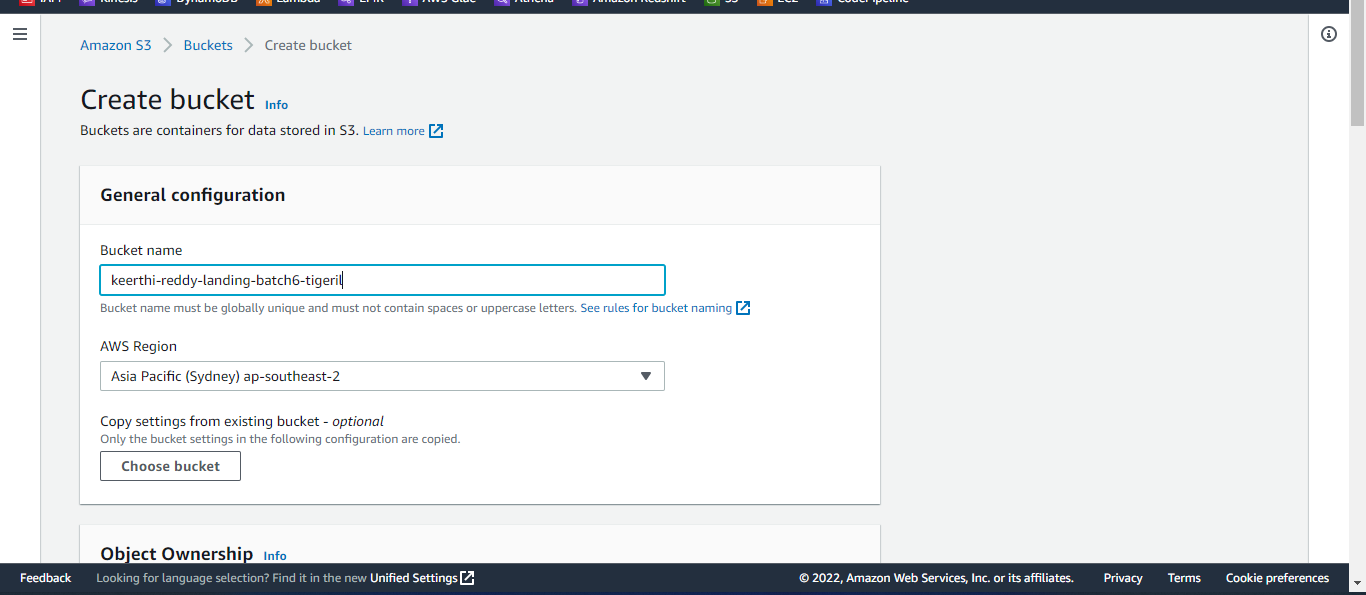
**Goal**: Create **S3** buckets for **Data Lake**

You need to **perform** the following **steps**:

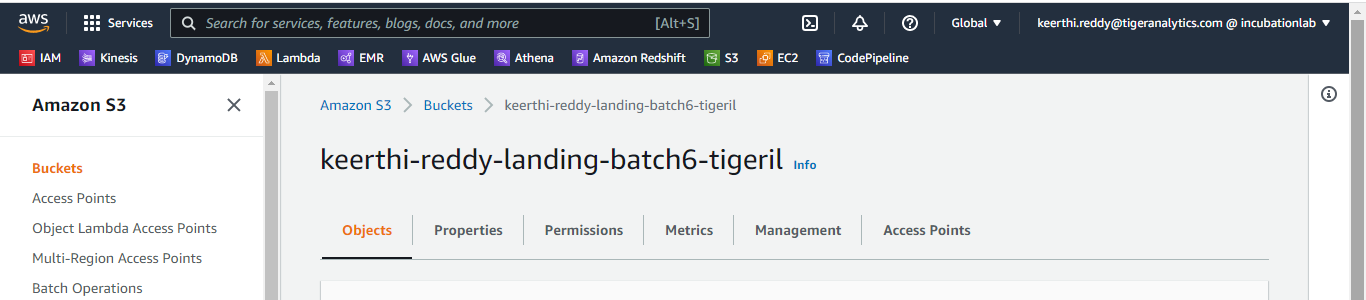
* Create three **S3** buckets for landing zone, raw zone, and staging zone.

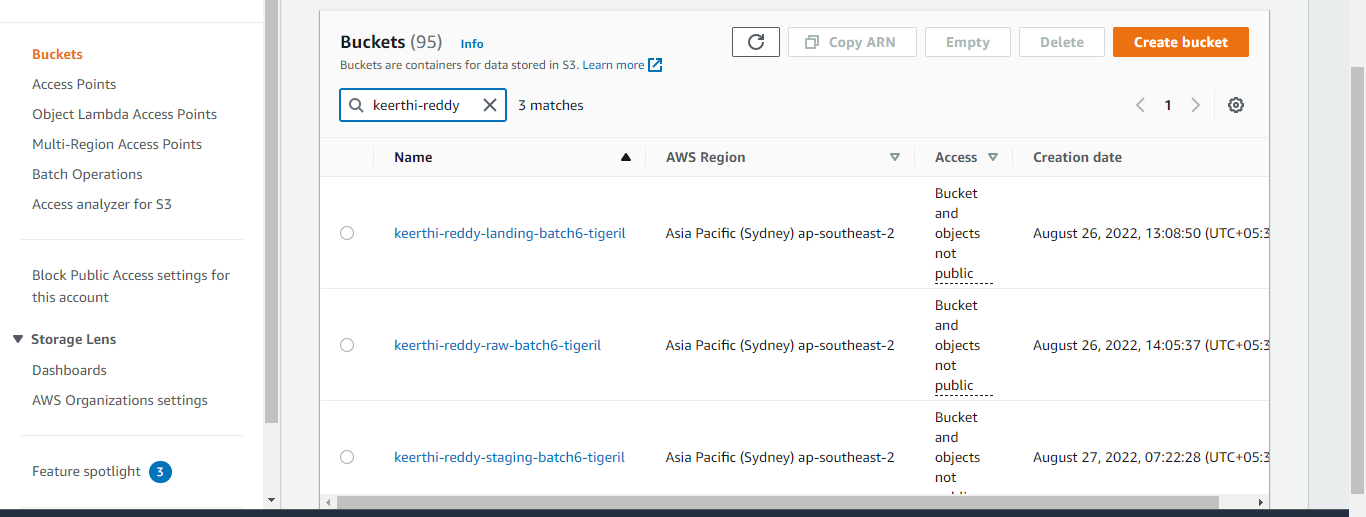


Click on create bucket

Graphical user interface, text, application, email

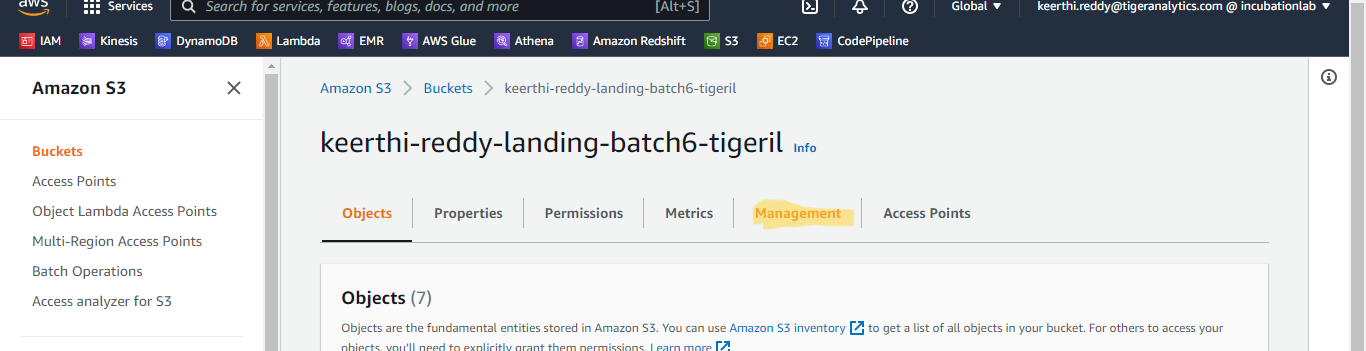
Description automatically generated



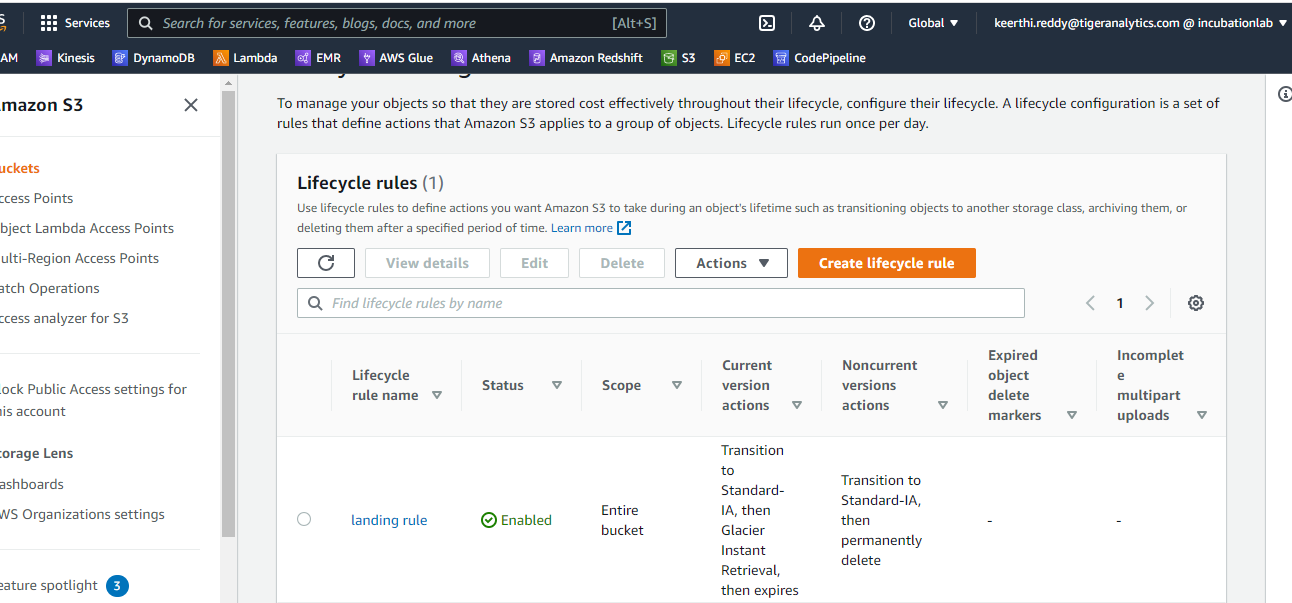


* Enable appropriate **life-cycle management** on these buckets.

After creating 3 buckets, we have to enable life-cycle management for each bucket



Click on create lifecycle rules



* Read & Write access for raw bucket should be limited to a service account for programmatic access only. This is for GDPR compliance. Use **IAM**

Here we have to block public access—go to permissions and click on edit option in block public access

Graphical user interface, text, application, email

Description automatically generated

Before editing read and write access to some account we have to unblock public access after providing access to some account we can block public access.

Raw bucket policy:-

{

"Version": "2012-10-17",

"Id": "Policy1661846406471",

"Statement": [

{

"Sid": "Stmt1661846231112",

"Effect": "Deny",

"Principal": "\*",

"Action": "s3:ListBucket",

"Resource": "arn:aws:s3:::keerthi-reddy-raw-batch6-tigeril",

"Condition": {

"StringNotEquals": {

"aws:username": "keerthi.reddy@tigeranalytics.com"

}

}

},

{

"Sid": "Stmt1661846373812",

"Effect": "Deny",

"Principal": "\*",

"Action": [

"s3:DeleteObject",

"s3:GetObject",

"s3:PutObject"

],

"Resource": "arn:aws:s3:::keerthi-reddy-raw-batch6-tigeril/\*",

"Condition": {

"StringNotEquals": {

"aws:username": "keerthi.reddy@tigeranalytics.com"

}

}

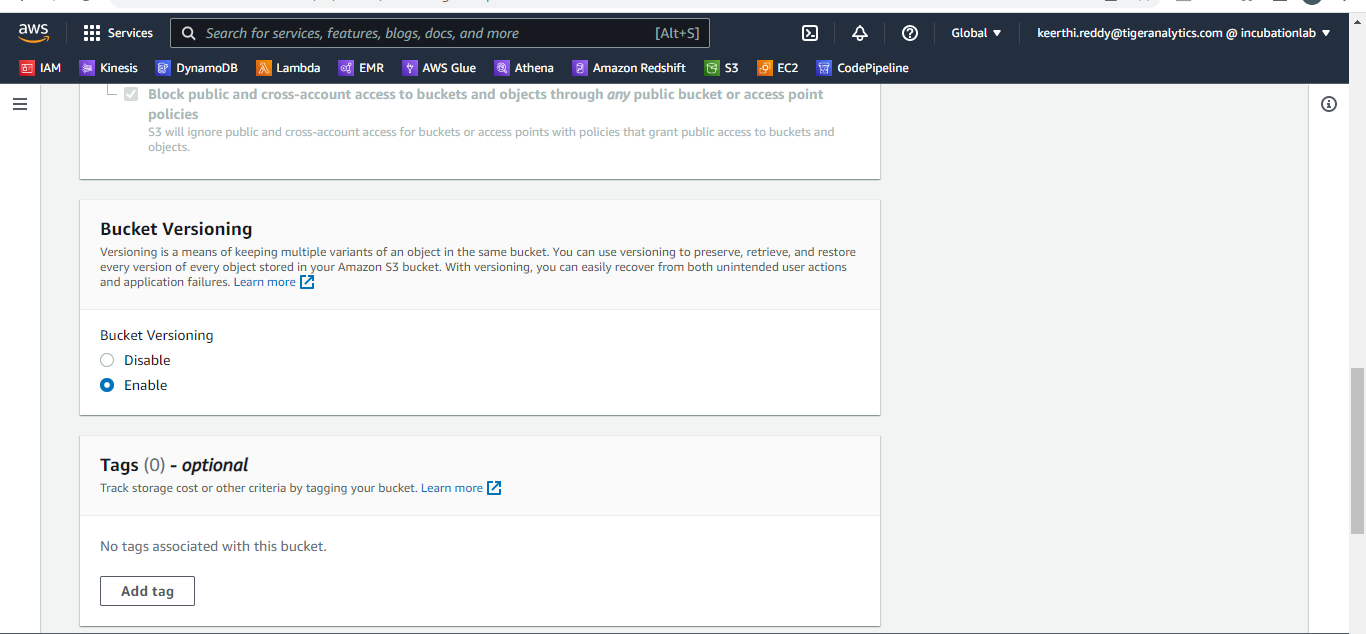
}

]

}

After providing bucket policy we can block public access to this bucket

* Enable bucket **versioning** on these buckets while creating the bucket.



* **Prepare sample data** in parquet format by referring to the schema.

First open jupyter notebook

run following commands:

pip install faker

pip install pyarrow

In local drive make one folder named "SampleData"

Then in jupyter notebook run these two programs

from faker import Faker

import pandas as pd

import json

import datetime

import time

import pyarrow as py

from random import randint

def input\_data(x):

    fake = Faker()

    date=fake.date\_time()

    timestamp=time.mktime(date.timetuple())

    # dictionary

    actives\_dict =[]

    for i in range(0, x):

        actives\_data={}

        actives\_data['advertising\_id']= fake.password(length= 8, special\_chars=False)

        actives\_data['city']= fake.city()

        actives\_data['location\_category']= fake.company()

        actives\_data['location\_granularities']= fake.country()

        actives\_data['location\_source']= fake.words()

        actives\_data['state'] = fake.state()

        actives\_data['timestamp'] = fake.date\_time()

        actives\_data['user\_id'] = fake.password(length= 8, special\_chars=False)

        actives\_data['user\_latitude'] = float(fake.latitude())

        actives\_data['user\_longitude'] = float(fake.longitude())

        actives\_data['month'] = str(actives\_data['timestamp'].month) #I have modified the code here.

actives\_data['date'] = actives\_data['timestamp'].date() #I have modified the code here

        actives\_dict.append(actives\_data)

    #print(actives\_data)

    actives\_df = pd.DataFrame(actives\_dict)

    actives\_df.to\_parquet("C:\\SampleData\\actives.parquet", engine="auto", compression ="snappy")

    print(actives\_df.head())

    print(actives\_dict)

    return actives\_dict

def main():

    number\_of\_activesdata = 100

    input\_data((number\_of\_activesdata))

main()

Viewership-Dataset

from faker import Faker

import pandas as pd

import json

import datetime

import time

import pyarrow as py

from random import randint

def input\_data(x):

    fake = Faker()

    date=fake.date\_time()

    timestamp=time.mktime(date.timetuple())

    # dictionary

    viewership\_dict =[]

    for i in range(0, x):

        viewership\_data={}

        viewership\_data['advertising\_id']= fake.password(length= 8, special\_chars=False)

        viewership\_data['channel\_genre'] = fake.city()

        viewership\_data['channel\_name'] = fake.name()

        viewership\_data['city']= fake.city()

        viewership\_data['device'] = fake.city()

        viewership\_data['device\_type'] = fake.city()

        viewership\_data['duration'] = fake.random\_int(900,18001)

        viewership\_data['grid\_id'] = fake.password(6,False,True,False)

        viewership\_data['language'] = fake.city()

        viewership\_data['location\_category']= str(fake.company())

        viewership\_data['location\_granularities']= str(fake.country())

        viewership\_data['location\_source']= fake.address()

        viewership\_data['record\_timestamp'] = fake.date\_time()

        viewership\_data['show\_genre'] = fake.state()

        viewership\_data['show\_name'] = fake.name()

        viewership\_data['state'] = fake.state()

        viewership\_data['user\_lat'] = float(fake.latitude())

        viewership\_data['user\_long'] = float(fake.longitude())

        viewership\_data['month'] = fake.month()

        viewership\_data['date'] = fake.date()

        viewership\_dict.append(viewership\_data)

    #print(actives\_data)

    viewership\_df = pd.DataFrame(viewership\_dict)

    viewership\_df.to\_parquet("C:\\SampleData\\viewership.parquet", engine="auto", compression ="snappy")

    print(viewership\_dict)

    return viewership\_dict

def main():

    number\_of\_viewershipdata = 100

    input\_data((number\_of\_viewershipdata))

main()