# Tableau + Python Lab Instructions

# Logging In

The Resources tab above contains the Username and Password for the VM.

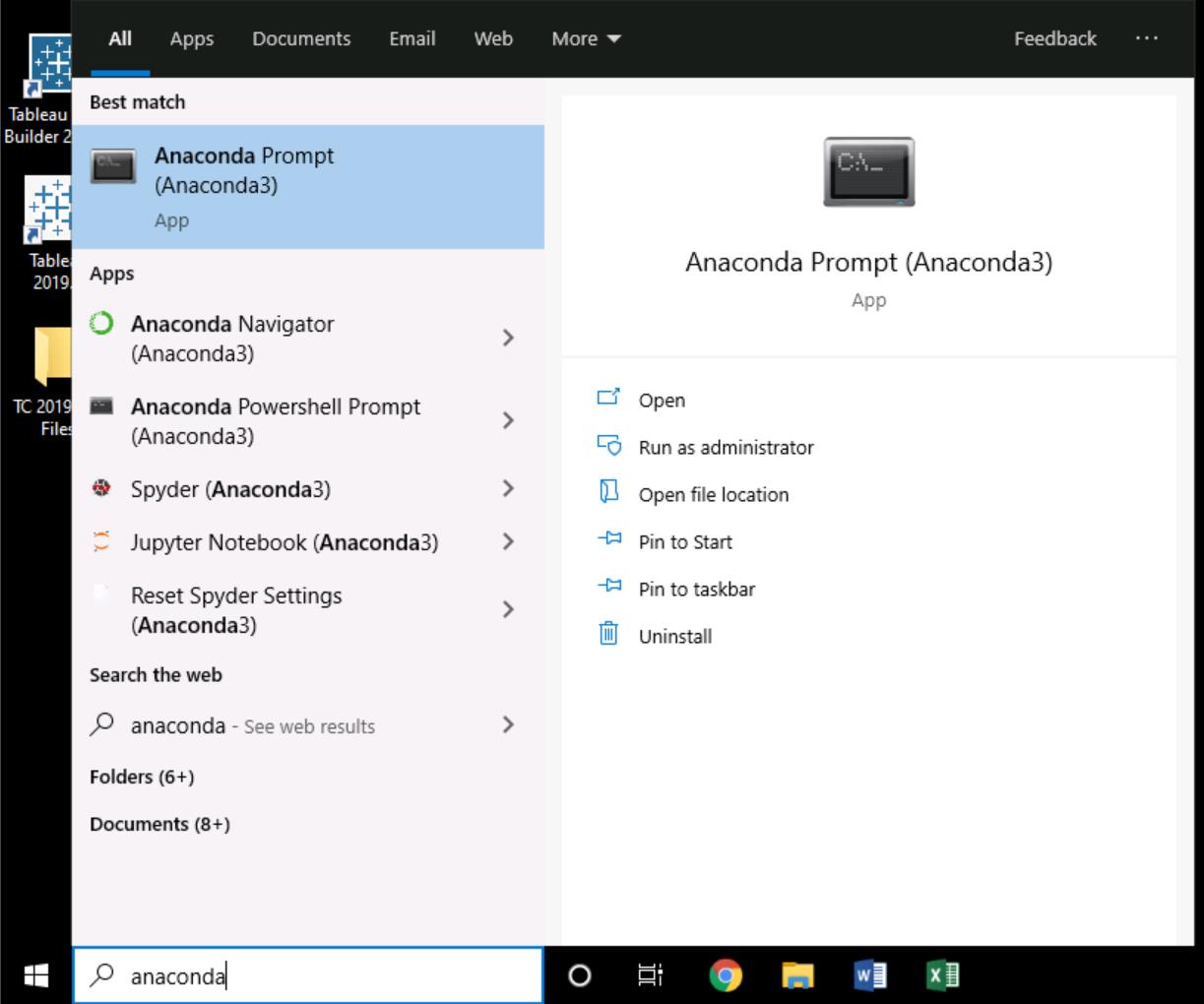
# Installation

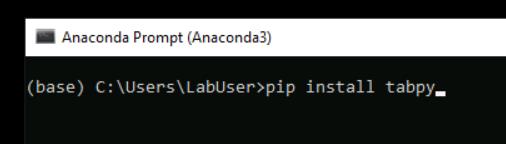
## Installing Python

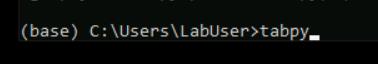
Python 3 has already been installed on your training machine from https://anaconda.com/distribution

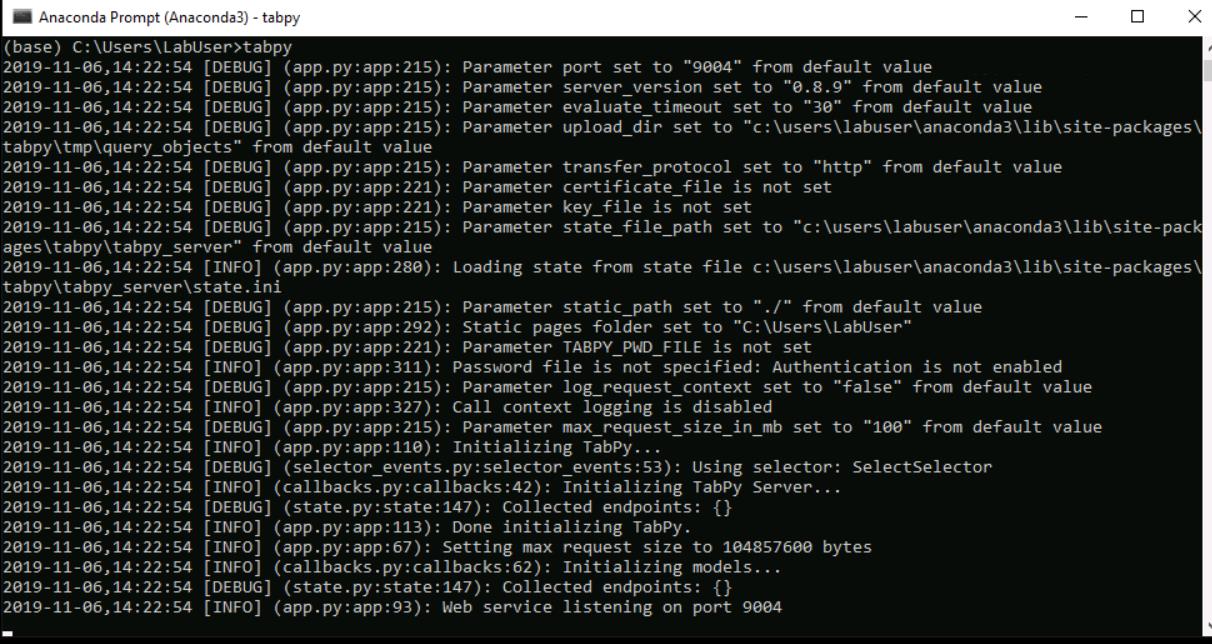
Anaconda is a convenient way to install Python plus Jupyter notebooks and all of the tools needed for a complete data science workstation.

## Installing TabPy

**Start an Anaconda command prompt.** 

**At the command prompt, type: pip install tabpy** 

**At the command prompt, start TabPy by typing: tabpy** 

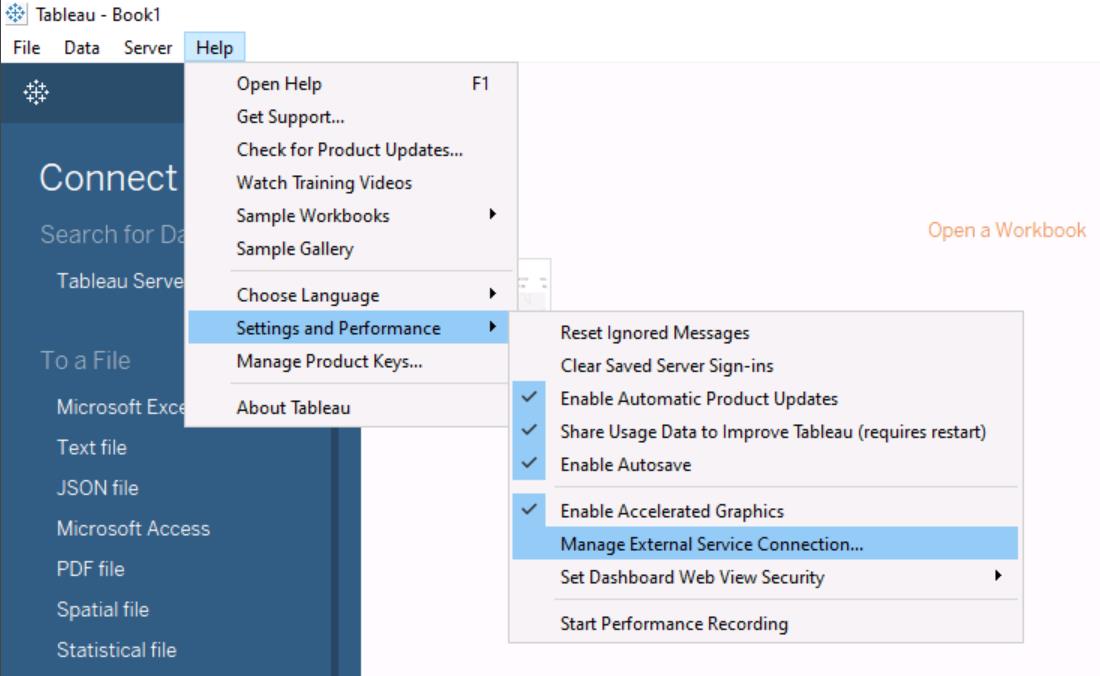


Be sure to **note on which port tabpy is running**. This will be the last output line from startup.cmd. **9004 is the default**.

# Tableau Desktop

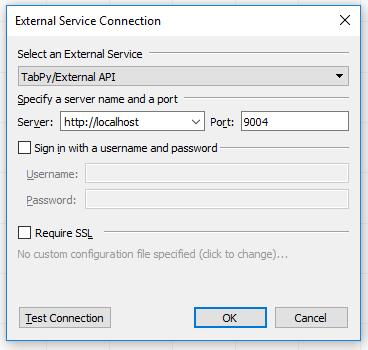
**Open Tableau Desktop 2019.3**.

## Connecting Tableau Desktop to TabPy

To connect Tableau Desktop to your TabPy server: **Select the Help menu, then select Manage External Service Connection** 

**Enter the server name or ip address of the machine on which TabPy is running.** In this case, "localhost" since it's running on this machine. **Enter the port number that you observed when you started TabPy (9004)**.

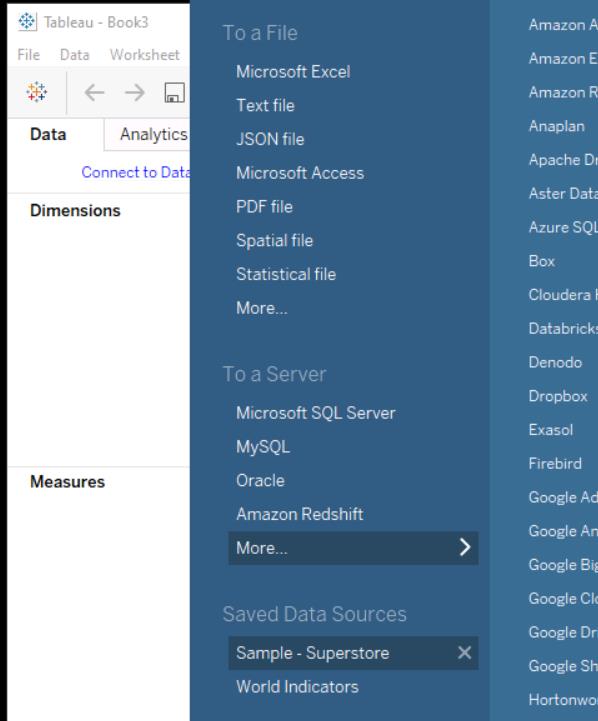
We have not configured any security on our TabPy server, so **leave the "sign in" and "Require SSL" boxes unchecked**.



## Executing Python Scripts from Tableau Desktop

Tableau can pass code to TabPy through one of four different functions: SCRIPT\_INT, SCRIPT\_REAL, SCRIPT\_STR and SCRIPT\_BOOL. Each of the four functions accommodates a different return type.

### Exercise 1

In Tableau Desktop, **Connect to the SuperStore sample dataset**. 

**Create a calculated field** by selecting Analysis / Create a Calculated Field…

**Name this new calculation "Region Country"**

**Type the following formula:**

SCRIPT\_STR(“

print(\_arg1)

print(\_arg2)

lst=[]

for i in range(0,len(\_arg1)):

lst.append(\_arg1[i]+', '+\_arg2[i])

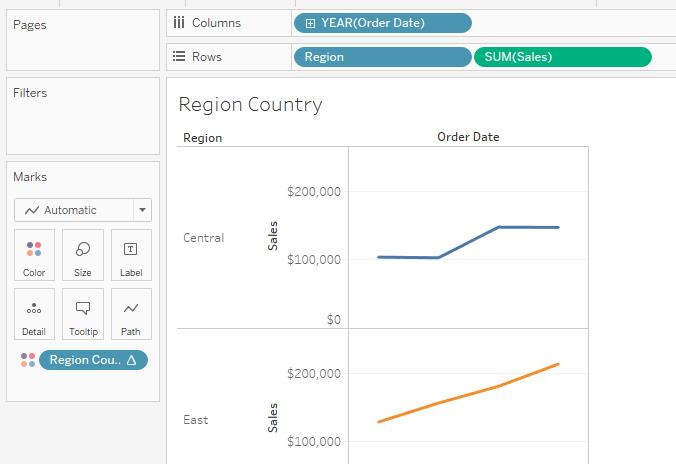
return lst

",

ATTR([Region]),ATTR([Country])

)



On an empty worksheet **Drag year of Order Date to columns. Region to Rows, Sales to rows. Drag the new Region Country calculation to color.** 

Observe the result of the RegionCountry calculation in the tooltip.

### Exercise 2

**Create a calculated field** by selecting Analysis / Create a Calculated Field…

**Name this new calculation "Discount Profit Correlation"**

SCRIPT\_REAL("

print(\_arg1)

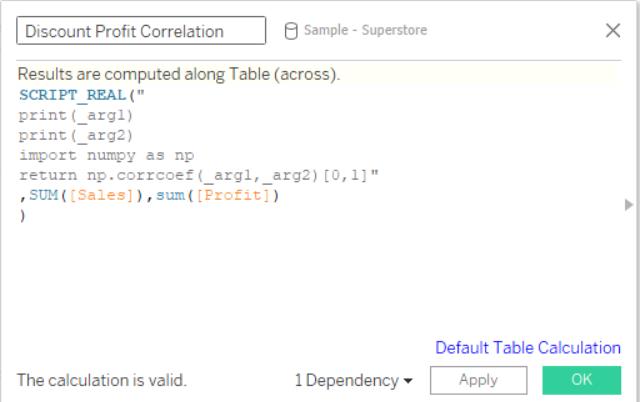
print(\_arg2)

import numpy as np

return np.corrcoef(\_arg1,\_arg2)[0,1]"

,SUM([Discount]),sum([Profit])

)



On the new sheet you created, **drag Region to columns and Segment to rows**. **Drag SalesProfitCorrelation to Text in the Marks ledge, or just double-click on it**.

Notice that sales and profit are more highly correlated for the consumer segment and the corporate segment than for the home office segment.

Reference "Using Python in Tableau Calculations" here: https://github.com/tableau/TabPy/blob/master/docs/TableauConfiguration.md

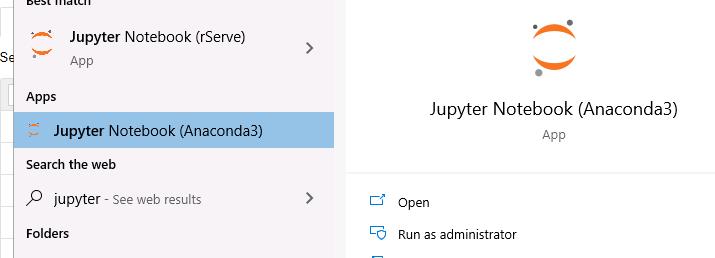
# Using Deployed Functions from Tableau Desktop

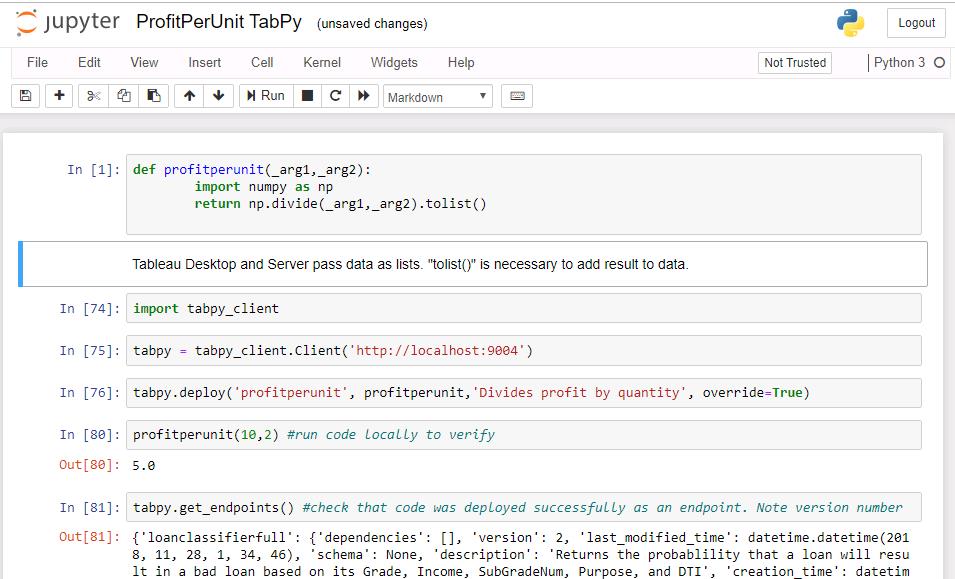
### Exercise 3

Some files that were supposed to be copied to the VM didn't make it. We'll pull them from github instead.

**Open GitHub Desktop. Select the TableaPrepPythonTitanic repository. From the Repository menu, choose the pull command.**

**In a new Anaconda prompt, type "pip install tabpy\_client" to install the libraries necessary for Python to interact with TabPy.**

**Start Jupyter Notebook (Anaconda3)** 

**In Jupyter Notebook, navigate to C:\Users\LabUser\Documents\GitHub\TableauPrepPythonTitanic\TC19\ and open "ProfitPerUnit TabPy"** 

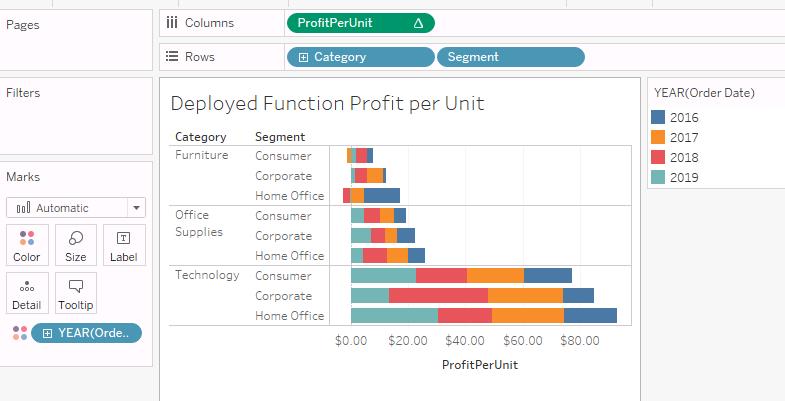
**In each cell sequentially, press Shift+Enter to execute the Python code for that cell.**

**In Tableau Desktop, create a Profit Per Unit calculation with the following script:**

SCRIPT\_REAL("

return tabpy.query('profitperunit',\_arg1,\_arg2)['response']",

SUM([Profit]),SUM([Quantity]))

**Explore Superstore with the new Profit Per Unit calcluation.** 

# Using Python with Tableau Prep Builder

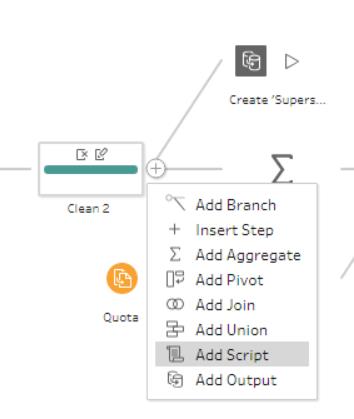
## Connecting to TabPy from Prep Builder

### Exercise 4

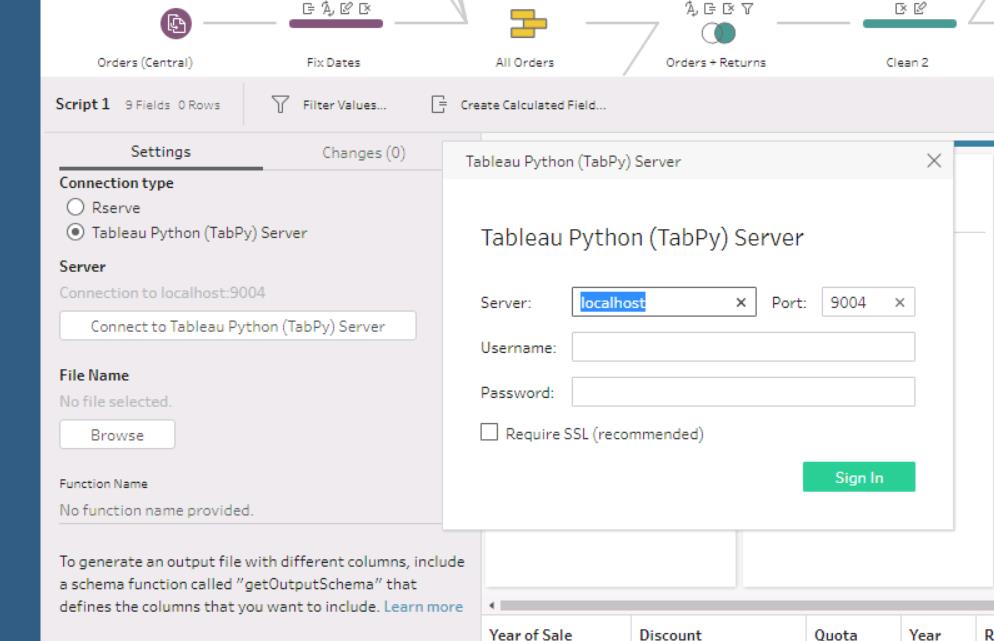
**Open Tableau Prep Builder**

**Open the Superstore Sample flow**

**Click the plus icon to the right of the "Clean 2" step**

**Select Add Script** 

**In the Settings panel at the left edge of the Prep builder UI, Select "Tableau Pyython (TabPy) Server" as the Connection type**

**Click "Connect to Tableau Python (TabPy) Server"** 

We will connect to our existing TabPy server. The same one we used for the Tableau Desktop exercises. **Enter "localhost" as the server and port 9004**

### Write the Python script

**Open Visual Studio Code**

Exercise scenario: Sales management would like to gain insight on the revenue effect if discounting. To support this, we will use Python to add a field to our flow called "non-discounted sales".

In this context, our Python script must be written with a function that can be called from Prep Builder. We choose the name of this function. It will accept a structure called a dataframe, and will return a dataframe.

Create a new file called nds.ps (nds = non-discounted sales).

First we'll simply print the dataframe to get a look at what our function will receive from Prep Builder.

**Add the following Python code to nds.py:**

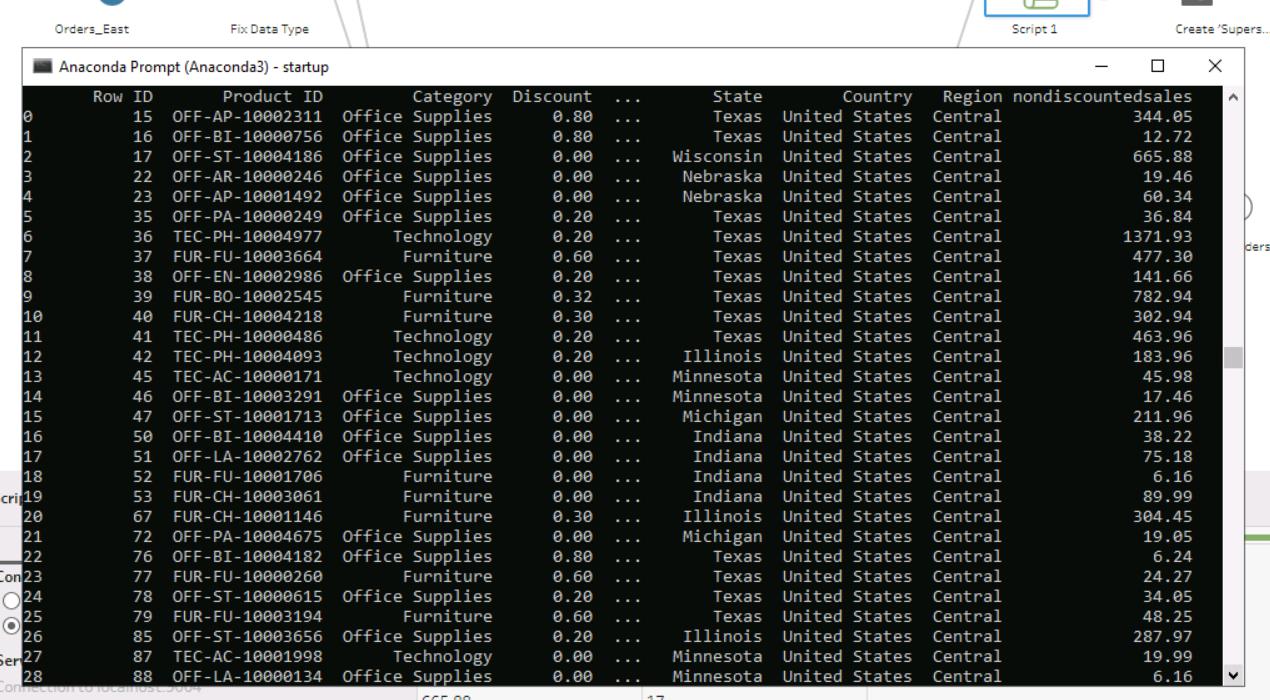
def nds(df):

print(df)

return(df)

**Save the file.**

**In the configuration panel for the script added above, browse to the nds.py file to connect this script step to our new Python script.**

**Look at the command prompt running TabPy. You should see the results of the print statement.** 

Let's add our calculation. **Replace the print(df) statement with the following:**

df['Non Discounted Sales'] = df['Sales'] / (1-df['Discount'])

This line creates a new column in the dataframe called "Non Discounted Sales".

**Be sure the two leading spaces are included. The Python specification requires the contents of a function to be indented.**

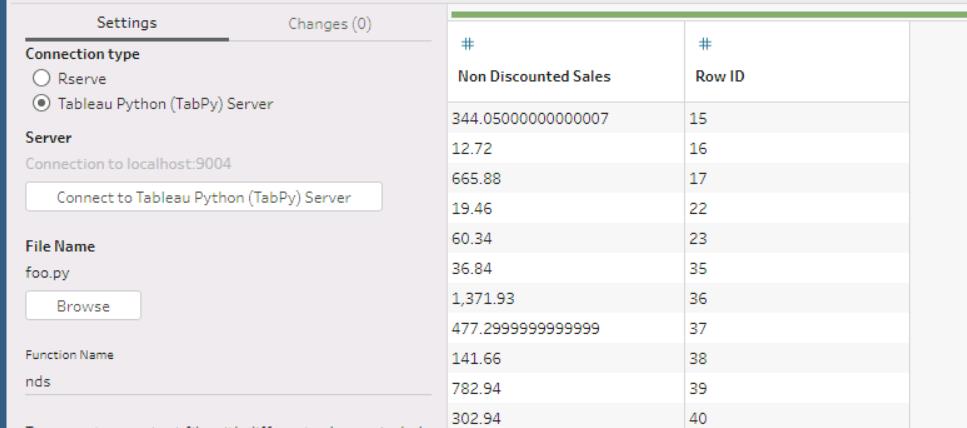
If the schema of the returned dataframe does not exactly match that of the calling dataframe, we need to include an additional function called "get\_output\_schema".

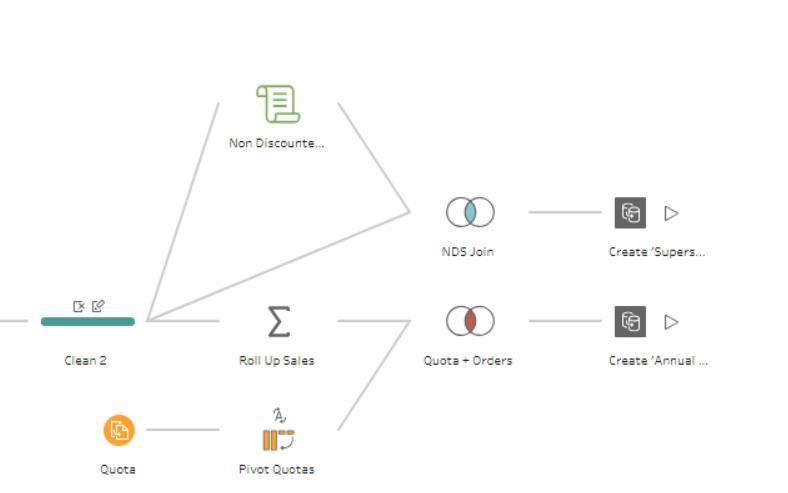
In this case we will return our new "Non Discounted Sales" column along with the "Row ID" column so that we can join this column back into the flow.

**Add the following to nds.ps:** def get\_output\_schema(): return pd.DataFrame({ 'Row ID': prep\_int(), 'Non Discounted Sales': prep\_decimal() })

**Save the file.**

**In Prep Builder, in the flow, click off of the script step icon then back on it. This will cause Prep Builder to re-execute the Python script.**

You should see the result of the script in Prep Builder. 

**Join the result of the script to the flow before the "Create Superstore Sales.tde" output step.** 

# Extras

Browse to https://www.tableau.com/about/blog/2019/10/using-tableau-preps-new-python-integration-predict-titanic-survivors and use the code at https://github.com/tabblogs/TableauPrepPythonTitanic to explore further.

## Thank You for being a Tableau Customer!