



Hewlett Packard
Enterprise

Vertica Integration with Tableau 10: Tips and Techniques

HPE Vertica Analytic Database

Document Release Date: 10/10/2016

Legal Notices

Warranty

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

The information contained herein is subject to change without notice.

Restricted Rights Legend

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

Copyright Notice

© Copyright 2006 - 2016 Hewlett Packard Enterprise Development LP

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Apache® Hadoop® and Hadoop are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries.

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

UNIX® is a registered trademark of The Open Group.

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

Contents

Introduction	6
Audience	6
About Tableau	6
Tableau Desktop	6
Tableau Server	7
Connecting Vertica and Tableau	7
Client Driver and Server Version Compatibility	8
Vertica ODBC/JDBC Client Installers	8
ODBC/JDBC Multiple Version Installations	9
Vertica ADO.NET Client Installers	9
New Features in Tableau 9 and Later	10
New Features in Tableau 10 and Later	11
Tips for Connecting to HPE Vertica from Tableau	12
Parallel Query Processing	12
Connection Pools	13
Set the Maximum Number of Connections for a Single Data Source	14
Set the Maximum Number of Connections for All Connections	15
Vertica MaxClientSessions Parameter	16
Live Connections Compared to Extracts	16
Multiple Table Connections Compared to Custom SQL	17
When to Use Custom SQL	17
Join Culling in Tableau	18
Checking for Well-Constructed Joins	18
Cross-Database Joins	20
Adding or Excluding Columns: Tableau 8	22
Tableau Best Practices	26
Tips for Dashboard Design	26
Tips for Using Parameters	27
Tips for Using Filters	28
Quick Filters	28
Quick Filters that Do Not Require Extra Queries	29
Quick Filters that Require Extra Queries	29
Filtering Ranges of Values	30

Vertica Integration with Tableau 10: Tips and Techniques

Filtering Discrete Values	30
Filtering Dimensions	31
Filtering Dates	32
Filtering Measures	32
Filter Actions and Data Source Filters	33
Context Filters	33
Cross-Data Source Filters	34
Tips for Calculations	34
Tableau Built-In Functions	35
Regular Expression Functions	36
How Regular Expression Functions Work	37
Table Calculations	38
Pass-Through Functions	39
Sets	40
Vertica Tuning Recommendations	41
Upgrade Vertica for More Efficient Query Processing	41
Create a Physical Design with Database Designer	41
Identify Sample Queries for Database Designer	42
Check Database Designer Projections	43
Use Live Aggregate Projections	43
Tips for Managing Vertica Resources	44
Create a Separate Resource Pool for Tableau	44
Create Cascading Resource Pools	45
Enabling Database Isolation Levels in Tableau	46
Enabling Native Connection Load Balancing from Tableau	46
Customize Your Connection to Vertica	48
TDC and TDS File Locations	48
Customization Details	49
Global Data Source Customizations	51
Create a TDC File	51
Important: Embedded Custom TDC Files	52
Check the TDC File	53
Single Data Source Customizations	53
Create a TDS File	54
Edit the TDS File	55
Apply the Modified TDS File	56
Check the TDS File	57
Troubleshooting Tools	58

Vertica Integration with Tableau 10: Tips and Techniques

Performance Recorder	58
Using Performance Recorder on Tableau Desktop	58
Using Performance Recorder on Tableau Server	59
Session Labels	59
Log Files	60
Tableau Server Log File	63
Support for Vertica Data Types	64
Known Issues and Workarounds	66
Multiple Active Result Sets Per Session	66

Introduction

This document describes tips and techniques that enhance your experience using Tableau Desktop and Tableau Server with Vertica.

Audience

This document is intended for customers using Tableau Desktop with Vertica.

Familiarity with Vertica and Tableau is assumed.

For more information, see:

- [Vertica Documentation](#)
- [Tableau Documentation](#)
- [Tableau User Community](#)

About Tableau

Tableau is a powerful and flexible business analytics tool that helps you quickly analyze, visualize, and share information. Tableau is available for Windows and Mac OS, and uses ODBC to connect to Vertica.

Tableau Desktop

Tableau Desktop allows you to connect to data and create visualizations and interactive dashboards with an easy-to-use, drag-and-drop interface.

For more information, see:

- To understand how Tableau works with data, read [Tableau Desktop: Overview](#).
- To understand Tableau query performance, watch the video [Understanding Tableau Query Performance](#).
- For an overview of tuning Tableau, read [Tuning Tableau and Your Database for Great Performance](#).

Tableau Server

Tableau Server is a web-based application that helps you publish and share the workbooks that you create using Tableau Desktop.

There are several deployment options for Tableau Server:

- Tableau Server is hosted by users on premises.
- Tableau Public is free and available to all users on the cloud.
- Tableau Online is Tableau Server hosted on Amazon Web Services (AWS) and managed by Tableau. Tableau Online ensures that users can only see their own data, not other users' data.

Connecting Vertica and Tableau

Before you connect Vertica and Tableau, download and install an Vertica ODBC driver from the [Vertica Client Drivers](#) website.

You connect using Tableau's Vertica connector, also known as the native Vertica connector. Tableau has optimized this connector for excellent performance between Tableau and Vertica.

For step-by-step instructions on connecting to Vertica using Tableau, see [Vertica Integration with Tableau Desktop: Connection Guide](#)

To learn more about Tableau and ODBC, see this [knowledge base article](#).

Client Driver and Server Version Compatibility

Usually, each version of the Vertica server is compatible with the previous version of the client drivers. This compatibility lets you upgrade your Vertica server without having to immediately upgrade your client software. However, some new features of the new server version may not be available through the old drivers.

The following table summarizes the compatibility of each recent version of the client drivers with the Vertica server versions.

Client Driver Version	Compatible Server Versions
6.1.x	6.1.x, 7.0.x, 7.1.x, 7.2.x, 8.0.x
7.0.x	7.0.x, 7.1.x, 7.2.x, 8.0.x
7.1.x	7.1.x, 7.2.x, 8.0.x
7.2.x	7.2.x, 8.0.x
8.0.x	8.0.x

Note: For Vertica FIPS-compliance, you must use server and client versions 8.0.x.

Vertica ODBC/JDBC Client Installers

The ODBC/JDBC client drivers are a separate installation from the ADO.NET drivers. (ADO.NET support is not available in Community Edition.) As noted in the compatibility table, the 6.x ODBC/JDBC client drivers do not support access to a non Vertica 6.x database and above. For example, you cannot use the new 6.x ODBC/JDBC client drivers to access a Vertica 5.x database. If you plan on having a mixed Vertica environment supporting both 5.x and 6.x Vertica database, consider keeping the 5.x drivers installed.

ODBC/JDBC Multiple Version Installations

The following ODBC/JDBC drivers are supported on a single machine:

- 4.x and 5.x ODBC/JDBC drivers can be installed on the same machine.
- 4.x and 6.x ODBC/JDBC drivers can be installed on the same machine.

It is not possible to have both 5.x and 6.x ODBC drivers on a single machine. If you install the 6.x version, it automatically overlays the existing 5.x installation, and any DSN defined against a 5.x Vertica database is not supported.

Vertica ADO.NET Client Installers

Prior to version 6.x, ADO.Net drivers must be uninstalled prior to installing a later version of the driver. The 6.x ADO.Net drivers require the Vertica database to be 6.0.0 or above. The ADO.NET 6.x driver only supports access to a Vertica 6.x server. The ADO.NET 4.x plug-in does not work with a Vertica 6.x server. If you plan on also using the ODBC bridge and you need to access both Vertica 5.x and 6.x databases, consider keeping the 5.x versions of the ODBC/JDBC drivers for the reasons stated previously.

New Features in Tableau 9 and Later

To find out about how these new features from Tableau 9 and later affect connecting to and working with Vertica, see the following topics in this document:

- [Parallel query processing](#)—Tableau can now open multiple connections to Vertica in order to send query requests in multiple concurrent threads.
- [Level-of-detail expressions](#)—Level of detail (LOD) expressions are a new type of calculation that allow users to compare data at several different aggregation levels. For example, you might want to compare the number of dropped calls in a certain area of country to the number of total dropped calls.

LOD expressions generate queries that Vertica computes. These queries include subqueries with inner joins and sometimes with cross joins.

- [Embedding TDC file customizations](#)—In Tableau Desktop, if you have created a custom Tableau Datasource Connection (TDC) file, Tableau automatically embeds the content of the *.tdc file into the workbook (*.twb).

New Features in Tableau 10 and Later

To find out about how these new features from Tableau 10 and later affect connecting to and working with Vertica, see the following topics in this document:

- [Cross-database joins](#)—Tableau can now join tables stored in different databases such as different Vertica databases.
- [Cross-data source filters](#)—Tableau can now filter multiple data sources at once based on a common dimension

Tips for Connecting to HPE Vertica from Tableau

The following sections describe how to configure your Tableau to Vertica connection so that you can optimize performance:

- [Parallel Query Processing](#)
- [Live Connections Compared to Extracts](#)
- [Multiple Table Connections Compared to Custom SQL](#)
- [Join Culling in Tableau](#)
- [Checking for Well-Constructed Joins](#)
- [Add or Exclude Columns](#)

For detailed information about connecting Tableau to Vertica, see

- [HPE Vertica Integration with Tableau: Connection Guide](#)
- [HPE Vertica](#) in the Tableau documentation

Parallel Query Processing

Tableau can now open multiple connections to Vertica in order to send query requests in multiple concurrent threads. Prior to Tableau 9, queries were always executed sequentially.

Tableau automatically creates as many connections as there are queries to execute in parallel, as long as it does not exceed the limit for parallel connections. (By default, the limit is 16 connections.)

However, the queries must be independent to run in parallel. If queries depend on the results of previous queries, Tableau sends the queries to Vertica sequentially.

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau

Use caution when taking advantage of this new parallel processing capability. Depending on the number of concurrent users, resource pool configuration settings and dashboard content, you may want to decrease the default number of allowed connections to avoid potential resource contention issues.

To view the behavior of parallel query processing, see the queries in:

- Tableau Performance Recorder, as described in [Interpret a Performance Recording Workbook](#).
- HP Vertica system table [QUERY_REQUESTS](#).

For information about changing the connection limit, see

- [Set the Maximum Number of Connections for a Single Data Source](#)
- [Set the Maximum Number of Connections for All Connections](#)

Connection Pools

Tableau creates connection pools for each unique connection string generated for the data sources in your workbooks. The default maximum number of connections in a connection pool is 16, but you can modify this maximum.

For example, if you have three data sources to the same server with the same connection string, they would execute in the same connection pool. On the other hand, if you have three data sources pointing to the same server and database but with different connection strings (such as different user IDs and passwords), each data source will have in its own connection pool. You can modify the maximum number of connections for each connection pool independently.

Set the Maximum Number of Connections for a Single Data Source

You can control the maximum number of connections from Tableau to Vertica using connection properties such as class, server, and database.

To set limits for data sources or for Vertica on specific servers, create a configuration file with the desired settings. The configuration file is an XML file named `connection-configs.xml`. Tableau parses this from top to bottom and chooses the first match for a specific connection.

When evaluating what connection limit to use for a connection, Tableau reviews the order of the items in the `connection-list.xml` file. To create this file, take these steps:

1. Create an XML file named `connection-configs.xml`.
2. For Tableau Server, save the file in the `config` directory in the `vizqlserver` folder, for example:

```
C:\ProgramData\Tableau\TableauServer\data\tabsvc\config\vizqlserver
```

Remember to copy the configuration file to all the `vizqlserver` folders on all Tableau machines.

For Tableau Desktop, create and save the file in the Tableau Desktop folder.

```
C:\Program Files\Tableau\Tableau 9.0 on a Windows computer
```

Note: The preceding example uses Tableau 9.0. Verify that your folder name matches the version of Tableau that you are using.

3. Copy and paste the following content into the file. Make sure that you modify

the values for server, database, and number of connections to reflect your setup.

In this example file:

- The first connection class sets a limit of 4 connections for any Tableau workbook that uses a combination of 15.126.227.222 for a server name and verticanow as a database name.
- The second connection class sets a limit of 10 connections for any Tableau workbook that has a server name of 172.16.116.45.
- The third connection class sets a limit of 12 connections for all other Tableau connections to Vertica.

```
<?xml version='1.0' encoding='ntf-8 ?>
<connection-list>
  <connection class='vertica' server='15.126.227.222' dbname='verticanow'>
    <limit max='4'>
      </limit>
    </connection>
  <connection class='vertica' server='172.16.116.45'>
    <limit max='10'>
      </limit>
    </connection>
  <connection class='vertica'
    <limit max='12'>
      </limit>
    </connection>
</connection-list>
```

Set the Maximum Number of Connections for All Connections

For Tableau Desktop, to set a connection limit for all connections during a particular session, use the `-DConnectionLimit` command-line switch.

For example, to launch Tableau desktop with a connection limit of 2 for all connections, enter this in a command prompt:

```
"C:\Program Files\Tableau\Tableau 9.0\bin\tableau.exe" -DConnectionLimit=2
```

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau

Note: The preceding example uses Tableau 9.0. Verify that your folder name matches the version of Tableau that you are using.

For Tableau Server, you can disable parallel query execution for all connections by using `tabadmin` and setting the maximum number of connections to 1:

```
tabadmin set native_api.connection.limit.globallimit 1
```

To set the connection limit to, for example, 20:

```
tabadmin set native_api.connection.limit.globallimit 20
```

Vertica MaxClientSessions Parameter

In Vertica, the maximum number of concurrent connections per cluster is limited by physical RAM of a single node (or number of threads per process). The default limit per node is 50. You can modify the maximum number of connections with the `MaxClientSessions` parameter.

Before changing this setting, see the information in the Vertica documentation:

- [System Limits](#)
- [Managing Sessions](#)

Live Connections Compared to Extracts

After you connect to Vertica, you can maintain a live connection or you can create a data extract. A *data extract* is data from Vertica that you store locally, in Tableau.

Hewlett Packard Enterprise recommends that you use a live connection when:

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau

- You need to leverage Vertica analytic capabilities.
- The volume of data is not conducive to creating an extract.
- Your workbook uses pass-through RAWSQL functions.
- You need near real-time analysis.
- You need robust user-level security.

Use data extracts *only* when:

- You need offline access to the data.
- Many users are accessing a single workbook with data that is not needed in real time.

Multiple Table Connections Compared to Custom SQL

Hewlett Packard Enterprise recommends that you create a data source by dragging multiple tables to the join area, instead of using the **New Custom SQL** option on the **Data Sources** window. This recommendation applies when you construct inner and outer join queries.

When you join multiple tables, Tableau takes advantage of the join culling capability.

When to Use Custom SQL

When you use custom SQL, Tableau wraps additional GROUP BY, ORDER BY, and WHERE clauses around your query. As a result, custom SQL queries become subqueries. Therefore, use custom SQL only if Tableau cannot generate the desired SQL (for example, for queries with UNION clauses and self-join queries).

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau

When using custom SQL, consider moving your query to Vertica. To do so, create a view with the query that you would have used in the custom SQL connection and then connect to that view from Tableau.

For more information, see the following topics in the Tableau documentation:

- [HP Vertica](#)
- [Connecting to a Custom SQL Query](#)

Join Culling in Tableau

With join culling, Tableau does not use every join from your Vertica connection all the time. If you are joining many tables in your connection, using every join would be expensive. Instead, Tableau uses a join only when required. If a join does not impact the result, Tableau discards the join.

Tableau uses join culling automatically on tables that have foreign keys defined. If you don't have foreign keys defined, on the **Data** menu, you can enable join culling for a data source by selecting **Assume referential integrity**.

Checking for Well-Constructed Joins

When you create a new connection to Vertica, pay attention to the joins that Tableau generates.

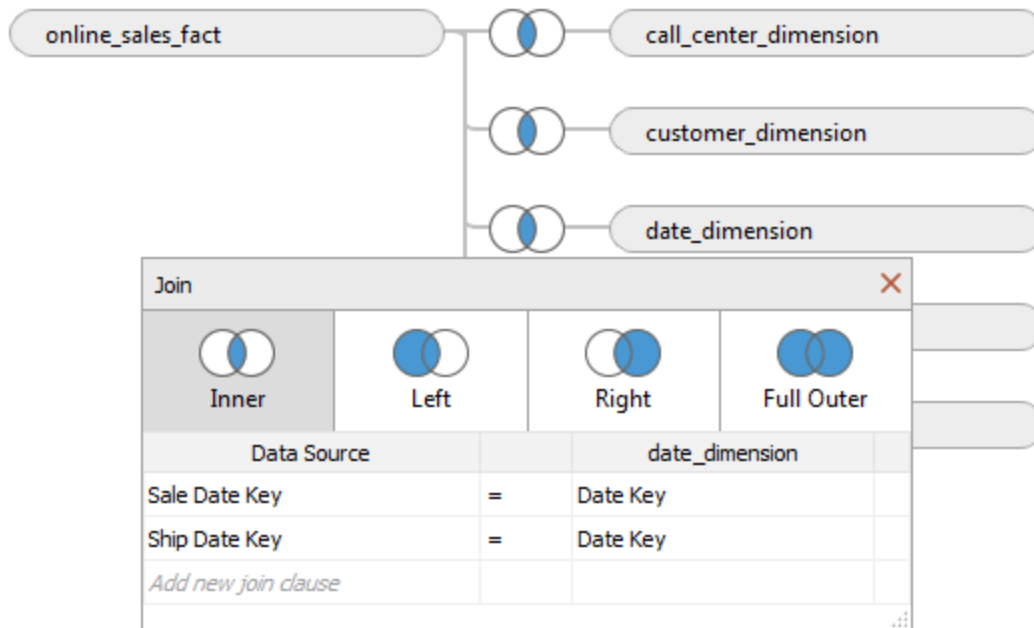
By default, Tableau performs joins based on the primary and foreign key relationships. These joins may not be correct for certain situations, for example, when a dimension table is used for more than one column in a fact table.

Consider an example. The columns `online_sales_fact.sale_date_key` and `online_sales_fact.ship_date_key` in the `online_sales_fact` table correspond to the same key from `date_dimension` table: `date_dimension.date_key`.

In this case, by default, Tableau creates a single join condition on the two keys. You can see this is not correct in the following graphic. The default behavior creates an incorrect join where the columns are in the same join:

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau



If the join is not correct, you can alter the join condition. For the correct behavior, shown in the following graphic, manually join each key to the fact table separately, in two joins:

Online Sales
Connected to HP Vertica

Server
172.16.116.149

Database
vmart

Schema
public

Table
Enter table name

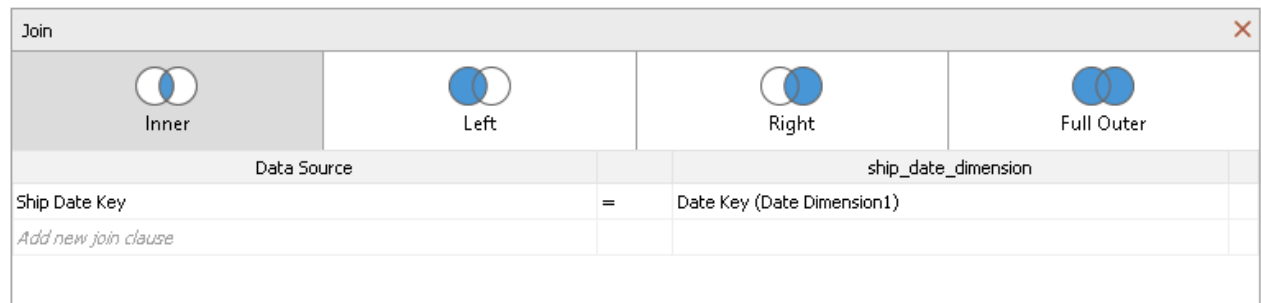
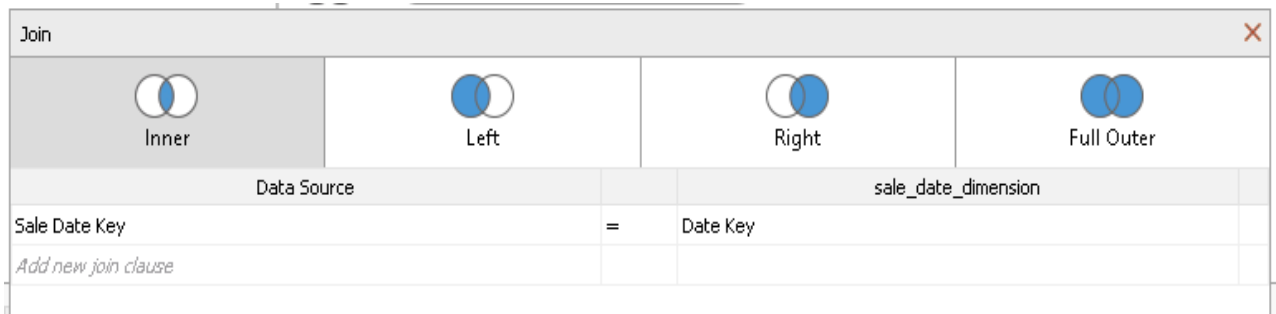
- airline2000
- airline2000r10000
- b
- basic_vertica_datatypes_pdi
- bool_table_pdi
- bool_Table_pdi_bulk

online_sales_fact

- call_center_dimension
- customer_dimension
- online_page_dimension
- product_dimension
- promotion_dimension
- sale_date_dimension
- ship_date_dimension
- shipping_dimension

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau



Cross-Database Joins

Cross-database joins allow you to join tables from multiple databases such as multiple Vertica databases.

To construct a cross-database join, you need to create a data source with multiple connections, one connection for each database in your join. Follow these steps:

1. Connect to one of your data sources.
2. On the Data Source tab, click the Add link for each database that your join needs to access.

After you have added all necessary connections, Tableau lists them on the left-hand side of the screen, each connection identified by a different color.

3. Drag and drop the tables from each connection into the canvas to create the join.
4. Check that the relationships that Tableau has created are correct. Edit them if necessary.

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau

It is important to understand that Tableau does not compute the join and aggregations in Vertica. Tableau issues separate queries to each database. Then Tableau transfers the data to the client and computes the joins and aggregations within the Tableau data engine.

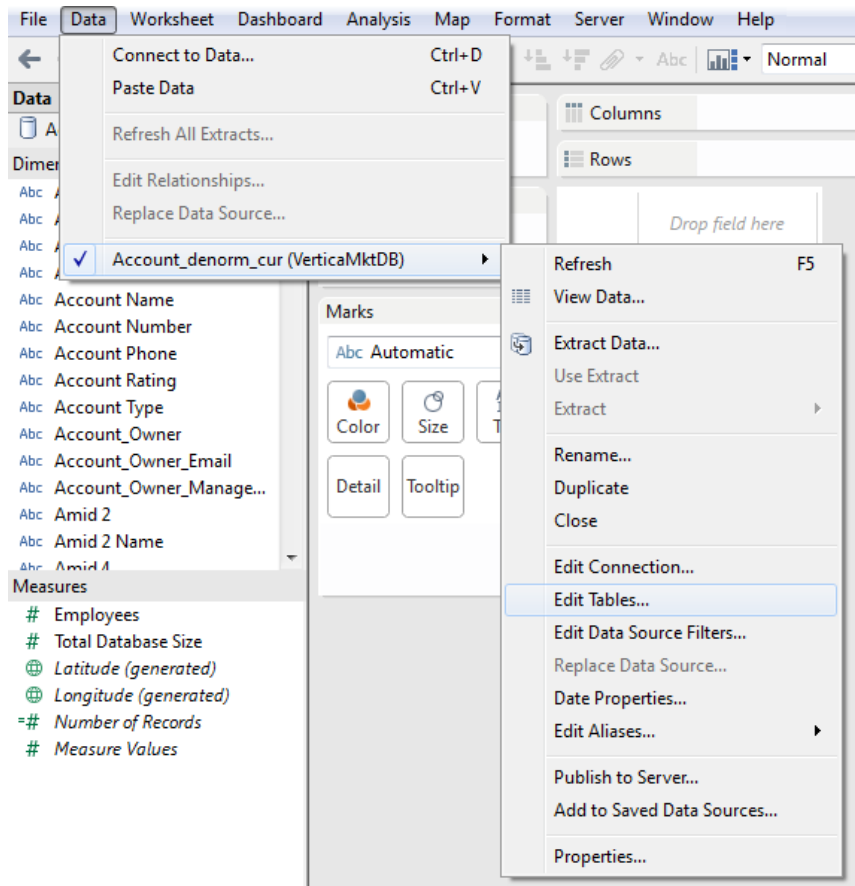
Exporting the data out of the data sources and computing the query in the Tableau data engine is not as efficient as performing the queries at each data source. The performance of your dashboards depends on the number of databases that are joined and the amount of data transferred.

For more information, see [Combine Data with Cross-Database Joins](#) in the Tableau documentation.

Adding or Excluding Columns: Tableau 8

Tableau Desktop 8.1

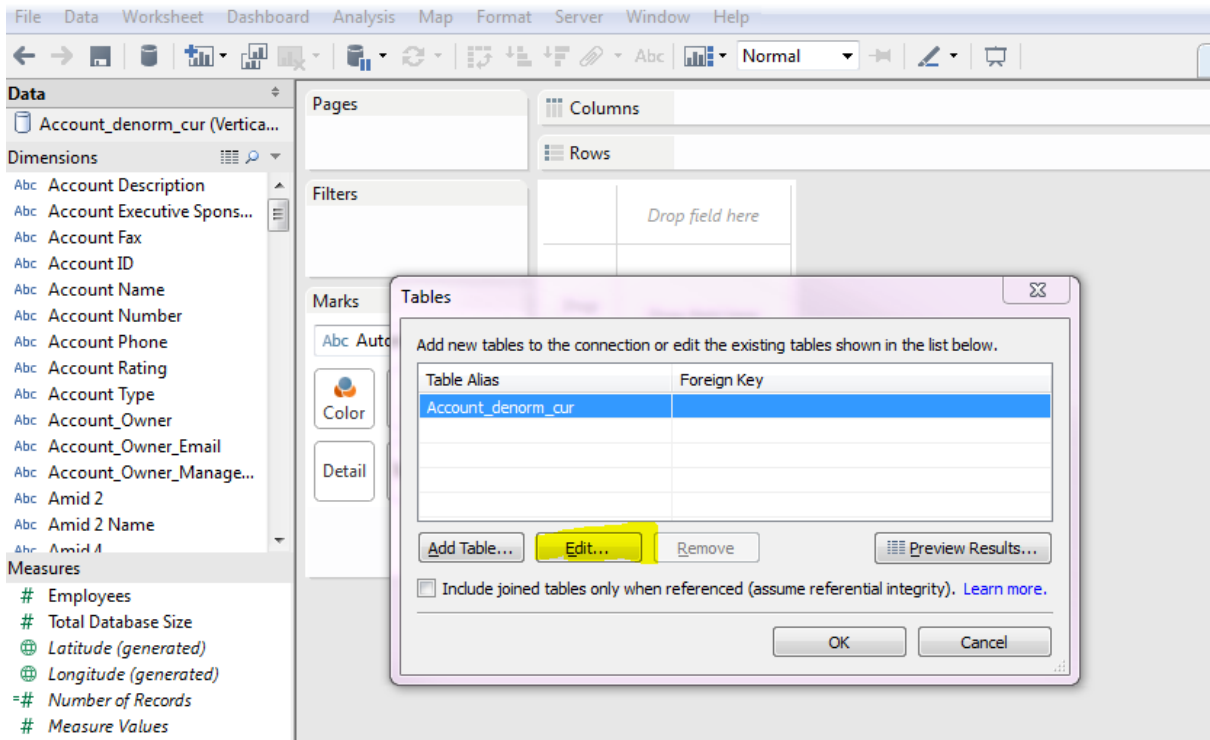
1. Select **Data**, choose your data source, and click **Edit Tables**.



2. Select the table you want to edit and then click **Edit**.

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau



3. Deselect fields you want to exclude from your workbook. Click **OK**.

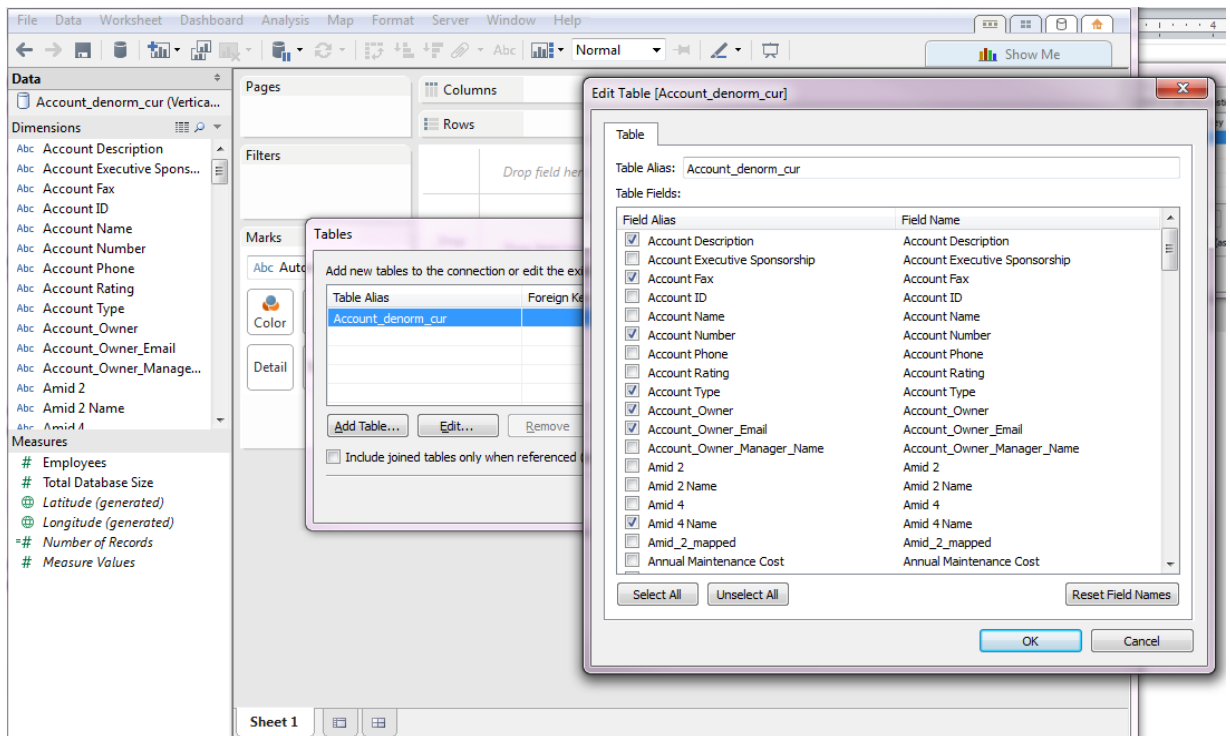
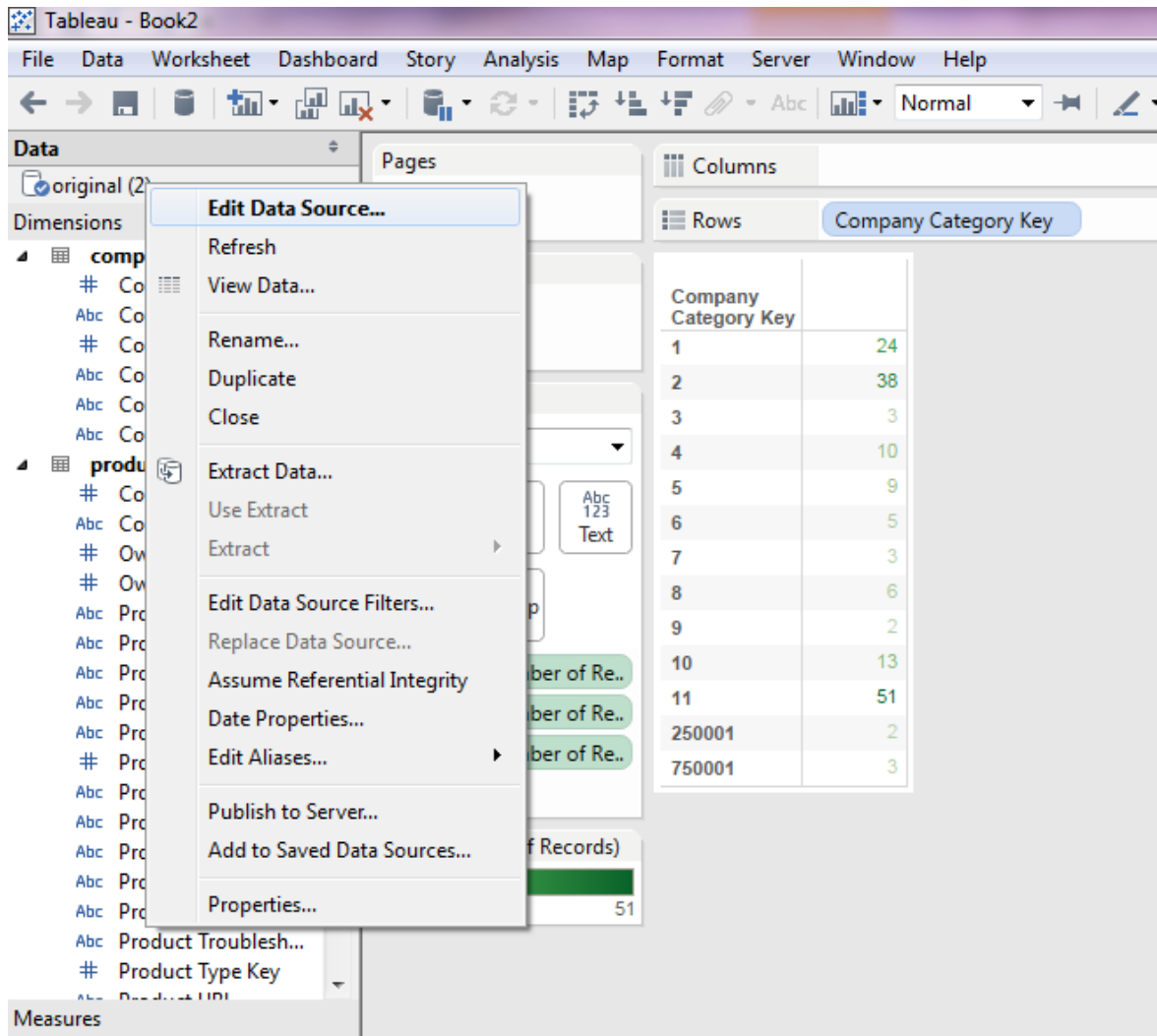


Tableau Desktop 8.2

This procedure includes both excluding a field, or including a field that had previously been excluded.

1. Right-click your data source and select **Edit Data Source**.



2. From the table, click the specific field you want to exclude. Select **Hide**.

Vertica Integration with Tableau 10: Tips and Techniques

Tips for Connecting to HPE Vertica from Tableau



Copy Go to Worksheet

Company key (company)	Company Category Key	Company Short Name	Company Long Name	Company Comment	Company
#	#	Abc	company	Abc	company

- To add a field back that has been excluded, select Show Hidden Fields. Click the specific field you want to add and select **Unhide**.



Copy Go to Worksheet Show hidden fields

Company key (company)	Company Category Key	Company Market Score	Company Strategic Score	Company Partnership Score	Company Short Name	Company Long Name	Company Comment	Company
#	#	#	#	#	company	Abc	Abc	company

Tableau Best Practices

The following sections describe techniques you can use to improve performance when connecting to Vertica from Tableau.

Tips for Dashboard Design

Consider the following recommendations for Tableau dashboard design:

- An efficient dashboard design combines action filters, data source filters, and parameters. The following sections contain more details about these techniques.

Note: If the data changes in your Vertica database, Tableau does not dynamically repopulate the parameters. You need to add new parameter values manually. In addition, Tableau supports only single-value selection for parameters.

For more information about parameters, see [QuickStart: Parameters](#).

- Avoid using quick filters, or use them in moderation. If you use quick filters, use text boxes instead of drop-down menus with long lists of values. If you use quick filters with long lists of values, use the Apply button as described in [Filtering Dimensions](#) in this document.

For more information, see [Quick Filters](#) in this document. For information about filter modes, see [Quick Start: Filter Modes](#) in the Tableau online help.

- As an alternative to quick filters, use action filters. Action filters allow you to use visualizations to filter other visualizations. When you use action filters, Tableau sends fewer queries to Vertica. For more information, see [Quick Start: Filter Actions](#) in the Tableau online help.

- Avoid using too many text tables (also called cross-tabs or Pivot Tables). If you use text tables, make sure they are small.
- If a calculation is complicated in Tableau, aggregate it in Vertica, for example, by using live aggregate projections, as described in [Use Live Aggregate Projections](#). Push down the execution of these calculations to the database, and let Tableau create the visualization.
- After you have finished designing your dashboards, exclude all unused columns from your data sources. Doing so allows your dashboards to load faster when you connect to Vertica. For information about excluding unnecessary columns, see [Add or Exclude Columns](#) in this document.

Tips for Using Parameters

Using parameters for conditional calculations in Tableau allows you to dynamically change a calculation. The following example shows three ways of using Tableau parameters, from slowest to fastest:

Slowest response time:

```
IF [Parameters].[Date Part Picker] = "Year"
THEN DATEPART('year',[Order Date])
ELSEIF [Parameters].[Date Part Picker] = "Quarter"
THEN DATEPART('quarter',[Date])
...
ELSE NULL END
```

Faster calculation:

```
IF [Parameters].[Date Part Picker] = 1
THEN DATEPART('year',[Order Date])
ELSEIF [Parameters].[Date Part Picker] = 2
THEN DATEPART('quarter',[Date])
...
ELSE NULL END
```

Fastest calculation:

```
DATEPART([Parameters].[Date Part Picker],[Order Date])
```

For more information, see [QuickStart: Parameters](#) and [Using Parameters with Filters](#) in the Tableau documentation.

Tips for Using Filters

Filters allow you to narrow the data displayed on your dashboard.

Follow these recommendations for using filters to include and exclude Vertica data from Tableau visualizations:

- [Quick Filters](#)
- [Filter Actions and Data Source Filters](#)
- [Context Filters](#)

Quick Filters

Consider these recommendations for using quick filters:

- Avoid filtering data using the **Exclude** option. The **Exclude** option prevents Tableau from leveraging Vertica data encoding and compression and forces Tableau to scan all the selected data
- Instead of utilizing drop-down or multi-select lists for your quick filter, use non-enumerated quick filters like the wildcard filter. With these filters, Tableau does not query each member of the dimension, resulting in improved response time.
- When filtering dates, use relative date filters or range date filters. Avoid using discrete date filters where possible.
- To avoid the extra queries associated with quick filters, use filter actions with the **Exclude all values** option checked, as shown in the following graphic. Doing so reduces the number of queries that are pushed down to Vertica.

For detailed recommendations for using quick filters, see the following sections:

- [Quick Filters that Do Not Require Extra Queries](#)
- [Quick Filters that Require Extra Queries](#)
- [Filtering Ranges of Values](#)
- [Filtering Discrete Values](#)
- [Filtering Dimensions](#)
- [Filtering Dates](#)
- [Filtering Measures](#)

Quick Filters that Do Not Require Extra Queries

Hewlett Packard Enterprise recommends that you avoid quick filters that require knowledge of the values listed in the filter. Tableau has to query Vertica for all potential values before the quick filter object can be rendered.

Quick filters that do not require knowledge of the values in the filter list include:

- Custom value list
- Wildcard match
- Relative date filters
- Browse period date filters

For more information, see:

- [Set Options for Filters in the View](#)
- [Filter Dimensions](#)

Quick Filters that Require Extra Queries

The following quick filters require knowledge of the values listed in the filter:

- Multiple value list
- Single value list
- Compact List
- Slider
- Measure filters
- Ranged date filters

Filtering Ranges of Values

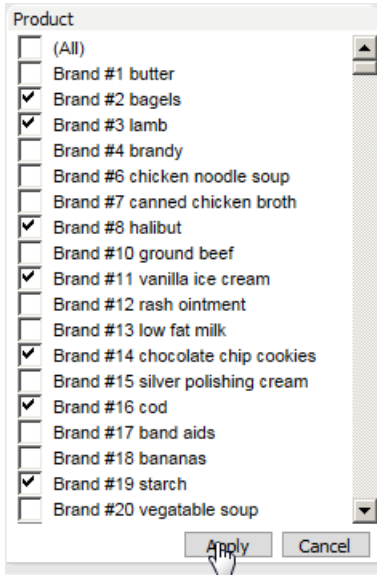
If your data has numeric dimensions or you define a condition to filter based on numeric values, filter the data using a range of values. Tableau can process a range of values faster than it can process a list of discrete values. When filtering a range of values, use the **Keep Only** or **Exclude** options, as described in [Select Data to Filter](#).

For examples of filtering a dimension with a range of values, see [Adding Conditions to Filters](#).

Filtering Discrete Values

Hewlett Packard Enterprise recommends that you use the **Apply** button when filtering data using multiple discrete values. Doing so prevents Tableau from issuing a query each time the user selects an item on the quick filter.

When using basic filters that include multiple discrete values, customize your filter and select or deselect multiple values. When you click **Apply**, Tableau issues a single query to Vertica that retrieves data based on all the selected values.



Filtering Dimensions

If you are using quick filters to filter a dimension, Hewlett-Packard recommends that you use quick filters that do not require an extra query. For more information, see [Quick Filters that Do Not Require Extra Queries](#).

If you need to use quick filters and they have large list of discrete values, Hewlett Packard Enterprise recommends using the **Apply** button, as described in [Filtering Discrete Values](#).

When possible, filter by a range of values. These filters are often faster to evaluate than discrete values. In addition, use filters by range instead of the **Keep Only** or **Exclude** filtering options. For examples of **Keep Only** and **Exclude** options, see [Select Data to Filter](#).

Filters by range apply only to numeric dimensions or when you define a condition to filter by based on a range. For examples of filtering a dimension by range, see [Adding Conditions to Filters](#).

For more information, see [Set Options for Filters in the View](#).

Filtering Dates

Date filters are a special kind of dimension filter. The method you use to filter dates impacts the efficiency of the resulting query:

- Relative date filters—Show a date range that is relative to a specific day.
- Range of date filters—Show a defined range of discrete dates.
- Discrete date filters—Show individual dates that you have selected from a list.

An efficient date filter is one that generates queries with a resulting WHERE clause that uses a ranged date filter.

Where possible, use relative or range filters, and avoid discrete filters when designing a dashboard. Filtering by date range leverages Vertica partitioning and compression capabilities.

For more information, see [Filter Dates](#).

Filtering Measures

When filtering by measures, for best performance, filter the data using a range of values instead of using multiple discrete values.

If you have a large data source, Tableau recommends that you create a set that contains the measure. You can then apply one or more filters to that set.

For more information, see

- [Filter Measures](#)
- [Quantitative Quick Filter Options](#)

Filter Actions and Data Source Filters

To avoid extra queries associated with quick filters, replace quick filters by more efficient filters like the following. Two alternatives to quick filters are described in the Tableau documentation:

- [Filter actions](#): Using charts to show related information between Tableau worksheets.
- [Data source filters](#): Filtering the data on the data source at connection time.

Context Filters

By default, all filters that you create in Tableau are computed independently and access all rows in your Vertica data source, without regard to other filters.

Context filters behave differently. Tableau implements context filters by writing the filter result set to a temporary table. Any other filters process only the data that the context filter filters and stores in the temporary table.

A context filter must meet the following conditions to ensure optimal performance:

- The context filter must reduce the size of the data set significantly, by filtering 90% of the data.
- The context filter should be used against slow changing dimensions. If you change the filter, the database must recompute and rewrite the temporary table.

For more information about context filters, see:

- [Context Filters](#)
- [Speeding up Context Filters](#)

Cross-Data Source Filters

Cross-data source filters allow you to filter data from multiple data sources at once based on a common dimension. By default, any fields that have the same name and data type across the data sources are identified as common or related.

To use cross-data source joins, take these steps:

1. Create a dashboard with visualizations from your data sources.
2. Open one of the worksheets.
3. Drag and drop the common fields to the Filters shelf to create the filter.
4. Bring the filter to your dashboard.
5. Click the arrow in the upper-right corner of the filter.
6. In the menu that appears, select **Apply to Worksheets > All Using Related Data Sources**.

This action applies this filter across all related data sources in your dashboard.

Before filtering the visualizations, Tableau issues separate queries to each data source to identify the values in the filter. If those values don't exist in all related data sources, Tableau excludes them from the list of values in the filter.

For more information, see [Filter Data Across Multiple Data Sources](#) in the Tableau 10 documentation.

Tips for Calculations

There are a number of ways perform calculations and use Vertica functions in Tableau. Here are some tips for:

- [Tableau Built-In Functions](#)
- [Level of Detail \(LOD\) Expressions](#)

Tableau Built-In Functions

In the **Create Calculated Field** interface, functions that display in bold always execute locally, in Tableau, on the aggregated results. Vertica does not execute these functions. For example:

- ASCII
- CHAR
- FIRST
- INDEX
- ISDATE

Any function that does not display in bold executes in the Vertica database. Some examples are:

- ABS
- CONTAINS
- DATE
- DATEADD
- FIND
- IF
- IFNULL

- LEFT
- MAX
- MIN
- POWER
- SPACE
- STARTSWITH
- TRIM
- ZN

Regular Expression Functions

Tableau 9.3 adds support for Vertica regular expression functions using built-in functions in the Tableau user interface. The new Tableau built-in functions for Vertica are:

Tableau built-in function	Vertica function called by Tableau function	Comments
REGEXP_MATCH (<i>string</i> , <i>pattern</i>)	REGEXP_LIKE(<i>string</i> , <i>pattern</i>)	These two functions have the same behavior.
REGEXP_REPLACE (<i>string</i> ,	REGEXP_REPLACE (<i>string</i> ,	These two functions have the same behavior.

Tableau built-in function	Vertica function called by Tableau function	Comments
<i>pattern, replacement</i>)	<i>pattern, replacement</i>)	
REGEXP_EXTRACT (<i>string, pattern</i>)	REGEXP_SUBSTR (<i>string, pattern, 1, 1, "</i> , 1)	When calling REGEXP_SUBSTR, Tableau sets the last parameter, <i>captured_subexp</i> , to 1. Doing so causes REGEXP_SUBSTR to return the substring captured by the first set of parentheses in the regular expression.
REGEXP_EXTRACT_NTH(<i>string, pattern, index</i>)	REGEXP_SUBSTR (<i>string, pattern, 1, 1, "</i> , <i>index</i>)	You can specify the last parameter of REGEXP_SUBSTR, <i>captured_subexp</i> , using the parameter <i>index</i> in the Tableau function REGEXP_EXTRACT_NTH. To return the entire string that matches the regular expression, set <i>index</i> to 0.

For detailed information about regular expression functions in Tableau and Vertica, see

- [Regular expression functions in the Tableau online help](#)
- [Regular expression functions in the Vertica documentation](#)

How Regular Expression Functions Work

When you call a Tableau built-in function, Tableau pushes the corresponding Vertica function down to Vertica. Vertica executes the function and returns the results to Tableau. Tableau displays the results in the dashboard as you specify.

Let's look at an example. Suppose you want the portion of a string that matches the first set of parentheses in the regular expression. When you use `REGEXP_EXTRACT` in your dashboard, it calls `REGEXP_SUBSTR`, with the last parameter, *captured_subexp*, set to 1. The `REGEXP_EXTRACT` function looks like this:

```
REGEXP_EXTRACT('abc 123', '[a-z]+\s+(\d+)') = '123' -- Tableau function
```

`REGEXP_EXTRACT` generates a query like the following and sends it to Vertica for execution. You cannot change this query:

```
SELECT REGEXP_SUBSTR('abc 123', '[a-z]+\s+(\d+)', 1, 1, '', 1)
```

Now suppose you want the full string that matches the regular expression. For this, call the Tableau function `REGEX_EXTRACT_NTH` with the index parameter set to 0:

```
REGEXP_EXTRACT_NTH('abc 123', '[a-z]+\s+(\d+)', 0) = 'abc 123' --function in Tableau
```

The Tableau function `REGEX_EXTRACT_NTH`, with the index parameter set to 0, generates a query that calls the Vertica function `REGEXP_SUBSTR` with *captured_subexp parameters* set to 0. This query returns the full string that matches the regular expression:

```
SELECT REGEXP_SUBSTR('abc 123', '[a-z]+\s+(\d+)', 1,1, '', 0) = 'abc 123' --function in Vertica
```

For more details, see [REGEXP](#) in the Vertica documentation.

Table Calculations

Table calculations execute only in Tableau. Some examples are:

- `RUNNING_SUM`
- `WINDOW`

Hewlett Packard Enterprise recommends that, when possible, you consider using level of detail (LOD) calculations that can achieve the same results.

LOD calculations execute in the underlying database, which allows them to take advantage of Vertica's speed.

Pass-Through Functions

Tableau does not support all database functions that Vertica provides. If a function is not available in Tableau, use pass-through functions to push the functions to Vertica.

For example, Tableau does not have a MOD or MEDIAN function; use RAWSQLAGG_REAL to compute MOD or MEDIAN at the database level. Use RAWSQLAGG_REAL to push down Vertica functions that return real values, like, for example, MOD or MEDIAN.

For a complete list of Vertica functions, see [SQL Functions](#).

For a complete list of Tableau pass-through functions, see [Pass-Through Functions \(RAWSQL\)](#).

Pass-Through Function Example

To use a Tableau pass-through function to run APPROXIMATE COUNT DISTINCT, create a calculated field:

1. Right-click a field and select **Create Calculated Field**.
2. Select **Pass Through** from the Functions drop-down list.
3. Select RAWSQLAGG_REAL.

Enable Non-DBADMIN Users for Pass-Through Functions

To enable Vertica non-DBADMIN users to access the pass-through functionality, add the public schema to the user's search path:

```
=> ALTER USER tableau_user SEARCH_PATH public, tableau_user_s;
```

Sets

Sets in Tableau are useful when you want to compare two or more data sets.

However, sets do not push query execution down to Vertica, so use sets only when working with small data sets.

For more information, see [Sets](#) in the Tableau documentation.

Vertica Tuning Recommendations

Consider the following techniques for improving Vertica performance when connected from Tableau:

- [Upgrade Vertica for More Efficient Query Processing](#)
- [Create a Physical Design with Database Designer](#)
- [Use Live Aggregate Projections](#)
- [Tips for Managing Vertica Resources](#)
- [Enabling Database Isolation Levels in Tableau](#)
- [Enabling Native Connection Load Balancing from Tableau](#)

Upgrade Vertica for More Efficient Query Processing

Tableau issues system-related queries to obtain metadata on Vertica tables. The queries use a `WHERE 1=0` clause. Vertica 7.1 and later optimize such queries. For more efficient query processing from Tableau to Vertica, upgrade Vertica to Release 7.1.0 or later.

Create a Physical Design with Database Designer

To get the best performance from your Vertica database, create a physical design for your database that optimizes both query performance and data compression.

The Vertica Database Designer automatically optimizes your physical design in the following ways:

- Analyzes your logical schema, sample data, and your sample queries.
- Creates a physical schema design (projections) that can be deployed automatically or manually.

- Can be run any time for additional optimization without stopping the database.
- Uses strategies to provide optimal query performance and data compression.

Database Designer minimizes the time you spend on manual database tuning and provides the ability to redesign the database incrementally to optimize for changing workloads over time.

For more information, see [Workflow for Running Database Designer](#).

Identify Sample Queries for Database Designer

You can submit sample queries to Database Designer before it creates or updates a physical design. In particular, if some queries are executing slowly, Database Designer can create a physical design that optimizes those slow queries

You can use Tableau to identify queries that are good candidates for the Database Designer in a number of ways:

- Use Tableau Performance Recorder. For information, refer to [Performance Recorder](#)
- Examine the Tableau log. Refer to [Log Files](#)
- Examine the Vertica log. Refer to [Log Files](#)
- Run SQL queries against the Vertica Data Collector tables.

The following query returns the 10 slowest queries from the system:

```
=> SELECT request FROM v_monitor.query_requests  
WHERE request_type = 'QUERY' ORDER BY request_duration_ms  
DESC LIMIT 10;
```

The following query returns the 10 most frequently executed queries:

```
=> SELECT request FROM (SELECT request, COUNT(request) cnt  
FROM v_monitor.query_requests WHERE request_type = 'QUERY'  
GROUP BY request) T ORDER BY cnt DESC LIMIT 10;
```

- Monitor queries in Management Console to better understand resource utilization when queries are executing.

For more information:

- [Monitoring and Configuring Resource Pools in MC](#)
- [Design Queries](#)
- [Best Practices: Queries for Database Designer](#) (video)

Check Database Designer Projections

After Database Designer creates a database design, examine the log files and review the projections. You can find data about the projections that Database Designer considered and deployed in two Data Collector tables:

- DC_DESIGN_PROJECTION_CANDIDATES
- DC_DESIGN_QUERY_PROJECTION_CANDIDATES

Verify that the projections are sorted on columns of importance according to the WHERE clauses of your sample queries. Low-cardinality columns should be encoded.

Use Live Aggregate Projections

When you create a live aggregate projection for a table, Vertica automatically aggregates data from that anchor table and loads it into the live aggregate projection. Because the data is already aggregated, retrieving the data directly from the live aggregate projection is faster than retrieving it from the anchor table .

Note: Once you create a projection, run the [START_REFRESH](#) or [REFRESH](#) function to load the data into the projection.

To access a live aggregate projection from Tableau, create a view and access the view from Tableau:

```
=> CREATE VIEW <projection_name> AS SELECT * FROM <projection_name>
```

For information on creating live aggregate projections, see:

- [CREATE PROJECTIONS \(Live Aggregate Projections\)](#)
- [Live Aggregate Projections](#)

Tips for Managing Vertica Resources

When connecting to Vertica from Tableau, follow these tips about managing database resources:

- [Create a Separate Resource Pool for Tableau](#)
- [Create Cascading Resource Pools](#)

Create a Separate Resource Pool for Tableau

Hewlett Packard Enterprise recommends that you create a separate resource pool and user for your Tableau application. Doing so reduces the impact of ETL jobs on query users.

The actual impact depends on the amount of memory on the machine and other factors such as how many other resource pools are created for other users.

The following example show how to create and manage access to the TABLEAU_POOL resource pool:

```
=> CREATE RESOURCE POOL TABLEAU_POOL MEMORYSIZE '4G' MAXMEMORYSIZE '84G' MAXCONCURRENCY 36;  
=> DROP USER tableau;  
=> DROP SCHEMA tableau_s;  
=> CREATE SCHEMA tableau_s;
```

```
=> CREATE USER tableau IDENTIFIED BY 'my_password' SEARCH_PATH tableau_s;  
=> GRANT USAGE ON SCHEMA tableau_s TO tableau;  
=> GRANT USAGE ON SCHEMA PUBLIC TO tableau;  
=> GRANT USAGE ON SCHEMA online_sales TO tableau;  
=> GRANT USAGE ON SCHEMA store TO tableau;  
=> GRANT SELECT ON ALL TABLES IN SCHEMA PUBLIC TO tableau;  
=> GRANT SELECT ON ALL TABLES IN SCHEMA store TO tableau;  
=> GRANT SELECT ON ALL TABLES IN SCHEMA online_sales TO tableau;  
=> GRANT ALL PRIVILEGES ON SCHEMA tableau_s TO tableau WITH GRANT OPTION;  
=> GRANT CREATE ON SCHEMA tableau_s TO tableau;  
=> GRANT USAGE ON RESOURCE POOL TABLEAU_POOL TO tableau;  
=> ALTER USER tableau RESOURCE POOL TABLEAU_POOL;
```

For more information, see

- [Best Practices: Resource Management](#) (video)
- [The Resource Manager](#) in the Vertica documentation

Create Cascading Resource Pools

When you create a resource pool, you assign a `RUNTIMECAP` that specifies the amount of time a query should run in that resource pool before it times out. If a query exceeds the `RUNTIMECAP`, it errors out *unless* you have assigned a secondary pool.

If you have defined a secondary pool, if the query exceed the `RUNTIMECAP` in the first resource pool, it continues executing in the secondary pool.

To designate a secondary pool, use the `CASCADE TO` parameter in the `ALTER RESOURCE POOL` or `CREATE RESOURCE POOL` statement.

For example, suppose you have a resource pool assigned to the Tableau user:

```
=> ALTER USER tableau RESOURCE POOL TABLEAU_POOL;
```

If the `TABLEAU_POOL` resource pool is not adequate, create a second resource pool (`TABLEAU_POOL_BACKUP`) for overflow query execution:

```
=> CREATE RESOURCE POOL TABLEAU_POOL_BACKUP RUNTIMECAP '5 minutes';  
=> ALTER RESOURCE POOL TABLEAU_POOL CASCADE TO TABLEAU_POOL_BACKUP;
```

For more information about cascading resource pools, see [Defining Secondary Resource Pools](#) in the Administrator's Guide.

Enabling Database Isolation Levels in Tableau

By default, Vertica uses the READ COMMITTED isolation level for every session. If your database has a different isolation level than READ COMMITTED, you might experience table locks.

To change the default isolation level for the database, set the TransactionIsolation configuration parameter in either the TDC or TDS file. The following examples show how to set the isolation level to READ COMMITTED:

```
<!-- METHOD 1 -->
<customization name='CAP_SET_ISOLATION_LEVEL_VIA_ODBC_API' value='yes' />
<customization name='CAP_ISOLATION_LEVEL_READ_COMMITTED' value='yes' />

<!-- METHOD 2 -->
<customization name='odbc-connect-string-extras' value='TransactionIsolation=Read
Committed;' />
```

To change the default isolation level for a specific session in vsql, use SET SESSION CHARACTERISTICS command:

```
=> SET SESSION CHARACTERISTICS AS TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

For details, see [Customize Your Connection to Vertica](#) in this document.

For information about Vertica isolation levels, see [Change Transaction Isolation Levels](#) in the product documentation.

Enabling Native Connection Load Balancing from Tableau

Native connection load balancing is a Vertica feature that spreads the CPU and memory overhead caused by client connections across the hosts in the database.

By default, Vertica does not use native connection load balancing.

You must enable native connection load balancing before making the changes described in [Native Connection Load Balancing](#) in Best Practices for OEM Customers.

Vertica Integration with Tableau 10: Tips and Techniques

Vertica Tuning Recommendations

Once you have enabled load balancing, make the following change to the Tableau Datasource Connection (TDC) file:

```
<customization name='odbc-connect-string-extras' value='ConnectionLoadBalance=1' />
```

For details, see [Customize Your Connection to Vertica](#) in this document.

For more information on native connection load balancing, see [Best Practices: Native Connection Load Balancing](#) (video).

Customize Your Connection to Vertica

If you need to investigate issues with your connection to Vertica, Tableau allows you to customize it by creating or modifying a Tableau Datasource Connection (TDC) file or a Tableau Data Source (TDS) file.

In Tableau Desktop 9 and later, if you have created a custom TDC file, Tableau automatically embeds the content of the *.tdc file into the workbook (*.twb).

Once you create these files, you must maintain them manually, editing them any time you need to change them. Because of this, you should use these files judiciously, relying on them only when you detect connection issues that you want to investigate further.

A TDC file applies customizations to all data sources; a TDS file allows you to customize individual data sources. Hewlett Packard Enterprise recommends using TDC files only. Try to avoid customizing connections for individual data sources using TDS files.

For more information, see:

[Making customizations global](#)

[Customizing and Tuning ODBC Connections.](#)

Important: TDS settings override global TDC settings. Use TDS settings only if you want to apply your settings to one specific connection.

TDC and TDS File Locations

Tableau saves TDC and TDS files in the following locations.

Tableau Desktop:

```
C:\Users\myuser\Documents\My Tableau Repository\Datasources
```

Tableau Server:

```
C:\ProgramData\Tableau\Tableau Server\data\tabsvc\server\Datasources
```


Customization Details

Based on your situation, apply the following modifications in a TDC or TDS file.

The Tableau customizations are as follows.

Customization	Recommended Setting
CAP_CREATE_TEMP_TABLES	Set to 'no' so that Tableau does not create temporary tables. See also: CAP_SELECT_INTRO.
CAP_ISOLATION_LEVEL_READ_COMMITTED	Set to 'yes' to force the transaction isolation level to Read Committed. Apply only if you are experiencing lock issues. Refer to Enabling Database Isolation Levels in Tableau .
CAP_ODBC_EXPORT_ALLOW_CHAR_UTF8	Set to 'yes' to allow the single-byte char data type for binding Unicode strings as UTF-8. This setting does not impact performance with Vertica.
CAP_ODBC_METADATA_SUPPRESS_EXECUTED_QUERY	Set to 'yes' for Vertica versions 7.0 and prior releases. Refer to Upgrade Vertica for More Efficient Query Processing .
CAP_QUERY_SUBQUERY_QUERY_CONTEXT	Set to 'yes' to force Tableau to use a subquery for context filters instead of a temporary table or locally cached results. Hewlett Packard Enterprise recommends this setting if you are using context filters. Example: <pre><customization name='CAP_QUERY_SUBQUERY_QUERY_CONTEXT' value='yes' /></pre>

Vertica Integration with Tableau 10: Tips and Techniques

Customize Your Connection to Vertica

CAP_SELECT_INTO	<p>Set to 'no' so that Tableau does not create a table on the fly from the result set of another query. See also: CAP_CREATE_TEMP_TABLES.</p> <p>Example:</p> <pre><customization name='CAP_SELECT_INTO ' value='no' /></pre>
odbc-connect-string-extras	<p>Use this customization. See the following examples.</p> <p>Note: This customization is case sensitive. Make sure that odbc-connect-string-extras is always lowercase.</p>

Define a session label:

```
<customization name='odbc-connect-string-extras'  
value='Label=tableau_query_session' />
```

Change the buffer size:

```
<customization name='odbc-connect-string-extras'  
value='ResultBufferSize=500000' />
```

Enable load balancing:

```
<customization name='odbc-connect-string-extras'  
value='ConnectionLoadBalance=1' />
```

Specify database isolation:

```
<customization name='odbc-connect-string-extras'  
value='TransactionIsolation=Read Committed' />
```

If you want to change multiple parameters, you must put them in one line, as in this example:

```
<connection-customization class='vertica' enabled='true'
```

```
version='9.0'>
<vendor name='vertica' />
<driver name='vertica' />
<customizations>
  <customization name='odbc-connect-string-extras'
    value='Label=tableau_session_label;ConnectionLoadBalance=1' />
</customizations>
</connection-customization>
```

Note: The preceding example specifies Tableau 9.0. Verify that the version you specify matches the version of Tableau that you are using.

Hewlett Packard Enterprise has tested the customizations in this table with Tableau 9. If you are using later Tableau versions, you may not need to apply these customizations.

For a complete list of customizations, see [Tableau Capability Customizations](#).

Global Data Source Customizations

To modify the connection behavior for all data sources connecting to Vertica, create a stand-alone Tableau Datasource Connection (TDC) file. The TDC file contains only a <connection-customization> section that Tableau applies to any Tableau connection to Vertica.

In Tableau Desktop 9 and later, if you have created a custom TDC file, Tableau automatically embeds the content of the *.tdc file into the workbook (*.twb).

Important: TDS settings override global TDC settings. Use TDS settings only if you want to apply your settings to one specific connection.

Create a TDC File

To create a TDC file, follow these steps:

1. Create a new file in the `Datasources` directory of the Tableau repository (**My Tableau Repository**), and name the file using the `.tdc` suffix (for example, `vertica.tdc`).

2. Copy and paste the following into your new TDC file.

```
<connection-customization class='vertica' enabled='true'  
  version='9.0'>  
  <vendor name='vertica' />  
  <driver name='vertica' />  
  <customizations>  
    <customization name='odbc-connect-string-extras'  
      value='Label=tableau_session_label;ConnectionLoadBalance=1' />  
  </customizations>  
</connection-customization>
```

This example sets `CAP_QUERY_SUBQUERY_QUERY_CONTEXT` to yes. This setting forces Tableau to use a subquery for context filters instead of a temporary table or locally cached results. Vertica strongly recommends this setting if you use or plan to use context filters in your workbooks.

This example also sets a session label for the connection ('odbc-connect-string-extras') and enables connection load balancing.

Important: The Tableau version mentioned within the file must be the same as the Tableau Desktop version you are using.

3. Save and close your `.tdc` file.

Important: Embedded Custom TDC Files

If you have created a custom `*.tdc` file, Tableau Desktop 9.2 automatically embeds the content of the `*.tdc` file in the workbook file (`*.twb`).

When you publish the Tableau workbook to Tableau Server or Tableau Online, the workbook use the customizations that you specified in your custom TDC file. So, as of Tableau Desktop 9, you no longer have to manually create a `*.tdc` file in Tableau Server.

Vertica Integration with Tableau 10: Tips and Techniques

Customize Your Connection to Vertica

To overwrite any customized settings, create a new TDC file. However, if you delete this new TDC file, Tableau Desktop uses the settings in the original custom TDC file, even if that file no longer exists.

To remove all customizations, you must manually edit the *.twb file:

1. Delete any custom *.tdc files.
2. Edit the *.twb file (it is an XML file) to reset the customizations to the desired settings.

Check the TDC File

To verify that the customizations in the TDC file have been applied to your connection, take these steps:

1. Delete all log files in: My Tableau Repository\Logs.
2. Open Tableau Desktop and connect to any table in Vertica. Open an existing workbook or create a new one.
3. Close Tableau.
4. Open your Tableau log file.
5. Search for the name of the customization you applied to your connection. For example, search for CAP_QUERY_SUBQUERY_QUERY_CONTEXT.

Single Data Source Customizations

Important: Only use a single data source customization when it is not necessary to have modifications applied to all Vertica data source connections.

To modify connection behavior for a specific data source connection:

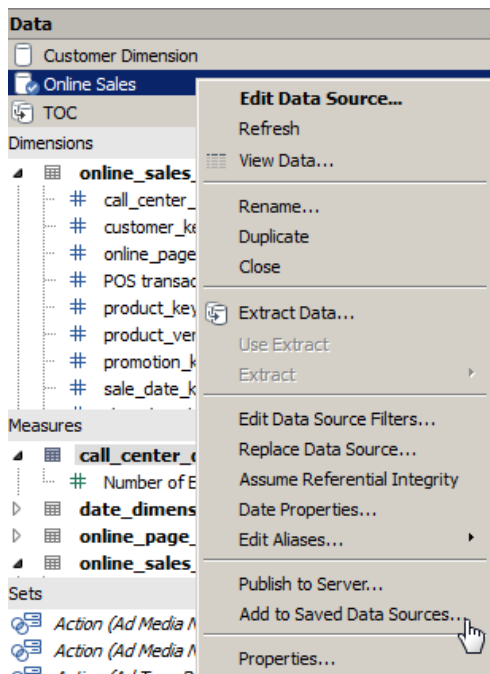
- Generate a Tableau Data Source (TDS) file.
- Edit the .tds file to add a <connection-customization> section for your specific use case.

Important: TDS settings override global TDC settings. Use TDS settings only if you want to apply your settings to one specific connection.

Create a TDS File

To create a TDS file, take these steps:

1. Right-click the data connection and select **Add to Saved Data Sources**.

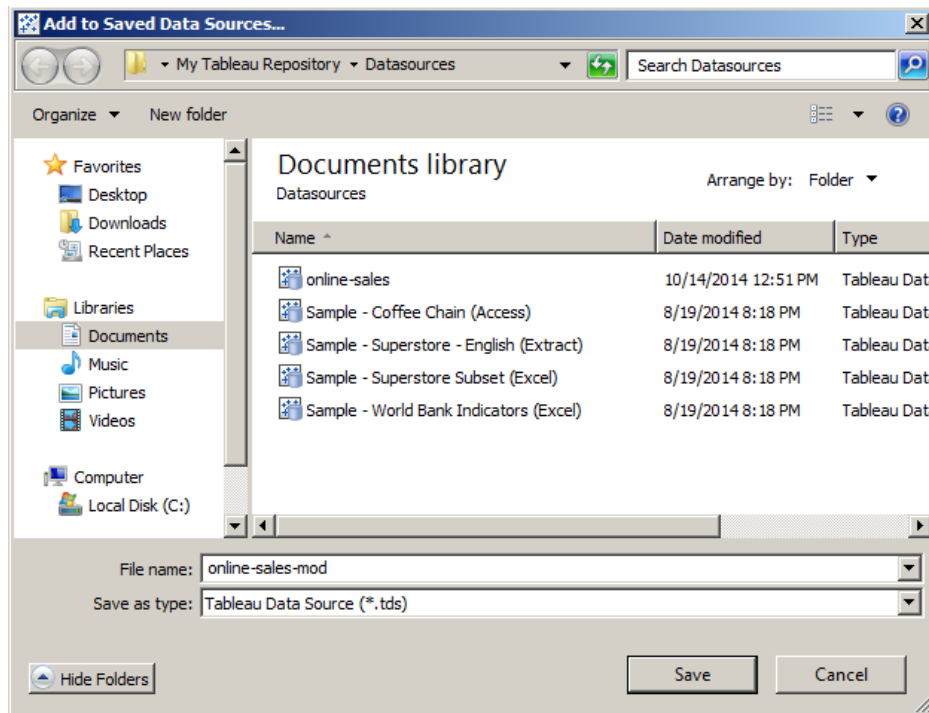


Alternatively, from the **Data** drop-down menu, select the data connection you want to customize and click **Add to Saved Data Sources**.

2. Enter a name and save the .tds file.

Vertica Integration with Tableau 10: Tips and Techniques

Customize Your Connection to Vertica



You can save the file in the Datasources directory of the Tableau repository, for example, My Tableau Repository\Datasources

3. Close the workbook.

Edit the TDS File

To edit the TDS file, take these steps:

1. Close your workbooks.
2. Open your saved .tds file in a text editor. The .tds file is an XML document that describes the connection to Vertica.
3. In the <connection> section, add a <connection-customization> section that contains the customizations for your specific scenario.

Important: The Tableau version mentioned within the TDC must be the same as the Tableau Desktop version you are using.

```
<connection-customization class='vertica' enabled='true'  
  version='9.0'>  
  <vendor name='vertica' />  
  <driver name='vertica' />  
  <customizations>  
    <customization name='odbc-connect-string-extras'  
      value='Label=tableau_session_label;ConnectionLoadBalance=1' />  
  </customizations>  
</connection-customization>
```

You can copy this example and add it to your `.tds` file. This modification has been tested with the Tableau Desktop and Vertica versions covered in this manual.

This example includes a session label for the connection ('odbc-connect-string-extras'). The session label is helpful for identifying the connection to facilitate troubleshooting.

4. Save and close your `.tds` file.

Apply the Modified TDS File

To apply your TDS file to a connection, take these steps:

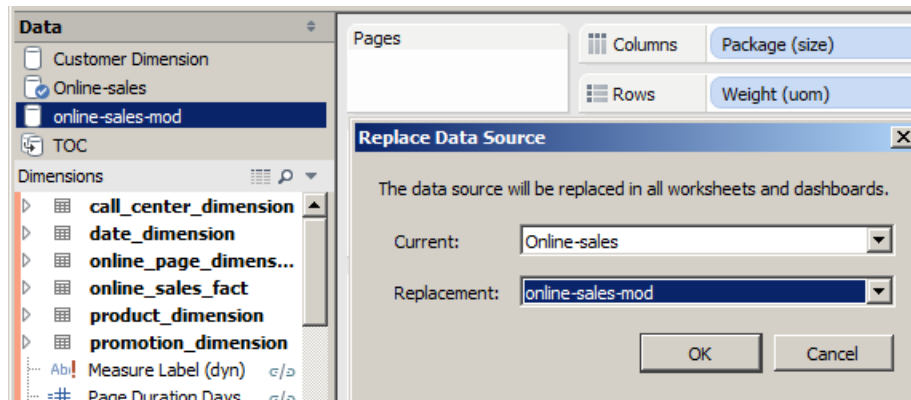
1. From your Tableau workbook, navigate to your `.tds` file.
2. Select the file and click **Open**.

Notice that the new data source is displayed under **Data**.

3. Right-click the original data source and select **Replace Data Source**.

Vertica Integration with Tableau 10: Tips and Techniques

Customize Your Connection to Vertica



4. From the **Replacement** list box, choose the name of your replacement data source.
5. Click **OK**.
6. Right-click the original data source, and select **Close**.

Check the TDS File

To verify that the customizations in the TDC file have been applied to your connection, take these steps:

1. Delete all log files in your logs directory (My Tableau Repository\Logs).
2. Open Tableau Desktop and connect to any table in Vertica. Open an existing workbook or create a new one.
3. Close Tableau.
4. Open your Tableau log file.
5. Search for the name of the customization you applied to your connection. For example, search for CAP_QUERY_SUBQUERY_QUERY_CONTEXT.

Troubleshooting Tools

The following topics describe tools you can use to identify the causes of poor performance of your Tableau workbooks:

- [Performance Recorder](#)
- [Session Labels](#)
- [Log Files](#)

Performance Recorder

The Tableau Performance Recorder helps you identify what events are affecting query performance.

If the bottleneck is query execution (retrieving the data), follow the recommendation in [Vertica Tuning Recommendations](#) in this document.

If the bottleneck is on creating the visualizations and the dashboard layout, follow the recommendation in [Tableau Best Practices](#) in this document.

Using Performance Recorder on Tableau Desktop

Use the Tableau Performance Recorder to view the performance of the SQL queries that Tableau pushes to Vertica. With this information, you can identify queries that are good candidates to submit to Database Designer. When you run Database Designer, it creates projections that optimize the performance of those sample queries.

For more information, see the Tableau documentation:

- To create a Tableau performance recording, see [Create a Performance Recording](#).
- To interpret results of a Tableau performance recording, see [Interpret a Performance Recording](#).

For information about submitting design queries to Database Designer, watch this video: [Best Practices: Queries for Database Designer](#).

Using Performance Recorder on Tableau Server

Run Performance Recorder first on your Tableau Desktop machine. If your workbook is slow in Tableau Desktop, it will be slow in Tableau Server. Before you run Performance Recorder on Tableau Server, restart Tableau Server to eliminate caching and inaccurate results. Then run Performance Recorder on Tableau Server.

Many factors can affect Tableau Server performance, including network latency. To identify if network latency is the problem:

1. Install Tableau Desktop on the Tableau server machine.
2. Run Performance Recorder using Tableau Desktop on the Tableau Server machine.
3. Compare the performance to the Performance Recorder results on the Tableau Desktop machine.

For information on how to run Performance Recorder in Tableau Server, see [Create a Performance Recording](#) in the Tableau online help.

Session Labels

When troubleshooting, you can identify a connection by configuring a Tableau session label.

To configure a Tableau session label, customize the connection attribute `odbc-connect-string-extras` in the TDC or TDS file. Once you have configured a session label, all sessions that Tableau creates will have this label.

To configure a session label in a TDC file, add this text:

```
<customization name='odbc-connect-string-extras' value='Label=tableau_query_session' />
```

To configure a session label in a TDS file, there are two ways to configure a session label:

- ```
<connection class='vertica' dbname='VMart' expected-driver-version='7.0'
odbc-connect-string-extras='Label=tableau_query_session' odbc-native-protocol='yes'
one-time-sql='' port='5433' schema='online_sales' server='part10' server-oauth=''
username='dbadmin' workgroup-auth-mode='as-is'>
```
- ```
<customization name='odbc-connect-string-extras' value='Label=tableau_query_session' />
```

For more information about tuning for ODBC, see [Customize Your Connection to Vertica](#) in this document.

Log Files

To monitor the performance of SQL queries that Tableau sends to Vertica, review the following log files:

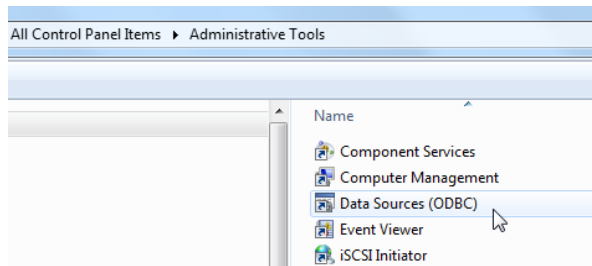
- ODBC log file
- Vertica log file
- Tableau log file

ODBC Log File

The ODBC Driver Manager has a trace facility that records the sequence of function calls that an ODBC application makes. You need to enable ODBC tracing and (optionally) specify the name of the log file in which Tableau records this information:

1. Before you start ODBC tracing, close Tableau.
2. Open the ODBC Data Source Administrator tool. Find the icon for **Data**

Sources (ODBC) in the Control Panel under Administrative Tools.



3. Click the **Tracing** tab.
4. Click **Start Tracing Now** to enable tracing. The button then changes to **Stop Tracing Now**.

If you want, specify the name and full path to the log file in the **Log File Path** text box, for example, `C:\Odbclogs\myOdbclog.log`.

5. Click **OK**.
6. Open Tableau to interact with your visualizations. The ODBC log starts collecting or capturing all calls between the Tableau workbook and the Driver Manager, and between the Driver Manager and the ODBC driver.
7. Close Tableau.
8. To stop tracing, in the ODBC Data Source Administrator tool, click **Stop Tracing Now**.
9. Review the content of the generated ODBC log file.

Vertica Log File

Vertica records all database events in a log file.

An example of the path to the Vertica log file is:

```
/home/dbadmin/vmart/v_vmart_node0001_catalog/vertica.log
```

For more information, see [Monitoring Log Files](#) in the Vertica documentation.

Tableau Desktop Log File

If you have a slow workbook, the Tableau log file might help you identify the problem. To determine if a slow query is causing a slow workbook, test the query generated in Tableau directly in vsql.

An example of the path to the log file is:

```
C:\Users\ant\Documents\My Tableau Repository\Logs\log.txt
```

To find the problematic query in the log file:

1. Close Tableau Desktop.
2. Navigate to My Documents\My Tableau Repository\Logs and delete all the log files in that folder.
3. Open Tableau Desktop.
4. Perform the steps that causes the problematic query to execute.
5. Close Tableau Desktop.
6. Navigate to My Documents\My Tableau Repository\Logs and open the log.txt file.
7. Search for the end-query tag and find the query, as in this example:

```
{  
  "ts": "2016-01-20T12:58:40.178",  
  "pid": 24676,  
  "tid": "5688",  
  "sev": "info",  
  "req": "-",
```

```
"sess": "-",
"site": "-",
"user": "-",
"k": "end-query",
"v": {
  "query": "SELECT AVG("inventory_fact"."qty_in_stock") AS "avg_qty_in_stock_ok"
           FROM "public"."inventory_fact" "inventory_fact" INNER JOIN
           "public"."date_dimension" "date_dimension" ON
           ("inventory_fact"."date_key" = "date_dimension"."date_key")
           WHERE (("date_dimension"."date" >= (DATE '2012-01-01')) AND
           ("date_dimension"."date" <= (DATE '2016-12-31'))
           HAVING (COUNT(1) > 0)"

  "cols": 1,
  "protocol-id": 0,
  "rows": 1,
  "elapsed": 0.095,
  "query-hash": 1493363951
}
}
```

8. Extract the query and run the query using vsql.
9. If, when you run the query, the performance is similar in vsql as in Tableau, you may need to rewrite the query.

Tableau Server Log File

In Tableau Server, the log files related to connecting and querying Vertica are in the following folder:

```
C:\ProgramData\Tableau\Tableau Server\data\tabsvc\vizqlserver\Logs
```

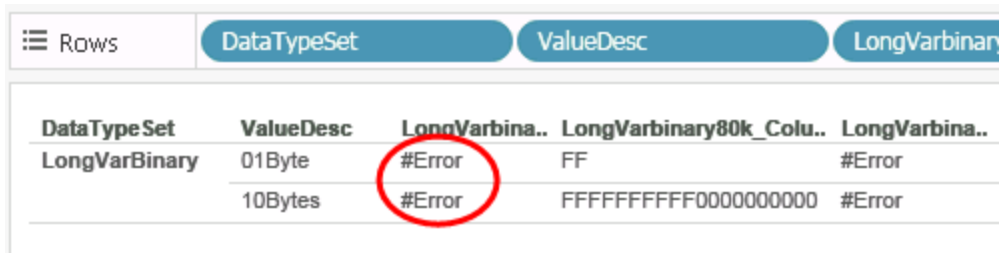
The Tableau Server log files are in JSON format.

For more information about Tableau Server log files, see [Work with Log Files](#).

Support for Vertica Data Types

Be aware of how Tableau handles Vertica data types.

- Tableau does not support LONG VARBINARY data types. Tableau displays #Error when Tableau tries to display LONG VARBINARY values:



DataTypeSet	ValueDesc	LongVarbina..	LongVarbinary80k_Colu..	LongVarbina..
LongVarBinary	01Byte	#Error	FF	#Error
	10Bytes	#Error	FFFFFFFFF00000000000	#Error

- Tableau displays up to 65,000 characters for LONG VARCHAR values. When Tableau tries to display a very large text value, it displays only the first 65000 characters of that string, truncating the remainder of the string.
- For TIMETZ and TIMESTAMPTZ data types, Tableau 10.0 does not support milliseconds and time zone offsets.
- Tableau supports the TIME data type but requires formatting to custom format: HH:mm:ss. Right-click the field, click Default properties > Date format.
 - The maximum number of digits for numeric values is 15. Tableau truncates very large numeric values to 15 digits as shown in these examples:

79228162514264337593543950335

displays as

792281625142643000000000000000

and

7922816251426433759354395.0335

displays as

79228162514264300000000000.00

- Digits to the right of the decimal point are truncated, for example:

7.9228162514264337593543950335

displays as

7.9228162514264300

and

79228162514264.337593543950335

displays as

79228162514264.3

- Tableau rounds some decimal values:
 - `-9999999999999999.9999` rounds to `-1000000000000000`.
- Tableau supports the FLOAT data type except for the following values:
 - NaN
 - +Infinity
 - -Infinity

Known Issues and Workarounds

The following topic describes a known issue when connecting from Tableau to the current version of Vertica. This topic also describes a workaround:

- [Multiple Active Result Sets Per Session](#)

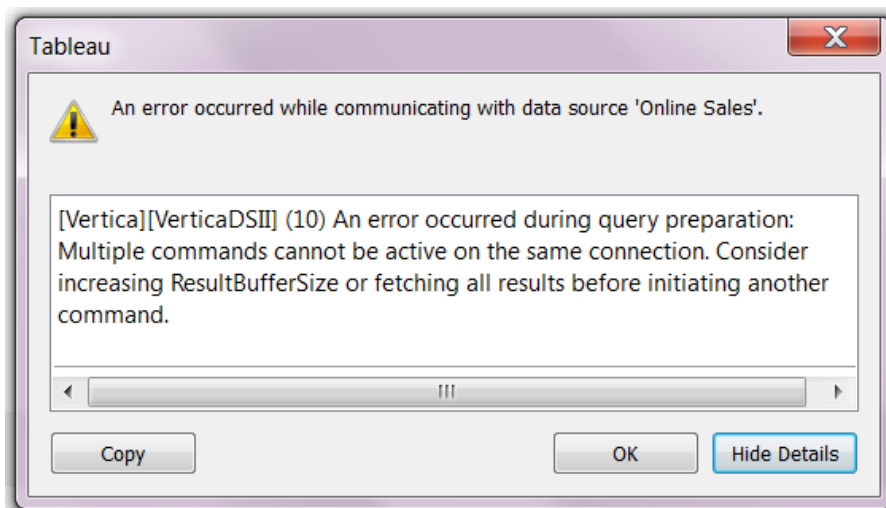
Multiple Active Result Sets Per Session

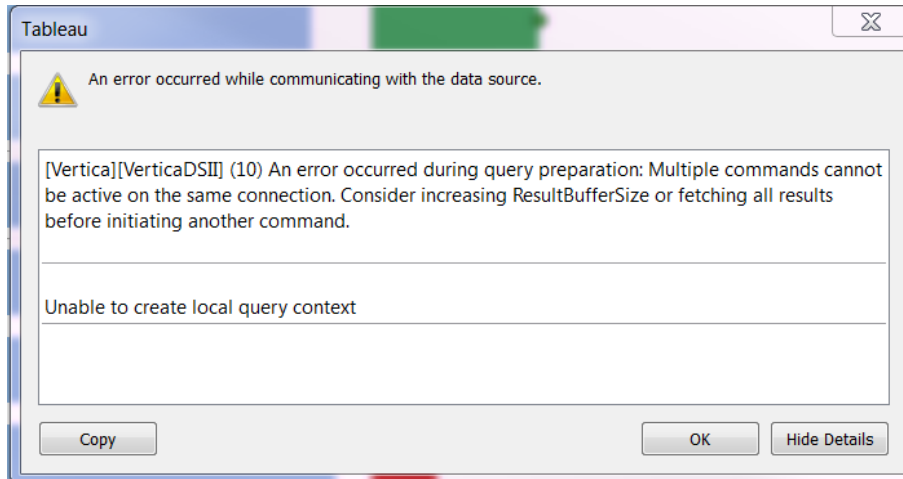
Issue

The Vertica driver does not support multiple queries sharing the same connection. Vertica can only run one query at time within a connection.

Scenario

If Tableau pushes multiple queries down to the Vertica server at the same time using the same connection, you may see the following errors:





Workaround

To work around this restriction, in the TDC file, change the value of the `ResultBufferSize` connection parameter.

If you are using large result sets in streaming mode, increase the size of your Vertica buffer. Setting the `ResultBufferSize` to 0 is equivalent to setting the buffer to an unlimited size. The default buffer size is 131072 bytes (128 KB). To increase the size of the buffer, add the following entry to your `.tdc` file, adjusting the `ResultBufferSize` according to the needs of your environment.

```
<customization name='odbc-connect-string-extras' value=' ResultBufferSize=500000' />
```

For information, see

- [Data Source Name \(DSN\) Connections Parameters](#) in the Vertica documentation.
- [Customize Your Connection to Vertica](#) in this document.

Vertica Integration with Tableau 10: Tips and Techniques

Known Issues and Workarounds