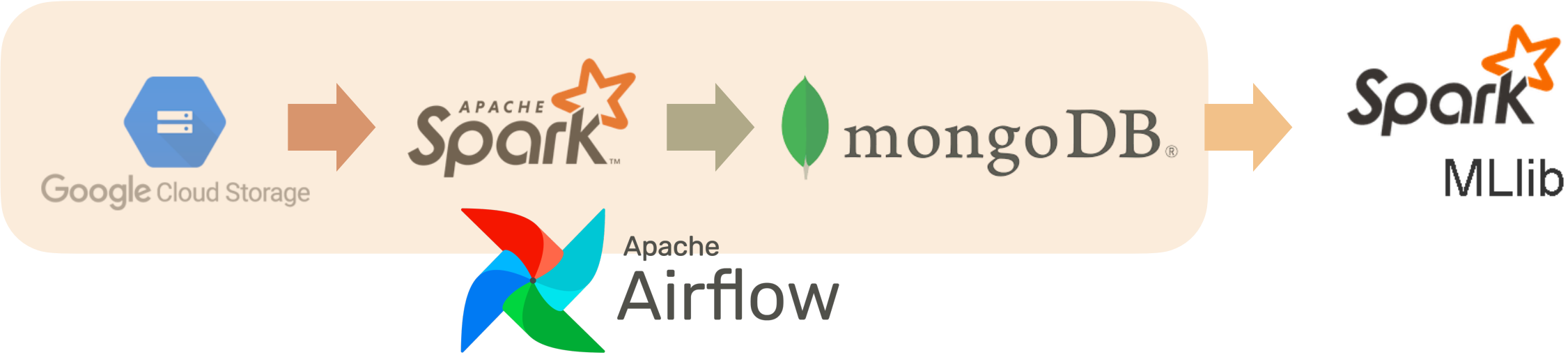
MSDS 697
Group Project Task2

DIANE WOODBRIDGE, PH.D

Overview

1. Goal : Building an Automated Scalable ML Pipeline



Task 2 Overview

1. Import data from GCS into MongoDB Atlas (Should have replicated shards).
   - Make sure to automate the following processes using Apache Airflow.
     - Fetching data using API (or crawling) and store to GCS
     - Creating an aggregate using data in GCS and storing to MongoDB
2. Query data.

Step 1. Data Collection in Google Cloud Storage

1. Repeat the steps that we discussed in MSDS694 [Videos][Docs]
    1. Create a Google Cloud Project and Storage Buckets
    2. Create/Download Service Account Key File
    3. Write your data acquisition code and include to Airflow
    - **Your Airflow tasks should download data to GS Buckets, not your local machine.**
    - Installing google-cloud-storage package will help.

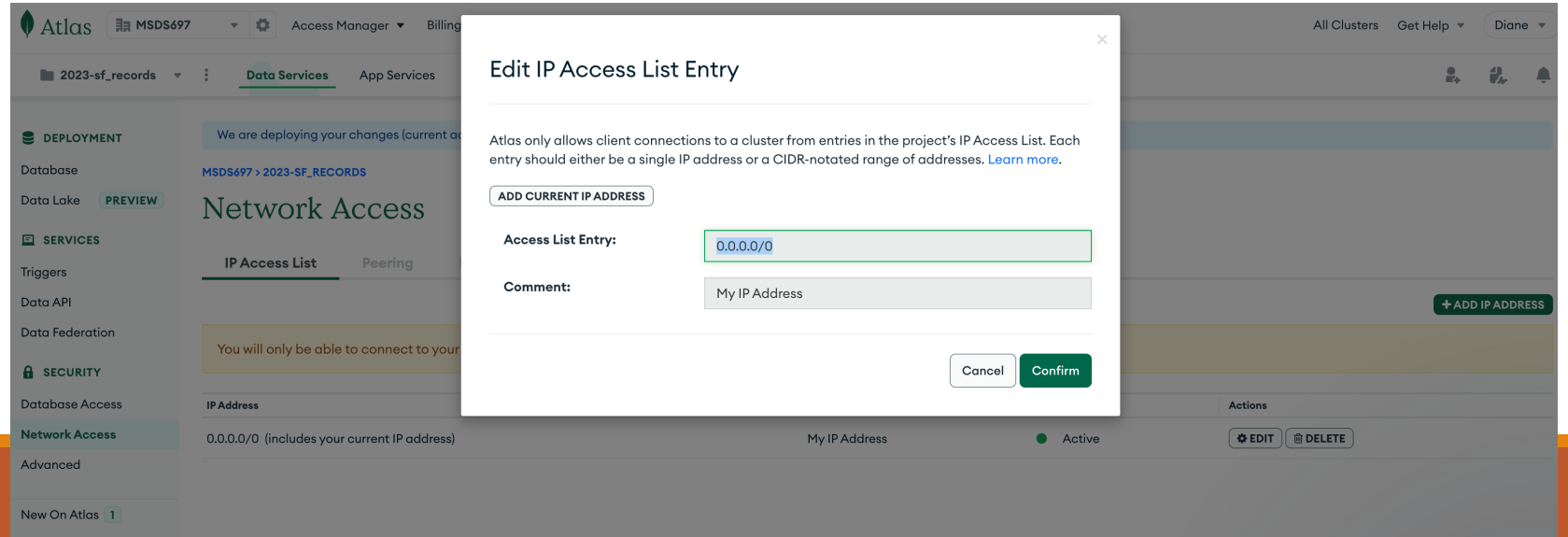
Step 2. Create MongoDB Atlas Account

1. Create an account via https://www.mongodb.com/cloud/atlas
   - It might require a credit card info. In that case, make sure to set up a **billing alert** to make sure it won't charge.
   - Each team will receive a MongoDB credit ($400) per team via Canvas.
     - Let me know if you need more credits.

Step 2. Create MongoDB Atlas Account

Todo

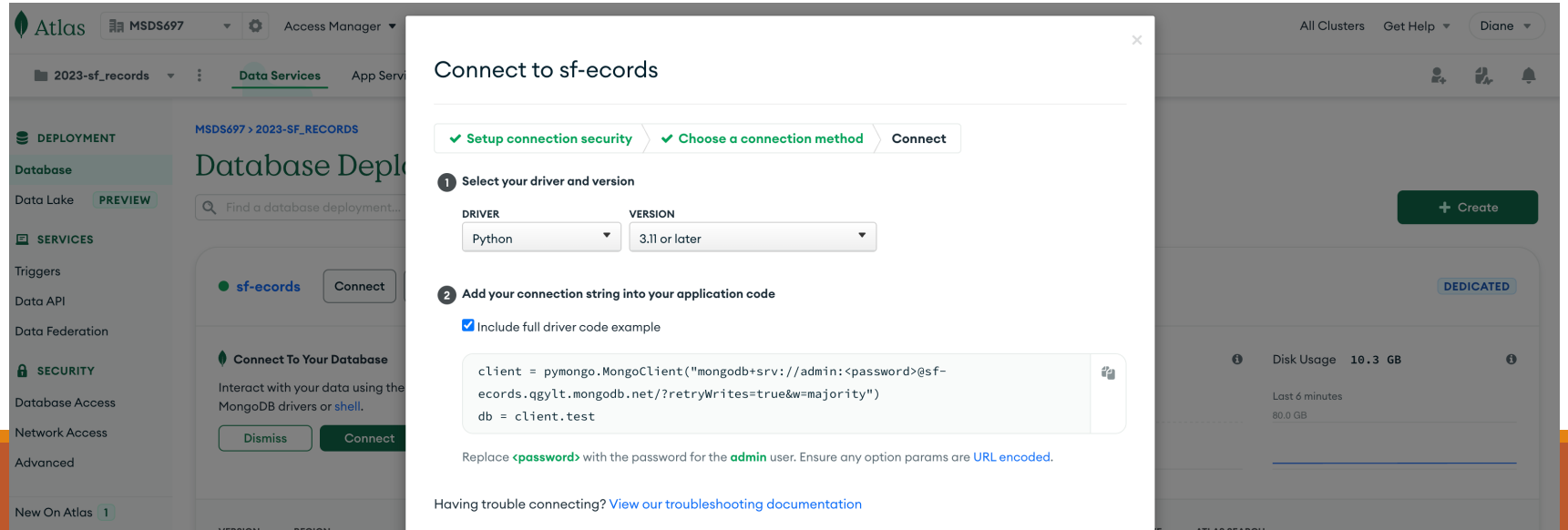
1. Create a project .
2. Create a cluster with **sharding** available for scalability.
3. Assign IDs and PWDs for your teammates.
4. Allow external IP addresses.
   - Security => Network Access => Add 0.0.0.0/0 (Everywhere) on the IP Access List.



Step 2. Create MongoDB Atlas Account

Todo

5. Copy the IP address of your cluster.
   - Connect => “Connect your application” => Python
     - The IP address located after @ and before /
       - Ex. mongodb+srv://admin:<password>@**msds697-cluster.qgylt.mongodb.net**/?retryWrites=true&w=majority



Step 3. Write your Spark script for Creating/Inserting Aggregates

Spark is powerful for processing a large amount of data.

1. In this task, write a pyspark code to preprocess data/create aggregates and store them to MongoDB Atlas.
   - Feel free to use SparkSQL.
     - The example that I shared includes (retreive_law_enforcement_data() in aggregates_to_mongo.py) include reading .csv files.
     - You can also load various data types using the Spark DataFrame interface [Link].
   - Note: Spark has a MongoDB connector as well. However, I used an action (.collect()) to store the data since I haven't covered it.

Step 4. Airflow

1. Add data cleansing and aggregation tasks using SparkOperator on Airflow
   - You can use BashOperator, but **SparkOperator** would be useful/more convenient.
     - `$ pip install apache-airflow-providers-apache-spark`
       - SparkSubmitOperator(*, application='', conf=None, conn_id='spark_default', files=None, py_files=None, archives=None, driver_class_path=None, jars=None, java_class=None, packages=None, exclude_packages=None, repositories=None, total_executor_cores=None, executor_cores=None, executor_memory=None, driver_memory=None, keytab=None, principal=None, proxy_user=None, num_executors=None, status_poll_interval=1, application_args=None, env_vars=None, verbose=False, spark_binary=None, **kwargs)

https://airflow.apache.org/docs/apache-airflow-providers-apache-spark/
https://airflow.apache.org/docs/apache-airflow-providers-apache-spark/stable/_api/airflow/providers/apache/spark/operators/spark_submit/index.html

Step 4. Airflow

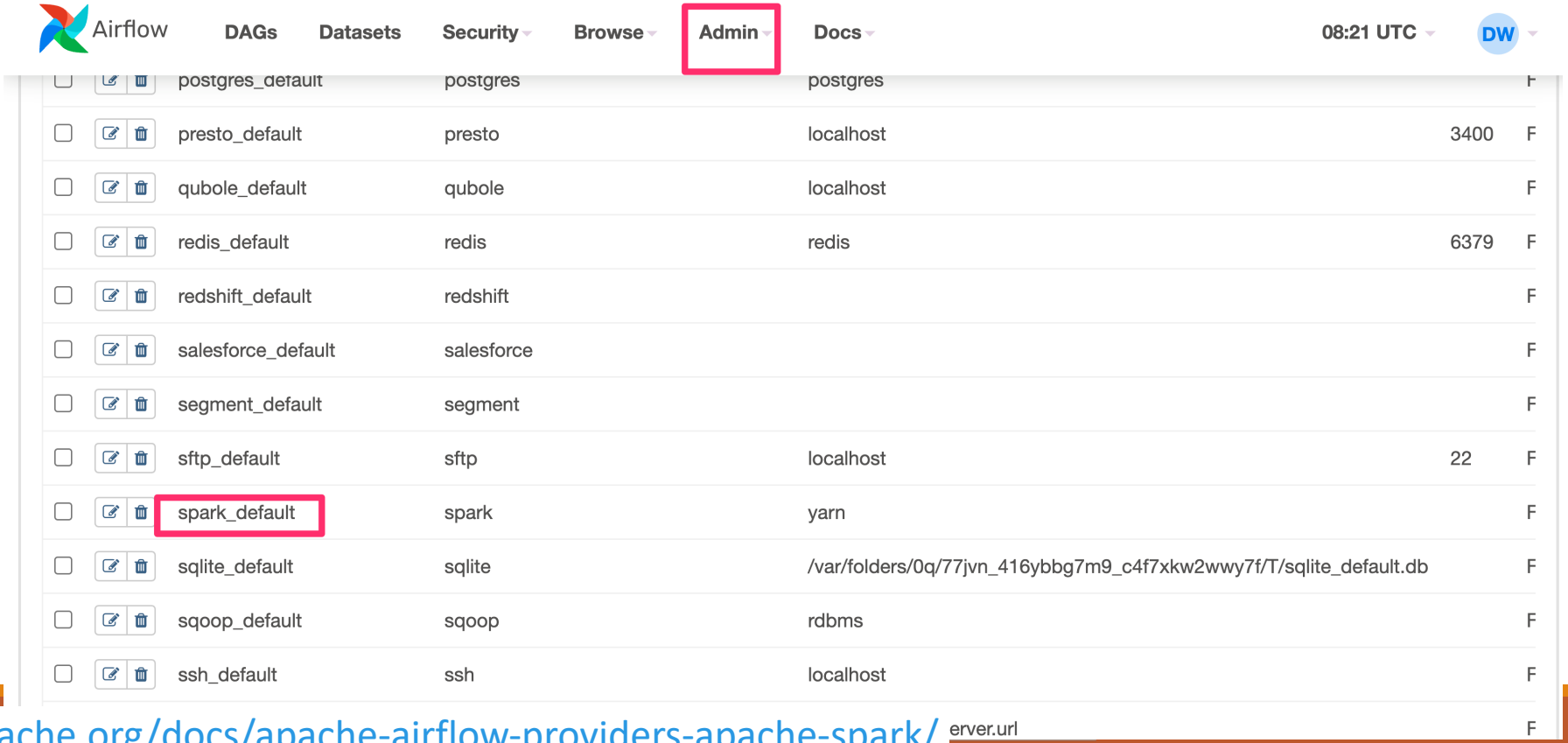
1. Add data cleansing and aggregation tasks using SparkOperator on Airflow
   - You can use BashOperator, but **SparkOperator** would be useful/more convenient.
     - Example.

```
create_insert_aggregate = SparkSubmitOperator(
    task_id="aggregate_creation",
    packages="com.google.cloud.bigdataoss:gcs-connector:hadoop2-1.9.17,org.mongodb.spark:mongo-spark-connector_2.12:3.0.1",
    exclude_packages="javax.jms:jms,com.sun.jdmk:jmxtools,com.sun.jmx:jmxri",
    conf={"spark.driver.userClassPathFirst":True,
          "spark.executor.userClassPathFirst":True,
          },
    verbose=True,
    application='aggregates_to_mongo.py'
)
```

https://airflow.apache.org/docs/apache-airflow-providers-apache-spark/
https://airflow.apache.org/docs/apache-airflow-providers-apache-spark/stable/_api/airflow/providers/apache/spark/operators/spark_submit/index.html

Step 4. Airflow

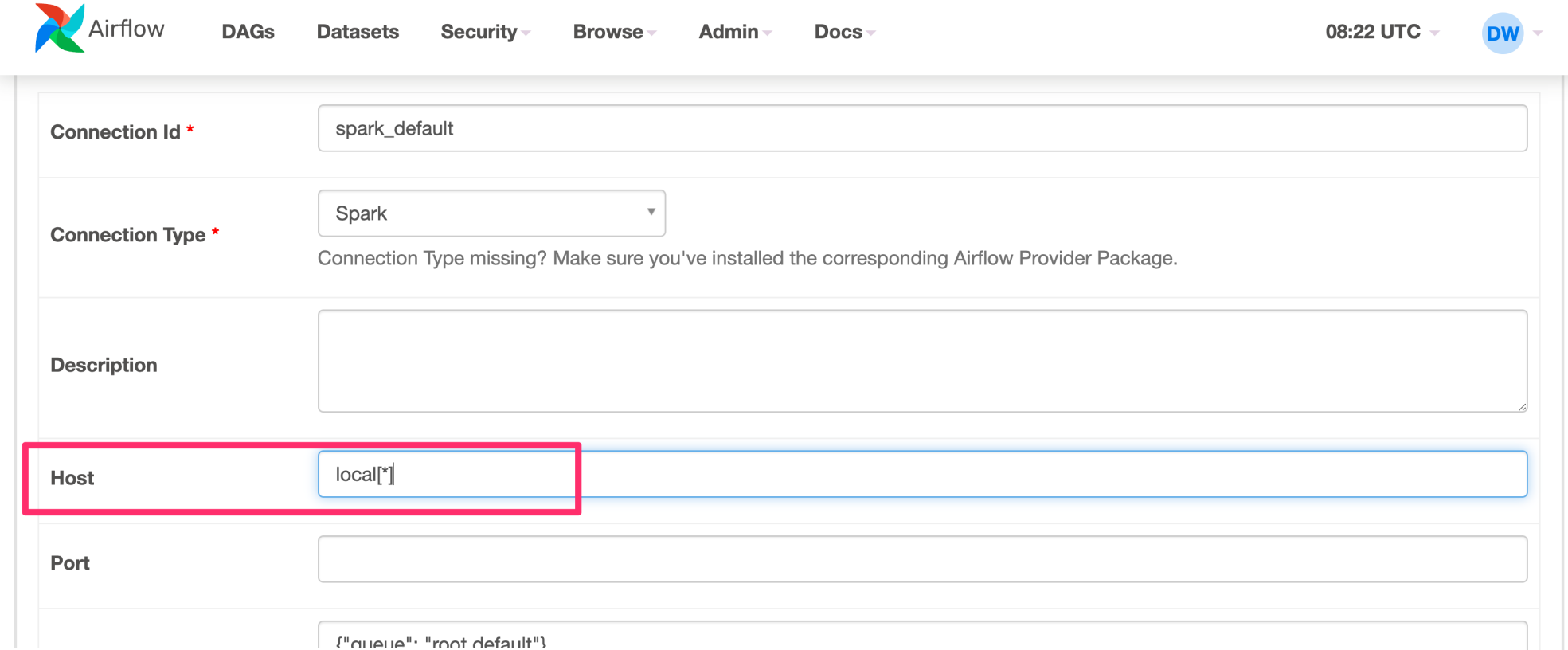
1. Update spark_default host to be local[*] or local
   - Go to Admin => spark_default => Host



https://airflow.apache.org/docs/apache-airflow-providers-apache-spark/

Step 4. Airflow

1. Update spark_default host to be local[*] or local
   - Go to Admin => spark_default => Host



https://airflow.apache.org/docs/apache-airflow-providers-apache-spark/