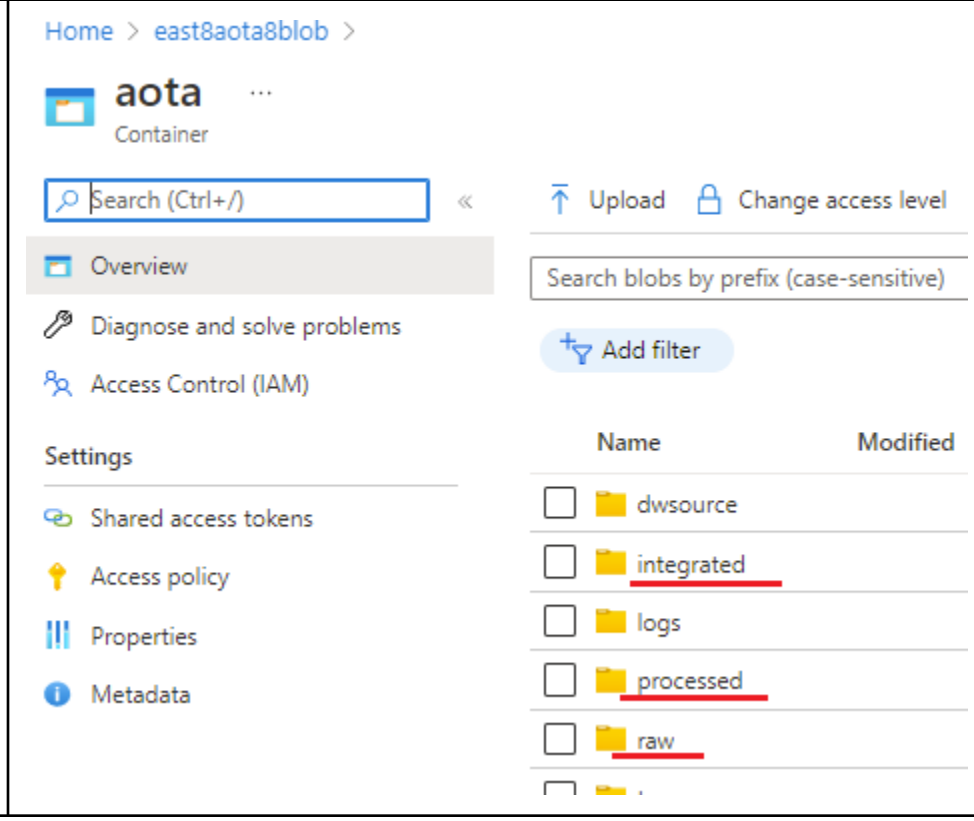
1. Deployment of downloading, ETL and Integraiton process

System information

Spark Cluster	HDInsight
Cluster OS	Ubuntu
Cloud Storage	Azure blob

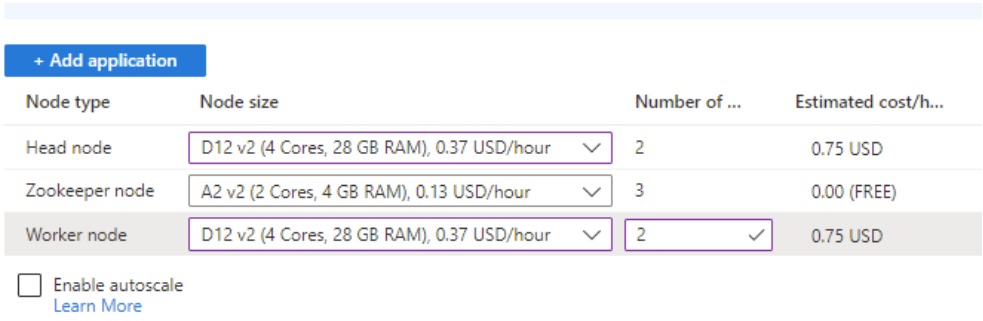
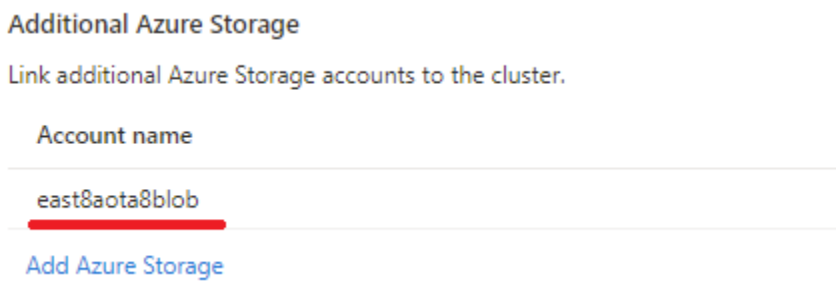
1. Create Azure blob and container

Azure Blob

	Storage account: east8aota8blob Container: aota
	

2. Create HDinsight cluster

Here are some details about setting up the HDinsight cluster

Cluster Configuration	
Add additional storage	When creating the cluster, we add an Azure storage account, 'east8aota8blob', to be the additional storage. In this way, our Spark program can freely access it through some settings. 
blobfuse	On Cluster, we installed blobfuse (on linux) which allows to mount Blob storage ('east8aota8blob'), as a local file system. In such a way, our python script can download the source data and directly save to it.

3. Deploy code HDinsight
   1) SSH to HDinsight Cluster
   2) Commands

---------- ssh----------

```
ssh sshuser@cluster-0531-ssh.azurehdinsight.net

---------- install moudle -==========

sudo pip3 install pyspark


------------------------------------ set up folder for aota
---------------------------------------------------------------------------
sudo mkdir /app
sudo chown sshuser /app
mkdir /app/aota


sudo mkdir /data
sudo chown sshuser /data


#create folder:
sudo mkdir /data/aota
sudo mkdir /data/aota/raw
sudo mkdir /data/aota/logs
sudo mkdir /data/aota/processed
sudo mkdir /data/aota/raw

sudo mkdir /data/aota/integrated

#change owner
sudo chown sshuser /data/aota/logs --recursive

sudo chown sshuser /data  --recursive

sudo chown sshuser /app  --recursive

------------------- how to upload code-----------------------------------------------------

upload to hdfs:/tmp of primary storage

------------------  download code --------------
hdfs dfs -ls /
hdfs dfs -get  /tmp  /app/aota


------------ after   copy logs  to hdfs --------------------
 hdfs dfs -put   /data/aota/logs  /tmp/logs

-------------------------------------------------
---------   export --------------
export SPARK_HOME=/usr/lib/spark
export  sshuser_HOME=/usr/lib/sshuser
```

```
#sudo nano  $SPARK_HOME/conf/spark-defaults.conf

--- add  user to root  group ??? -----------------

sudo  usermod -G root  sshuser
```

4. Running

Process	Timing
Download	**Start**: **2022-05-29_17:43:29** **End:** **2022-05-29_17:43:59** **Duration**: **30 seconds**
ETL (cleanse/TRansform)	**Start**: **2022-05-29_17:44:40** **End:** **2022-05-29_18:14:42** **Duration**: **30 minutes**
Integration process	**Start**: **2022-05-31_19:44:00** **End:** **2022-05-31_19:55:46** **Duration**: **12 minutes**

2. Deployment of DW-ETL

Preparation:

1. Convert ‘integration layer’ parquet files to CSV files
2. Upload CSV files to a S3 bucket using AWS cli command.

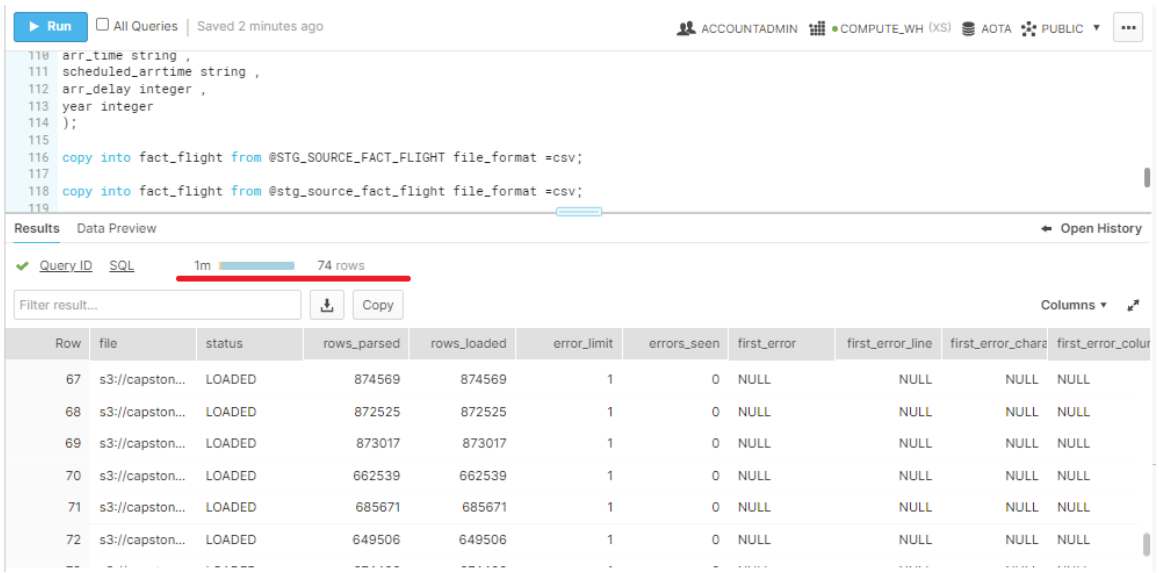
Timing: it took 30 minutes to load csv files to S3 bucket

Timing	Start: 22:09:48.40 End: 21:39:21.61

	Duration: 30 minutes

Load data to Snowflake,

1) Create database and tables in Snowflakes
2) Create stage
3) Using ‘copy’ command to load data from S3 bucket to Snowflake tables

Row count loaded	59 millions
Time	1 minute
Screen shot	

Row	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_line	first_error_chara	first_error_colu
67	s3://capston...	LOADED	874569	874569	1	0	NULL	NULL	NULL	NULL
68	s3://capston...	LOADED	872525	872525	1	0	NULL	NULL	NULL	NULL
69	s3://capston...	LOADED	873017	873017	1	0	NULL	NULL	NULL	NULL
70	s3://capston...	LOADED	662539	662539	1	0	NULL	NULL	NULL	NULL
71	s3://capston...	LOADED	685671	685671	1	0	NULL	NULL	NULL	NULL
72	s3://capston...	LOADED	649506	649506	1	0	NULL	NULL	NULL	NULL