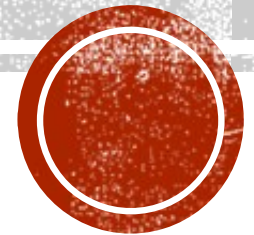
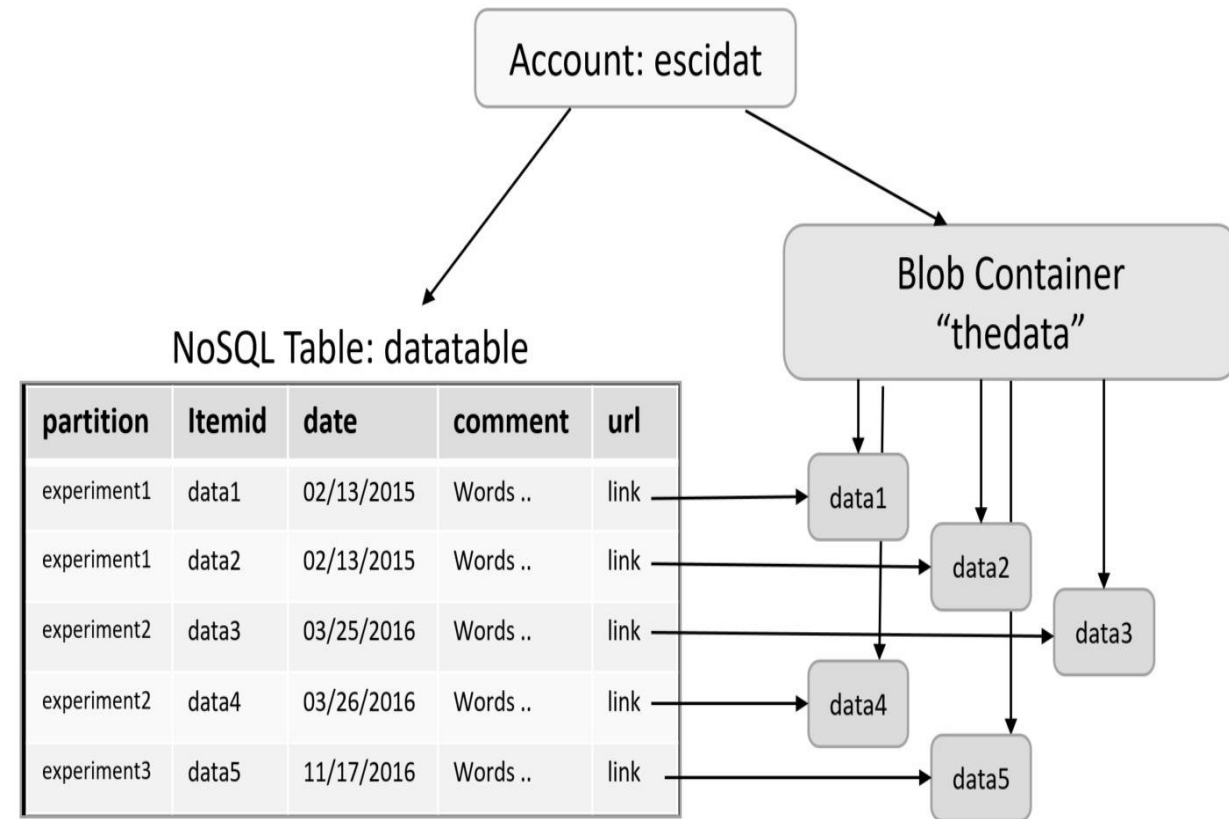


# CLOUD DATA STORAGE HOMEWORK



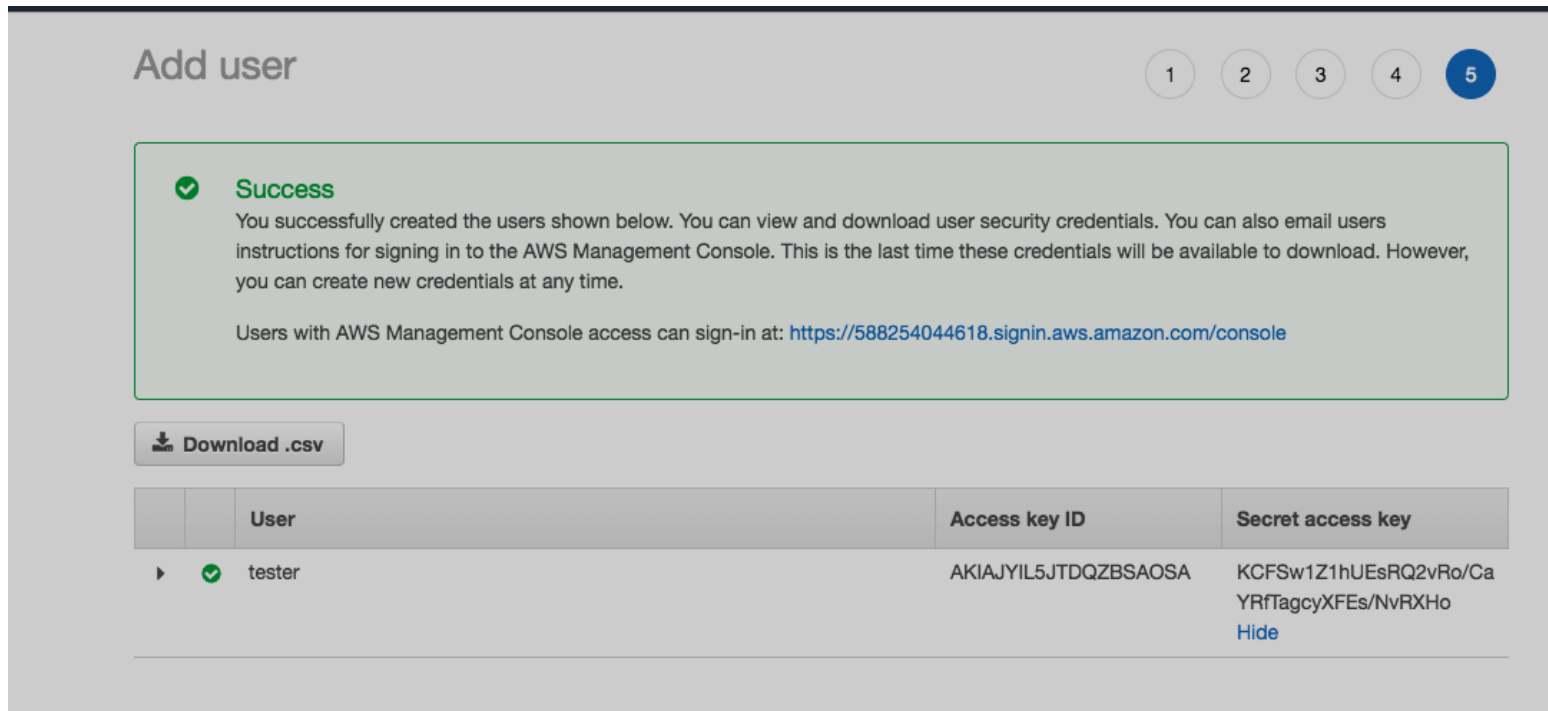
# PROBLEM DESCRIPTION

- Experiments may have different purposes and therefore, they will capture different data points.
- NoSQL DB allow you to upload your experiment data – with their distinctions- to a one NoSQL DB.
- In this assignment, you are tasked to build a NoSQL Dynamo DB for various experiments data.
- General information about all experiments will be stored in master CSV file. Each record in the master CSV will provide general information about an experiment. This CSV will be useful to capture common data points among all experiments. However, each experiment will have its own set of data points that may differ from other experiments. So, each record in the master CSV will include a location to a file on the local hard disk to provide all the special data points that are unique for each experiment (In the screenshot on the right, please notice the URL column)
- Your application will read the experiments CSV and upload the data for each experiment to your NoSQL Database.
- In order to do so, please execute the following steps:
  - Create S3 Instance Object
  - Create DynamoDB Table.
  - Read the master CSV for Experiments
  - Read the data for each experiment based on the location stored in the master CSV
  - Upload the BLOB data to your NoSQL DB and Fill your NoSQL DB with experiment data



# THIS IS SUGGESTED SOLUTION AND YOU DON'T NEED TO FOLLOW IT!

- **Note: You may find Homework Supportive materials Document on Canvas helpful for you.**
- First setup AWS account and download security credential from the Amazon IAM (Identity and **A**ccess **M**anagement)



The screenshot shows the 'Add user' page in the AWS IAM console. At the top, there are five numbered steps, with step 5 being the active one. A green success message indicates that the user 'tester' was created successfully. Below the message is a 'Download .csv' button. A table displays the user's details, including the Access key ID and the Secret access key. The secret key is partially visible and has a 'Hide' link next to it.

**Add user**

1 2 3 4 **5**

✓ **Success**  
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: <https://588254044618.signin.aws.amazon.com/console>

Download .csv

	User	Access key ID	Secret access key
▶ ✓	tester	AKIAJYL5JTDQZBSAOSA	KCFSw1Z1hUEsRQ2vRo/CaYRfTagcyXFEs/NvRXHo <a href="#">Hide</a>



# SUGGESTED SOLUTION — CONT'D

Use the Amazon Python Boto3 SDK (<https://pypi.org/project/boto3/>) to

- Access resource using a pair of *access* and *secret* keys
- Create a S3 bucket *datacont-name*
  - *Hint#1: make your S3 bucket publicly accessible.*
  - *Hint#2: You may face some problems with creating your S3 buckets due to selecting a reserved bucket name, so try different bucket names if you face such issue.*
- Upload blobs to the bucket

```
import boto3
s3 = boto3.resource('s3',
    aws_access_key_id='YOUR ACCESS KEY',
    aws_secret_access_key='your secret key' )
```

```
s3.create_bucket(Bucket='datacont', CreateBucketConfiguration={
    'LocationConstraint': 'us-west-2'})
```

```
# Upload a file, 'test.jpg' into the newly created bucket
s3.Object('datacont', 'test.jpg').put(
    Body=open('/home/mydata/test.jpg', 'rb'))
```

rb = read binary



# SUGGESTED SOLUTION — CONT'D

- Create a DynamoDB table to store metadata and references to S3 objects

```
dyndb = boto3.resource('dynamodb', region_name='us-west-2' )

# The first time that we define a table, we use
table = dyndb.create_table(
    TableName='DataTable',
    KeySchema=[
        { 'AttributeName': 'PartitionKey', 'KeyType': 'HASH'},
        { 'AttributeName': 'RowKey', 'KeyType': 'RANGE' }
    ],
    AttributeDefinitions=[
        { 'AttributeName': 'PartitionKey', 'AttributeType': 'S' },
        { 'AttributeName': 'RowKey', 'AttributeType': 'S' }
    ]
)

# Wait for the table to be created
table.meta.client.get_waiter('table_exists')
                        .wait(TableName='DataTable')

# If the table has been previously defined, use:
# table = dyndb.Table("DataTable")
```



# SUGGESTED SOLUTION – CONT'D

- Read the metadata from a CSV file with the following format:

Item id, experiment id, date, filename, comment string

- Move the data objects into the blob store
- Enter the metadata row into the table

```
import csv
urlbase = "https://s3-us-west-2.amazonaws.com/datacont/"
with open('\path-to-your-data\experiments.csv', 'rb') as csvfile:
    csvf = csv.reader(csvfile, delimiter=',', quotechar='|')
    for item in csvf:
        body = open('path-to-your-data\datafiles\'\''+item[3], 'rb')
        s3.Object('datacont', item[3]).put(Body=body)
        md = s3.Object('datacont', item[3]).Acl()
            .put(ACL='public-read') #Set URL for the data file is to be
            #publicly readable.
        url=urlbase +item[3]
        metadata_item={'PartitionKey': item[0], 'RowKey': item[1],
            'description' : item[4], 'date' : item[2], 'url':url}
        table.put_item(Item=metadata_item)
```



# DELIVERABLES

- GitHub URL. Your repository should contain the following:
  - Screenshot of your AWS Account with IAM – Similar to Slide# 3 (25%)
    - Make sure to show the name of your account in the Screenshot
  - Screenshot showing you installed Amazon Python Boto3 SDK (25%)
  - A print of your code (25%)
  - Screenshot of your query to the newly constructed table (25%)

**Your GitHub repository should be public**



# FOR MORE INFORMATION

- Detailed code and steps for execution can be found in the guidelines document on the website.
- AWS - [aws.amazon.com](https://aws.amazon.com)
  - Signup for AWS educate
  - <https://aws.amazon.com/education/awseducate/>

