Vertica Database 2.0.5-0

Database Administrator's Guide

Date of Publication: 1/8/2008

Copyright Notice

Vertica Systems, Inc.

Three Dundee Park Drive, Suite 102

Andover, MA 01810-3723

Phone: (978) 475-1070

Fax: (978) 475-6855

E-Mail: info@vertica.com

Web site: http://www.vertica.com
(http://www.vertica.com)

Trademarks

Open Source Software Acknowledgements

Spread

Contents

Technical Support

To submit problem reports, questions, comments, and suggestions, please use the Technical Support page on the Vertica Systems, Inc. Web site:

http://www.vertica.com/support (http://www.vertica.com/support)

You must be a registered user in order to access the page.

Before reporting a problem, please run the Diagnostics Utility described in the Troubleshooting Guide and attach the resulting .zip file.

About the Documentation

Where to Find the Vertica Documentation

Vertica Systems, Inc. recommends that you copy the Vertica documentation from the database server (any cluster host) to a client system on which you can use a browser and/or the Adobe Reader.

Database Server Systems

The Vertica Database Documentation Set is automatically installed in the **/opt/vertica/doc/** directory on all cluster hosts. If you have a browser and/or the Adobe Reader installed on a cluster host, you can access the documentation directly.

`/opt/vertica/doc/HTML/Master/index.htm`
`/opt/vertica/doc/JDBC/index.htm`
`/opt/vertica/doc/PDF/`*`book-name`*`.pdf`

Database Client Systems

To create a copy of the Vertica documentation on a client system, do one of the following:

- Download the documentation package (**.tar.gz** or **.zip**) from the Vertica Systems, Inc. Web site and extract the files to a directory on the client system, using the original pathnames
- Copy the documentation directories from a database server system to a convenient location in your client system. All cross-references within the HTML documentation are relative so there is no location dependency.

`<`*`your-location`*`>HTML/Master/index.htm`
`<`*`your-location`*`>JDBC/index.htm`
`<`*`your-location`*`>PDF/`*`book-name`*`.htm`

World Wide Web

You can read and/or download the Vertica documentation from the **Vertica Systems, Inc. Web site**:

http://www.vertica.com/v-zone/product_documentation http://www.vertica.com/v-zone/product_documentation.

You need a V-Zone login to access the Product Documentation page.

The documentation on the Vertica Systems, Inc. Web site is updated each time a new release is issued. If you are using an older version, refer to the documentation on your database server or client systems.

Reading the HTML Files

The Vertica documentation files are provided in HTML browser format for platform independence. The HTML files only require a browser that can **display frames** properly and has **JavaScript** enabled. The HTML files **do not require a Web (HTTP) server**.

The Vertica documentation has been tested on the following browsers:

- Internet Explorer 7
- FireFox
- Opera
- Safari

Please report any script, image rendering, or text formatting problems to ***Technical Support*** (on page 11).

The Vertica documentation may contain links to Web sites of other companies or organizations that Vertica does not own or control. If any of these links are broken, please inform us.

Printing the PDF Files

The documentation files are supplied in **Adobe Acrobat™ PDF** document format for the purpose of making printed copies as needed. The documents are designed to be printed on standard 8½ x 11 paper using full duplex (two sided printing).

You can open and print any of the PDF documents using the **Adobe Reader**. (You can download the latest version of the free Acrobat Reader from the ***Adobe Web site*** (http://www.adobe.com/products/acrobat/readstep2.html).)

HTML links to the PDF files are provided here for browser access.

- Database Administrator's Guide
- Database Administrator's Guide (Advanced)
- Glossary of Terms
- Installation Guide
- Product Overview
- Quick Start
- Release Notes
- SQL Programmer's Guide
- SQL Reference Manual
- Troubleshooting Guide

Suggested Reading Paths

This section provides a suggested reading path for various types of users. Read the manuals listed under All Users first.

All Users

- Product Overview (basic concepts critical to understanding Vertica)
- Quick Start (step-by-step guide to getting Vertica up and running)
- Glossary of Terms (glossary of terms)

System Administrators

- Installation Guide (platform configuration and software installation)
- Release Notes (release-specific information)
- Troubleshooting Guide (general troubleshooting information)

Database Administrators

- Installation Guide (platform configuration and software installation)
- Database Administrator's Guide (database configuration, loading, security, and maintenance)
- Troubleshooting Guide (general troubleshooting information)

Application Developers

- SQL Programmer's Guide (connecting to a database, queries, transactions, etc.)
- SQL Reference Manual (Vertica-specific language information)
- Troubleshooting Guide (general troubleshooting information)

Where to Find Additional Information

Visit the ***Vertica Systems, Inc. Web site*** (http://www.vertica.com) to keep up to date with:

- Downloads
- Frequently Asked Questions (FAQs)
- Discussion forums
- News, tips, and techniques

Typographical Conventions

It is important to understand the terms and typographical conventions used in this document.

General Convention	Description
colored bold text	introduces new terms defined either in the text, the glossary, or both.
normal italic text	indicates emphasis and the titles of other documents.
UPPERCASE TEXT	indicates the name of an SQL command or keyword.
`monospace text`	indicates literal interactive or programmatic input/output.
`italic monospace text`	indicates user-supplied information in interactive or programmatic input/output.
`bold monospace text`	indicates literal interactive user input
↵	indicates the Return/Enter key; implicit on all user input that includes text

SQL Syntax Convention	Description
indentation	is an attempt to maximize readability; SQL is a free-form language.
backslash \	continuation character used to indicate text that is too long to fit on a single line.
braces { }	indicate required items.
brackets [ ]	indicate optional items.
ellipses ...	indicate an optional sequence of similar items.
vertical ellipses ⋮	indicate an optional sequence of similar items or that part of the text has been omitted for readability.
vertical line |	within braces or brackets, indicates a choice .

Preface

This document describes how to set up and maintain a Vertica database. You may find that these tasks are extremely simple in Vertica compared to other database management systems.

Prerequisites

This document assumes that you have already:

- become familiar with the concepts discussed in the Product Overview.
- performed the procedures described in the Installation Guide
  - constructed a hardware platform
  - installed Linux
  - installed Vertica (configured a cluster of hosts)
- followed the tutorial in the Quick Start to get experience with the process of setting up an example database

Audience

This document is intended for anyone with responsibility for configuring, loading, securing, and maintaining a Vertica database.

For More Information

There is a great deal of published literature available about dimensional modeling and data warehouse design in general.

Overview

This document describes the functions performed by a Vertica database administrator (DBA). Perform these tasks using only the **dedicated database administrator account** that was created when you installed <DBMS>. The examples in this documentation set assume that the administrative account name is **dbadmin**.

- To perform certain cluster configuration and administration tasks, the DBA (users of the administrative account) must be able to supply the **root password** for those hosts. If this requirement conflicts with your organization's security policies, these functions must be performed by your IT staff.
- If you perform administrative functions using a different account, Vertica encounters file ownership problems.
- If you share the administrative account password, make sure that only one user runs the Administration Tools at any time. Otherwise, automatic configuration propagation does not work correctly.

Configuring the Database

This section describes:

- the Vertica Database ***Configuration Procedure*** (page 23)
- detailed information about designing a Logical Schema
- detailed information about creating the Physical Schema

Vertica strongly recommends that you follow the tutorial procedure in the Quick Start to get hands-on experience with the process of configuring a database before you begin this section.

Configuration Procedure

This section describes the tasks required to set up a Vertica database.

This section assumes that you have obtained a valid license key file, installed the Vertica rpm package and run the installation script as described in the Installation Guide.

To complete configuration procedure, you will use the:

- Administration Tools (see the ***Administration Tools Reference*** (page 75) for details)
- vsql interactive interface. If you are unfamiliar with Dialog-based user interfaces, read ***Using the Graphical User Interface*** (page 70) before you begin.

Vertica strongly recommends that you follow the tutorial procedure in the Quick Start to get hands-on experience with the process of configuring a database before you begin this section.

Prepare Disk Storage Locations

Prepare the disk locations that will contain the Vertica catalog and data files (physical schema) on drives **local to each host** in the cluster. You can use a single directory to contain both the catalog and data files or you can use separate directories. If using separate directories, they can be on different drives.

The catalog and data directory pathnames must be identical on each host in the cluster and the directories must be owned by the Database Administrator.

If you place catalog and data files on a shared drive for testing purpose, each host must have its own catalog and data locations. In other words, hosts cannot share catalog or data locations.

Specifying Disk Storage at Installation Time

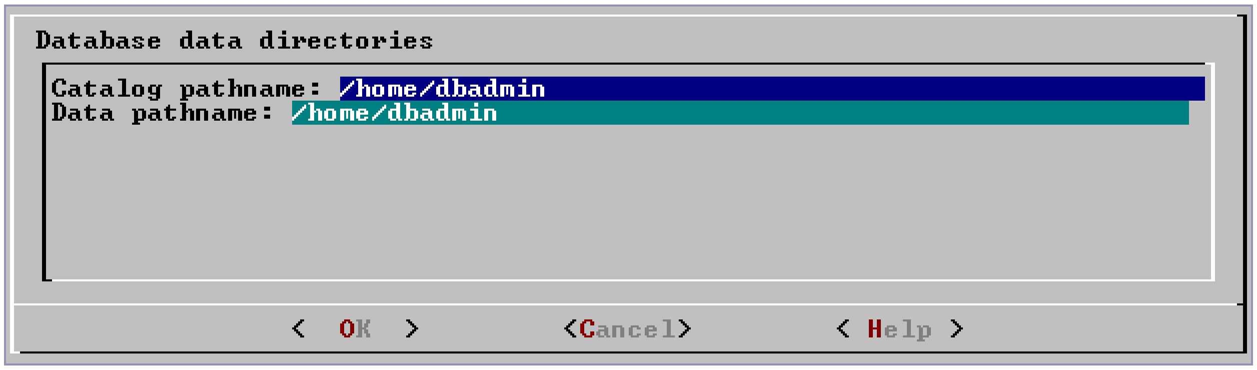
When you installed Vertica, the `install_vertica` script's `data_directory` parameter allowed you to specify a directory to contain database data and catalog files. The default is the Database Administrator's default home directory:

```
/home/dbadmin
```

There is no requirement that you actually use this directory. It is created for your convenience. However, if you choose a different location, make sure that the location exists on each host in the cluster.

Specifying Disk Storage at Database Creation Time

When you invoke the ***Create Database*** (page 82) command in the Administration Tools, the following dialog allows you to specify the actual catalog and data locations. These locations much exist on each host in the cluster and must be owned by the database administrator.



```
Database data directories
Catalog pathname: /home/dbadmin
Data pathname: /home/dbadmin
```

When you click OK, Vertica automatically creates the following subdirectories:

*catalog-pathname/database-name/node-name*_catalog/

*data-pathname/database-name/node-name*_data/

Catalog and data pathnames must contain only alphanumeric characters and cannot have leading space characters. Failure to comply with these restrictions may result in database creation failure.

Prepare the Logical Schema Script

Designing the Logical Schema

Designing the logical schema for a Vertica database is essentially an exercise in dimensional modeling, the industry standard for data warehousing. See ***The Logical Schema (Tables and Constraints)*** (page 35) for details.

Creating the Logical Schema Script

To create your logical schema, you must prepare an SQL script (plain text file, typically with an extension of .sql) that:

1. creates the tables in your database using the CREATE TABLE command
2. defines the necessary constraints and ALTER TABLE command

You can generate a script file using:

- a schema designer application
- a schema extracted from an existing database
- a text editor
- one of the example database *example-name*`_define_schema.sql` scripts as a template (see the example database directories in /opt/vertica/doc/)

In your script file, make sure that:

- each statement ends with a semicolon
- you are using only data types supported by Vertica (see the SQL Reference Manual for details)

Once you have created a database, you can test your schema script by executing it. If you encounter errors, simply drop all tables, correct the errors, and execute it again.

Prepare Actual Data Files

Prepare the actual data files for your initial ***bulk load*** (page 60). Vertica Systems, Inc. recommends that you load your fact table data using multiple files of roughly 10GB to 50GB each. This size provides several advantages:

- You can use one of the fact table data files as a ***sample data file*** (page 26).
- You can load just enough fact table data to ***Test a Partially Loaded Fact Table*** (page 32) before loading the remainder.
- If a single fact table load fails and rolls back, you do not lose an excessive amount of time.

See ***Bulk Loading*** (page 60) for more information.

How to Name Data Files

Name each data file to **match the corresponding table** in the logical schema. Case does not matter. Use the extension `.tbl` or whatever you prefer. For example, if a table is named Stock_Dimension, name the corresponding data file `stock_dimension.tbl`. When using multiple data files, append *_nnn* (where nnn is a positive integer in the range 001 to 999) to the filename. For example, `stock_dimension.tbl_001`, `stock_dimension.tbl_002`, etc.

Prepare Sample Data Files

These steps involve preparing sample data files for the Database Designer to use in producing the Physical Schema design. If your actual data files are available at this point, you can use a subset of the full data set as sample data. Otherwise, you can use data generated by a commercial data generator application.

How Much Data to Prepare

The Database Designer can read up to one hundred thousand (100K) rows of data (1000 uniform seeks reading 100 rows at a time) of sample fact table data. If there are 100K or fewer rows in the sample data file, all rows are sampled. If the data file is larger, the sampling is uniformly distributed to avoid skew problems. (An internal cycle in the data may cause skew problems but is unlikely.)

How to Name Data Files

Name each data file to **match the corresponding table** in the logical schema. Case does not matter. Use the extension `.tbl` or whatever you prefer. For example, if a table is named Stock_Dimension, name the corresponding data file `stock_dimension.tbl`. When using multiple data files, append *_nnn* (where nnn is a positive integer in the range 001 to 999) to the filename. For example, `stock_dimension.tbl_001`, `stock_dimension.tbl_002`, etc.

Prepare Load Scripts

Prepare SQL scripts that use the COPY...DIRECT or LCOPY...DIRECT statement to load data directly into physical storage. You will need scripts that load the:

- dimension tables
- partial fact table (10GB to 50GB per node)

the remainder of the fact table data See the following sections for details:

- ***Bulk Loading*** (page 60)
- ***Using Load Scripts*** (page 63)
- ***Using Parallel Load Streams*** (page 64)

You can use the load scripts included in the example databases in the Quick Start as templates.

Prepare a Sample Query Script

Prepare an SQL script file containing a series of queries (SELECT statements) that represent, as accurately as possible, your actual database workload. These queries are used by the Database Designer to create projections.

Make sure that each query ends with a semicolon.

How Many Queries

Provide as many sample queries as possible (up to 30,000). The Database Designer requires a reasonably accurate representation of your query workload in to produce an optimal physical schema design.

Types of Queries

Make sure to include your most frequently executed, resource-intensive, and time-critical queries.

Where to Get Queries

As source material, you can use:

- a log of the queries running on an existing database
- queries generated by a business analytics tool such as a report generator

Create the Database

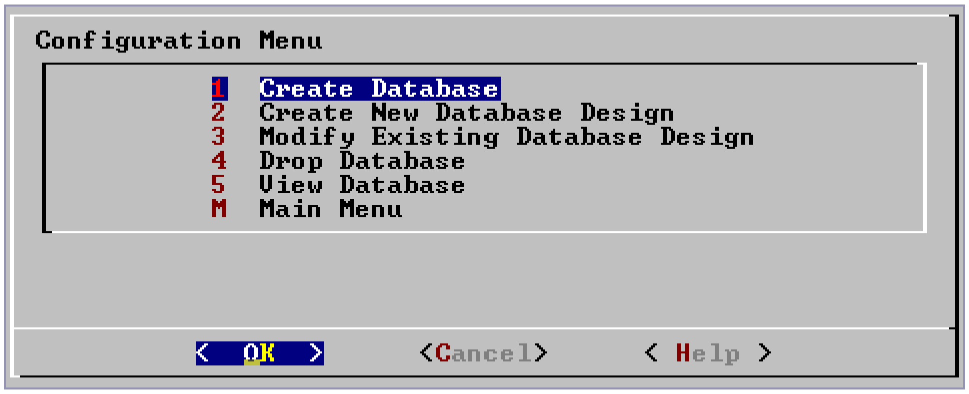
1. **Run the Administration Tools**

   Start up the Administration Tools on one of the database hosts.

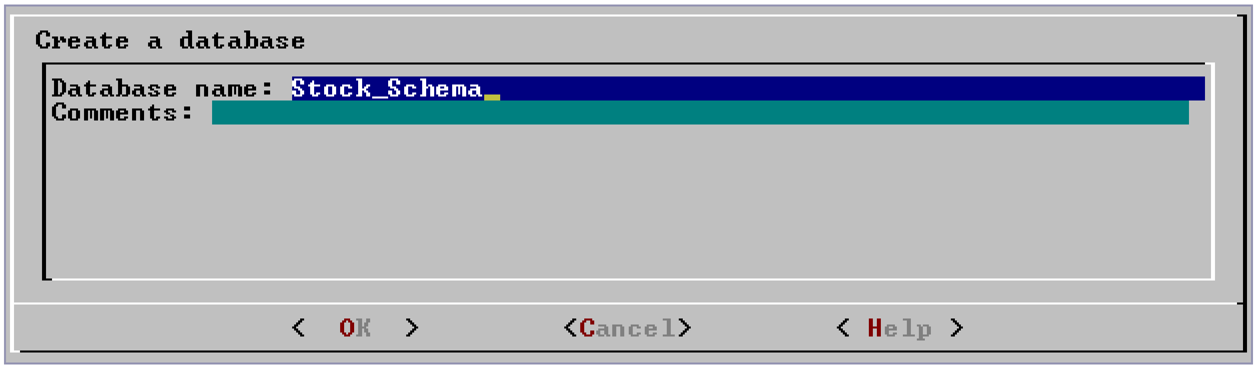
   ```
   $ /opt/vertica/bin/adminTools
   ```

   If you are using a remote terminal application such as PuTTY or a Cygwin bash shell, see ***Notes for Remote Terminal Users*** (on page 71).

2. **Accept the license agreement** (once only).
3. Specify the **location of your license file** (once only). See ***Managing Your License Key*** (page 49) for more information.
4. On the **Configuration menu**, select **Create Database**.

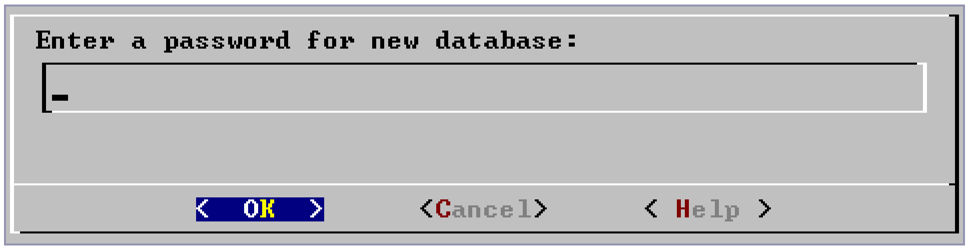


5. **Enter the name** of the database and an optional comment.



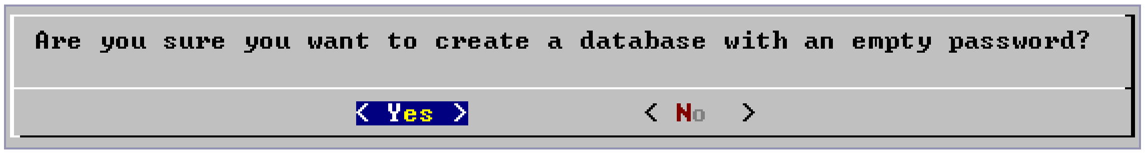
```
Create a database
Database name: Stock_Schema_
Comments:
```

6. **Enter the password**.



```
Enter a password for new database:
```

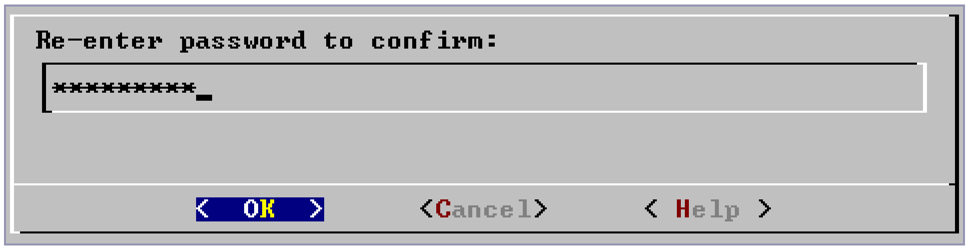
If you **do not enter a password**, the following dialog appears.



```
Are you sure you want to create a database with an empty password?
```

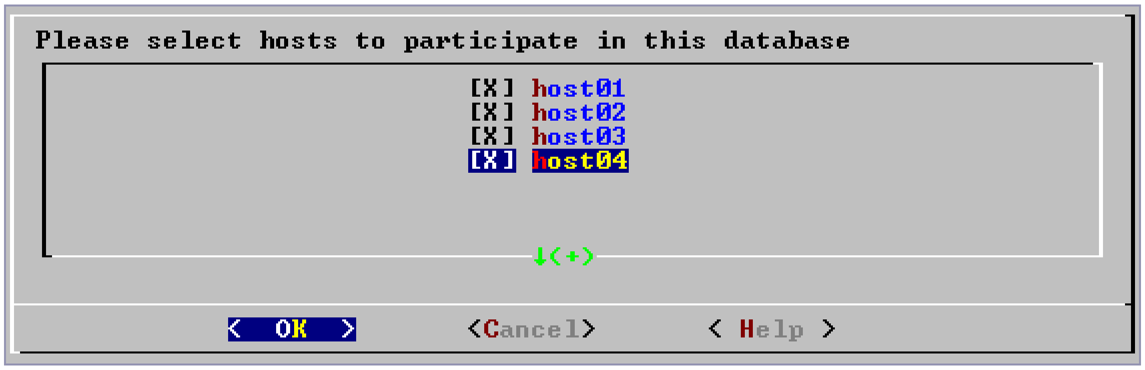
WARNING: If you do not enter a password at this point, the database is permanently set to trust authentication (no password required), in which ALTER USER cannot be used to change the superuser password. Unless the database is for evaluation or academic purposes, Vertica Systems, Inc. strongly recommends that you enter a superuser password.

7. If you entered a password **enter the password again**.



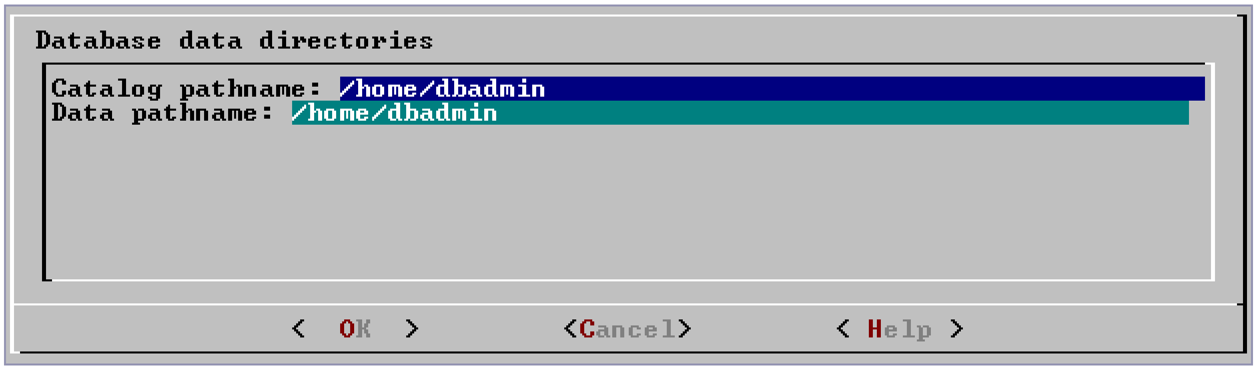
```
Re-enter password to confirm:
**********_
```

8. **Select the hosts** to include in the database. The hosts in this list are the ones that were specified at installation time (`install_vertica -s`).



```
Please select hosts to participate in this database

                    [X] host01
                    [X] host02
                    [X] host03
                    [X] host04

                     ↓(+)

      <  OK  >        <Cancel>       < Help >
```

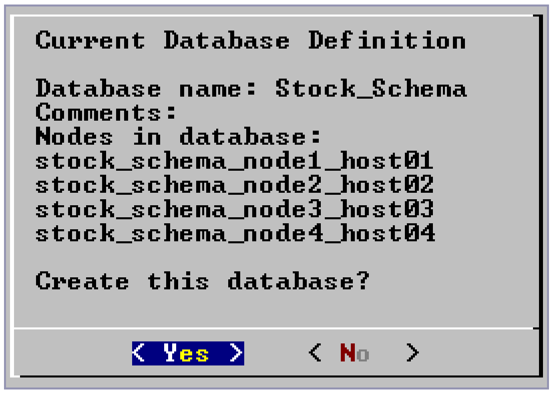
9. Specify the directories in which to store the data and catalog files.



```
Database data directories

Catalog pathname: /home/dbadmin
Data pathname: /home/dbadmin

        <  OK  >        <Cancel>        < Help >
```

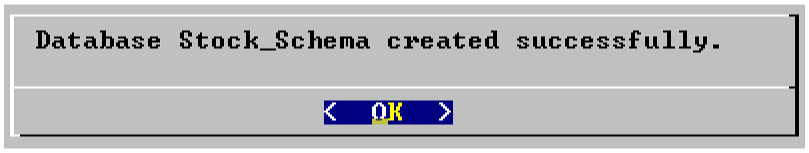
Catalog and data pathnames must contain only alphanumeric characters and cannot have leading space characters. Failure to comply with these restrictions may result in database creation failure.

10. Check the current database definition for correctness.



```
Current Database Definition

Database name: Stock_Schema
Comments:
Nodes in database:
stock_schema_node1_host01
stock_schema_node2_host02
stock_schema_node3_host03
stock_schema_node4_host04

Create this database?

        < Yes >    <  No  >
```

11. This message indicates that you have successfully created a database.



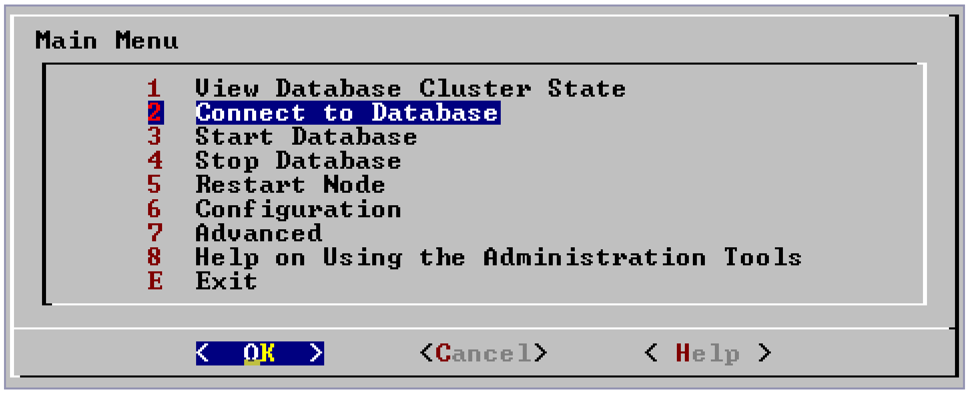
```
Database Stock_Schema created successfully.

                  < OK >
```

If you get an error message, see:

- ***Startup Problems*** (page 102)
- Node Failures

Create the Logical Schema

1. **Connect to the Database**



```
Main Menu

          1  View Database Cluster State
          2  Connect to Database
          3  Start Database
          4  Stop Database
          5  Restart Node
          6  Configuration
          7  Advanced
          8  Help on Using the Administration Tools
          E  Exit

       < OK >        <Cancel>       < Help >
```

Connect to the database. See ***Connect to Database*** (page 76) for details.

```
Welcome to the vsql, Vertica_Database v2.0.5-0 interactive terminal.
Type:  \h for help with SQL commands
       \? for help with vsql commands
       \g or terminate with semicolon to execute query
       \q to quit

=>
```

2. **Execute the Logical Schema Script**

   Using the \i meta-command in vsql to execute the SQL ***logical schema script*** (page 24) that you prepared earlier. For example:

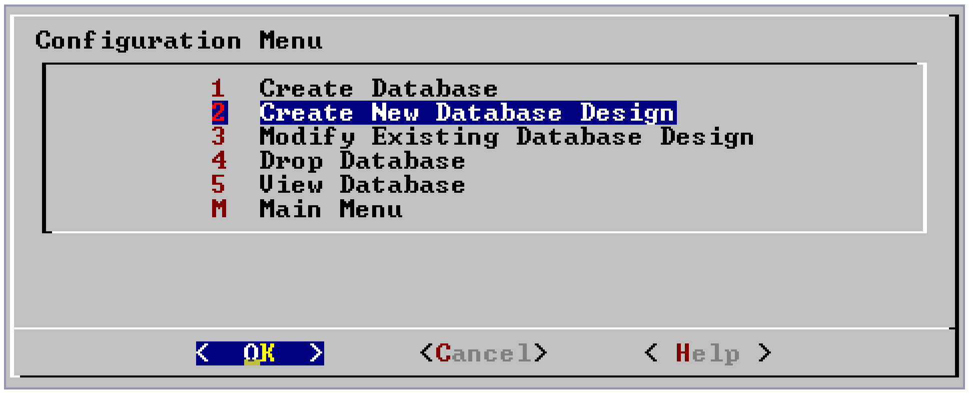
3. **Disconnect From the Database**

   Use the \q meta-command in vsql to return to the Administration Tools.

Run the Database Designer

1. **Run the Database Designer**

Run the Database Designer to create an SQL projection script. Use the ***sample data files*** (page 26) and ***sample query script*** (page 26) that you prepared earlier.



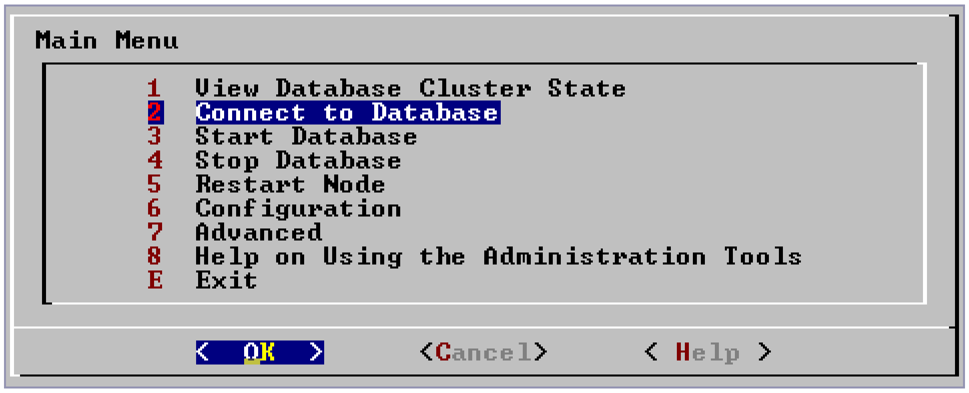
```
Configuration Menu

        1   Create Database
        2   Create New Database Design
        3   Modify Existing Database Design
        4   Drop Database
        5   View Database
        M   Main Menu

      <  OK  >      <Cancel>      < Help >
```

2. **Check designer.log** and **designer_error.msg** to make sure that no problems occurred.
3. **Examine the projection script** created by the Database Designer.

See ***Using the Database Designer*** (page 45) and the ***Create New Database Design*** (page 84) command for more information.

Create the Physical Schema

1. **Connect to the Database**



```
Main Menu

     1   View Database Cluster State
     2   Connect to Database
     3   Start Database
     4   Stop Database
     5   Restart Node
     6   Configuration
     7   Advanced
     8   Help on Using the Administration Tools
     E   Exit

      <  OK  >      <Cancel>      < Help >
```

Connect to the database. See ***Connect to Database*** (page 76) for details.

```
Welcome to the vsql, Vertica_Database v2.0.5-0 interactive terminal.
Type:  \h for help with SQL commands
       \? for help with vsql commands
       \g or terminate with semicolon to execute query
       \q to quit

=>
```

2. **Execute the Database Designer Script**

Using the \i meta-command in vsql, execute the SQL projection script produced by the Database Designer. This scripts creates the projections that make up your Physical Schema.

You can add and drop projections after data is loaded. However, be aware that adding a projection begins a refresh operation that copies data into the new projection from other projections. The refresh operation may take some time to complete. You can monitor the refresh operation by examining the log files.

Test the Empty Database

1. Test for Sufficient Projections

Using the \i meta-command in vsql, execute the ***sample query script*** (page 26) that you prepared earlier. No data has been loaded so the queries return empty result sets. If you get an error about insufficient projections, check the Troubleshooting Guide for possible reasons.

2. Check the Design for K-Safety

Using the \i meta-command in vsql, execute the following statement:

```
=> SELECT MARK_DESIGN_KSAFE(1);
```

The statement checks the physical schema design (projections) and, if the design meets all requirements, enables recovery. For more information, see SELECT MARK_DESIGN_KSAFE in the SQL Reference Manual).

Test a Partial Data Load

1. Load Dimension Tables

Load the dimension table data files using the SQL ***load scripts*** (page 26) and ***actual data files*** (page 25) you prepared earlier.

2. Partially Load the Fact Table

Load 10GB to 50GB of fact table data using the SQL ***load scripts*** (page 26) and ***actual data files*** (page 25) that you prepared earlier. You will complete the fact table bulk load in the next section.

3. Monitor System Resource Usage

In separate Linux shells, run the `top` and `iostat` utilities as described in ***Monitoring System Resource Usage*** (page 56) and watch them while your queries are running.

4. Check Query Execution Times

1. Use the vsql \timing meta-command to enable the display of query execution time in milliseconds.
2. Execute the SQL sample query script that you prepared earlier.
3. Execute several ad-hoc queries.

If you discover that any of the ad-hoc queries are unacceptably slow, you can drop all tables, append the queries to the sample query script, and start over. The Database Designer may create additional projections based on those queries in order to improve performance.

At this point Vertica may return errors indicating a primary key or foreign key constraint violation. If that happens, drop all tables, correct the error, and start over. Vertica generally does not actively enforce constraints for performance reasons but detects some constraint violations while performing a join.

5. **Check Projection Usage**

   In addition to checking query execution times, you can use the EXPLAIN command on certain queries to determine whether or not they are using the expected projections. See the SQL Reference Manual for details.

Complete the Data Load

1. **Monitor System Resource Usage**

   Continue to run the `top`, `free`, and `df` utilities and watch them while your load scripts are running (as described in ***Monitoring System Resource Usage*** (page 56)). You can do this on any or all nodes in the cluster. Make sure that the system is not swapping excessively (watch `kswapd` in `top`) or running out of swap space (watch for a large amount of used swap space in `free`).

2. **Complete the Fact Table Load**

   Execute the remainder of the SQL fact table load scripts.

Set Up Security

In the process of creating a database, you have the option of specifying a Database Superuser password, which permanently enables password authentication for the database. If you choose not to supply a password, the database is permanently set to trust authentication (no password required). This is a one-time-only decision. You cannot enable or disable password authentication later.

1. **Create Database Users**

   This task involves using the CREATE USER command to create additional database user accounts.

2. **Assign Privileges**

   This task involves using the GRANT (Schema) and GRANT (Table) commands to grant appropriate access privileges to users.

In the database with trust authentication, the GRANT and REVOKE statements work as expected. You cannot, however, depend on them for security because anyone can connect as the Database Superuser without supplying a password.

Notes

- Newly-created users do not have access to schema PUBLIC by default. Make sure to **GRANT USAGE ON SCHEMA PUBLIC** to all users you create.

Set Up Incremental Loads

This task involves using the COPY command and/or the Data Manipulation Language commands INSERT, UPDATE, and DELETE as described in Using Data Manipulation Language.

Vertica recommends that you use a series of COPY commands to load 10k to 100k tuples per transaction. You can load data into the WOS by using COPY without the DIRECT keyword. This reduces disk usage while allowing you to run queries using AT EPOCH LATEST.

The Logical Schema (Tables and Constraints)

A logical schema consists of a set of tables and referential integrity constraints in a Vertica database. The objects in the logical schema are visible to SQL users. The logical schema does not include projections, which make up the physical schema.

Designing a logical schema for Vertica database is essentially an exercise in dimensional modeling, the universal standard for data warehousing. This document explains only the most basic concepts of dimensional modeling. If you are converting a normalized schema to a data warehouse schema, contact ***Technical Support*** (page 11) for a consultation or refer to the published literature on the topic.

To implement your logical schema design, write an SQL script (plain text file) that creates the tables in your database and defines the required referential integrity constraints. You will use vsql to execute this script to create the actual logical schema of your database.

Data Warehouse Schema Types

The tables in a Vertica database are assumed to obey a ***star schema*** (page 36) or ***snowflake schema*** (page 37) design, which are typical schema designs used in data warehouses. These designs produce excellent performance on any Vertica system platform. Each design is discussed in a separate section below.

If your full schema cannot be described as a star or snowflake, consider reorganizing the schema into one or more stars and/or snowflakes; a single Vertica database can hold multiple star and/or snowflake schemas that are queried separately and the result tables joined in the application program. (Multiple schemas may be included in a future release.)

If you have a denormalized single-table schema, you can consider it to be the fact table in a star schema. If you do this, however:

- Avoid using SELECT * queries, which in this case defeat the advantages of using a column store.
- Use the LIMIT Clause to avoid very large query result sets.

Porting an Existing Schema

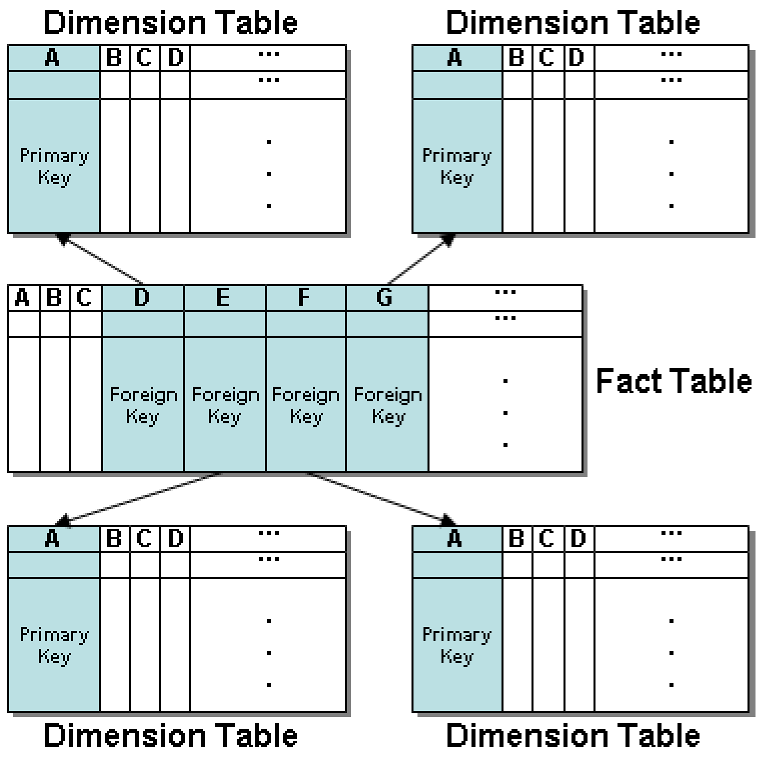
You may need to reconstruct the CREATE TABLE and CREATE VIEW statements, etc., that define an existing schema. An ETL tool can do this for you or your source database system may have a way to generate DDL. For example:

- DB2 has the db2look tool.
- Oracle has the DBMS_METADATA package.

Keep in mind that these tools (especially from Oracle) tend to use proprietary data types and additional storage clauses, so some edits will be needed.

Star Schema

The star schema (sometimes called a star join schema) is the simplest data warehouse schema. In a star schema design there is a central fact table with a large number of tuples, optionally surrounded by a collection of dimension tables, each with a lesser number of tuples. Every dimension table participates in a 1::n join with the fact table. In other words, the primary key of a dimension table is a foreign key in the fact table.



A fact table is a table that represents quantitative or factual data. For example, in a business, a fact table might represent orders.

A fact table is often located at the center of a star schema or snowflake schema. It typically has a large number of tuples and is surrounded by a collection of dimension tables, each with a lesser number of tuples. The fact table participates in a join with every dimension table. In other words, a fact table can contain data but generally contains many foreign keys, each of which matches the primary key of each in a dimension table.

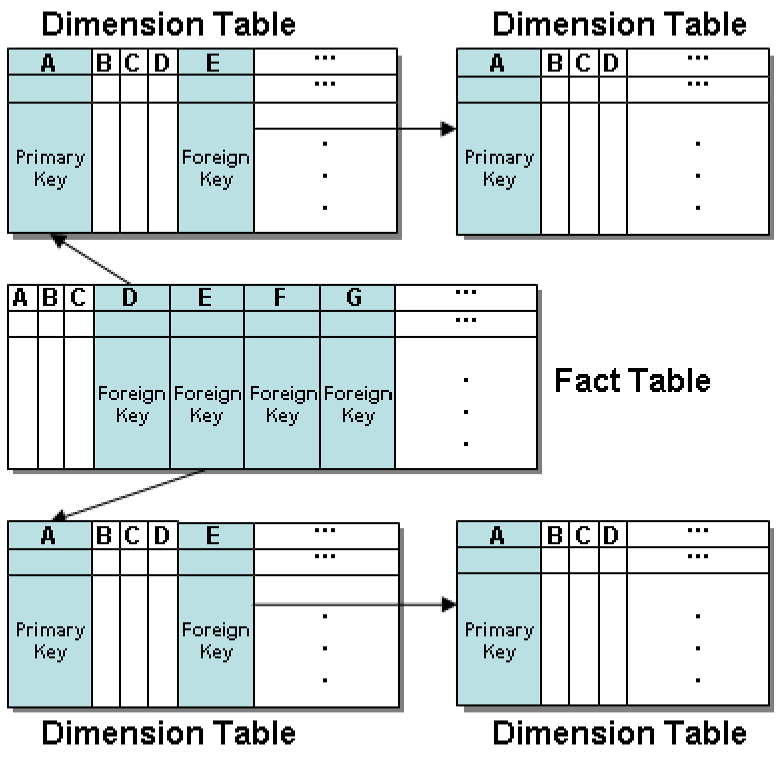
A dimension table (sometimes called a lookup or reference table) is one of a set of companion tables to a fact table in a star schema. It contains the actual data corresponding to the foreign keys in a fact table. For example, a business might use a dimension table to contain item codes and descriptions.

Dimension tables can be connected to other dimension tables to form a hierarchy of dimensions in a snowflake schema.

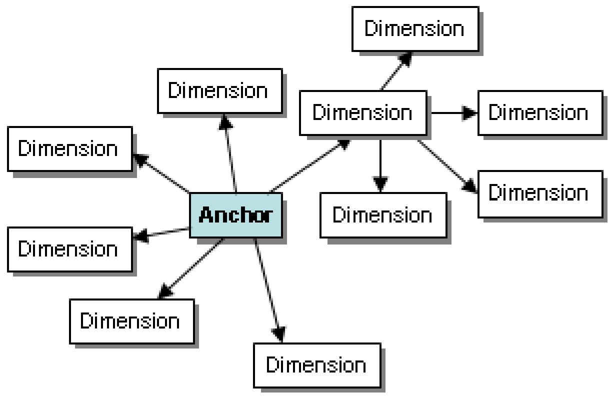
The Retail Sales Example Database in the Quick Start is an example of a star schema.

Snowflake Schema

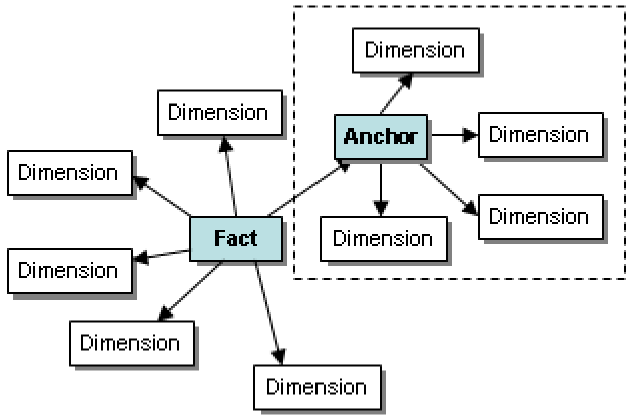
A snowflake schema is the same as a star schema except that a dimension table can be normalized (hierarchically decomposed) into additional dimension tables. Every dimension table participates in a 1::n join with the fact table or another dimension table. In other words, the primary key of a dimension table is a foreign key in a fact table or another dimension table.



In a snowflake schema, the fact table used in a query or projection is called the Anchor Table. A query or projection based on that anchor table can include columns from any table in the schema, as shown below.



However, you also can use a dimension table as an anchor table. There is one restriction: a query or projection based on that table can include only its own columns and those of its dimension tables. It cannot include another fact table. In the diagram below, a query or projection based on the indicated anchor table is limited to the tables inside the box.



Creating Tables

Create the tables using the CREATE TABLE command. The example databases described in the Quick Start include sample SQL scripts that demonstrate this procedure. For example:

```
CREATE TABLE Product_Dimension (
  Product_Key              INTEGER NOT NULL,
  Product_Description      VARCHAR(128),
  SKU_Number               CHAR(32),
  Category_Description     CHAR(32),
  Department_Description   CHAR(32),
  Package_Type_Description CHAR(32),
  Package_Size              CHAR(32),
  Fat_Content               INTEGER,
  Diet_Type                 CHAR(32),
  Weight                    INTEGER,
  Weight_Units_Of_Measure   CHAR(32),
  Shelf_Width               INTEGER,
  Shelf_Height              INTEGER,
  Shelf_Depth               INTEGER);
```

Adding Join Constraints

In order to join schema tables, you must declare primary key and foreign constraints that link the tables:

- Each dimension table must have a primary key constraint.
- The fact table must contain a foreign key constraint for each foreign key column that it contains.

These constraints are required by Vertica but are *not* enforced[1] when performing INSERT, UPDATE, DELETE, or COPY operations. This is analogous to the RELY constraint in Oracle, which provides a mechanism for asserting that a given constraint is believed to be true for the purposes of query optimization and performance.

You can specify constraints on single columns in a CREATE TABLE statement. For example:

```
CREATE TABLE DIM1 (C1 INTEGER CONSTRAINT DIM1PK PRIMARY KEY, C2 INTEGER, ... )
```

Because a constraint name is optional, you can write the example as:

```
CREATE TABLE DIM1 (C1 INTEGER PRIMARY KEY, C2 INTEGER, ... )
```

The matching foreign key constraint would look like this:

```
CREATE TABLE FACT1 (C1 INTEGER FOREIGN KEY REFERENCES DIM1(C1), C2 INTEGER ...
)
```

Adding Multi-Column Constraints

To specify multi-column (compound) keys, you must use an ALTER TABLE statement in addition to CREATE TABLE. For example:

```
CREATE TABLE DIM1 (C1 INTEGER, C2 INTEGER, ... )
ALTER TABLE ADD CONSTRAINT DIM1PK PRIMARY KEY (C1, C2);
```

The matching foreign key constraint would look like this:

```
CREATE TABLE FACT1 (C1 INTEGER, C2 INTEGER ... )
ALTER TABLE FACT1 ADD CONSTRAINT FACT1FK FOREIGN KEY REFERENCES DIM1(C1,C2)
... )
```

Vertica recommends that you use only named table constraints.

Footnotes

[1] There is one instance in which Vertica detects violations of referential integrity constraints and/or primary key uniqueness: using INSERT or COPY to load a tuple into the anchor table of a join projection. Do not depend on this checking to detect consistency problems when loading data.

Modifying Tables

You can use the ALTER TABLE statement to:

- add new columns to tables
- add table constraints

drop named table constraints Adding a new column to a table:

- automatically adds the new column with a unique column name to all superprojections of the table
- populates the column according to the column-constraint (DEFAULT for example).
- does not affect the K-Safety of the physical schema design.

The Physical Schema (Projections)

A physical schema consists of a set of projections used to store data on disk. The projections in the physical schema are based on the objects in the Logical Schema.

A projection is a special case of a materialized view that provides physical storage for data. A projection can contain some or all of the columns of one or more tables. A projection that contains all of the columns of a table is called a superprojection. A projection that joins one or more tables is called a pre-join projection. Most projections are used for ad-hoc query processing and K-safety but it is possible to have query-specific projections.

This section is primarily about using the Database Designer to create a physical schema design: a set of projections that provides K-Safety.

Using the Database Designer

Vertica Systems, Inc. strongly recommends that you use the Database Designer to implement your physical schema. The Database Designer generates an SQL script containing CREATE PROJECTION statements that create:

- fact table and dimension table superprojections needed to ensure K-Safety
- pre-join projections for optimal query performance

The Database Designer's user interface is part of the Administration Tools, specifically **Configuration Menu** > ***Create New Database Design*** (on page 84). The interactive dialog is described in ***Database Designer Dialog*** (on page 86). To use the Database Designer from the command line, see Writing Database Designer Scripts in the Database Administrator's Guide (Advanced).

Database Designer produces the following output files:

- **designer_error.msg**
- **designer.log**
- the **design script** which is named:

  `database-name_design_basic.sql`

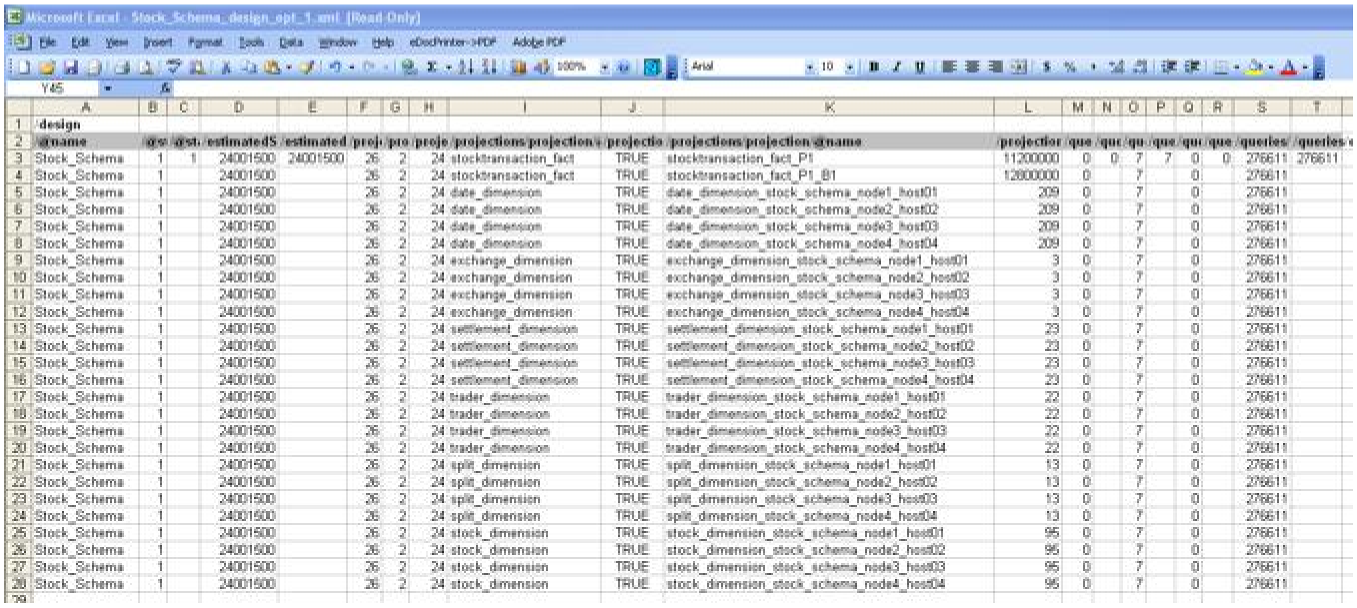
  or

  `database-name_design_opt_n.sql`

  where *n* is the number of optimization stages used in the design.

  If you run the Database Designer again, the old design file is renamed by appending a timestamp extension.

- a matching **XML file** that you can open using any XML-compliant utility. For example, using a spreadsheet:


Microsoft Excel - Stock_Schema_design_opt_1.xml [Read-Only]
File Edit View Insert Format Tools Data Window Help eDocPrinter->PDF Adobe PDF
100%
Arial 10
Y45

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	/design																			
2	/@name	/@s	/@st	/estimatedS	/estimated	/proj	/pro	/proje	/projections/projection	/projectio	/projections/projection/@name	/projectior	/que	/quc	/qu	/que	/qu	/que	/queries/	/queries/
3	Stock_Schema	1	1	24001500	24001500	26	2	24	stocktransaction_fact	TRUE	stocktransaction_fact_P1	11200000	0	0	7	7	0	0	276611	276611
4	Stock_Schema	1		24001500		26	2	24	stocktransaction_fact	TRUE	stocktransaction_fact_P1_B1	12800000	0		7		0		276611	
5	Stock_Schema	1		24001500		26	2	24	date_dimension	TRUE	date_dimension_stock_schema_node1_host01	209	0		7		0		276611	
6	Stock_Schema	1		24001500		26	2	24	date_dimension	TRUE	date_dimension_stock_schema_node2_host02	209	0		7		0		276611	
7	Stock_Schema	1		24001500		26	2	24	date_dimension	TRUE	date_dimension_stock_schema_node3_host03	209	0		7		0		276611	
8	Stock_Schema	1		24001500		26	2	24	date_dimension	TRUE	date_dimension_stock_schema_node4_host04	209	0		7		0		276611	
9	Stock_Schema	1		24001500		26	2	24	exchange_dimension	TRUE	exchange_dimension_stock_schema_node1_host01	3	0		7		0		276611	
10	Stock_Schema	1		24001500		26	2	24	exchange_dimension	TRUE	exchange_dimension_stock_schema_node2_host02	3	0		7		0		276611	
11	Stock_Schema	1		24001500		26	2	24	exchange_dimension	TRUE	exchange_dimension_stock_schema_node3_host03	3	0		7		0		276611	
12	Stock_Schema	1		24001500		26	2	24	exchange_dimension	TRUE	exchange_dimension_stock_schema_node4_host04	3	0		7		0		276611	
13	Stock_Schema	1		24001500		26	2	24	settlement_dimension	TRUE	settlement_dimension_stock_schema_node1_host01	23	0		7		0		276611	
14	Stock_Schema	1		24001500		26	2	24	settlement_dimension	TRUE	settlement_dimension_stock_schema_node2_host02	23	0		7		0		276611	
15	Stock_Schema	1		24001500		26	2	24	settlement_dimension	TRUE	settlement_dimension_stock_schema_node3_host03	23	0		7		0		276611	
16	Stock_Schema	1		24001500		26	2	24	settlement_dimension	TRUE	settlement_dimension_stock_schema_node4_host04	23	0		7		0		276611	
17	Stock_Schema	1		24001500		26	2	24	trader_dimension	TRUE	trader_dimension_stock_schema_node1_host01	22	0		7		0		276611	
18	Stock_Schema	1		24001500		26	2	24	trader_dimension	TRUE	trader_dimension_stock_schema_node2_host02	22	0		7		0		276611	
19	Stock_Schema	1		24001500		26	2	24	trader_dimension	TRUE	trader_dimension_stock_schema_node3_host03	22	0		7		0		276611	
20	Stock_Schema	1		24001500		26	2	24	trader_dimension	TRUE	trader_dimension_stock_schema_node4_host04	22	0		7		0		276611	
21	Stock_Schema	1		24001500		26	2	24	split_dimension	TRUE	split_dimension_stock_schema_node1_host01	13	0		7		0		276611	
22	Stock_Schema	1		24001500		26	2	24	split_dimension	TRUE	split_dimension_stock_schema_node2_host02	13	0		7		0		276611	
23	Stock_Schema	1		24001500		26	2	24	split_dimension	TRUE	split_dimension_stock_schema_node3_host03	13	0		7		0		276611	
24	Stock_Schema	1		24001500		26	2	24	split_dimension	TRUE	split_dimension_stock_schema_node4_host04	13	0		7		0		276611	
25	Stock_Schema	1		24001500		26	2	24	stock_dimension	TRUE	stock_dimension_stock_schema_node1_host01	95	0		7		0		276611	
26	Stock_Schema	1		24001500		26	2	24	stock_dimension	TRUE	stock_dimension_stock_schema_node2_host02	96	0		7		0		276611	
27	Stock_Schema	1		24001500		26	2	24	stock_dimension	TRUE	stock_dimension_stock_schema_node3_host03	95	0		7		0		276611	
28	Stock_Schema	1		24001500		26	2	24	stock_dimension	TRUE	stock_dimension_stock_schema_node4_host04	96	0		7		0		276611	
29																				

Invalid Logical Schema

The physical schema produced by the Database Designer is only as good as the logical schema you provide. For example, a logical schema that is:

- not a ***Star Schema*** (page 36) or a ***Snowflake Schema*** (page 37) cannot produce a valid physical schema
- missing ***primary key/foreign key constraints*** (page 40) cannot produce segmented fact table superprojections

Insufficient Sample Queries or Data

The physical schema produced by the Database Designer is only as good as the sample queries and data you provide. For example, a set of queries that:

- contains no joins cannot produce create pre-join projections
- contains no predicates cannot produce fast pre-join projections

Load Problems

Load failure can be caused by:

- a file name that does not have a corresponding table
- an invalid pathname (must be absolute or relative to where the Admin Tools are running)
- the wrong number of columns in a table
- invalid data types
- invalid data formatting

The Database Designer writes rejected rows into a file so that you can correct and reload them.

Writing Custom Projections

In most cases, the projections created by the Database Designer provide excellent query performance within physical constraints, but the accuracy of the Database Designer is limited to the information it can get from your input files, particularly the example queries. Should the projection created by the Database Designer not meet your needs, you can write custom projections as described in Defining Custom Projections in the Database Administrator's Guide (Advanced).

Vertica Systems, Inc. strongly recommends that you contact ***Technical Support*** (on page 11) before and after designing your physical schema. Technical Support can inspect your physical schema design for problems before you execute it.

Requirements for a K-Safe Physical Schema Design

If you choose to write custom projections, be aware that your physical schema design must meet the following requirements.

- Segmented projections must have K buddy projections (projections that have identical columns and segmentation criteria, except that corresponding segments are placed on different nodes).
- Unsegmented projections must be replicated on all nodes.

In general, only small dimension tables are unsegmented.

Operating the Database

Managing Your License Key

Vertica license keys provide full product functionality for a specific period of time or forever if you purchase a perpetual license. There is a grace period during which Vertica continues to work after the license has expired. The length of the grace period depends on the license.

Obtaining a License Key File

To obtain a license key file:

1. Go to http://www.vertica.com/support (http://www.vertica.com/support) and follow the instructions there. You will be given download access to a file whose name includes your organization and the date the license was requested.
2. Download the license file to the `/tmp` directory on the Administration Host.

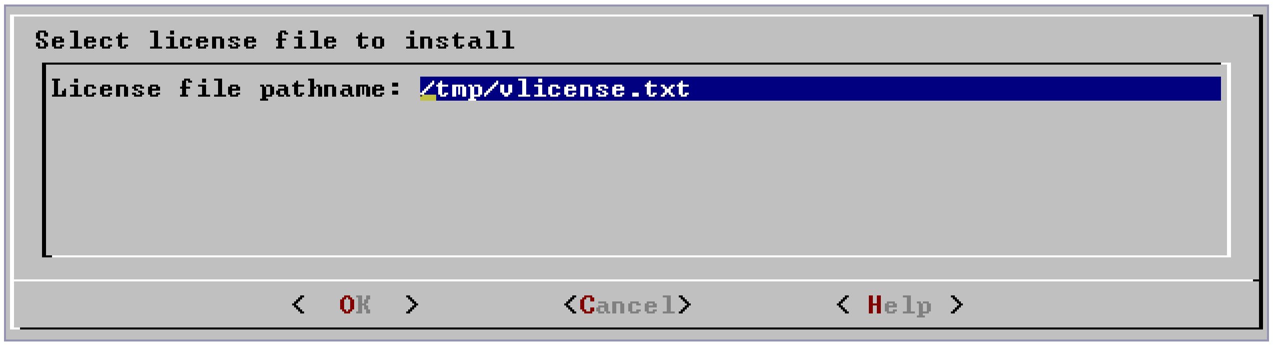
Vertica does not use the downloaded file. It must first be installed into each database.

Installing a New or Upgrade Installation License Key

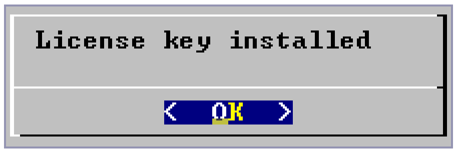
1. Obtain a license key as described above.
2. Install Vertica as described in the Installation Guide.

The first time you log in as the Database Administrator and run the Vertica Administration Tools, the user interface displays:

1. the EULA (license agreement). Type "accept" to proceed.
2. a form asking for the pathname to the license key file that you downloaded from the Vertica Web site. The default pathname is `/tmp/vlicense.key`. If this is correct, press OK. Otherwise, enter the **absolute pathname** of the file in the **bottom field** of the form and press OK.



```
Select license file to install
License file pathname: /tmp/vlicense.txt
```

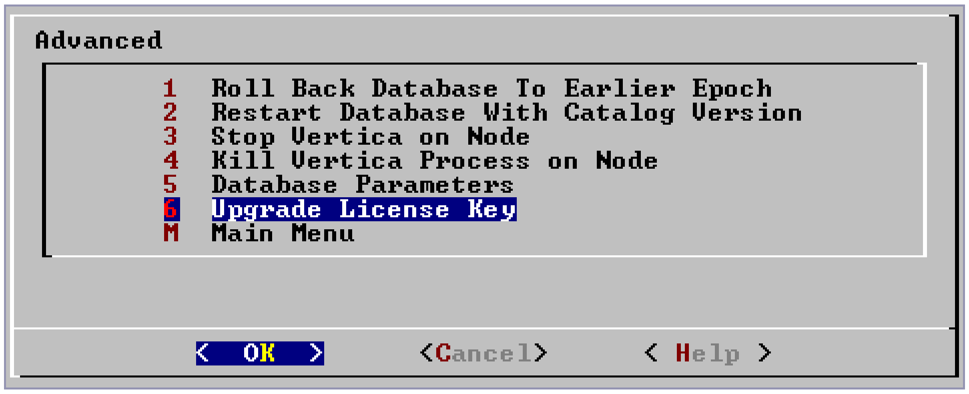


3. License validation does not occur until you create a database.

Installing an Renewal License Key

When a license is nearing expiration Vertica logs warning messages indicating the expiration date. When a license expires (the grace period has expired) Vertica logs an invalid license error and stops executing queries. In that case:

1. Obtain a license key as described above.
2. Start a database.
3. In the Administration Tools, go to **Advanced** > **Upgrade License Key**


Advanced
1 Roll Back Database To Earlier Epoch
2 Restart Database With Catalog Version
3 Stop Vertica on Node
4 Kill Vertica Process on Node
5 Database Parameters
6 Upgrade License Key
M Main Menu
< OK >
<Cancel>
< Help >

Using Vertica Functions to Manage License Keys

The following functions are described in the SQL Reference Manual:

- Use the INSTALL_LICENSE function to install the license key in the global catalog.
- Use the DISPLAY_LICENSE function to display the license information.

Vertica automatically installs your new license if and when you start another database.

Starting and Stopping the Database

Starting a Vertica Database

1. Use ***View Database Cluster State*** (page 75) to make sure that **all nodes are down** and that no other database is running. If not all nodes are down, see ***Shutdown Problems*** (page 100).
2. Use ***Start Database*** (page 77) to start the database.
3. Check the log files to make sure that **no startup problems occurred**, as described in ***Monitoring the Database*** (page 52).
4. If the startup fails, see ***Startup Problems*** (page 102).

Stopping a Vertica Database

1. Use ***View Database Cluster State*** (page 75) to make sure that **all nodes are up**. If not all nodes are up, see ***Restarting a Node*** (page 79).
2. Make sure that there are **no open connections** to the database. On each host:

```
$ netstat | grep 5433
tcp        0      0 host01:5433         host01:33558
   ESTABLISHED
tcp        0      0 host01:33558        host01:5433
   ESTABLISHED
```

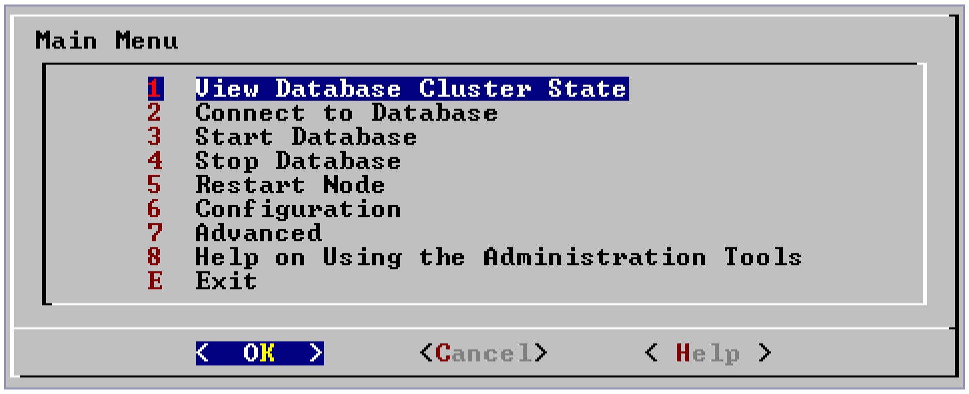
3. Use ***Stop Database*** (page 78) to stop the database.
4. If the shutdown fails, see ***Shutdown Problems*** (page 100).

Monitoring the Database

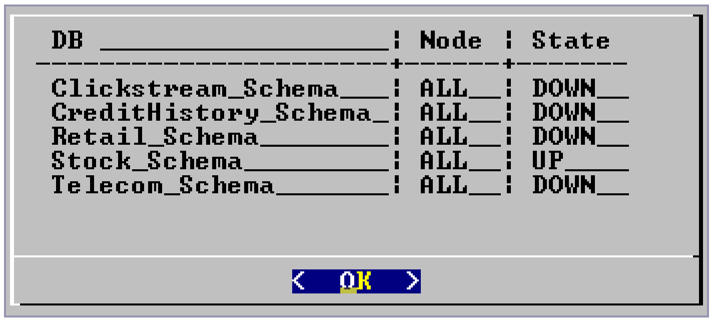
View Database Cluster State

The Administration Tools provide a ***View Database Cluster State*** (page 75) tool. This tool shows the current state of the nodes in the database.

1. On the Main Menu, select **View Database Cluster State**.

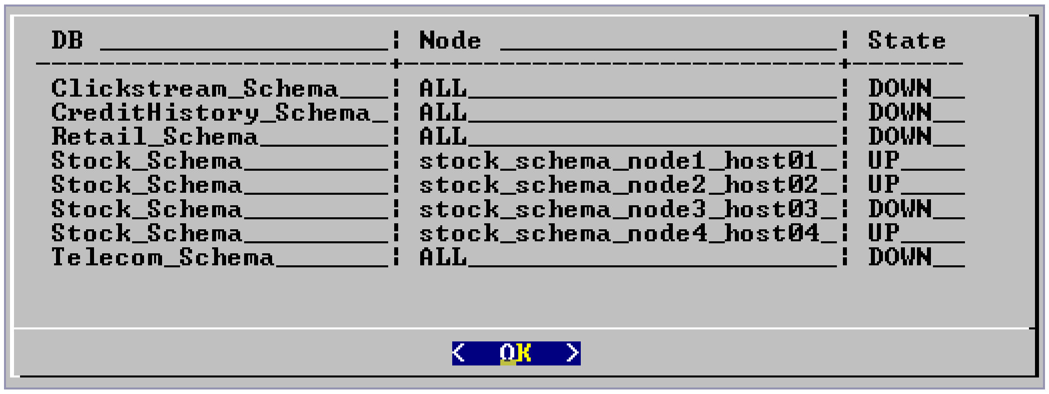


2. The normal state of a running database is ALL UP. The normal state of a stopped database is ALL DOWN.



```
 DB __________________| Node | State
---------------------+------+-------
 Clickstream_Schema___| ALL__| DOWN__
 CreditHistory_Schema_| ALL__| DOWN__
 Retail_Schema________| ALL__| DOWN__
 Stock_Schema_________| ALL__| UP____
 Telecom_Schema_______| ALL__| DOWN__
```

3. If some nodes are UP and some DOWN, restart the database as described in ***Starting and Stopping the Database*** (page 51) (unless you have a known node failure and wish to continue in that state.)



```
 DB __________________| Node _____________________| State
---------------------+---------------------------+-------
 Clickstream_Schema___| ALL_______________________| DOWN__
 CreditHistory_Schema_| ALL_______________________| DOWN__
 Retail_Schema________| ALL_______________________| DOWN__
 Stock_Schema_________| stock_schema_node1_host01_| UP____
 Stock_Schema_________| stock_schema_node2_host02_| UP____
 Stock_Schema_________| stock_schema_node3_host03_| DOWN__
 Stock_Schema_________| stock_schema_node4_host04_| UP____
 Telecom_Schema_______| ALL_______________________| DOWN__
```

4. Nodes that are INITIALIZING or RECOVERING indicate that ***failure recovery*** (page 99) is in progress.
5. Nodes in other states (such as NEEDS_CATCHUP) are transitional and can be ignored unless they persist. In that case, call ***Technical Support*** (on page 11).

For more information, see:

- the ***Advanced*** (page 92) section of the Administration Tools
- ***Startup Problems*** (page 102)

  Shutdown Problems (page 100)

Monitoring the Log Files

When a Database is Running

When a Vertica database is running, each node in the cluster writes messages into a file named **vertica.log**. For example, the Tuple Mover and the transaction manager write INFO messages into vertica.log at specific intervals even when there is no WOS activity.

To monitor a running database in real time:

1. Log into the database administrator account on any or all of the nodes in the cluster.
2. Enter:

```
$ tail -f catalog-path/database-name/node-name_catalog/vertica.log
```

catalog-path	the catalog pathname specified when you created the database (see ***Create Database*** (page 82) in the Database Administrator's Guide).
database-name	the database name (case sensitive)
node-name	the node name as specified in the database definition (see ***View Database*** (page 91) in the Database Administrator's Guide).

When No Database is Running

When no database is running, each node in the cluster writes messages into a file named **dbLog**. For example, if a database fails to start before it can write messages into `vertica.log`, check the file:

```
catalog-path/database-name/dbLog
```

where *catalog-path* and *database-name* are as described above.

See Also

- ***Rotating Log Files*** (page 54)

Rotating the Log Files

Using the logrotate Tool

You can use the Linux tool ***logrotate*** http://www.linuxcommand.org/man_pages/logrotate8.html (see `man logrotate`) to manage Vertica log files. The steps to configure log rotation for your installation are as follows:

1. On each node, edit the file **/opt/vertica/config/vertica.logrotate** to provide the paths to your database log files and to alter configuration parameters as required. By default, vertica.logrotate operates on both **vertica.log** and **dbLog**, as described in ***Monitoring the Log Files*** (page 53). It rotates each log file once a week, compresses them, and keeps them on disk for one year.
2. Copy the vertica.logrotate file to the system logrotate directory and change its name to **vertica**.

   ```
   $ cp /opt/vertica/config/vertica.logrotate /etc/logrotate.d/vertica
   ```
3. Optionally create a cron job to automatically rotate the log files.

Manual Log Rotation

To perform manual log rotation, use the following procedure. You can use this procedure to implement a custom log rotation process. No log messages are lost during the procedure.

1. Rename or archive the vertica.log file that is produces. For example:

   ```
   $ mv vertica.log vertica.log.1
   ```
2. Send the Vertica process the USR1 (gracefully restart) signal. For example:

   ```
   $ killall -USR1 vertica
   ```

   or

   ```
   $ ps -ef | grep -i vertica
   $ kill -USR1 process-id
   ```

Using the SQL Monitoring API

Vertica provides an API for monitoring various features and functions within a database in the form of virtual tables that can be queried using a limited form of the SELECT statement. You can use external tools to query the virtual tables and act upon the information as desired. For example, you can a third-party monitoring tool to periodically query the K-Safety level of the database. If it falls below the desired level indicating a host failure, the monitoring tool can use any means necessary to notify the database administrator and/or appropriate IT personnel.

The virtual tables that make up the monitoring API are described in the SQL Reference Manual. They are:

Virtual Table	Description
VT_COLUMN_STORAGE	monitors the amount of disk storage used by each column of

	each projection on each node.
VT_DISK_STORAGE	monitors the amount of disk storage used by the database on each node.
VT_LOAD_STREAMS	monitors load metrics for each load stream on each node.
VT_PROJECTION_STORAGE	monitors the amount of disk storage used by each projection on each node.
VT_QUERY_METRICS	monitors the sessions and queries executing on each node.
VT_RESOURCE_USAGE	monitors system resource management on each node.
VT_SYSTEM	monitors the overall state of the database.
VT_TABLE_STORAGE	monitors the amount of disk storage used by each table on each node.
VT_TUPLE_MOVER	monitors the status of the Tuple Mover on each node.

Query Syntax

Virtual table queries use a different processing mechanism than database queries Thus, the virtual tables support only a very limited set of query capabilities:

- Only the FROM Clause is allowed in the SELECT statement.
- Historical queries are not allowed, including AT EPOCH LATEST.

When a cluster is the Recovering state, the server refuses connection requests and thus cannot be monitored using the SQL monitoring API.

Examples

```
SELECT CURRENT_EPOCH, K_SAFETY FROM VT_SYSTEM;
SELECT * FROM VT_RESOURCE_USAGE;
SELECT NODE, TOTAL_USER_SESSIONS, TOTAL_QUERIES_EXECUTED FROM
VT_QUERY_METRICS;
```

Monitoring Processes

You can use ps to monitor the database and Spread processes running on each node in the cluster. For example:

```
$ ps aux | grep /opt/Vertica/bin/Vertica
$ ps aux | grep /opt/Vertica/sbin/spread
```

You should see exactly one Vertica process and exactly one Spread process on each node. To monitor Administration Tools and connector processes:

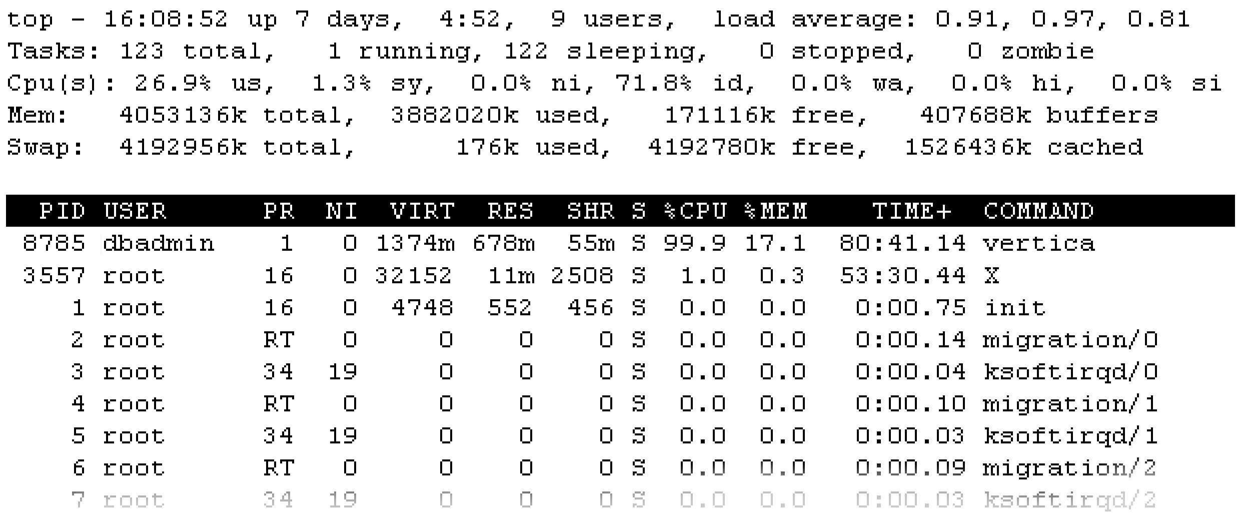
```
$ ps aux | grep Vertica
```

There may be many connection processes but at most one Administration Tools process.

Monitoring System Resource Usage

You may find it helpful to monitor system resource usage on any or all nodes in the cluster.

1. Log into the database administrator account on a node.
2. Run the **top** utility. A high CPU percentage in `top` indicates that Vertica is CPU-bound. For example:



```
top - 16:08:52 up 7 days,  4:52,  9 users,  load average: 0.91, 0.97, 0.81
Tasks: 123 total,   1 running, 122 sleeping,   0 stopped,   0 zombie
Cpu(s): 26.9% us,  1.3% sy,  0.0% ni, 71.8% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:   4053136k total,  3882020k used,   171116k free,   407688k buffers
Swap:  4192956k total,      176k used,  4192780k free,  1526436k cached

  PID USER        PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 8785 dbadmin      1   0 1374m 678m  55m S 99.9 17.1  80:41.14 vertica
 3557 root        16   0 32152  11m 2508 S  1.0  0.3  53:30.44 X
    1 root        16   0  4748  552  456 S  0.0  0.0   0:00.75 init
    2 root        RT   0     0    0    0 S  0.0  0.0   0:00.14 migration/0
    3 root        34  19     0    0    0 S  0.0  0.0   0:00.04 ksoftirqd/0
    4 root        RT   0     0    0    0 S  0.0  0.0   0:00.10 migration/1
    5 root        34  19     0    0    0 S  0.0  0.0   0:00.03 ksoftirqd/1
    6 root        RT   0     0    0    0 S  0.0  0.0   0:00.09 migration/2
    7 root        34  19     0    0    0 S  0.0  0.0   0:00.03 ksoftirqd/2
```

   Some possible reasons for high CPU usage are:

   - The Tuple Mover runs automatically and thus consumes CPU time even if there are no connections to the database. If you believe this to be a problem, contact ***Technical Support*** (on page 11).
   - The pdflush process (a set of worker threads for writing back dirty filesystem data) is consuming a great deal of cpu time, possibly driving the load way up. Adding RAM appears to make the problem worse. Log in to root and change the Linux parameter swappiness to 0.

```
# echo 0 > /proc/sys/vm/swappiness
```

   - Some information sources:

     TechRepublic http://techrepublic.com.com/5206-6230-0.html?forumID=36&threadID=175191&start=0

     Red Hat https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=150653

     Indiana University Unix Systems Support Group http://www.ussg.iu.edu/hypermail/linux/kernel/0404.3/0744.html

3. Run the **iostat** utility. A high idle time in `top` at the same time as a high rate of blocks read in `iostat` indicates that Vertica is disk-bound. For example:

```
$ /usr/bin/iostat
Linux 2.6.9-22.ELsmp (qa0)      07/13/2007

avg-cpu:  %user   %nice    %sys %iowait   %idle
```

```
           0.83    0.00    0.13    0.02   99.03

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read
  Blk_wrtn
hda               0.37         3.40        10.37    2117723
  6464640
sda               0.46         1.94        18.96    1208130
  11816472
sdb               0.26         1.79        15.69    1114792
  9781840
sdc               0.24         1.80        16.06    1119304
  10010328
sdd               0.22         1.79        15.52    1117472
  9676200
md0               8.37         7.31        66.23    4554834
  41284840
```

Managing Database Security

Authentication and authorization in a Vertica database are based on SQL standard users and privileges except that each database has a superuser who can bypass the authorization mechanism. The superuser has the same name as the Linux user that created the database (the Database Administrator). Creating a database allows the DBA to specify the superuser password or an empty string (no password).

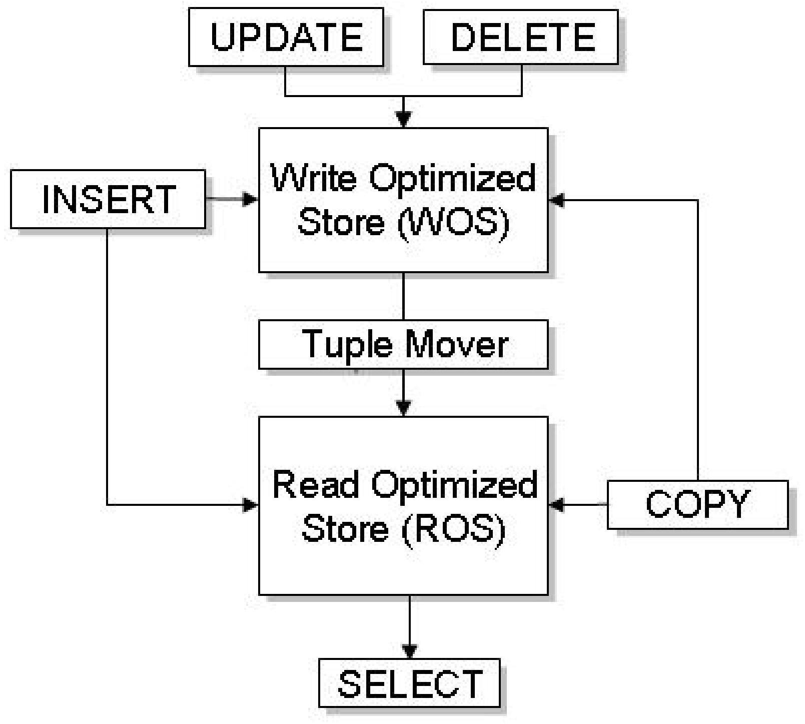
WARNING: If the DBA does not specify a superuser password at database creation time, the database authentication method is permanently set to "trust," which allows any user to log in without supply a password.

The superuser can:

- create normal database users but not other superusers
- drop normal database users
- grant privileges to normal database users
- revoke privileges granted to normal database users

Loading and Modifying Data

Vertica's hybrid storage model provides a great deal of flexibility when loading and modifying data.



The tuple mover is the component of Vertica that moves the contents of the Write Optimized Store (WOS) into the Read Optimized Store (ROS). This data movement is known as a moveout. Normally, the tuple mover runs automatically in the background at preset intervals and is referred to as the ATM.

Bulk Loading

Using the COPY, LCOPY and INSERT Commands

Using the COPY and LCOPY Commands

The COPY and LCOPY commands with the DIRECT keyword write multiple rows directly into the ROS and are recommended for large bulk loads (more than 100MB) and infrequent bulk loads (no more than one per day). The COPY and LCOPY commands commit the current transaction and commit themselves automatically. However, some rows may be rejected and reported in the log files.

A large initial bulk load may temporarily affect query performance while Vertica organizes the data.

The COPY and LCOPY commands without the DIRECT keyword load data into memory (WOS) and are recommended for small bulk loads (less than 100MB) and frequent bulk loads (more than one per day).

The recommended way to use COPY and LCOPY is in script files, as described in ***Using Load Scripts*** (page 63). You can, however, use these commands interactively by piping a text file to vsql and executing COPY or LCOPY with the standard input stream as the input file. For example:

```
$ cat fact_table.tbl | vsql -c "COPY FACT_TABLE FROM STDIN DELIMITER '|'
DIRECT"
$ cat fact_table.tbl | vsql -c "COPY FACT_TABLE FROM STDIN DELIMITER '|'
DIRECT"
```

Using the INSERT Command

If necessary, you can force the INSERT command to write single rows directly to the ROS.

Performing the Initial Database Load

Use COPY...DIRECT to load the database for the first time and load the dimension tables before loading the fact table. Loads run at 30-50GB/hour and can be done on several nodes at once, raising the load rate to as high as 200GB/hour.

Extracting Data from an Existing Database

If possible, export the data in text form to a local file or attached disk.

- ETL products typically use ODBC or JDBC to extract data, which gives them program-level access to modify column values as needed for the load files.
- Database systems provide a variety of export methods.

Oracle does not provide a tool than can export to text. To export data, run a SELECT query in Oracle's SQL*Plus command line query tool using a specified column delimiter, suppressed headers, etc. Redirect the output to a local file.

Dimension tables generally fit into a single load file. Split the fact table data into 250-500GB load files. For example, a 10 TB fact table would require 20-40 load files.

Choose a column-value delimiter character that does not appear in any CHAR(N) or VARCHAR(N) data values. The vertical bar (|) may be a good choice. You can use a query to test for the existence of a certain character in a column. For example:

```
SELECT COUNT(*) FROM T WHERE X LIKE '%|%'
```

If only a few rows contain |, you can eliminate them from the load file using a WHERE clause and load them separately using a different delimiter.

Oracle has a REGEX_REPLACE function that can substitute one substring with another but this will slow down the unload operation significantly. A better approach is to use a WHERE clause to avoid problem rows in the main load file, and the negated WHERE clause with REGEX_REPLACE for just the problem rows.

Moving Data from an Existing Database

Consider using:

- USB 2.0 (or possibly SATA) disks.
- a fast local network connection.

Deliver chunks of data to the different Vertica nodes, by connecting the transport disk or writing files from network copy.

Loading From a Local Hard Disk

USB 2.0 disks can deliver data at about 30 MB per second, or 108 GB per hour, which is fast enough. USB 2.0 disks are easy to use for transporting data from Linux to Linux. Simply set up a ext3 filesystem on the disk and write large files there. Linux 2.6 has USB plug-and-play support, so a USB 2.0 disk is instantly usable on various Linux systems.

For other variants of UNIX, if there is no common filesystem format available, you can use the disk without a filesystem for a single large file. For example:

```
$ cp bigfile /dev/sdc1
```

Even without a filesystem on the disk, plug-and-play support still works on Linux to provide a device node for the disk. To find out the assigned device, plug in the disk and enter:

```
$ dmesg | tail -40
```

SATA disks are usually internal, but can be external, or unplugged safely internally.

Loading Over the Network

A 1Gbps (gigabit per second) network can deliver about 50 MB/s, or 180GB/hr. Vertica can load about 200GB in four hours on each of four nodes. Thus a dedicated 1Gbps LAN should be usable. Slower LANs will be proportionally slower, and non-local networks are probably untenable because the delays over distance slow down the TCP protocol to a small fraction of its apparent bandwidth, even without competing traffic.

Loading From Windows

For loading files directly from Windows to Linux, use NTFS. Although Red Hat Linux as originally installed can read Windows FAT32 file systems, this is not recommended.

Trickle Loading

Using INSERT, UPDATE, and DELETE

The SQL data manipulation language (DML) commands INSERT, UPDATE, and DELETE perform the same functions that they do in a row-oriented database with the following exceptions:

- UPDATE and DELETE always write tuples to the WOS.
- UPDATE writes two tuples into the WOS: one with new data and one marked for deletion.
- DELETE does not actually delete data from disk storage; it simply marks tuples as deleted so that they can be found by historical queries. The ability to delete data from disk storage will be available in a future release.

You can intermix the INSERT, UPDATE, and DELETE commands. Vertica follows the SQL-92 transaction model. In other words, you do not have to explicitly start a transaction but you must use a COMMIT or ROLLBACK command (or COPY) to end a transaction. Cancelling a DML statement causes the current transaction to roll back.

You cannot use AT EPOCH LATEST or AT TIME in an INSERT ... SELECT statement. To run the query in snapshot isolation mode, use the SET TRANSACTION CHARACTERISTICS statement to set the isolation level to **READ COMMITTED**.

Avoiding WOS Overflow

Loading data into the WOS is a tradeoff of speed vs. potential memory overflow. Writing to the WOS is much faster than writing to the ROS, but writing too much data too quickly can overrun the amount of memory available. When that happens, transactions roll back until the tuple mover can catch up and move data from the WOS to the ROS.

It is difficult to provide guidelines for writing data to the WOS other than that single INSERT commands cause an inefficient usage of WOS space. If you experience frequent WOS overflow, contact ***Technical Support*** (on page 11). A support representative may be able to change some internal parameters in order to optimize tuple mover behavior for your database.

Using Load Scripts

This section describes how to write and run a load script using the COPY command.

Writing a Load Script

The COPY command requires an absolute pathname for a data file. It does not accept relative pathnames. However, you can specify the locations of your data files relative to your Linux working directory using vsql variables.

1. Create a vsql variable containing your Linux current directory.
   ```
   \set t_pwd `pwd`
   ```

2. Create another vsql variable that uses a path relative to the Linux current directory variable for a specific data file.
   ```
   \set input_file '\'':t_pwd'/Date_Dimension.tbl\''
   ```
3. Use the second variable in the COPY statement.
   ```
   COPY Date_Dimension FROM :input_file DELIMITER '|';
   ```
4. Repeat steps 2 and 3 for all data files. Load the dimension tables before the fact table.

Running a Load Script

You can run a load script on any host, as long as the data files are on that host.

1. Change your Linux working directory to the location of the data files.
   ```
   $ cd /opt/vertica/doc/retail_example_database
   ```
2. Run the Administration Tools.
   ```
   $ /opt/vertica/bin/adminTools
   ```
3. Connect to the database.
4. Run the load script.

Using Parallel Load Streams

You can use multiple connections to load a Vertica database. The optimal number of load streams depends on several factors including the number of nodes, the physical and logical schemas, host processors, memory, disk space, and so forth. Too many load streams can cause systems to run out of memory.

Vertica Systems, Inc. has successfully tested eight streams on four nodes but does not guarantee that you can do the same on your database.

Loading Character Data

Character Set

The type of a data file must be compatible with the database character set. For example, if the data file is type ASCII or UTF-8, you can be load it into a UTF-8 database. However, if the data file is type ISO 8859-1 (Latin1), which is not compatible with UTF-8, the column values containing multi-byte characters cannot be displayed in the result set.

1. Use the **file** command to check the type of a data file. For example:
   ```
   $ file Date_Dimension.tbl
   Date_Dimension.tbl: ASCII text
   ```
   The file command may indicate ASCII TEXT even though the file contains multi-byte characters.
2. Use the **wc** command to check for this problem. For example:
   ```
   $ wc Date_Dimension.tbl
     1828   5484 221822 Date_Dimension.tbl
   ```

If the wc command returns an error such as `Invalid or incomplete multibyte or wide character`, the data file is using an incompatible character set.

Using Quoted Characters as Literals

You can use the backslash character (\) to quote data characters that would otherwise be taken as special characters. In particular, the following characters must be preceded by a backslash if they appear as part of a column value:

- the `COPY ... DELIMITER` character (default is the tab character)
- the `COPY ... NULL` string (default is \N)
- the backslash character itself
- newline and other control characters

Examples

In these examples, the DELIMITER is comma for visibility.

`,1,2,3,`
`,1,2,3`
`1,2,3,`
	Leading and trailing delimiters are ignored. Thus, the rows all have three columns.	

`123,\n,\\n,456`	Using the default null string (\n), the row would be interpreted as:
	`123`
	`NULL`
	`\n`
	`456`
		Using a non-default null string, the row would be interpreted as:
		`123`
		`newline`
		`\n`
		`456`

`123,this\, that\, or the other,something else,456`		

	The row would be interpreted as:
	`123`
	`this, that, or the other`
	`something else`
	`456`	

Backing Up the Database

1. Shut down the database cleanly using the ***Stop Database*** (page 78) command.
2. Back up the catalog and data directories associated with each node definition in the database.
3. Back up the `/opt/vertica/config` directory. The Administration Tools need that information to work correctly.

Using the Administration Tools

Vertica provides a set of tools that allows you to perform administrative tasks quickly and easily. Most of the database administration tasks in Vertica can be done using the Administration Tools. If your administrative tasks go beyond those provided by the tools, contact ***Technical Support*** (page 11) for information.

Always run the Administration Tools using the Database Administrator account on the Administration Host if possible. Make sure that no other Administration Tools processes are running.

If the Administration Host is down, run the Administration Tools on a different node in the cluster. That node permanently takes over the role of Administration Host.

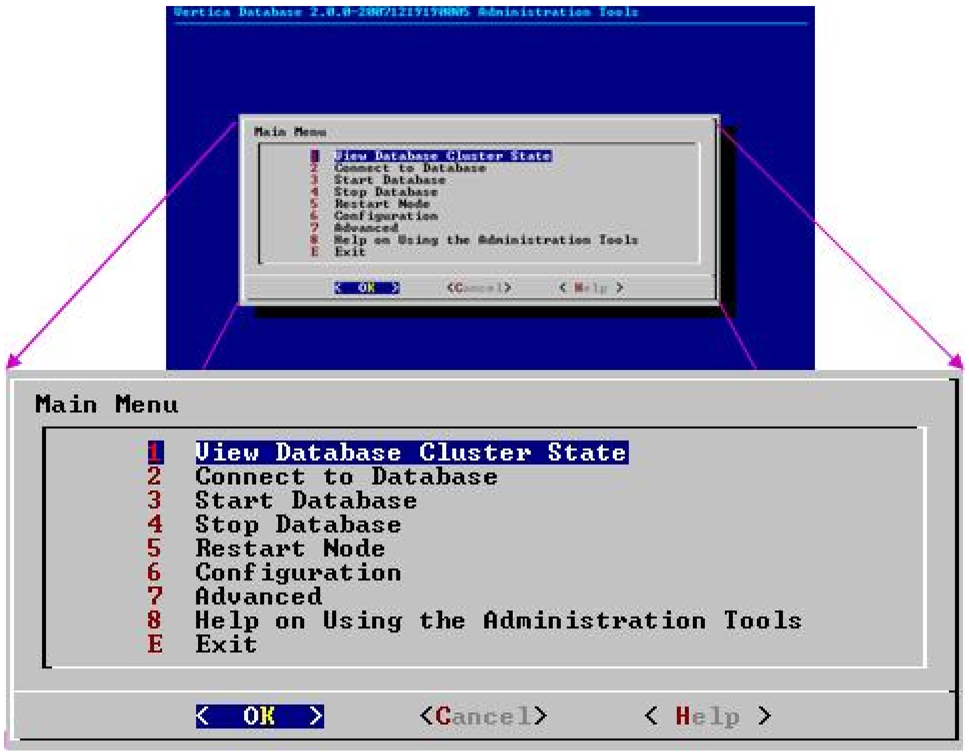
Running the Administration Tools

At the Linux command line:

```
$ /opt/vertica/bin/adminTools [ -t | --tool ] toolname [ options ]
```

toolname	is one of the tools described in the ***Administration Tools Reference*** (page 75).	
options	`-h` `--help`	shows a brief help message and exits.
	`-a` `--help_all`	lists all command line sub-commands and options as described in the Writing Administration Tools Scripts section of the Database Administrator's Guide (Advanced).

If you omit the *toolname* and *options*, the **Main Menu** dialog box appears inside your console or terminal window with a dark blue background and a title on top. The screen captures used in this documentation set are cropped down to the dialog box itself, as shown below.



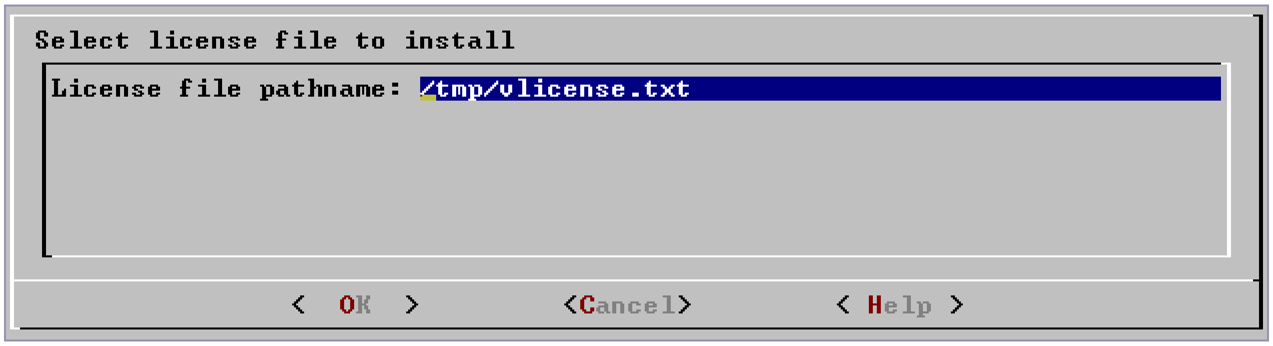
If you are unfamiliar with this type of graphical user interface, read ***Using the Graphical User Interface*** (page 70) before you do anything else.

First Time Only

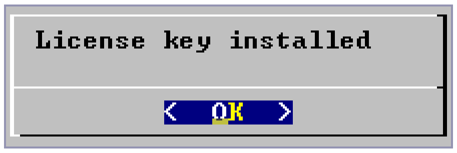
The **first time** you log in as the Database Administrator and run the Administration Tools, the user interface displays:

1. the EULA (license agreement). Type **accept** to proceed.

2. a form asking for the pathname to the license key file that you downloaded from the Vertica Web site. The default pathname is `/tmp/vlicense.key`. If this is correct, press OK. Otherwise, enter the **absolute pathname** of the file and press OK.



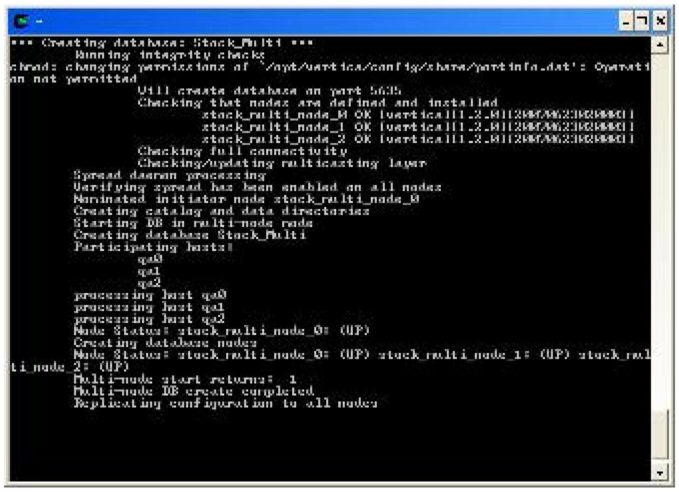
```
Select license file to install

License file pathname: /tmp/vlicense.txt
```



```
License key installed
```

Between Dialogs

While the Administration Tools are working, you will see the command line processing in a window similar to the one shown below. Do not interrupt the processing.



```
*** Creating database: Stack_Multi ***
        Running integrity checks
chmod: changing permissions of `/opt/vertica/config/share/portinfo.dat': Operati
on not permitted
                Will create database on port 5435
                Checking that nodes are defined and installed
                        stack_multi_node_0 OK [vertica][1.2.0][20070623020001]
                        stack_multi_node_1 OK [vertica][1.2.0][20070623020001]
                        stack_multi_node_2 OK [vertica][1.2.0][20070623020001]
                Checking full connectivity
                Checking/updating multicasting layer
        Spread daemon processing
        Verifying spread has been enabled on all nodes
        Nominated initiator node stack_multi_node_0
        Creating catalog and data directories
        Starting DB in multi-node mode
        Creating database Stack_Multi
        Participating hosts:
                qa0
                qa1
                qa2
        processing host qa0
        processing host qa1
        processing host qa2
        Node Status: stack_multi_node_0: (UP)
        Creating database nodes
        Node Status: stack_multi_node_0: (UP) stack_multi_node_1: (UP) stack_mul
ti_node_2: (UP)
        Multi-node start returns:  1
        Multi-node DB create completed
        Replicating configuration to all nodes
```

Using the Graphical User Interface

The Vertica Administration Tools are implemented using Dialog, a graphical user interface that works in terminal (character-cell) windows.The interface responds to mouse clicks in some terminal windows, particularly local Linux windows, but you may find that it only responds to keystrokes. Thus, this section describes how to use the Administration Tools using only keystrokes.

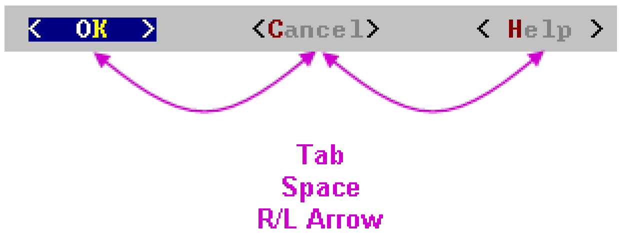
This section does not describe every possible combination of keystrokes that can be used to accomplish a particular task. Feel free to experiment and to use whatever keystrokes you prefer.

Return

In all dialogs, when you are ready to execute a command, select a file, or cancel the dialog, press the **Return** key. The command descriptions in this section do not explicitly instruct you to press **Return**.

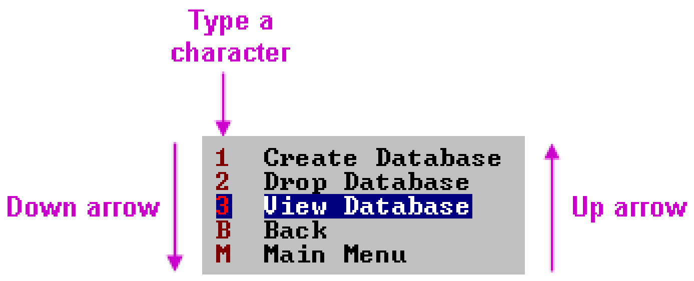
OK - Cancel - Help

The OK, Cancel, and Help buttons are present on virtually all dialogs. Use the **tab**, **space bar**, or right and left **arrow keys** to select one or the other. When you have made your selection press **Return**. The same keystrokes apply to dialogs that present a choice of **Yes** or **No**.



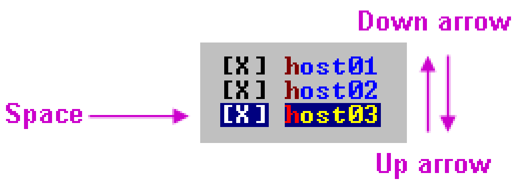
Menu Dialogs

Some dialogs require you to choose one command from a menu. Type the alphanumeric **character** shown or use the up and down **arrow keys** to select a command. When you have made your selection, press **Return**.



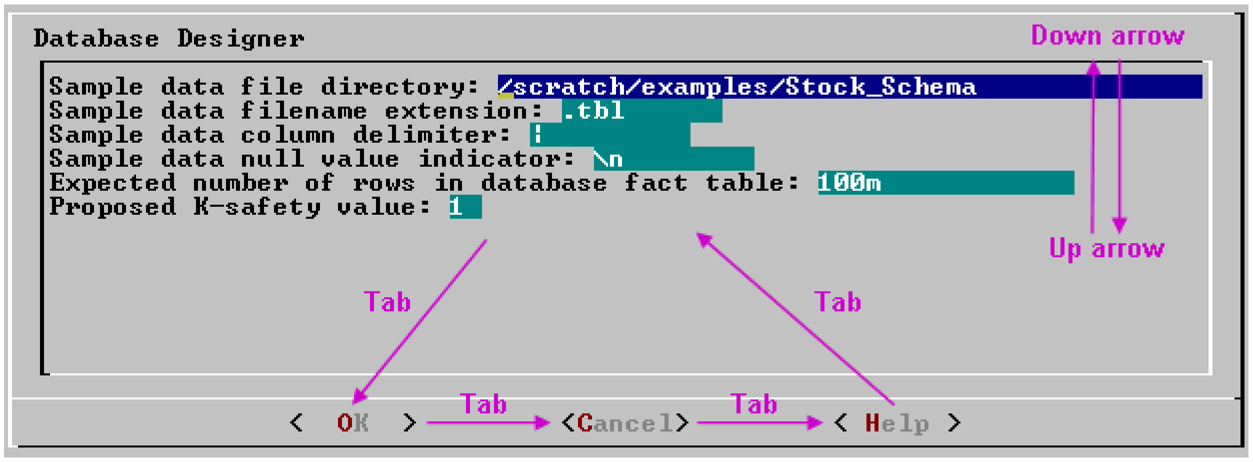
List Dialogs

In a list dialog, use the up and down **arrow keys** to highlight items, then use the **space bar** to select the items, marking them with an X. Some list dialogs allow you to select multiple items. when you have finished selecting items, press **Return**.



Form Dialogs

In a form dialog, use the **tab** key to cycle between OK, Cancel, Help, and the form field area. Once the cursor is in the form field area, use the up and down **arrow keys** to select an individual field (highlighted) and **enter information**. When you have finished entering information in all fields, press **Return**.



```
Database Designer

Sample data file directory: /scratch/examples/Stock_Schema
Sample data filename extension: .tbl
Sample data column delimiter: |
Sample data null value indicator: \n
Expected number of rows in database fact table: 100m
Proposed K-safety value: 1
```

Help Buttons

Online help is provided in the form of text dialogs. If you have trouble viewing the help, see ***Notes for Remote Terminal Users*** (page 71) in this document.

Notes for Remote Terminal Users

The appearance of the graphical interface depends on the color and font settings used by your terminal window. The screen captures in this document were made using the default color and font settings in a Cygwin Bash Shell running on Windows XP. If you are using a remote terminal application such as PuTTY or a Cygwin bash shell, make sure your window is at least:

23 characters wide and 81 characters high

If you are using PuTTY, you can make the Administration Tools look like the screen captures in this document:

1. In the PuTTY Configuration dialog, create or load a saved session.
2. Click Category: **Window** > **Appearance**.

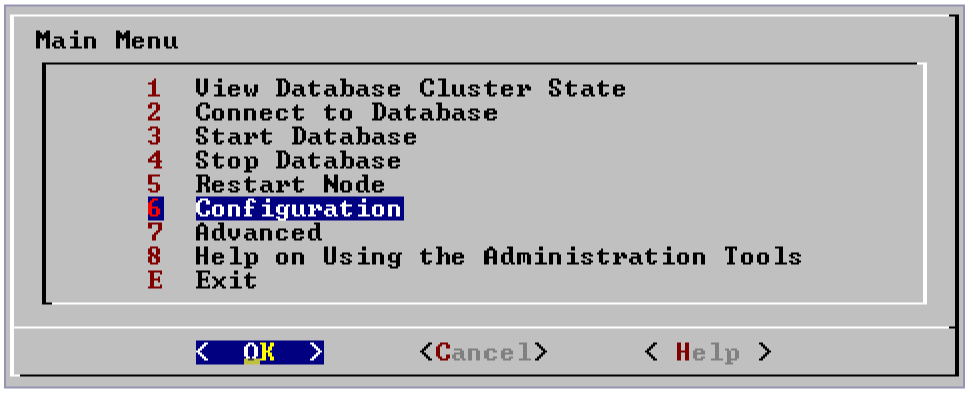
3. In the **Font settings**, click the **Change...** button.
4. Select Font: **Terminal** Font Style: **Regular** Size: **10**
5. Click **OK**.
6. Click Category: **Session**
7. Click **Save**

Repeat these steps for each existing session that you use to run the Administration Tools.

Using the Online Help

In a Menu Dialog

1. Use the **up and down arrow keys** to choose the command on which you want help.

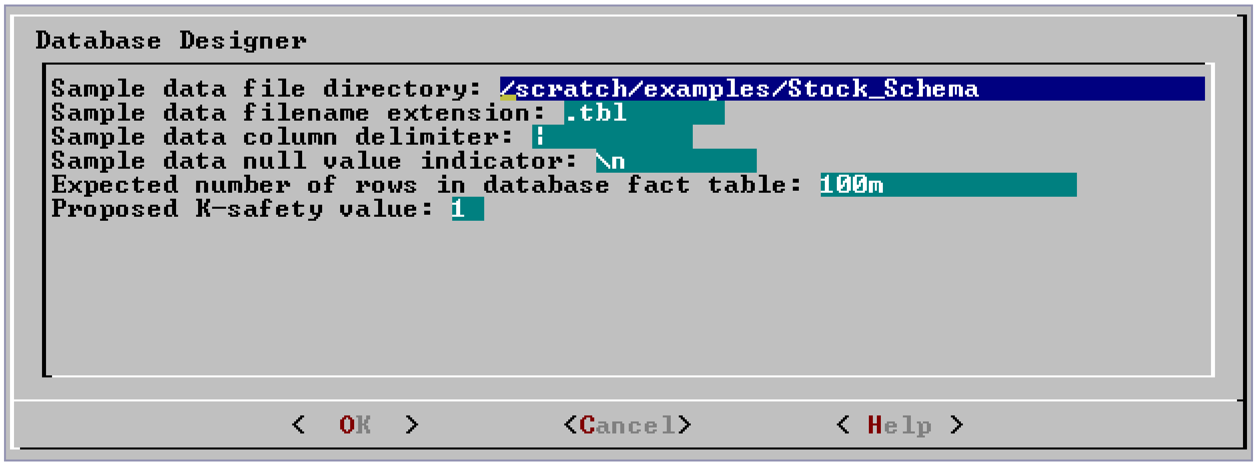


```
Main Menu

      1  View Database Cluster State
      2  Connect to Database
      3  Start Database
      4  Stop Database
      5  Restart Node
      6  Configuration
      7  Advanced
      8  Help on Using the Administration Tools
      E  Exit

     <  OK  >      <Cancel>      < Help >
```

2. Use the **Tab key** to move the cursor to the **Help button**.
3. Press **Enter** (Return).

In a Form Dialog

1. Use the **up and down arrow keys** to choose the field on which you want help.



```
Database Designer

Sample data file directory: /scratch/examples/Stock_Schema
Sample data filename extension: .tbl
Sample data column delimiter: |
Sample data null value indicator: \n
Expected number of rows in database fact table: 100m
Proposed K-safety value: 1

     <  OK  >      <Cancel>      < Help >
```

2. Use the **Tab key** to move the cursor to the **Help button**.
3. Press **Enter** (Return).

Scrolling

Some help files are too long for a single screen. Use the **up and down arrow keys** to scroll through the text.

Password Authentication

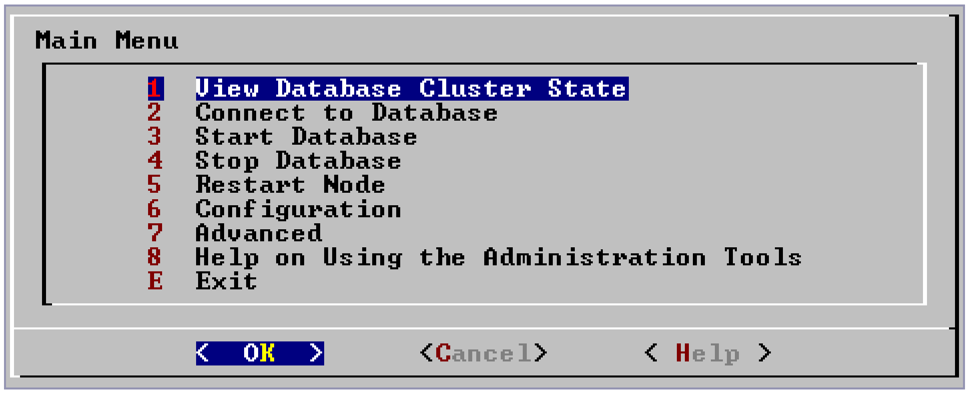
The Administration Tools maintain a session context with regard to password authentication. In other words, you are required to enter a password for a database once per Administration Tools session. If you enter it correctly, you will not be asked to enter it again until the next time you run the Administration Tools. If you enter an incorrect password you must exit the Administration Tools and run it again.

Administration Tools Reference

View Database Cluster State

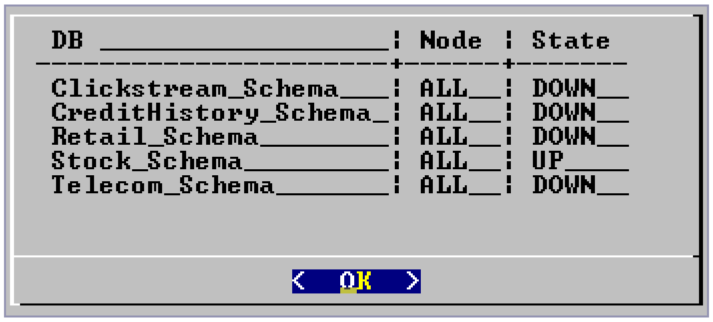
This tool shows the current state of the nodes in the database.

1. On the Main Menu, select **View Database Cluster State**.



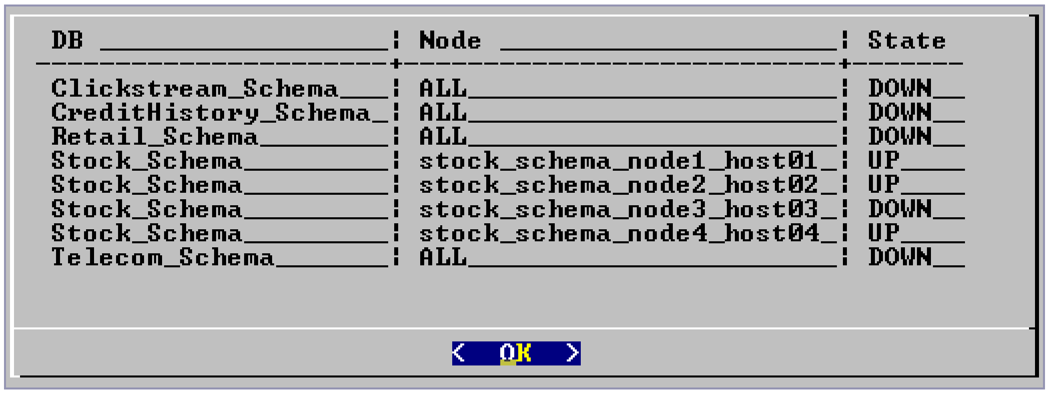
```
Main Menu

        1   View Database Cluster State
        2   Connect to Database
        3   Start Database
        4   Stop Database
        5   Restart Node
        6   Configuration
        7   Advanced
        8   Help on Using the Administration Tools
        E   Exit

          <  OK  >       <Cancel>       < Help >
```

2. The normal state of a running database is ALL UP. The normal state of a stopped database is ALL DOWN.



```
 DB ___________________| Node | State
-----------------------+------+-------
 Clickstream_Schema___| ALL__| DOWN__
 CreditHistory_Schema_| ALL__| DOWN__
 Retail_Schema________| ALL__| DOWN__
 Stock_Schema_________| ALL__| UP____
 Telecom_Schema_______| ALL__| DOWN__

              <  OK  >
```

3. If some nodes are UP and some DOWN, restart the database as described in ***Starting and Stopping the Database*** (page 51) (unless you have a known node failure and wish to continue in that state.)



```
 DB ___________________| Node _____________________| State
-----------------------+---------------------------+-------
 Clickstream_Schema___| ALL_______________________| DOWN__
 CreditHistory_Schema_| ALL_______________________| DOWN__
 Retail_Schema________| ALL_______________________| DOWN__
 Stock_Schema_________| stock_schema_node1_host01_| UP____
 Stock_Schema_________| stock_schema_node2_host02_| UP____
 Stock_Schema_________| stock_schema_node3_host03_| DOWN__
 Stock_Schema_________| stock_schema_node4_host04_| UP____
 Telecom_Schema_______| ALL_______________________| DOWN__

                       <  OK  >
```

4. Nodes that are INITIALIZING or RECOVERING indicate that ***failure recovery*** (page 99) is in progress.
5. Nodes in other states (such as NEEDS_CATCHUP) are transitional and can be ignored unless they persist. In that case, call ***Technical Support*** (on page 11).

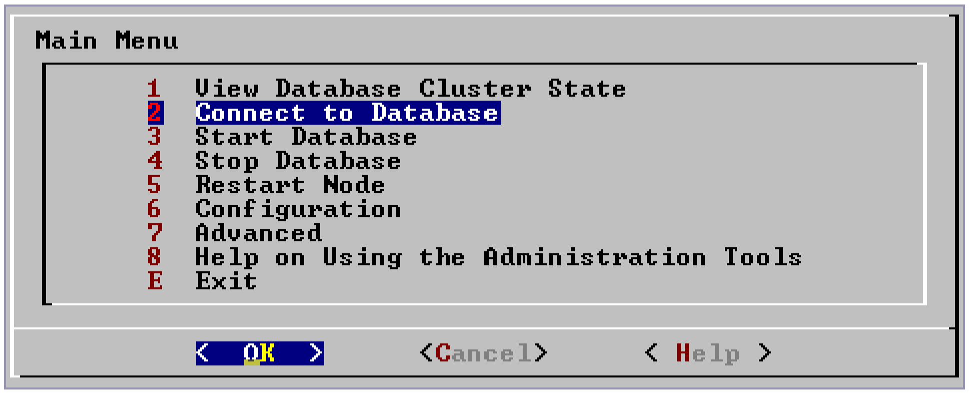
For more information, see:

- the ***Advanced*** (page 92) section of the Administration Tools
- ***Startup Problems*** (page 102)
- ***Shutdown Problems*** (page 100)

Connect to Database

This tool connects to a running database with vsql. You can use the Administration Tools to connect to a database from any node within the database while logged into any user account with access privileges. You cannot use the Administration Tools to connect from a host that is not a database node. To connect from other hosts, run vsql as described in Connecting From the Command Line in the SQL Programmer's Guide.

1. On the **Main Menu**, select **Connect to Database**.



```
Main Menu

        1  View Database Cluster State
        2  Connect to Database
        3  Start Database
        4  Stop Database
        5  Restart Node
        6  Configuration
        7  Advanced
        8  Help on Using the Administration Tools
        E  Exit

          <  OK  >       <Cancel>       < Help >
```

2. **Supply the database password** if asked:

   ```
   Password:
   ```

   The Administration Tools maintain a session context with regard to password authentication. In other words, you are required to enter a password for a database once per Administration Tools session. If you enter it correctly, you will not be asked to enter it again until the next time you run the Administration Tools. If you enter an incorrect password you must exit the Administration Tools and run it again.

3. The Administration Tools **connect to the database** and transfer control to vsql.

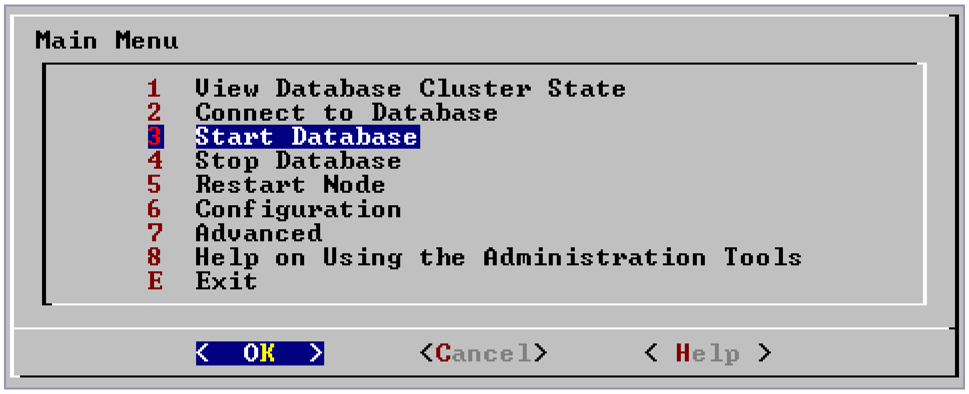
   ```
   Welcome to the vsql, Vertica_Database v2.0.5-0 interactive terminal.
   Type:  \h for help with SQL commands
          \? for help with vsql commands
          \g or terminate with semicolon to execute query
          \q to quit
   Stock_Schema=>
   ```

Refer to Connecting with vsql in the SQL Programmer's Guide for more information.

Start Database

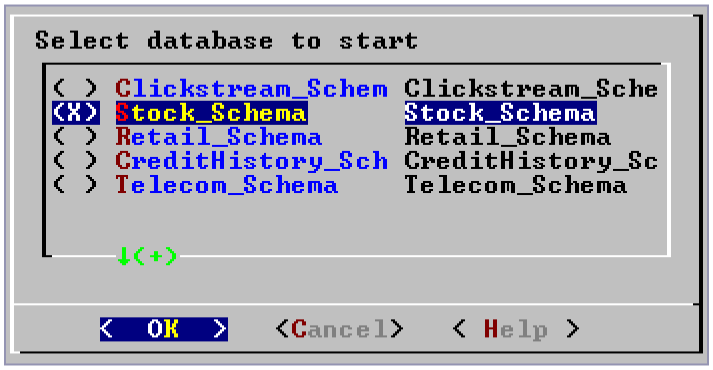
This tool starts an existing database.

1. Use ***View Database Cluster State*** (page 75) to make sure that **all nodes are down** and that no other database is running. If not all nodes are down, see ***Shutdown Problems*** (page 100).
2. On the **Main Menu**, select **Start Database**.



```
Main Menu

        1  View Database Cluster State
        2  Connect to Database
        3  Start Database
        4  Stop Database
        5  Restart Node
        6  Configuration
        7  Advanced
        8  Help on Using the Administration Tools
        E  Exit

           <  OK  >      <Cancel>      < Help >
```

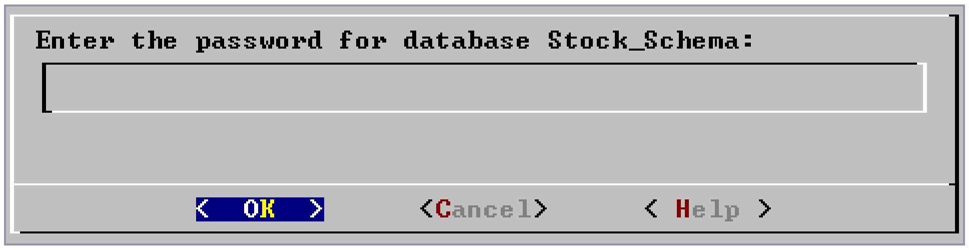
3. **Select** the database to start.



```
Select database to start

( ) Clickstream_Schem  Clickstream_Sche
(X) Stock_Schema       Stock_Schema
( ) Retail_Schema      Retail_Schema
( ) CreditHistory_Sch  CreditHistory_Sc
( ) Telecom_Schema     Telecom_Schema

     ↓(+)

      <  OK  >   <Cancel>   < Help >
```

Vertica Systems, Inc. strongly recommends that you start only one database at a time. If you start more than one database at any time, the results are unpredictable. Users may encounter resource conflicts or perform operations in the wrong database.

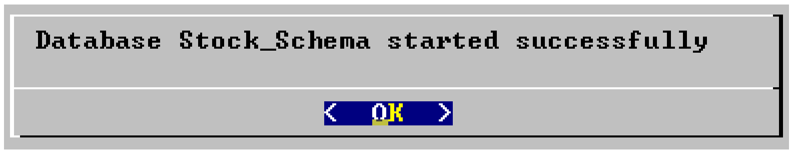
4. Enter the database **password**.



```
Enter the password for database Stock_Schema:

           <  OK  >      <Cancel>      < Help >
```

The Administration Tools maintain a session context with regard to password authentication. In other words, you are required to enter a password for a database once per Administration Tools session. If you enter it correctly, you will not be asked to enter it again until the next time you run the Administration Tools. If you enter an incorrect password you must exit the Administration Tools and run it again.

5. This message confirms that the database started successfully.



```
Database Stock_Schema started successfully
```

6. Check the log files to make sure that **no startup problems occurred**, as described in ***Monitoring the Database*** (page 52).

Notes

- If the database does not start successfully, see ***Startup Problems*** (page 102).

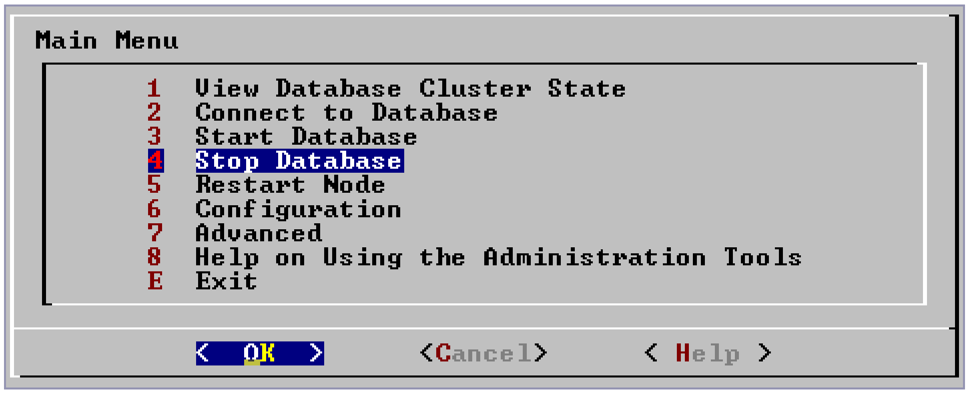
Stop Database

This tool stops a running database.

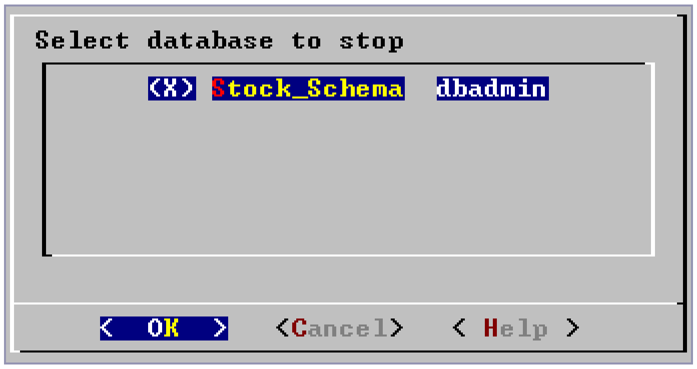
1. Use ***View Database Cluster State*** (page 75) to make sure that **all nodes are up**. If not all nodes are up, see ***Restarting a Node*** (page 79).
2. Make sure that there are **no open connections** to the database. On each host:

```
$ netstat | grep 5433
tcp        0      0 host01:5433          host01:33558
   ESTABLISHED
tcp        0      0 host01:33558         host01:5433
   ESTABLISHED
```

3. On the **Main Menu**, select **Stop Database**.

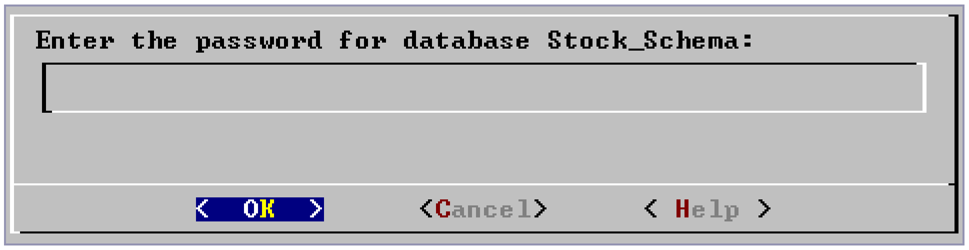


4. **Confirm** that you want to stop the currently running database.



```
Select database to stop
    (X) Stock_Schema   dbadmin

<  OK  >   <Cancel>   < Help >
```

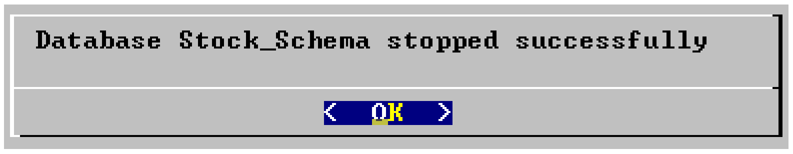
5. **Enter the password** if asked.



```
Enter the password for database Stock_Schema:

<  OK  >   <Cancel>   < Help >
```

The Administration Tools maintain a session context with regard to password authentication. In other words, you are required to enter a password for a database once per Administration Tools session. If you enter it correctly, you will not be asked to enter it again until the next time you run the Administration Tools. If you enter an incorrect password you must exit the Administration Tools and run it again.

6. This message confirms that the database has been successfully stopped.



```
Database Stock_Schema stopped successfully

<  OK  >
```

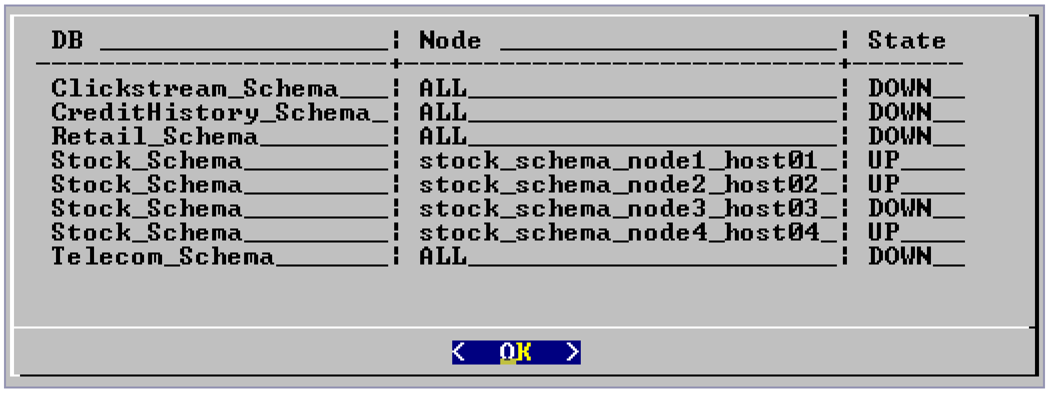
Notes

- If the database does not stop successfully, see ***Shutdown Problems*** (page 100).

Restart Node

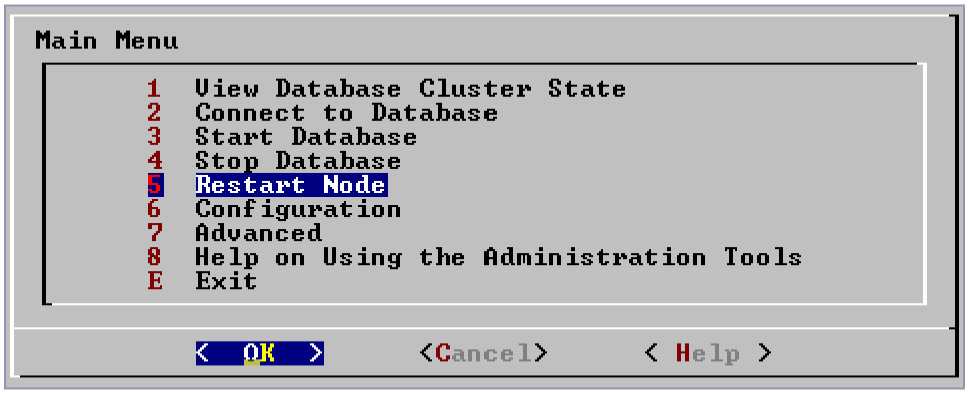
This tool restarts the Vertica process one or more nodes in a running database. Use this tool when a cluster host reboots while the database is running. The Spread daemon starts automatically but the Vertica process does not, thus the node does not automatically rejoin the cluster.

1. On the **Main Menu**, select **View Database Cluster State**



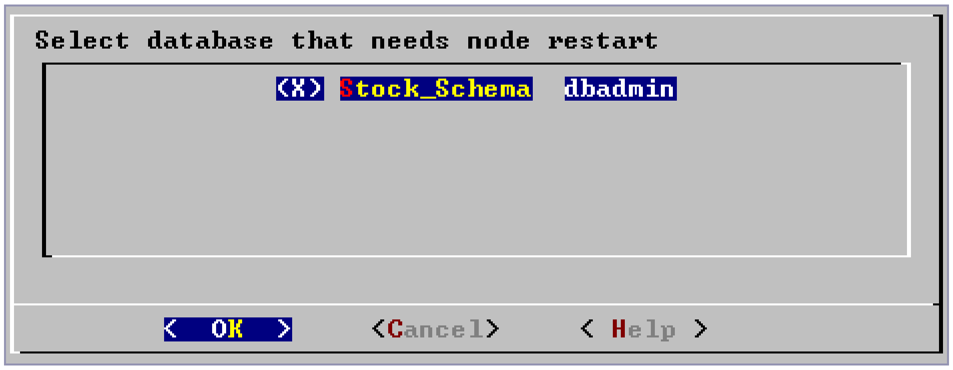
```
DB ____________________¦ Node _____________________¦ State
----------------------+-------------------------+-------
Clickstream_Schema___¦ ALL_______________________¦ DOWN__
CreditHistory_Schema_¦ ALL_______________________¦ DOWN__
Retail_Schema________¦ ALL_______________________¦ DOWN__
Stock_Schema_________¦ stock_schema_node1_host01_¦ UP____
Stock_Schema_________¦ stock_schema_node2_host02_¦ UP____
Stock_Schema_________¦ stock_schema_node3_host03_¦ DOWN__
Stock_Schema_________¦ stock_schema_node4_host04_¦ UP____
Telecom_Schema_______¦ ALL_______________________¦ DOWN__
```

2. If one or more nodes are down, select **Restart Node**.



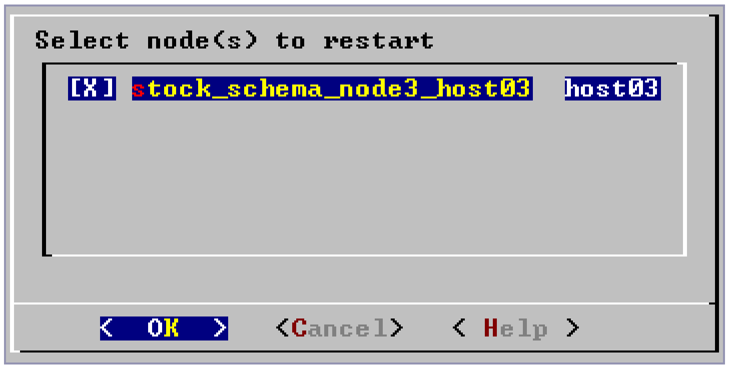
```
Main Menu
      1  View Database Cluster State
      2  Connect to Database
      3  Start Database
      4  Stop Database
      5  Restart Node
      6  Configuration
      7  Advanced
      8  Help on Using the Administration Tools
      E  Exit
```

3. **Select the database** that has a node needing a restart.



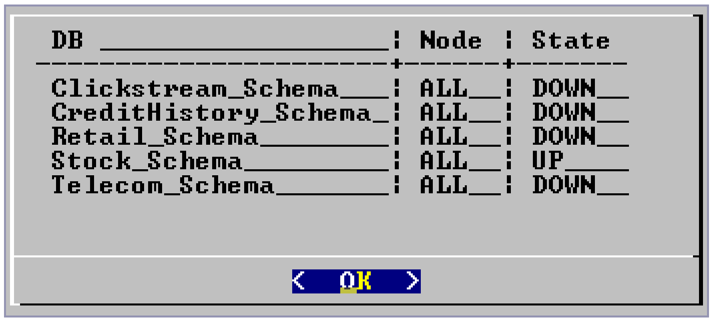
```
Select database that needs node restart
          (X) Stock_Schema  dbadmin
```

4. **Select the nodes** that require a restart.



```
Select node(s) to restart

[X] stock_schema_node3_host03   host03
```

5. Select **View Database Cluster State** again to make sure that all nodes are up.



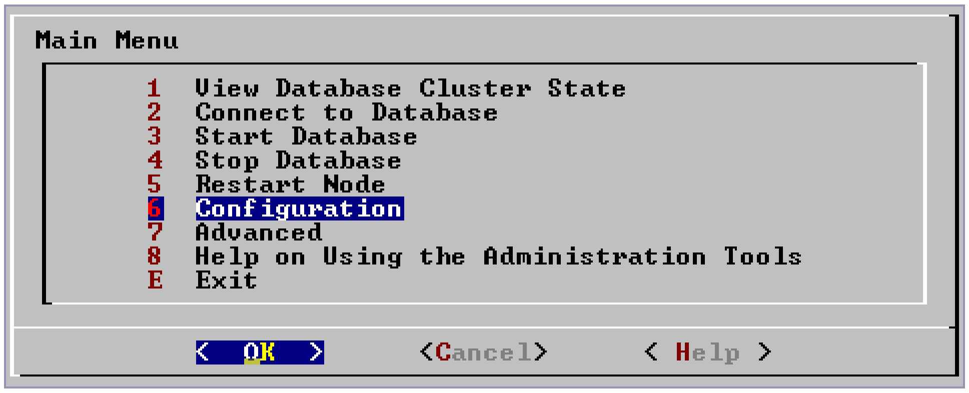
```
 DB ____________________| Node | State
-----------------------+------+-------
 Clickstream_Schema___| ALL__| DOWN__
 CreditHistory_Schema_| ALL__| DOWN__
 Retail_Schema________| ALL__| DOWN__
 Stock_Schema_________| ALL__| UP____
 Telecom_Schema_______| ALL__| DOWN__
```

Configuration

The main configuration menu allows you to:

- create, drop, and view databases
- use the Database Designer to create or modify a physical schema design

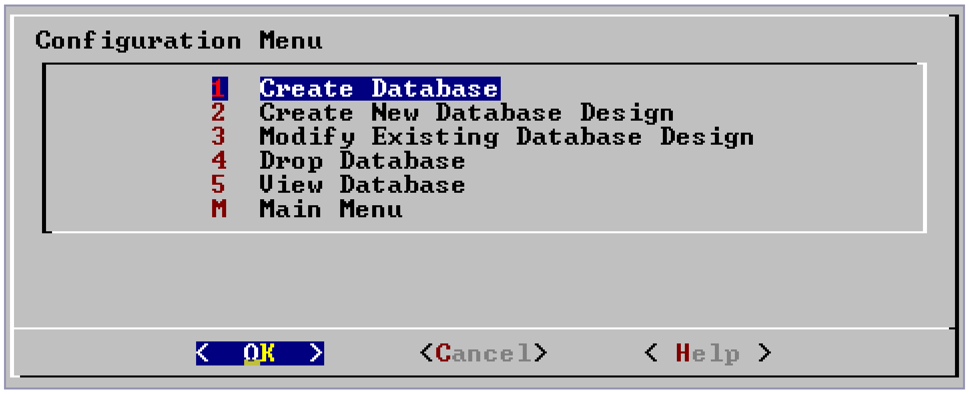
1. On the **Main Menu**, select **Configuration**.



```
Main Menu

      1  View Database Cluster State
      2  Connect to Database
      3  Start Database
      4  Stop Database
      5  Restart Node
      6  Configuration
      7  Advanced
      8  Help on Using the Administration Tools
      E  Exit
```

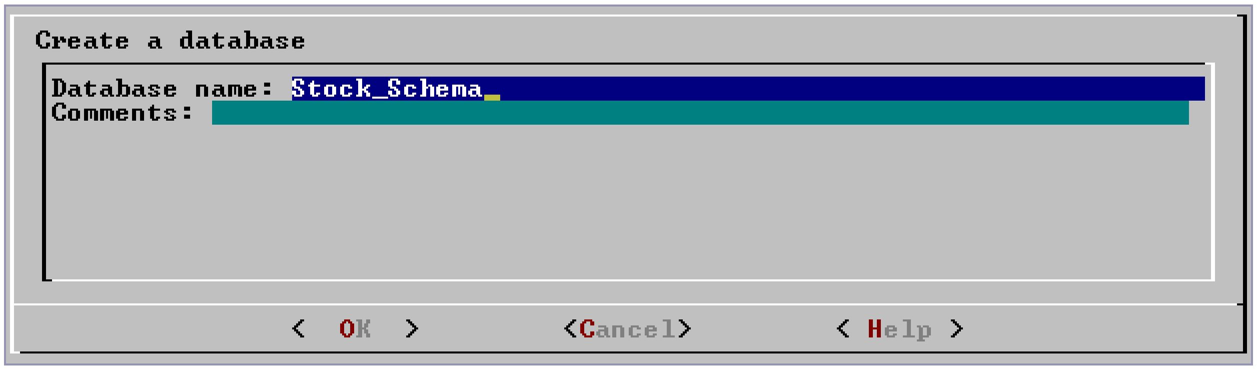
Create Database

1. On the **Configuration menu**, select **Create Database**.



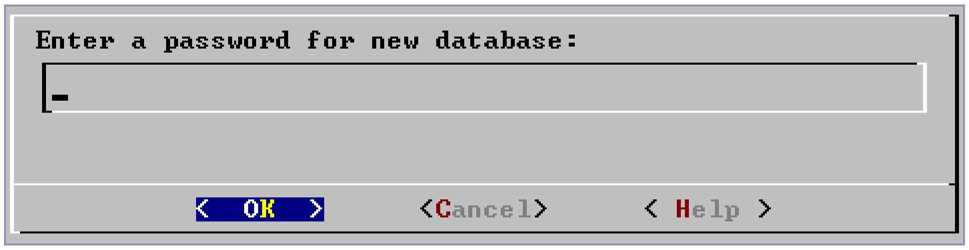
```
Configuration Menu

        1  Create Database
        2  Create New Database Design
        3  Modify Existing Database Design
        4  Drop Database
        5  View Database
        M  Main Menu

      <  OK  >      <Cancel>      < Help >
```

2. **Enter the name** of the database and an optional comment.



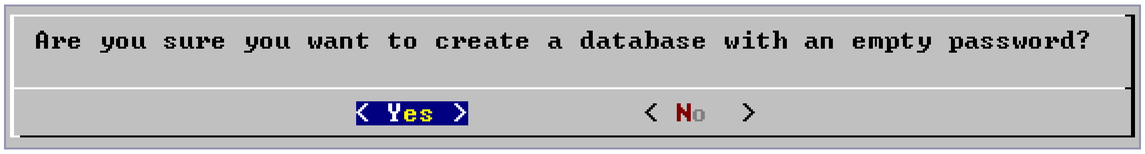
```
Create a database

Database name: Stock_Schema_
Comments:

      <  OK  >      <Cancel>      < Help >
```

3. **Enter the password**.



```
Enter a password for new database:

_

      <  OK  >      <Cancel>      < Help >
```

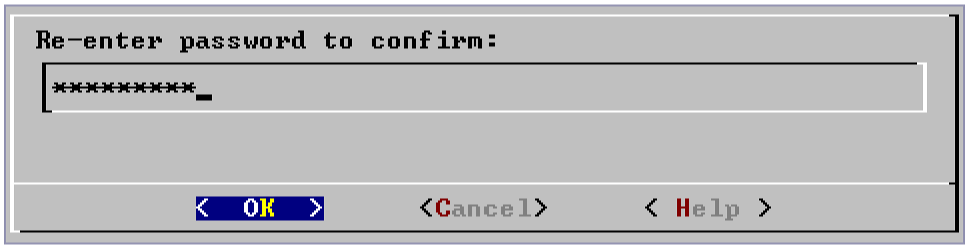
If you **do not enter a password**, the following dialog appears.



```
Are you sure you want to create a database with an empty password?

            < Yes >        < No  >
```

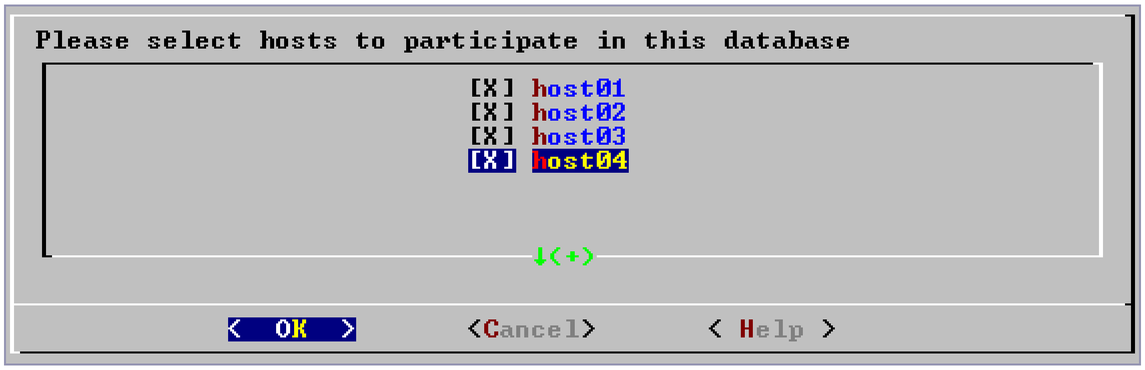
WARNING: If you do not enter a password at this point, the database is permanently set to trust authentication (no password required), in which ALTER USER cannot be used to change the superuser password. Unless the database is for evaluation or academic purposes, Vertica Systems, Inc. strongly recommends that you enter a superuser password.

4. If you entered a password **enter the password again**.



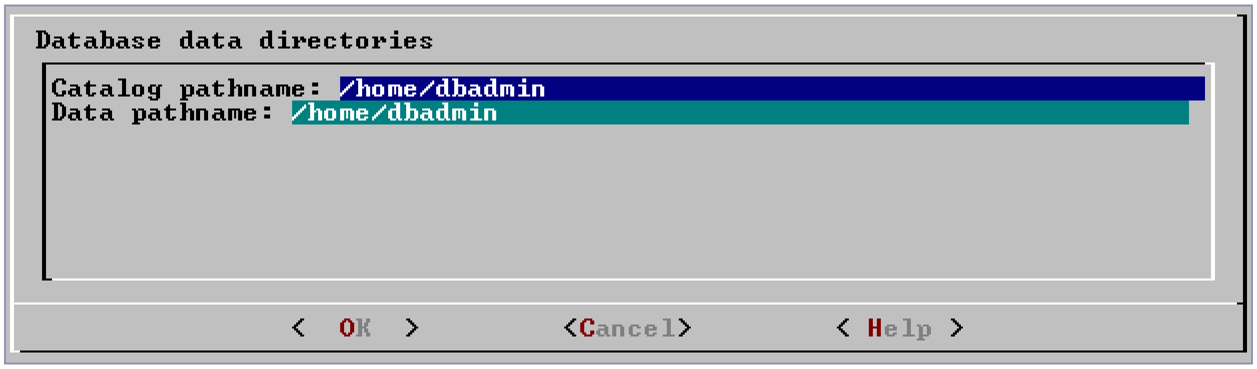
```
Re-enter password to confirm:
**********_
```

5. **Select the hosts** to include in the database. The hosts in this list are the ones that were specified at installation time (`install_vertica -s`).



```
Please select hosts to participate in this database
[X] host01
[X] host02
[X] host03
[X] host04
```

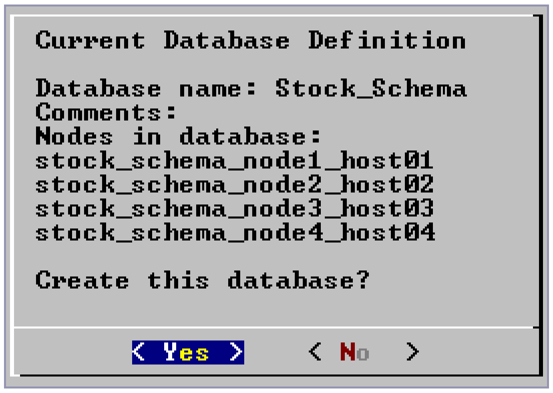
6. Specify the directories in which to store the data and catalog files.



```
Database data directories
Catalog pathname: /home/dbadmin
Data pathname: /home/dbadmin
```

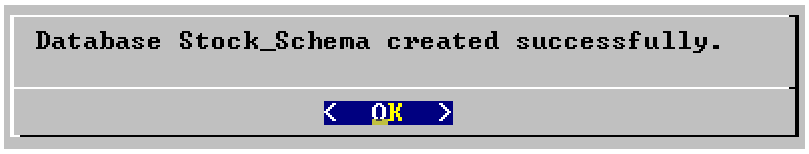
Catalog and data pathnames must contain only alphanumeric characters and cannot have leading space characters. Failure to comply with these restrictions may result in database creation failure.

7. Check the current database definition for correctness.



```
Current Database Definition

Database name: Stock_Schema
Comments:
Nodes in database:
stock_schema_node1_host01
stock_schema_node2_host02
stock_schema_node3_host03
stock_schema_node4_host04

Create this database?
```

8. This message indicates that you have successfully created a database.



```
Database Stock_Schema created successfully.
```

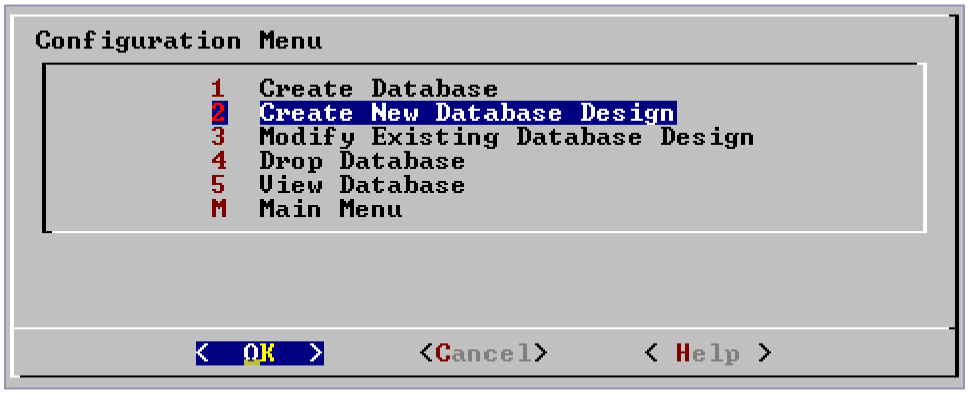
If you get an error message, see:

- ***Startup Problems*** (page 102)
- Node Failures

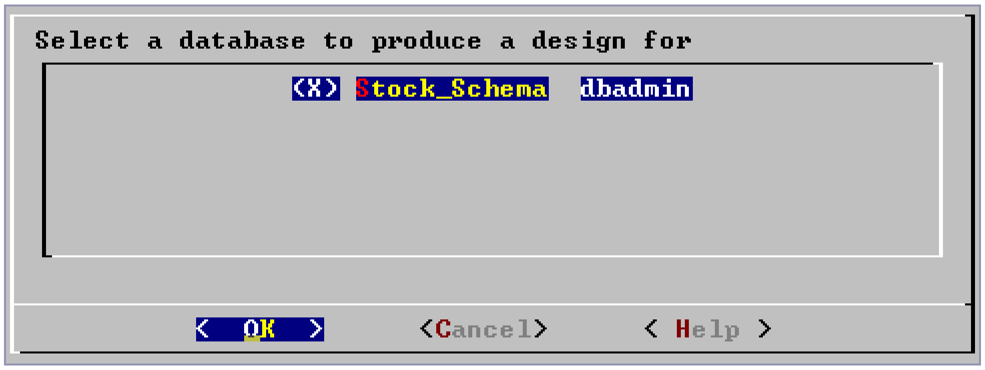
Create New Database Design

This tool runs the Database Designer.

1. On the **Configuration Menu**, select **Create New Database Design**.

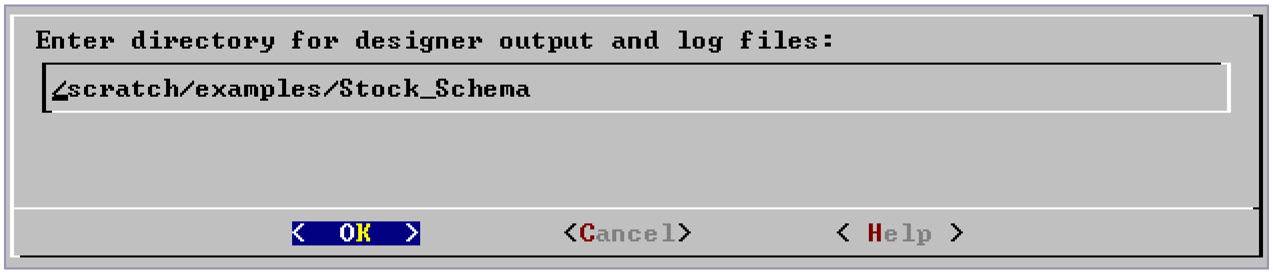


2. **Select the database** on which to run the Database Designer. The default is the currently active database.



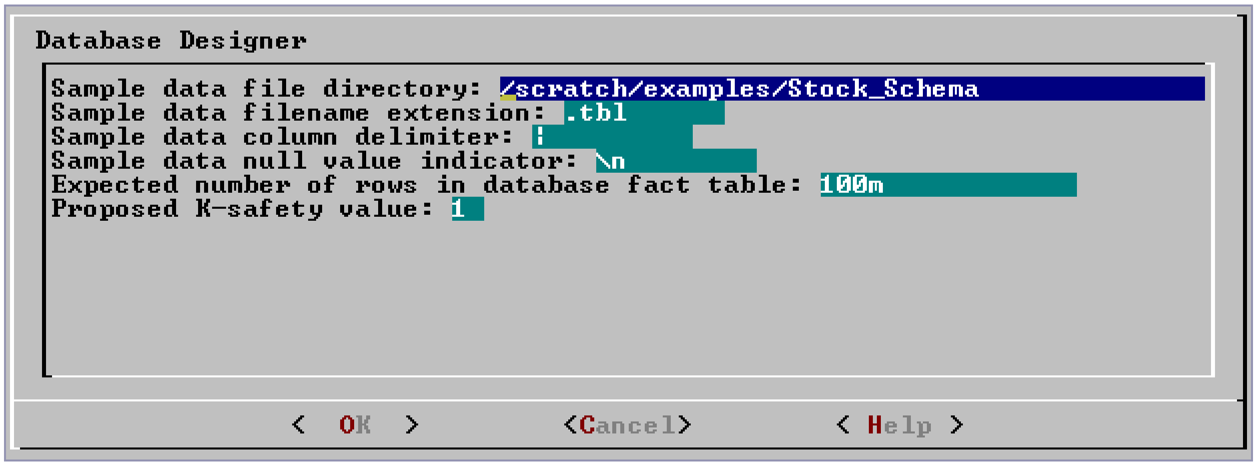
```
Select a database to produce a design for

          (X) Stock_Schema   dbadmin

   <  OK  >        <Cancel>       < Help >
```

3. **Specify the pathname** for the Database Designer physical schema design script and log files.



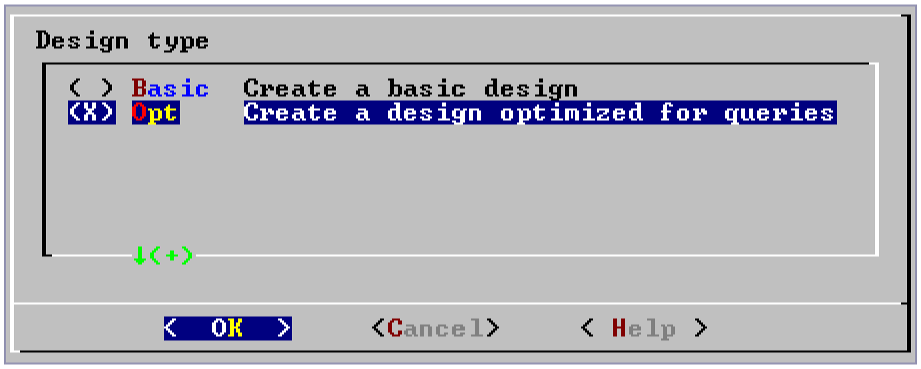
```
Enter directory for designer output and log files:
/scratch/examples/Stock_Schema

          <  OK  >          <Cancel>          < Help >
```

4. Set the Database Designer **parameters**. See the ***Database Designer Dialog*** (on page 86) for detailed descriptions of each input parameter. See ***Using the Graphical User Interface*** (page 70) for information about using form dialogs.



```
Database Designer
Sample data file directory: /scratch/examples/Stock_Schema
Sample data filename extension: .tbl
Sample data column delimiter: |
Sample data null value indicator: \n
Expected number of rows in database fact table: 100m
Proposed K-safety value: 1

          <  OK  >          <Cancel>          < Help >
```

5. **Choose** a Design type.

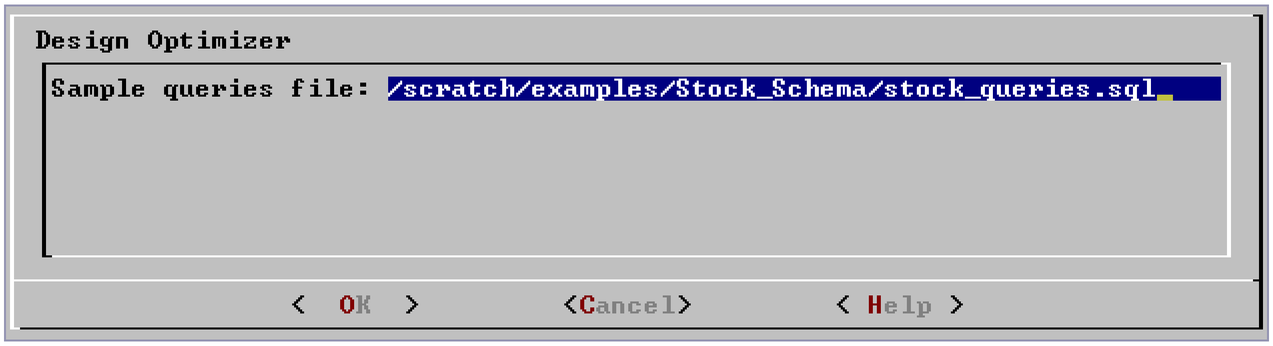


```
Design type

 ( ) Basic   Create a basic design
 (X) Opt     Create a design optimized for queries
```

Create a basic design produces a design that consists only of single-table superprojections.

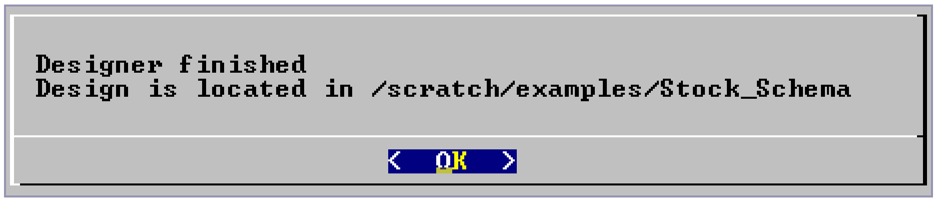
Create a design optimized for queries produces a design that includes pre-join projections based on the sample queries you provide.

6. If you chose to create an **optimized design**, enter the pathname of the file containing your **sample queries** (as described in ***Prepare a Sample Query Script*** (page 26)).



```
Design Optimizer

 Sample queries file: /scratch/examples/Stock_Schema/stock_queries.sql
```

7. Several minutes may pass before the following **success message** appears.



```
Designer finished
Design is located in /scratch/examples/Stock_Schema
```

8. See ***Using the Database Designer*** (page 45) for information about examining the designer output files.

Database Designer Dialog

This section provides a detailed description of the Database Designer dialog.

Although some of the parameters are the same, the syntax of the Database Designer input parameters is different from that of the COPY command. You do not have to specify doubled backslash characters as the column delimiter or null value indicator for the Database Designer.

Sample data file directory:

Default: your current directory

Enter the absolute or relative pathname of the directory containing your sample data files (relative to the current directory when you ran the Administration Tools).

Name each sample data file to match the corresponding table in the logical schema. Case does not matter. For example, if a table is named Stock_Dimension, name the corresponding data file `stock_dimension.tbl`.

When using multiple data files for a fact table, append *_nnn* (where `nnn` is a positive integer in the range 001 to 999) to the filename (not the extension). For example: `stock_dimension_001.tbl`, `stock_dimension_002.tbl`, and so forth. In that case, the Database Designer samples only the lowest numbered file.

If the sample fact table data file contains fewer than 100K rows, the Database Designer samples all rows. Otherwise, the Database Designer does up to 1000 uniform seeks reading blocks of 100 rows at a time. The sampling is uniformly distributed to avoid skew problems.

Sample data filename extension:

Default: .tbl

Enter the filename extension of the sample data files. The default is the extension used by the sample data files in the Quick Start.

Sample data column delimiter:

Default: vertical bar (|)

Enter the single character used to delimit column values in the sample data file. The default (vertical bar) is the delimiter used in the sample data files described in the Quick Start. See the COPY command in the SQL Reference Manual for details about specifying a delimiter.

Do not use the backslash character (\) as a delimiter.

Sample data null value indicator:

Default: backslash lowercase en (\n)

Enter the character string that represents a null value in your sample data file. If you specify an empty field (delete the default \n value), the null data value indicator is two consecutive column delimiters.

The default null indicator in the Database Designer (\n) is the same as the null indicator used in the sample data files described in the Quick Start. See the COPY command in the SQL Reference Manual for details about specifying null values.

Expected number of rows in database fact table:

Default: 10M

Enter the maximum number of rows that will be stored in the actual fact table at any given time. You can use the letters K and M to mean thousands and millions, respectively. Case does not matter.

The number you specify here is used for cost benefit analysis and does not have to be perfectly accurate.

If you intend to specify a disk space budget (see below), choose a number in the middle of the possible range of values.

If the disk space budget is unlimited choose the bottom of the range.

For example, if the fact table will contain 200M to 300M rows, specify 250M if using a space budget; 200M if unlimited.

Proposed K-safety value

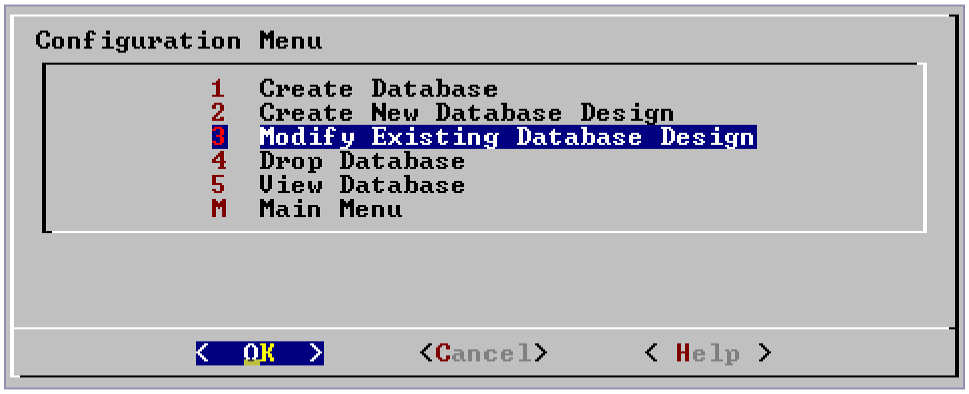
Default: 1

Enter 0 for a projection script that contains only the projections needed for a functional database. Enter 1 for a projection script that includes the replicated dimension tables and buddy projections required for a database with K=1 (a database that can function with one node down).

Modify Existing Database Design

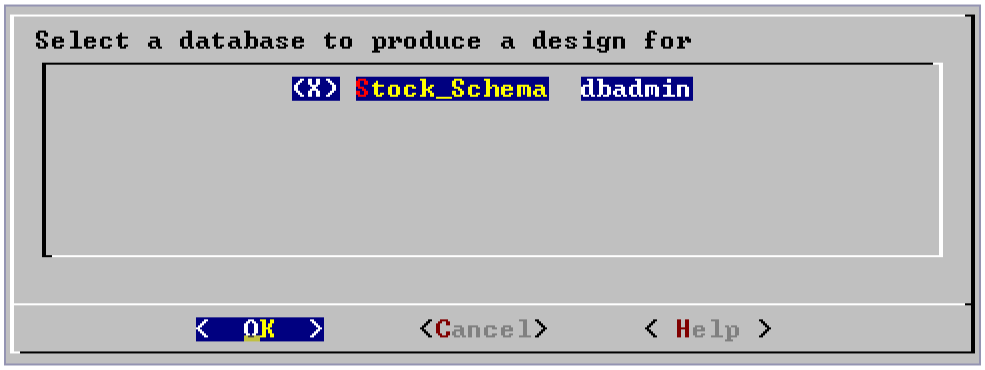
This tool runs the Database Designer.

1. On the **Configuration Menu**, select **Modify Existing Database Design**.



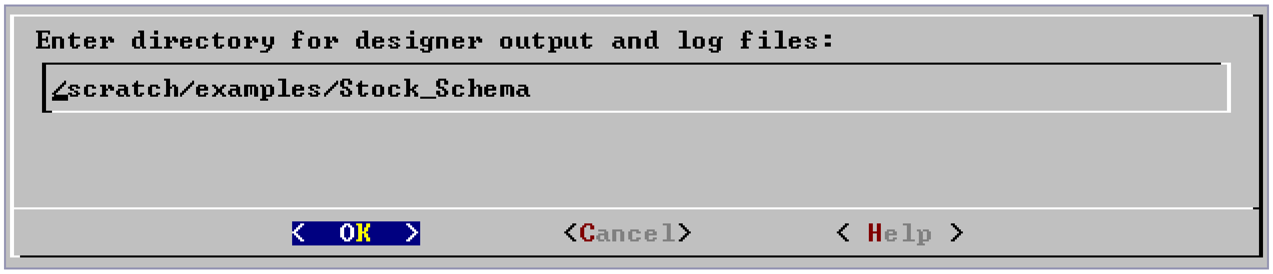
```
Configuration Menu

            1   Create Database
            2   Create New Database Design
            3   Modify Existing Database Design
            4   Drop Database
            5   View Database
            M   Main Menu

        <  OK  >      <Cancel>      < Help >
```

2. **Select the database** on which to run the Database Designer. The default is the currently active database.



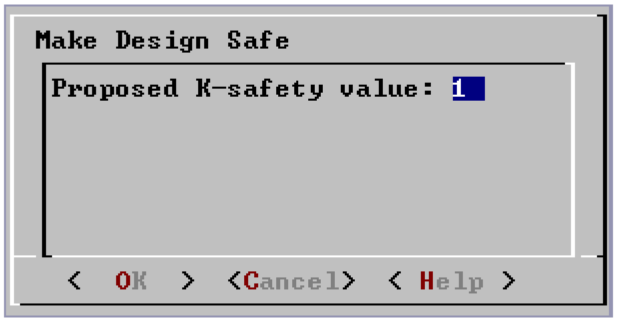
```
Select a database to produce a design for

                (X) Stock_Schema   dbadmin

        <  OK  >      <Cancel>      < Help >
```

3. **Specify the pathname** for the Database Designer physical schema design script and log files.



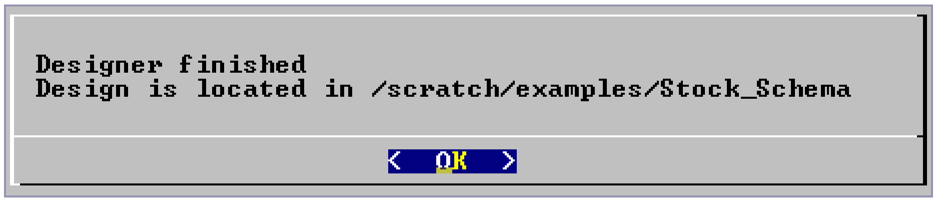
```
Enter directory for designer output and log files:
/scratch/examples/Stock_Schema
```

4. Enter the desired **K-safety value** (1 in Vertica V2.0).



```
Make Design Safe
Proposed K-safety value: 1
```

5. Several minutes may pass before the following **success message** appears.



```
Designer finished
Design is located in /scratch/examples/Stock_Schema
```

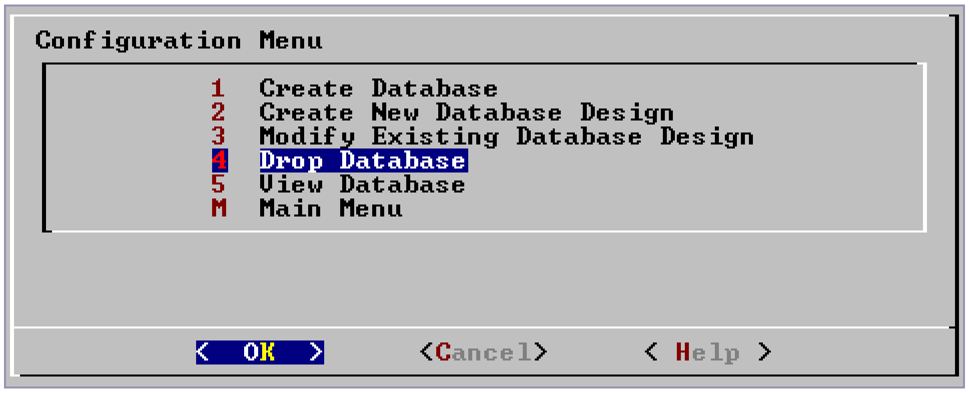
6. See ***Using the Database Designer*** (page 45) for information about examining the designer output files.

Drop Database

This tool drops an existing database. Only the Database Administrator is allowed to drop a database.

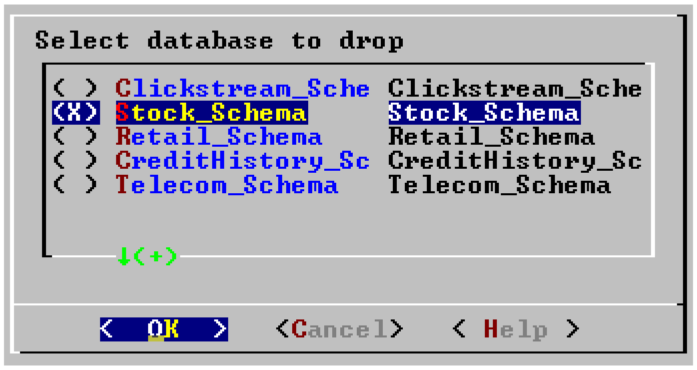
1. Stop the database as described in ***Stop Database*** (page 78).

2. On the **Configure Database** dialog, select **Drop Database**.



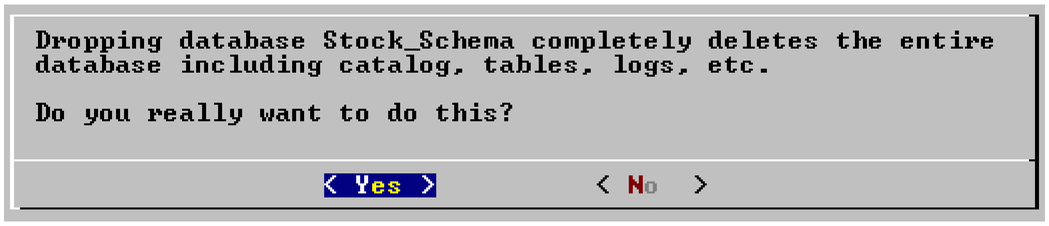
```
Configuration Menu

            1   Create Database
            2   Create New Database Design
            3   Modify Existing Database Design
            4   Drop Database
            5   View Database
            M   Main Menu

          <  OK  >      <Cancel>      < Help >
```

3. **Select** the database to drop.



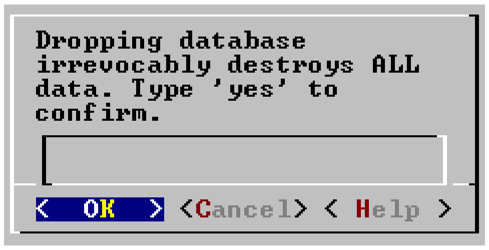
```
Select database to drop

( ) Clickstream_Sche  Clickstream_Sche
(X) Stock_Schema      Stock_Schema
( ) Retail_Schema     Retail_Schema
( ) CreditHistory_Sc  CreditHistory_Sc
( ) Telecom_Schema    Telecom_Schema

      ↓(+)

     <  OK  >   <Cancel>   < Help >
```

4. **Confirm** that you really want to drop the database.



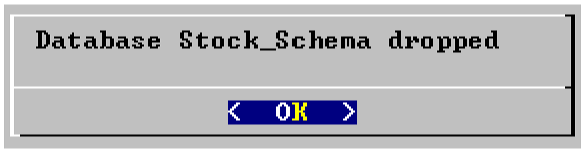
```
Dropping database Stock_Schema completely deletes the entire
database including catalog, tables, logs, etc.

Do you really want to do this?

                < Yes >            < No   >
```

5. **Type yes** to confirm again that you really want to drop the database.



```
Dropping database
irrevocably destroys ALL
data. Type 'yes' to
confirm.

<  OK  > <Cancel> < Help >
```

6. This message indicates that you have successfully dropped the database.

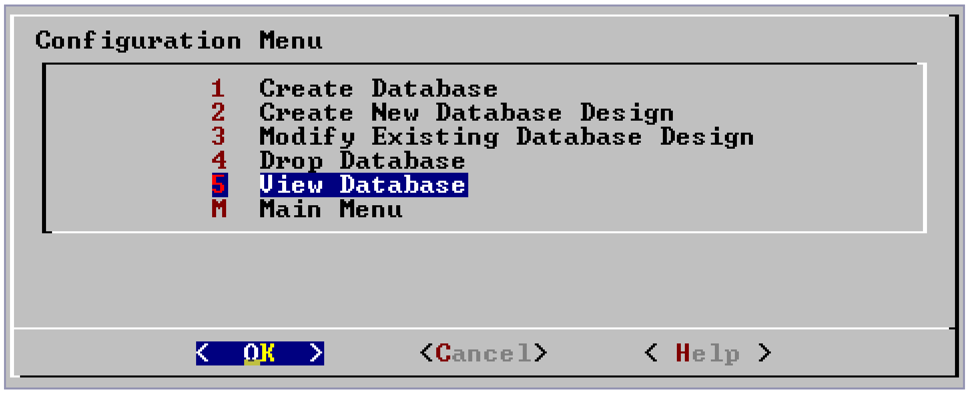


```
Database Stock_Schema dropped

< OK >
```

View Database

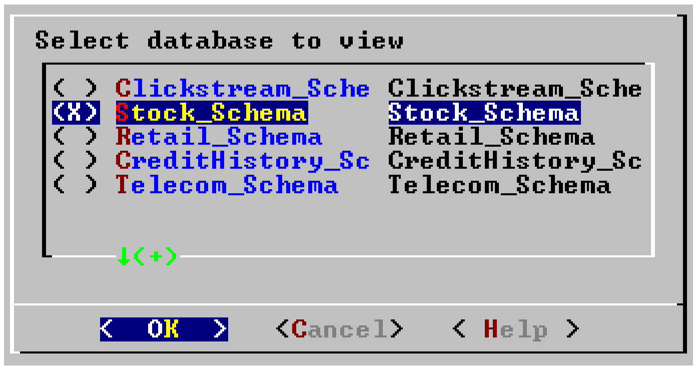
This tool displays the characteristics of an existing database.

1. On the **Configure Database** dialog, select **View Database**.



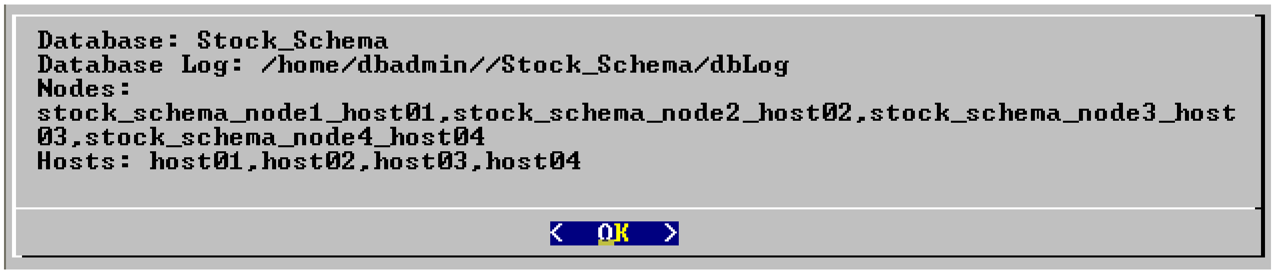
```
Configuration Menu

          1   Create Database
          2   Create New Database Design
          3   Modify Existing Database Design
          4   Drop Database
          5   View Database
          M   Main Menu

     <  OK  >      <Cancel>      < Help >
```

2. **Select the database** to view.



```
Select database to view

( ) Clickstream_Sche  Clickstream_Sche
(X) Stock_Schema      Stock_Schema
( ) Retail_Schema     Retail_Schema
( ) CreditHistory_Sc  CreditHistory_Sc
( ) Telecom_Schema    Telecom_Schema

    ↓(+)

  <  OK  >    <Cancel>    < Help >
```

3. Vertica **displays** the database information.



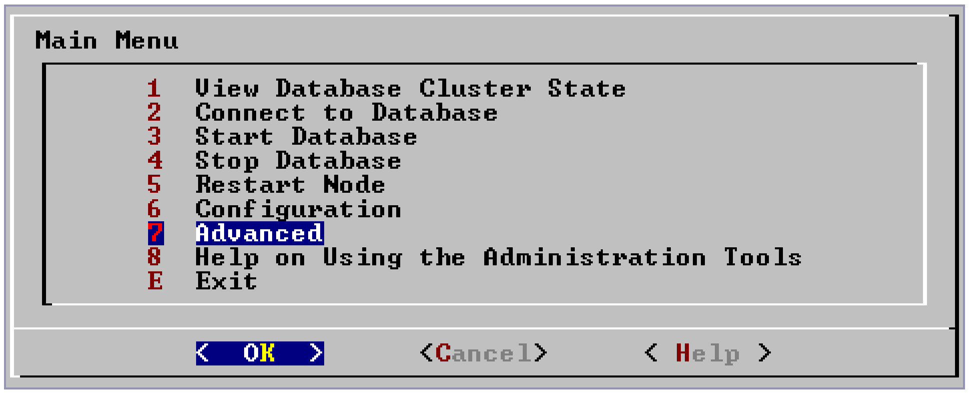
```
Database: Stock_Schema
Database Log: /home/dbadmin//Stock_Schema/dbLog
Nodes:
stock_schema_node1_host01,stock_schema_node2_host02,stock_schema_node3_host
03,stock_schema_node4_host04
Hosts: host01,host02,host03,host04

< OK >
```

Advanced

This tool provides interactive recovery and repair commands.

Use these commands only when instructed to do so by ***Technical Support*** (on page 11).

1. On the **Main Menu**, select **Advanced**.



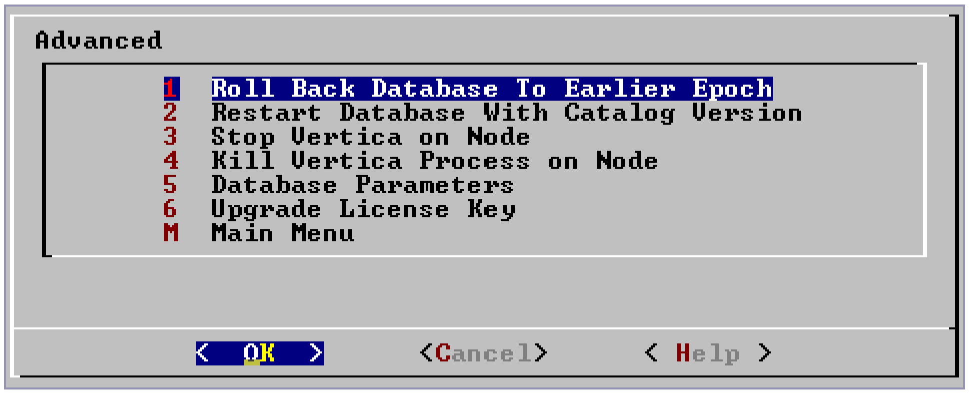
```
Main Menu

      1  View Database Cluster State
      2  Connect to Database
      3  Start Database
      4  Stop Database
      5  Restart Node
      6  Configuration
      7  Advanced
      8  Help on Using the Administration Tools
      E  Exit

      <  OK  >      <Cancel>      < Help >
```

Roll Back Database to Earlier Epoch

Use this command only when instructed to do so by ***Technical Support*** (on page 11).

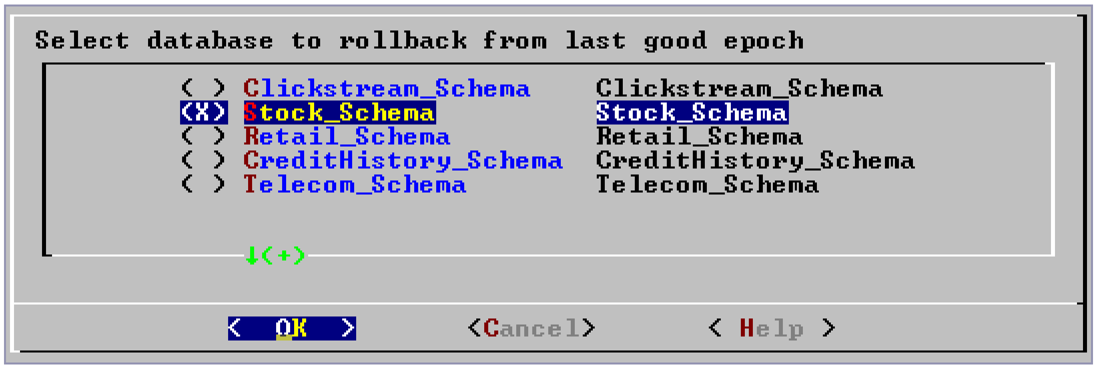
Vertica provides the ability to roll the entire database back to a specific epoch primarily to assist in the correction of human errors during data loads or other accidental corruptions. For example, suppose that you have been performing a bulk load and the cluster went down during a particular COPY command. You may wish to discard all epochs back to the point at which the previous COPY command committed and re-execute the one that did not finish. You can determine that point by examining the log files (see ***Monitoring the Log Files*** (page 53)).

1. On the **Advanced** menu, select **Roll Back Database To Earlier Epoch**.



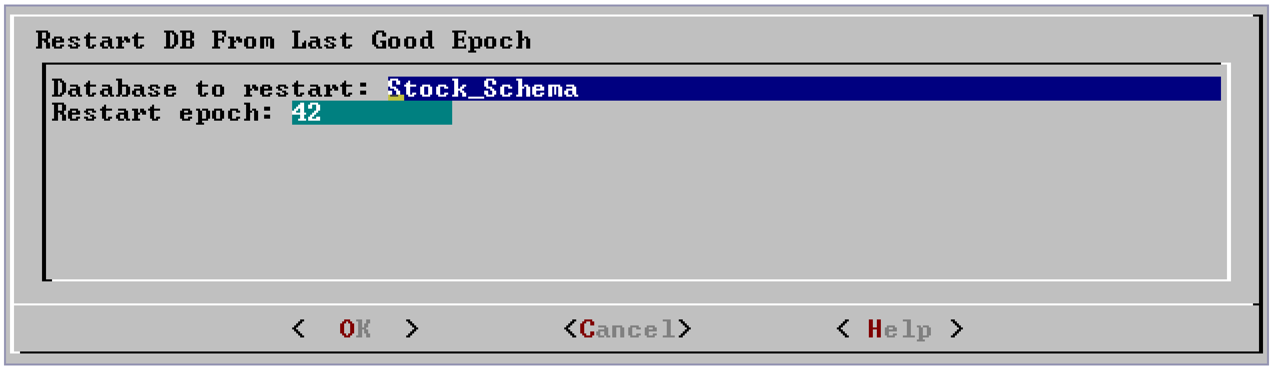
```
Advanced

       1  Roll Back Database To Earlier Epoch
       2  Restart Database With Catalog Version
       3  Stop Vertica on Node
       4  Kill Vertica Process on Node
       5  Database Parameters
       6  Upgrade License Key
       M  Main Menu

      <  OK  >      <Cancel>      < Help >
```

2. **Select** the database to roll back. The database must be stopped.



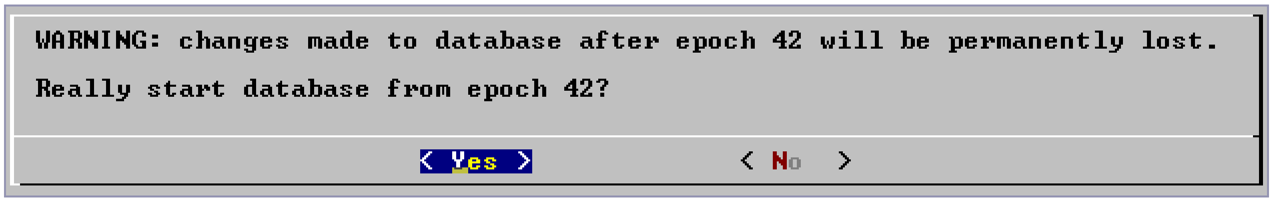
```
Select database to rollback from last good epoch

        ( ) Clickstream_Schema    Clickstream_Schema
        (X) Stock_Schema          Stock_Schema
        ( ) Retail_Schema         Retail_Schema
        ( ) CreditHistory_Schema  CreditHistory_Schema
        ( ) Telecom_Schema        Telecom_Schema

            ↓(+)

        <  OK  >      <Cancel>      < Help >
```

3. **Accept the suggested restart epoch** or specify a different one.



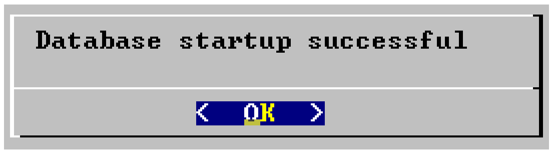
```
Restart DB From Last Good Epoch

Database to restart: Stock_Schema
Restart epoch: 42

        <  OK  >      <Cancel>      < Help >
```

4. **Confirm** that you want to discard the changes after the specified epoch.



```
WARNING: changes made to database after epoch 42 will be permanently lost.

Really start database from epoch 42?

            < Yes >           < No  >
```

5. The database restarts successfully.



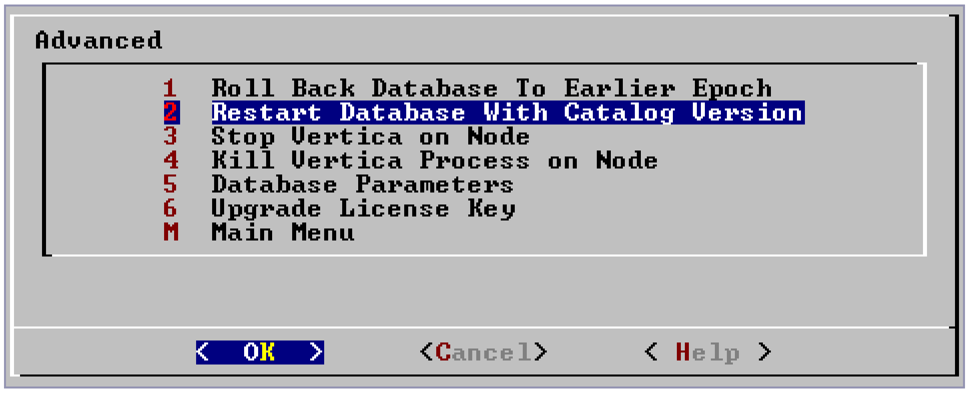
```
Database startup successful

        <  OK  >
```

Restart Database With Catalog Version

Use this command only when instructed to do so by ***Technical Support*** (on page 11).

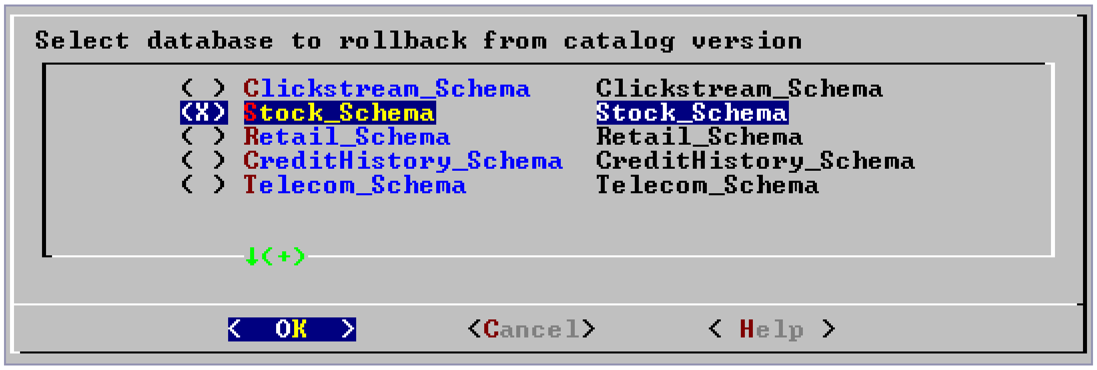
This command discards changes made to the catalog back to a specific version. When a database starts up, all instances of Vertica must agree on a common catalog version. The dialog below appears when one or more nodes cannot agree on a catalog version, a situation that can occur when a cluster cannot shut down safely. The command allows you to discard versions of the catalog that did not get replicated on all nodes.

1. On the **Advanced** menu select **Roll Back Database To Catalog Version**.



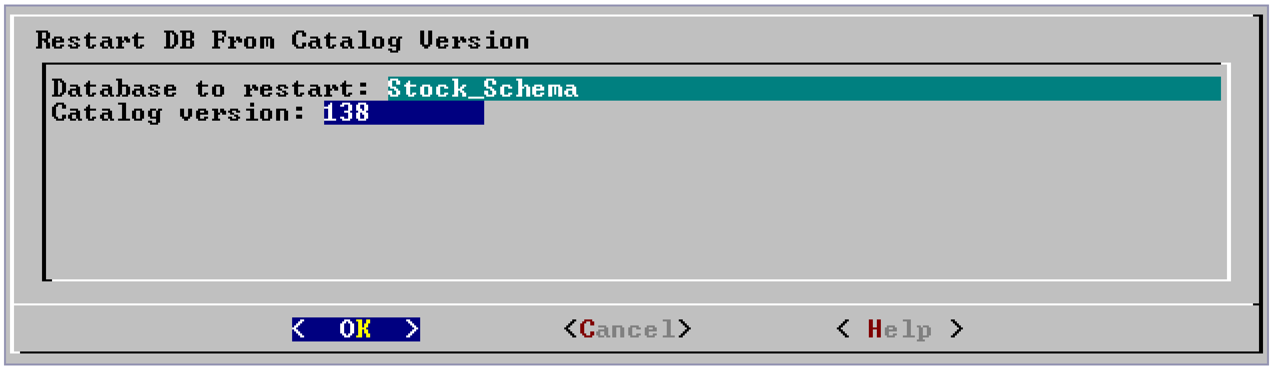
```
Advanced

        1  Roll Back Database To Earlier Epoch
        2  Restart Database With Catalog Version
        3  Stop Vertica on Node
        4  Kill Vertica Process on Node
        5  Database Parameters
        6  Upgrade License Key
        M  Main Menu
```

2. **Select** the database to roll back. The database must be stopped.



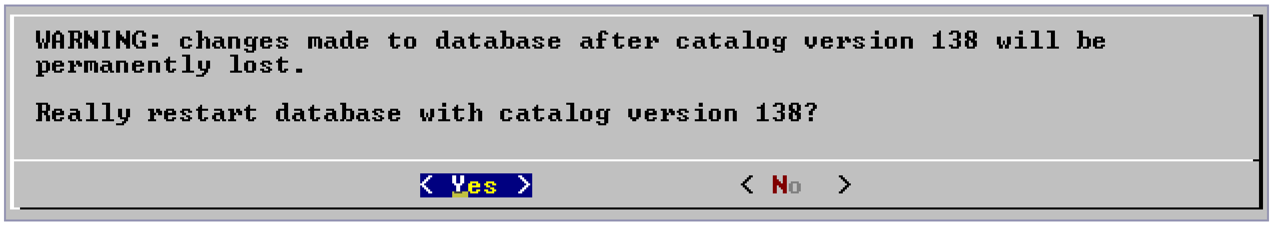
```
Select database to rollback from catalog version

        ( ) Clickstream_Schema     Clickstream_Schema
        (X) Stock_Schema           Stock_Schema
        ( ) Retail_Schema          Retail_Schema
        ( ) CreditHistory_Schema   CreditHistory_Schema
        ( ) Telecom_Schema         Telecom_Schema
```

3. **Click OK unless instructed to specify a catalog version number** by ***Technical Support*** (on page 11).



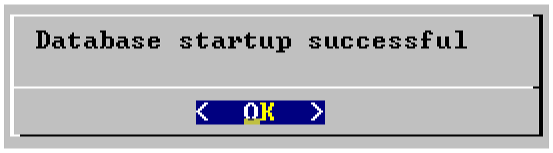
```
Restart DB From Catalog Version

 Database to restart: Stock_Schema
 Catalog version: 138
```

4. Confirm that you really want to discard the catalog changes made after the specified version.



```
WARNING: changes made to database after catalog version 138 will be
permanently lost.

Really restart database with catalog version 138?
```

5. The database restarts successfully.



```
Database startup successful

            <  OK  >
```

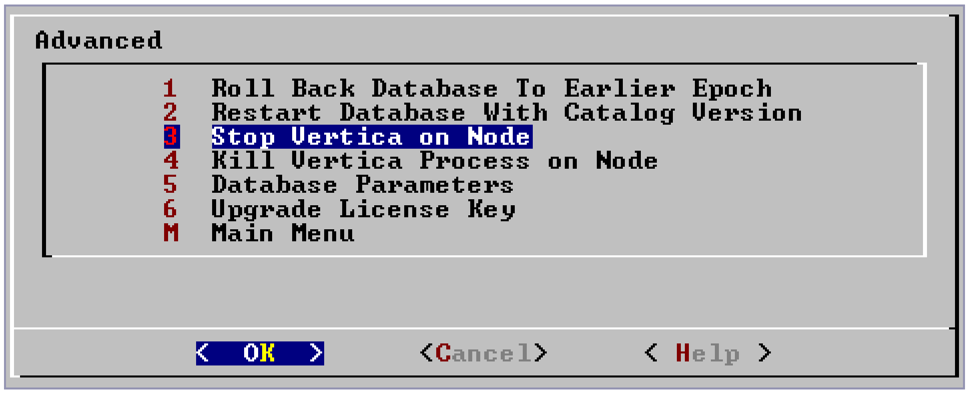
If the restart fails, you may have specified a catalog version that is not known to any node.

Stop Vertica on Node

This command attempts to gracefully shut down the Vertica process on a node.

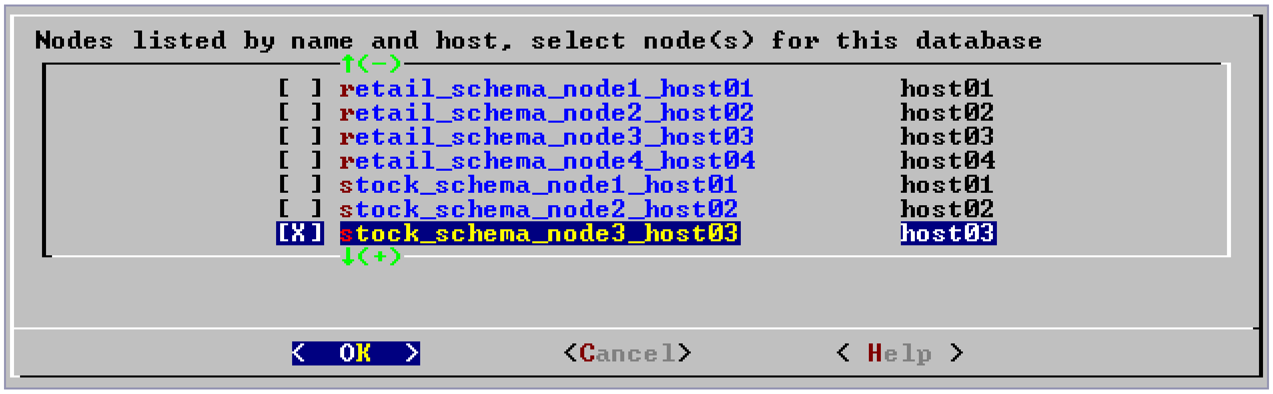
Do not use this command unless ***Stop Database*** (page 78) was unsuccessful.

1. On the **Advanced** menu, select **Stop Vertica on node**.



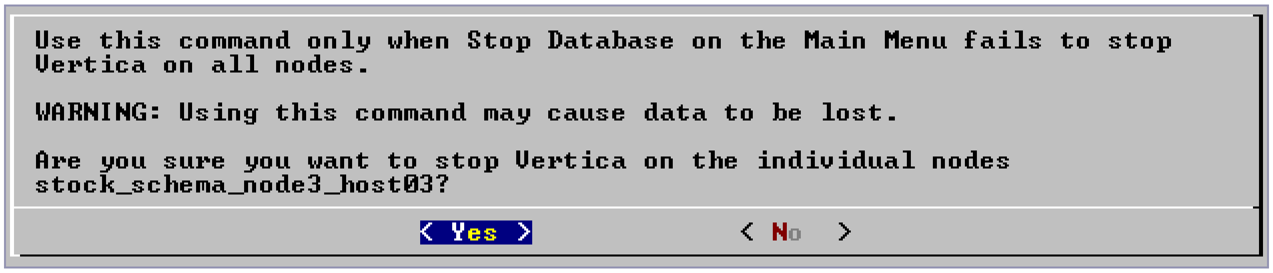
```
Advanced

          1  Roll Back Database To Earlier Epoch
          2  Restart Database With Catalog Version
          3  Stop Vertica on Node
          4  Kill Vertica Process on Node
          5  Database Parameters
          6  Upgrade License Key
          M  Main Menu

        <  OK  >       <Cancel>       < Help >
```

2. **Select the nodes** to stop.



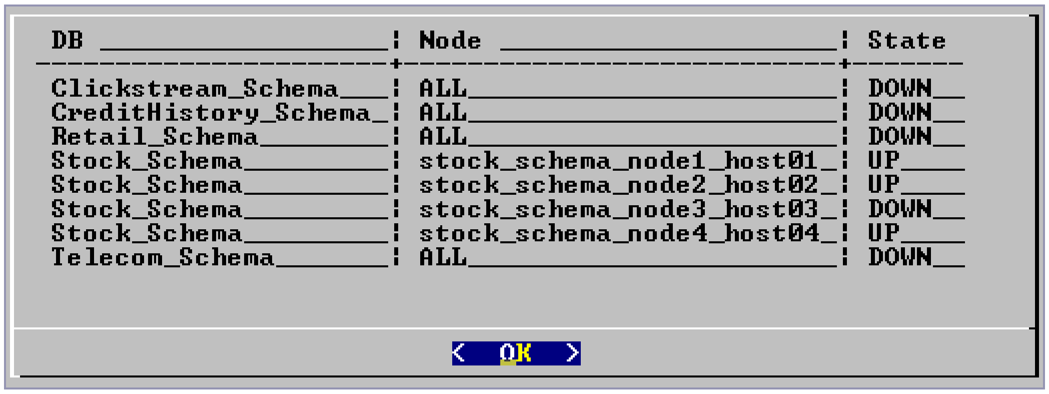
```
Nodes listed by name and host, select node(s) for this database
                     ↑(-)
                [ ] retail_schema_node1_host01         host01
                [ ] retail_schema_node2_host02         host02
                [ ] retail_schema_node3_host03         host03
                [ ] retail_schema_node4_host04         host04
                [ ] stock_schema_node1_host01          host01
                [ ] stock_schema_node2_host02          host02
                [X] stock_schema_node3_host03          host03
                     ↓(+)

             <  OK  >       <Cancel>       < Help >
```

3. **Confirm** that you want to stop the nodes.



```
Use this command only when Stop Database on the Main Menu fails to stop
Vertica on all nodes.

WARNING: Using this command may cause data to be lost.

Are you sure you want to stop Vertica on the individual nodes
stock_schema_node3_host03?
```

4. If the command succeeds ***View Database Cluster State*** (page 75) shows that the selected nodes are DOWN.



```
 DB _____________________| Node _____________________| State
------------------------+----------------------------+-------
 Clickstream_Schema___| ALL_______________________| DOWN__
 CreditHistory_Schema_| ALL_______________________| DOWN__
 Retail_Schema________| ALL_______________________| DOWN__
 Stock_Schema_________| stock_schema_node1_host01_| UP____
 Stock_Schema_________| stock_schema_node2_host02_| UP____
 Stock_Schema_________| stock_schema_node3_host03_| DOWN__
 Stock_Schema_________| stock_schema_node4_host04_| UP____
 Telecom_Schema_______| ALL_______________________| DOWN__
```

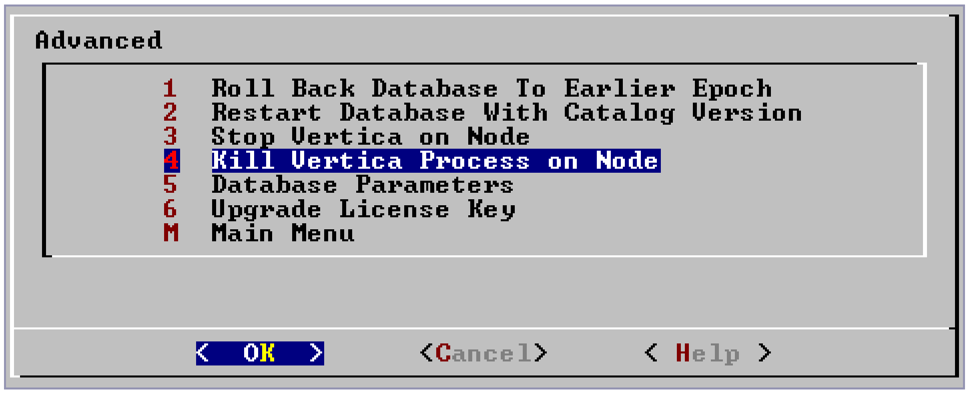
If the command fails to stop any selected nodes, proceed to ***Kill Vertica Process on Node*** (page 96).

Kill Vertica Process on Node

This command sends a kill signal to the Vertica process on a node.

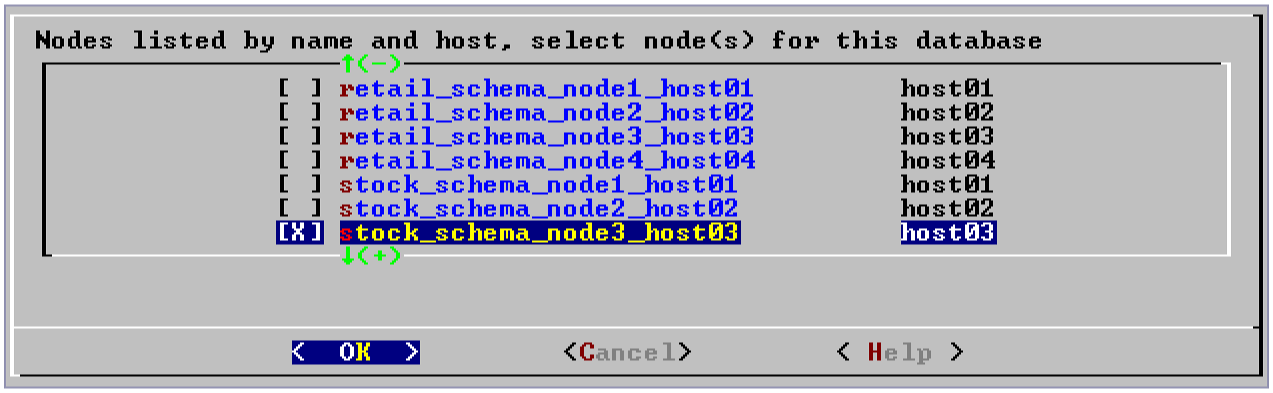
Do not use this command unless you have already tried ***Stop Database*** (page 78) and ***Stop Vertica on Node*** (page 95) and both were unsuccessful.

1. On the **Advanced** menu, select **Kill Vertica Process on node**.



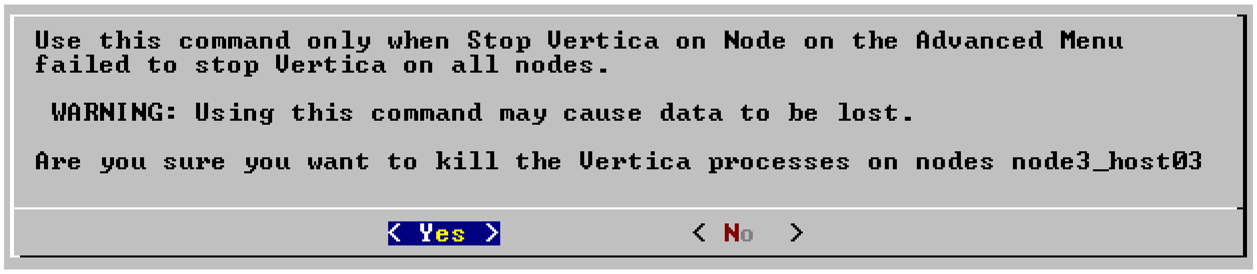
Advanced
1 Roll Back Database To Earlier Epoch
2 Restart Database With Catalog Version
3 Stop Vertica on Node
4 Kill Vertica Process on Node
5 Database Parameters
6 Upgrade License Key
M Main Menu
< OK >
<Cancel>
< Help >

2. Select the nodes on which to kill the Vertica process.



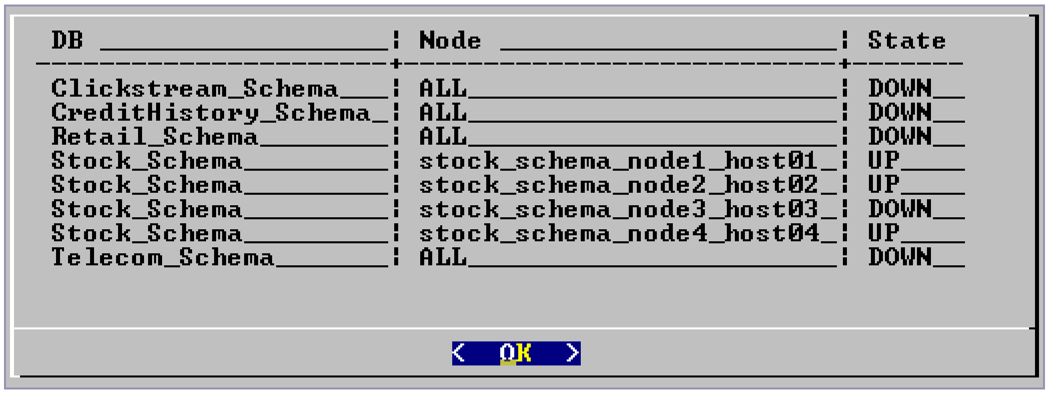
```
Nodes listed by name and host, select node(s) for this database
                        ↑(-)
          [ ] retail_schema_node1_host01          host01
          [ ] retail_schema_node2_host02          host02
          [ ] retail_schema_node3_host03          host03
          [ ] retail_schema_node4_host04          host04
          [ ] stock_schema_node1_host01           host01
          [ ] stock_schema_node2_host02           host02
          [X] stock_schema_node3_host03           host03
                        ↓(+)

            <  OK  >          <Cancel>          < Help >
```

3. Confirm that you want to kill the processes.



```
Use this command only when Stop Vertica on Node on the Advanced Menu
failed to stop Vertica on all nodes.

 WARNING: Using this command may cause data to be lost.

Are you sure you want to kill the Vertica processes on nodes node3_host03

                  < Yes >            < No  >
```

4. If the command succeeds ***View Database Cluster State*** (page 75) shows that the selected nodes are DOWN.



```
 DB ____________________| Node ____________________| State
-----------------------+--------------------------+-------
 Clickstream_Schema___| ALL_______________________| DOWN__
 CreditHistory_Schema_| ALL_______________________| DOWN__
 Retail_Schema________| ALL_______________________| DOWN__
 Stock_Schema_________| stock_schema_node1_host01_| UP____
 Stock_Schema_________| stock_schema_node2_host02_| UP____
 Stock_Schema_________| stock_schema_node3_host03_| DOWN__
 Stock_Schema_________| stock_schema_node4_host04_| UP____
 Telecom_Schema_______| ALL_______________________| DOWN__

                       <  OK  >
```

If the command fails to kill any selected nodes, see ***Shutdown Problems*** (page 100).

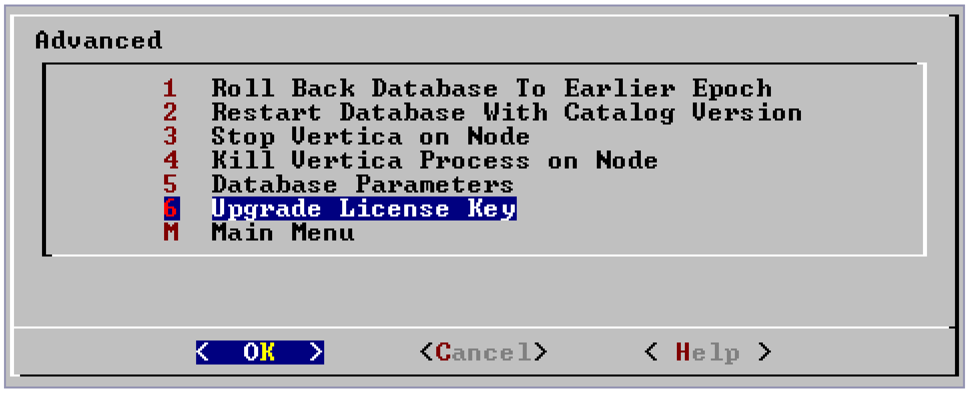
Database Parameters

This command is for ***Technical Support*** (on page 11) purposes only. Use it only when requested by a support representative.

Upgrade License Key

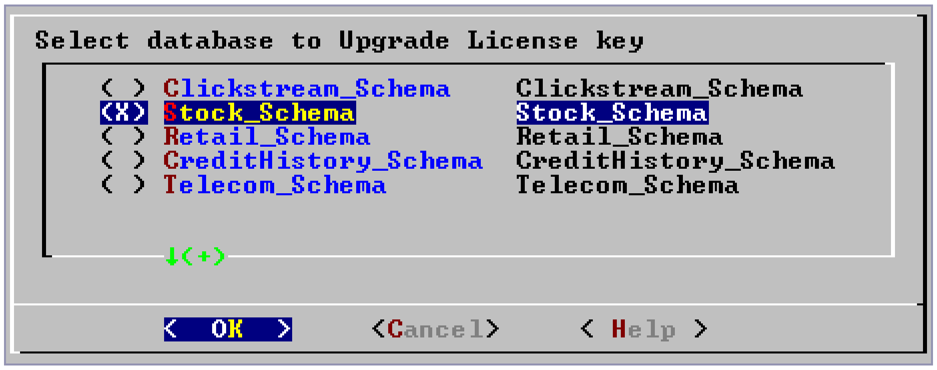
This command copies a license key file into the database. See ***Managing Your License Key*** (page 49) for more information.

1. On the **Advanced** menu select **Upgrade License Key**.



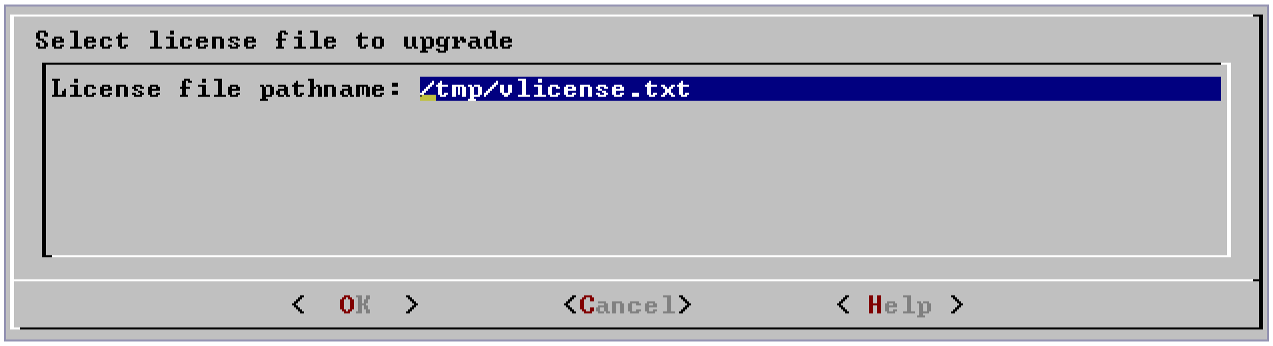
```
Advanced

        1  Roll Back Database To Earlier Epoch
        2  Restart Database With Catalog Version
        3  Stop Vertica on Node
        4  Kill Vertica Process on Node
        5  Database Parameters
        6  Upgrade License Key
        M  Main Menu

     <  OK  >      <Cancel>      < Help >
```

2. **Select the database** for which to upgrade the license key.



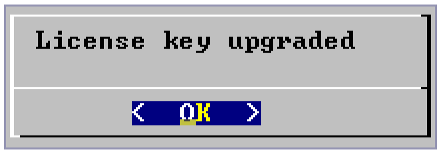
```
Select database to Upgrade License key

   ( ) Clickstream_Schema      Clickstream_Schema
   (X) Stock_Schema            Stock_Schema
   ( ) Retail_Schema           Retail_Schema
   ( ) CreditHistory_Schema    CreditHistory_Schema
   ( ) Telecom_Schema          Telecom_Schema

       ↓(+)

     <  OK  >      <Cancel>      < Help >
```

3. Enter the **absolute pathname of your downloaded license key file** and click OK.



```
Select license file to upgrade
 License file pathname: /tmp/vlicense.txt

          <  OK  >         <Cancel>         < Help >
```

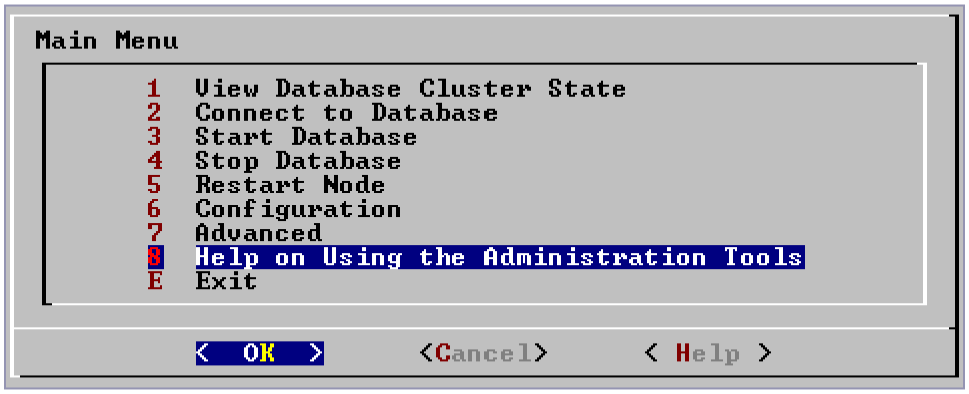
4. This message indicates that the upgrade succeeded.



```
License key upgraded

      <  OK  >
```

Help on Using the Administration Tools

This command displays a help screen about using the Administration Tools.



Most of the online help in the Administration Tools is context-sensitive. For example, if you up the use up/down arrows to select a command, press tab to move to the Help button, and press return, you get help on the selected command.

Failure 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. Vertica has a unique approach to recovering a node that is based on querying replicas of the data stored on other nodes. For example, a hardware failure may cause a node to lose database objects or to miss changes made to the database (INSERTs, UPDATEs, etc.) while offline. When the node comes back on line, it recovers lost objects and catches up with changes by querying the other nodes.

K represents the maximum number of nodes in a database that can fail and recover with no loss of data. In Vertica V2.0, the value of K can be zero (0) or one (1). The value of K can be one (1) only when the Physical Schema design meets certain requirements. The designs generated by the Database Designer are K-Safe.

For a database to be safe, no more than K nodes can be down, as shown below.

Nodes	Nodes DOWN	State of Database
4	0	Safe.
4	1	Precarious. Data loss can occur.
4	2	Inoperative. Automatic shutdown.

Automatic Recovery

Automatic recovery handles startup after a single node failure without intervention. For example, when a failed node comes back on line and rejoins the database, it recovers its lost data by querying the other nodes, as long as there are enough active nodes to ensure K-Safety. Transactions can continue to commit during the recovery process, except for a short period at the end of the recovery.

In the case of multiple node failures, automatic recovery shuts down the database. When you restart the database, automatic recovery attempts to start all the nodes and to recover objects on any node that lost objects as a result of a failure. If successful, ***View Database Cluster State*** (page 75) shows all nodes UP. Otherwise, automatic recovery writes out information that can be used for manual recovery and then shuts down again.

Manual Recovery

When automatic recovery fails to restart the database, use the ***Start Database*** (page 77) command and follow the built-in manual recovery steps. The database is not available for connections until manual recovery is complete.

Administrator Specified Recovery

Situations not handled by the built-in manual recovery steps are documented in both the Database Administrator's Guide and the Troubleshooting Guide for convenience. These are classified as:

- ***Shutdown Problems*** (page 100)
- ***Startup Problems*** (page 102)
- Node Failures

Tools for correcting these situations are described in the ***Advanced*** (page 92) section of the ***Administration Tools*** (page 75).

Shutdown Problems

This section describes some of the known problems that can occur when stopping a database. Vertica automatically shuts down a database when one of the following events occurs:

- the administrator uses the ***Stop Database*** (page 78) command.
- the cluster becomes unsafe.

Cannot shut down while users are connected

Error Message

```
NOTICE:  Cannot shut down while users are connected
```

Explanation

There are still open connections to the database.

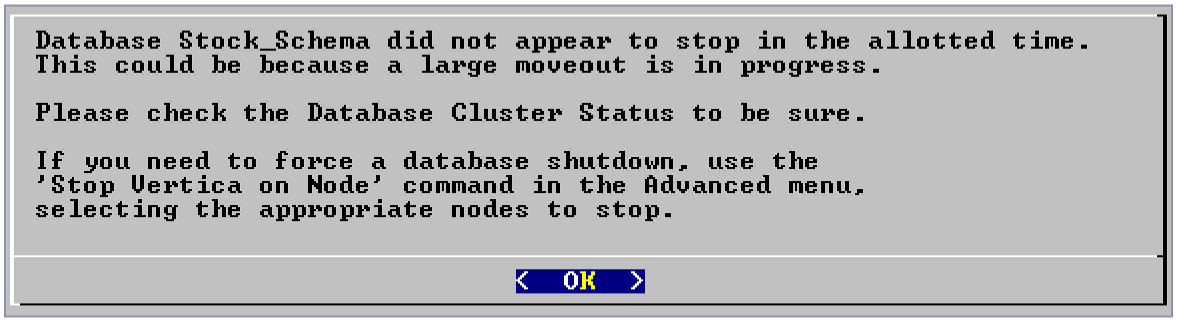
Workaround

Make sure that there are **no open connections** to the database. On each host:

```
$ netstat | grep 5433
tcp        0      0 host01:5433         host01:33558        ESTABLISHED
tcp        0      0 host01:33558        host01:5433         ESTABLISHED
```

Database ... did not appear to stop...

Error Message



```
Database Stock_Schema did not appear to stop in the allotted time.
This could be because a large moveout is in progress.

Please check the Database Cluster Status to be sure.

If you need to force a database shutdown, use the
'Stop Vertica on Node' command in the Advanced menu,
selecting the appropriate nodes to stop.
```

Explanation

If you have a large database, the Vertica process may need more time to complete a moveout. Shutting down the database cancels in-progress sessions and stops the ATM so it is unlikely that any other operations have not completed.

Workaround

Vertica Systems, Inc. recommends that you wait as long as possible before taking action. You can cause data loss by, for example, interrupting a database that is still performing a moveout.

1. ***Check the log files*** (page 53) to see if any messages are being logged. A moveout causes a pair of <INFO> messages to appear in the log files. If you a message indicating that a moveout has begun but no message indicating that it finished (commited), the moveout is still running.
2. Check the ***processes*** (page 55) and ***system resource usage*** (page 56) for database activity.
3. If you are certain that no moveouts are still in progress, go to **Advanced** > ***Stop Vertica on Node*** (page 95).
4. If that does not work, go to **Advanced** > ***Kill Vertica Process on Node*** (page 96). This command forces the cluster to go through recovery at startup.

Startup Problems

This section describes some of the known problems that can occur when starting a database. Startup fails on:

- individual nodes when automatic recovery fails for a node.
- all nodes when:
  - automatic recovery is not possible
  - manual recovery was not specified properly
  - manual recovery failed
  - the cluster becomes unsafe

If a database fails to start before it can write messages into vertica.log, check the file *catalog-path*`/`*database-name*`/dbLog`.

Startup successful, but some nodes are recovering

Error Message

```
Startup successful, but some nodes are recovering. You can use the View
Database Cluster State option to check progress.
Press RETURN to continue
```

Explanation

This message typically indicates an abnormal shutdown of one or more nodes.

Workaround

Postpone query and load processing until all nodes are up.

Error starting database, no nodes are up

Error Message

```
Error starting database, no nodes are up
Press RETURN to continue
```

Explanation

An unknown problem is preventing the database from starting.

Workaround

Reboot the hosts and try to start the database. If unsuccessful, contact ***Technical Support*** (on page 11).

Database startup successful, but it may be incomplete

Error Message

```
Database startup successful, but it may be incomplete. Some nodes remain in a
transitional state. This state may be cause by cluster catalog inconsistency.
See Database Cluster State in Main Menu for details. If this state persists,
try using the Advanced menu to Stop Vertica on all nodes, then Restart
Database with Catalog Version.
Press RETURN to continue
```

Explanation

Some nodes are in a transitional state: not up but not recovering.

Workaround

If this error persists, try using the ***Stop Vertica on Node*** (page 95) command in the ***Advanced*** (page 92) menu to stop Vertica on all nodes. Then use the ***Roll Back Database With Catalog Version*** (page 93) command.

Error Message

```
Database startup successful, but it may be incomplete. Some nodes remain in a
transitional state.
See Database Cluster State in Main Menu for details.
Press RETURN to continue
```

Explanation

This message indicates that the Database Administrator chose not to wait when prompted and that the database cannot start.

Workaround

If this error persists, contact ***Technical Support*** (on page 11).

Database did not start cleanly on initiator node!

Error Message

```
ERROR: Database did not start cleanly on initiator node!
Stopping all nodes
Issuing shutdown command to database
```

Explanation

Configuration problems can cause this error.

Workaround

1. Check hostname resolution as described in Check Hostname Resolution section of the Installation Guide.
2. Examine `/etc/hosts` on each node and specify a fully qualified domain name and an unqualified hostname. For example:

   ```
   192.168.1.99 node01.fqdname.com node01
   ```

TIMEOUT ERROR: Could not login with SSH

Error Message

```
TIMEOUT ERROR: Could not login with SSH. Here is what SSH said:
Last login: Sat Dec 15 18:05:35 2007 from node01
```

Explanation

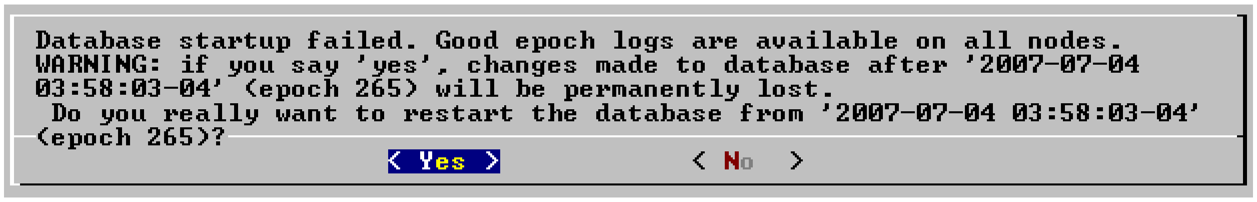
Installing Vertica on a host that is missing the mount point **/dev/pts** may result in the error when creating a database.

Workaround

Make sure that `/dev/pts` is mounted..

Good epoch logs are available on all nodes

Error Message



```
Database startup failed. Good epoch logs are available on all nodes.
WARNING: if you say 'yes', changes made to database after '2007-07-04
03:58:03-04' (epoch 265) will be permanently lost.
 Do you really want to restart the database from '2007-07-04 03:58:03-04'
(epoch 265)?
```

Explanation

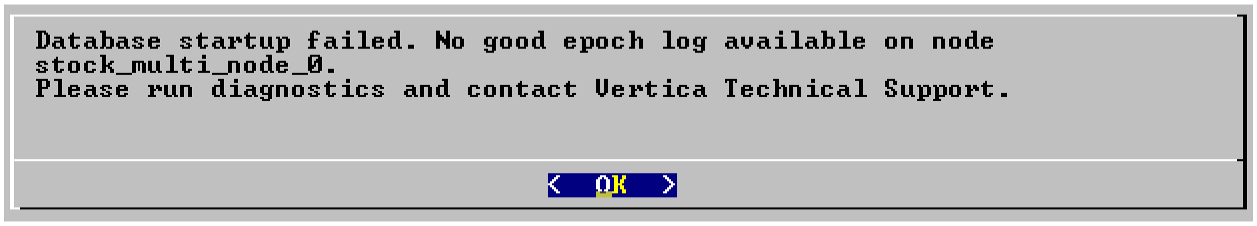
A startup attempt failed due to database inconsistency across the cluster. Vertica has determined that it can probably restart and continue running at an earlier epoch.

Workaround

Restarting from the suggested epoch erases any changes made to the database subsequent to that epoch, across the cluster. It is likely that these changes were incomplete and erasing them will allow the cluster to proceed normally using the data saved prior to the epoch.

No good epoch log available on node

Error Message



```
Database startup failed. No good epoch log available on node
stock_multi_node_0.
Please run diagnostics and contact Vertica Technical Support.
```

Explanation

There are a number of possible reasons for this error message, including an abnormal startup or shutdown. Every node in the cluster must be started with the same recovery epoch. Non-matching recovery epochs occur when a cluster has experienced an unsafe shutdown.

Workaround

1. Make sure that all nodes are powered on.
2. ***Start the database*** (page 77) again.
3. Make sure that all nodes have Spread running (see Check Spread). If necessary, restart Spread where it is not running and ***start the database*** (page 77).
4. On each node that did not start up, examine ***dbLog*** (page 52) for the cause of the failure.
5. If the cause cannot be determined, it's likely that a node has no catalog version or epoch log from which to recover. Run diagnostic tests (see Using Diagnostic Tools) and contact ***Technical Support*** (on page 11).

Nodes stuck in INITIALIZING state

Error Message

In rare cases, some or all nodes can get stuck in the INITIALIZING state when trying to start the database.

Explanation

This is known to happen when a database crashes while CREATE TABLE or other DDL commands are executing.

Workaround

1. **Advanced** > ***Stop Vertica on Node*** (page 95) to stop all nodes.
2. **Main Menu** > ***Start Database*** (page 77).

Spread is not running

Error Message

```
spreadCheck fails
spread may not be running on all nodes
        Error while starting/enabling multicasting to all hosts
        The following hosts are not running spread:
```

Workaround

1. Use Check Spread to verify that Spread is not running.
2. Examine `/tmp/adminTools-`*username*`.log` for problems.
3. Examine `/tmp/spread*.log` and `/var/log/spreadd.log` for problems.
4. Make sure the /etc/hosts file is correct (see Define the Loopback Address in the Installation Guide.
5. If necessary, restart Spread.

Restart the Spread Daemon

1. Log in as root.

   $ su - root

   password: <root-password>

   #

   You can use sudo (if enabled) if you do not have the root password.
2. Restart the Spread daemon:
   ```
   # /etc/init.d/spreadd restart
   ```
3. Make sure the daemon is running:
   ```
   # ps ax | grep spread
   ```

Index