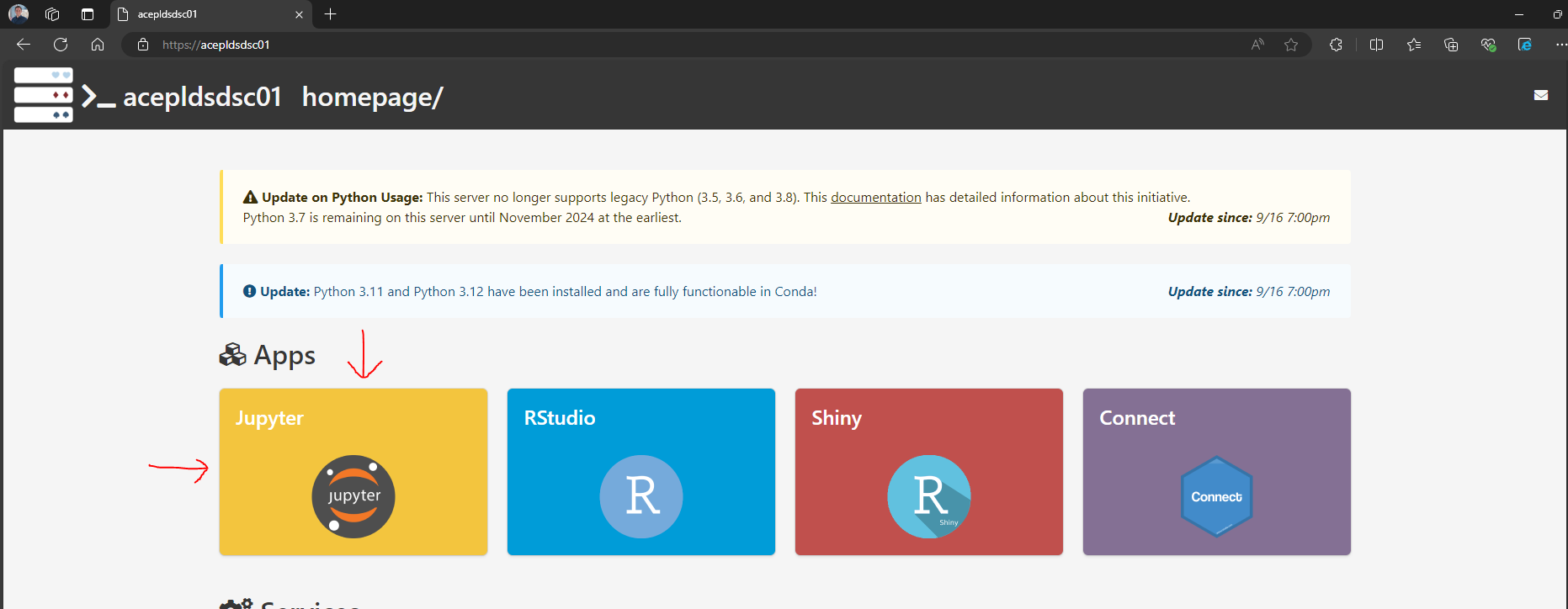
**Business Statement:**

The Oracle to Snowflake Pipeline is crucial for automating data retrieval and connecting multiple data warehouses into a cohesive pipeline. By leveraging Snowflake’s scalable computing resources, the pipeline can efficiently handle complex queries, allowing our team to programmatically access data in Oracle. Additionally, this pipeline offers new capabilities for ingesting Excel files from internal partners directly into Snowflake, simplifying data querying and transformations.

*Disclaimer:* For this pipeline, all the code implementation, key-pair generation, and password storage will be done in JupyterLab on the ACE servers. While it is possible to do this in other programs like PuTTY and Visual Studio Code, this document is only an instruction guide for how to connect Oracle and Snowflake using Jupyter and the ACE servers. If you wish to get the same access as me, Peter Alonzo, for PLDA, please reference this MTE ticket: [RITM4730777 - mTE Portal (service-now.com)](https://thehartford.service-now.com/ess?id=mte_ticket&table=sc_req_item&sys_id=51721cfd1bc092100ce1a8a0604bcb95)

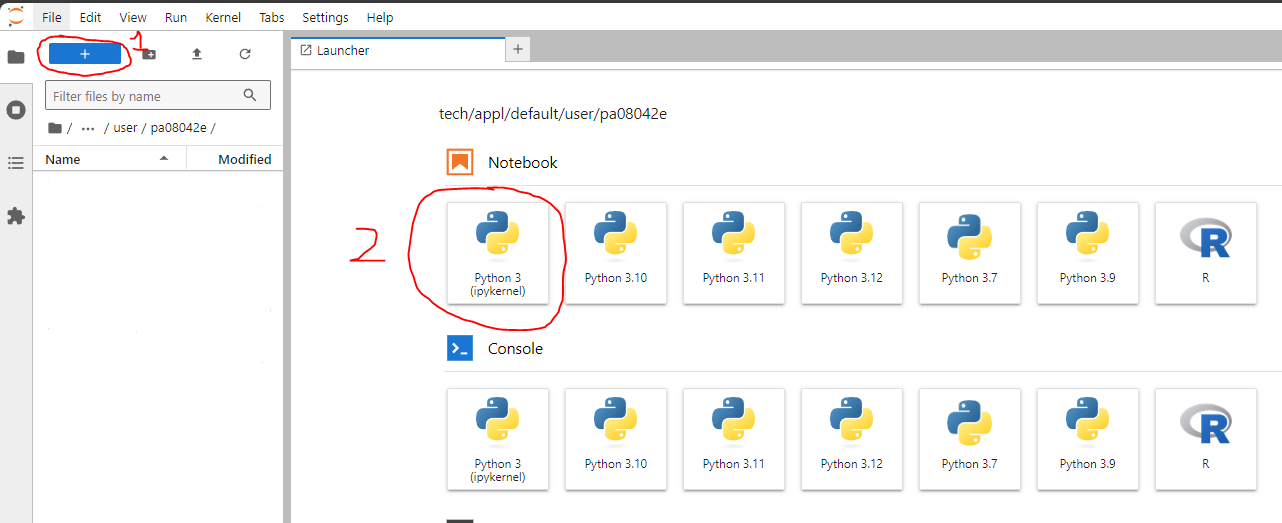
**Section 1: Accessing ACE servers and Creating a Jupyter Notebook**

When you first get access to the ACE servers, a user workspace is usually created that allows you to create notebooks, terminals, etc. In Microsoft Edge or your preferred browser, navigate to the ACE server you were given permission to and log in with your ACE credentials. Once you are logged in, click on Jupyter to launch JupyterLab. For reference, I use acepldsdsc01 as my preferred ACE server.



You will now need to navigate to your workspace. This will allow us to run the pipeline program and generate the public/private key pair for Snowflake. Your workspace location on the ACE server will vary based on your department so please reach out to your manager or myTechExpress to find a place where you can work. For users who used my ticket to get ACE server access, the path to find your workspace is “tech/appl/default/user/<your username>”.

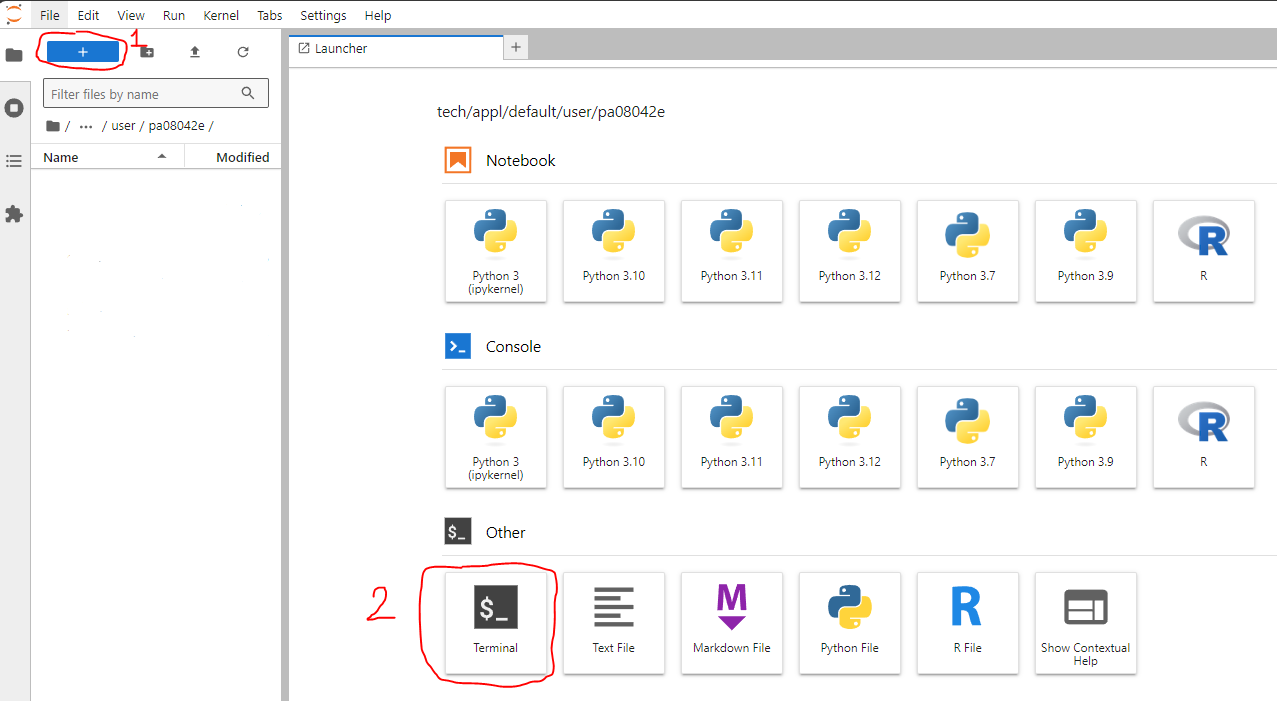
Once you are in your workspace, you can either drag my pipeline program directly into this folder or you can create your own notebook. To make your own, click on the blue “+” sign in the top-left and select a new Jupyter notebook. For this program, I chose the default python kernel notebook.



**Section 2: Public/Private Key pair for Snowflake**

To programmatically upload tables to Snowflake, a public/private key pair needs to be created and linked to your Snowflake account. Since HIG employees log in to Snowflake using SSO credentials and we have default read/write permissions for our personal schema, an MTE ticket needs to be used to allow us to programmatically write to Snowflake.

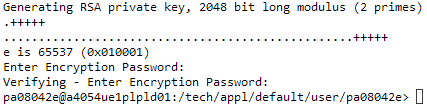
In your workspace, click on the blue “+” sign and open a new terminal.



First, you need to run the command to generate a private key. In your terminal, please type the following command:

*openssl genrsa 2048 | openssl pkcs8 -topk8 -inform PEM -out <name your private key file>.p8*

After hitting enter, you will be prompted to enter an encryption password. This can be anything you want to help authenticate your access to the private key. Please write down this password somewhere, as we will need it for our pipeline program.



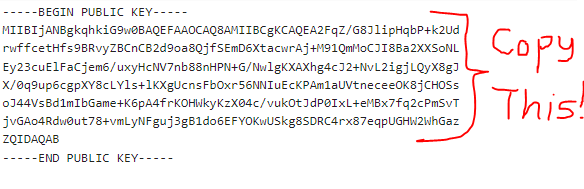
Once you’ve entered your encryption password, you should see the file generated in the same folder as your Jupyter notebook. Now, we can generate the public key. In the terminal, enter the following:

*openssl rsa -in <name of your private key file>.p8 -pubout -out <name your public key file>.pub*

Once you hit enter, you will be prompted to enter the encryption password you set for the private key. After entering your encryption password, you will now have files created for your key pair! The last step is filling out an MTE ticket that will connect your public key to your Snowflake account. Please fill out a “Snowflake Cloud Data Platform Access Services” ticket like this:

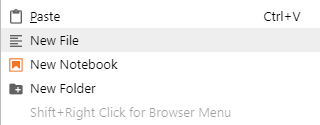
[RITM4862315 - mTE Portal (service-now.com)](https://thehartford.service-now.com/ess?id=mte_ticket&table=sc_req_item&sys_id=b52fa14b1b95de10aea40ed3604bcbd4)

For the public key field of the ticket, enter the contents of your public key file into the textbox excluding the header and footer. Once the ticket is fulfilled, you will now be able to programmatically access Snowflake with python!



**Section 3: Setting Up Configuration File for Passwords (.cfg)**

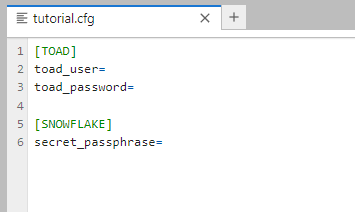
Now that our key pair is generated and being linked to our Snowflake account by myTechExpress, we can set up a configuration file to store our passwords! This will be useful for sharing code with others, as directly embedding passwords into our code is terrible for security practices. In the white space below your notebook and key files, right-click and select New File.



You can name your configuration file anything you want but make sure you end in .cfg. As an example:

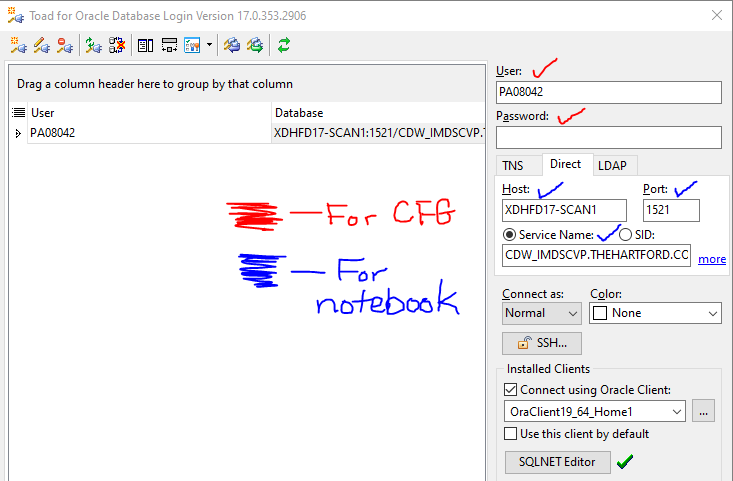


From there, please follow the template configuration file I provided and store any credentials you may need for this pipeline. For the Snowflake section, please enter the encryption password you made during the key-pair creation in the *secret\_passphrase* variable. If you want to connect to Oracle, we will fill out the Toad section of the cfg file in Section 4. If you are only concerned with writing tables to Snowflake, please skip to Section 5.

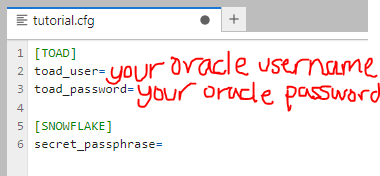


**Section 4: Transforming Oracle Query to Pandas Dataframe**

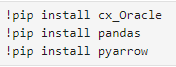
With all the preparation work done, we can now start extracting data from our warehouses using Python! For this step, you will need to have been granted access to your desired database in Oracle via an MTE ticket. Given that you have already tested your access, we can save your credentials in our notebook and cfg file.



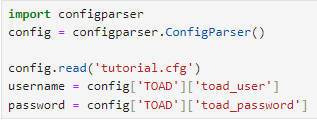
For the two cells in red, you can enter these values into your cfg file under the Toad section. It will look something like this:



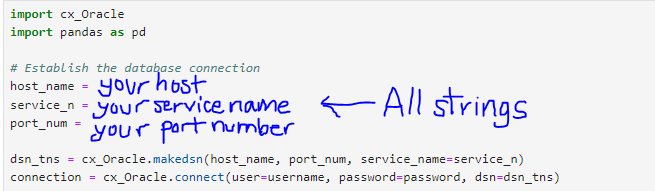
Once our Oracle credentials are saved, we can now run the first few cells of our notebook. To ensure you have the correct libraries installed, please run the following cell at the top of your notebook:



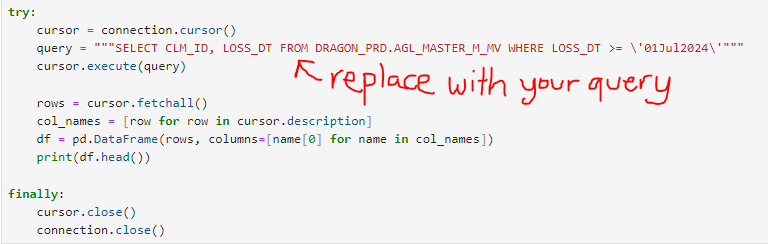
We also need to run a cell to read our configuration file and save our login credentials as variables. This will be crucial for establishing save security practices and keeping your access to Oracle private. Please run the following cell in your notebook (replace tutorial with the name of your cfg file):



With our login credentials saved, we can now establish a database connection to Oracle. Using the username and password variables you created in the previous cell, please fill in the information for your desired host name, port number, and service name found in the first screenshot of Section 4:



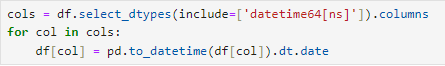
After establishing a database connection to Oracle, we can now write a SQL query to Oracle and save its results as a pandas dataframe. To ensure your query will work properly, try writing your query in the editor in Oracle and see if you get the correct results. Once you verify that, you can use the following code to programmatically save the results in our notebook as a dataframe (df):



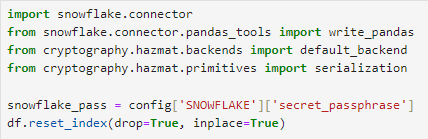
**Section 5: Uploading Dataframes to Snowflake**

Assuming your MTE ticket made in Section 2 was fulfilled, you will now be able to upload tables to Snowflake! Please note that if you skipped Section 4, you would need to create a pandas dataframe with data you wish to export. Please ensure that you run “!pip install pandas” and “import pandas as pd” before attempting to make a dataframe. This dataframe can be made manually or derived from an Excel/CSV file. If you didn’t skip Section 4, you can export the dataframe you created from your Oracle query.

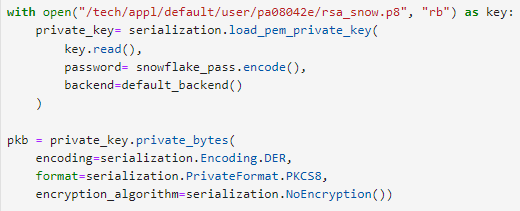
If your dataframe has any date fields you wish to export to Snowflake, I highly recommend running the following cell. The function we use to write to Snowflake in this tutorial will convert timestamp fields to Unix epoch time, which can be difficult to fix in Snowflake.



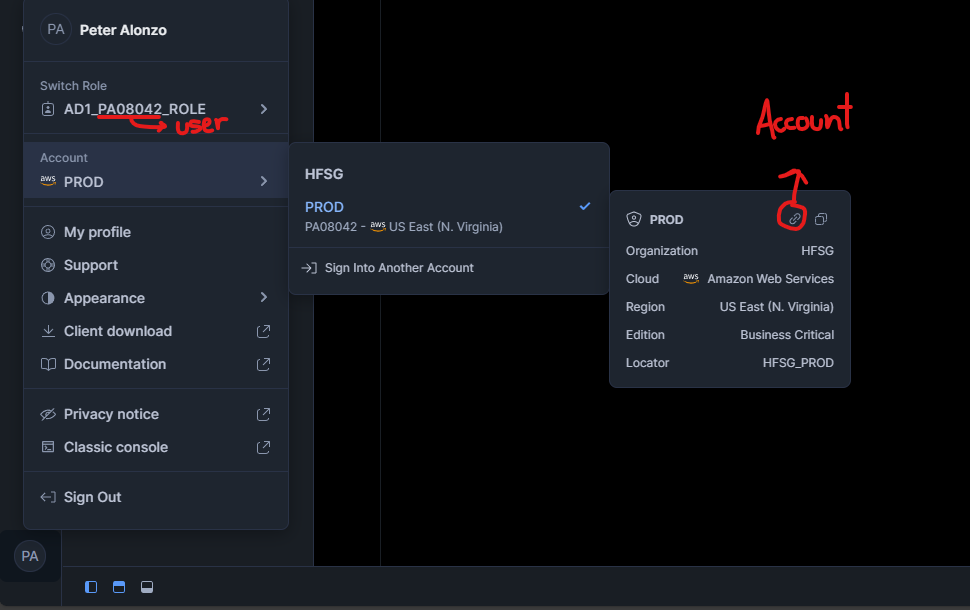
After fixing your date columns, we can now import the necessary libraries needed to use our private key and write the dataframe to Snowflake. The code below imports every method you will need for this step, and we also save our encryption password as a variable. Finally, the reset\_index line is there to ensure our dataframe is written properly.



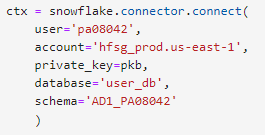
With the snowflake encryption key saved, you will need to load an encrypted version of your private key to use for the snowflake connector. You will need to specify the file path for the p8 file that holds your private key. This is the same path shown in your terminal when you generated the key pair.



The final step is creating a Snowflake connector with your encrypted private key and your Snowflake account. You can find the necessary information by hovering over your account in the bottom-left corner of Snowflake.

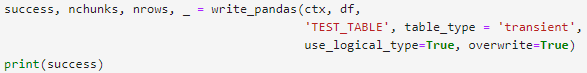


When you use the account URL, you can remove the “https://” and “.snowflakecomputing.com” when you put it into your connector variable. Assuming you are writing to your personal schema, your connector code will look something like this:



Finally, we can use the write\_pandas method to write your dataframe to the schema specified in your connector variable. The ‘TEST\_TABLE’ parameter will be your desired table name and you can choose to overwrite or not. For more information about the write\_pandas method, please visit:

[Python Connector API | Snowflake Documentation](https://docs.snowflake.com/en/developer-guide/python-connector/python-connector-api#write_pandas)



For any questions regarding this documentation or the pipeline program, please reach out to:

|  |  |
| --- | --- |
| Peter Alonzo (Associate Data Engineer) | [peter.alonzo@thehartford.com](mailto:peter.alonzo@thehartford.com) (Author) |
| William Fraher (Associate Software Engineer) | [william.fraher2@thehartford.com](mailto:william.fraher2@thehartford.com) |
| Kevin Bodie (Director of Data Engineering) | [kevin.bodie@thehartford.com](mailto:kevin.bodie@thehartford.com) |