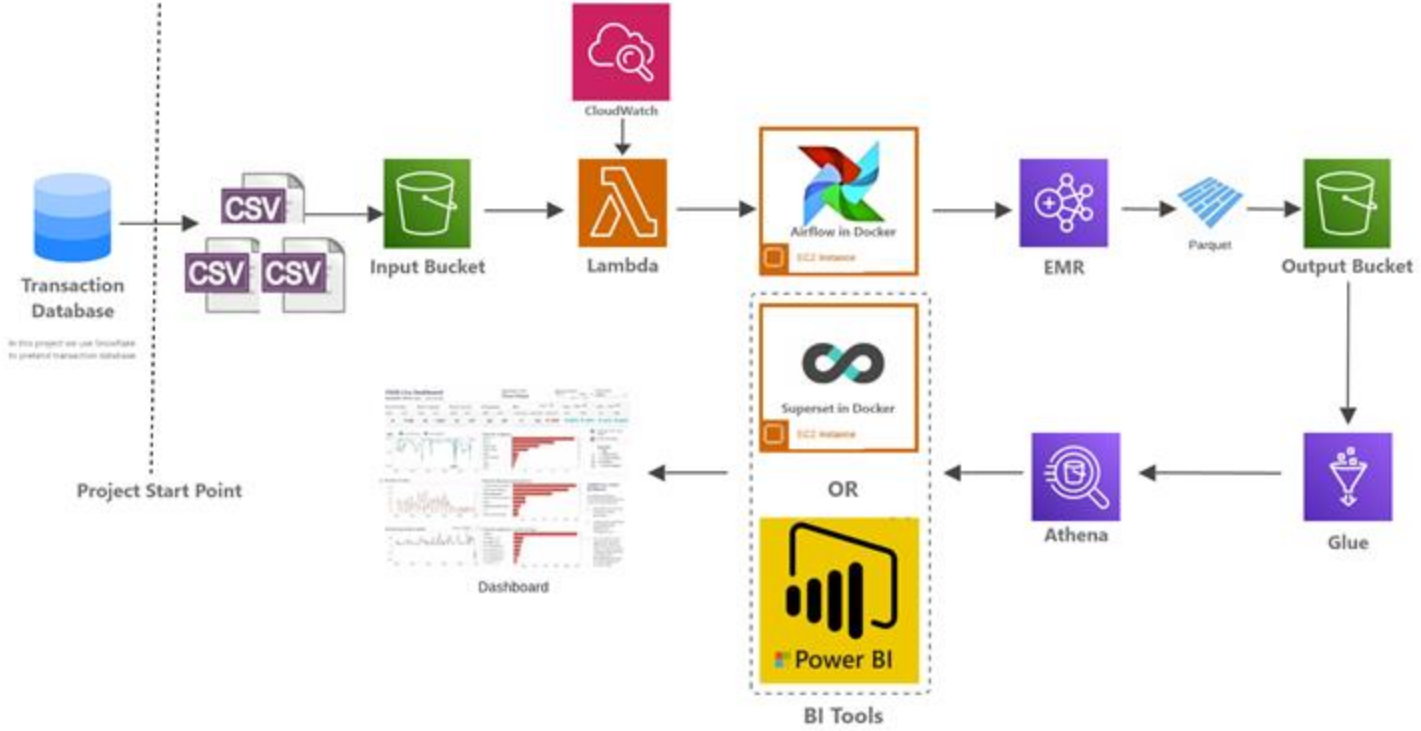
# Data Pipeline

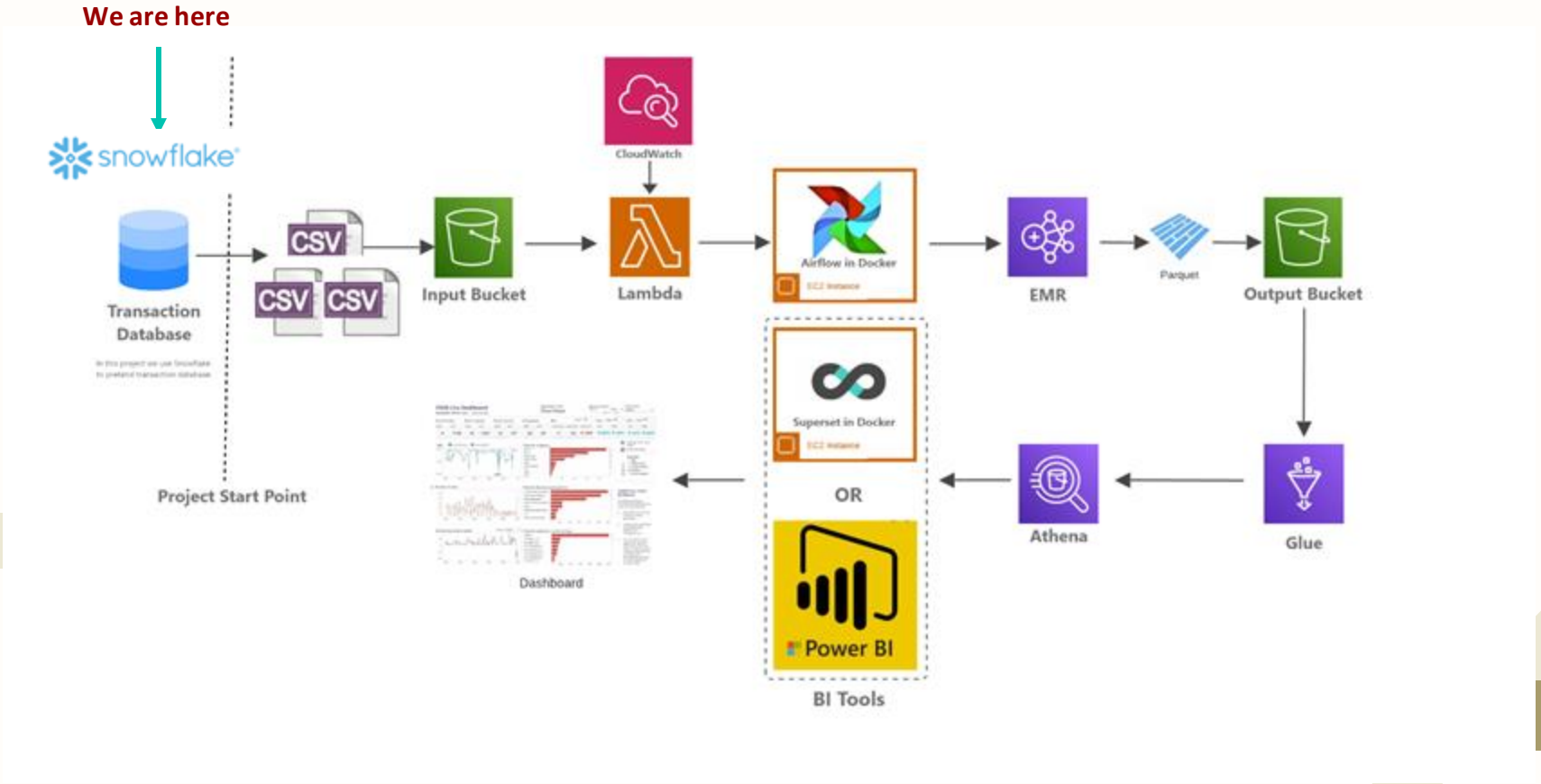
Bootcamp Student Edition
By Shaheer Khan

# Architecture 1



# Table of Contents

# Architecture 1



# Transactional Database: Snowflake

## Tasks

The following tasks were completed in snowflake to set up data extraction from the makeshift transactional database (Snowflake):

**Part A: Setting up**

1. Set up a warehouse
2. Create and use databases
3. Create and use schema
4. Create and use the tables for sales, product, store, calendar, inventory

**Part B: Creating Integrations with S3 for data dump**

1. Create a staging layer
2. Grant the appropriate permissions and Roles

**Part C: Creating Stored Procedure using Python to automate data dumping to S3 from Snowflake**

1. Show demo of the script

# Architecture 1

# Data Lake: S3 Bucket

## Tasks

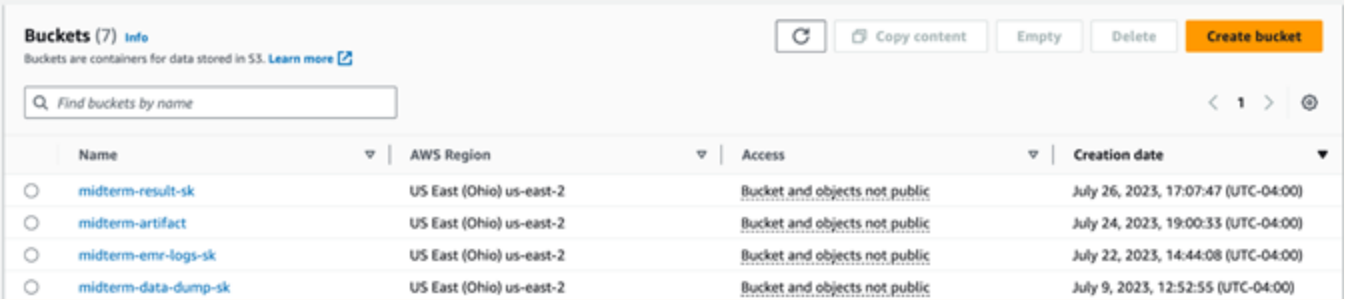
The following buckets were created:

- midterm-result-sk: to store output files from emr transformation
- midterm-artifact: to store the pyspark code
- midterm-emr-logs-sk: to store logs from the emr cluster
- midterm-data-dump-sk: to store csvs from snowflake (our transactional database/OLTP)

**Tip**:

Don't forget to store the latest pyspark code in the midterm-artifact bucket using the CLI command in terminal

aws s3 cp transformation.py s3://midterm-artifact



# Architecture 1

# ~~Cloudwatch~~ + Lambda

## Tasks

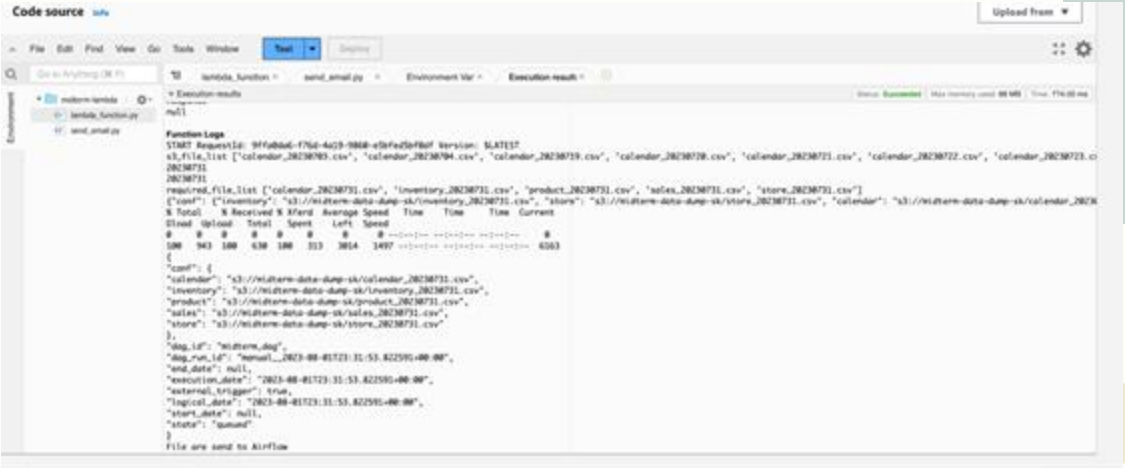
### Part A: Check Files are ready for EMR

1. Scan S3 bucket to check if all ‘today’s’ files are ready, when all are ready, send a signal to airflow to start EMR;

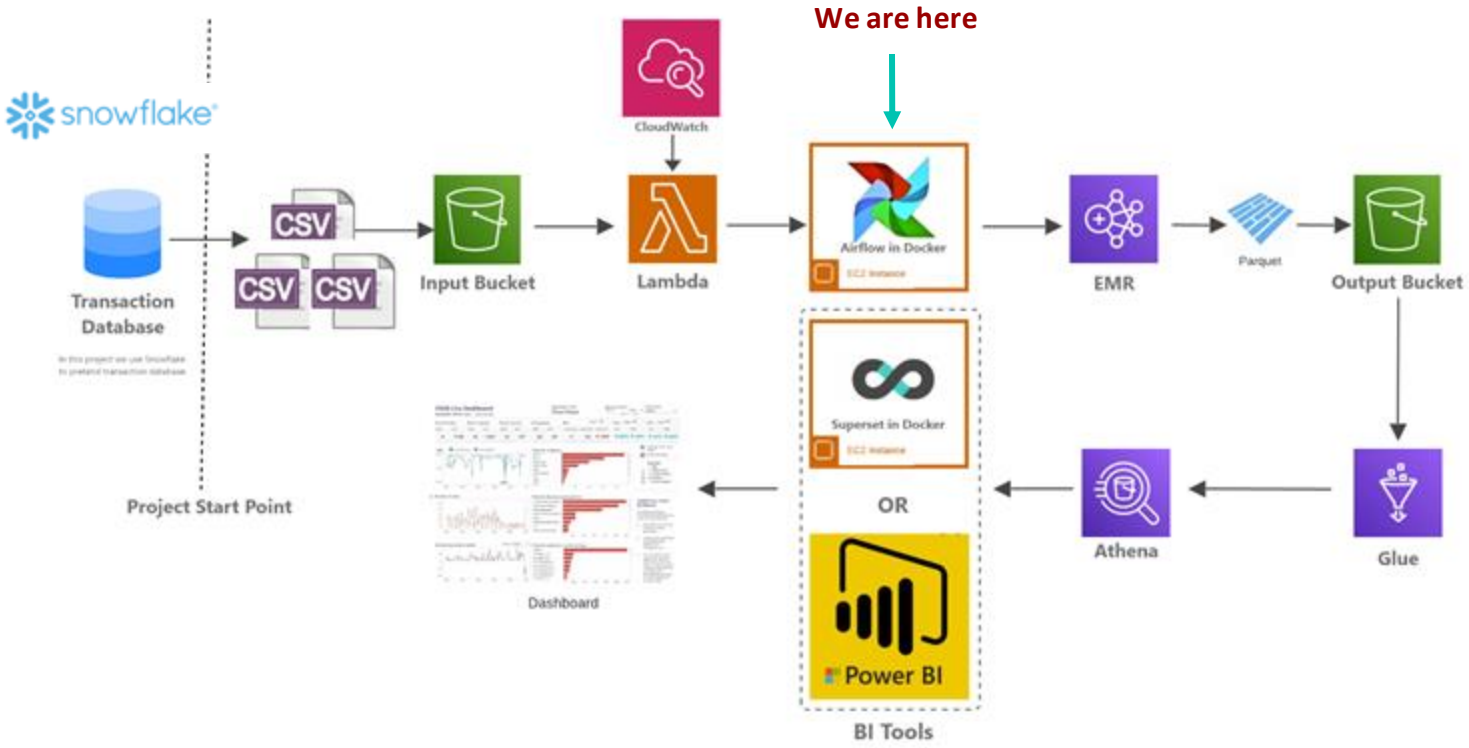
### Part B: Email Alert

1. If files are not ready, send an email to inform you that today’s files are not ready

## Output



# Architecture 1



# Airflow

## Tasks

After Lambda sends a signal to Airflow, the Airflow will unpack the data from Lambda as the parameter for EMR and store in xcoms.

The following tasks were done in airflow:

1. Once signal from Lambda is received, then automatically start the EMR cluster
2. Then execute the steps in EMR to transform the data
3. Then terminate the EMR cluster once the transformation is done


## Problems/Things Learned

I ran into the following problems:

- Debugging Errors in the dags (show demo)
- Permission Errors
- Creating AWS connections in Airflow
- Automating cluster creation and termination
- Printing output of tasks in airflow (show demo)
- Use this command in cloud shell to get cluster details to help automate the creation of cluster

  aws emr describe-cluster --cluster-id j-1K48XXXXXXHCB

# Architecture 1

# EMR

## Task

The following tasks were carried out using AWS EMR:

1. Read data from S3 using xcom values ingested using Spark Steps arguments (show demo)
2. *TO BE CONTINUED:* Do data transformation process to generate a dataframe to meet the business requirement
3. *TO BE CONTINUED:* After transformation, save the final dataframe as a parquet file to a new S3 output bucket. Copy the files of store, product and calendar (previously dumped to S3 bucket) to the new output S3 bucket. So the new parquet file together with the store, product and calendar files in the output folder will be used for later data analysis with Athena

## Problems/Things Learned

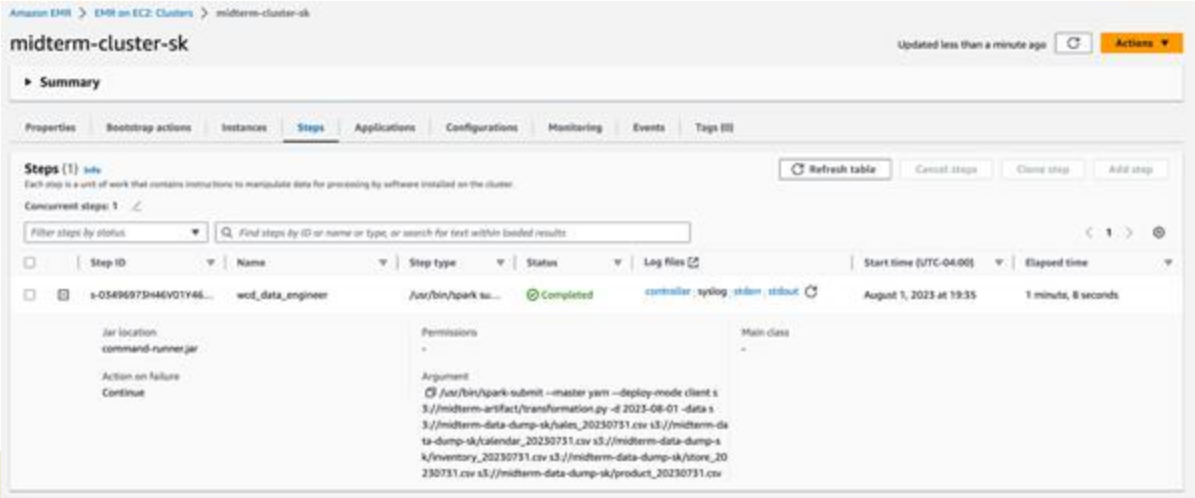
- Using argparse library to read in Airflow xcoms (show demo)
- Debugging roles and permissions for EC2 and S3 and EMR
- Running Spark Jobs locally instead of an EMR to test code (show demo)
- Doing transformation work on Databricks for ease of use

## Next Steps:

- Get a new data set and do the transformations
- Save the transformed data as a parquet file
- Use the same infrastructure and refine pyspark code

# EMR

## Snippets of Output




```
['s3://midterm-data-dump-sk/sales_20230731.csv', 's3://midterm-data-dump-sk/calendar_20230731.csv', 's3://midterm-data-dump-sk/inventory_20230731.csv', 's3://midterm-data-dump-sk/store_20230731.csv', 's3://midterm-data-
dump-sk/product_20230731.csv']
data[0] s3://midterm-data-dump-sk/sales_20230731.csv
date 2023-08-01
date_str 20230801
+--------+--------+---------+----------+----------+---------+-----------+---------+--------+----------+----------+---------+
|TRANS_ID|PROD_KEY|STORE_KEY|  TRANS_DT|TRANS_TIME|SALES_QTY|SALES_PRICE|SALES_AMT|DISCOUNT|SALES_COST|SALES_MGRN|SHIP_COST|
+--------+--------+---------+----------+----------+---------+-----------+---------+--------+----------+----------+---------+
|  244054|  455222|     8103|2020-10-09|        12|    25.00|      37.94|   721.06|    0.10|    610.39|    338.01|     5.08|
|  244056|  637817|     8103|2020-06-04|        16|     2.40|     999.99|  2423.98|    0.00|   4819.97|  -1820.00|    13.99|
|  244058|  492902|     8103|2022-10-25|        17|    30.00|      14.03|   356.94|    0.08|    569.08|   -148.26|     9.37|
|  244060|  612619|     8103|2022-01-10|        18|    13.60|     107.53|  1782.68|    0.08|   1547.29|    280.64|     5.81|
|  244062| 1039077|     8103|2020-05-25|        18|    34.00|      27.18|   622.89|    0.10|    719.53|    204.49|     8.23|
+--------+--------+---------+----------+----------+---------+-----------+---------+--------+----------+----------+---------+
```

# Athena and Glue

## Explore fixing schema in Glue to get better insights

Input format
org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat

Output format
org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat

Serde serialization lib
org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe

Schema | Partitions | Indexes

**Schema** (13)
View and manage the table schema.

Edit schema as JSON | Edit schema

Filter schemas

| # | Column name | Data type | Partition key | Comment |
|---|---|---|---|---|
| 1 | trans_id | string | - | - |
| 2 | prod_key | string | - | - |
| 3 | store_key | string | - | - |
| 4 | trans_dt | string | - | - |
| 5 | trans_time | string | - | - |
| 6 | sales_qty | string | - | - |
| 7 | sales_price | string | - | - |
| 8 | sales_amt | string | - | - |
| 9 | discount | string | - | - |
| 10 | sales_cost | string | - | - |
| 11 | sales_mgrn | string | - | - |
| 12 | ship_cost | string | - | - |
| 13 | date | string | Partition (0) | - |

# BI Tools (Superset)

## Coming Soon