BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

TEAM MEMBERS:-

NAGAPPAN

AKASH

SIVAKARTHICK

SANDEEP

BARATH KUMAR

Development part-1:

o In this phase, We're going to do three important steps in big data analysis &cloud.

o They are,

- Loading the data to cloud,
- Reading a data from Cloud,
- Data preprocessing

Selection of dataset:

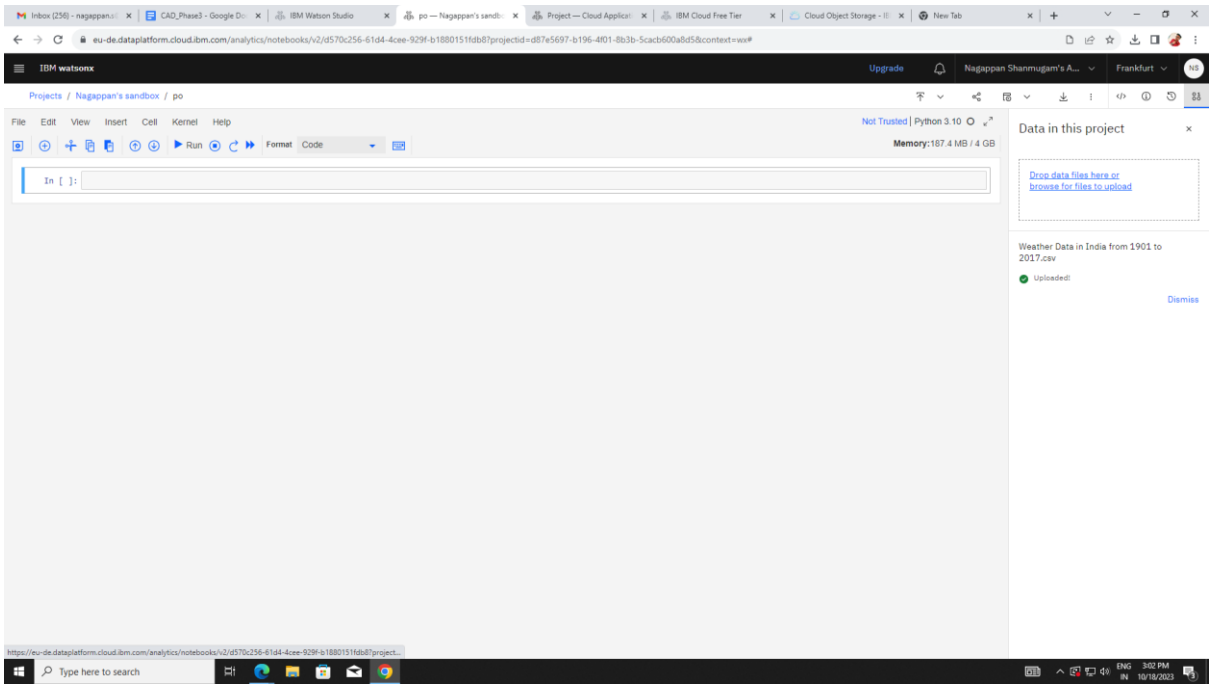
As per the project, We've to select social trends or climate dataset.

BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

So, We took weather data in india from 1901 to 2017 which is in the CSV format for our big data analysis.

Loading the data to the cloud object storage:

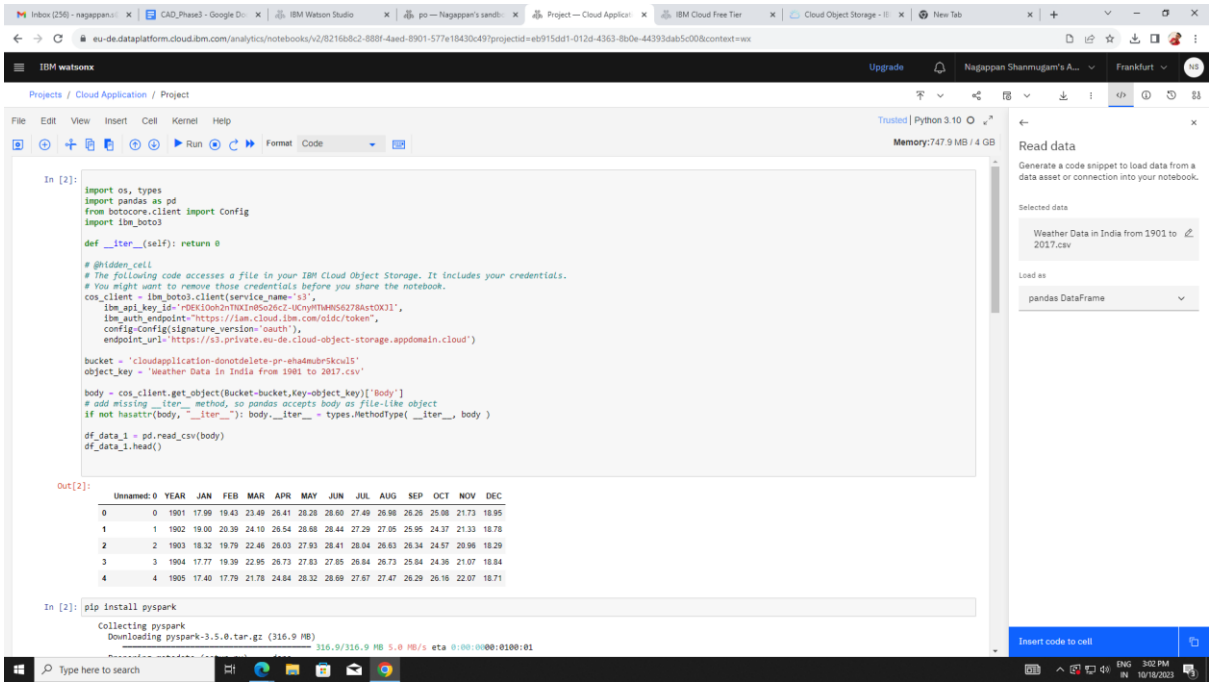
Step-1: Go to the upload asset to project, Then browse the dataset & upload.



BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

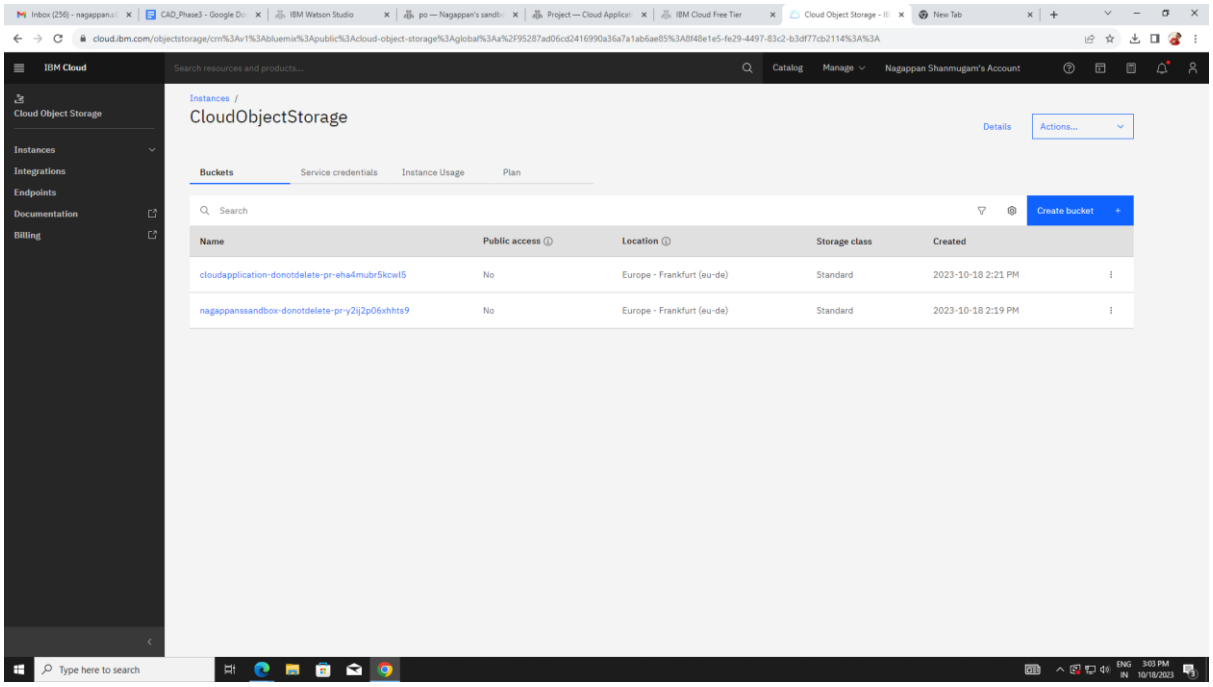
Step-2: After step-1, go to code snippets, then click read data & select the data from project.

Then click load as panda frame, the above codes will be display.



BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

Step-3: Buckets are created in IBM cloud object storage



BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

Reading the data from cloud object storage

Step-1: As we already known, Data are loaded in cloud object storage as pandas data frame.

Code [1]

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='rDEKiOoh2nTNXIn0So26cZ-UCnyMTWHNS6278AstOXJl',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.eu-de.cloud-object-storage.appdomain.cloud')

bucket = 'cloudapplication-donotdelete-pr-eha4mubr5kcwl5'
object_key = 'Weather Data in India from 1901 to 2017.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df_data_1 = pd.read_csv(body)
df_data_1.head()
```

Output[1]

BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

Unnamed: 0	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
0	1901	17.99	19.43	23.49	26.41	28.28	28.60	27.49	26.98	26.26	25.08	21.73	18.95
1	1902	19.00	20.39	24.10	26.54	28.68	28.44	27.29	27.05	25.95	24.37	21.33	18.78
2	1903	18.32	19.79	22.46	26.03	27.93	28.41	28.04	26.63	26.34	24.57	20.96	18.29
3	1904	17.77	19.39	22.95	26.73	27.83	27.85	26.84	26.73	25.84	24.36	21.07	18.84
4	1905	17.40	17.79	21.78	24.84	28.32	28.69	27.67	27.47	26.29	26.16	22.07	18.71

Step-2: Convert pandas data frame into pyspark dataframe.

Code[2]

```
from pyspark.sql.types import *
from pyspark.sql.functions import *
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.feature import HashingTF, Tokenizer, StopWordsRemover
import pyspark
from pyspark.sql import SparkSession
#create Spark session
appName = "Weather Report"
spark = SparkSession \
.builder \
.appName(appName) \
.config("spark.some.config.option", "some-value") \
.getOrCreate()
data_frame=spark.createDataFrame(df_data_1)
display(data_frame)
```

Output[2]

BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

DataFrame[Unnamed: 0: bigint, YEAR: bigint, JAN: double, FEB: double, MAR: double, APR: double, MAY: double, JUN: double, JUL: double, AUG: double, SEP: double, OCT: double, NOV: double, DEC: double]

Data Preprocessing:

o Data preprocessing is a crucial step in the big data analysis on IBM cloud object storage.

o It involves:

- Data cleaning,
- Data transformation,
- Organizing the raw data,
- Handling duplicates,
- Handling missing data

o Handling Duplicates:

Code[3]

```
#preprocessing

#handling duplicates

a = df.count()

b = df.dropDuplicates().count()

c = a - b

print("No.of original data: ", a)

print("No of data rows after deleting duplicated data: ", b)

print("number of duplicated data: ", c)
```

Output[3]

```
No.of original data:  117

No of data rows after deleting duplicated data:  117

number of duplicated data:  0
```

o Handling missing data:

Code[4]

```
#Handling missing data:
```

BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

```
no_miss_val = df.dropDuplicates().dropna(how="any",
subset=["JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG","SEP","OCT","NOV","DEC"])

miss_val= a - no_miss_val.count()

print("number of missing value rows: ", miss_val)
```

Output[4]

```
number of missing value rows:  0
```

Drop duplicated data and fill missing data with mean value

Code [5]

```
#Drop duplicated data and fill missing data with mean value

m1 = df.groupBy().avg("JAN").take(1)[0][0]

print("Mean of JAN: ", m1)

m2 = df.groupBy().avg("DEC").take(1)[0][0]

print("Mean of DEC ", m2)

TweetCleanData=df.fillna( {'JAN': m1, 'DEC': m2})

df.groupBy().avg("JAN").show()

df.describe('JAN','DEC').show()
```

Output[5]

```
Mean of JAN:  18.42324786324786

Mean of DEC  19.173333333333332

+-----------------+

|         avg(JAN)|

+-----------------+

|18.42324786324786|

+-----------------+

+-------+------------------+------------------+

|summary|               JAN|               DEC|

+-------+------------------+------------------+

|  count|               117|               117|

|   mean| 18.42324786324786|19.173333333333332|

| stddev|0.6129631662723346|0.6359123231693377|
```

BIG DATA ANALYSIS WITH IBM CLOUD DATABASES [PHASE-3]

| min| 17.25| 17.98|

| max| 20.92| 21.89|

+-------+------------------+------------------+

o Correlation

Code [6]

```
#correlation
cor = df.corr('JAN', 'DEC')
print("correlation between JAN & DEC", cor)
```

Output[6]

correlation between JAN & DEC 0.5021710390764857

Technologies used:

- Watson Studio
- Cloud object Storage
- Pyspark
- Pandas