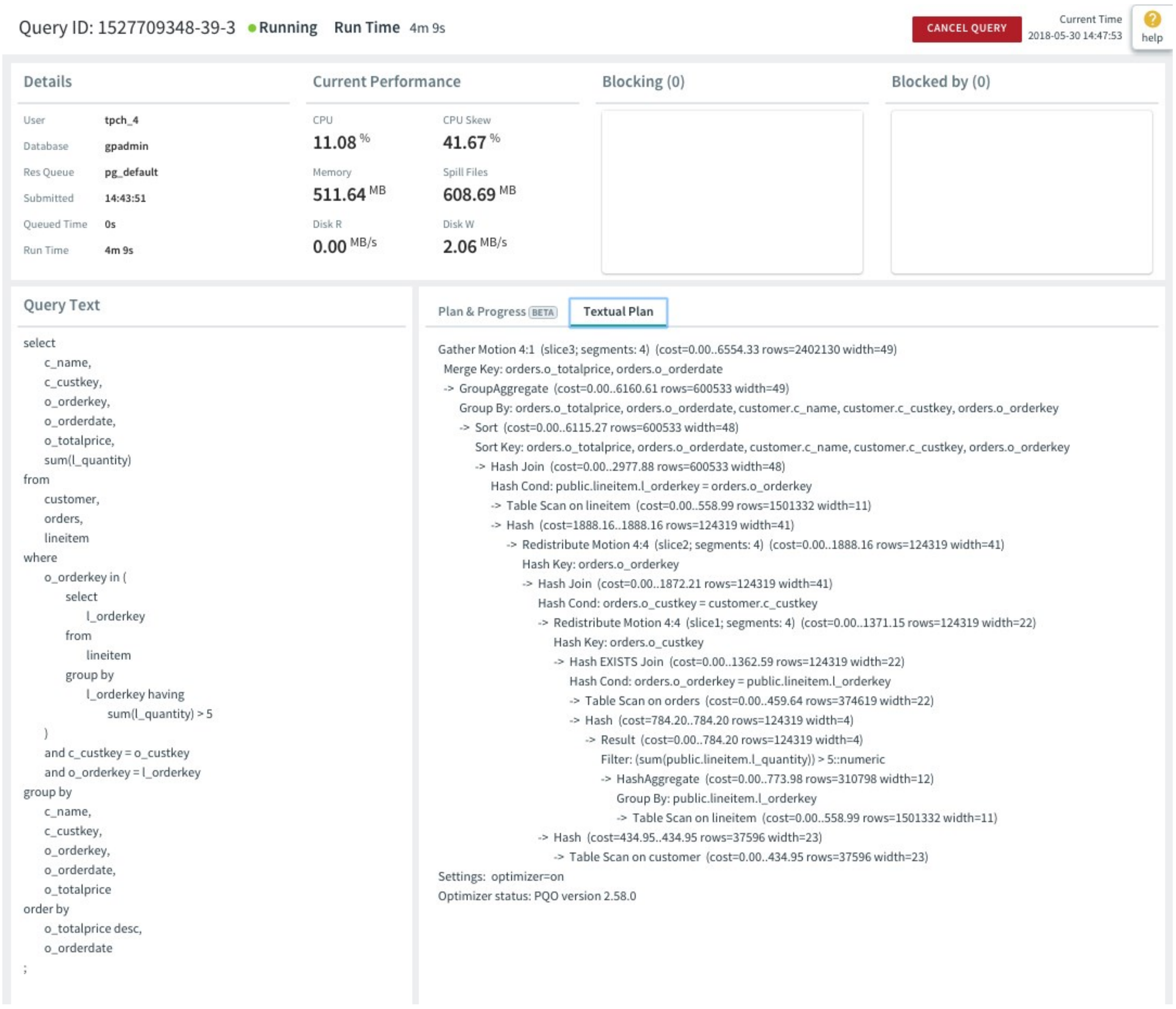
To display this information, click the **Expand** button on the far right. This displays a screen with a dropdown menu from which you can choose which slice metrics you want to display:



Textual Plan

Select the **Textual Plan** tab and click **RUN EXPLAIN** to generate the text representation of the explain plan. The `EXPLAIN` command is run against the current state of the database, so the plan generated may differ from the the plan used when the query executed.

Note: The **RUN EXPLAIN** button is dimmed if Command Center is unable to generate the explain plan. Command Center is unable to generate the explain plan if the size of the query text is greater than 100K characters or if the query text contains multiple statements.



User	tpch_4
Database	gpadmin
Res Queue	pg_default
Submitted	14:43:51
Queued Time	0s
Run Time	4m 9s

CPU	11.08 %
CPU Skew	41.67 %
Memory	511.64 MB
Spill Files	608.69 MB
Disk R	0.00 MB/s
Disk W	2.06 MB/s

```
select
    c_name,
    c_custkey,
    o_orderkey,
    o_orderdate,
    o_totalprice,
    sum(l_quantity)
from
    customer,
    orders,
    lineitem
where
    o_orderkey in (
        select
            l_orderkey
        from
            lineitem
        group by
            l_orderkey having
                sum(l_quantity) > 5
    )
    and c_custkey = o_custkey
    and o_orderkey = l_orderkey
group by
    c_name,
    c_custkey,
    o_orderkey,
    o_orderdate,
    o_totalprice
order by
    o_totalprice desc,
    o_orderdate
;
```

```
Gather Motion 4:1  (slice3; segments: 4)  (cost=0.00..6554.33 rows=2402130 width=49)
 Merge Key: orders.o_totalprice, orders.o_orderdate
 -> GroupAggregate  (cost=0.00..6160.61 rows=600533 width=49)
     Group By: orders.o_totalprice, orders.o_orderdate, customer.c_name, customer.c_custkey, orders.o_orderkey
     -> Sort  (cost=0.00..6115.27 rows=600533 width=48)
         Sort Key: orders.o_totalprice, orders.o_orderdate, customer.c_name, customer.c_custkey, orders.o_orderkey
         -> Hash Join  (cost=0.00..2977.88 rows=600533 width=48)
             Hash Cond: public.lineitem.l_orderkey = orders.o_orderkey
             -> Table Scan on lineitem  (cost=0.00..558.99 rows=1501332 width=11)
             -> Hash  (cost=1888.16..1888.16 rows=124319 width=41)
                 -> Redistribute Motion 4:4  (slice2; segments: 4)  (cost=0.00..1888.16 rows=124319 width=41)
                     Hash Key: orders.o_orderkey
                     -> Hash Join  (cost=0.00..1872.21 rows=124319 width=41)
                         Hash Cond: orders.o_custkey = customer.c_custkey
                         -> Redistribute Motion 4:4  (slice1; segments: 4)  (cost=0.00..1371.15 rows=124319 width=22)
                             Hash Key: orders.o_custkey
                             -> Hash EXISTS Join  (cost=0.00..1362.59 rows=124319 width=22)
                                 Hash Cond: orders.o_orderkey = public.lineitem.l_orderkey
                                 -> Table Scan on orders  (cost=0.00..459.64 rows=374619 width=22)
                                 -> Hash  (cost=784.20..784.20 rows=124319 width=4)
                                     -> Result  (cost=0.00..784.20 rows=124319 width=4)
                                         Filter: (sum(public.lineitem.l_quantity)) > 5::numeric
                                         -> HashAggregate  (cost=0.00..773.98 rows=310798 width=12)
                                             Group By: public.lineitem.l_orderkey
                                             -> Table Scan on lineitem  (cost=0.00..558.99 rows=1501332 width=11)
                         -> Hash  (cost=434.95..434.95 rows=37596 width=23)
                             -> Table Scan on customer  (cost=0.00..434.95 rows=37596 width=23)
Settings:  optimizer=on
Optimizer status: PQO version 2.58.0
```

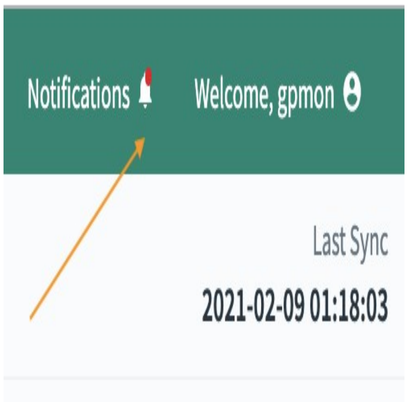
The textual plan is the output of the Greenplum Database `EXPLAIN` command for the query. The query plan steps are labeled with arrows (`->`) and the structure of the query plan tree is indicated with indentation.

The `Optimizer status:` line at the bottom of the textual plan reports whether the explain plan was generated using the GPORCA optimizer (PQO) or the legacy query optimizer.

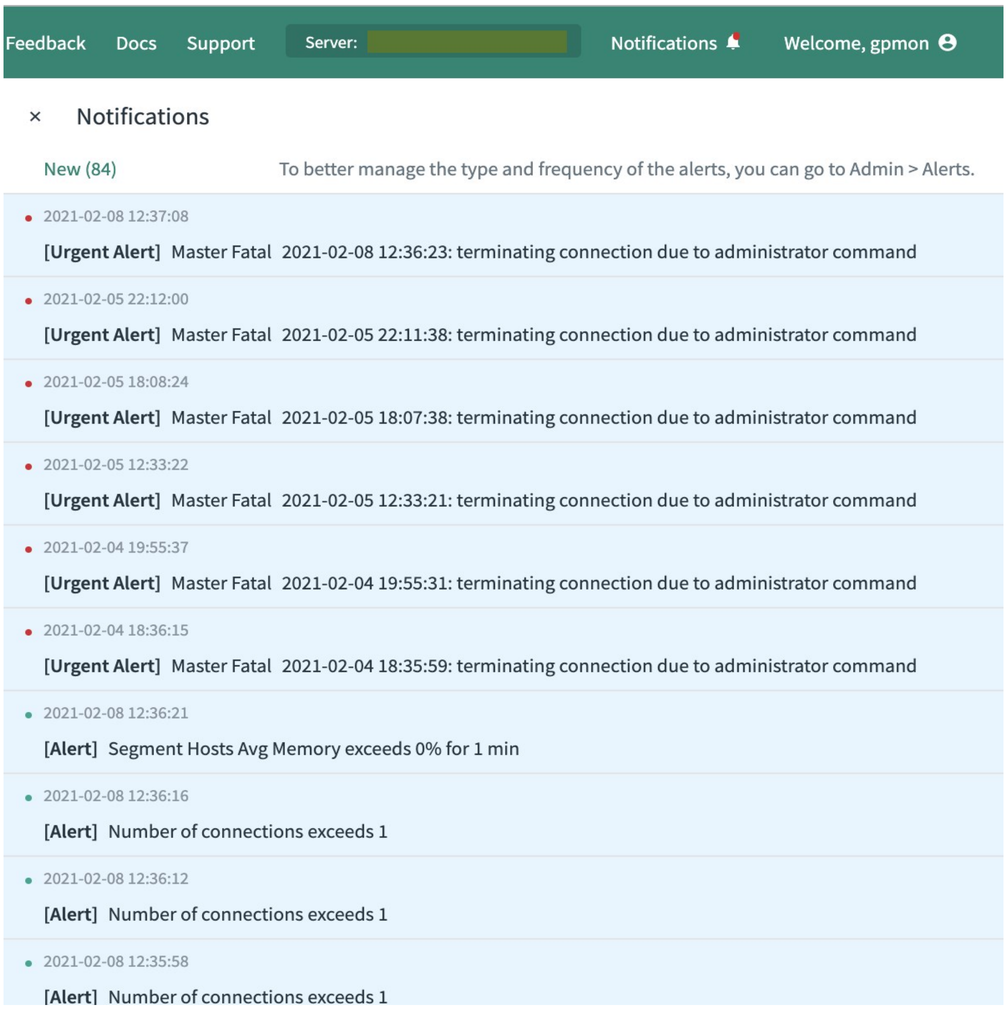
For help reading the textual explain plan see the `EXPLAIN` command in the *Greenplum Database Reference Guide* and Query Profiling in the *Greenplum Database Administrator Guide*. See Query Execution for descriptions of the query operators.

Alerts

To quickly view any notification alerts, use the **Notifications** label on the header bar of the Command Center dashboard.



When you open the **Notifications** window, unread notifications are highlighted in light blue. The alerts are ordered by time and are kept for 7 days. When you close the **Notifications** window, all alerts are considered read.



To better manage the type and frequency of the alerts, you can go to Admin > Alerts.

```
2021-02-08 12:37:08
[Urgent Alert]  Master Fatal  2021-02-08 12:36:23: terminating connection due to administrator command
2021-02-05 22:12:00
[Urgent Alert]  Master Fatal  2021-02-05 22:11:38: terminating connection due to administrator command
2021-02-05 18:08:24
[Urgent Alert]  Master Fatal  2021-02-05 18:07:38: terminating connection due to administrator command
2021-02-05 12:33:22
[Urgent Alert]  Master Fatal  2021-02-05 12:33:21: terminating connection due to administrator command
2021-02-04 19:55:37
[Urgent Alert]  Master Fatal  2021-02-04 19:55:31: terminating connection due to administrator command
2021-02-04 18:36:15
[Urgent Alert]  Master Fatal  2021-02-04 18:35:59: terminating connection due to administrator command
2021-02-08 12:36:21
[Alert]  Segment Hosts Avg Memory exceeds 0% for 1 min
2021-02-08 12:36:16
[Alert]  Number of connections exceeds 1
2021-02-08 12:36:12
[Alert]  Number of connections exceeds 1
2021-02-08 12:35:58
[Alert]  Number of connections exceeds 1
```

To customize the alert notifications, go to the **Admin> Alerts** page, where an administrator can set up **alert rules**. The rules help detect and respond to events that occur in the Greenplum Database system and in currently executing database queries. When a rule is matched, Command Center logs a record.

You can set up email alerts by configuring an SMTP server in Greenplum Database or in Command Center. Additionally, you can create a `send_alert.sh` shell script to forward alerts to other destinations, such as an SMS gateway or a Slack channel. If the script is present, Command Center runs it whenever an alert is raised.

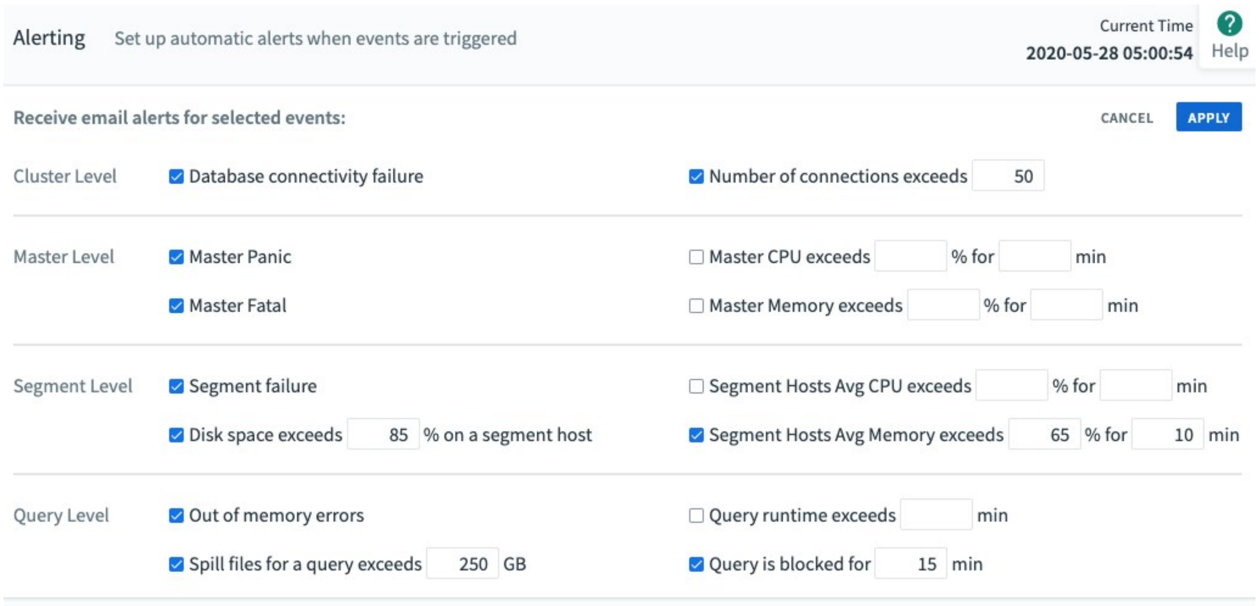
Command Center creates the `gpmetrics` schema in the gpperfmon database to store both rules and log records. See gpmetrics Schema Reference for information about the `gpcc_alert_rule` and `gpcc_alert_log` tables in the `gpmetrics` schema.

This topic contains the following subtopics:

- Configuring Alert Rules
- Configuring Alert Email
- Creating a Send Alert Script

Configuring Alert Rules

Click **EDIT** to manage alert event rules. To enable an alert rule, enter any data required in the fields and check the box. Uncheck the box to disable the rule. Click **ALERT** when you have finished making changes to the alert configuration.



Receive email alerts for selected events:

Cluster Level	✓ Database connectivity failure	✓ Number of connections exceeds 50
Master Level	✓ Master Panic	☐ Master CPU exceeds % for min
	✓ Master Fatal	☐ Master Memory exceeds % for min
Segment Level	✓ Segment failure	☐ Segment Hosts Avg CPU exceeds % for min
	✓ Disk space exceeds 85 % on a segment host	✓ Segment Hosts Avg Memory exceeds 65 % for 10 min
Query Level	✓ Out of memory errors	☐ Query runtime exceeds min
	✓ Spill files for a query exceeds 250 GB	✓ Query is blocked for 15 min

Cluster Level

Database connectivity failure

An alert is raised when either of these conditions is detected:

- Command Center is running on the master host, the master host is up, but the database is down or cannot be reached.
- Command Center is running on the standby master host, the standby master host is up, but the master host is down or cannot be reached, or the database is down or cannot be reached.

Command Center checks three times before raising the alert.

If the host where Command Center should be running is down, no alert is raised.

Number of connections exceeds [N]

An alert is raised when the total number of database connections exceeds the number specified. The number of connections is checked every 30 seconds. After an alert is raised, the metrics collector checks the number of connections every 30 minutes until the number of connections drops below the threshold, and then it resumes checking every 30 seconds.

Master Level

Master Panic

An alert is raised when messages with PANIC severity are detected at the master host. An PANIC error causes all database sessions to abort.

Master Fatal

An alert is raised when messages with FATAL severity are detected at the master host. A FATAL error causes the current session to abort. To reduce noise, fatal messages logged due to password authentication errors do not raise alerts.

Master CPU exceeds [%] for [N] min

An alert is raised when the CPU usage on the master host exceeds the specified percentage for the specified number of minutes. Command Center samples CPU usage on the master host every 15 seconds and calculates the mean of the samples.

Master Memory exceeds [%] for [N] min

An alert is raised when the percent of memory used on the master host exceeds the specified percentage for the specified number of minutes. Command Center samples memory usage on the master host every 15 seconds and calculates the mean of the samples. Only memory in use is considered; memory for buffers and cache is not included.

Segment Level

Segment failure

An alert is raised when one or more failed segments are detected. After the alert email is raised, Command Center will raise the alert every 30 minutes until the segments are recovered.

Total disk space exceeds [%] on a segment host

An alert is raised when the total of disk space in use for all segment hosts exceeds the specified percentage. Command Center gathers the *available disk space* and *total disk space* from each segment host in the Greenplum Database cluster. The *percent of total disk space in use* is calculated by the following formula:

```
    100 - sum(<available disk space>) / sum(<total disk space>) * 100
```

A disk space alert is raised no more than once every 24 hours.

Segment Hosts Avg CPU exceeds [%] for [N] min

An alert is raised when the average percent of CPU used for all segment hosts exceeds the specified percentage for the specified number of minutes. Command Center samples all segment hosts every 15 seconds and calculates the mean of the samples.

Segment Hosts Avg Memory exceeds [%] for [N] min

An alert is raised when the average memory for all segment hosts exceeds the specified percentage for the specified number of minutes. Command Center samples all segment hosts every 15 seconds and calculates the mean of the samples. Only memory in use is considered; memory for buffers and cache is not included.

Query Level

Out of memory errors

An alert is raised when an executing query fails with an out of memory (OOM) error. Note that no alert is raised if there is insufficient memory to start the query.

Spill files for a query exceeds [N] GB

An alert is raised when the total disk space consumed by a running query's spill files exceeds the specified number of gigabytes. An alert is raised only once per query.

Query runtime exceeds [N] min

An alert is raised when a query runtime exceeds the number of minutes specified. This alert is raised just once for a query.

Query is blocked for [N] min

An alert is raised if a query remains in a blocked state for longer than the specified number of minutes. If an alert is raised, and then the query unblocks, runs, and blocks again for the specified time, an additional alert is raised. Blocked time excludes the time a query is queued before it runs. It is possible for a "Query runtime exceeds [N] min" rule to also trigger while a query is blocked.

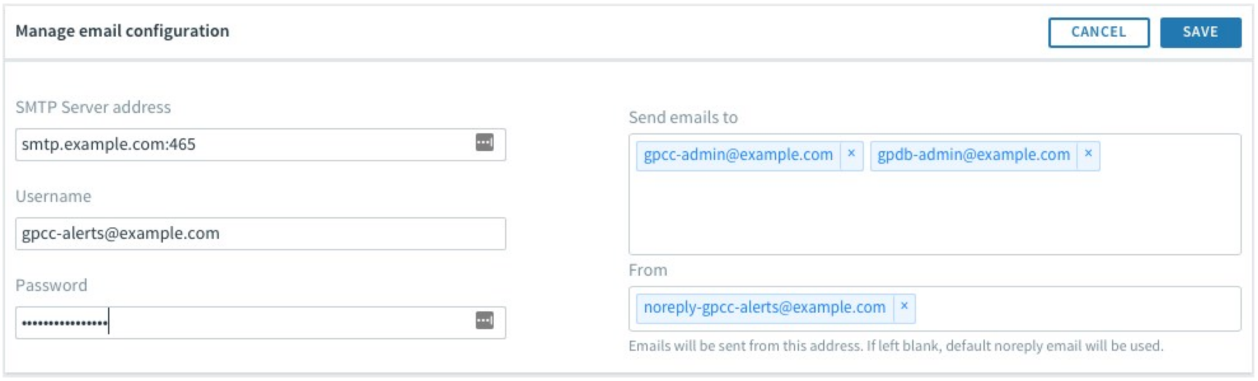
Configuring Alert Email

Command Center requires an SMTP server to send alert emails.

Command Center first attempts an encrypted TLS connection and then falls back to an unencrypted connection if TLS is not supported. The SMTP server must support one of the following authentication methods: NONE, PLAIN, LOGIN, or CRAM-MD5. Command Center will use the most secure of these methods the SMTP server allows.

Configuring email With Command Center

Click **EDIT** in the **Manage email configuration** panel.



SMTP Server address: smtp.example.com:465

Username: gpcc-alerts@example.com

Password: ••••••••••••••••

Send emails to: gpcc-admin@example.com, gpdb-admin@example.com

From: noreply-gpcc-alerts@example.com

Emails will be sent from this address. If left blank, default noreply email will be used.

The alert email configuration is set with the following Greenplum Database server configuration parameters:

SMTP Server address

The name or IP address of the SMTP server and the SMTP port number. The port number is typically

587 for connections with TLS encryption or 465 without encryption. Example: `smtp.example.com:465`

Username

The username of the account to authenticate with the SMTP server. This is an optional field, only required if the SMTP server requires a username for authentication. Example: `gpcc-alerts@example.com`

Password

The password for the SMTP username. For security, the password is masked. This field is optional, only needed if the SMTP server requires a username and password for authentication.

Send emails to

To add an address to the list, enter the address and press Enter. To remove an email address, click the `x` on the address.

From

The email address to use for the `From:` address in the alert email. Example: `do-not-reply@example.com`.

If you leave this field blank, Command Center uses the default value, `noreply-gpcc-alerts@pivotal.io`.

When you click **SAVE**, Command Center sends a test email to the addresses in the **Send emails to** field. The email contains a list of the currently configured alert rules. If there is an error in the SMTP server or username/password configuration and the email cannot be sent, Command Center displays an error message.

Configuring email for Greenplum Database

The following server configuration parameters are used to configure SMTP email for Greenplum Database.

gp_email_smtp_server

The SMTP server and port. Example: `smtp.example.com:465`

gp_email_smtp_userid

The name of a user to authenticate with the SMTP service. Example: `gpcc-alerts@example.com`

gp_email_smtp_password

The password for the SMTP user.

gp_email_from

The email address to set as the email sender. Example: `noreply-gpcc-alerts@example.com`

gp_email_to

A semicolon-separated list of email addresses to receive alert messages. Example `gpcc-admin@example.com;gpdb-admin@example.com`

Command Center uses the `gp_email_smtp_server`, `gp_email_smtp_userid`, and `gp_email_smtp_password` parameters if they are set. It ignores the remaining parameters.

You can check the current value of a configuration parameter by running the `gpconfig -s` command on the master host, for example:

```
$ gpconfig -s gp_email_smtp_server
```

Use the `gpconfig -c` option to set the values of server configuration parameters, for example:

```
$ gpconfig -c gp_email_smtp_server -v "smtp.example.com:465"
$ gpconfig -c gp_email_smtp_userid -v "gpcc-alerts@example.com"
$ gpconfig -c gp_email_smtp_password -v "changeme"
$ gpconfig -c gp_email_from -v "gpcc-alerts@example.com"
$ gpconfig -c gp_email_to -v "gpcc-admin@example.com;gpdb-admin@example.com"
```

Run `gpstop -u` to reload the configuration files after changing these configuration parameters.

Creating a Send Alert Script

The send alert script is a shell script that you can use to send Command Center alerts to destinations such as SMS gateways, pagers, team collaboration tools like Slack, chat servers, archive files, alternative email servers, and so on. You can use the send alert script in addition to sending email from Command Center, or as an alternative to sending alert emails from Command Center.

Command Center looks for the script `$MASTER_DATA_DIRECTORY/gpmetrics/send_alert.sh` on the host where Command Center is running—either the master host or standby host. If the file exists and is executable by the gpadmin user, Command Center executes the script. The following variables are set on the command line when the script runs.

Variable	Description
LINK	URL of the Greenplum Command Center web server.
QUERYID	ID of the query, if the alert was triggered by a query.
SERVERNAME	Name of the Greenplum Command Center server.
QUERYTEXT	The text of the query, if the alert was triggered by a query.
ACTIVERULENAME	Current text the of rule, with user-specified values included.
LOGID	Value of this alert's `id` column in the `gpmetrics.gpcc_alert_log` table.
RULEDESCRIPTION	Text of the rule, including user-specified values, at the time the alert was raised.
ALERTDATE	Date the alert was raised.
ALERTTIME	Time the alert was raised.
SUBJECT	Subject line for email.

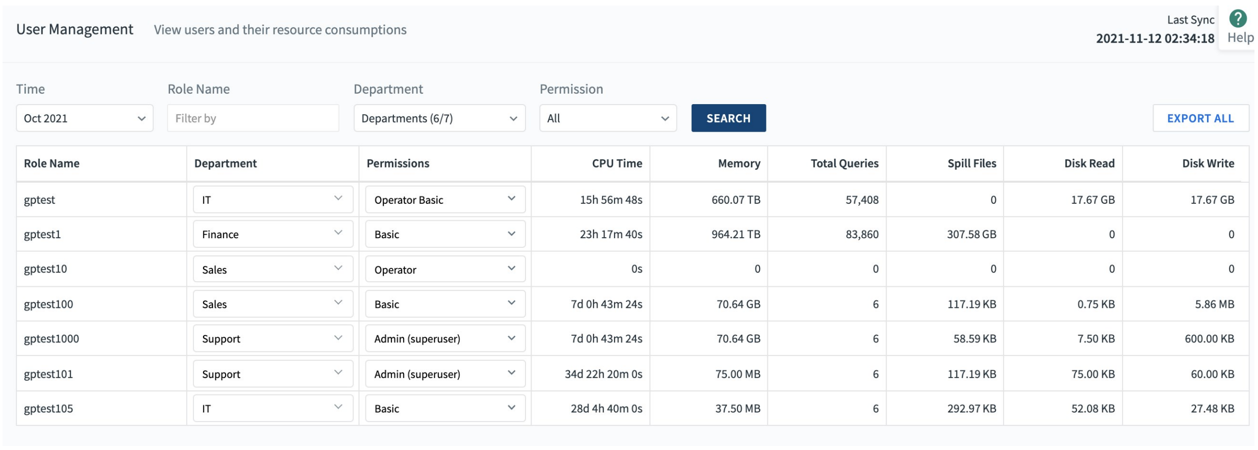
An example script that you can customize is provided at `$GPCC_HOME/alert-email/send_alert.sh.sample`. The example formats the alert as HTML email text and pipes it through the Linux `mail` command.

To set up a send alert script:

1. Copy the `$GPCC_HOME/alert-email/send_alert.sh.sample` file to `$MASTER_DATA_DIRECTORY/gpmetrics/send_alert.sh`.

2. Customize the script with code to format and deliver the alert to your desired destination.
3. Run `gpcc start` to restart Command Center and enable the script.

Managing Users



Role Name	Department	Permissions	CPU Time	Memory	Total Queries	Spill Files	Disk Read	Disk Write
gptest	IT	Operator Basic	15h 56m 48s	660.07 TB	57,408	0	17.67 GB	17.67 GB
gptest1	Finance	Basic	23h 17m 40s	964.21 TB	83,860	307.58 GB	0	0
gptest10	Sales	Operator	0s	0	0	0	0	0
gptest100	Sales	Basic	7d 0h 43m 24s	70.64 GB	6	117.19 KB	0.75 KB	5.86 MB
gptest1000	Support	Admin (superuser)	7d 0h 43m 24s	70.64 GB	6	58.59 KB	7.50 KB	600.00 KB
gptest101	Support	Admin (superuser)	34d 22h 20m 0s	75.00 MB	6	117.19 KB	75.00 KB	60.00 KB
gptest105	IT	Basic	28d 4h 40m 0s	37.50 MB	6	292.97 KB	52.08 KB	27.48 KB

The Command Center's **User Management** view allows administrators to peform a variety of user management tasks, including:

- Displaying resource usage for each role for a specific month
- Displaying resource usage for specific roles
- Displaying resource usage for for one or more specific departments
- Displaying resource usage for roles with no department
- Displaying roles holding a specific permission
- Displaying resource usage for all permission levels
- Creating a department name
- Assigning a department name to a specific role
- Changing a role's permissions
- Exporting resource usage of specific roles
- Renaming a department using a user-defined function
- Deleting an unwanted department name from the `gpperfmon` database using a user-defined function

The resources displayed include:

- CPU Time
- Memory
- Total Queries
- Spill Files
- Disk Read

- Disk Write

Display Resource Usage by Role Per Month

This task requires Operator Basic or higher permission.

To display resource usage by role per month:

1. Click the Time dropdown menu
2. Select a month
3. Click the SEARCH button

Command Center displays resource usage for all roles for that month.

Display Resource Usage for Specific Roles

This task requires Operator Basic or higher permission.

Note: The results returned may not cover up to the preceding 6 hours, due to delays.

To display resource usage for specific roles:

1. Enter the role name in the Role Name box
2. Click the SEARCH button

Command Center displays resource usage for that role.

Note: The resource usage of the gpmon and gpadmin roles is not included.

Display Resource Usage for Departments

These tasks require Operator Basic or higher permission.

To display resource usage for a specific department:

1. Select a department from the Department dropdown menu
2. Click the SEARCH button

Command Center displays resource usage for that department.

To display resource usage for all departments:

1. Select all departments from the Department dropdown menu
2. Click the SEARCH button

Command Center displays resource usage for all departments.

To display resource usage for a selection of departments:

1. Select multiple departments from the Department dropdown menu
2. Click the SEARCH button

Command Center displays resource usage for this selection of departments.

Display Resource Usage for Roles with No Department

This task requires Operator Basic or higher permission.

To display resource usage for roles with no department:

1. Select "--" from the Department dropdown
2. Click the SEARCH button

Command Center displays resource usage for roles who have no department.

Display Resource Usage for Roles Holding a Specific Permission

This task requires Operator Basic or higher permission.

To display resource usage for roles holding a specific permission:

1. Select a permission from the Permission dropdown menu
2. Click the SEARCH button

Command Center displays resource usage for roles with that permission.

Display Resource Usage for All Permission Levels

This task requires Operator Basic or higher permission.

To display resource usage for all permission levels:

1. Select "All" from the "Permission" dropdown menu
2. Click the SEARCH button

Command Center displays resource usage for all permission levels.

Create a Department Name

This task requires Admin permission.

To create a department name and apply it to a specific role:

1. Click the dropdown menu in the Department column
2. Enter a new department name
3. Hit Enter
4. Click the Department dropdown again; the new department name is now visible

Assign a Department Name to a Specific Role

This task requires Admin permission.

To assign a department name to a specific role:

1. Click the dropdown menu in the Department column
2. Select a name from the list

3. That department name is now associated with the role

Change a Role's Permissions

This task requires Admin permission.

To set a role's permission level:

1. Click the dropdown menu in the Permissions column
2. Select the desired permission
3. That permission is now associated with the role

Export Resource Usage of Specific Roles

This task requires Operator Basic permission.

To export resource usage of specific roles:

1. Specify search conditions in the **User Management** interface
2. Click the SEARCH button
3. View the resource usage report
4. Click the EXPORT ALL button

A CSV file that begins with the string `gpcc_user_usage_report` is downloaded to your machine's default download location.

Rename a Department with the `gpcc_update_department` User-Defined Function

You may use the `gpcc_update_department` user-defined function to rename a department name, passing it the old name (`from_name`) and the new name (`to_name`). Once it has run, all the roles associated with the old name will now be associated with the new name.

For more information on the `gpcc-rename_dept_name` user-defined function, see the gpmetrics Schema Reference documentation page.

Delete a Department Name from the gpperfmon Database with the `gpcc_delete_department` User-Defined Function

You may use the `gpcc_delete_department` user-defined function to delete a department name, passing it the name of the department you want to delete. You may only delete department names that have no roles associated with them.

For more information on the `gpcc_delete_department` user-defined function, see the gpmetrics Schema Reference documentation page.

Managing Greenplum Database Workloads

About Greenplum Command Center Workload Management

Greenplum Database segment hosts have a set amount of memory, CPU, I/O, and network resources. If these resources start to run short due to heavy database load, queries can fail or the entire database system can become unacceptably slow. For this reason, it is important to manage database requests to prevent resource depletion.

Note: Greenplum Database has two resource management systems: *resource queues* and *resource groups*. Command Center workload management is based on resource groups, the resource management system introduced in Greenplum Database version 5. Resource groups require enabling Linux control groups (cgroups), so Greenplum Database initially is set to perform resource management using resource queues.

When Command Center starts, it checks the resource management system enabled in your Greenplum Database system. If you have not yet enabled resource groups in your Greenplum Database system, the Command Center workload management view displays a message encouraging you to enable resource groups, including a link to documentation with the procedure for completing the task. When you start Command Center after enabling resource groups, click the **ENABLE WORKLOADS** button. Command Center presents a view to help you set the intial resource group configuration by importing your existing resource queues to resource groups.

See Using Resource Groups in the *Greenplum Database Administrator Guide* for a full description of resource management features available with resource groups and instructions to enable resource groups in Greenplum Database.

On the Command Center **Workload> Resource Groups** view, you can perform the following tasks:

- Create new resource groups in Greenplum Database
- Delete existing resource groups
- Change the number of concurrent transactions each resource group allows
- Change the percentages of available system CPU and memory each resource group manages
- Change the default resource group assignment for Greenplum Database roles
- Set the amount of time a session can be idle in a resource group before it is killed

See Resource Groups for details about creating resource groups and query timeout rules.

For more information about Linux cgroups and Greenplum Database resource groups see Using Resource Groups in the *Greenplum Database Administrator Guide*.

On the Command Center **Workload> Workload Mgmt** view, you can perform the following tasks:

- Create workload management rules to cancel a query or move it to a different resource group, based on conditions such as the amount of CPU time used, planner cost, query running time, and disk I/O utilitization.
- Create workload management rules to terminate an idle session based on which resource group it belongs to and how long it has been idle.
- Monitor workload management rules when they perform actions against matching queries or idle sessions.

Workload management rules allow you to flexibly assign transactions to resource groups or cancel queries or idle sessions. The following sections provide more information about these Command Center features.

About Query Assignment Rules and Workload Rules

Greenplum Database defers to the workload management database extension to assign transactions to resource groups. Command Center users with Admin permission level can create assignment rules in Command Center to assign transactions to resource groups before they run.

When a transaction begins, Greenplum Database calls the workload management extension to determine the resource group. The extension evaluates the assignment rules and, if a matching rule is found, returns that rule's resource group. If no assignment rule matches, Greenplum Database falls back to the default behavior, assigning the transaction to the resource group specified for the current user in the `pg_authid` system table.

Assignment rules can redirect a transaction to a resource group based on any combination of query tag and/or the Greenplum Database role that executes the transaction.

A query tag is a user-defined *name*=*value* pair that you can set in a Greenplum Database session when resource group-based workload management is enabled. Query tags are defined by setting the `gpcc.query_tags` parameter on the connect string when requesting a database connection or in the session with `SET gpcc.query_tags TO '<query-tags>'`. Multiple query tags can be set by separating them with a semicolon. Query tags are set before a transaction begins and cannot be changed inside of a transaction.

If multiple assignment rules would match the same query, Command Center matches only the first assignment rule (the rule with the lowest ID number) to the query.

See Workload Management for details about creating assignment rules, and for examples that use query tags.

Workload Rules

A workload rule defines:

- For workload rules for queries, additional conditions beyond query assignment rule conditions under which a running query may be canceled or (for Greenplum versions 6.8 or later) moved into a different resource group

Workload rules for queries are first matched to a transaction based on any combination of the database role and query tags. After the transaction execution begins, additional conditions are evaluated to determine if configured CPU time, I/O, slice, and/or planner cost limits are exceeded.

- For workload rules for idle sessions, the conditions under which an an idle session may be terminated

If all configured conditions for a workload rule are exceeded, the rule action action is performed against the transaction. Available actions are either cancelling the transaction or, for Greenplum versions 6.8 or later, moving the transaction to a different resource group.

Command Center always ensures that a given workload rule is applied only once to a matched transaction. For example, two workload rules could be created with one rule that moves queries from resource group A to group B based on planner cost, and another rule that moves queries from any resource group to group A based on the query running time. The first of these rules would only be applied to a given query once, even if the second rule again placed the query back into resource group A.

See Workload Management for details about creating workload rules.

About Timeout Rules

Command Center administrators can create session timeout rules for resource groups that specify the maximum time that a session can remain idle before it is terminated. When the session process on the Greenplum Database master becomes idle and the specified time has elapsed, the session terminates itself.

Session timeout rules are per resource group, allowing you to determine how long to wait for different classes of transactions. You can include a list of Greenplum Database roles to exempt from the rule and a custom message to add to the session termination error output.

Warning: Creating timeout rules for the `admin_group` resource group is not recommended. Avoid creating rules that kill sessions started by Greenplum Database client applications that are designed to have a long-lasting or persistent session. For example, the `gpmmon` process creates a session as the `gpmon` role to update the gpperfmon database. An idle session timeout rule that kills idle `gpmon` sessions too quickly could cause Greenplum Database to log many session termination messages and unnecessarily create new sessions.

See Workload Management for details about creating timeout rules.

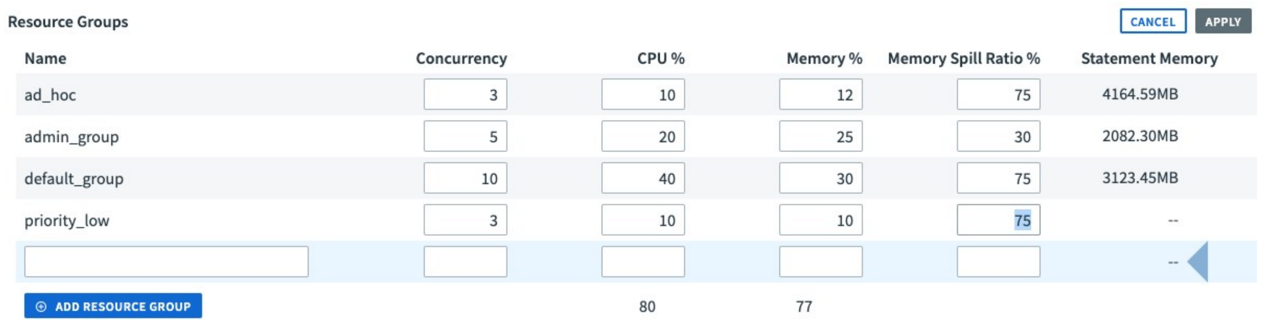
Resource Groups

Defining Resource Groups and Resource Attributes

Command Center allows you to view resource groups that have been created in Greenplum Database, to add or delete resource groups, and to edit the resource group attributes **Concurrency**, **CPU %**, and **Memory %**.

Note: To change values of the `MEMORY_AUDITOR`, `CPUSET`, `MEMORY_SHARED_QUOTA`, or

MEMORY_SPILL_RATIO resource group attributes, use the ALTER RESOURCE GROUP SQL command.



Name	Concurrency	CPU %	Memory %	Memory Spill Ratio %	Statement Memory
ad_hoc	3	10	12	75	4164.59MB
admin_group	5	20	25	30	2082.30MB
default_group	10	40	30	75	3123.45MB
priority_low	3	10	10	75	--
					--
		80	77		

1. Click **EDIT** to open the Resource Group editor.
2. To delete a resource group, select the resource group, and click the minus sign that appears at the right.

   You cannot delete the default_group or admin_group resource groups. You cannot delete a resource group that is assigned to any Greenplum Database role.
3. To add a resource group, click **ADD RESOURCE GROUP** and enter a name for the resource group in the **Name** column. Resource group names must be unique and are case-sensitive.
4. Adjust the values of the **Concurrency**, **CPU %**, and **Memory %** resource group attributes.

   Concurrency
   The maximum number of concurrent transactions, including active and idle transactions, that are permitted in the resource group. **Concurrency** sets the CONCURRENCY attribute of the resource group. The total of the **Concurrency** columns cannot exceed the value of the Greenplum Database max_connections master server configuration parameter.

   CPU %
   The percentage of CPU resources available to this resource group. The percentage is the portion of the total CPU percentage allocated for all resource groups (reserved CPUs excluded), which is set with the gp_resource_group_cpu_limit server configuration parameter. **CPU %** sets the CPU_RATE_LIMIT attribute of the resource group.

   Memory %
   The percentage of memory resources available to this resource group. The percentage is the portion of the total memory allocated for all resource groups, which is set with the gp_resource_group_memory_limit Greenplum Database configuration parameter. Changing the **Memory %** value sets the MEMORY_LIMIT attribute of the resource group.

   Memory Spill Ratio % The memory usage threshold for memory-intensive transactions. When a transaction reaches this threshold, it spills to disk.

   Statement Memory The amount of memory allocated to a query. This column is recalculated as you adjust other resource group settings. If a query needs more memory, it is allocated from the resource group shared memory pool and the global shared memory pool, if available.

   The totals of the **CPU %** and **Memory %** columns must not exceed 100%. You should not allow the total of the **CPU %** column to exceed 90%, because this could cause resource-intensive queries to consume nearly all CPU, starving other Greenplum Database processes.

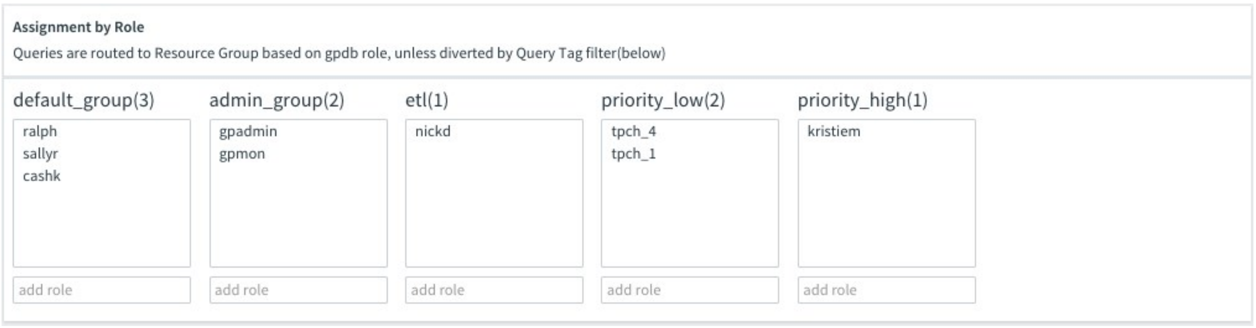
If the total of the **Memory %** column is less than 100%, the unreserved memory is part of the resource group shared global memory pool. See "Global Shared Memory" in Using Resource Groups in the *Greenplum Database Administrator Guide* for information about the global resource group shared memory pool.

5. Click **Apply** to save your changes or click **Cancel** to abandon your changes.

Assigning Roles to Resource Groups

Every Greenplum Database role is assigned to a single resource group in the `pg_roles` system table. Transactions executed by a role are managed by its assigned resource group, unless you create an assignment rule to override the default.

You can view the current resource group assignments for all roles and change a role's resource group by adding it to a different resource group.



Assignment by Role

Queries are routed to Resource Group based on gpdb role, unless diverted by Query Tag filter(below)

default_group(3)	admin_group(2)	etl(1)	priority_low(2)	priority_high(1)
ralph	gpadmin	nickd	tpch_4	kristiem
sallyr	gpmon		tpch_1	
cashk				
add role	add role	add role	add role	add role

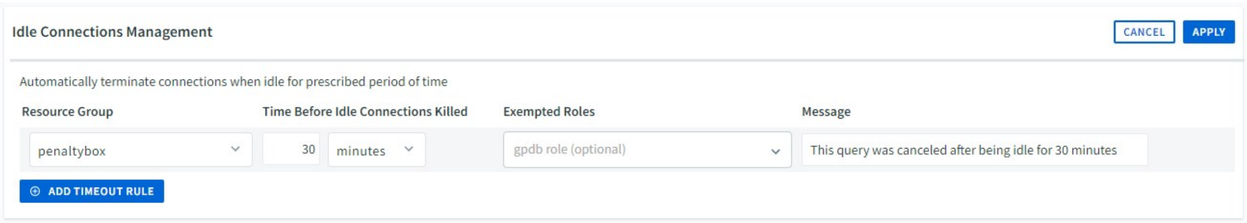
To move a role to a different resource group:

1. Enter all or part of the role name in the **add role** field beneath the new resource group.
2. Choose the role from the list that is displayed and press Enter.

The change is immediately applied to the Greenplum Database `pg_roles` system table.

Managing Idle Connections

Timeout rules set the amount of time a session can be idle before it is terminated. You create a timeout rule for each resource group. See About Timeout Rules for more information about timeout rules.



Idle Connections Management

Automatically terminate connections when idle for prescribed period of time

Resource Group	Time Before Idle Connections Killed		Exempted Roles	Message
penaltybox	30	minutes	gpdb role (optional)	This query was canceled after being idle for 30 minutes

1. To add a timeout rule, click **EDIT** in the **Idle Connections Management** section.
2. Click **ADD TIMEOUT RULE** and fill in the fields.

   Resource Group Choose a resource group from the list.

   Time before idle connections killed The amount of time before an idle session is terminated. Enter an integer value for the number of seconds or minutes and choose **minutes** or **seconds** from the list. A value of 600 seconds, for example, terminates the session after it has been idle for 10 minutes. See About Timeout Rules for warnings about setting timeouts for the `admin_group` resource group and resource groups with roles that require persistent sessions.

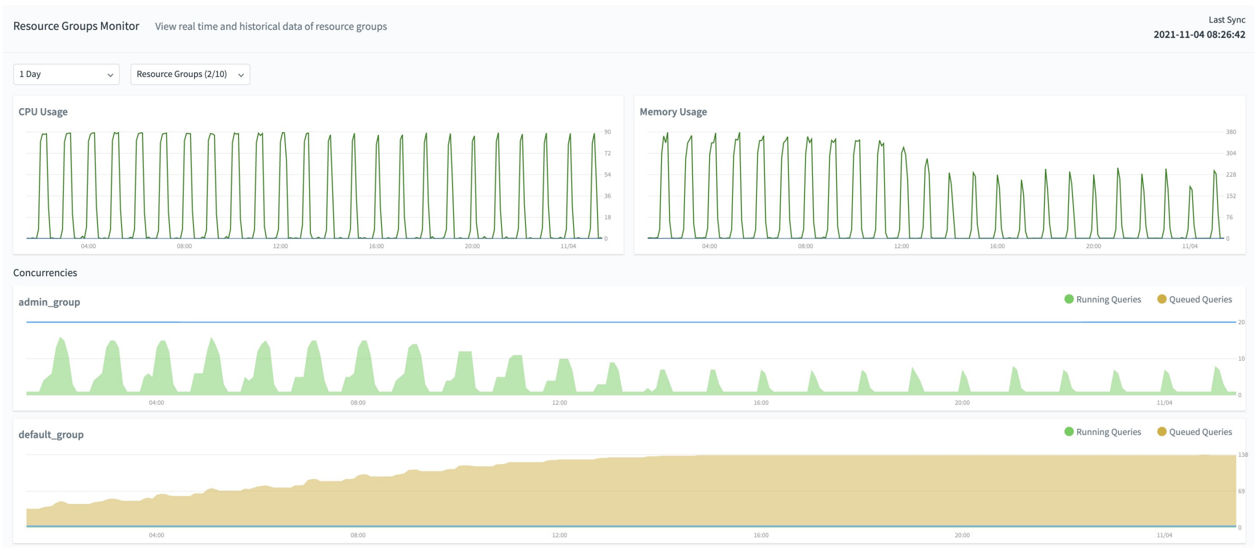
Exempted roles The rule is not applied to roles added to this field. Open the list to choose roles to exempt from the rule. Only roles for the selected resource group are included in the list. To remove a role from the field, click the **x** next to the role name.

Message Enter the message to log when a session is killed by this rule.

3. Click **APPLY** to save your changes.

Monitoring Resource Groups

You can determine whether your resource group settings are optimal by viewing resource utilization of resource groups, using the Command Center **Resource Groups> Monitor** view.



You may view the following information:

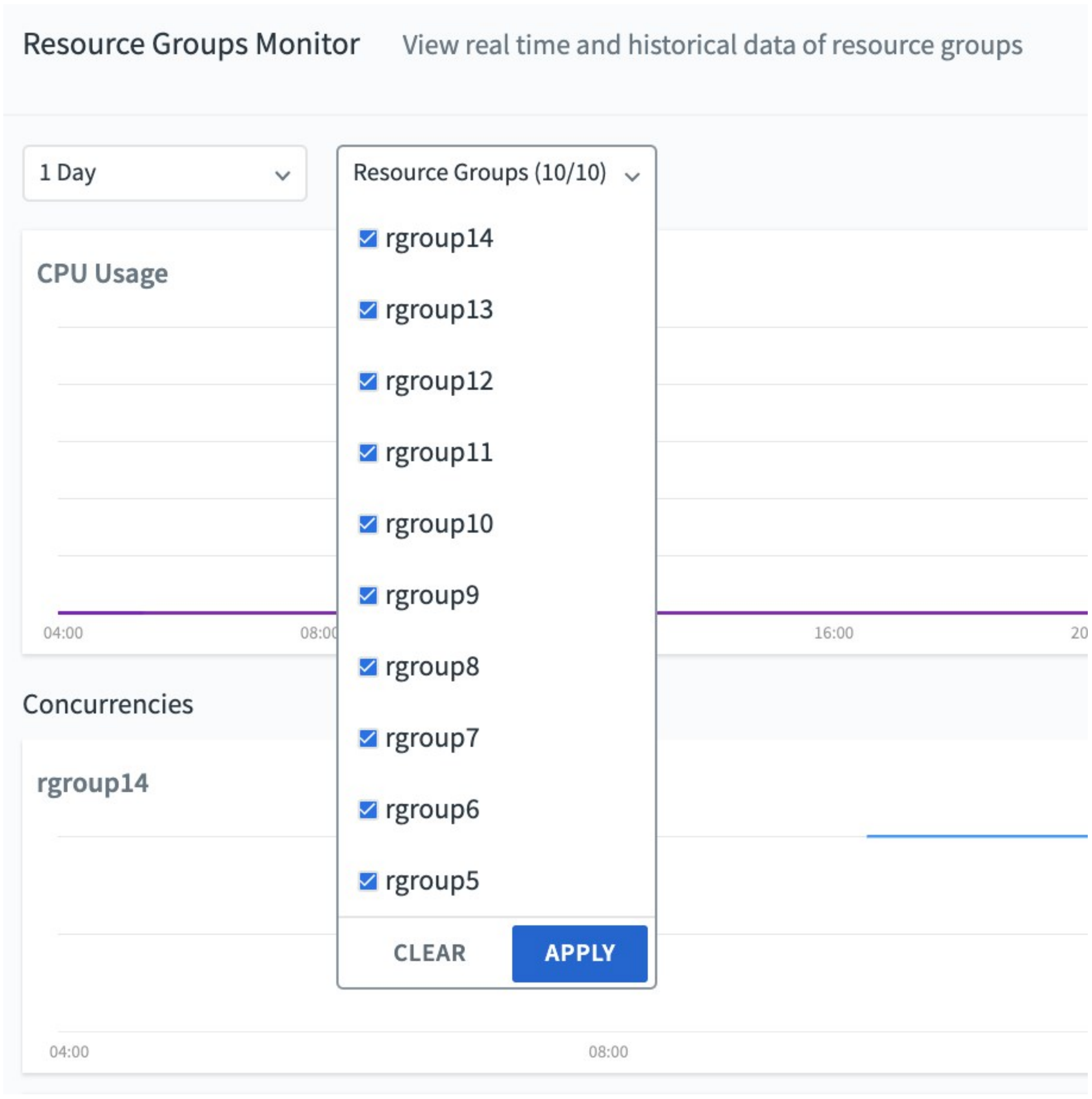
- CPU utilization of a single resource group
- CPU utilization of a selection of resource groups
- Memory utilization of a specific resource group
- Memory utilization of a selection of resource groups
- Concurrency status of selected resource groups

Note: To perform these tasks, you must have a permission level of Operator Basic or above.

About Resource Group Selection

Using the dropdown menus on the Resource Groups Monitor page, you may select:

- the time interval for the data you are interested in viewing
- the resource groups whose data to display; you may select up to 10 resource groups at a time. Once you have selected resource groups whose information you want to view, you may click **Apply** to save the selection or **Clear** to clear it.



Workload Management

Defining Workload Management Rules

You can define workload management rules for both queries and for idle sessions.

Defining Workload Management Rules for Queries

Query assignment rules enable you to configure which resource group a transaction uses before the transaction begins. Workload rules monitor additional conditions for queries, such as the amount of CPU time or disk I/O consumed, and enable you to cancel a running query or (for Greenplum version 6.8 or later) move a query to a different resource group after it has started. You configure both types of rule on the **Workload> Workload Mgmt** page using the same rule interface.

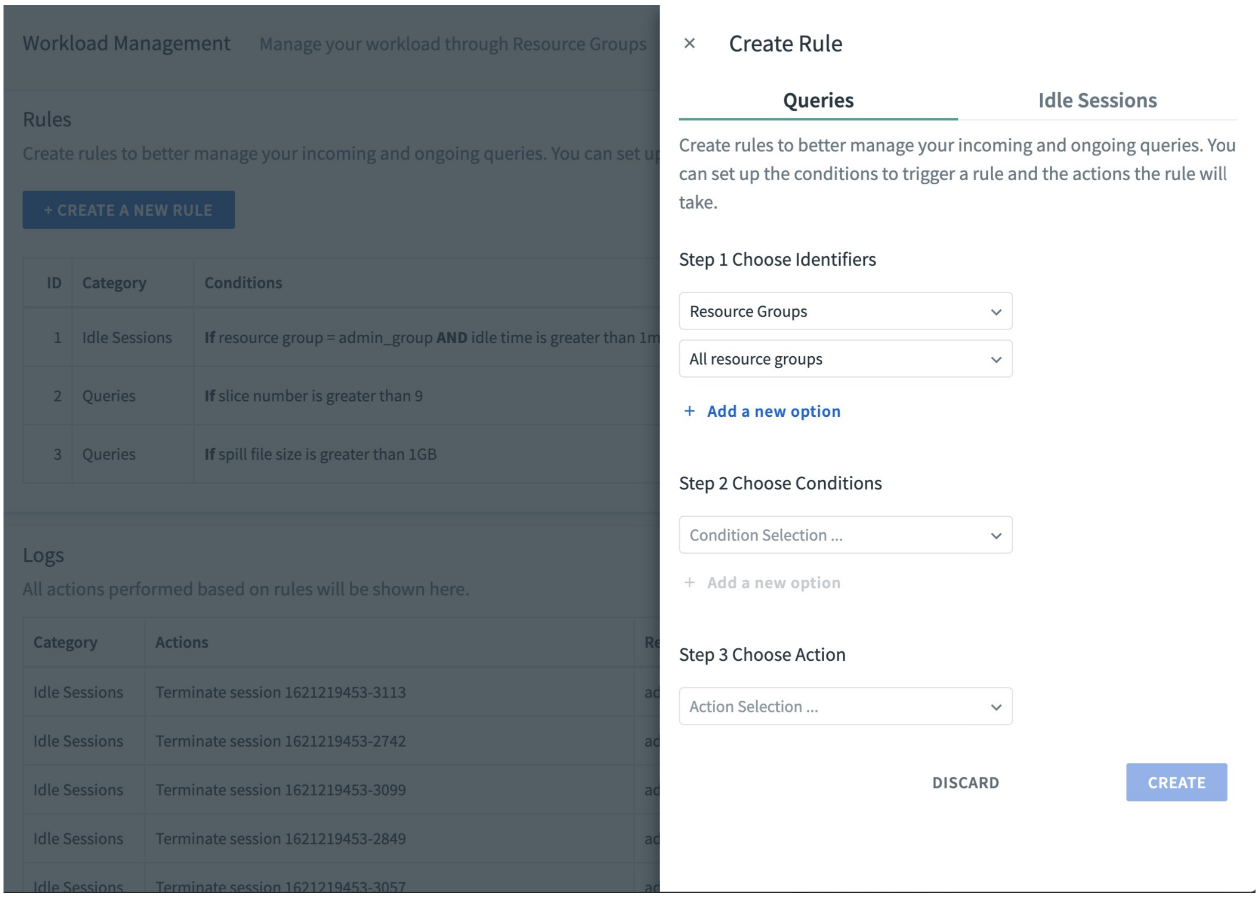
Transactions are first matched to a configured rule using any combination of user-defined query tags, the current role in the database session, or the resource group originally used for the transaction. When no rule matches, the transaction remains assigned to the role's default resource group. See About Assignment Rules for more information about assignment rules.

A rule that provides additional conditions acts as a query assignment rule, and determines the resource group where the statement should run.

A rule that provides additional conditions such as a maximum CPU time, Disk I/O, planner cost, or slices used, is a workload rule. With workload rules, Command Center monitors the query to evaluate those conditions. If all conditions are met, the workload rule action is performed.

A additional configuration parameter, `wlm_short_query_threshold`, is provided to ensure that only queries that run for the configured number of seconds are canceled or moved according to workload rules. This can help in preventing Command Center from applying workload rules to short-running queries. See Greenplum Command Center Parameters for information on changing these parameters.

If Command Center cannot successfully apply any rule's action (for example, if an attempt to move a query to another resource group fails due to resource availability), then the action is retried 2 times, after waiting a minimum of 15 seconds between attempts. You can configure the wait interval by using the `wlm_query_cooldown_time` configuration parameter.



Workload Management Manage your workload through Resource Groups

Rules

Create rules to better manage your incoming and ongoing queries. You can set u

+ CREATE A NEW RULE

ID	Category	Conditions
1	Idle Sessions	If resource group = admin_group **AND** idle time is greater than 1m
2	Queries	If slice number is greater than 9
3	Queries	If spill file size is greater than 1GB

Logs

All actions performed based on rules will be shown here.

Category	Actions	Re
Idle Sessions	Terminate session 1621219453-3113	ac
Idle Sessions	Terminate session 1621219453-2742	ac
Idle Sessions	Terminate session 1621219453-3099	ac
Idle Sessions	Terminate session 1621219453-2849	ac
Idle Sessions	Terminate session 1621219453-3057	ac

× Create Rule

Queries | Idle Sessions

Create rules to better manage your incoming and ongoing queries. You can set up the conditions to trigger a rule and the actions the rule will take.

Step 1 Choose Identifiers

Resource Groups

All resource groups

+ Add a new option

Step 2 Choose Conditions

Condition Selection ...

+ Add a new option

Step 3 Choose Action

Action Selection ...

DISCARD CREATE

1. To edit an existing query rule, click **EDIT** next to the rule definition to open the Workload Management Rules editor.
2. To delete a query rule, click **EDIT** next to an existing rule and then click **DELETE** in the Workload Management Rules editor.
3. To create a new query rule, click **CREATE A NEW RULE** and fill in the fields in each section.
    1. Choose one or more identifiers to match queries to the rule. By default, rules are applied to all resource groups. Click **Add a new option** to include additional identifiers to filter based on query tags and/or database role. Note that queries are

matched to a rule only if all of the configured identifiers match.

Resource Groups
Choose a resource group name from the list, or accept the default **All resource groups** to match the rule with any available resource group. Note that if you *remove* the **Resource Groups** identifier from your rule, Command Center still uses **All resource groups** as the default identifier for matching a query's resource groups.

Query Tags
Enter one or more query tags to match against the `gpcc.query_tags` parameter in the Greenplum Database session. A query tag is a user-defined *<name>=<value>* pair. Separate multiple query tags with semicolons. See Defining and Setting Query Tags for more information about query tags.

Role
Enter a role name in the field to match the rule with a role of the same name in the database session.

2. To create an query assignment rule, define no additional conditions for the rule (skip to the next step). To create a workload rule, choose one or more conditions that a matched, running query must meet before the selected action is performed. Click **Add a new option** to specify multiple conditions, all of which much apply before any action is taken.

   CPU Time The maximum amount of CPU time that the query consumes, specified in seconds, minutes, or hours.

   Planner Cost
   The query planning cost assigned to the query. Specify two separate values in the fields. The first field, **ORCA cost**, indicates the maximum Pivotal Query Optimizer (GPORCA) planning cost for the query. The second field, **Planner cost**, indicates the maximum Postgres Planner cost for the query, used only if the query falls back to using the Postgres Planner instead of GPORCA. Note that the two optimizers use different cost models, as well as different algorithms, to determine the cost of a query execution plan. See Query Profiling in the Greenplum Documentation for more information.

   Both planner cost values are required, because you cannot predict which planner will be chosen for a particular query.

   Query Running Time
   The maximum total running time for the query, specified in seconds, minutes, or hours.

   Slice Number
   The maximum number of slices allocated by the query planner to execute the query in parallel on different segments.

   Spill File Size
   The maximum value of a query's spill file size, specified in MB, GB, or TB.

   Total Disk I/O

The maximum total disk I/O that the query performs, specified in MB, GB, or TB. Note that Command Center can only make an approximate estimate of the actual I/O used by a query, so actual usage may ocassionaly exceed the configured value.

3. Choose an action to perform when a query is matched to the rule and all conditions apply. For query assignment rules (rules that have no additional conditions), chose **Move to another resource group**.

   Cancel query
   Cancel the query that matches all of the configured conditions. You must specify at least one condition (creating a workload rule) to trigger this action.

   Move to another resource group
   For a query assignment rule, this action defines the resource group where the query executes. If Command Center matches the rule based on the query tags and/or role identifiers for the rule, the configured source group is used to execute the query.

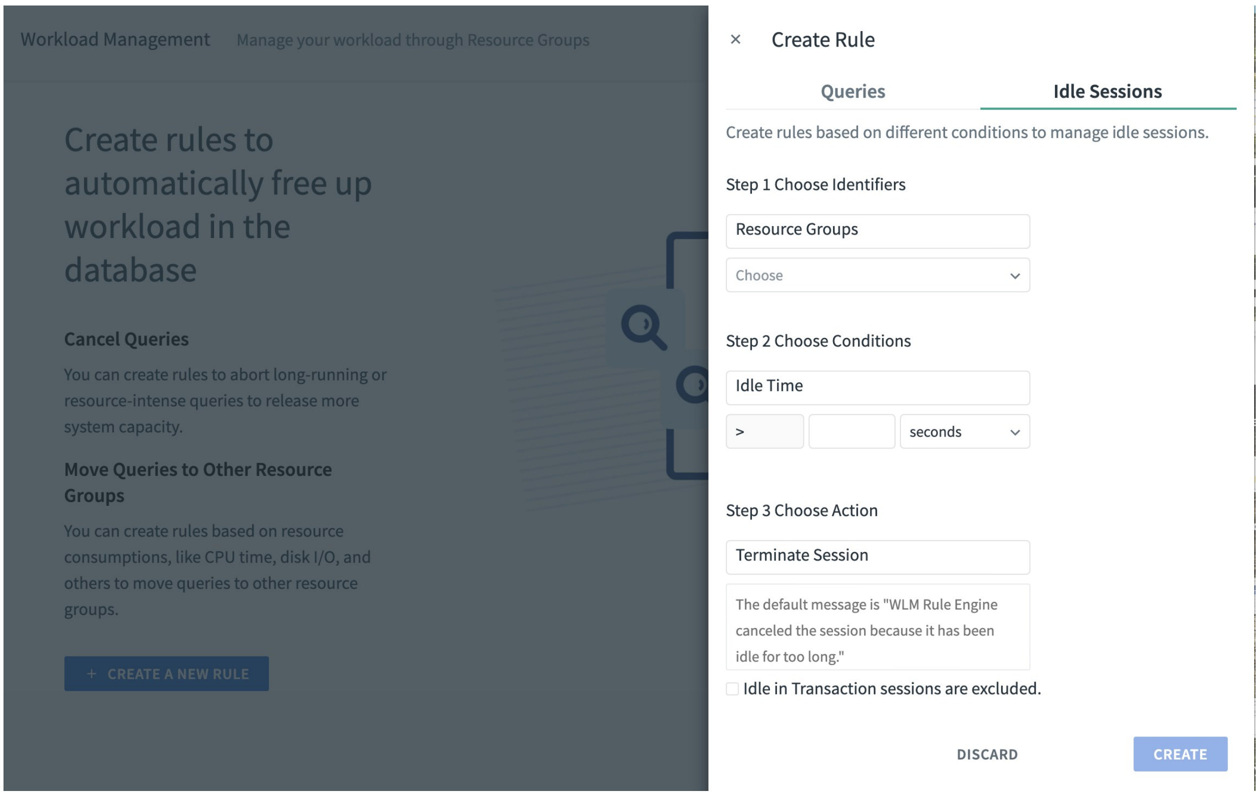
   For a workload rule that configures one or more conditions, this action moves the running query to the specified resource group if all of the configured conditions are met. Note that moving an active query as part of a workload rule is supported only when using Greenplum version 6.8 or later. You can create a workload rule with this action in earlier versions of Greenplum, but the rule will be created in the **Inactive** state.

4. Click **CREATE** to create the new rule, or **DISCARD** to dismiss the window without creating the rule.

4. Use the **Active/Inactive** toggle to make a rule active or inactive.

Defining Workload Management Rules for Idle Sessions

You can create a workload rule that terminates an idle session associated with a particular resource group once the session has been idle for a specified number of seconds, minutes, or hours. If all conditions are met, the workload rule action is performed.


Workload Management
Manage your workload through Resource Groups
Create rules to automatically free up workload in the database
Cancel Queries
You can create rules to abort long-running or resource-intense queries to release more system capacity.
Move Queries to Other Resource Groups
You can create rules based on resource consumptions, like CPU time, disk I/O, and others to move queries to other resource groups.
+ CREATE A NEW RULE
×
Create Rule
Queries
Idle Sessions
DISCARD
CREATE

Create rules based on different conditions to manage idle sessions.

Step 1 Choose Identifiers

Resource Groups

Choose

Step 2 Choose Conditions

Idle Time

> | | seconds

Step 3 Choose Action

Terminate Session

The default message is "WLM Rule Engine canceled the session because it has been idle for too long."

☐ Idle in Transaction sessions are excluded.

1. To edit an existing idle sessions rule, click **EDIT** next to the rule definition to open the Workload Management Rules editor.
2. To delete an idle sessions rule, click **EDIT** next to an existing rule and then click **DELETE** in the Workload Management Rules editor.
3. To create a new idle sessions rule, click **CREATE A NEW RULE** and fill in the fields in each section.
    1. Choose a resource group name from the list.
    2. Choose an idle time for the session -- in seconds, minutes, or hours.
    3. In the box under Terminate Session, enter a message to print to the log file and to the console where the session is running. If you do not enter anything, the default message "WLM Rule Engine canceled the session because it has been idle for too long" will be printed.
    4. To exclude Idle in Transaction sessions from the new rule, click the checkbox labeled "Idle in Transaction sessions are excluded."
    5. Click **CREATE** to create the new rule, or **DISCARD** to dismiss the window without creating the rule.

Defining and Setting Query Tags

A query tag is a user-defined *<name>=<value>* pair, set in the Greenplum Database `gpcc.query_tags` parameter in the Greenplum Database session. The `gpcc.query_tags` parameter is defined when the `gp_wlm` database extension is enabled in the postgres database. If you try to set query tags when the `gp_wlm` extension is not enabled, you get an unrecognized configuration parameter error. To see if the extension is enabled, run the following command.

```
$ psql postgres -c "\dx"
                   List of installed extensions
  Name   | Version | Schema |              Description
--------+---------+--------+-------------------------------------
 gp_wlm | 0.1     | gpcc   | Greenplum Workload Manager Extension
(1 row)
```

When you submit a transaction and the `gp_wlm` extension is enabled, Greenplum Database calls the `gp_wlm` extension to determine the resource group for the transaction. The extension evaluates the current role and query tags set in the session against the rules you have defined in Command Center. If there is a match, the extension returns the rule's resource group. If there is no match, Greenplum Database assigns the transaction to the role's default resource group.

The following command, executed in the Greenplum Database session, sets the `appName` and `appUser` query tags to "tableau" and "bi_sales", respectively.

```
=# SET gpcc.query_tags TO 'appName=tableau;appUser=bi_sales';
```

To match a rule, all tags in the rule's query tag field must be present in the `gpcc.query_tags` parameter in the database session. The order of the tags is not significant, and the `gpcc.query_tags` parameter can have a superset of the tags defined in the `queryTags` value.

When query tags are surrounded by single quotes and those tags have been set by the `PGOPTIONS` parameter, the single quotes become part of the query tags. In this case, Command Center ignores the single quotes, rather than processing them as part of the query tag name -- which would cause resource group assignment rules to fail.

If you set the `gpcc.query_tags` parameter inside of a transaction, you must commit the transaction before the new query tags are used to evaluate assignment rules.

You can set the value of the `gpcc.query_tags` parameter using the `SET` command, as in the example above, or as a connection parameter with database clients that support it, such as `psql`. Following are two examples that show how to specify query tags on the `psql` command line.

```
$ PGOPTIONS="-c gpcc.query_tags=appName=tableau;appUser=bi_sales" psql

$ psql postgresql://mdw:5432/postgres?options="-c gpcc.query_tags%3DappName%3Dtableau;
appUser%3Dbi_sales"
```

In the second example, it is necessary to code the equals signs as `%3D` to prevent `psql` from interpreting the query tags as command-line arguments.

Note: Setting query tags in a `psql -c` command does not work because resource group assignment occurs before the command specified with the `-c` option is executed. For example, this command will not have the desired effect.

```
psql -c "SET gpcc.query_tags TO 'appName=tableau;appUser=bi_sales'; SELECT * FROM sale
s_data;"
```

Monitoring Workload Rules

The **Logs** section displays a row of information for each instance where a query or idle session both matched a workload rule and triggers the workload rule conditions to perform an action. Note that query assignment rules are not currently logged in this section.

Columns of the log entry provide details about how the rule was applied:

Actions Summarizes the rule actions that were taken for the specified query ID.

Resource Group The resource group that matched the query or idle session rule.

Role The database role of the query session.

Rule The ID number of the rule. This ID matches an ID from the list of **Rules** on this page. Place your cursor over the ID to display a summary of the rule conditions that were in effect when the action was triggered.

Status Indicates whether the rule's action succeeded or failed.

Execution Time The time when the rule action was performed.

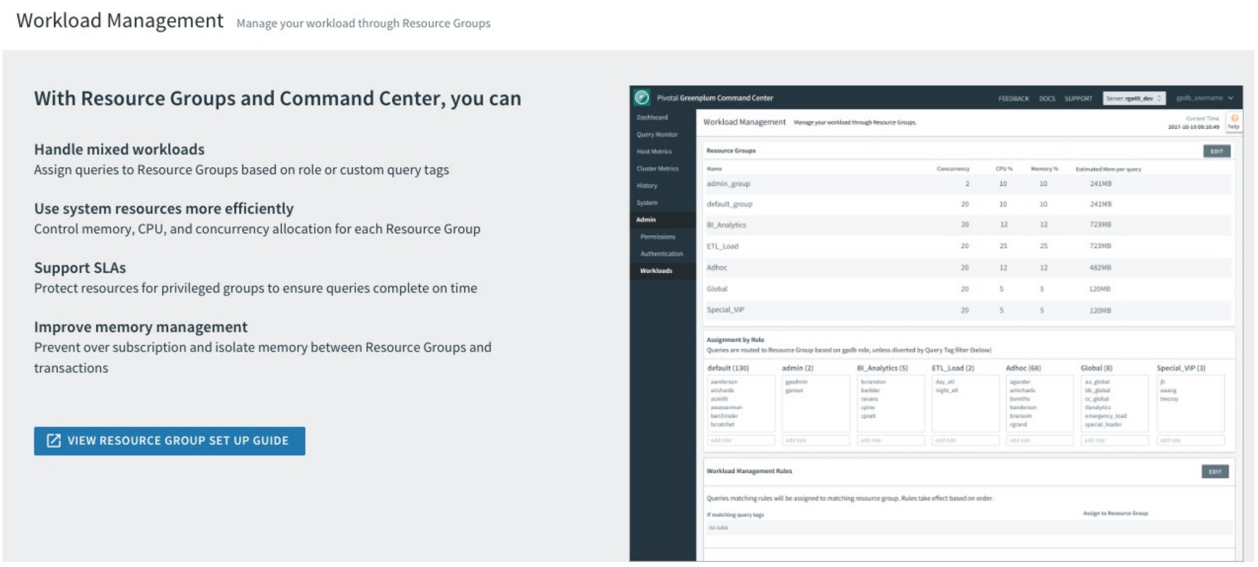
Importing Resource Queues to Resource Groups

Greenplum Command Center workload management works with resource groups, the new Greenplum Database resource management system. The default resource management system for Greenplum Database is resource queues. To use the Command Center workload management features, you must first enable resource groups in Greenplum Database.

Command Center can assist you in enabling resource groups and in importing existing resource queues to resource groups.

Step One: Enable Resource Groups in Greenplum Database

If your Greenplum Database system is still configured to use resource queues, the Command Center **Workload Mgmt** view describes the benefits of resource groups and workload management with Command Center and provides a link to the Greenplum Database documentation to help you enable resource groups.


Workload Management Manage your workload through Resource Groups

With Resource Groups and Command Center, you can

Handle mixed workloads
Assign queries to Resource Groups based on role or custom query tags

Use system resources more efficiently
Control memory, CPU, and concurrency allocation for each Resource Group

Support SLAs
Protect resources for privileged groups to ensure queries complete on time

Improve memory management
Prevent over subscription and isolate memory between Resource Groups and transactions

Workload Management Manage your workload through Resource Groups.

Current Time 2017-10-10 09:10:40

Resource Groups

Name	Concurrency	CPU %	Memory %	Estimated Mem per query
admin_group	2	10	10	241MB
default_group	20	10	10	241MB
BI_Analytics	20	12	12	723MB
ETL_Load	20	25	25	723MB
Adhoc	20	12	12	482MB
Global	20	5	5	120MB
Special_VIP	20	5	5	120MB

Assignment by Role
Queries are routed to Resource Group based on gpdb role, unless diverted by Query Tag filter (below)

default (130)	admin (2)	BI_Analytics (5)	ETL_Load (2)	Adhoc (68)	Global (8)	Special_VIP (3)
aanderson	gpadmin	bcranston	day_etl	agander	aa_global	jli
archards	gpmon	bwilder	night_etl	amichaels	bb_global	swang
asmith		cevans		bsmiths	cc_global	tmccoy
awasserman		cpine		banderson	danalytics	
barchinder		cpratt		bransom	emergency_load	
bcratchet				cgrand	special_loader	
add role	add role	add role	add role	add role	add role	add role

Workload Management Rules

Queries matching rules will be assigned to matching resource group. Rules take effect based on order.

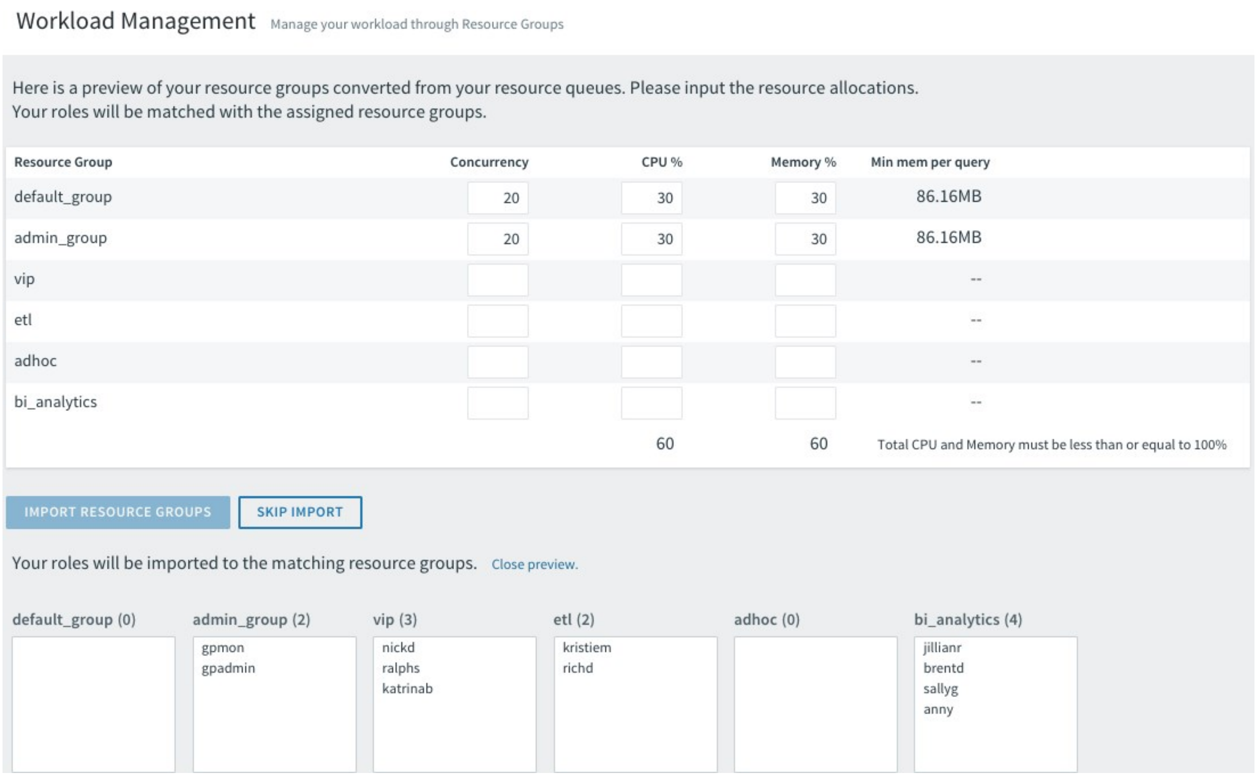
If matching query tags	Assign to Resource Group
no rules	

Click **VIEW RESOURCE GROUP SET UP GUIDE** for instructions to enable resource groups in your Greenplum Database system.

Step Two: Preview and Configure Resource Group Imports

After you have enabled resource groups and restarted Greenplum Database, restart Command Center (`gpcc start`), log in, and choose **Workload Mgmt**.

The workload management view now displays a preview of resource groups converted from your existing resource queues. You can use this one-time view to convert your Greenplum Database resource queues to resource groups.


Workload Management Manage your workload through Resource Groups

Here is a preview of your resource groups converted from your resource queues. Please input the resource allocations.
Your roles will be matched with the assigned resource groups.

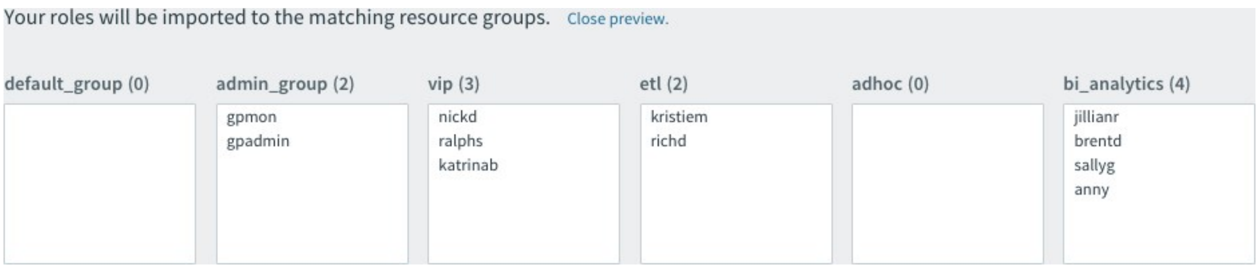
Resource Group	Concurrency	CPU %	Memory %	Min mem per query
default_group	20	30	30	86.16MB
admin_group	20	30	30	86.16MB
vip				--
etl				--
adhoc				--
bi_analytics				--
		60	60	Total CPU and Memory must be less than or equal to 100%

Your roles will be imported to the matching resource groups. Close preview.

default_group (0)	admin_group (2)	vip (3)	etl (2)	adhoc (0)	bi_analytics (4)
	gpmon	nickd	kristiem		jillianr
	gpadmin	ralphs	richd		brentd
		katrinab			sallyg
					anny

The resource group list includes the required `admin_group` and `default_group` resource groups, and a row for each of your existing resource queues.

Roles are assigned to the resource group matching the resource queue to which they are assigned. Click the **Preview roles** link to see the role assignments.


Your roles will be imported to the matching resource groups. Close preview.

default_group (0)	admin_group (2)	vip (3)	etl (2)	adhoc (0)	bi_analytics (4)
	gpmon	nickd	kristiem		jillianr
	gpadmin	ralphs	richd		brentd
		katrinab			sallyg
					anny

If you want to set up resource groups later, you can click **SKIP IMPORT**. Only the `default_group` and `admin_group` resource groups are created. Roles with the superuser attribute are assigned to the `admin_group` resource group; roles without superuser privilege are assigned to the `default_group` resource group.

If you want Command Center to import resource queues to resource groups, you must complete the resource allocation fields for all resource groups.

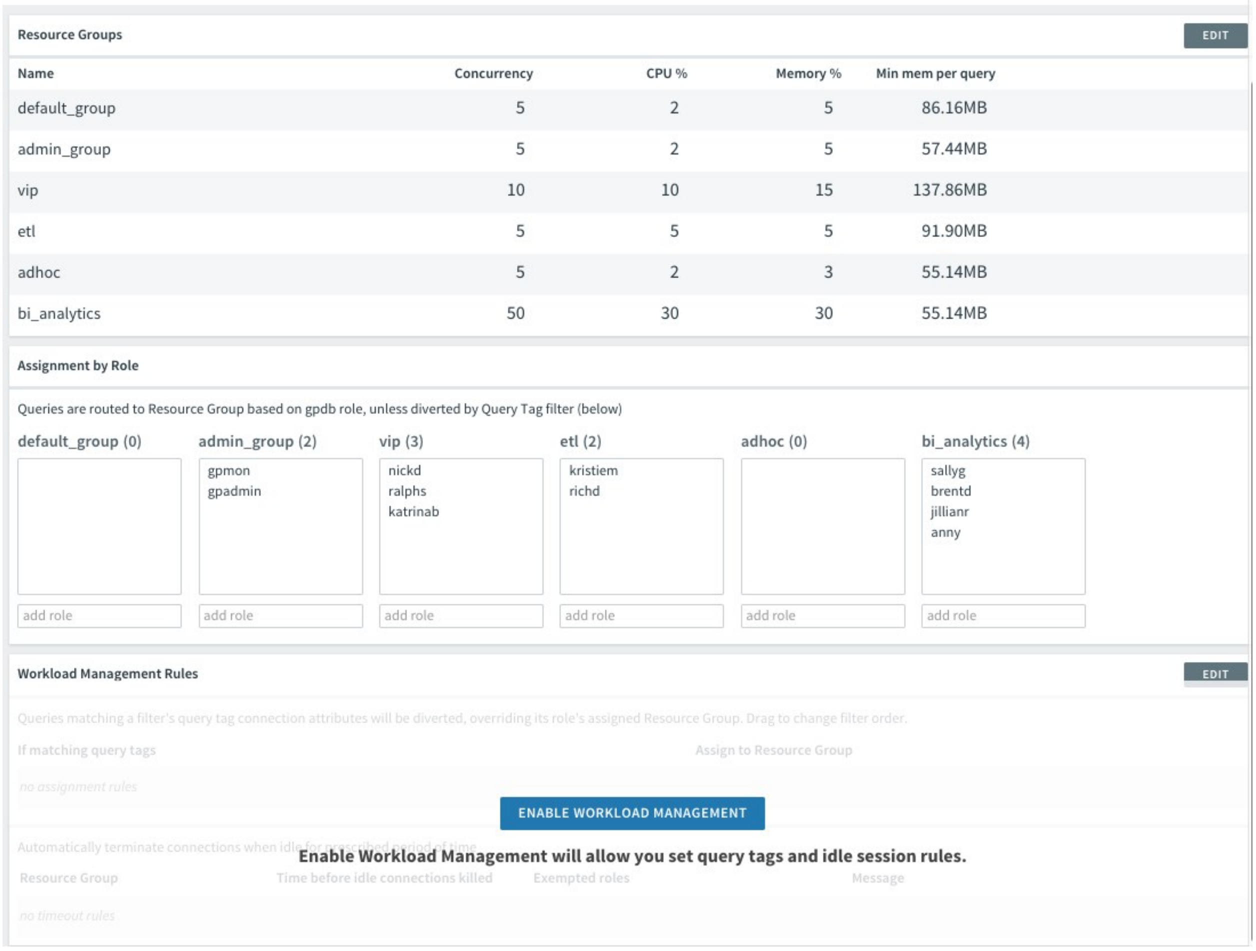
Set the **Concurrency**, **CPU %**, and **Memory %** resource group attributes to allocate Greenplum Database resources to the resource queues. The **Concurrency** fields must each contain a positive integer. The **CPU %** and **Memory %** fields must each contain positive integers between 1 and 99 and the totals for the **CPU %** and **Memory %** columns must not exceed 100%. See Defining Resource Groups and Resource Attributes for help determining the values to enter.

The **IMPORT RESOURCE GROUPS** button is disabled until you have entered valid values in the allocation fields for every resource group.

When you are ready to import the resource groups, click **IMPORT RESOURCE GROUPS** to create the resource groups.

Step Three: Enable Command Center Workload Management

After you import (or skip importing) resource queues to resource groups, you can enable Command Center workload management.



Name	Concurrency	CPU %	Memory %	Min mem per query
default_group	5	2	5	86.16MB
admin_group	5	2	5	57.44MB
vip	10	10	15	137.86MB
etl	5	5	5	91.90MB
adhoc	5	2	3	55.14MB
bi_analytics	50	30	30	55.14MB

Queries are routed to Resource Group based on gpdb role, unless diverted by Query Tag filter (below)

default_group (0)	admin_group (2)	vip (3)	etl (2)	adhoc (0)	bi_analytics (4)
	gpmon	nickd	kristiem		sallyg
	gpadmin	ralphs	richd		brentd
		katrinab			jillianr
					anny
add role	add role	add role	add role	add role	add role

Queries matching a filter's query tag connection attributes will be diverted, overriding its role's assigned Resource Group. Drag to change filter order.

If matching query tags	Assign to Resource Group
no assignment rules	

Enable Workload Management will allow you set query tags and idle session rules.

Automatically terminate connections when idle for prescribed period of time

Resource Group	Time before idle connections killed	Exempted roles	Message
no timeout rules			

Click **ENABLE WORKLOAD MANAGEMENT** to enable workload management in Greenplum Command Center. Greenplum Command Center creates the `gp_wlm` extension, the `gpcc.workload_config` table, and the associated user-defined functions in Greenplum Database.

You are now able to use the Command Center Workload Management interface to add, remove, and configure resource groups; change role assignments; and define workload management rules.

See Workload Management for help using the Command Center Workload Management view.

Troubleshooting Enabling Resource Groups

If you experience problems enabling resource groups in Greenplum Command Center, review the following list to ensure prerequisites are met and all of the dependencies are properly configured.

- Red Hat 6.x and 7.x and CentOS 6.x and 7.x are currently supported.
- You must be running Greenplum Database version 5.7.0 or later.
- Configure the Linux cgroups kernel feature on your hosts by following the instructions at "Prerequisite" in Using Resource Groups.
- Make sure the `/etc/cgconfig.d/gpdb.conf` file contains the objects perm, cpu, and cpuacct. If the document is incorrect and the `gp_resource_manager` configuration parameter is set to `"group"`, Greenplum Database can hang at startup.

```
group gpdb {
  perm {
    task {
      uid = gpadmin;
      gid = gpadmin;
    }
    admin {
      uid = gpadmin;
      gid = gpadmin;
    }
  }
  cpu {
  }
  cpuacct {
  }
}
```

- On Red Hat 7, make sure you run `cgconfigparser -L /etc/cgconfig.d` to parse changes to the `/etc/cgconfig.d/gpdb.conf` file. This command must also be set up to run at boot time.
- Set the Greenplum Database `gp_resource_manager` server configuration parameter to `"group"` and restart Greenplum Database.

```
$ gpconfig -c gp_resource_manager -v "group"
$ gpstop -ar
```

Verify by showing the value of the parameter:

```
$ gpconfig -s gp_resource_manager
Values on all segments are consistent
GUC          : gp_resource_manager
Master  value: group
Segment value: group
```

- After installing a Pivotal Greenplum Database distribution, the `shared_preload_libraries` configuration parameter contains the metrics collector and workload manager extension shared libraries. Make sure these libraries are still present:

```
$ gpconfig -s shared_preload_libraries
Values on all segments are consistent
GUC          : shared_preload_libraries
```

```
Master  value: $libdir/metrics_collector,$libdir/gp_wlm
Segment value: $libdir/metrics_collector,$libdir/gp_wlm
```

Check that the shared libraries exist at `$GPHOME/lib/postgresql/metrics_collector.so` and `$GPHOME/lib/postgresql/gp_wlm.so`. If the libraries do not exist, make sure you have installed the Pivotal Greenplum Database distribution. These extensions are not available in the Greenplum Database Open Source version.

If the shared library files exist in the `$GPHOME/lib/postgresql` directory, but not in the `shared_preload_libraries` parameter, add them with the `gpconfig` command:

```
$ gpconfig -c shared_preload_libraries -v '\$libdir/metrics_collector,\$libdir/
gp_wlm'
```

Note that adding the libraries to the `shared_preload_libraries` parameter does not enable the metrics_collector or gp_wlm extensions, but is a prerequisite for enabling them.

- The gpmon user must be able to connect to databases from the Command Center host. Make sure to add a `host` entry like the following in the `$MASTER_DATA_DIRECTORY/pg_hba.conf` file.

```
host   all   gpmon   <IP_of_host>/32   md5
```

- Check whether the `gp_wlm` extension is installed. The extension is added when you click **Enable Workloads** in the Greenplum Command Center **Workload Mgmt** view.

```
$ psql postgres
\dx
postgres=# \dx
                   List of installed extensions
  Name   | Version | Schema |             Description
--------+---------+--------+--------------------------------------
 gp_wlm | 0.1     | gpcc   | Greenplum Workload Manager Extension
```

- Make sure the `gpcc.workload_config` table and functions are present in the postgres database:

```
$ psql postgres
postgres=# \d gpcc.*
  Table "gpcc.workload_config"
  Column  |  Type   | Modifiers
----------+---------+-----------
 dist_col | integer |
 config   | json    |
Distributed by: (dist_col)
postgres=# \df gpcc.*
    List of functions
 Schema |        Name         | Result data type | Argument data types  |  Type
--------+---------------------+------------------+----------------------+------
-
 gpcc   | get_workload_config | json             |                      | norma
l
 gpcc   | set_workload_config | boolean          | wlm_json_config json | norma
l
 (2 rows)
```

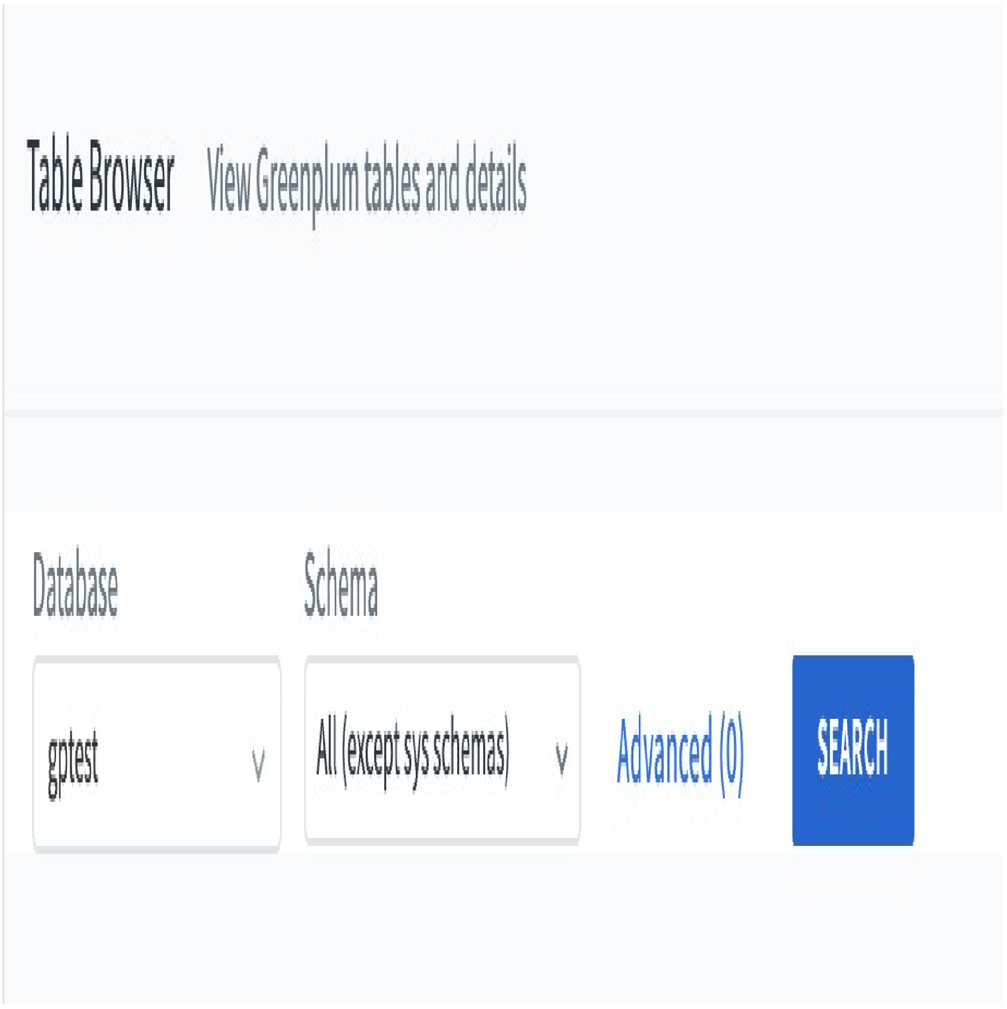
If the `gpcc.workload_config` table or the functions are not present, dropping and recreating the gp_wlm extension will create them. Note that any assignment rules saved in the `gpcc.workload_config` table will have to be recreated in Command Center.

```
$ psql postgres
postgres=# DROP EXTENSION gp_wlm;
DROP EXTENSION
postgres=# CREATE EXTENSION gp_wlm;
CREATE EXTENSION
```

Greenplum Database Table Browser

Command Center users with Basic permission can view details about the tables in a Greenplum database.

1. Select **Table Browser** to view the Command Center Table Browser.
2. On the **Table Browser** view, choose a database and then use the dropdown to select a schema. All filter criteria except **Database** are optional.



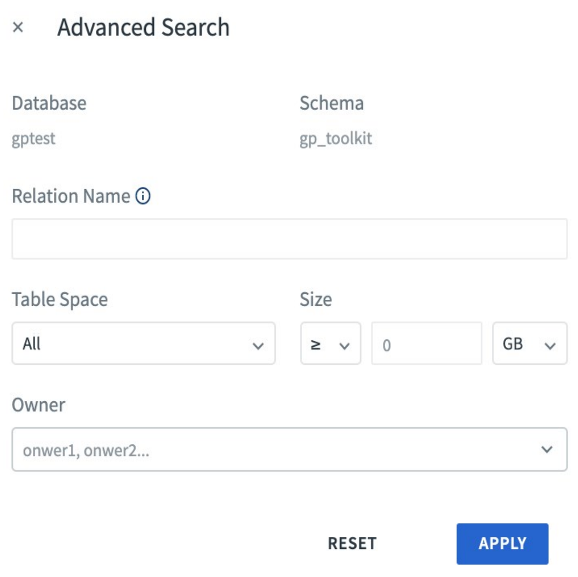
Database

Select the name of the database to browse.

Schema

Enter the name of a schema to select only the tables that belong to that schema. The default is "All (except sys schemas)" where system schemas include: `pg_catalog`, `pg_toast`, `pg_bitmapindex`, `pg_aoseg`, `information_schema`, or `gp_toolkit`.

3. Click **Advanced** to display more search criteria for the database and the schema you have



Advanced Search

Database	Schema
gptest	gp_toolkit

Relation Name

Table Space	Size		
All	≥	0	GB

Owner

onwer1, onwer2...

selected.

Relation Name

Enter the table name for your search. You may enter a `%` wildcard to match any characters.

Table Space

Select the table space from the dropdown options.

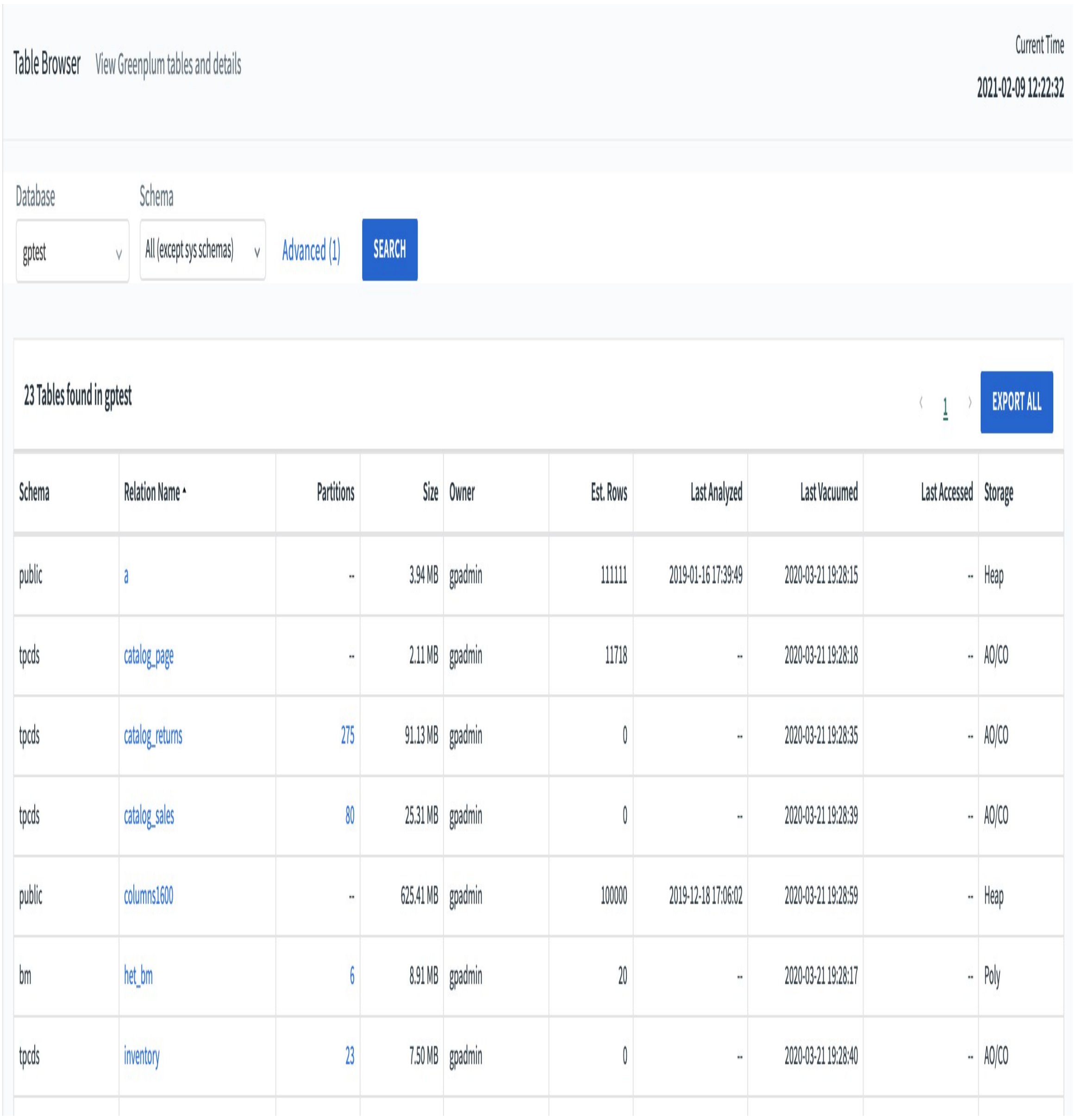
Table Size

To select tables by size, select a ≤ (less than or equal) or ≥ (greater than or equal) operator, and enter a size number with a size unit (KB, MB, GB, TB).

Owner

Enter a role name to select only tables owned by the role.

4. Click **SEARCH** to display tables matching the criteria you have entered.



23 Tables found in gptest

Schema	Relation Name	Partitions	Size	Owner	Est. Rows	Last Analyzed	Last Vacuumed	Last Accessed	Storage
public	a	--	3.94 MB	gpadmin	111111	2019-01-16 17:39:49	2020-03-21 19:28:15	--	Heap
tpcds	catalog_page	--	2.11 MB	gpadmin	11718	--	2020-03-21 19:28:18	--	AO/CO
tpcds	catalog_returns	275	91.13 MB	gpadmin	0	--	2020-03-21 19:28:35	--	AO/CO
tpcds	catalog_sales	80	25.31 MB	gpadmin	0	--	2020-03-21 19:28:39	--	AO/CO
public	columns1600	--	625.41 MB	gpadmin	100000	2019-12-18 17:06:02	2020-03-21 19:28:59	--	Heap
bm	het_bm	6	8.91 MB	gpadmin	20	--	2020-03-21 19:28:17	--	Poly
tpcds	inventory	23	7.50 MB	gpadmin	0	--	2020-03-21 19:28:40	--	AO/CO

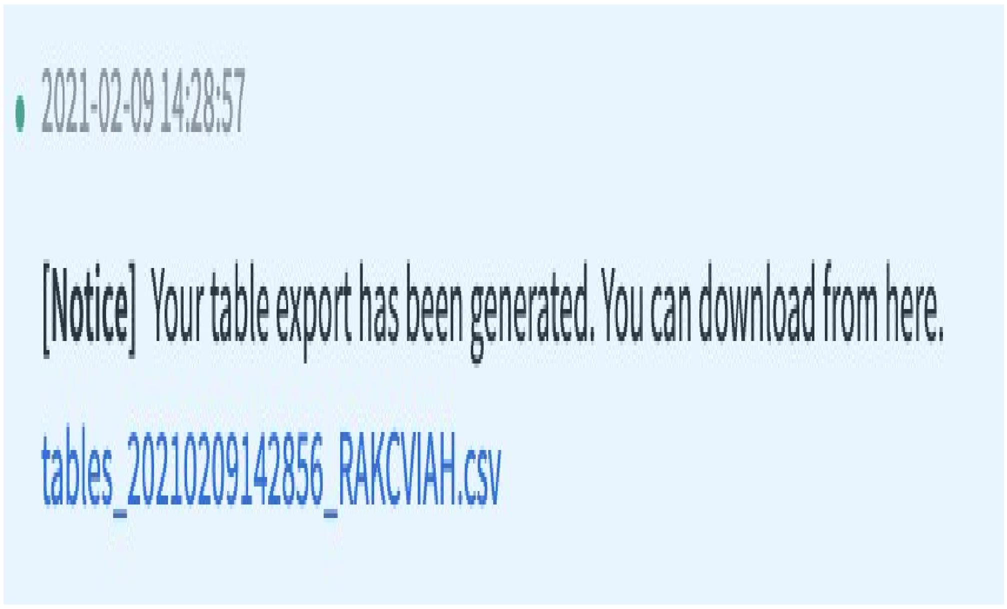
Table Browser statistics are updated every 300 seconds by default. Activity that occurs between updates is not displayed in the Table Browser view until Command Center updates the table statistics at the end of the sampling interval. This means that, for example, it can take up to five minutes for a new table to appear in the Table Browser. The times displayed for `Last Analyzed`, `Last Vacuumed`, and `Last Accessed` are the time that Command Center updates the table statistics after the event has been detected, not the actual time of the event. The actual event could have occurred at any point in the previous five minutes (or in the time period specified by `stats_check_interval`, if you have modified the parameter).

You can change the sampling interval by setting the `stats_check_interval` parameter in the `$GPCC_HOME/conf/app.conf` configuration file. Restart Command Center after updating the configuration file. See Command Center Console Parameters for more information about the `app.conf` file.

Note: using the `gppersistentrebuild` utility might refresh the `Last Accessed` table data even if the table has not been accessed by a query.

5. Click the **EXPORT ALL** button, at the top right of the table results, to save the search results in a local file. The file is created in the background and when the export is ready, a

notification is entered at the Notifications center, with a link to a CSV file, similar to:



2021-02-09 14:28:57

[Notice] Your table export has been generated. You can download from here.

tables_20210209142856_RAKCVIAH.csv

If the export operation is greater than 1 million rows, an alert is raised.

You can change the sampling interval by setting the `stats_check_interval` property in the `$MASTER_DATA_DIRECTORY/gpmetrics/gpcc.conf` configuration file. Restart Command Center after updating the configuration file. See the gpmetrics Configuration File Reference for more information about the `gpcc.conf` file.

Table List

The **Table Browser** table list displays these columns:

Schema

Name of the schema the table belongs to.

Relation Name

Name of the table.

Partitions

Total number of leaf partitions. If the table has no partitions, this column contains `0`.

Size

Combined size of the table's files on all segments.

Owner

Database role name of the table owner.

Est. Rows

Estimated number of rows in the table. For external tables this column contains `0`.

Last Analyzed

Time the table was last analyzed. The actual event occurred at some point during the preceding sampling interval.

Last Vacuumed

Time the table was last vacuumed. (Does not include `AUTOVACUUM`.) The actual event occurred at some point during the preceding sampling interval.

Last Accessed

Time the table was last accessed. (`SELECT`, `INSERT`, `UPDATE`, `DELETE`, or `TRUNCATE` operations.) The actual event occurred at some point during the preceding sampling interval.

Storage

Storage type for the table: Heap, AO, AO/CO, External, or Poly (Polymorphic - used for tables that combine different storage types in a single table).

Table Details

Click the name of a table in the **Relation Name** column to view details for that table.

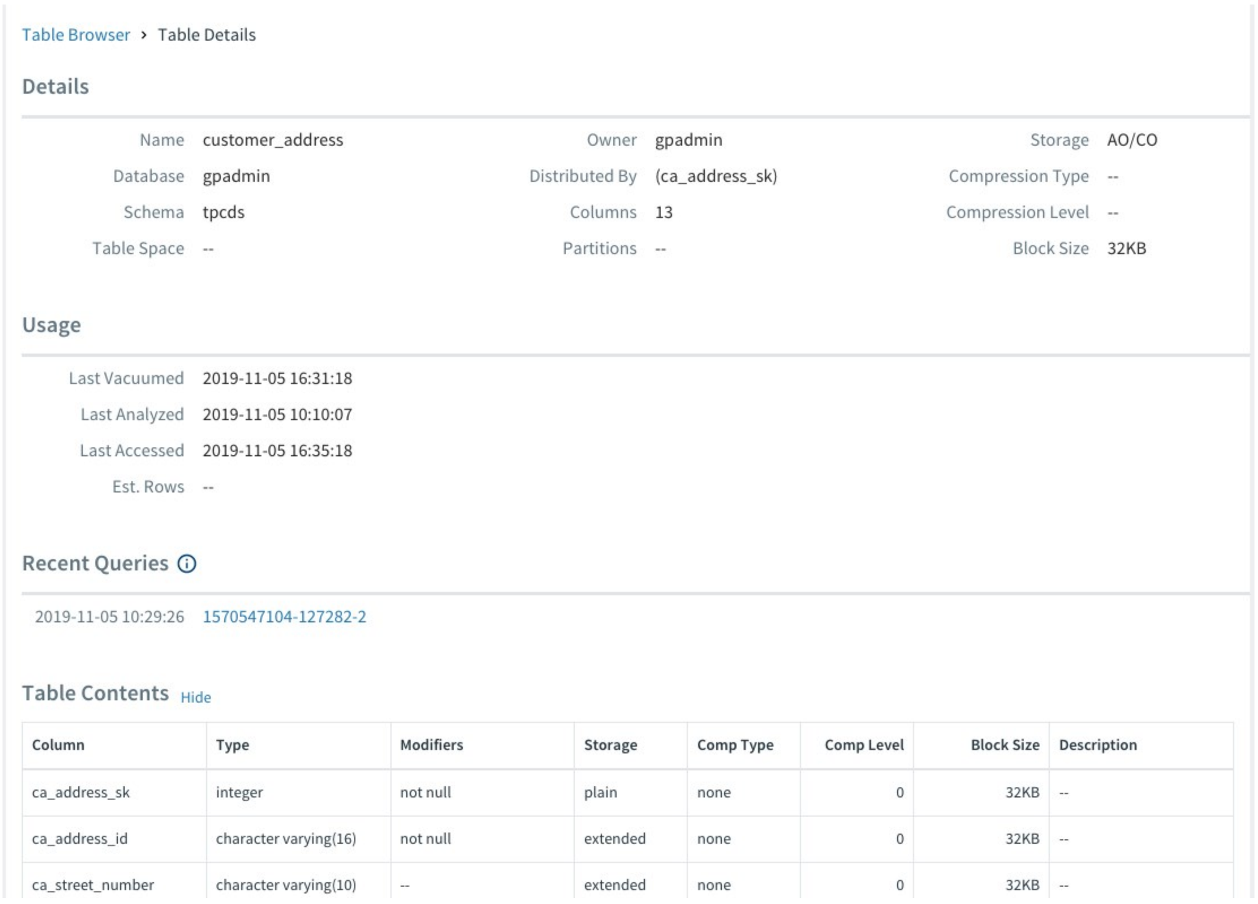


Table Browser › Table Details

Details

Name	customer_address	Owner	gpadmin	Storage	AO/CO
Database	gpadmin	Distributed By	(ca_address_sk)	Compression Type	--
Schema	tpcds	Columns	13	Compression Level	--
Table Space	--	Partitions	--	Block Size	32KB

Usage

Last Vacuumed	2019-11-05 16:31:18
Last Analyzed	2019-11-05 10:10:07
Last Accessed	2019-11-05 16:35:18
Est. Rows	--

Recent Queries

2019-11-05 10:29:26	1570547104-127282-2

Table Contents Hide

Column	Type	Modifiers	Storage	Comp Type	Comp Level	Block Size	Description
ca_address_sk	integer	not null	plain	none	0	32KB	--
ca_address_id	character varying(16)	not null	extended	none	0	32KB	--
ca_street_number	character varying(10)	--	extended	none	0	32KB	--

The **Table Detail** view displays this information:

Name

Name of the table.

Database

Name of the database.

Schema

Name of the schema the table belongs to.

Table Space

Name of the tablespace where the table's files reside.

Owner

Database role name of the table owner.

Distributed By

Distribution policy for the table. This can be a list of the distribution key columns `(key1, key2, key3, ...)` or `Randomly`.

Columns

Number of user columns in the table. System columns are not included in the count.

Partitions

Number of partitions the table has, including middle-level and leaf partitions. Click `Show` to list up to 100 partitions. The list includes the partition's name, table size, partition criteria, storage type, and compression parameters. Middle-level partitions are displayed with a different background color.

Storage

Storage type for the table: Heap, AO, AO/CO, External, or Poly (Polymorphic - used for tables that combine different storage types in a single table).

Compression Type

For append-optimized tables, the type of compression used for the table: `ZLIB`, `ZSTD`, `QUICKLZ`, `RLE_TYPE`.

Compression Level

For append-optimized tables, the level of compression. For `ZLIB`, the level is 1 (fastest compression) to 9 (highest compression ratio). For `ZSTD`, the level is 1 (fastest compression) to 19 (highest compression ratio). For `QUICKLZ`, the level is 1. For `RLE_TYPE` the compression level is 1 (fastest compression) to 4 (highest compression ratio).

Blocksize

Size, in bytes, for each block in the table: 8k, 16K, 1M, or 2M.

Usage Section

Last Analyzed

Time the table was last analyzed.

Last Vacuumed

Time the table was last vacuumed. (Does not include `AUTOVACUUM`.)

Last Accessed

Time the table was last accessed. (`SELECT`, `INSERT`, `UPDATE`, `DELETE`, or `TRUNCATE` operations.)

Est. Rows

Estimated number of rows in the table.

Recent Queries Section

This section lists the five most recent queries in the query history to access the table. The list is limited to queries that executed in the last 30 days and that ran for ten seconds or more.

Click a Query ID to view the Query History Details for the query.

Table Contents Tab

A list of the table's columns, types, constraints, and compression types.

Column

Name of the column.

Type

Data type of the column.

Modifiers

Constraints defined for the column.

Storage

Storage type for the column. `Plain` is for fixed-length values such as `integer` and is inline, uncompressed. `Main` is for inline, compressible data. `Toast` is for data that exceeds the size of a data block and is stored in chunks in a separate table associated with the user table. `Extended` is for external, compressed data. `Extended` is the default for most data types that support non-plain storage.

Compression Type

The type of compression used for the column: `none`, `ZLIB`, `QUICKLZ`, `RLE_TYPE`.

Compression Level

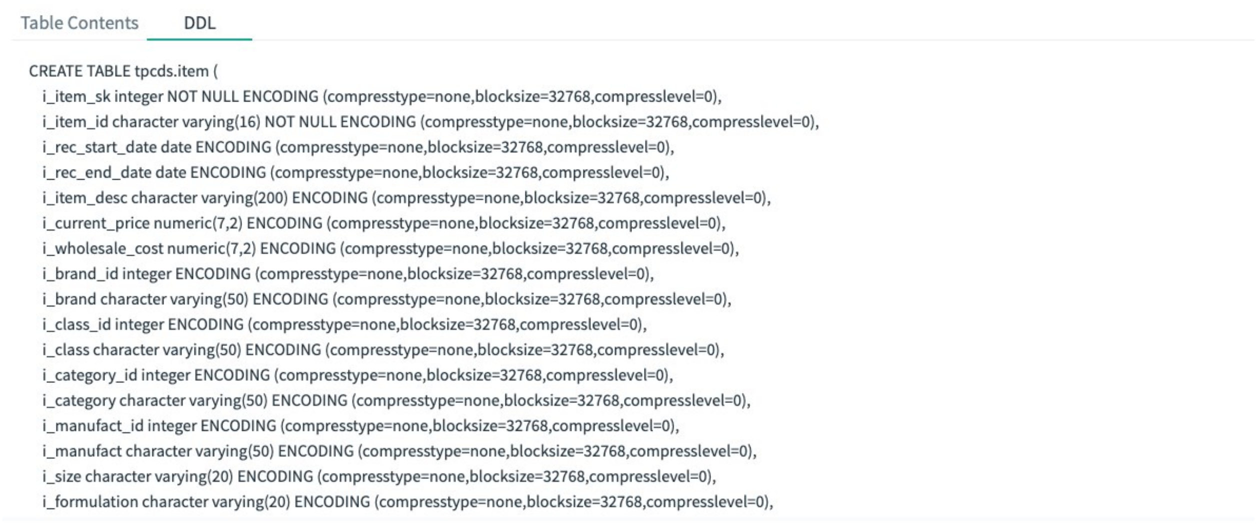
The level of compression. For `ZLIB`, the level is 1 (fastest compression) to 9 (highest compression ratio). For `QUICKLZ`, the level is 1. For `RLE_TYPE` the compression level is 1 (fastest compression) to 4 (highest compression ratio).

Blocksize

Size, in bytes, for each storage block.

DDL Tab

Click the DDL tab to display the `CREATE TABLE` statement for the table.



```
CREATE TABLE tpcds.item (
  i_item_sk integer NOT NULL ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_item_id character varying(16) NOT NULL ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_rec_start_date date ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_rec_end_date date ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_item_desc character varying(200) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_current_price numeric(7,2) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_wholesale_cost numeric(7,2) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_brand_id integer ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_brand character varying(50) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_class_id integer ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_class character varying(50) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_category_id integer ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_category character varying(50) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_manufact_id integer ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_manufact character varying(50) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_size character varying(20) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
  i_formulation character varying(20) ENCODING (compresstype=none,blocksize=32768,compresslevel=0),
```

Query Monitor Help Topics

- CPU
- CPU Skew
- Locks
- Query Optimization
- Memory
- Spill Files

CPU

The **CPU percent** metric is the average current CPU percentage for all backend processes executing this query. The percentages for all processes running a query on each segment are averaged, and then the average of all those values is calculated to render this metric.

You can manage the percentage of CPU that queries can consume by creating workloads and specifying the maximum percent of CPU each workload can consume. That percentage is further divided among the segments running on each host and then among the concurrent queries the workload can execute.

CPU allocated to idle workloads is reallocated to active queries and reclaimed when the idle workload becomes active again. This means that the **CPU percent** value for a query can exceed limits defined for workloads and can increase and decrease as other queries start or finish.

Memory and disk I/O resources are more likely causes for degraded query performance than lack of CPU cycles. The ways to reduce CPU contention mirror the solutions for insufficient memory:

- Reduce concurrency of workloads to make more CPU available to each query.
- Reduce the number of workloads and reallocate CPU to the remaining workloads.

If CPU is not constrained and the size of spill files for some queries is very large, make sure that the `gp_workfile_compress_algorithm` server configuration parameter is set to `zlib` and not `none`. Compressing spill files reduces disk I/O, but uses CPU cyles to compress and decompress the data.

See Using Resource Groups for more about managing performance with resource groups.

If your Greenplum Database system is configured to manage resources with resource queues, see Using Resource Queues.

CPU Skew

What is CPU Skew?

CPU skew occurs when the work to execute a query is not distributed evenly among the segments.

The **CPU** metric is the average of the CPU percentages used by each process executing the query. The **CPU skew** metric is a variance statistic based on the difference between the average and each segment's current **CPU** metric. The smaller the **CPU skew**, the more equally the work is distributed. The **CPU skew** metric varies between 0.0 (no skew) and 1.0. The lower the skew metric the more fully the database cluster's resources are utilized.

CPU skew is usually related to the volume of data processed by the segments while executing the query execution plan. There are two types of skew you should investigate: data skew and computational skew.

Data Skew

A high CPU skew may be an indication of data skew, where tables used by the query are distributed unevenly, so that some segments have more data to process than their peers. You can check for data skew in a table by running a query like this one:

```
=# SELECT gp_segment_id, COUNT(*) FROM <table-name> GROUP BY gp_segment_id;
```

The row count should be approximately equal for each segment. If the rows are distributed unevenly, check the distribution key for the table. A good distribution key is a column or list of columns with unique or nearly unique values, such as the table's primary key. Setting the distribution policy to `DISTRIBUTED RANDOMLY` also ensures a well-distributed table, but precludes taking advantage of performance-enhancing strategies such as co-location for tables with equivalent primary keys.

Computational Skew

High CPU skew can be the result of computational skew, which occurs during query execution. Some of the operations in the query plan can cause some segments to do more work than others. For example, joins, sorts, or aggregations on columns with low cardinality or unevenly distributed values can contribute to CPU skew by causing some segments to process many more tuples than others.

See Distribution and Skew in the *Greenplum Database Administrator Guide* and Tuning SQL Queries in the *Greenplum Database Best Practices* guide for more help finding the causes of skew.

Locks

Greenplum Command Center displays the locks currently held by queries and queries blocked by locks.

A block occurs when one query needs to acquire a lock that conflicts with a lock held by another query. If a query is blocked for a long period of time, you can investigate the blocking query and, if necessary, cancel one of the queries.

Locks can be acquired using the `LOCK TABLE` SQL statement. Some SQL commands acquire locks automatically. Following are descriptions of the lock modes, the Greenplum Database commands that acquire them, and which lock modes conflict with them.

ACCESS SHARE

Acquired by `SELECT` and `ANALYZE` commands.

Conflicts with ACCESS EXCLUSIVE locks.

In general, any query that only reads a table and does not modify it acquires this lock mode.

ROW SHARE

Acquired by `SELECT FOR SHARE` command.

Conflicts with EXCLUSIVE and ACCESS EXCLUSIVE locks.

A ROW SHARE lock is placed on the specified table and an ACCESS SHARE lock on any other tables referenced in the query.

ROW EXCLUSIVE

Acquired by `INSERT` and `COPY` commands.

Conflicts with SHARE, SHARE ROW EXCLUSIVE, EXCLUSIVE, and ACCESS EXCLUSIVE locks.

A ROW EXCLUSIVE lock is placed on the specified table and ACCESS SHARE locks are placed on any other referenced tables.

SHARE UPDATE EXCLUSIVE

Acquired by `VACUUM` and `VACUUM FULL`.

Conflicts with the SHARE UPDATE EXCLUSIVE, SHARE, SHARE ROW EXCLUSIVE, EXCLUSIVE, and ACCESS EXCLUSIVE locks.

SHARE UPDATE EXCLUSIVE protects a table against concurrent schema changes and `VACUUM` runs.

SHARE

Acquired by `CREATE INDEX`.

Conflicts with ROW EXCLUSIVE, SHARE UPDATE EXCLUSIVE, SHARE ROW EXCLUSIVE, EXCLUSIVE, and ACCESS EXCLUSIVE locks.

Protects a table against concurrent data changes.

SHARE ROW EXCLUSIVE

This lock mode is not automatically acquired by any Greenplum Database command.

Conflicts with ROW EXCLUSIVE, SHARE UPDATE EXCLUSIVE, SHARE, SHARE ROW EXCLUSIVE, EXCLUSIVE, and ACCESS EXCLUSIVE locks.

EXCLUSIVE

Acquired by `UPDATE`, `SELECT FOR UPDATE`, and `DELETE` commands in Greenplum Database.

Conflicts with ROW SHARE, ROW EXCLUSIVE, SHARE UPDATE EXCLUSIVE, SHARE, SHARE ROW EXCLUSIVE, EXCLUSIVE, and ACCESS EXCLUSIVE locks. This lock mode allows only concurrent ACCESS SHARE locks - a table can be read by another transaction while this lock is held. This is more restrictive locking than in regular PostgreSQL.

ACCESS EXCLUSIVE

Acquired by the `ALTER TABLE`, `DROP TABLE`, `TRUNCATE`, `REINDEX`, `CLUSTER`, and `VACUUM FULL` commands. Default lock mode for `LOCK TABLE` statements that do not specify a lock mode. Also briefly acquired by `VACUUM` (without `FULL`) on append-optimized tables during processing.

Conflicts with all locks.

This lock mode guarantees that the holder is the only transaction accessing the table in any way.

For more on locks in Greenplum Database queries, see the LOCK command Reference. See also Tuning SQL Queries.

Memory

The Greenplum Command Center Query Monitor reports the current total memory consumed by all processes executing a query. When there is insufficient memory available for a query to complete, the query has an error status in the query monitor and an out of memory error is logged.

If you have enabled resource groups in Greenplum Database, you can manage the amount of memory available to queries by tuning resource group parameters, and by setting Greenplum Database configuration parameters that affect resource group memory.

- For a detailed description of resource group memory management, see Using Resource Groups in the *Greenplum Database Administrator Guide*.
- If you are using resource queues, see Memory and Resource Management with Resource Queues and Using Resource Queues for ways to troubleshoot memory problems with resource queues.
- See Tuning SQL Queries for help with query optimization.

The following summary describes the resource group parameters and related Greenplum Database server configuration parameters that determine the amount of memory available to database queries and how configuration choices affect concurrency, spill file usage, and query performance.

Resource Group Memory Configuration Parameters

A resource group has parameters `CONCURRENCY`, `MEMORY_LIMIT`, `MEMORY_SHARED_QUOTA`, and `MEMORY_SPILL_RATIO`, which determine how much memory is allocated to execute a query. The `CPU_LIMIT` parameter has no effect on memory allocation. See the `CREATE RESOURCE GROUP` SQL reference for command syntax and information about these parameters.

`MEMORY_LIMIT`

This parameter sets the amount of memory the resource group manages as a percentage of the memory available to resource groups. The sum of all resource groups' `MEMORY_LIMIT`s must not exceed 100. If the sum of all resource groups' `MEMORY_LIMIT`s is less than 100, the remaining, unallocated memory is *global resource group shared memory*, available to queries from all resource groups on a first-come, first-served basis.

`MEMORY_SHARED_QUOTA`

A resource group divides the memory it manages into a fixed portion and a shared portion, called *resource group shared memory*. This parameter specifies the percentage of a resource group's memory that is shared. The default is 20 and the value can range from 0 to 100.

`CONCURRENCY`

This parameter limits the number of concurrent transactions a resource group allows. The fixed portion of the memory the resource group manages is divided equally among `CONCURRENCY` transaction slots. Every transaction starts with this fixed portion of memory and, if needed, Greenplum Database uses additional memory from the resource group shared memory and global resource group shared memory.

`MEMORY_SPILL_RATIO`

This parameter sets a limit for the amount of memory a query can use before it spills to disk. The parameter value is expressed as a percentage of the fixed memory allocation. The default is 20 and the value can range from 0 to 100. A higher value uses more memory, but can improve query performance. A transaction can override this value by setting the `memory_spill_ratio` configuration parameter in the session.

When a query executes, Greenplum Database allocates memory to it from the fixed portion of the resource group's memory. If the query needs more memory and the resource group has available shared memory, Greenplum Database allocates additional memory. If insufficient shared memory is available, Greenplum Database allocates additional memory from global shared memory, if available. If the required memory is not available the transaction fails.

Greenplum Database Memory Configuration Parameters

The following Greenplum Database configuration parameters affect resource group memory allocation and concurrency.

`gp_resource_group_memory_limit`

This Greenplum Database server configuration parameter sets the percentage of each host's system memory to be managed by resource groups. The default is `0.7` (70%). This memory is divided equally among the primary segments on each host, and further divided among resource groups with the `MEMORY_LIMIT` resource group parameter. Any memory not allocated to resource groups becomes global shared memory available to queries from all resource groups. See `gp_resource_group_memory_limit` for a complete reference for this parameter.

`gp_resgroup_memory_policy`

This parameter determines how Greenplum Database allocates memory to query operators. The default value, `eager_free`, re-allocates memory from completed operators to operators later in the query plan. The alternative value for this parameter, `auto`, allocates a fixed amount of memory to operators that are not memory-intensive and the rest to the memory-intensive operators. The default value is usually the optimal setting. See `gp_resgroup_memory_policy` for a complete reference for this parameter.

`memory_spill_ratio`

A transaction can override the resource group's `MEMORY_SPILL_RATIO` value by setting the `memory_spill_ratio` configuration parameter in the session. The value is a percentage of the fixed memory allocation for transactions in the resource group, expressed as an integer from 0 to 100. The performance of queries with low memory requirements can be improved by setting the `memory_spill_ratio` parameter in the session to a low setting, for example 0 to 2. See `memory_spill_ratio` for more information about this parameter.

Query Plan Execution

The Greenplum Database legacy and GPORCA query optimizers generate execution plans that produce the results requested by the query. A plan is a sequence of operators, such as table scans, joins, sorts, aggregates, and data motions.

When you select a query on the Command Center **Query Monitor** view, a **Query Details** view presents a graphical representation of the execution plan.

You can switch between the graphical and texual representations of the query execution plan by selecting the **Plan & Progress** tab or the **Texual Plan** tab. In the textual format, each plan node is flagged with an arrow (`->`). In the graphical view, the nodes are represented by boxes that fill as the plan executes.

A query execution plan executes from the bottom up. Each node in the plan performs an operation and passes results up to the next node in the plan.

The `Optimizer status:` line on the **Textual Plan** tab reports whether the explain plan was generated using the GPORCA optimizer or the legacy query optimizer.

Slices and Gangs

Segments can work on portions of a query in parallel, each segment executing operators independently on their local data. When the plan requires exchanging data between segments, a data motion operator coordinates the data transfer between segments. The plan is divided into "slices" where these data motions occur.

A data motion node in a textual query plan identifies the slice and the number of segments participating in the motion.

Example:

```
->  Broadcast Motion 4:4  (slice2; segments: 4)  (cost=0.00..867.15 rows=10000 width=3
0)
```

In a broadcast motion, each segment broadcasts all of its rows for a table over the network so that every segment has a complete copy of the table. In this example, the broadcast motion marks the completion of `slice2` with four segments sending and four segments receiving.

Each segment has one or more backend processes working on a slice. Backend processes working on the same slice are called a "gang".

Operators

Operators are processes that take as input database tables or the output from other operators, and perform some action to produce a transformed output.

Scan Operators

Init plan

A query that runs before the main query is optimized to find the partitions to scan.

Sequential scan

The optimizer may choose a sequential table scan if there is no index on the condition column or if most rows are expected to satisify the condition. Because each segment scans an equal portion of the data in parallel with other segments, a table scan in Greenplum Database is very efficient. A query on a partitioned table may be able to eliminate partitions to make the scan even faster.

Append-only scan

Scans rows in a row-oriented, append-optimized table.

Append-only columnar scan

Scans rows in a column-oriented, append-optimized table.

Dynamic table scan

Scans selected partitions in a partitioned table.

Function scan

A Function Scan node selects the partitions to scan. The function can be one of the following:

- `gp_partition_expansion` - chooses all nodes
- `gp_partition_selection` - chooses a partition with an equality expression
- `gp_partition_inversion` - chooses partitions with a range expression

Index scan

Scans a B-tree index on a table to find rows. The rows are then retrieved from disk.

Bitmap index scan

A Bitmap Index Scan is an index scan optimized by storing rows in a bitmap instead of retrieving them from the table immediately. When the scan is complete, rows in the bitmap are retrieved with a Bitmap Heap Scan operation.

BitmapAnd and BitmapOr

Generates a new bitmap by running logical AND or OR on multiple bitmaps.

Bitmap heap scan

Retrieves rows from heap storage using a bitmap generated by a Bitmap index scan or BitmapAnd or BitmapOr operation.

Nested loop with inner sequential scan join

For each row in the first table, the operator tests every row in the second table with a sequential scan.

One table must be broadcast so that every segment can compare all rows of one table with the rows it has from the other table. This is expensive and is best used only for small tables.

Nested loop with inner index scan

For each row in the first table, the operator searches an index on the second table.

One table must be broadcast so that every segment can compare all rows of one table with the rows it has from the other table.

Append

Concatenates data sets. For example, combines rows scanned from multiple partitions.

Filter

Selects rows using a `WHERE` clause.

Limit

Limits the number of rows returned.

Materialize

Saves results from a subselect so that it is not necessary to process the inner table for every row in the outer table.

Join Operators

Hash join

Creates a hash table on the join key of the smaller table. Scans the larger table and looks up matching rows in the hash table. Hash join is very fast. The hash table is held in memory, so a hash join can use a lot of memory, depending on the size of the smaller table.

Sort merge join

The tables to be joined are sorted on the join attribute and then scanned in parallel to find the matching values. This is a good join method for tables that are too large to use a hash join.

Product join

Joins every qualifying row in the first table with every qualifying row in the second table. This type of join can be very expensive if spill files must be used.

Sort and Aggregate Operators

Sort

Sorts rows to prepare for operations such as an aggregation or merge join.

Group by

Groups rows by one or more columns.

Group / hash aggregate

Aggregates rows using a hash.

Motion Operators

Broadcast motion

Every segment sends its own local data to all other segment instances so that every segment instance has a complete local copy of the table.

Redistribution motion

Sends data from one table to another segment so that matching rows are located together, enabling a local join.

Gather motion

All segments send rows to the master where they are gathered into a single result set.

DML Operators

Assert

Performs constraints checking.

Split

Used for update operations.

Spill Files

Greenplum Command Center reports the total size for all spill files created for a query.

Greenplum Database creates spill files, also called workfiles, to save intermediate results when there is insufficient memory to execute a query in memory. Disk I/O is much slower than memory access, so a query that creates spill files will take longer to complete.

Investigating Spill File Usage

The `gp_toolkit` schema contains views you can use to see details about spill file usage for current queries. You can see the number and sizes of spill files created for each operator in a query execution plan, and totals by query and segment. This is useful information to detect data skew and to help tune queries.

See the gp_toolkit Administrative Schema reference for descriptions of these views.

Eliminating or Reducing Spill Files

You can work to eliminate spill files by increasing the amount of memory available to the query or by optimizing the query to use the memory available more efficiently.

You may be able to revise the query to prevent spilling by eliminating or postponing memory-intensive operators.

Following are some ways to increase memory available to queries when resource group resource management is enabled in Greenplum Database.

- Decrease the resource group's concurrency so that each query's share of memory increases.
- Increase the resource group's `MEMORY_SHARED_QUOTA` parameter to increase the amount of resource group shared memory.
- Decrease the percentage of memory allocated to all resource groups to increase the amount of global shared memory.

When resource queue resource management is active, Greenplum Database can detect and terminate "runaway" queries that consume a high percentage of available memory. You can prevent runaway queries by limiting the number of spill files created or the total size of spill files created. See the `gp_workfile_limit*` configuration parameters below for more information.

If you cannot prevent queries from spilling, it is important to ensure that the number of spill files created is mnimized and that problems such as CPU or data skew are found and corrected. Skew can create excessive numbers of spill files on one or more segments.

To minimize disk usage and I/O when spill files are created, make sure the `gp_workfile_compress_algorithm` configuration parameter is set to 'zlib' and not 'none'.

Limiting Spill Files with Server Configuration Parameters

Greenplum Database by default limits the number of spill files allowed per query for each segment to 100,000. You can raise or lower this limit, and you can also limit the number of spill files for all queries on a segment, and limit the disk space consumed by spill files per query and per segment. Use the following Greenplum Database server configuration parameters to manage spill files.

`gp_workfile_limit_files_per_query`

Sets the maximum number of spill files allowed per query per segment. Default is 100,000.

`gp_workfile_limit_per_query`

Sets the maximum disk size an individual query is allowed to use for spill files at each segment. The default value is 0, which means no limit is enforced.

`gp_workfile_limit_per_segment`

Sets the maximum total disk size that all running queries are allowed to use for creating spill files at each segment. The default value is 0, which means a limit is not enforced.

`gp_workfile_compress_algorithm`

Specifies the compression algorithm to use for spill files when a hash aggregation or hash join operation spills to disk during query processing. The default is `'none'`. Set to `'zlib'` to enable compression. Using compression reduces the number of I/O operations at the expense of increased CPU.

See also Managing Spill Files Generated by Queries.

Alert Help Topics

Database connectivity failure

What is this alert?

Command Center raises an alert if it is unable to connect to the Greenplum Database system. If Command Center is running on the Greenplum Database master host, it is likely that the database is down or not accepting connections.

If Command Center is running on the standby master host, the problem could be that the master host is down, there is a networking issue, or that the database is down or not accepting connections.

Command Center tries to connect to the database three times before issuing an alert.

What to do?

If Command Center is running on the Greenplum Database master host:

- Run the `gpstate` command-line utility on the Greenplum master to check the status of the database system. Correct any problems identified in the command output.

If Command Center is running on the Greenplum Database standby master host:

- Log in to the standby master host as the gpadmin user and then SSH to the master host.

  ```
  $ ssh gpadmin@<master-hostname>
  ```

  Be sure that <master-hostname> matches the `hostname` column in the `gp_segment_configuration` system table. If you are unable to reach the master host, make sure the host is running and resolve any networking issues.
- Run the `gpstate` command-line utility on the Greenplum master to check the status of the database system. Correct any problems identified in the command output.

Segment failure

What is this alert?

Command Center checks the status of the Greenplum Database segments every 30 seconds and raises an alert if any segments are down or running in their non-preferred roles. This alert will be raised hourly until an administrator has recovered the failed segments.

With segment mirroring enabled, Greenplum Database can tolerate a primary or mirror segment failure as long as there is a working instance for every segment in the cluster. If both the primary and mirror instances fail for any single segment, Greenplum Database cannot run queries. For this reason, it is important to recover the failed segment instance to protect from loss of service.

Segment instances have a "preferred role," either primary or mirror. When all segment instances are in their preferred roles, each segment host has the same number of primary and mirror segments. If a primary instance fails, its mirror instance assumes the primary role and the distribution of primary segments is no longer balanced. This can slow down query processing because a host with more primary segments than others can take longer to complete queries.

What to do

Restore the failed segments and return the segments to their preferred roles so that the cluster is in balance.

See Recovering From Segment Failures for steps to recover Greenplum Database segments.

Average memory (segment hosts) exceeds [%] for [min]

What is this alert?

This alert warns of high memory consumption on the Greenplum Database segment hosts for an extended period.

Every 15 seconds, the Greenplum Database metrics collector extension samples the memory in use on each segment host. Memory used for kernel cache and buffers is excluded. The average for all segments is calculated. If the average percentage remains above the threshold that is set for the alert for the number of minutes specified, an alert is issued.

If memory consumption is increasing over time, queries could start to faile with out of memory errors.

What to do?

Check the **Query Monitor** to see if there is unusually heavy query activity.

Look for active queries that perform hash joins or sorts on a large number of tuples. If possible, optimize the queries to eliminate rows earlier so that these memory-intensive operations process a fewer number of tuples.

Adjust resource queues or resource groups to limit the number of concurrent transactions.

Use the `ps` command to identify non-database processes consuming excessive memory. Kill unnecessary processes or move them to another server.

Memory (master) exceeds [%] for [min]

What is this alert?

When the master host memory usage is higher than the specified percentage for more than the specified number of minutes, Command Center raises an alert.

Command Center samples the memory in use on the master host every fifteen seconds. Memory used for kernel buffers and cache is excluded from the calculation. An alert is raised if the samples collected during the number of minutes specified are all higher than the specified percentage.

What to do

Use the `ps` command to identify non-database processes consuming excessive memory and stop them or relocate them to other servers.

If Command Center is running on the master host, restart it on the standby master host.

Total disk space exceeds [%]

What is this alert?

This alert is raised when the percentage of segment host disk space in use exceeds the percentage specified in the alert rule. The master disk space is not included in the calculation. The alert is raised once a day until the percentage drops below the percentage in the alert rule.

What to do

This alert warns you so that you can add disk storage or free up storage in order to prevent a catastrophic disk full error that could interrupt Greenplum Database service.

Here are some suggestions for freeing space on Greenplum Database hosts.

- Archive and remove backup files
- Archive and drop older partitions
- Rotate OS and database log files

- Drop unneeded external tables and their data files
- Vaccuum database tables and catalog tables

Query is blocked for [min]

What is this alert?

If a query that has started to execute is blocked by another query for the specified number of minutes, Command Center raises an alert.

Queries that are queued, but have not yet started, do not trigger this alert.

Alert details

The alert contains the Query ID, database name and user, and run-time details.

Qid": {"Tmid":1541113373,"Ssid":6968,"Ccnt":3},"Database":"postgres","User":"gpmon","SubmitTime":"2018 -11-02T16:10:04+08:00","StartTime":"2018-11-02T16:10:04+08:00","QueryText":""}

Need alert type specific JSON example to write this out

What to do

1. Use the **Query Monitor** to locate the blocked query and the query that blocks it.
2. Determine whether the blocking query is executing properly:
    - Is the query also blocked?
    - Is the query blocking a large number of other queries?
    - Is the query creating excessive spill files?
    - Is the query running in the correct resource group or resource queue?
    - Is the query running longer than usual?
    - Does the query have excessive data or CPU skew?
3. Determine whether you should allow the query to complete, or cancel the query so that the blocked queries can resume.

Number of connections exceeds [n]

What is this alert?

This alert is raised when the number of concurrent connections at the Greenplum Database master instance exceeds a specified number. The number specified should be set lower than the `max_connections` server configuration parameter so that when you receive the alert you can act before Greenplum Database begins to reject client connection requests. For example, if `max_connections` for the master is set to 100, you could set an alert to 80.

What to do

Look for idle or stale connections and terminate them

Users can connect to Greenplum Database using a client such as `psql`, execute queries and remain connected, but inactive, leaving the connection in an idle state. Greenplum Database eventually releases resources used by idle connections, but once the maximum number of allowed connections has been reached, new connection requests are denied.

Use the `pg_stat_activity` system view to find idle connections.

```
SELECT datname, procpid, sess_id, usename, current_query from pg_stat_activity;
```

Use the `pg_cancel_backend(<PID>)` function to cancel idle connections.

Determining *how long* a connection has been idle is not possible with just the information in the Greenplum Database 5.x system tables. You can, however, see this information by creating the `session_level_memory_consumption` view in the database. Follow the instructions at Viewing Session Memory Usage Information to create this view in each database.

After you install the `session_level_memory_consumption` view, a query like the following shows the idle connections with the length of time they have been idle.

```
SELECT a.datname, application_name, a.usename, a.sess_id, procpid,
    now()-idle_start AS time_idle
FROM pg_stat_activity a, session_state.session_level_memory_consumption b
WHERE a.sess_id = b.sess_id AND b.idle_start < now();
ORDER BY time_idle DESC;
```

CPU (master) exceeds [%] for [min]

What is this alert?

The metrics collector extension checks CPU utilization on the master host every 15 seconds. If the percentage of CPU in use is higher than the percentage specified in the alert rule for the number of minutes specified in the rule, Command Center raises an alert.

The Greenplum Database master uses the greatest amount of CPU at the start of a query, while planning the query, and at the end of the query, while gathering results from the segments. For a large result set it is normal to see a spike in the query's CPU usage on the master during the gather operation. With many queries running concurrently, the CPU spikes should even out.

What to do

Begin by viewing the Command Center **Query Monitor** and **Query Details** views to see if there are one or more very large queries nearing completion, or if the high usage can be explained by unusual workloads or heavy query activity.

If the query monitor offers no explanation for high CPU usage, you should investigate master host processes using Linux utilities such as `ps` and `top` to identify processes consuming the CPU. If the process ID of an errant process is a `postgres` process, you can query the `pg_stat_activity` system table to find the query and, if needed, use the `pg_terminate_backend()` function to terminate the query.

See also

Investigating a Performance Problem

Out of memory error

What is this alert?

If a query requests additional memory and is denied, the query fails with an out of memory error and an alert is raised.

What to do

Greenplum Database has two ways to manage memory resources: resource queues and resource groups. Resource queues deal primarily with fixed quantities of memory, where resource groups deal with portions—percentages—of available memory.

If you use resource groups to manage memory, you can use the **Workload Mgmt** view to adjust them so that more memory is available to queries that are failing due to out of memory errors. If you use resource queues, you use the `CREATE RESOURCE QUEUE` and `ALTER RESOURCE QUEUE` SQL commands to configure them.

There are many factors to consider when allocating memory for queries, including configuring the operating system, allocating a share of memory to Greenplum Database, and configuring a set of resource queues or resource groups to share the memory available to Greenplum Database.

For complete information about how Greenplum Database manages memory and how to configure it, see:

- Using Resource Groups
- Using Resource Queues

Query runtime exceeds [min]

What is this alert?

An alert is raised if the total runtime for a query is greater than the specified number of minutes. The alert is raised once per query.

Run time is calculated from the time the query begins to execute. The time the query was queued is excluded.

Alert details

- query ID
- database name

- user name
- time the query was submitted
- time the query started

What to do

Use the **Query Monitor** to check the execution status of the query.

If the query is blocked, investigate the queries that hold the locks.

Average CPU (segment hosts) exceeds [%] for [min]

What is this alert?

Command Center samples CPU usage on all segment hosts every 15 seconds and calculates the average CPU usage for the cluster. An alert is raised if the average CPU usage is greater than the specified percentage for longer than the specified number of minutes.

What to do

Use the Command Center **Query Monitor** to identify currently running individual queries with high CPU usage.

Use the Command Center **History** view to see CPU load during the period prior to the alert and identify completed queries using too much CPU.

Check CPU usage using operating system utilities such as `ps` and `top` to identify any operating system processes that are consuming excessive CPU, for example backup, restore, or ETL processes.

Spill files for a query exceeds [GB]

What is this alert?

Command Center raises an alert if the combined size of spill files for any query exceeds the specified number of gigabytes. This alert is raised just once per query.

Greenplum Database creates spill files to temporarily store data on disk when the data exceeds the amount of memory allocated to the operation. Because memory I/O is much faster than disk I/O, a query that creates spill files takes longer to complete than it would if there was sufficient memory available to avoid creating spill files.

What to do

Use the Command Center **Query Monitor** to view the plan for the query identified in the alert.

If possible, revise the query so that more rows are eliminated earlier in the plan, eliminating or reducing the size of spill files.

Consider reconfiguring the resource queue or resource group that manages the query to make more memory available to the query. If you use resource groups to manage resources, you can use the Command Center **Workload Mgmt** view to modify resource allocations.

For more information

- Managing Spill Files Generated by Queries
- Using Resource Queues for information about configuring Greenplum Database resource queues
- Using Resource Groups for information about configuring resource groups

FATAL happened on master

What is this alert?

When a message with severity `FATAL` is written to the Greenplum Database log on the master host, Command Center raises an alert.

A message with severity level `FATAL` indicates an error occurred that caused the current session to fail. Fatal errors can be caused by user error, such as misspelling a database name, or by system error, such as an out of memory error, failure to access a resource, or network failure.

Here are some examples that could result in FATAL messages:

- A user or application attempts to connect to a database that does not exist.
- The master host is unable to create a connection to a segment host.
- A memory allocation request is refused due to low memory.
- A user or administrator cancels a command.

Note: To avoid excessive alert emails, Command Center does not raise alerts for fatal password authentication errors.

What to do?

The message text often provides the information needed to determine the cause and remedy for the error.

Recurring errors without an evident cause require investigation to find the problem.

Here are some questions to consider.

Does the Command Center Dashboard indicate a normal Database State? Also check the Dashboard for low disk space, unusually high CPU or Memory usage, and a large number of blocked queries. Problems indicated on the Dashboard can help identify the source of problems.

Does the error occur only when the Greenplum Database system is under heavy load? Some system resources can be depleted under heavy load and may require reconfiguration. For example:

- The SSH service on segment hosts may limit the number of active connections.
- Out of memory errors could require more conservative settings for system memory parameters or tuning resource groups to limit the number of concurrent queries or the

amount of memory allocated to queries.

Does the message occur with a specific query, UDF, or application? Check the Greenplum Database log files and application log files for messages that help to identify the location and cause of the error.

If the error continues and you cannot find the cause, create a ticket with Pivotal Support. Support may ask you to provide log files, core files, configuration files, schemas, and system information so that they can help you to resolve the problem.

PANIC happened on master

What is this alert?

When a message with severity level `PANIC` is written to the Greenplum Database log on the master host, Command Center raises an alert.

A `PANIC` message reports a critical error that requires immediate attention. A panic occurs when the system encounters an unexpected state or an error condition that could not be handled in the software. All currently executing jobs in the cluster are cancelled when a `PANIC` occurs.

Some examples of conditions that could cause a PANIC are:

- Unable to access critical files
- Full disk or memory buffer
- Unexpected data values, such as badly formatted transaction IDs
- Seg faults or null pointer references

What to do

Contact Pivotal Support for help before you attempt to troubleshoot the problem. Continuing after a `PANIC` could lead to data loss and may make it more difficult to diagnose the problem and recover your system.