Best Practices for HP Vertica OEM Customers

HP Vertica Analytic Database

Software Version: 7.0.x

Document Release Date: 2/24/2014

Legal Notices

Warranty

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

The information contained herein is subject to change without notice.

Restricted Rights Legend

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

Copyright Notice

Trademark Notices

Adobe® is a trademark of Adobe Systems Incorporated.

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

UNIX® is a registered trademark of The Open Group.

Contents

About This Document

This document outlines best practices and tips for Hewlett-Packard OEM customers.

OEM customers embed the HP Vertica database with their own application and deliver it as an on-the-premises solution. Typically, the end users of the application do not interact with the HP Vertica database directly; they interact with the database only through the application.

The information in this document addresses technical issues for three types of tasks that Hewlett-Packard's OEM customer must perform to optimize the database in their application:

Part 1: Developing and Testing the HP Vertica Database to Work with a Customer Application

During your application development, install an HP Vertica database and load it with sample data. You can simulate and test the behavior of the database and application in a production-like environment.

The following sections describe the steps to take and the HP Vertica features to focus on for the HP Vertica database to work as desired with your application at a customer site:

Installing the HP Vertica Database

To install the HP Vertica software on your development system, follow the instructions in the Installation Guide of the HP Vertica documentation.

Loading Representative Data

To effectively test the way the HP Vertica database interacts with your application in a production environment, load data that is representative of the data that might be used in a production environment. Representative data allows you to effectively test database functionality and performance.

For more information about loading data into an HP Vertica database, see Bulk Loading Data in the Administrator's Guide.

Identifying Typical Queries

Identify common queries that your application uses. If you supply these queries to Database Designer, Database Designer uses those queries to create projections that maximize performance throughput in your database.

Optimizing the Database with Database Designer

An HP Vertica database stores its data in a physical format called *projections*. For optimal performance, projections should be optimized based on data and workload characteristics.

As an HP Vertica OEM customers, you should optimize the projection design during your development cycle.You can standardize on that design for all deployments of your application.

The following sections describe the process of tuning projections using Database Designer:

- What Does Database Designer Do? in the Administrator's Guide gives an overall view of what Database Designer does for your database.
- Results of Running Database Designer in the Administrator's Guide explains what benefits your database achieves by running Database Designer.
- Exporting the Optimized Database in this guide contains information about how to deploy the optimized projections when the database is embedded with your application.

After you create your database for the first time and load the typical queries and sample data on your development system, create a comprehensive database design as described in Creating a Comprehensive Design Using Database Designer in the Administrator's Guide.

What Does Database Designer Do?

Database Designer uses sophisticated strategies to create physical schemas or database designs. These designs provide excellent ad-hoc query performance while using disk space efficiently.

Database Designer can create two types of designs:

- A comprehensive design allows you to create new projections for all tables in your database.
- A query-specific design creates projections for all tables referenced in the queries you supply to Database Designer.

Database Designer accepts unlimited sample queries for a comprehensive design and up to 100 sample queries in the query input file for a query-specific design.

Results of Running Database Designer

In most cases, the projections that Database Designer creates provide excellent query performance within physical constraints while using disk space efficiently. Database Designer:

- Recommends buddy projections with the same sort order/ The buddy projections can significantly improve load, recovery, and site node performance. By default, Database Designer recommends both super and non-super segmented projections with a buddy of the same sort order.

- Automatically rebalances data after you add or remove nodes.
- Produces higher quality designs by considering UPDATE and DELETE statements.

Database Designer creates two scripts:

- Design script—Contains all the DDL statements that are needed to create the design.
- Deployment script—Contains all the CREATE PROJECTION scripts that are needed to create the projections for the design. The deployment script is a subset of the design script.

Exporting the Optimized Database in this guide describes how to configure these scripts so that all customers can be standardized on the optimal projections. The scripts must be portable to deployments with different numbers of nodes and K-safety levels.

Exporting the Optimized Database

Once you have optimized the database using Database Designer and other techniques, make sure that the optimized projections are available for all customers in the deployed database.

The following HP Vertica functions generate scripts that create the database on the customer's cluster:

- EXPORT_CATALOG—Generates a SQL script that recreates a physical schema design in its current state on a different cluster.
- EXPORT_OBJECTS—Generates a SQL script that recreates catalog objects (only the non-virtual objects to which you have access) on a different cluster. Running the generated SQL script on another cluster creates all referenced objects before their dependent objects.
- EXPORT_TABLES—Generates a SQL script that recreates a logical schema (schemas, tables, constraints, and views) on a different cluster.

Once these scripts have been created, edit the scripts as follows:

- Modify the CREATE PROJECTION statements to include the appropriate segmentation clauses.
- Modify the CREATE PROJECTION statements to specify ALL NODES KSAFE <*value*>. As a result, all unsegmented projections are replicated on all nodes, with the designated K-safety value. If you don't specify a K-safety value, HP Vertica uses the current K-safety value, or 1.

Managing Database Resources

For large data sets of complex workloads, Hewlett-Packard recommends that you run the database software on dedicated servers.

In some installations, you may have to run the HP Vertica database on a shared server with your application. In those situations, configure CPU and memory resources for both the database and your application to ensure that neither application can monopolize either CPU or memory from the other. Consider the following tips to manage resources between them:

Configuring CPU Resources

Configure CPU resources in two ways:

- Pinning HP Vertica processes to certain CPUs
- Setting run-time priorities for resource pools

Pinning HP Vertica Processes to Certain CPUs

To pin an entire HP Vertica server process and its child processes to execute on a specified set of CPUs in the cluster, use the following configuration parameters. These parameters do not affect external processes and applications:

- `PINPROCESSORS`: Number of CPUs for HP Vertica processes to be pinned to.
- `PINPROCESSORSOFFSET`: Offset for the start of CPUs to be used by the HP Vertica server.

For example, in a cluster where each node has 24 CPUs, numbered 0 through 23, the server process can be restricted to CPUs #16–23 by entering the following statements:

- `SELECT SET_CONFIGURATION_PARAMETER('PINPROCESSORS', 8)`
- `SELECT SET_CONFIGURATION_PARAMETER('PINPROCESSORSOFFSET', 16)`

Pin your application to the processors that are not allocated to the HP Vertica database. To make sure that other applications do not use the processors assigned to the HP Vertica server, use the Linux `cgroups` command.

Setting Run-Time Priorities for Resource Pools

For each resource pool, manage resources that are assigned to queries that are already running in that resource pool. Assign each resource pool a run-time priority of HIGH, MEDIUM, or LOW. These settings determine the amount of run-time resources (such as CPU and I/O bandwidth) assigned to queries in the resource pool when they run. Queries in a resource pool with a HIGH priority are assigned greater run-time resources than those in resource pools with MEDIUM or LOW run-time priorities.

Configuring Memory Resources

Set the `MAXMEMORYSIZE` parameter on the `GENERAL` resource pool to limit the amount of overall system memory that you allocate to the HP Vertica database.

To ensure that there is no swapping, make sure your application is limited to the memory that the database is not using.

For example, if the MAXMEMORYSIZE for the GENERAL pool for the database is set to 32 GB on a system with 48 GB of RAM, your application should use no more than (48–32) = 16 GB of RAM.

Loading Data into the Database

While your application is running, you load data into your HP Vertica database on a regular basis. You can collect large amounts of data to load into your database at once, as long as you can meet your latency service-level agreement (SLA).

Simulate your data-loading operations in a production environment. Make sure to replicate the maximum expected rates of loading and spiking.

The following sections describe how HP Vertica stores data, how to load data into your database, and how you can tune the processes and monitor the system to prevent performance problems:

- WOS and ROS
- How Loading Works
- Loading Data from a File
- Loading Data from Your Application
- Parallel Load Streams
- Trickle Loading
- Monitoring Load Operations

WOS and ROS

Write Optimized Store (WOS) is a memory-resident data structure for storing INSERT, UPDATE, DELETE, and COPY (without /*+DIRECT*/ hints) actions. To support very fast data load speeds, the WOS stores records without data compression or indexing. The WOS organizes data by epoch and holds both committed and uncommitted transaction data.

Read Optimized Store (ROS) is a highly optimized, read-oriented, disk storage structure. The ROS makes heavy use of compression and indexing. You can use the COPY...DIRECT and INSERT (with /*+DIRECT*/ hints) statements to load data directly into the ROS.

The Tuple Mover (TM) is the Vertica Analytic Database database optimizer component that moves data from memory (WOS) to disk (ROS). The Tuple Mover runs in the background, performing some tasks automatically at time intervals determined by its configuration parameters.

For information about the WOS, ROS, and Tuple Mover, see Hybrid Storage Model in the Concepts Guide.

How Loading Works

When you load data into a database, HP Vertica first moves it into memory (WOS) by default. (You can override this default, and specify that HP Vertica move the data directly to disk (ROS). The Tuple Mover runs in the background. Periodically, the Tuple Mover

- Moves data from the WOS to the ROS.
- Combines like-size ROS containers.
- Deletes purges records.

If you move too much data too frequently to the WOS, the data can spill to disk. No data is lost, but this can cause performance problems.

Loading Data from a File

If the data to be loaded into your database is stored in a file, use the COPY statement to load the data. You can also use COPY to load data from a data stream.

In its basic form, use COPY as follows:

```
COPY to_table FROM data_source
```

For more information, see the following sections of the HP Vertica documentation:

- Using COPY and COPY LOCAL in the Administrator's Guide
- COPY in the SQL Reference Manual
- COPY LOCAL in the SQL Reference Manual

Loading Data from Your Application

If the application needs to push data into the HP Vertica database, use the driver associated with the programming language that your application uses. For more information, see:

- Loading Data through ODBC in the Programmer's Guide
- Loading Data through JDBC in the Programmer's Guide
- Loading Data through ADO.NET in the Programmer's Guide

Parallel Load Streams

When you have a large amount of data to load, use parallel load streams to distribute the load operations across the cluster. To do this, create threads from multiple nodes that are connected to the HP Verticadatabase and load the data. This approach lets you use vsql, ODBC, ADO.NET, or JDBC. You can load server-side files, or you can load client-side files using the COPY from LOCAL statement.

Best practices for parallel load streams are:

- Issue a single multi-node COPY command that loads different files from different nodes specifying the *nodename* option for each file.
- Issue a single multi-node COPY command that loads different files from any node, using the ON ANY NODE option.
- Use the COPY x WITH SOURCE PloadDelimitedSource option to parallel load using all cores on the server node on which the file resides.

For additional information, see Using Parallel Load Streams in the Administrator's Guide.

Trickle Loading

Trickle loading is the process of loading data into your database periodically. Trickle loading that the load process is sustainable and doesn't result in too many ROS containers.

HP Vertica uses the transaction isolation level of READ COMMITTED, which allows users to see the most recently committed data without holding any locks. READ COMMITTED also allows new data to be loaded while concurrent queries are running.

Tune the trickle-loading operations by:

- Adjusting batch size. Generally, larger load batches are more efficient.
- Altering the moveout/mergeout parameters:
  - MergeOutInterval
  - MoveOutInterval
  - MoveOutMaxAgeTime
  - MoveOutSizePct
- To limit the load process to a specific resource so that other resources are available to queries, create a resource pool specifically for the trickle-loading process.

Monitoring Load Operations

When you simulate the load operations while developing your application, there are two system tables to monitor. These tables help you make sure that you have configured load operations in a way that does not negatively impact the database or application performance:

- PROJECTION_STORAGE
- PARTITIONS

PROJECTION_STORAGE

While developing your application, monitor the ROS count over time. Do this by querying the ROS_COUNT field in the PROJECTION_STORAGE system table and querying partitions that show the ROS_COUNT per partition.

Make sure the ROS_COUNT field is not approaching the default maximum of 1024. If this is sustainable for the duration of a partition, and through the time for an old partition to merge, your loading techniques should work fine.

```
VMart=> SELECT projection_id, projection_name, ros_count
        FROM projection_storage;
   projection_id   |       projection_name       | ros_count
-------------------+-----------------------------+-----------
 45035996273719430 | promotion_dimension_super   |         1
 45035996273720042 | online_page_dimension_super |         1
 45035996273719788 | employee_dimension_super    |         1
 45035996273719276 | store_dimension_super       |         1
 45035996273719610 | customer_dimension_super    |         1
 45035996273722132 | T_super                     |         1
 45035996273719098 | product_dimension_super     |         1
 45035996273719942 | warehouse_dimension_super   |         1
 45035996273720000 | shipping_dimension_super    |         1
 45035996273720498 | online_sales_fact_super     |         1
 45035996273719536 | vendor_dimension_super      |         1
 45035996273720206 | store_sales_fact_super      |         1
 45035996273720652 | inventory_fact_super        |         1
 45035996273718920 | date_dimension_super        |         1
 45035996273720336 | store_orders_fact_super     |         1
 45035996273720100 | call_center_dimension_super |         1
 45035996273722194 | T2_super                    |         1
(17 rows)
```

PARTITIONS

While developing your application, monitor the PARTITIONS system table. Make sure that the number of ROS containers per partition is not reaching the maximum of 1024.

In addition, monitor the `ros_size_bytes` field to see if there are any small ROS containers that you can merge.

```
=> SELECT projection_name, ros_id, ros_size_bytes
   FROM partitions;
 projection_name |      ros_id       | ros_size_bytes
-----------------+-------------------+----------------

 t_p_node0001    | 45035996273740461 |             90
 t_p_node0001    | 45035996273740477 |             99
 t_p_node0001    | 45035996273740493 |             99
```

Dropping Partitions Due to Space Constraints

If your hardware has fixed disk space, you may need to configure a regular process to roll out old data by dropping partitions.

For example, if you have only enough space to store data for a fixed number of days, configure HP Vertica to drop the partition with the oldest date. To do this, create a time-based job scheduler like `cron` to drop the partition on a regular basis during low-load periods.

If the ingest rate for data has peaks and valleys, you can use two techniques to manage how you drop partitions:

- Set up a process to check the disk space on a regular (daily) basis. If the percentage of used disk space exceeds a certain threshold, like 80%, drop the oldest partition.
- Add an artificial column in a partition that increments based on a metric like row count. For example, that column might increment each time that the row count increases by 100 rows. Set up a process that queries that column on a regular (daily) basis. If the value in the new column exceeds a certain threshold, for example, 100, drop the oldest partition and set the column value back to 0.

For more information about partitions, see the following sections in the HP Vertica documentation:

- CREATE TABLE in the SQL Reference Manual
- Working with Table Partitions in the Administrator's Guide

Load Balancing

In HP Vertica, load balancing supports multiple client connections through a IP address that is shared among all nodes in a cluster. Your cluster needs to balance incoming client requests across nodes, as well as prevent node exclusion from clients in the case of node failure.

While you are developing your application, simulate a high load of activity in your cluster that mimics the expected behavior once the application is up and running.

There are three ways you can perform load balancing on your database cluster:

- IP Virtual Server (IPVS)
- Native Connection Load Balancing
- Third-Party Load Balancers

For more detailed information, see the Connection Load Balancing and Adding Nodes sections in the Administrator's Guide.

IP Virtual Server (IPVS)

The IP Virtual Server (IPVS) running on a database node provides load balancing for an HP Vertica database cluster.

IPVS balances the connection streams, and is made up of the following components:

- **The Virtual IP (VIP)**—The IP address that is accessed by all client connections.
- **Real server IPs (RIP)**—The IP addresses of client network interfaces used for connecting database clients to the database engine.
- **Cluster**—A cluster of real HP Vertica servers (nodes).
- **Virtual server**—The single point of entry that provides access to a cluster, based on dynamic node selection.

The IPVS load balancer is two-node stand-by redundancy only. The IPVS redundancy model is different than that of the HP Vertica Analytics Platform.

HP Vertica provides the ability to add, remove, and replace nodes on a live cluster that is actively processing queries. This allows you to scale the database without interrupting users. You can also consider refreshing or dropping projections.

For detailed information, see the Connection Load Balancing and Adding Nodes sections in the Administrator's Guide.

Native Connection Load Balancing

The HP Vertica server and client libraries have native connection load balancing built in. This feature spreads the CPU and memory overhead caused by client connections across all nodes in the database cluster. Native connection load balancing can prevent some nodes from becoming burdened with many client connections while other nodes in the cluster have only a few client connections.

You must enable native connection load balancing on both the HP Verticaserver and the client.

For more information about native connection load balancing, see the following topics in the Administrator's Guide:

- About Native Connection Load Balancing
- Enabling and Disabling Native Connection Load Balancing
- Monitoring Native Connection Load Balancing

To learn how to enable native connection load balancing with your specific client software, see the following topics in the Programmer's Guide:

- Enabling Native Connection Load Balancing in JDBC
- Enabling Native Connection Load Balancing in ODBC
- Enabling Native Connection Load Balancing in ADO.NET

Third-Party Load Balancers

You can perform load balancing across your HP Vertica database cluster using third-party load balancers.

Third-party load balancers can be software applications or hardware appliances. They manage the inbound and outboard data traffic and network connections throughout the cluster. Their goal is to distribute the workload evenly among all the nodes in the cluster.

There are many third-party load balancers available. Consider the following characteristics of your configuration before choosing a load balancer for the cluster your HP Vertica database is running on:

- Size of the cluster
- Potential future growth of your cluster
- Security
- Performance

Delete and Update Operations

Many OEMs have small configuration tables with more of an online transaction processing (OLTP) workload. Your HP Vertica database can handle this situation efficiently.

The following sections explain how to configure your database to ensure optimal performance during delete and update operations:

- How HP Vertica Deletes and Updates Data
- Buffering Deletes and Updates
- Monitoring Deletes and Updates
- Purging Deleted Data

How HP Vertica Deletes and Updates Data

In HP Vertica, delete operations do not actually remove rows from physical storage. Instead, the HP Vertica DELETE statement marks rows as deleted.

Update operations have the same effect; the UPDATE statement is actually a combined delete and insert operation. The deleted rows remain in physical storage until the Tuple Mover performs a purge operation to permanently remove deleted data so that the disk space can be reused.

Note: For more information about how deletes and updates work in HP Vertica, see Understanding the Tuple Mover in the Administrator's Guide.

Buffering Deletes and Updates

The most efficient way for HP Vertica to perform delete and update operations is to update many database records at a time using fewer statements. You accomplish this by first buffering database deletes and updates in your application before submitting them

to HP Vertica. That way, you can manage the flow of the delete and update operations so that they do not interfere with normal database performance.

For more information, see Best Practices for DELETEs and UPDATEs in the Administrator's Guide.

Monitoring Deletes and Updates

The frequency of submitting the deletes and updates to the database should be sustainable and should not negatively impact performance. While you develop your application, make sure to monitor the delete and update operations in the database. To do this, monitor the `DELETE_VECTORS` system table to view the number of deleted records and their corresponding delete vectors that are still physically stored in the system:

```
VMart=> SELECT node_name, schema_name, projection_name,
           deleted_row_count, used_bytes FROM delete_vectors;
-[ RECORD 1 ]-----+-------------------------
node_name         | v_vmart_node0001
schema_name       | public
projection_name   | product_dimension_super
deleted_row_count | 32345
used_bytes        | 524288
-[ RECORD 2 ]-----+-------------------------
node_name         | v_vmart_node0001
schema_name       | public
projection_name   | employee_dimension_super
deleted_row_count | 943
used_bytes        | 54788
-[ RECORD 3 ]-----+-------------------------
node_name         | v_vmart_node0001
schema_name       | public
projection_name   | warehouse_dimension_super
deleted_row_count | 20
used_bytes        | 16384
```

Purging Deleted Data

Consider scheduling a nightly purge of small tables that receive frequent updates. This operation rewrites the tables while removing deleted records. Run the nightly purge against small tables that would not be expensive to rebuild and that have many deleted records.

For more information, see PURGE_TABLE in the SQL Reference Manual.

Part 2: Packaging the HP Vertica Database with Your Application

Once you have installed your HP Vertica database, loaded sample data into it, and tested the database operation in a simulated production environment, create a script that installs and configures the database with your application. This script should

- Perform its operations "silently", in a way that is invisible to your end users.
- Be portable so that it can install the database on a cluster with any number of nodes.

The following sections describe what functions the script should perform.

Installing the HP Vertica Database at the Command Line

The packaging script should contain commands to execute the HP Vertica installation procedure.

Important: HP Vertica must be installed separately for each user. You cannot create a system image for subsequent installations, which may cause all users to use the same SSH keys. This situation can result in serious security issues.

The commands in the packaging script should perform the following tasks:

1. Login as the root user.
2. Create a properties file that enables non-interactive setup by supplying the parameters you want HP Vertica to use. The properties file is described in Installing HP Vertica Silently in the Installation Guide.
3. Install the database by running the `install_vertica` script, as described in Installing HP Vertica Silently in the Installation Guide.

a. Make sure to run the scripts with the `-y` parameter to avoid the script asking for EULA agreement.

b. On a single-node installation, consider using the host name or IP address instead of localhost or 127.0.0.1 because of the following limitations:

   i. The `-s` parameter is required for multi-node installations. On single-node installations the parameter is optional and the default is `localhost`. However, if you plan to expand to additional hosts later, you must use the `-s` parameter. If you do not use the `-s` parameter, your HP Vertica installation cannot be upgraded to a multi-node deployment.

   ii. On a single-node localhost installation, the installer does not set up a passwordless ssh. HP Vertica's backup scripts require that the administrator be able to log into the node via ssh without a password. Consequently, you must manually enable passwordless ssh logins for any single-node installation if you want to use the backup scripts.

   iii. If either side of a connection is a single-node cluster installed to local house, or you did not specify a host name or IP address for your single-node cluster, importing and exporting data fails.

4. To save any warnings that the `install_vertica` script returns, redirect any warning messages to a separate file by specifying the `redirect_output = filename` parameter in the properties file. After the installation completes, review the warnings and correct any problems that are critical.

5. The following steps are optional:
   - Installing the HP Vertica Client Drivers in the Programmer's Guide
   - Optionally Install the vsql Client Application on Non-Cluster Hosts in the Installation Guide

6. Disconnect from the Administration Host.

7. Unless you used the `-L` parameter in the properties file, install the license key you downloaded.

Note: To use data or catalog directories other than those that the `install_vertica` script creates, create those directories and make sure that dbadmin has ownership of those directories.

Creating the Database

To initially create the HP Vertica database to embed with your customer application, add the following command to the packaging script:

```
$ /opt/vertica/bin/admintools --tool create_db [ options ]
```

Command-Line Options:

Option	Function
`-h` `--help`	Show this help message and exit.

Option	Function
`-s NODES` `--hosts=NODES`	Comma-separated list of hosts to participate in database.
`-d DB` `--database=DB`	Name of database to be created.
`-c CATALOG` `--catalog_path=CATALOG`	[Optional] Path of catalog directory if not using compat21.
`-D DATA` `--data_path=DATA`	[Optional] Path of data directory if not using compat21.
`-p DBPASSWORD` `--password=DBPASSWORD`	[Optional] Database password in single quotes.
`-l LICENSEFILE` `--license=LICENSEFILE`	[Optional] Database license.
`-P POLICY` `--policy=POLICY`	[Optional] Database restart policy.
`--compat21`	Use HP Vertica Analytic Database 2.1 method using node names instead of host names.

The following example creates a database named `mydb`:

```
$ admintools -t create_db
    -s 10.20.100.66,10.20.100.67,10.20.100.68
    -d mydb -c /home/dbadmin/mydb/catalog
    -D /home/dbadmin/mydb/data -l /home/dbadmin/vlicense.dat

Info: no password specified, using none
Distributing changes to cluster.
10.20.100.66 OK [vertica][(6, 1, 2)][20130430][x86_64]
10.20.100.67 OK [vertica][(6, 1, 2)][20130430][x86_64]
10.20.100.68 OK [vertica][(6, 1, 2)][20130430][x86_64]
Checking full connectivity
Creating database mydb
Node Status: v_mydb_node0001: (DOWN)
Node Status: v_mydb_node0001: (INITIALIZING)
Node Status: v_mydb_node0001: (VALIDATING LICENSE)
Node Status: v_mydb_node0001: (UP)
Creating database nodes
Creating node v_mydb_node0002 (host 10.20.100.67)
Creating node v_mydb_node0003 (host 10.20.100.68)
Node Status: v_mydb_node0001: (UP) v_mydb_node0002: (UP)
    v_mydb_node0003: (DOWN)
Node Status: v_mydb_node0001: (UP) v_mydb_node0002: (UP)
    v_mydb_node0003: (UP)
Database mydb created successfully.
```

Configuring the Database

Execute the scripts that you created in Exporting the Optimized Database in this guide that

- Create the physical schema.
- Create the catalog objects.
- Create the logical schema.
- Create projections that have been configured for use with the application.

Starting the Database

To start your HP Vertica database, add a command to your packaging script that uses the `admintools` script with either the `start_db` or `restart_db` option, as described in this section. The `restart_db` option allows you to restart the database at the last good epoch, minimizing any loss of data.

start_db Option

```
$ /opt/vertica/bin/admintools --tool start_db [ options ]
```

Command-Line Options

Option	Purpose
`-h` `--help`	Show this help message and exit.
`-d` *DB* `--database=`*DB*	Name of database to be started.
`-p` *DBPASSWORD* `--password=`*DBPASSWORD*	Database password in single quotes.
`-i` `--noprompts`	Do not stop and wait for user input. (Default: False.)

restart_db Option

```
$ /opt/vertica/bin/admintools --tool restart_db [ options ]
```

Command-Line Options

Option	Purpose
`-h` `--help`	Show this help message and exit.
`-d` *DB* `--database=`*DB*	Name of database to be restarted.
`-e` *EPOCH* `--epoch=`*EPOCH*	Epoch at which the database is to be restarted. If `'last'` is given as the argument, the database restarts from the last good epoch.
`-p` *DBPASSWORD* `--password=`*DBPASSWORD*	Database password in single quotes.
`-i` `--noprompts`	Do not stop and wait for user input. (Default: False.)

Connecting to the Database

To connect to the HP Vertica database that you have started, add the following command to the packaging script:

```
$ /opt/vertica/bin/admintools --tool connect_db [ options ]
```

Command-Line Options

Option	Purpose
`-h` `--help`	Show this help message and exit.
`-d` *DB* `--database=`*DB*	Name of database to connect to.
`-p` *DBPASSWORD* `--password=`*DBPASSWORD*	Database password in single quotes.

Backing Up the Database

Hewlett-Packard strongly recommends backing up the database on a regular basis. This task is especially important for a single-node environment where K-safety = 0.

Hewlett-Packard provides a comprehensive utility, the `vbr.py` Python script, that lets you back up, restore, list backups, and copy your database. You can create full and incremental database backups and snapshots of specific schemas or tables to use with a multi-tenant database. Using `vbr.py`, you can save your data to a variety of locations:

- A local directory on the nodes in the cluster
- One or more hosts outside of the cluster
- A different HP Vertica cluster (effectively cloning your database)

Create regular snapshots of your data by running `vbr.py` from a `cron` job or other task scheduler.

Part 3: On-going Database Maintenance

Once your application is up and running at the customer site, monitor your database performance in the following areas. These steps help you prevent performance issues and down time:

- Monitoring System Resources for Concurrent Use
- Updating Statistics
- Optimizing the Physical Schema
- Optimizing Query Performance
- Database Recovery

Monitoring System Resources for Concurrent Use

If you are running Vertica Analytic Database in a multi-user environment, you may have several processes competing concurrently for system resources. The built-in GENERAL resource pool is preconfigured based on your system's RAM and machine cores. You can customize the GENERAL pool or define new resource pools and configure them for memory usage, concurrency, and query priority.

To monitor use of system resources over time, query the following system tables:

- RESOURCE_ACQUISITIONS lists details about each resource (memory, open file handles, threads) acquired by each request for each resource pool in the system.
- RESOURCE_POOL_DEFAULTS (systab) lists default values for parameters in each internal and user-defined resource pool.
- RESOURCE_POOL_STATUS contains configuration settings of the various resource pools in the system, including internal pools.
- RESOURCE_POOLS contains information about settings that each resource pool was configured with.
- RESOURCE_QUEUES lists information about requests pending for various resource pools.
- RESOURCE_REJECTIONS monitors requests for resources that are rejected by the **Resource Manager**.
- RESOURCE_REJECTION_DETAILS records an entry for each resource request that HP Vertica denies. This information is useful for determining if there are resource space issues and which users/pools encounter problems
- SYSTEM_RESOURCE_USAGE provides history about system resources, such as memory, CPU, network, disk, I/O.

For more information about managing resource pools, see Best Practices for Managing Workload Resources in the Administrator's Guide.

Updating Statistics

The HP Vertica query optimizer relies on up-to-date statistics for tables, schemas, and the database. The statistics allow the optimizer to determine the most efficient plan to execute a query.

The following statistics can affect optimizer decisions:

- Eligible projections to answer the query
- The best order in which to perform joins
- Plans involving different algorithms, such as hash join/merge join or group by hash/group by pipelined operations
- Data distribution algorithms, such as broadcast and resegmentation

Without reasonably accurate statistics, the optimizer could choose a suboptimal projection or a suboptimal join order for a query.

As you update your data over time, the statistics become out of date. Stale statistics can cause the optimizer to choose a less-than-optimal plan for executing a query. The query plan that the EXPLAIN statement creates contains information about statistics.

How often you update statistics depends on how often your data changes. HP Vertica recommends that you schedule statistics analysis on a regular basis. This helps ensure that the optimizer chooses the most efficient plan for executing your queries.

For guidance on updating statistics, see Collecting Database Statistics in the Administrator's Guide.

HP Vertica provides the following functions related to collecting statistics:

- ANALYZE_HISTOGRAM—Collects and aggregates statistics from all nodes that store projections associated with the specified table or column. You can specify what percentage of data to read from disk.
- ANALYZE_STATISTICS—Collects and aggregates statistics from all nodes that store projections associated with the specified table or column.
- DROP_STATISTICS—Removes statistics for the specified table and lets you optionally specify the category of statistics to drop.
- EXPORT_STATISTICS—Exports statistics to an XML file.
- IMPORT_STATISTICS—Imports statistics from an XML file created by EXPORT_STATISTICS.

Optimizing the Physical Schema

After you load data into your database, your first step should always be to run Database Designer. Database Designer creates a physical schema that provides optimal query performance.

The first time you run Database Designer, make sure that Database Designer has representative queries and data on which to base the design. Then create a comprehensive design.

Consider rerunning Database Designer to create an incremental design when:

- The data has changed significantly.
- You use different queries on a regular basis.
- You encounter performance issues with your queries.

For more information, see Creating a Database Design in the Administrator's Guide.

Optimizing Query Performance

When you submit a query to HP Vertica for processing, the HP Vertica query optimizer automatically chooses a set of operations to compute the requested result. These operations together are called a *query plan*. The choice of operations can drastically affect the run-time performance and resource consumption needed to compute the query results. Depending on the properties of the projections defined in your database, the query optimizer can choose faster and more efficient operations to compute the query results.

As your database grows over time, queries could degrade in performance. It is important to monitor the performance of your queries—especially queries that are run frequently or access large data sets.

If you encounter these problems, the first steps are to

- Analyze the statistics and the histogram for your database tables and run Database Designer, supplying queries that are experiencing performance problems.
- Review the QUERY_EVENTS system table. This table identifies whether there are issues with the planning phase of a query.

Database Recovery

Recovery is the process of restoring the database to a fully functional state after one or more nodes in the system has experienced a software- or hardware-related failure. Those failures can include:

- Hardware failure of one or more nodes
- Clean shutdown of the database
- Unclean shutdown of the database, which can be caused by
  - A critical node failed, leaving part of the database's data unavailable.
  - A power failure that causes all nodes to reboot.
  - HP Vertica processes on the nodes exited due to a software or hardware failure.

A failure can cause a node to lose database objects or to miss changes made to the database (INSERTs, UPDATEs, and so on) while offline. The recovery process recovers lost objects and catches up with changes by querying the other nodes.

For detailed information about database recovery after specific types of failure, see Recovering the Database in the Administrator's Guide.

Hardware Maintenance

The HP Vertica 7.0.x Supported Platforms document describes the hardware required to run the HP Vertica database.

For on-going maintenance of your cluster nodes, follow the guidance in Adding or Removing Cluster Nodes in this guide.

Adding or Removing Cluster Nodes

You can scale your HP Vertica cluster up or down to meet the needs of your database. The most common way to scale up your cluster is to add nodes to your database cluster. These nodes can accommodate more data and provide better query performance.

However, if your cluster has more nodes than it needs or if you need to divert hardware for other uses, you can scale down your cluster.

You scale your cluster by adding or removing nodes. You can add or remove nodes without having to shut down or restart the database.

After adding a node or before removing a node, HP Vertica rebalances the data. During rebalancing, HP Vertica moves data around the cluster to populate the new nodes or moves data off nodes that are about to be removed from the database. Data may also be exchanged between nodes that are not being added or removed to maintain robust intelligent K-safety. If HP Vertica determines that the data cannot be rebalanced in a single iteration due to a lack of disk space, then HP Vertica performs the rebalancing in multiple iterations.

For detailed information about managing your cluster, see Managing Nodes in the Administrator's Guide.

Troubleshooting Performance

If your application performance is slow, you need to determine if your application or the HP Vertica software is causing the problems.

First, evaluate your system for performance issues. If you modify your application but are still experiencing issues, evaluate your HP Vertica database for performance issues.

HP Vertica provides many tools to evaluate the performance of your database. The following sections of the Administrator's Guide describe them in detail:

- Analyzing Workloads
- Using Diagnostic Tools
- Understanding Query Plans
- Profiling Database Performance

The Optimizing Query Performance section of the Programmer's Guide gives information about improving the performance of your database queries.

We appreciate your feedback!

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Best Practices for HP Vertica OEM Customers (Vertica Analytic Database 7.0.x)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to vertica-docfeedback@hp.com.