

Teradata Vantage™ Modules for Jupyter

Installation Guide

Release 3.0

March 2021




Copyright and Trademarks

Copyright © 2018 - 2021 by Teradata. All Rights Reserved.

Copyright Reference

All copyrights and trademarks used in Teradata documentation are the property of their respective owners. For more information, see [Trademark Information](#).

Product Safety

Safety type	Description
	Indicates a situation which, if not avoided, could result in damage to property, such as to equipment or data, but not related to personal injury.
	Indicates a hazardous situation which, if not avoided, could result in minor or moderate personal injury.
	Indicates a hazardous situation which, if not avoided, could result in death or serious personal injury.

Third-Party Materials

Non-Teradata (i.e., third-party) sites, documents or communications ("Third-party Materials") may be accessed or accessible (e.g., linked or posted) in or in connection with a Teradata site, document or communication. Such Third-party Materials are provided for your convenience only and do not imply any endorsement of any third party by Teradata or any endorsement of Teradata by such third party. Teradata is not responsible for the accuracy of any content contained within such Third-party Materials, which are provided on an "AS IS" basis by Teradata. Such third party is solely and directly responsible for its sites, documents and communications and any harm they may cause you or others.

Warranty Disclaimer

Except as may be provided in a separate written agreement with Teradata or required by applicable law, the information available from the Teradata Documentation website or contained in Teradata information products is provided on an "as-is" basis, without warranty of any kind, either express or implied, including the implied warranties of merchantability, fitness for a particular purpose, or noninfringement.

The information available from the Teradata Documentation website or contained in Teradata information products may contain references or cross-references to features, functions, products, or services that are not announced or available in your country. Such references do not imply that Teradata Corporation intends to announce such features, functions, products, or services in your country. Please consult your local Teradata Corporation representative for those features, functions, products, or services available in your country.

The information available from the Teradata Documentation website or contained in Teradata information products may be changed or updated by Teradata at any time without notice. Teradata may also make changes in the products or services described in this information at any time without notice.

Machine-Assisted Translation

Certain materials on this website have been translated using machine-assisted translation software/tools. Machine-assisted translations of any materials into languages other than English are intended solely as a convenience to the non-English-reading users and are not legally binding. Anybody relying on such information does so at his or her own risk. No automated translation is perfect nor is it intended to replace human translators. Teradata does not make any promises, assurances, or guarantees as to the accuracy of the machine-assisted translations provided. Teradata accepts no responsibility and shall not be liable for any damage or issues that may result from using such translations. Users are reminded to use the English contents.

Feedback

To maintain the quality of our products and services, e-mail your comments on the accuracy, clarity, organization, and value of this document to: docs@teradata.com.

Any comments or materials (collectively referred to as "Feedback") sent to Teradata Corporation will be deemed nonconfidential. Without any payment or other obligation of any kind and without any restriction of any kind, Teradata and its affiliates are hereby free to (1) reproduce, distribute, provide access to, publish, transmit, publicly display, publicly perform, and create derivative works of, the Feedback, (2) use any ideas, concepts, know-how, and techniques contained in such Feedback for any purpose whatsoever, including developing, manufacturing, and marketing products and services incorporating the Feedback, and (3) authorize others to do any or all of the above.

Contents

Chapter 1: Teradata Vantage Modules for Jupyter	5
Welcome to Teradata Vantage	5
Teradata Vantage Modules for Jupyter	5
Part I: Docker Installation	8
Chapter 2: Docker Installation Instructions	9
Downloading and Installing Docker	9
Loading the Docker Image for JupyterLab	9
Starting the Docker Container for JupyterLab to Persist User Data on a Docker-Managed Volume	9
Starting the Docker Container for JupyterLab to Bind to an Existing Directory	10
About Accessing the Docker Image for JupyterLab from a Remote Machine	11
About Connecting to the Docker Container for JupyterLab from a Browser	12
About Launching the Docker Container for JupyterLab from the Docker Toolbox on Windows	12
Stopping the Docker Container for JupyterLab	12
Chapter 3: Docker Container Troubleshooting	14
Running with Docker Toolbox	14
Verifying Correct Windows Version for Docker	14
Part II: Manual Installation into JupyterLab	15
Chapter 4: Manual Installation into JupyterLab Instructions	16
Preparing for Manual Installation into JupyterLab	16
Installing the Teradata SQL Kernel and Extensions	17
Updating PATH to Include Required Directories	18
Starting JupyterLab	19
Connecting to JupyterLab	20
About Accessing JupyterLab from a Remote Machine	20
Stopping JupyterLab	21
Uninstalling the Teradata SQL Kernel and Extensions	21
Chapter 5: Kernel and Extensions Troubleshooting	22
Troubleshooting the Teradata Kernel and Extensions	22
Part III: Upgrades	23
Chapter 6: Docker Image Upgrade	24

Upgrading the Docker Image	24
Chapter 7: Kernel and Extensions Upgrade	25
Upgrading Kernel and Extensions	25
Appendix A: Additional Information	26

Teradata Vantage Modules for Jupyter

Welcome to Teradata Vantage

Teradata Vantage™ is our flagship analytic platform offering, which evolved from our industry-leading Teradata® Database. Until references in content are updated to reflect this change, the term Teradata Database is synonymous with Teradata Vantage.

Advanced SQL Engine (was NewSQL Engine) is a core capability of Teradata Vantage, based on our best-in-class Teradata Database. Advanced SQL refers to the ability to run advanced analytic functions beyond that of standard SQL.

Teradata Vantage Modules for Jupyter

Teradata Vantage Modules for Jupyter allows users to do the following:

- Access Vantage in Python, R or SQL from JupyterLab notebooks
- Explore objects in a Teradata Vantage catalog
- Manage connections to Teradata Vantage

The Docker image includes JupyterLab, Teradata Vantage Modules for Jupyter, and other components to run as a Docker container on a client machine. A manual installation allows you to install components on a client machine that already has JupyterLab.

Teradata Vantage Modules for Jupyter is also available for installation as a fully Single Sign-On integrated AppCenter app. For information about Teradata Vantage Modules for Jupyter as an SSO integrated AppCenter app, contact your Teradata Account Representative.

What is Included?	Docker Image Installation	Manual Installation into JupyterLab
Navigator module	•	•
Teradata Connections module	•	•
Teradata SQL kernel	•	•
Teradata Package for Python	•	
Teradata Package for R	•	
Sample Teradata SQL notebooks	•	•
Sample Teradata Package for Python notebooks	•	
Sample Teradata Package for R notebooks	•	

What is Included?	Docker Image Installation	Manual Installation into JupyterLab
JupyterLab and other libraries helpful to data scientists	•	

Navigator Module

The Teradata SQL extension module includes a navigator module for viewing and browsing objects in Teradata Vantage, regardless of the language you are using in your notebook (Python, R, SQL).

The navigator provides the following:

- Hierarchical display of database objects
- Column metadata showing data type and indexes
- Row Count, Show DDL, Refresh and Column Distribution menu options

Connection Manager

The Teradata SQL extension module includes a connection manager. This allows you to manage connections to Vantage for use by the Navigator and SQL notebooks.

The Connection Manager provides the following:

- Connection management to add, remove, edit, copy, list, and test connections
- User interface that is independent of the SQL notebook
- Connections that are shared with the Navigator and SQL notebooks

SQL Kernel

The Teradata SQL extension module includes a Teradata SQL kernel. This allows you to run SQL commands on Vantage from a JupyterLab notebook.

The SQL kernel provides the following:

- Connection management to add, remove, connect, and list connections
- Query engine that uses embedded Teradata SQL driver
- SQL-aware notebook with SQL content assist and syntax checking
- Result set renderer that displays result data in an easy-to-read format, using scrollable grid with support to copy cells
- Execution history that stores execution metadata to recall SQL commands at a later time
- Visualization using Vega library to display charts, graphs, plots, and more
- Magic commands that provide additional custom kernel options to enhance your Teradata Vantage user experience
- Support for basic data import from a .csv file of up to 500k rows
- Preference settings allow users to modify logging options for the SQL Kernel

Python

With the Teradata Package for Python you can execute Python statements on Vantage from a JupyterLab notebook. For information about using the [Teradata Package for Python](#), see the following documents:

- *Teradata Package for Python User Guide*
- *Teradata Package for Python Function Reference*

Note:

If performing a manual installation, use the procedures in *Teradata Package for Python User Guide*, B700-4006.

R

With the Teradata Package for R you can execute R statements on Vantage from a JupyterLab notebook. For information about using the [Teradata Package for R](#), see the following documents:

- *Teradata Package for R User Guide*
- *Teradata Package for R Function Reference*

Note:

If performing a manual installation, use the procedures in *Teradata Package for R User Guide*, B700-4005.

Examples and Tutorials

Sample SQL, Python and R notebooks and additional information about using Teradata Vantage Modules for Jupyter can be found at: <https://teradata.github.io/jupyterextensions/>.



Docker Installation

Docker Installation Instructions

Downloading and Installing Docker

You must have Docker installed on your computer before you can install the Docker image for JupyterLab.

1. Select and download the appropriate Docker for your operating system:

Operating system	Docker Link
macOS	https://docs.docker.com/docker-for-mac/install/
Windows	https://docs.docker.com/docker-for-windows/install/
Linux	https://docs.docker.com/install/linux/docker-ce/ubuntu/

2. Follow the instructions on the above site to install Docker on your computer.

Note:

For questions about installing Docker on your computer, contact Docker for support.

Loading the Docker Image for JupyterLab

1. Download the Docker image for JupyterLab from the following Teradata site <https://teradata.github.io/jupyterextensions/>.

2. Load the Docker image for JupyterLab into Docker on your local machine:

```
docker load -i teradatajupyterlabext_<version>.tar.gz
```

3. (OPTIONAL) Change the name of the installed Docker image.

```
docker tag teradatajupyterlabext_<version> <any name of your choice>
```

Starting the Docker Container for JupyterLab to Persist User Data on a Docker-Managed Volume

This task describes how to start the Docker image so that user data persists on a Docker-managed volume.

Note:

On a macOS, you cannot easily access the volume outside of the running Docker.

1. Start the Docker container using the following command. It creates a Docker-managed volume `<volume_name>` and persists notebooks, Teradata SQL kernel connections, logs, result sets, and more inside the volume.

```
docker run -p 127.0.0.1:8888:8888 -v <volume_name>:/home/jovyan/
JupyterLabRoot teradatajupyterlabext
```

Note:

You can name the `<volume_name>` anything you choose, but the `/home/jovyan/JupyterLabRoot` must be entered exactly as shown.

Note:

The command window displays the URL and token needed to access JupyterLab from your browser.

2. Use the following commands to manage docker volumes:

Command	Description
<code>docker volume list</code>	Lists docker volumes.
<code>docker volume prune</code>	Prunes all volumes.
<code>docker volume rm vol-name</code>	Removes named volume. Note: You may need to prune your container before removing a volume. The container must be stopped before you prune using the command <code>docker stop container name</code> .
<code>docker container prune</code>	Prunes all stopped containers.

See Docker documentation for a complete list of commands used to manage docker volumes.

Starting the Docker Container for JupyterLab to Bind to an Existing Directory

This task describes how to start the Docker image to bind to an existing directory on your machine where you may have stored notebooks or files you want to access from the JupyterLab instance running inside the Docker container.

Note:

The GettingStarted notebooks and other files provided with this Docker image are hidden by your existing directory.

1. Start the Docker container:

```
docker run -p 127.0.0.1:8888:8888 -v /path/to/my/directory:/home/jovyan/
JupyterLabRoot terataajupyterlabext
```

Note:

You must map your directory to the /home/jovyan/JupyterLabRoot directory that resides inside the Docker image.

Note:

If you are using Windows consult the Docker documentation for directory path limitations or guidance.

Note:

The command window displays the URL and token needed to access JupyterLab from your browser.

About Accessing the Docker Image for JupyterLab from a Remote Machine

We recommend that you run the Docker image for JupyterLab and the browser on the same machine. You can access the Docker image from a browser running on a different machine.

Important:

If you choose to deploy this configuration, it is important that you not allow other users to access the same JupyterLab instance. Do not share your JupyterLab private token. A single JupyterLab instance should only be used by a single user. Sharing the instance allows all resources to become available to other users, including result set data and connections.

If you choose to use a remote browser remember the following:

- Consult the Jupyter documentation at https://jupyterlab.readthedocs.io/en/stable/public_server.html before accessing JupyterLab Docker from a remote machine.
- We recommend that you configure the Docker image to use HTTPS which requires a TLS certificate and key file. The certificate file must be named `tls.cert.pem`, and the key file must be named `tlskey.key`.
- The certificate files should be placed in a directory you bind to the /home/jovyan/certs directory ("/path/to/my/certfile/directory" in docker run command below). When the Docker image is started, it will detect the certificate files and will start up using HTTPS. You can also use Docker volume or different local directory to persist your notebooks as described above.

```
docker run -p 8888:8888 -v /path/to/my/certfile/directory:/home/jovyan/certs -v
teradata-vol:/home/jovyan/JupyterLabRoot terataajupyterlabext
```

or

```
docker run -p 8888:8888 -v /path/to/my/certfile/directory:/home/jovyan/certs
-v /path/to/my/directory:/home/jovyan/JupyterLabRoot teradatajupyterlabext
```

About Connecting to the Docker Container for JupyterLab from a Browser

When connecting to JupyterLab from a browser, keep the following items in mind:

- When the Docker image is started, a URL containing a token is displayed. Copy this URL to paste into your browser to connect to JupyterLab.
- You can also connect to JupyterLab using the URL: `http://localhost:8888` by entering the token when prompted

Note:

The JupyterLab service is configured for token authentication for security reasons. The token is displayed in the command window where you started the Docker image.

- If you mapped 8888 to a different port in your docker run command at setup, use the port you specified instead of 8888.
- A new token is generated each time the docker image is restarted.

About Launching the Docker Container for JupyterLab from the Docker Toolbox on Windows

If you have installed Docker Toolbox on your Windows system, remember the following:

- When you start your Docker container do not specify the `127.0.0.1` IP address.

```
docker run -p 8888:8888 -v /path/to/my/directory:/home/jovyan/JupyterLabRoot teradatajupyterlabext
```
- To connect JupyterLab use the URL `http://192.168.99.100:8888` and specify the token when prompted.

Stopping the Docker Container for JupyterLab

1. Run the following commands:

OS	Commands
Windows, macOS or Linux	<pre>docker ps docker stop <container id or container name></pre>

OS	Commands
Windows	If you add the -it options to the Docker run command, you can also stop the Docker container by entering the following in the terminal where you started the Docker container. Control-C
macOS	You can also stop the Docker container by entering the following in the terminal where you started the Docker container. Command-C
Linux	You can also stop the Docker container by entering the following in the terminal where you started the Docker container. Control-C

Docker Container Troubleshooting

Running with Docker Toolbox

When running on Windows with Docker Toolbox, be sure you have not specified the IP address 127.0.0.1 when running the Docker container. Docker Toolbox uses a special IP address 192.168.99.100. Specify either no IP address or the special IP address:

```
docker run -p 8888:8888 -v teradata-vol:/home/jovyan/  
JupyterLabRoot teradatajupyterlabext
```

Or

```
docker run -p 192.168.99.100:8888:8888 -v teradata-vol:/home/  
jovyan/JupyterLabRoot  
teradatajupyterlabext
```

Access the Docker container from your browser using the following URL (the IP address is required):

<https://192.168.99.100:8888?token...>

Verifying Correct Windows Version for Docker

Be sure to check the Docker website (<https://docs.docker.com/docker-for-windows/install/>) to confirm you meet the requirements for running Docker on Windows.

If your Windows system does not meet the requirements listed in the **What to know before you install** section on the above Docker website, you can upgrade your Windows machine or install Docker Toolbox https://docs.docker.com/toolbox/toolbox_install_windows/.



Manual Installation into JupyterLab

Manual Installation into JupyterLab Instructions

Preparing for Manual Installation into JupyterLab

1. If you do not already have JupyterLab installed on your desktop:
 - a. Follow the instructions in the JupyterLab documentation to install the supported version of JupyterLab from https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html.
Be sure your PATH is set correctly depending on the mechanism you choose to install JupyterLab.
 - b. Install the Node.js and npm packages from <https://www.npmjs.com/get-npm>.
2. If JupyterLab is already installed on your desktop:
 - a. From an appropriate command window verify that you have the supported versions of the following products installed:

Note:

See [Upgrading Kernel and Extensions](#) to upgrade from a previously installed release.

Product	Supported Version
JupyterLab	3.0.x
Node	v12.0 or later
npm	6.9.0 or later

Prompt Window	Command
Command terminal on macOS	<ul style="list-style-type: none"> Verify the version of JupyterLab: <code>jupyter lab --version</code> Verify the version of npm and node.js: <code>npm --version</code> <code>node --version</code>
Command window on Windows	<ul style="list-style-type: none"> Verify the version of JupyterLab: <code>jupyter lab --version</code> Verify the version of npm and node.js: <code>npm --version</code> <code>node --version</code>
Command terminal on Linux	<ul style="list-style-type: none"> Verify the version of JupyterLab: <code>jupyter lab --version</code> Verify the version of npm and node.js: <code>npm --version</code>

Prompt Window	Command
	<code>node --version</code>

Installing the Teradata SQL Kernel and Extensions

Downloading the Teradata SQL Kernel and Extensions

1. Download the appropriate Teradata Kernel zip for your operating system from <https://teradata.github.io/jupyterextensions/>.

Operating System	Teradata Kernel zip
macOS	<code>teradatasqlmacos_<version>.zip</code>
Windows	<code>teradatasqlwin_<version>.zip</code>
Linux	<code>teradatasqllinux_<version>.zip</code>

2. Unzip to your local file system.

The zip contains the teradatakernel binary, the Teradata extension tarballs, a directory containing license files, a directory called `teradatasql` which contains a `kernel.json` file that identifies the teradatakernel binary, and sample SQL notebooks.

Note:

For macOS, you must remove the quarantine attribute that macOS automatically applies to downloaded files, using one of the following commands:

- If you have not unzipped the file, before unzipping, run this command on the file:

```
xattr -d com.apple.quarantine teradatasqlmacos_<version>.zip
```

- If you have already unzipped the file, run this command on the directory:

```
xattr -dr com.apple.quarantine <directory name>
```

Note:

You must add the directory that contains the teradatakernel binary to your PATH.

Note:

Do not change the name of the `teradatasql` directory.

3. Update the PATH.

Follow the instructions in [Updating PATH to Include Required Directories](#) to update your PATH to include the location of your terdatakernel binary.

4. Run the following command in a command window to check that the correct version of the terdatakernel binary is in your PATH.

```
terdatakernel --version
```

Installing the Teradata SQL Kernel into JupyterLab

1. From a command terminal or window browse to the directory where you unzipped the .zip file.
2. Install the kernel:

```
jupyter kernelspec install terdatasql
```

3. Verify that the kernel was installed into JupyterLab:

```
jupyter kernelspec list
```

Installing the Extensions into JupyterLab

1. From a command terminal or window browse to the directory that contains the extensions from the .zip file:

```
jupyter labextension install teradata_database_explorer-<version>.tgz
```

```
jupyter labextension install teradata_resultset_renderer-<version>.tgz
```

```
jupyter labextension install teradata_sqlhighlighter-<version>.tgz
```

```
jupyter labextension install teradata_connection_manager-<version>.tgz
```

```
jupyter labextension install teradata_preferences-<version>.tgz
```

2. Verify that the extensions were installed in JupyterLab:

```
jupyter labextension list
```

Updating PATH to Include Required Directories

For macOS Installations

1. Add the following line to the ~/.bashrc or ~/.bash_profile file if you are using bash (or an equivalent for other shells):

```
export PATH=<required directory>:$PATH
```

2. Confirm that the directory is added:

```
echo $PATH
```

For Windows Installations

1. From the Control Panel, click **System > Advanced system settings > Environment Variables** and add the required directory to the PATH variable.

Note:

Depending on how you install and launch JupyterLab, it may be necessary to set your System PATH environment variable, not the User environment variable.

2. From a new command window, confirm the directory is added:

```
echo %PATH%
```

For Linux Installations

1. Add the following line to the ~/.bashrc or ~/.bash_profile file if you are using bash (or an equivalent for other shells):

```
export PATH=<required directory>:$PATH
```

2. Confirm that the directory is added:

```
echo $PATH
```

Starting JupyterLab

1. Return to the directory where you unzipped the Teradata zip file.

Note:

This is the location where any Notebooks and the Teradata workspace directory are created.

Note:

You don't need to start JupyterLab from the directory where you unzipped the Teradata zip file as long as the directory containing the teradatakernel executable is in your PATH.

If you start in a different directory, you need to upload the notebooks directory from the directory where you unzipped, into your JupyterLab workspace once it's running. Use the **Upload** button on the JupyterLab toolbar.

2. From a command terminal or window start JupyterLab:

```
jupyter lab
```

Note:

On Windows, if you receive an error indicating the `msvcp140.dll` library is missing, go to <https://support.microsoft.com/help/2977003/the-latest-supported-visual-c-downloads> and follow the instructions to download and install the missing library.

Note:

The command window displays the URL and token needed to access JupyterLab from your browser.

Connecting to JupyterLab

If your default browser does not automatically connect you to JupyterLab, you can connect manually.

1. Connect to JupyterLab using one of the following methods:

Option	Instructions
Use URL and token	Retrieve the URL from the command window from which you launched JupyterLab.
Use URL and manually enter token	Use <code>localhost:8888</code> and include the token from the command window when prompted.

About Accessing JupyterLab from a Remote Machine

We recommend that you run JupyterLab and the browser on the same machine. You can access JupyterLab running on a different machine.

Important:

- If you choose to deploy this configuration, it is important that you not allow other users to access the same JupyterLab instance. Do not share your JupyterLab private token. A single JupyterLab instance should only be used by a single user. Sharing the instance allows all resources to become available to other users, including result set data and connections.

If you choose to run JupyterLab on a remote machine remember the following:

- Consult the Jupyter documentation at https://jupyterlab.readthedocs.io/en/stable/public_server.html before running JupyterLab on a remote machine.
- We recommend that you configure JupyterLab to use HTTPS which requires a TLS certificate and key file.
- From a command terminal or window you should start JupyterLab with HTTP/TLS:

```
jupyter lab --certfile=tlscert.pem --keyfile=tlskey.key
```

Stopping JupyterLab

1. In the window where you launched JupyterLab, on Windows type Ctrl-C or on macOS type Command-C.

Uninstalling the Teradata SQL Kernel and Extensions

1. In a command terminal or window, type:

```
jupyter kernelspec uninstall teradatasql  
jupyter labextension uninstall teradata_database_explorer  
jupyter labextension uninstall teradata_resultset_renderer  
jupyter labextension uninstall teradata_sqlhighlighter  
jupyter labextension uninstall teradata_connection_manager  
jupyter labextension uninstall teradata_preferences
```

Kernel and Extensions Troubleshooting

Troubleshooting the Teradata Kernel and Extensions

Issue	Resolution
Errors starting the kernel	Always be sure your PATH is set correctly as directed by Jupyter installation instructions, any Jupyter pre-requisite software, and as noted above in the Manual Installation instruction sections. Errors starting the kernel may be the result of missing or misspelled directories in the PATH.
Confirm that kernel is executable	After extracting the files from the .zip file, confirm that the kernel binary is executable. Once the directory to the kernel is in your PATH you can type the command: <code>teradatakernel -h</code> to confirm the kernel is executable.
Wrong environment activated when running JupyterLab	If you use a packaging tool to install JupyterLab and the Teradata SQL extensions which supports different environments, be sure to activate the correct environment each time you run JupyterLab.
Kernel not started	When you open a Teradata SQL notebook you should see Teradata SQL in the upper right corner. This tells you the Teradata SQL kernel is running. If it says no kernel or you get no response when using any of the magic commands, your kernel has not started.
Navigator does not start correctly on Windows	When running on Windows, if the navigator does not start correctly (for example, no drop down displayed, etc.) you may need to manually downgrade a Python package called tornado to version 4.5.3. The command to install a specific version of tornado will depend on the packaging tool you used to install JupyterLab.
On Windows, new version of extensions not picked up	On Windows, if you detect that the new versions of the extensions are not running in JupyterLab, the old extensions may still be cached. Stop JupyterLab and run the following commands: <code>jupyter cache clean</code> <code>npm cache clean --force</code> Then re-install the latest extensions.
On macOS, if the instructions for removing the quarantine attribute were not followed, a dialog box message will be displayed and the kernel will not start.	Double check that you have removed the quarantine attribute that macOS automatically applies to downloaded files, using the following commands: <ul style="list-style-type: none"> If you have not unzipped the file, before unzipping, run this command on the file: <code>xattr -d com.apple.quarantine teradata_sql_macos_<version>.zip</code> If you have already unzipped the file, run this command on the directory: <code>xattr -dr com.apple.quarantine <directory name></code>
Trouble loading extensions	You must have external network access enabled for npm to install the extensions. If this access is blocked by a firewall, enable the firewall before you install the extensions.



Upgrades

Docker Image Upgrade

Upgrading the Docker Image

1. [Shut down any running instances of the Docker image for JupyterLab.](#)
2. [Optional] If you need to manually remove the previous Docker image for JupyterLab:

```
docker rmi -f teradatajupyterlabext
```
3. [Follow the instructions for installing a new version of the Docker image for JupyterLab.](#)
Make sure you use port number 8888 instead of the previously used port number.

Kernel and Extensions Upgrade

Upgrading Kernel and Extensions

1. [Shut down any running instances of JupyterLab](#) .
2. [Remove the previous version of the Teradata SQL kernel and extensions](#) .
3. Upgrade to the supported version of JupyterLab.

The command to upgrade JupyterLab depends on the packaging tool used to install JupyterLab.

Note:

Uninstall the previous JupyterLab version and then install the new version. There are a number of additional Jupyter packages that must be manually uninstalled before installing the new version of JupyterLab. Depending on your packaging tool, list all packages starting with "jupyter" and manually remove these packages before installing the new JupyterLab version.

4. [Follow the instructions for installing a new version of Teradata SQL Kernel and Extensions](#) .

Note:

For macOS, you must remove the quarantine attribute that macOS automatically applies to downloaded files, using one of the following commands:

- If you have not unzipped the file, before unzipping, run this command on the file:
`xattr -d com.apple.quarantine teradata_sqlmacos_<version>.zip`
 - If you have already unzipped the file, run this command on the directory:
`xattr -dr com.apple.quarantine <directory name>`
-

Note:

Make sure you update PATH to include the location of the new SQL kernel.

You can run `teradatakernel --version` from a command window to check this.

Additional Information

Changes and Additions

Date	Release	Description
March 2021	3.0	<ul style="list-style-type: none"> Updated feature in Navigator Module and SQL kernel Updated the supported versions for the product Updated Troubleshooting the Teradata Kernel and Extensions with the resolution for Trouble loading extensions Updated Upgrading Kernel and Extensions
September 2020	2.1	<ul style="list-style-type: none"> Updated the support of new JupyterLab version Removed the requirement for ZeroMQ Updated features in Navigator Module and SQL Kernel
July 2020	2.0.1	Included availability for installation as a fully Single Sign-On integrated AppCenter app
March 2020	2.0	<ul style="list-style-type: none"> Included Connection Manger Included instruction to manually install new connection_management extension for desktop installations Included version number in zip file and docker image Added --version command line option to the teradatakernel Updated the support of new JupyterLab version Updated ZeroMQ information for macOS
October 2019	1.2	<ul style="list-style-type: none"> Included Teradata R drivers and libraries in Docker image Included support for Linux Updated to Troubleshooting Updated ZeroMQ information for Windows
April 2019	1.1	Initial release

Teradata Links

Link	Description
https://docs.teradata.com/	Search Teradata Documentation, customize content to your needs, and download PDFs. Customers: Log in to access Orange Books.
https://support.teradata.com	One-stop source for Teradata community support, software downloads, and product information. Log in for customer access to:

Link	Description
	<ul style="list-style-type: none">• Community support• Software updates• Knowledge articles
https://www.teradata.com/University/Overview	Teradata education network
https://support.teradata.com/community	Link to Teradata community