

HP Vertica Analytics Platform 6.1.x

Getting Started Guide

Doc Revision 3

Copyright© 2006-2013 Hewlett-Packard

Date of Publication: Monday, September 23, 2013



Contents

| | |
|---------------------------|----------|
| Syntax Conventions | 4 |
|---------------------------|----------|

| | |
|---------------------------------|----------|
| Getting Started Overview | 5 |
|---------------------------------|----------|

| | |
|---|---|
| HP Vertica Virtual Analytics Platform | 6 |
|---|---|

| | |
|--|----------|
| Installing the Example Database | 8 |
|--|----------|

| | |
|-------------------------------|-----------|
| VMart Example Database | 10 |
|-------------------------------|-----------|

| | |
|-----------------------------|----|
| Public Schema | 12 |
| inventory_fact | 14 |
| customer_dimension | 14 |
| date_dimension | 15 |
| employee_dimension | 16 |
| product_dimension | 16 |
| promotion_dimension | 17 |
| shipping_dimension | 18 |
| vendor_dimension | 18 |
| warehouse_dimension | 19 |
| Store Schema | 20 |
| store_orders_fact | 20 |
| store_sales_fact | 21 |
| store_dimension | 22 |
| Online_Sales Schema | 23 |
| online_sales_fact | 23 |
| call_center_dimension | 24 |
| online_page_dimension | 25 |
| VMart Sample Queries | 25 |
| vmart_query_01.sql | 25 |
| vmart_query_02.sql | 26 |
| vmart_query_03.sql | 27 |
| vmart_query_04.sql | 27 |
| vmart_query_05.sql | 28 |
| vmart_query_06.sql | 29 |
| vmart_query_07.sql | 29 |
| vmart_query_08.sql | 30 |
| vmart_query_09.sql | 31 |

| | |
|---|-----------|
| Tutorial: Setting up an Example Database | 32 |
|---|-----------|

| | |
|--|----|
| Step 1: Set Up the Example Environment | 33 |
| Step 2: Create the Example Database | 34 |
| Create the Example Database Using the Administration Tools | 35 |
| Create the Example Database Using Management Console | 37 |

| | |
|--|---------------|
| Step 3: Define the Database Schema..... | 37 |
| Step 4: Load the Data..... | 38 |
| Step 5: Create a Comprehensive Design | 39 |
| Step 6: Connect to the Database and Run a Simple Query | 42 |
| Step 7: Test the Optimized Design | 43 |
| Step 8: (Optional) Generate Custom Data Files | 45 |
| Running Simple Queries | 47 |
| Cleanup Procedure | 48 |
| Administration Tools Keystrokes | 49 |
| Notes for Remote Terminal Users | 50 |
| Copyright Notice | 51 |

Syntax Conventions

The following are the syntax conventions used in the HP Vertica documentation.

| Syntax Convention | Description |
|---------------------------------------|--|
| Text without brackets/braces | Indicates content you type, as shown. |
| < <i>Text inside angle brackets</i> > | Represents a placeholder for which you must supply a value. The variable is usually shown in italics. See <i>Placeholders</i> below. |
| [<i>Text inside brackets</i>] | Indicates optional items; for example, CREATE TABLE [<i>schema_name</i>]. <i>table_name</i> The brackets indicate that the <i>schema_name</i> is optional. Do not type the square brackets. |
| { <i>Text inside braces</i> } | Indicates a set of options from which you choose one; for example: QUOTES { ON OFF } indicates that exactly one of ON or OFF must be provided. You do not type the braces: QUOTES ON |
| Backslash \ | Represents a continuation character used to indicate text that is too long to fit on a single line. |
| Ellipses ... | Indicate a repetition of the previous parameter. For example, option[, ...] means that you can enter multiple, comma-separated options. Showing ellipses in code examples might also mean that part of the text has been omitted for readability, such as in multi-row result sets. |
| Indentation | Is an attempt to maximize readability; SQL is a free-form language. |
| <i>Placeholders</i> | Represent items that must be replaced with appropriate identifiers or expressions and are usually shown in italics. |
| Vertical bar | Is a separator for mutually exclusive items. For example: [ASC DESC] Choose one or neither. You do not type the square brackets. |

Getting Started Overview

Follow the steps in this guide to set up an HP Vertica database and run example queries. The examples provided in this guide require that you have completed the following steps:

- Installed HP Vertica on a cluster, as described in the Installation Guide, or obtained a **Virtual Machine** with HP Vertica installed on it.
- Are logged in to the server as the Database Administrator user; for example, `dbadmin`.

User interfaces

You'll access your database either by an SSH client or through the terminal utility in your Linux Console, such as `vsql`. By following this tutorial, you use the following user interfaces:

- The Linux command line (shell) interface
- The HP Vertica Administration Tools (see the Administrator's Guide for details)
- The `vsql` client interface (see the Programmer's Guide for details)

Sample databases: VMart

HP Vertica provides a sample multi-schema database, called **VMart** (page [10](#)), which you can use to learn and test. The steps in the tutorial refer to the VMart schema. If you installed HP Vertica using the RPM, you'll find the VMart schema in the `/opt/vertica/examples/VMart_Schema` directory.

Tutorial: Setting up an example database

The Tutorial describes how to configure an HP Vertica database that you'll use to run sample queries. It assumes that you have already installed HP Vertica on a cluster of hosts, as described in the Installation Guide. You can copy the example database to non-cluster hosts for reference purposes, but you must perform the tasks in the Tutorial on the Administration Host.

Note: HP also provides a simple database installation script that is quicker and easier—but also less comprehensive—than this tutorial. See **Installing the Example Database** (page [8](#)) for details.

Example queries

The example database includes several queries that are intended to represent queries that might be used in a real business. You'll also use these queries as inputs to the Database Designer. Once you're comfortable running the example queries, you might want to write your own. See **VMart Sample Queries** (page [25](#)).

Cleanup procedure

When you have finished with the tutorial, you can restore your host machines to their original state. Instructions are provided in **Cleanup Procedure** (page [48](#)).

HP Vertica Virtual Analytics Platform

HP Vertica is available as a Virtual Machine (VM) that is pre-installed on a 64-bit CentOS image and comes with a license for 500 GB of data storage.

Important! HP Vertica virtual machines are not supported in a production environment. The VMs are provided for evaluation purposes.

The VM image is preconfigured with the following hardware settings.

- 1 CPU
- 1024 MB RAM
- 50 GB Hard Disk (SCSI, not preallocated, single file storage)
- Bridged Networking

To download the virtual machine images:

The HP Vertica virtual machine is available both as an OVF template (for VMWare vSphere 4.0) and as a VMDK file (for VMWare Server 2.0 and VMWare Workstation 7.0). Download the appropriate file for your VMWare deployment from the **myVertica portal** <http://my.vertica.com/>.

Check for HP Vertica Updates

The virtual-machine image might not include the latest available HP Vertica release. After you install your virtual machine, verify the version of HP Vertica by typing:

```
# rpm -qa | grep vertica
<package-name>
```

The name of the rpm package that the command returns contains the version and build numbers. If there is a later version of HP Vertica, you can download it from the **myVertica portal** <http://my.vertica.com/Downloads> tab. Upgrade instructions are provided in the Installation Guide.

To start the virtual machine:

- 1 Open the appropriate HP Vertica VM image file in VMWare. For example, open the VMX file if you are using VMWare Workstation or the OVF template if you are using VMWare vSphere.
- 2 Navigate to the settings for the VM image and adjust the network settings so that they are compatible with your virtual network.
- 3 Start the virtual machine. For example, in VMWare Workstation, select **VM > Power > Power On**.

To log into the virtual machine:

The first time you boot the virtual machine you are automatically logged in and a web page displays further instructions. To log back into the virtual machine, use the following username and password:

- **Username:** dbadmin

- **Password:** `password`
- **Root Password:** `password`

Important: The `dbadmin` user has `sudo` privileges. Be sure to change the `dbadmin` and root passwords with the `passwd` command.

Installing the Example Database

HP Vertica gives you two options to install the example database:

- An advanced-but-simple example database installation using the Administration Tools interface. See **Tutorial: Setting up an Example Database** (page [32](#)) in this guide for details.
- A quick installation script that lets you create the example database and start using it immediately, as described in this topic.

Installing the example database from a script

The scripts are located in `/opt/vertica/sbin` and are called:

- `install_example` — Creates a database on the default port (5433), generates data, creates the schema and a default superprojection, and loads the data.
- `delete_example` — Drops the database

1 In a terminal window, log in as the DBA user:

```
# su dbadmin
```

2 Change to the `/example` directory and run the install script:

```
$ /opt/vertica/sbin/install_example VMart
```

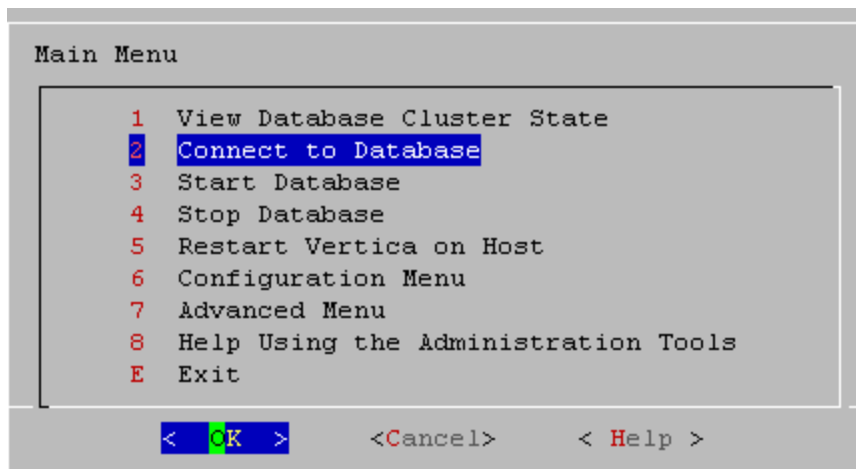
Note: If you have not already done so, you must accept the EULA (one time only) using the Administration Tools. You'll also do that in **Step 2** (page [34](#)) of the Tutorial.

3 Connect to the database:

```
$ /opt/vertica/bin/vsql
```

Alternatively connect to the database using the Administration Tools and select **Connect to Database** from the Main Menu:

```
$ admintools
```



4 Run a simple query. For example, to count all the records in the `store_sales_fact` table:

```
=> SELECT COUNT(1) FROM store.store_sales_fact;
```


The example database log files, `ExampleDelete.txt` and `ExampleInstall.txt`, are written to `/opt/vertica/examples/log`.

Example Database Script

While you can create your own queries, the VMart example directory contains a sample query script file that you can use help to get you started quickly. You can find the sample script at this path: `/opt/vertica/examples/VMart_Schema`

The following table describes the scripts available, where *{identifier}* is the name of the example database:

| Script Name | Description |
|---|--|
| <code>{identifier}_count_data.sql</code> | Counts rows of all example database tables |
| <code>{identifier}_define_schema.sql</code> | Defines the schema for each table |
| <code>{identifier}_gen</code> | Is the sample data generator |
| <code>{identifier}_load_data.sql</code> | Loads data to the corresponding tables using COPY DIRECT |
| <code>{identifier}_queries.sql</code> | Contains all sample queries |
| <code>{identifier}_schema_drop.sql</code> | Drops all example database tables |
| <code>{identifier}_query_###.sql</code> | Are the individual queries; for example query #1 through “n” |

Deleting the example database

To remove an example database:

- 1 Log in as the DBA user; for example:

```
# su dbadmin
```
- 2 Run the `delete_example` script:

```
$ /opt/vertica/sbin/delete_example <example_name>
```

where *<example_name>* is the name of the example database you provided to the install script.

See Also

Tutorial: Setting up an Example Database (page [32](#))

VMart Example Database

HP Vertica ships with an example database, called the VMart Example Database, which is based on a fictional department store chain that has an online storefront in addition to traditional brick and mortar stores. VMart is used throughout this tutorial and its database contains the following schemas:

- **Public** (page [12](#)) schema
- **Store** (page [20](#)) schema
- **Online_Sales** (page [23](#)) schema

Each schema is described in a separate section.

Note: The data that your queries return might differ from the example output shown in this guide because the sample data generator is random.

The number of example databases you create within a single HP Vertica installation is limited only by the disk space available on your system; however, HP strongly recommends that you start only one example database at a time to avoid unpredictable results.

VMart database location and scripts

The VMart example database is installed in `/opt/vertica/examples/VMart_Schema`. This folder contains query script files that you can use to get started quickly. You can use the scripts as templates for your own applications.

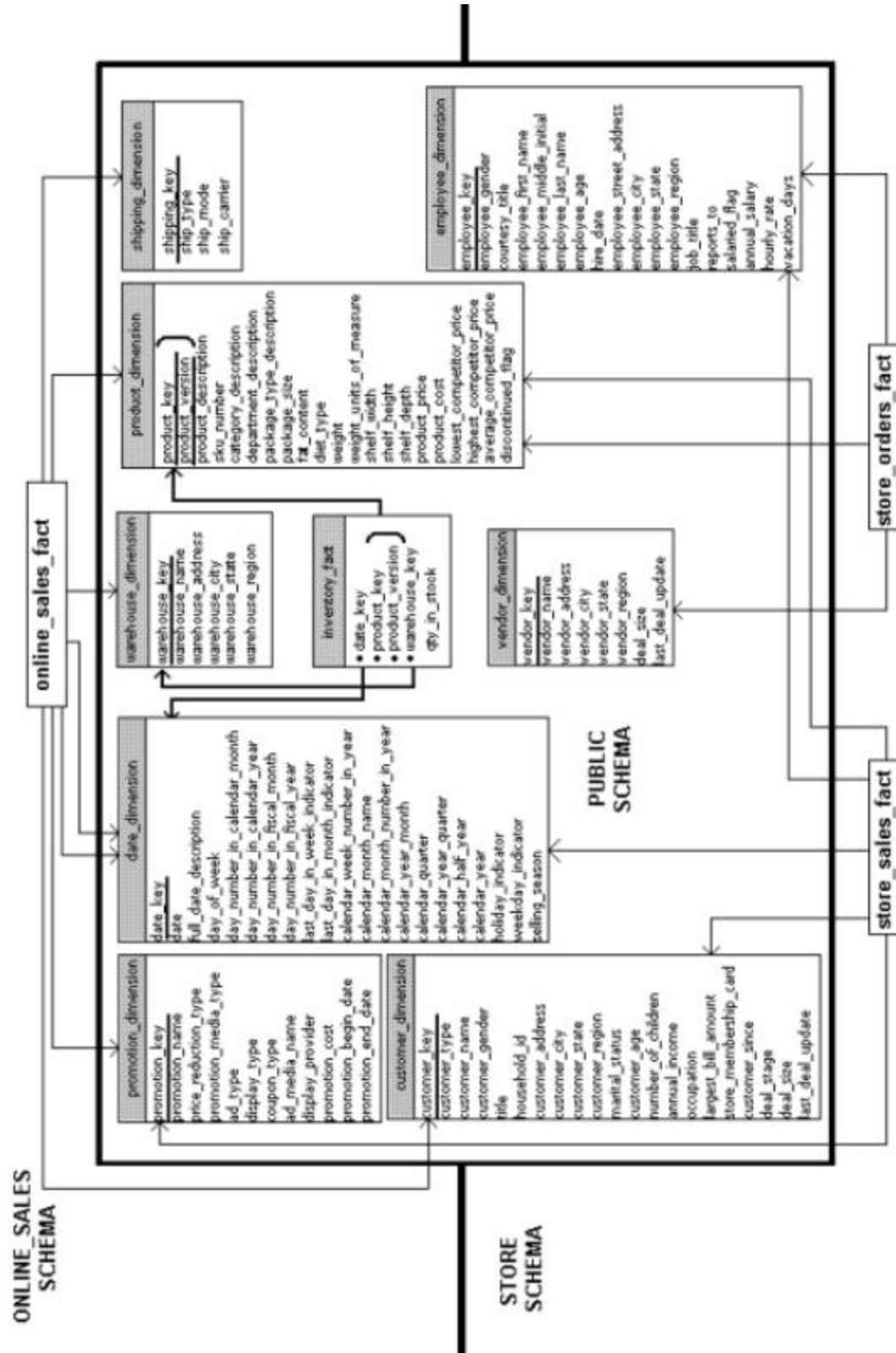
The following table describes the scripts available, where *{identifier}* is the name of the example database:

| Script/file name | Description |
|--------------------------------------|--|
| <code>vmart_count_data.sql</code> | SQL script that does a COUNT(*) of each table, which you can use to verify load. |
| <code>vmart_define_schema.sql</code> | SQL script that defines the logical schema for each table and referential integrity constraints. |
| <code>vmart_gen.cpp</code> | Data generator source code (C++). |
| <code>vmart_gen</code> | Data generator executable file. |
| <code>vmart_load_data.sql</code> | SQL script that loads the generated sample data to the corresponding tables using COPY DIRECT. |
| <code>vmart_queries.sql</code> | SQL script contains concatenated queries for use as a training set for the Database Designer. |
| <code>vmart_query_#.sql</code> | SQL scripts that contain individual queries; for example query #1 through “n” |
| <code>vmart_schema_drop.sql</code> | SQL script that drops all example database tables. |

| | |
|----------|---|
| README | Text file containing instructions for using the data generator. |
| Time.txt | Text file containing pre-computed data for date dimension tables. |

Public Schema

The Public schema is a snowflake schema. The following graphic illustrates the Public schema and its relationships with tables in the Online_Sales and Store schemas.



inventory_fact

This table contains information about each product in inventory.

| Column Name | Data Type | NULLs |
|-----------------|-----------|-------|
| Date_key | INTEGER | No |
| Product_key | INTEGER | No |
| Product_version | INTEGER | No |
| Warehouse_key | INTEGER | No |
| Qty_in_stock | INTEGER | No |

customer_dimension

This table contains information about all the retail chain's customers.

| Column Name | Data Type | NULLs |
|---------------------|--------------|-------|
| Customer_key | INTEGER | No |
| Customer_type | VARCHAR(16) | Yes |
| Customer_name | VARCHAR(256) | Yes |
| Customer_gender | VARCHAR(8) | Yes |
| Title | VARCHAR(8) | Yes |
| Household_id | INTEGER | Yes |
| Customer_address | VARCHAR(256) | Yes |
| Customer_city | VARCHAR(64) | Yes |
| Customer_state | CHAR(2) | Yes |
| Customer_region | VARCHAR(64) | Yes |
| Marital_status | VARCHAR(32) | Yes |
| Customer_age | INTEGER | Yes |
| Number_of_children | INTEGER | Yes |
| Annual_income | INTEGER | Yes |
| Occupation | VARCHAR(64) | Yes |
| Largest_bill_amount | INTEGER | Yes |

| | | |
|-----------------------|-------------|-----|
| Store_membership_card | INTEGER | Yes |
| Customer_since | DATE | Yes |
| Deal_stage | VARCHAR(32) | Yes |
| Deal_size | INTEGER | Yes |
| Last_deal_update | DATE | Yes |

date_dimension

This table contains information about dates. It is generated from a file containing correct date/time data.

| Column Name | Data Type | NULLs |
|-------------------------------|-------------|-------|
| Date_key | INTEGER | No |
| Date | DATE | Yes |
| Full_date_description | VARCHAR(18) | Yes |
| Day_of_week | VARCHAR(9) | Yes |
| Day_number_in_calendar_month | INTEGER | Yes |
| Day_number_in_calendar_year | INTEGER | Yes |
| Day_number_in_fiscal_month | INTEGER | Yes |
| Day_number_in_fiscal_year | INTEGER | Yes |
| Last_day_in_week_indicator | INTEGER | Yes |
| Last_day_in_month_indicator | INTEGER | Yes |
| Calendar_week_number_in_year | INTEGER | Yes |
| Calendar_month_name | VARCHAR(9) | Yes |
| Calendar_month_number_in_year | INTEGER | Yes |
| Calendar_year_month | CHAR(7) | Yes |
| Calendar_quarter | INTEGER | Yes |
| Calendar_year_quarter | CHAR(7) | Yes |
| Calendar_half_year | INTEGER | Yes |
| Calendar_year | INTEGER | Yes |
| Holiday_indicator | VARCHAR(10) | Yes |
| Weekday_indicator | CHAR(7) | Yes |
| Selling_season | VARCHAR(32) | Yes |

employee_dimension

This table contains information about all the people who work for the retail chain.

| Column Name | Data Type | NULLs |
|-------------------------|--------------|-------|
| Employee_key | INTEGER | No |
| Employee_gender | VARCHAR(8) | Yes |
| Employee_title | VARCHAR(8) | Yes |
| Employee_first_name | VARCHAR(64) | Yes |
| Employee_middle_initial | VARCHAR(8) | Yes |
| Employee_last_name | VARCHAR(64) | Yes |
| Employee_age | INTEGER | Yes |
| Hire_date | DATE | Yes |
| Employee_street_address | VARCHAR(256) | Yes |
| Employee_city | VARCHAR(64) | Yes |
| Employee_state | CHAR(2) | Yes |
| Employee_region | CHAR(32) | Yes |
| Job_title | VARCHAR(64) | Yes |
| Reports_to | INTEGER | Yes |
| Salaried_flag | INTEGER | Yes |
| Annual_salary | INTEGER | Yes |
| Hourly_rate | FLOAT | Yes |
| Vacation_days | INTEGER | Yes |

product_dimension

The product_dimension table describes all products sold by the department store chain.

| Column Name | Data Type | NULLs |
|-----------------|-----------|-------|
| Product_key | INTEGER | No |
| Product_version | INTEGER | No |

| | | |
|--------------------------|--------------|-----|
| Product_description | VARCHAR(128) | Yes |
| Sku_number | CHAR(32) | Yes |
| Category_description | CHAR(32) | Yes |
| Department_description | CHAR(32) | Yes |
| Package_type_description | CHAR(32) | Yes |
| Package_size | CHAR(32) | Yes |
| Fat_content | INTEGER | Yes |
| Diet_type | CHAR(32) | Yes |
| Weight | INTEGER | Yes |
| Weight_units_of_measure | CHAR(32) | Yes |
| Shelf_width | INTEGER | Yes |
| Shelf_height | INTEGER | Yes |
| Shelf_depth | INTEGER | Yes |
| Product_price | INTEGER | Yes |
| Product_cost | INTEGER | Yes |
| Lowest_competitor_price | INTEGER | Yes |
| Highest_competitor_price | INTEGER | Yes |
| Average_competitor_price | INTEGER | Yes |
| Discontinued_flag | INTEGER | Yes |

promotion_dimension

The promotion_dimension describes every promotion ever done by the retail chain.

| Column Name | Data Type | NULLs |
|----------------------|--------------|-------|
| Promotion_key | INTEGER | No |
| Promotion_name | VARCHAR(128) | Yes |
| Price_reduction_type | VARCHAR(32) | Yes |
| Promotion_media_type | VARCHAR(32) | Yes |
| Ad_type | VARCHAR(32) | Yes |
| Display_type | VARCHAR(32) | Yes |
| Coupon_type | VARCHAR(32) | Yes |
| Ad_media_name | VARCHAR(32) | Yes |

| | | |
|----------------------|--------------|-----|
| Display_provider | VARCHAR(128) | Yes |
| Promotion_cost | INTEGER | Yes |
| Promotion_begin_date | DATE | Yes |
| Promotion_end_date | DATE | Yes |

shipping_dimension

This table contains information about the shipping companies that the retail chain uses.

| Column Name | Data Type | NULLs |
|--------------|-----------|-------|
| Shipping_key | INTEGER | No |
| Ship_type | CHAR(30) | Yes |
| Ship_mode | CHAR(10) | Yes |
| Ship_carrier | CHAR(20) | Yes |

vendor_dimension

This table contains information about each vendor that provides products sold through the retail chain.

| Column Name | Data Type | NULLs |
|------------------|-------------|-------|
| Vendor_key | INTEGER | No |
| Vendor_name | VARCHAR(64) | Yes |
| Vendor_address | VARCHAR(64) | Yes |
| Vendor_city | VARCHAR(64) | Yes |
| Vendor_state | CHAR(2) | Yes |
| Vendor_region | VARCHAR(32) | Yes |
| Deal_size | INTEGER | Yes |
| Last_deal_update | DATE | Yes |

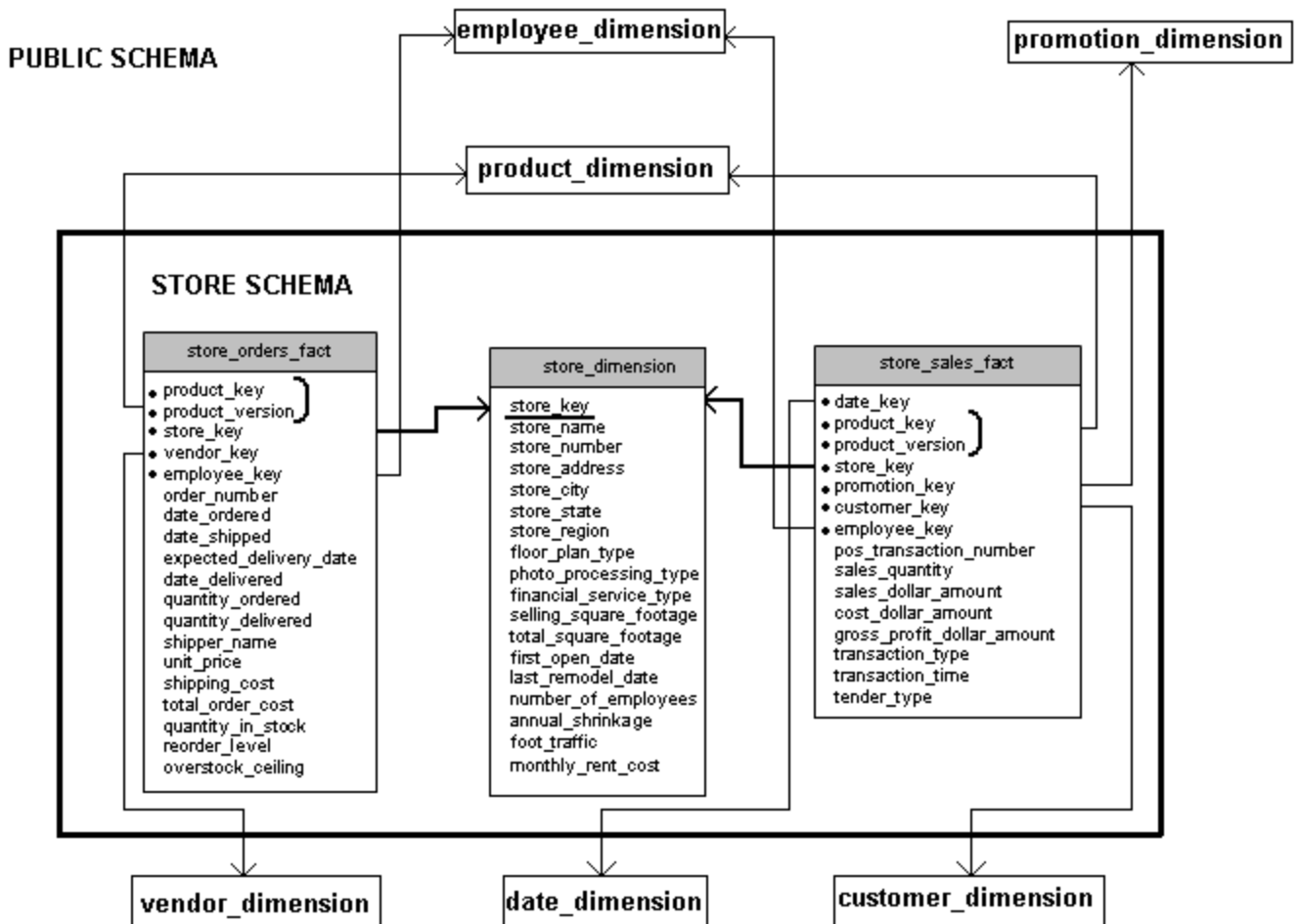
warehouse_dimension

This table provides information about each of the chain's warehouses.

| Column Name | Data Type | NULLs |
|-------------------|--------------|-------|
| Warehouse_key | INTEGER | No |
| Warehouse_name | VARCHAR(20) | Yes |
| Warehouse_address | VARCHAR(256) | Yes |
| Warehouse_city | VARCHAR(60) | Yes |
| Warehouse_state | CHAR(2) | Yes |
| Warehouse_region | VARCHAR(32) | Yes |

Store Schema

The Store schema is a snowflake schema that contains information about the retail chain's brick and mortar stores. The following graphic illustrates the Store schema and its relationship with tables in the Public schema.



store_orders_fact

This table contains information about all orders made at the company's brick and mortar stores.

| Column Name | Data Type | NULLs |
|------------------------|-------------|-------|
| Product_key | INTEGER | No |
| Product_version | INTEGER | No |
| Store_key | INTEGER | No |
| Vendor_key | INTEGER | No |
| Employee_key | INTEGER | No |
| Order_number | INTEGER | No |
| Date_ordered | DATE | Yes |
| Date_shipped | DATE | Yes |
| Expected_delivery_date | DATE | Yes |
| Date_delivered | DATE | Yes |
| Quantity_ordered | INTEGER | Yes |
| Quantity_delivered | INTEGER | Yes |
| Shipper_name | VARCHAR(32) | Yes |
| Unit_price | INTEGER | Yes |
| Shipping_cost | INTEGER | Yes |
| Total_order_cost | INTEGER | Yes |
| Quantity_in_stock | INTEGER | Yes |
| Reorder_level | INTEGER | Yes |
| Overstock_ceiling | INTEGER | Yes |

store_sales_fact

This table contains information about all sales made at the company's brick and mortar stores.

| Column Name | Data Type | NULLs |
|-----------------|-----------|-------|
| Date_key | INTEGER | No |
| Product_key | INTEGER | No |
| Product_version | INTEGER | No |
| Store_key | INTEGER | No |
| Promotion_key | INTEGER | No |

| | | |
|----------------------------|-------------|-----|
| Customer_key | INTEGER | No |
| Employee_key | INTEGER | No |
| Pos_transaction_number | INTEGER | No |
| Sales_quantity | INTEGER | Yes |
| Sales_dollar_amount | INTEGER | Yes |
| Cost_dollar_amount | INTEGER | Yes |
| Gross_profit_dollar_amount | INTEGER | Yes |
| Transaction_type | VARCHAR(16) | Yes |
| Transaction_time | TIME | Yes |
| Tender_type | VARCHAR(8) | Yes |

store_dimension

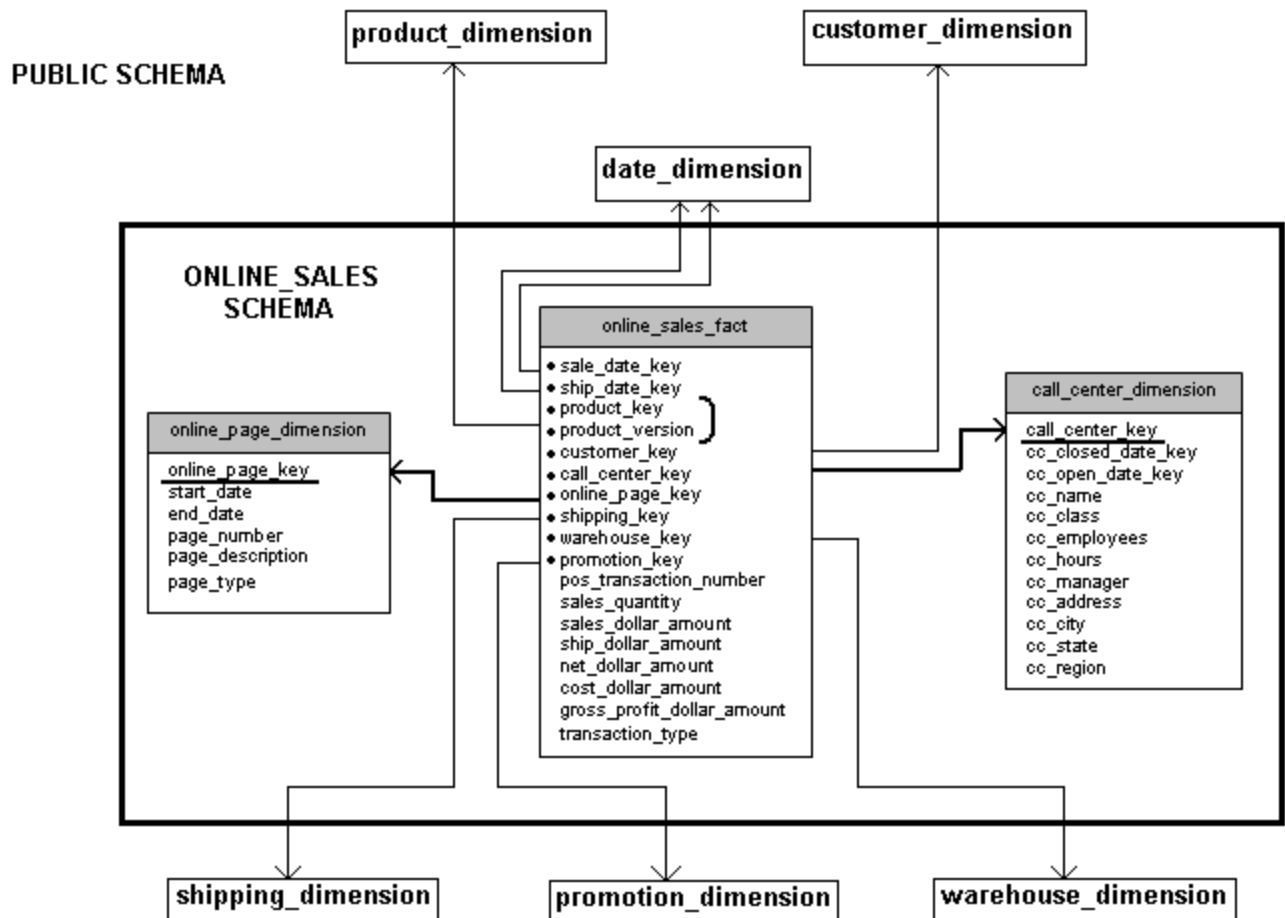
This table contains information about each brick and mortar store within the retail chain.

| Column Name | Data Type | NULLs |
|------------------------|--------------|-------|
| Store_key | INTEGER | No |
| Store_name | VARCHAR(64) | Yes |
| Store_number | INTEGER | Yes |
| Store_address | VARCHAR(256) | Yes |
| Store_city | VARCHAR(64) | Yes |
| Store_state | CHAR(2) | Yes |
| Store_region | VARCHAR(64) | Yes |
| Floor_plan_type | VARCHAR(32) | Yes |
| Photo_processing_type | VARCHAR(32) | Yes |
| Financial_service_type | VARCHAR(32) | Yes |
| Selling_square_footage | INTEGER | Yes |
| Total_square_footage | INTEGER | Yes |
| First_open_date | DATE | Yes |
| Last_remodel_date | DATE | Yes |
| Number_of_employees | INTEGER | Yes |
| Annual_shrinkage | INTEGER | Yes |

| | | |
|-------------------|---------|-----|
| Foot_traffic | INTEGER | Yes |
| Monthly_rent_cost | INTEGER | Yes |

Online_Sales Schema

The Online_Sales schema is a snowflake schema. The following graphic illustrates the Online_Sales schema and its relationship with tables in the Public schema.



online_sales_fact

The online_sales_fact table describes all the items purchased through the online store front.

| Column Name | Data Type | NULLs |
|----------------------------|-------------|-------|
| Sale_date_key | INTEGER | No |
| Ship_date_key | INTEGER | No |
| Product_key | INTEGER | No |
| Product_version | INTEGER | No |
| Customer_key | INTEGER | No |
| Call_center_key | INTEGER | No |
| Online_page_key | INTEGER | No |
| Shipping_key | INTEGER | No |
| Warehouse_key | INTEGER | No |
| Promotion_key | INTEGER | No |
| Pos_transaction_number | INTEGER | No |
| Sales_quantity | INTEGER | Yes |
| Sales_dollar_amount | FLOAT | Yes |
| Ship_dollar_amount | FLOAT | Yes |
| Net_dollar_amount | FLOAT | Yes |
| Cost_dollar_amount | FLOAT | Yes |
| Gross_profit_dollar_amount | FLOAT | Yes |
| Transaction_type | VARCHAR(16) | Yes |

call_center_dimension

The call_center_dimension table describes all the chain's call centers.

| Column Name | Data Type | NULLs |
|-----------------|-------------|-------|
| Call_center_key | INTEGER | No |
| Cc_closed_date | DATE | Yes |
| Cc_open_date | DATE | Yes |
| Cc_name | VARCHAR(50) | Yes |
| Cc_class | VARCHAR(50) | Yes |
| Cc_employees | INTEGER | Yes |
| Cc_hours | CHAR(20) | Yes |

| | | |
|------------|--------------|-----|
| Cc_manager | VARCHAR(40) | Yes |
| Cc_address | VARCHAR(256) | Yes |
| Cc_city | VARCHAR(64) | Yes |
| Cc_state | CHAR(2) | Yes |
| Cc_region | VARCHAR(64) | Yes |

online_page_dimension

The online_page_dimension table describes all the pages in the online store front.

| Column Name | Data Type | NULLs |
|------------------|--------------|-------|
| Online_page_key | INTEGER | No |
| Start_date | DATE | Yes |
| End_date | DATE | Yes |
| Page_number | INTEGER | Yes |
| Page_description | VARCHAR(100) | Yes |
| Page_type | VARCHAR(100) | Yes |

VMart Sample Queries

The VMart example database includes several queries that are intended to represent queries that might be used in a real business. To run a sample query:

```
$ \i <script_name>
```

Once you're comfortable running the sample queries, you might want to write your own.

vmart_query_01.sql

Query

```
-- vmart_query_01.sql
-- FROM clause subquery
-- Return the values for five products with the
-- lowest-fat content in the Dairy department

SELECT fat_content
```

```
FROM (
  SELECT DISTINCT fat_content
  FROM product_dimension
  WHERE department_description
  IN ('Dairy') ) AS food
ORDER BY fat_content
LIMIT 5;
```

Example output

```
fat_content
-----
          80
          81
          82
          83
          84
(5 rows)
```

vmart_query_02.sql

Query

```
-- vmart_query_02.sql
-- WHERE clause subquery
-- Asks for all orders placed by stores located in Massachusetts
-- and by vendors located elsewhere before March 1, 2003:

SELECT order_number, date_ordered
FROM store.store_orders_fact orders
WHERE orders.store_key IN (
  SELECT store_key
  FROM store.store_dimension
  WHERE store_state = 'MA')
  AND orders.vendor_key NOT IN (
  SELECT vendor_key
  FROM public.vendor_dimension
  WHERE vendor_state = 'MA')
  AND date_ordered < '2003-03-01';
```

Example output

```
order_number | date_ordered
-----+-----
          1584 | 2003-01-05
          39396 | 2003-02-05
          83738 | 2003-01-04
           8898 | 2003-02-05
          69712 | 2003-01-06
          74866 | 2003-01-03
          75397 | 2003-02-06
          60069 | 2003-01-10
```

```

      85854 | 2003-01-03
      21982 | 2003-02-03
      47766 | 2003-02-07
      31284 | 2003-02-03
      28005 | 2003-01-09
      79963 | 2003-02-01
      19515 | 2003-02-05
(15 rows)

```

vmart_query_03.sql

Query

```

-- vmart_query_03.sql
-- Noncorrelated subquery
-- Requests female and male customers with the maximum
-- annual income from customers

```

```

SELECT customer_name, annual_income
FROM public.customer_dimension
WHERE (customer_gender, annual_income) IN (
    SELECT customer_gender, MAX(annual_income)
    FROM public.customer_dimension
    GROUP BY customer_gender);

```

Example output

```

customer_name | annual_income
-----+-----
James M. McNulty |          999979
Emily G. Vogel   |          999998
(2 rows)

```

vmart_query_04.sql

Query

```

-- vmart_query_04.sql
-- IN predicate
-- Find all products supplied by stores in Massachusetts

SELECT DISTINCT s.product_key, p.product_description
FROM store.store_sales_fact s, public.product_dimension p
WHERE s.product_key = p.product_key
AND s.product_version = p.product_version AND s.store_key IN (
    SELECT store_key
    FROM store.store_dimension
    WHERE store_state = 'MA')
ORDER BY s.product_key;

```

Example output

| product_key | product_description |
|-------------|----------------------------------|
| 1 | Brand #1 butter |
| 1 | Brand #2 bagels |
| 2 | Brand #3 lamb |
| 2 | Brand #4 brandy |
| 2 | Brand #5 golf clubs |
| 2 | Brand #6 chicken noodle soup |
| 3 | Brand #10 ground beef |
| 3 | Brand #11 vanilla ice cream |
| 3 | Brand #7 canned chicken broth |
| 3 | Brand #8 halibut |
| 3 | Brand #9 camera case |
| 4 | Brand #12 rash ointment |
| 4 | Brand #13 low fat milk |
| 4 | Brand #14 chocolate chip cookies |
| 4 | Brand #15 silver polishing cream |

...

vmart_query_05.sql**Query**

```
-- vmart_query_05.sql
-- EXISTS predicate
-- Get a list of all the orders placed by all stores on
-- January 2, 2003 for the vendors with records in the
-- vendor_dimension table

SELECT store_key, order_number, date_ordered
FROM store.store_orders_fact
WHERE EXISTS (
    SELECT 1
    FROM public.vendor_dimension
    WHERE public.vendor_dimension.vendor_key =
store.store_orders_fact.vendor_key)
    AND date_ordered = '2003-01-02';
```

Example output

| store_key | order_number | date_ordered |
|-----------|--------------|--------------|
| 213 | 148816 | 2003-01-02 |
| 111 | 184148 | 2003-01-02 |
| 89 | 279732 | 2003-01-02 |
| 115 | 3677 | 2003-01-02 |
| 212 | 117057 | 2003-01-02 |
| 65 | 198323 | 2003-01-02 |
| 238 | 246942 | 2003-01-02 |
| 140 | 257554 | 2003-01-02 |

| | | | | |
|-----|--|--------|--|------------|
| 43 | | 79699 | | 2003-01-02 |
| 219 | | 240925 | | 2003-01-02 |
| 249 | | 4789 | | 2003-01-02 |
| 12 | | 234175 | | 2003-01-02 |
| 119 | | 176211 | | 2003-01-02 |
| 107 | | 249378 | | 2003-01-02 |
| 228 | | 251959 | | 2003-01-02 |

(15 rows)

vmart_query_06.sql

Query

```
-- vmart_query_06.sql
-- EXISTS predicate
-- Orders placed by the vendor who got the best deal
-- on January 4, 2004

SELECT store_key, order_number, date_ordered
FROM store.store_orders_fact ord, public.vendor_dimension vd
WHERE ord.vendor_key = vd.vendor_key
AND vd.deal_size IN (
    SELECT MAX(deal_size)
    FROM public.vendor_dimension)
AND date_ordered = '2004-01-04';
```

Example output

| store_key | | order_number | | date_ordered |
|-----------|--|--------------|--|--------------|
| 45 | | 202416 | | 2004-01-04 |
| 113 | | 66017 | | 2004-01-04 |
| 121 | | 251417 | | 2004-01-04 |
| 24 | | 250295 | | 2004-01-04 |
| 9 | | 188567 | | 2004-01-04 |
| 166 | | 36008 | | 2004-01-04 |
| 27 | | 150241 | | 2004-01-04 |
| 148 | | 182207 | | 2004-01-04 |
| 198 | | 75716 | | 2004-01-04 |

(9 rows)

vmart_query_07.sql

Query

```
-- vmart_query_07.sql
-- Multicolumn subquery
-- Which products have the highest cost,
-- grouped by category and department

SELECT product_description, sku_number, department_description
```

```
FROM public.product_dimension
WHERE (category_description, department_description, product_cost) IN (
    SELECT category_description, department_description,
    MAX(product_cost) FROM product_dimension
    GROUP BY category_description, department_description);
```

Example output

| product_description | sku_number | department_description |
|-------------------------------------|------------|------------------------|
| Brand #7979 cheddar cheese | SKU-#7979 | Dairy |
| Brand #2197 sushi | SKU-#2197 | Seafood |
| Brand #28902 strawberries | SKU-#28902 | Produce |
| Brand #54595 sliced turkey | SKU-#54595 | Meat |
| Brand #26127 chocolate chip cookies | SKU-#26127 | Bakery |
| Brand #32608 chocolate ice cream | SKU-#32608 | Frozen Goods |
| Brand #27213 shrimp | SKU-#27213 | Seafood |
| Brand #12533 canned green beans | SKU-#12533 | Canned Goods |
| Brand #3957 canned tuna | SKU-#3957 | Canned Goods |
| Brand #22103 cod | SKU-#22103 | Seafood |
| ... | | |

vmart_query_08.sql

Query

```
-- vmart_query_08.sql
-- Using pre-join projections to answer subqueries
-- between online_sales_fact and online_page_dimension

SELECT page_description, page_type, start_date, end_date
FROM online_sales.online_sales_fact f, online_sales.online_page_dimension d
WHERE f.online_page_key = d.online_page_key
AND page_number IN
    (SELECT MAX(page_number)
     FROM online_sales.online_page_dimension)
AND page_type = 'monthly' AND start_date = '2003-06-02';
```

Example output

| page_description | page_type | start_date | end_date |
|----------------------------|-----------|------------|------------|
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |
| Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11 |

```

Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11
Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11
Online Page Description #1 | monthly | 2003-06-02 | 2003-06-11
(12 rows)

```

vmart_query_09.sql

Query

```

-- vmart_query_09.sql
-- Equi join
-- Joins online_sales_fact table and the call_center_dimension
-- table with the ON clause

SELECT sales_quantity, sales_dollar_amount, transaction_type, cc_name
FROM online_sales.online_sales_fact
INNER JOIN online_sales.call_center_dimension
ON (online_sales.online_sales_fact.call_center_key
    = online_sales.call_center_dimension.call_center_key
    AND sale_date_key = 156)
ORDER BY sales_dollar_amount DESC;

```

Example output

| sales_quantity | sales_dollar_amount | transaction_type | cc_name |
|----------------|---------------------|------------------|-------------------|
| 7 | 513 | purchase | Southeastern |
| 3 | 439 | purchase | Southwest |
| 10 | 425 | purchase | North Midwest |
| 5 | 364 | purchase | North Midwest |
| 7 | 320 | purchase | Pacific Northwest |
| 2 | 314 | purchase | Pacific Northwest |
| 9 | 299 | purchase | California |
| 9 | 265 | purchase | Central Midwest |
| 9 | 247 | purchase | Southwest |
| 6 | 221 | purchase | Central Midwest |
| 1 | 198 | purchase | Central Midwest |
| 5 | 177 | purchase | Central Midwest |
| 7 | 131 | purchase | Southwest |
| 10 | 110 | purchase | North Midwest |
| 2 | -329 | return | Other |
| ... | | | |

Tutorial: Setting up an Example Database

Prerequisites

Before you proceed, HP Vertica must be installed on one host or a cluster of hosts, as described in the Installation Guide. HP recommends a minimum of three hosts in the cluster.

Audience

This tutorial targets anyone who wants to learn how to create and run an HP Vertica database. No special database knowledge is required at this point, though a rudimentary knowledge of basic SQL commands could be useful when you begin to run queries.

Objectives

You'll follow the simple steps below to create a fully-functioning, comprehensive design using the **VMart example database** (page [10](#)).

- 1 **Set up the example environment** (page [33](#))
- 2 **Create the example database** (page [34](#))
- 3 **Define the database schema** (page [37](#))
- 4 **Load the data** (page [38](#))
- 5 **Create a comprehensive design** (page [39](#))
- 6 **Connect to the database and run a simple query** (page [42](#))
- 7 **Test the optimized design** (page [43](#))
- 8 **(Optional) Generate custom data files** (page [45](#))

It's that easy! The whole process takes about 15 minutes, and when you are finished, you can proceed directly to **Running Simple Queries** (page [47](#)).

Notes

- This tutorial uses an HP Vertica-provided query, but you can follow the same set of procedures later, when you create your own design and use your own queries file.
- If, in the future, you have a query that you want to optimize, you can create an enhanced (incremental) design with additional projections to be tuned specifically for the query you provide. See Creating a Query-specific Design Using the Database Designer in the Administrator's Guide.
- For additional information about managing your designs, see Designing a Physical Schema in the Administrator's Guide.

Step 1: Set Up the Example Environment

In this procedure, you set up the example Vmart database environment.

- 1 Stop all databases running on the same host on which you plan to install your example database.

Tip: If you are unsure, run the Administration Tools and select View Database Cluster State. The State column should show DOWN values on pre-existing databases.

- 2 Log in to a terminal using the database administrator account that was created during product installation.

The default account name is `dbadmin`.

- 3 Create a directory for the example files on the Administration Host, such as under `/tmp`. This tutorial uses a `/tmp/examples` folder structure:

```
$ cd /tmp
$ mkdir examples
```

Note: Do not use the default data directory `/home/dbadmin`.

- 4 Copy the the **Vmart example database** (page [10](#)) files to the examples directory.

If you installed the product rpm on a database server, the example databases are located in `/opt/vertica/examples` on the host.

```
$ cp -r /opt/vertica/examples/VMart_Schema/* /tmp/examples
```

- 5 Set your current directory to the example database directory you created:

```
$ cd examples
```

Note: Do not change directories while following this tutorial. Some of the steps depend on being set to a specific directory.

- 6 Run the sample data generator program:

```
$ ./vmart_gen
```

Let the program run with the default parameters, which you can review in the README file.

Tip: If you are using VMware, the fact table load could fail. Specify a smaller fact table size, such as 1000000 (1M) rows, as described in **Step 8: (Optional) Generate Custom Data Files** (page 45). The maximum size of a bulk load depends on the system resources and cannot be determined accurately.

```
Using default parameters
datadirectory = ./
numfiles = 1
seed = 20177
null = ''
timefile = Time.txt
numfactsalesrows = 5000000
numfactororderrows = 300000
numprodkeys = 60000
numstorekeys = 250
numpromokeys = 1000
numvendkeys = 50
numcustkeys = 50000
numempkeys = 10000
numwarehousekeys = 100
numshippingkeys = 100
numonlinepagekeys = 1000
numcallcenterkeys = 200
numfactonlinesalesrows = 5000000
numinventoryfactrows = 300000
gen_load_script = false
Data Generated successfully !
```

If the `vmart_gen` executable does not work correctly, recompile it, as follows, and run the sample data generator script again. For example:

1. `$ g++ vmart_gen.cpp -o vmart_gen`
2. `$ chmod +x vmart_gen`
3. `$./vmart_gen`

(This example uses the GNU C++ compiler, which is a **free download** (<http://gcc.gnu.org/>). You can use any other C++ compiler.)

- 7 Proceed to **Step 2** (page 34).

Step 2: Create the Example Database

In this procedure, you create the example database using the Administration Tools. You can also choose to use the Management Console.

Notes

- If you have not used Administration Tools before, see the **Administration Tools Keystrokes** (on page 49) for a quick reference. If you are using a remote terminal application, such as PuTTY or a Cygwin bash shell, see **Notes for Remote Terminal Users** (page 50).
- If you want to use Management Console, the console should already be installed and you should be familiar with its concepts and layout. See Management Console in the Concepts Guide and Installing and Configuring Management Console in the Installation Guide.

Create the Example Database Using the Administration Tools

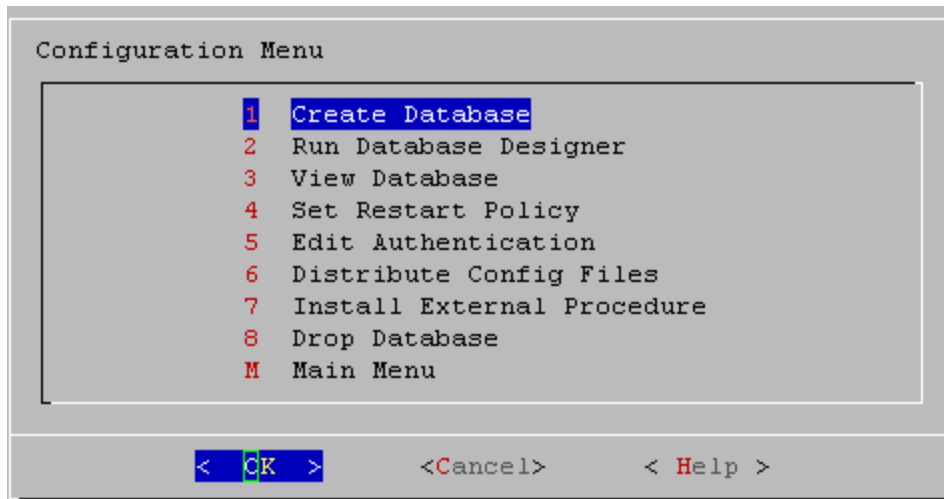
- 1 Run the Administration Tools.

```
$ /opt/vertica/bin/admintools
```

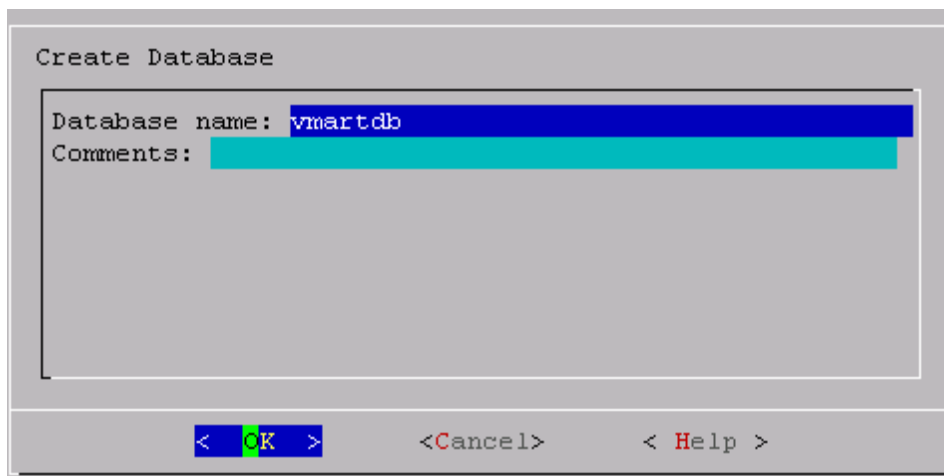
- 2 Accept the license agreement (once only).
- 3 Specify the location of your license key file (once only).

Note: HP Vertica Community Edition users do not need to complete this step. Their license is included in the Community Edition rpm.

- 4 From the Administration Tools Main Menu, click **Configuration Menu** and click **OK**.
- 5 Click **Create Database** and click **OK**.



- 6 Name the database **vmartdb** and click **OK**.

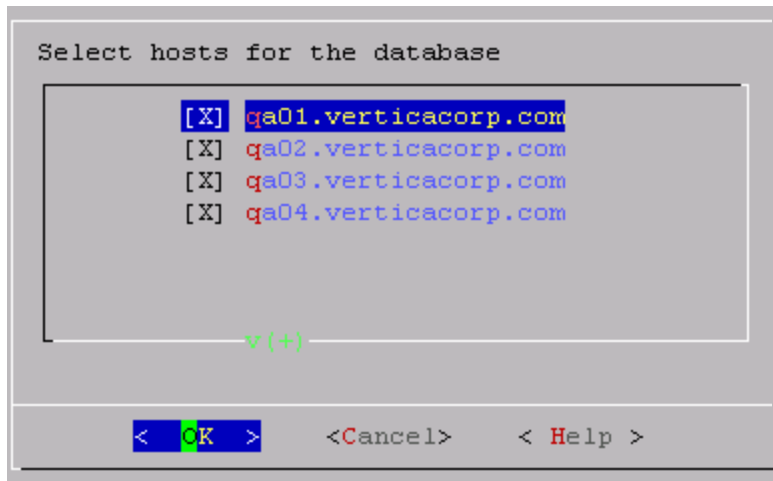


- 7 Click **OK** to bypass the password, and click **Yes** to confirm.

Note: There is no need for a database superuser password in this tutorial. When you create a production database, however, always specify a superuser password. Otherwise, the database is permanently set to trust authentication (no passwords).

- 8 Select the hosts you want to include from your HP Vertica cluster and click **OK**.

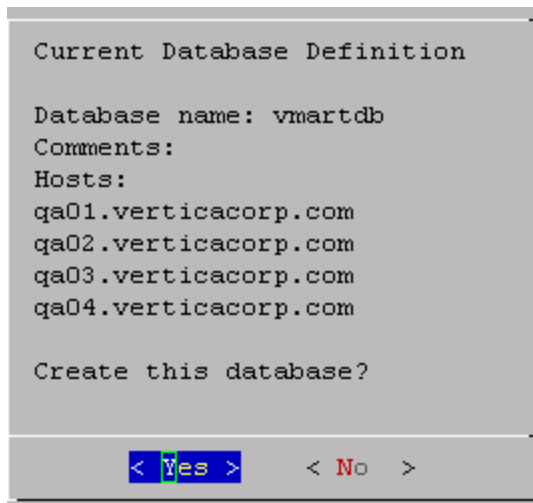
This example creates the database on a 4-host cluster. (Note that if you are using the HP Vertica Community Edition, you are limited to 3-nodes only.)



- 9 Click **OK** to select the default paths for the data and catalog directories.

- Catalog and data paths must contain only alphanumeric characters and cannot have leading space characters. Failure to comply with these restrictions could result in database creation failure.
- When you create a production database, you'll likely specify other locations than the default. See Prepare Disk Storage Locations in the Administrator's Guide for more information.

- 10 Click **Yes** to create the database.



Note: During database creation, HP Vertica automatically creates a set of node definitions based on the database name and the names of the hosts you selected and returns a success message.

- 11 Click **OK** to close the "Database vmartdb created successfully" message.
- 12 Proceed to **Step 3** (page [37](#)).

Create the Example Database Using Management Console

- 1 Connect to Management Console and log in.
Note: If you need login credentials, see the DBA who administers your Management Console.
- 2 On the **Home** page under Tasks, click **Manage Databases**.
- 3 On the **Databases and Clusters** page, click **Create database on existing cluster**.
- 4 Follow the on-screen wizard, which prompts you to provide the following information:
 - Database name, which must be between 3-25 characters, starting with a letter, and followed by any combination of letters, numbers or underscores.
 - (Optional) database password of the administration user (database superuser) for the database you want to create and connect to.
 - IP address of a node in your database cluster, typically the IP address of the administration host.
- 5 Click **Next** and proceed to **Step 3** (page [37](#)).

See Also

Managing Databases in the Concepts Guide

Step 3: Define the Database Schema

Now that you have created a database, define the schema.

- 1 On the Administration Tools **Configuration Menu**, click **Main Menu** and click **OK**.
- 2 Click **Connect to Database** and click **OK**.

You'll see the following prompt:

```
Welcome to vsql, the Vertica Analytic Database interactive terminal.
```

```
Type:  \h or \? for help with vsql commands
        \g or terminate with semicolon to execute query
        \q to quit
```

```
=>
```

- 3 To create the logical schema, run the SQL schema definition script using the `\i` meta-command in vsql:

```
vmartdb=> \i vmart_define_schema.sql
```

A series of CREATE TABLE and ALTER TABLE statement scrolls on the terminal window:

```
vmartdb=> \i vmart_define_schema.sql
CREATE SCHEMA
CREATE SCHEMA
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
ALTER TABLE
CREATE TABLE
CREATE TABLE
ALTER TABLE
CREATE TABLE
ALTER TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
ALTER TABLE
vmartdb=>
```

The `vmart_define_schema.sql` file creates the tables and referential integrity constraints that make up the logical schema.

- 4 Proceed to **Step 4** (page [38](#)).

Step 4: Load the Data

In this brief step, you'll load data into the schema you created in the previous step. HP Vertica automatically creates a superprojection for each table into which data is loaded.

- 1 Load data into the VMart database using the `vmart_load_data.sql` script.

```
vmartdb=> \i vmart_load_data.sql
```

```

      200
(1 row)

Rows Loaded
-----
      5000000
(1 row)

Rows Loaded
-----
      3000000
(1 row)

Rows Loaded
-----
      5000000
(1 row)

Rows Loaded
-----
      3000000
(1 row)

vmartdb=> █
```

Note: It could take several minutes to load the default five-million row fact table on a typical hardware cluster. You can check the load by examining the `vertica.log` file, as described in Monitoring Log Files in the Administrator's Guide.

- 2 Proceed to **Step 5** (page [39](#)).

Step 5: Create a Comprehensive Design

In this procedure you'll create a comprehensive design using Database Designer through the Administration Tools interface. These steps assumes that you have already performed the following prerequisite steps:

- 1 **Set up the example environment** (page [33](#))
- 2 **Created the example database** (page [34](#))
- 3 **Defined the database schema** (page [37](#))
- 4 **Loaded the data** (page [38](#))

Note: If you have a query you want to optimize after you create a comprehensive design, you can create an incremental design later. See Creating a Query-specific Design Using the Database Designer for details.

Create the comprehensive design using the Database Designer

- 1 To exit the vsql session and return to the Main Menu in the Administration Tools, type `\q`. Alternatively, restart the Administration Tools:

```
$ /opt/vertica/bin/admintools
```

2 From the **Main Menu**, click **Configuration Menu** and click **OK**.

3 From the **Configuration Menu**, click **Run Database Designer** and click **OK**.

4 Select **vmartdb** as the database and click **OK**.

If you are asked to enter the password for the database, click **OK** to bypass. No password was assigned in **Step 2: Create the Example Database** (page [34](#)), so you do not need to enter one now.

5 Click **OK** to accept the default directory (`/tmp/examples`, unless you changed it) for storing Database Designer output and log files. **Note this location**.

Note: If you choose to not deploy your design now, Database Designer saves the SQL script to deploy the design in the default directory where you can review and manually deploy it later.

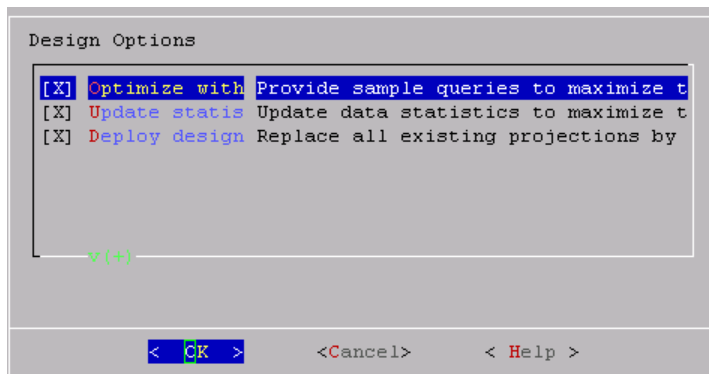
6 In the **Database Designer** window, enter a name for the design, for example, **vmart_design**, and click **OK**. Design names can contain only alphanumeric characters or underscores. No other special characters are allowed.

7 To create a complete initial design, in the **Design Type** window, click **Comprehensive** and click **OK**.

8 Because the Vmart design is a multi-schema database, select all three schemas for your design, and click **OK**.

If you include a schema that contains tables without data, the Administration Tools notifies you that designing for tables without data could be suboptimal. You can choose to continue, but HP recommends that you click **Cancel** and deselect the schemas that contain empty tables before you proceed.

9 In the **Design Options** window, because the Vmart design is a multi-schema database, accept all three options (described below) and click **OK**.



Generally, you enable all three options because Database Designer is best positioned to generate a new comprehensive design and create a complete set of projections for the tables in the selected schema. The three options are:

- **Optimize with queries:** Supplying the Database Designer with queries is especially important if you want to optimize the database design for query performance.

Database Designer does not impose hard limits to the number of queries or tables it accepts as input. However, it is limited by system resources, concurrent loads, and query/schema complexity. HP recommends that you limit the design input to 100 queries.

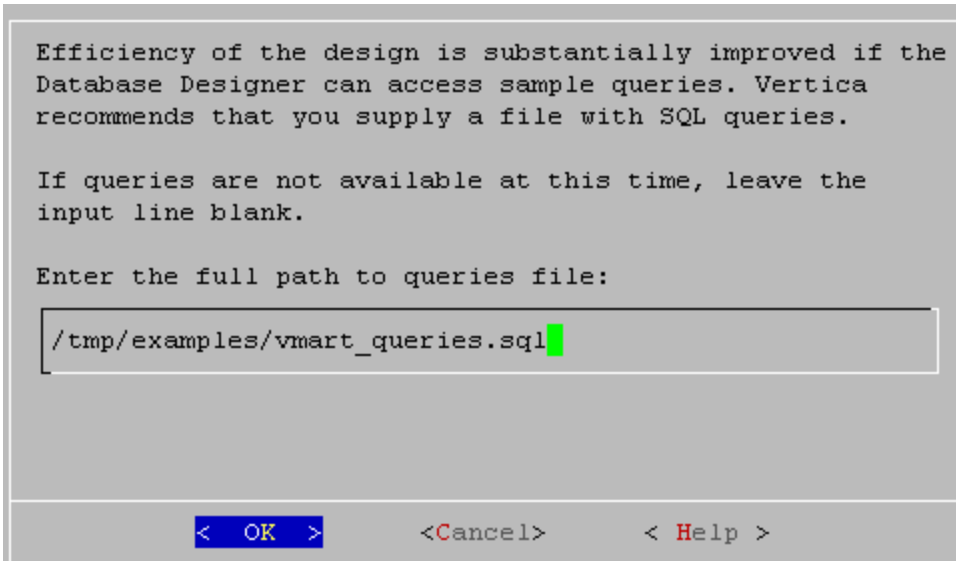
- **Update statistics:** Accurate statistics help the Database Designer choose the best strategy for data compression. If you select this option, the database statistics are updated to maximize design quality.

Updating statistics takes time and resources, so if the current statistics are up to date, this step is unnecessary. When in doubt, update statistics.

- **Deploy design:** The new design is automatically deployed, which means that during deployment, new projections are added, some existing projections might be retained, and any unnecessary existing projections are removed. Any new projections are refreshed so that they are populated with data.

Note: For large databases, a full design session could take a long time, but it is best to allow this process to complete uninterrupted. If the session must be canceled, use Ctrl+C.

- 10** If you selected the **Optimize with queries** option, you must enter the full path to the file containing the queries that will be run on your database. In this example it is:
/tmp/examples/vmart_queries.sql



Efficiency of the design is substantially improved if the Database Designer can access sample queries. Vertica recommends that you supply a file with SQL queries.

If queries are not available at this time, leave the input line blank.

Enter the full path to queries file:

/tmp/examples/vmart_queries.sql

< OK > < Cancel > < Help >

The queries in the query file must be delimited with a semicolon (;).

Note: Although there is no hard limit to the number of queries or tables you can provide as input to a comprehensive design, Database Designer is limited by system resources, concurrent loads, and query/schema complexity. HP recommends that you limit the design input to 100 queries.

- 11** Choose the **K-safety value** you want. This example uses 1. Click **OK**.

Note: If you are creating a comprehensive design on a single node, you are not asked to enter a K-safety value.

- 12** Choose **Balanced** for the Database Designer's design priority and click **OK**.

The design priorities are:

- **Balanced** query/load performance tells Database Designer to create a design that is balanced between database size and query performance.
- **Query** load performance creates a design focused on faster query performance, which might recommend additional projections. These projections could result in a larger database storage size.

- **Load** performance is optimized for loads, minimizing database size, potentially at the expense of query performance.

13 When the informational message displays, click **Proceed**.

Database Designer:

- Sets up the design session
- Examines table data
- Loads queries from the query file you provided
- Creates the design
- Deploys the design or saves a SQL file containing the design, depending on what you selected for the Deploy design option in step 9.

Depending on system resources, the design process could take several minutes.

14 When Database Designer finishes, press **Enter** to return to the Administration Tools menu.

15 After you have created your design, query the system table `DESIGN_STATUS` to see the steps taken to create the design. If you also deployed the design, those steps are listed in the system table `DEPLOY_STATUS`:

```
vmartdb=> SELECT * FROM V_MONITOR.DESIGN_STATUS;  
vmartdb=> SELECT * FROM V_MONITOR.DEPLOY_STATUS;
```

When you run Database Designer using the Administration Tools, it creates a backup of the current design of your database before deploying the new design. This backup is stored in the directory you specified in step 5 and is named `catalog_dump.sql`.

16 Proceed to **Step 6** (page [42](#)).

Step 6: Connect to the Database and Run a Simple Query

Proceeding directly from Step 5:

- 1** Click **Main Menu** and click **OK**.
- 2** Click **Connect to Database** and click **OK**.

The vsql welcome prompt displays:

```
Welcome to vsql, the Vertica Analytic Database interactive terminal.
```

```
Type:  \h or \? for help with vsql commands  
        \g or terminate with semicolon to execute query  
        \q to quit
```

```
vmartdb=>
```

- 3** Use the `\i` meta-command to execute the HP Vertica-provided example query script:

```
vmartdb=> \i vmart_query_03.sql
```

You results will be similar to:

| customer_name | annual_income |
|------------------|---------------|
| Emily G. Vogel | 999998 |
| James M. McNulty | 999979 |

(2 rows)

4 Proceed to **Step 7** (page [43](#)).

See Also

- **Running Simple Queries** (page [47](#))
- Creating a Query-specific Design Using the Database Designer in the Administrator's Guide

Step 7: Test the Optimized Design

Check query execution times to test your optimized design:

- 1 Use the `vsq| \timing` meta-command to enable the display of query execution time in milliseconds.

Execute a SQL sample query script to test your schema and load scripts for errors.

Note: Include a sample of queries your users are likely to run against the database. If you don't have any real queries, just write simple SQL that collects counts on each of your tables. Alternatively, you can skip this step.

- 2 Execute several ad hoc queries

1. Run Administration Tools and select Connect to Database.
2. Use the `\i` meta-command to execute the query script; for example:

```
vmartdb=> \i vmart_query_03.sql
```

| customer_name | annual_income |
|------------------|---------------|
| James M. McNulty | 999979 |
| Emily G. Vogel | 999998 |

(2 rows)

Time: First fetch (2 rows): 58.411 ms. All rows formatted: 58.448 ms

```
vmartdb=> \i vmart_query_06.sql
```

| store_key | order_number | date_ordered |
|-----------|--------------|--------------|
| 45 | 202416 | 2004-01-04 |
| 113 | 66017 | 2004-01-04 |
| 121 | 251417 | 2004-01-04 |
| 24 | 250295 | 2004-01-04 |
| 9 | 188567 | 2004-01-04 |
| 166 | 36008 | 2004-01-04 |
| 27 | 150241 | 2004-01-04 |
| 148 | 182207 | 2004-01-04 |
| 198 | 75716 | 2004-01-04 |

(9 rows)

Time: First fetch (9 rows): 25.342 ms. All rows formatted: 25.383 ms

Once the database is optimized, it should run queries efficiently. If you discover queries that you want to optimize, you can modify and update the design. See [Modifying Designs and Creating a Query-specific Design Using the Database Designer in the Administrator's Guide](#).

Proceed to the optional **Step 8** (page [45](#)).

Step 8: (Optional) Generate Custom Data Files

The example database provided with HP Vertica includes a sample data generator program that produces output files whose names correspond to the tables in the logical schema. Each data generator has a similar set of input parameters that allow you to specify the number of rows of data to generate for any subset of the tables. To see a detailed list of the parameters for any example database, examine the README file in the example database directory.

Tip: You can repeat the tutorial using custom data files to test larger data sizes.

Syntax

```
./example_gen [ --files files ]
               [ --seed seed ]
               [ --time_file path ]
               [ --fact_table_name rows ]
               [ --dimension_table_name rows ] ...
```

Parameters

| | |
|-----------------------------------|---|
| <code>vmart_gen</code> | The VMart file generator. |
| <code>files files</code> | <p>Splits the fact table data into the specified number of files. By default, the data generator produces a single, unnumbered fact table data file. If you specify a value of two (2) or more, the data generator numbers the files by appending an underscore character (_) and three digits to the file name, starting at _001. For example:</p> <pre>./vmart_gen --files 3</pre> <p>produces:</p> <pre>VMart_Fact_001.tbl VMart_Fact_002.tbl VMart_Fact_003.tbl</pre> <p>Default: 1</p> |
| <code>seed seed</code> | <p>Is the seed for the pseudo-random number generator. If you use the same seed each time you run the data generator, you get the same data files (excluding external factors); for example, <code>seed 9999</code>.</p> <p>Default: 20177</p> |
| <code>time_file path</code> | <p>Is the pathname of the pre-computed time data input file used to generate the Date Dimension table.</p> <p>Default: <code>./Time.txt</code></p> <p>This file is provided for each example database and the date range may vary; for example 2000-2004 or 2003-2007.</p> |
| <code>fact_table_name rows</code> | <p>Is the name of the fact table in <code>vmart_gen</code> followed by the number of rows of data to generate for the fact table.</p> <p>Default: 5,000,000 (five million)</p> |

| | |
|--|--|
| <code>dimension_table_name rows</code> | Is the name of a dimension table in <code>vmart_gen</code> (other than the <code>Date_Dimension</code> table) followed by the number of rows of data to generate for that dimension table. |
|--|--|

Notes

- The number of rows in `Date_Dimension` tables is determined by the time data input file supplied with the example database.
- If you are using multiple fact table data files, make sure that your fact table load script(s) contain the correct file names as described in Using Load Scripts.

Examples

```
./vmart_gen
```

```
./vmart_gen --files 3
```

```
/home/dbadmin/Vmart_Schema/examples/vmart_gen \  
--seed 9999  
--time_file /home/dbadmin/Vmart_Schema/examples/Time.txt \  
--inventory_fact 100000 \  
--customer_dimension 500 \  
--date_dimension 500 \  
--employee_dimension 50 \  
--product_dimension 500 \  
--promotion_dimension 500 \  
--shipping_dimension 500 \  
--vendor_dimension 500 \  
--warehouse_dimension 500 \  
--promotion_dimension 100
```

Running Simple Queries

The VMart example database includes example SQL queries that represent the kinds of queries you might use in a production database. If you copy the query files to a client system, you can connect to the example database and execute the queries using any of the methods described in the Programmer's Guide.

To run an example query using `vsql` on a cluster host:

1 Run Administration Tools and select Connect to Database.

```
Welcome to vsql, the Vertica Analytic Database interactive terminal.
```

```
Type:  \h or \? for help with vsql commands
        \g or terminate with semicolon to execute query
        \q to quit
```

```
=>
```

2 Use the `\i` meta-command to execute the query script:

```
vmartdb=> \i vmart_query_01.sql
```

See **VMart Sample Queries** (page [25](#)) for the example queries supplied with HP Vertica.

Cleanup Procedure

If you want to clean up your host and start over from scratch, use the following steps.

Drop the database

- 1 In a terminal window, log in to the database administrator account that was created by the installation script. The default account name is `dbadmin`.
- 2 Run the Administration Tools.
`$ /opt/vertica/bin/admintools`
- 3 If necessary, stop any running database (Main Menu **Stop Database**).
- 4 Click **Configuration Menu** and click **OK**.
- 5 Click **Drop Database** and click **OK**.
- 6 In the **Select database to drop** window, select the database you want to drop and click **OK**.
- 7 Click **Yes** to confirm.
- 8 In the next window type `yes` (lowercase) to confirm and click **OK**.

Uninstall HP Vertica

- 1 Perform the steps in Uninstalling HP Vertica in the Installation Guide.

Other

- 1 Optionally remove the `dbadmin` account on all cluster hosts.
- 2 Remove any example database directories you created.

See Also

For complete descriptions of each Admin Tools dialog, refer to the Administration Tools Reference in the Administrator's Guide.

Administration Tools Keystrokes

This is only a quick reference. It is not a complete guide to keystroke usage. See Using the Administration Tools in the Administrator's Guide for full details.

| | |
|----------------------|---|
| Return | Run selected command. |
| Tab | Move cursor from OK to Cancel to Help to menu or to OK... |
| Up/Down Arrow | Move cursor up and down in menu, window, or help file. |
| Space | Select item in list. |
| Character | Select corresponding command from menu. |

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 PuTTY terminal application running on a Windows platform.

Note: If you are using a remote terminal application, such as PuTTY or a Cygwin bash shell, make sure your window is at least 81 characters wide and 23 characters high.

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

- 1 In a PuTTY window, right click the title area and select Change Settings.
- 2 Create or load a saved session.
- 3 In the Category dialog, click Window > Appearance.
- 4 In the Font settings, click the Change... button.
- 5 Select Font: Courier New: Regular Size: 10
- 6 Click Apply.

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

You can also change the translation to support UTF-8:

- 1 In a PuTTY window, right click the title area and select Change Settings.
- 2 Create or load a saved session.
- 3 In the Category dialog, click Window > Translation.
- 4 In the "Received data assumed to be in which character set" drop-down menu, select UTF-8.
- 5 Click Apply.

Copyright Notice

Copyright© 2006-2013 Hewlett-Packard, and its licensors. All rights reserved.

| |
|---|
| Hewlett-Packard 150 CambridgePark Drive Cambridge, MA 02140 Phone: +1 617 386 4400 E-Mail: info@vertica.com Web site: http://www.vertica.com (http://www.vertica.com) |
|---|

The software described in this copyright notice is furnished under a license and may be used or copied only in accordance with the terms of such license. Hewlett-Packard software contains proprietary information, as well as trade secrets of Hewlett-Packard, and is protected under international copyright law. Reproduction, adaptation, or translation, in whole or in part, by any means — graphic, electronic or mechanical, including photocopying, recording, taping, or storage in an information retrieval system — of any part of this work covered by copyright is prohibited without prior written permission of the copyright owner, except as allowed under the copyright laws.

This product or products depicted herein may be protected by one or more U.S. or international patents or pending patents.

Trademarks

HP Vertica™, the HP Vertica Analytics Platform™, and FlexStore™ are trademarks of Hewlett-Packard.

Adobe®, Acrobat®, and Acrobat® Reader® are registered trademarks of Adobe Systems Incorporated.

AMD™ is a trademark of Advanced Micro Devices, Inc., in the United States and other countries.

DataDirect® and DataDirect Connect® are registered trademarks of Progress Software Corporation in the U.S. and other countries.

Fedora™ is a trademark of Red Hat, Inc.

Intel® is a registered trademark of Intel.

Linux® is a registered trademark of Linus Torvalds.

Microsoft® is a registered trademark of Microsoft Corporation.

Novell® is a registered trademark and SUSE™ is a trademark of Novell, Inc., in the United States and other countries.

Oracle® is a registered trademark of Oracle Corporation.

Red Hat® is a registered trademark of Red Hat, Inc.

VMware® is a registered trademark or trademark of VMware, Inc., in the United States and/or other jurisdictions.

Other products mentioned may be trademarks or registered trademarks of their respective companies.

Information on third-party software used in HP Vertica, including details on open-source software, is available in the guide [Third-Party Software Acknowledgements](#).