# VerticaPy Installation Guide (Windows)

The following is a beginner-friendly installation guide for VerticaPy and Jupyter for Windows platforms.

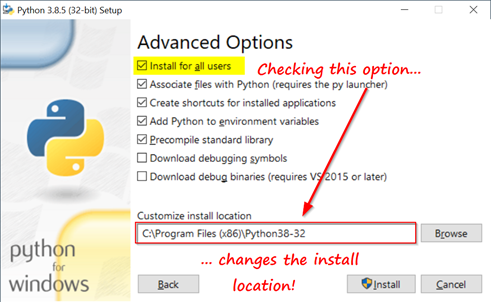
For detailed instructions on setting up Vertica for VerticaPy on other platforms, see the [VerticaPy Documentation](https://www.vertica.com/python/installation.php).

1. [Download the installer for the latest version of Python](https://www.python.org/downloads/).
2. Run the installer.
3. Check **Add Python x.y to PATH**.
4. Click **Customize installation**.

Sample Screenshot:



1. Check **Install for all users.**
2. Click **Install**.

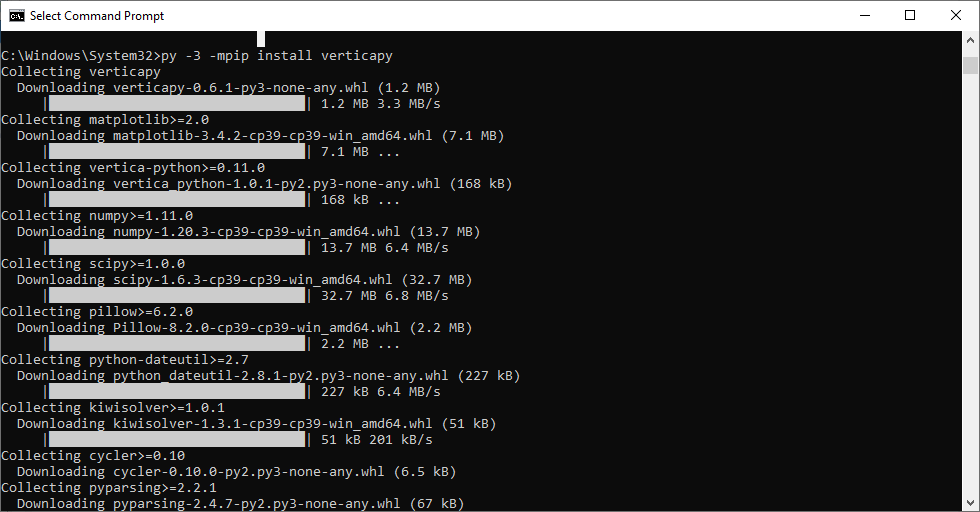


1. Open a PowerShell or Command Prompt.
2. Run the following:

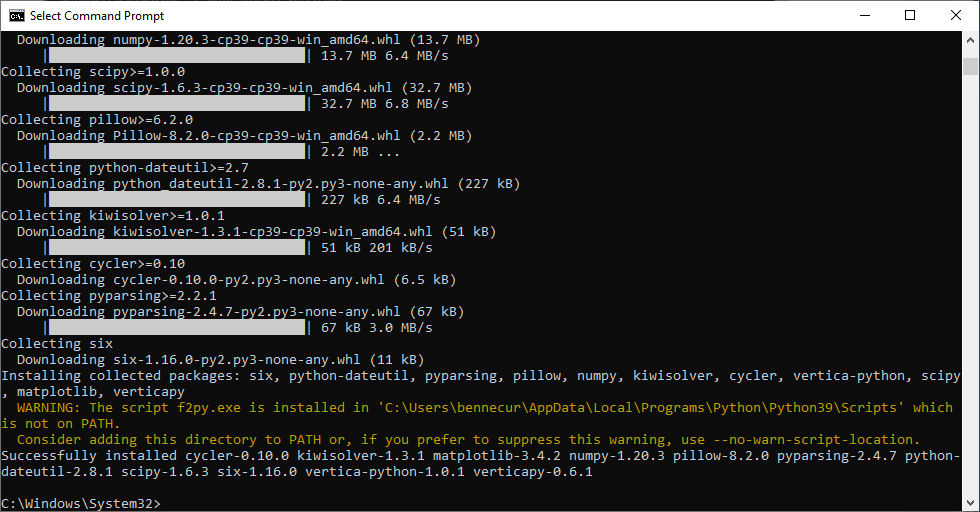


If you get an error stating that “pip3” couldn’t be found in the PATH, run the following:



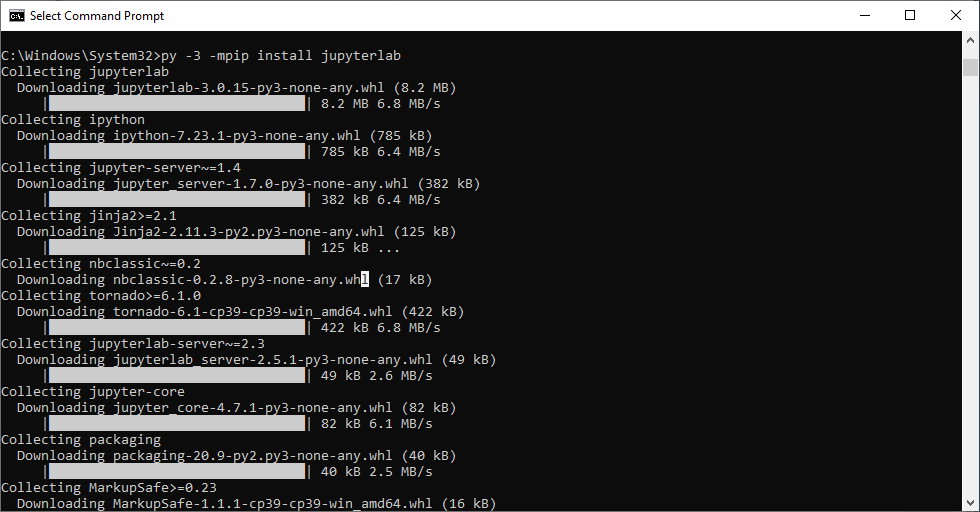


1. If you get an error about the **Scripts** directory not being in the PATH as shown below, you may need to [add it manually](#_Manually_Adding_"Scripts" ).



1. Run one the following to install **JupyterLab**. If one doesn’t work, try the other.





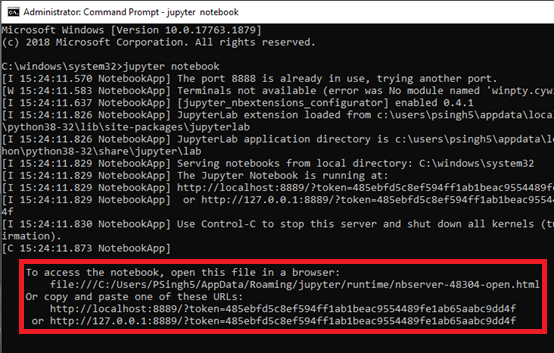
1. Navigate to a directory for your notebooks.



1. Run the following command to start Jupyter. The Jupyter Hub should open in your browser.



If your browser doesn’t open the Jupyter hub automatically, visit the notebook link in the console with your browser.



1. Click on the **New** dropdown and select a Python version to create a new Jupyter notebook.

Graphical user interface, application

Description automatically generated

1. Import the VerticaPy library into your environment and press the **Return** key to run the command through the Python interpreter. A new cell should appear for your next command.

Graphical user interface, text, application, email

Description automatically generated

1. Create a dictionary with your Vertica database’s host, username, password, and database name. For example:



1. Create a new auto connection by passing the dictionary and a name for the connection to the [new\_auto\_connection()](https://www.vertica.com/python/documentation_last/connect/new_auto_connection/) function,

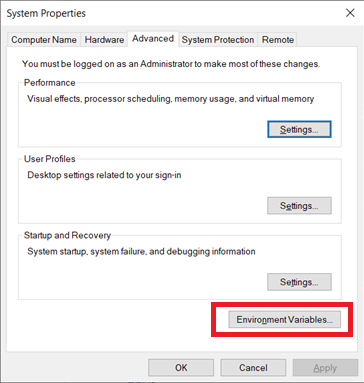


1. Connect to your Vertica database by passing the name of your auto connection to the [change\_auto\_connection()](https://www.vertica.com/docs/verticamlpython/connect/change_auto_connection/) function.

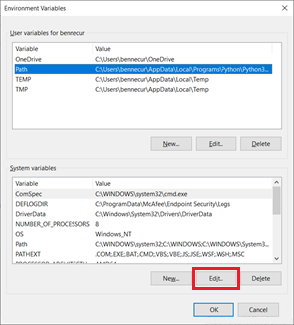


## Manually Adding "Scripts" to the PATH

1. Open the **Start** menu and search for “**Edit environment variables for your account**.” The **System Properties** window appears.
2. In the **Advanced** tab, click on **Environment Variables…** to open the **Environment Variables** window.

****

1. Click the **Path** variable to highlight it and click **Edit**. The **Edit Environment Variable** window opens.



1. Click **New** to add entry specified in original error and then click **OK**. This entry might look something like:



1. Restart your PowerShell or Command Prompt.

# Explore VerticaPy

## Loading Datasets

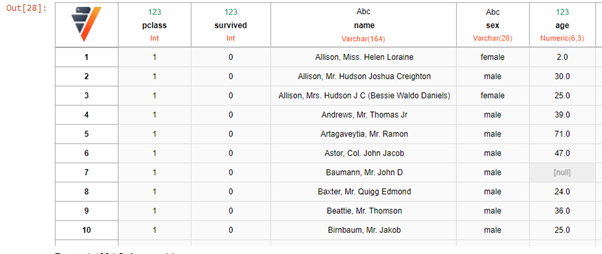
VerticaPy has many built in datasets to play around with. For this example, we'll use the classic the **"**titanic**"** dataset, which contains information on the passengers aboard the ship.

Run the following to load the **titanic** dataset into the [vDataFrame](https://www.vertica.com/python/documentation_last/vdataframe/object/index.php):



You can then view the loaded dataset with the following:





## Querying with SQL Syntax

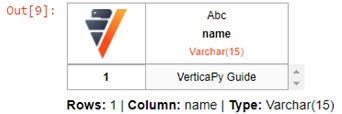
VerticaPy automatically converts our Python commands to SQL queries, but we can also use SQL queries directly with the built-in ["SQL Magic" Extension](https://www.vertica.com/python/documentation_last/extensions/sql/).

Here, we query for 10 rows from the "**titanic**" table:





This statement creates and queries a new "**test**" table:



## Data Exploration with VerticaPy

You can either use SQL or pandas-like functions to manipulate the data in your database. In both cases, VerticaPy pushes the query to the database and returns the results.

In this first example, we use SQL's [avg()](https://www.vertica.com/docs/10.1.x/HTML/Content/Authoring/SQLReferenceManual/Functions/Aggregate/AVGAggregate.htm) function to calculate the average of the "survived" column from the "titanic" dataset. This SQL query is pushed to the database as-is. As with a standard SQL SELECT function call, this returns the result in a table.



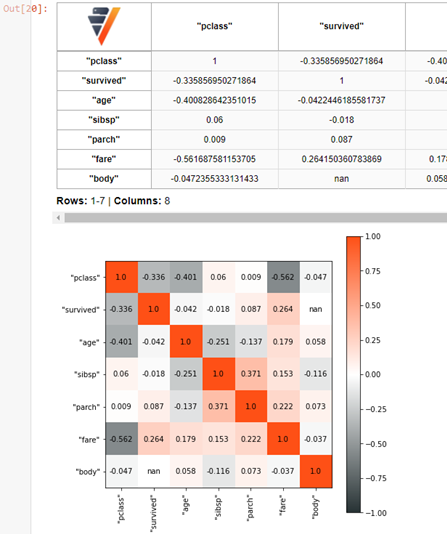


In this second example, we use the [vDataFrame.avg()](https://www.vertica.com/python/documentation_last/vdataframe/vcolumn-methods/avg/) method to calculate the average of the same column. In this case, VerticaPy converts the Python command to SQL before pushing it to the database, and, since we didn't use SELECT, the result is a float instead of a table.

The calculated average, however, is identical, because it all boils down to the same query passed to the database.

A basic data exploration exercise might be to calculate the correlations between different variables. You can calculate and create a heatmap of these correlations with the [vDataFrame.corr()](https://www.vertica.com/python/documentation_last/vdataframe/main-methods/corr/) method.



## Python Scripts with VerticaPy

You can also use VerticaPy in Python scripts outside of a Jupyter notebook.

This repository includes an example script "churn-workflow.py." This file goes through the full data science cycle from start to finish. This cycle covers data exploration, data preparation, data modeling, model evaluation, and model deployment, and saving the model to the Vertica database. For a detailed walkthrough this particular study, see the [Telco Churn example](https://www.vertica.com/python/examples/telco_churn/) in the VerticaPy documentation.

To run this example, simply replace the database connection properties with your personal Vertica instanceand run the script by navigating to the directory of the file, and running"churn-workflow.py" with the following command (run in PowerShell or the Command Prompt):

The output should look something like this:

