In this assignment, you will demonstrate your understanding of PySpark and Airflow.

The first part of this assignment focuses on PySpark. In the first part of this assignment, you will be required to create PySpark Docker *containers* using the [Bitnami Spark Docker *image*](https://hub.docker.com/r/bitnami/spark)

*.* Next, you will be asked to analyze data about flight delays by using SQL *queries.*

The second part of this assignment focuses on Airflow. You will start by initializing Airflow Docker *containers.* Next, you will be required to define a simple DAG that uses the PythonOperator Airflow *Operator* to return the value of a *function.* Finally, you will run your code inside the Airflow UI to ensure your DAG works as expected*.*

Before beginning this assignment, review the submission instructions below to ensure that you collect the required screenshots as you progress through the assignment.

**To complete this assignment, follow these steps:**

**Part 1: PySpark**

1. Create a new folder called bitnami. Open a command prompt window and navigate to the bitnami folder. Run the following curl command to download the docker-compose.yml file:

| curl https://raw.githubusercontent.com/bitnami/containers/refs/heads/main/bitnami/spark/docker-compose.yml -o docker-compose.yml |
| --- |

1. Provide a screenshot showing that you correctly pulled the image and that the docker-compose.yaml file is present.
2. In a Terminal window, create *containers* by running the command below:  
   docker-compose up  
   Provide a screenshot of your Docker Desktop to show that you correctly pulled the *containers*.
3. Download the departuredelays.csv file on your machine. Perform a Docker copy command to copy this file to the bitnami\_spark\_1 *container*. Open the bitnami\_spark\_1 container to verify that you copied the file successfully. Provide a screenshot to show that you successfully copied the departuredelays.csv file to the bitnami\_spark\_1 *container.*
4. In the CLI window, type the correct command to open PySpark. Provide a screenshot to show that you successfully opened PySpark.
5. In the CLI window, type the correct command to import the correct *package* to start a PySpark session. Provide a screenshot to show that you successfully started a PySpark session.
6. In the CLI window, type the correct command to define a PySpark session named spark. Set the appName *argument* equal to Assignment19.3. Provide a screenshot to show that you successfully defined the spark PySpark session.
7. In the CLI window, define a variable, assignment19\_3\_data, to hold the path to the departuredelays.csv file. Provide a screenshot to show that you successfully defined the assignment19\_3\_data variable.
8. In the CLI window, define a *dataframe*, df, that contains all of the entries in the departuredelays.csv file. Provide a screenshot to show that you successfully defined the df *dataframe* that contains all of the entries in the departuredelays.csv file.
9. In the CLI window, on the df *dataframe* use the createOrReplaceTempView *method* to create a view of the *dataframe*. Name the view assignment19\_3\_table. Provide a screenshot to show that you successfully created a view of the assignment19\_3\_table *dataframe*.
10. In the CLI window, type an SQL *query* to select the first 15 flights from Philadelphia International Airport (PHL) to Dallas Fort Worth International Airport (DFW) that had a delay of greater than 150 minutes. Provide a screenshot to show that you selected the correct entries from your data. Your data should display the first 15 flights from PHL to DFW that had a delay of greater than 150 minutes.
11. In the CLI window, type an SQL *query* to select the first 10 flights that have a distance of less than 200 miles. Provide a screenshot to show that you selected the correct entries from your data. Your data should display the first 10 flights that have a distance of less than 200 miles. Ensure that the resulting table contains all of the columns in the original dataset.
12. In the CLI window, type an SQL *query* to select the first 10 flights that have a distance of greater than 600 miles. Provide a screenshot to show that you selected the correct entries from your data. Your data should display the first 10 flights that have a distance greater than 600 miles. Ensure that the resulting table contains all of the columns in the original dataset.

**Part 2: Airflow**

1. Create an empty folder called airflow\_assignment on your local machine. Navigate to the airflow\_assignment folder within a Terminal window and pull the Airflow file using the following command:

| curl 'https://airflow.apache.org/docs/apache-airflow/2.1.1/docker-compose.yaml' -o docker-compose.yaml |
| --- |

1. Provide a screenshot of your Terminal window response to show that you correctly pulled the Airflow file.
2. Open the generated docker-compose.yaml file using VS Code. Under "environment", set the AIRFLOW\_\_CORE\_\_LOAD\_EXAMPLES variable equal to false. Provide a screenshot to show the changed example value (AIRFLOW\_\_CORE\_\_LOAD\_EXAMPLES set to false).
3. In the Terminal window, run the docker-compose up command to start the Airflow Docker *container*. Open Docker Desktop and select the *container* to see all of the parts. Provide a screenshot to show that the Airflow Docker *containers* are running.
4. In your web browser, navigate to https://localhost:8080/. Log in to Airflow using the default username and password: “airflow”. Provide a screenshot of your browser window to show that you have successfully logged in to Airflow.
5. Open VS Code and navigate to the dags folder inside the airflow\_assignment folder. Inside this folder, create a new empty file called module19\_assignment.py. Provide a screenshot to show that you created the module19\_assignment.py file.
6. Copy the command below to import the required *libraries* for your DAG:  
   from datetime import timedelta, datetime  
   from airflow import DAG  
   from airflow.operators.python\_operator import PythonOperator  
   from airflow.utils.dates import days\_ago  
   Provide a screenshot to show that you correctly imported the required *libraries.*
7. In the module19\_assignment.py file, add the following lines of code to define your DAG:  
   default\_args = {  
    'owner': 'XXX',  
    'depends\_on\_past': False,  
    'start\_date': days\_ago(2),  
    'email': ['XXXX@YYY.com'],  
    'email\_on\_failure': False,  
    'email\_on\_retry': False,  
    'retries': 1,  
   }  
   Set the owner *argument* equal to your last name and replace the email field with your email address. Provide a screenshot to show that you set up your DAG correctly, including your last name and email address.
8. In the module19\_assignment.py file, define a Python *function*, square(), that takes one positional *argument*, x, and returns that number squared. Provide a screenshot to show that you defined the square() *function* correctly.
9. Copy the code below into the module19\_assignment.py file to define a DAG object:  
   dag = DAG(  
    'python\_square\_operator',  
    description = 'Squaring a number using Airflow',  
    schedule\_interval = “0 12 \* \* \*”,  
    Start\_date = datetime(2017,3,20), catchup = False)  
   Provide a screenshot to show that you correctly defined the DAG object.
10. In the module19\_assignment.py file define a *Task*, t1, that will return the square of the number 13. Declare this *Task* as a PythonOperator. Set the task\_id parameter equal to square. Set the python\_callable parameter equal to the Python *function* that you defined previously. Set the dag parameter to be equal to dag. Provide a screenshot to show that you defined the DAG *Task* correctly.
11. Navigate to https://localhost:8080/ to confirm that your DAG is configured correctly. Provide a screenshot of the Airflow UI to show that your DAG is configured correctly.
12. Start the DAG by toggling the switch to the left of its name. Select the DAG and open up the Graph View. Select the DAG again and open “logs”. Provide a screenshot of the log to show that the DAG ran successfully.

**Submission Instructions:**

Your submission for this assignment should be a Word document that includes the following screenshots, each labeled for the step that the screenshot represents:

**Part 1: PySpark**

1. Provide a screenshot showing that you correctly pulled the *image* and the docker-compose.yaml file is present.
2. Provide a screenshot of your Docker Desktop to show that you correctly pulled the *containers.*
3. Provide a screenshot to show that you successfully copied the departuredelays.csv file to the bitnami\_spark\_1 *container*.
4. Provide a screenshot to show that you successfully opened PySpark.
5. Provide a screenshot to show that you successfully started a PySpark session.
6. Provide a screenshot to show that you successfully defined the spark PySpark session.
7. Provide a screenshot to show that you successfully defined the assignment19\_3\_data variable.
8. Provide a screenshot to show that you successfully defined the df *dataframe* that contains all of the entries in the departuredelays.csv file.
9. Provide a screenshot to show that you successfully created a view of the assignment19\_3\_table *dataframe*.
10. Provide a screenshot to show that you selected the correct entries from your data. Your data should display the first 15 flights from PHL to DFW that had a delay of greater than 150 minutes.
11. Provide a screenshot to show that you selected the correct entries from your data. Your data should display the first 10 flights that have a distance of less than 200 miles and the resulting table should contain all of the columns in the original dataset.
12. Provide a screenshot to show that you selected the correct entries from your data. Your data should display the first 10 flights that have a distance greater than 600 miles and the resulting table should contain all of the columns in the original dataset.

**Part 2: Airflow**

1. Provide a screenshot of your Terminal window response to show that you correctly pulled the Airflow file.
2. Provide a screenshot to show the changed example value ( AIRFLOW\_\_CORE\_\_LOAD\_EXAMPLES set to false).
3. Provide a screenshot to show that the Airflow Docker *containers* are running.
4. Provide a screenshot of your browser window to show that you have successfully logged in to Airflow.
5. Provide a screenshot to show that you created the module19\_assignment.py file.
6. Provide a screenshot to show that you correctly imported the required *libraries*.
7. Provide a screenshot to show that you set up your DAG correctly, including your last name and email address.
8. Provide a screenshot to show that you defined the square() *function* correctly.
9. Provide a screenshot to show that you correctly defined the DAG object.
10. Provide a screenshot to show that you defined the DAG *Task* correctly.
11. Provide a screenshot of the Airflow UI to show that your DAG is configured correctly.
12. Provide a screenshot of the log to show that the DAG ran successfully.