New Project

```
sc
import pyspark
from pyspark import SparkContext, SparkConf
from pyspark.sql import SparkSession
sc=SparkContext.getOrCreate()
from pyspark.sql.functions import split
from pyspark.sql import functions as F
from pyspark.sql.functions import when
from pyspark.sql.functions import count
from pyspark.sql.functions import desc
```

```
def Domain_name(df1):
    # Separating domain name from email address into new column in df2
    df2=df1.withColumn('Domain_name_temp', split(df1['email_address_non_pii'], '@').getItem(1))
    # count of public uuid and groupby through Domain Name
    df3=df2.groupBy("Domain_name_temp").agg(count("public_uuid").alias("Count")).sort(desc("Count"))
    # Creating a list from Dataframe (df3) with domain names
    list1=df3.select(df3.Domain_name_temp).rdd.flatMap(lambda x: x).collect()
    # Creating another list with top 10 domain names
    list2=list1[0:10]
    print(list2)
    # Creating new column which contains domain name in top 10 otherwise NULL
    df4=df2.withColumn("Domain_Name", F.when((df2.Domain_name_temp).isin(list2),df2.Domain_name_temp).otherwise("NULL"))
    # Deleting Domain name temp (temporary) column
    df4=df4.drop(df4.Domain_name_temp)
    return df4

# Reading file from S3
df1 = spark.read.parquet('s3://ha-prod-analytics-datalake-eagle-edw-goldeneye-us-east-1/tier2_dap_marketing_email/ocelot/userprimary/date=20220903/hour=12/')
# Calling Domain name and assigning into df DataFrame
df = Domain_name(df1)
df.show(2,0)
```

```
[u'gmail.com', u'hotmail.com', u'yahoo.com', u'aol.com', u'comcast.net', u'icloud.com', u'hotmail.co.uk', u'outlook.com', u'hotmail.fr', u'orange.fr']
+-----------------------------------+---------------------------------------------------+---------------------------------------------------+-----------------------------------------
--------+-------------+----------------------+---------------------------+----------------+-----+----+---------+-----------------------------------------------------
---------------------------------------------------+---------------+-------------------+--------------------+-----------------+----------------+------------------+--------------------------------+------
-----------------------------+---------------------------------------+---------------------------------+-------------------------------+-----------------------------+--------------------------+-----------------
-------------+--------------------------------+-----------------------------------+-----------------------------+-----------------------------+-------------------------------+-------------------------------+---------
------------------------+---------------------------+----------------------------+------------------------------+------------+-----------------------------------------------------+---------
-----------------+-------------------------------+-------------------------------------+---------------------------------+-----------+------------------+---------------------+----------------------+---
--------------------+------------------------------+-------------------------+--------------------------+-----------------------------+---------------------------------------+--------
-------------------------+-------------------------------------+-----------------------------+------------------------------------+-----------------------------------+---------------+---
---------------+------------------------+------------------------+------------------------+-------------------------------+------------------------------+----------------------+------
------------------------+-------------------------------+------------------------------+-------------------------------+---------------------------------+-------------------------------+
---------------+----------------+---------------+---------------+---------------+-----------------+---------------+--------------------+--------------------+---------------+----------------------+---------
------------+-------------------+---------------------------------------------------+-------------------------------+---------------------------------+------------------------------------+---
-------------------------------------------+---------------------------+-----------------------------+--------------------------+-----------------------------------------------------+------
--------------------+----------------------------+----------------------------+-------------------------------------+-----------------------------------------+---------------+---------------
-------------------------+-------------------------------------------+-------------------------------------+----------------------------------------+---------------------+--------------
------+--------------------+--------------------+--------------------+--------------------+------------------------+------------------------+------------------------+------------------------+-------------
```

```
# df.write.parquet('s3://ha-prod-omnidata-us-east-1/marketing/email/ocelot/temp/data_products/mock_training/output/{your-Name}/user_primary')
# write.mode(SaveMode.Overwrite).parquet
# write.mode(SaveMode.Append).parquet
# write.mode(SaveMode.Ignore).parquet
# write.mode(SaveMode.ErrorIfExists).parquet
```

```
# MODULE-4 VALIDATION
```

```
# Count input data
df1.count()
```

```
106940846
```

```
# count of public uuid and groupby through Domain Name
df.groupBy("Domain_Name").agg(count("public_uuid").alias("Count")).sort(desc("Count")).show()
```

```
+-------------+--------+
|  Domain_Name|   Count|
+-------------+--------+
|    gmail.com|40577898|
|         NULL|34942591|
|  hotmail.com|11283914|
|    yahoo.com|10338685|
|      aol.com| 3032822|
|  comcast.net| 1500085|
|   icloud.com| 1243959|
|hotmail.co.uk| 1022080|
|  outlook.com| 1011293|
|   hotmail.fr|  998037|
|    orange.fr|  989482|
+-------------+--------+
```

```
# count output data
df.count()
```

```
106940846
```