Share Market History Data Analysis

LTIMindtree

PRESENTATION BY

NAME – Arpit Muduli

COMPANY – LTIMindtree

INTERN ID – T 6504

TECH. USED – AWS, PYTHON, OLTP DATABASE

GUIDED BY – Kanchan Patra

DATE OF SUBMISSION – April 19, 2024

TOTAL NUMBER OF CASES – 4

INTRODUCTION

This case study is given by LTIMINDTREE under Internship program.

The aim of this case study to work on the industry-based problem statements and achieve the solution for these problem statements.

The problem statement is divided into many numbers of cases. Each of the case has its solution based on different services provided by LTIMINDTREE. My objective is to discover and apply my learnings to find the appropriate solution for the cases. The different services which are used here to solve the cases of the problem statement are as follows:

- AWS Cloud
  - S3
  - IAM
  - Lambda
  - DynamoDB
  - RDS
- Python
- RDS/PostgreSQL

CASE STUDY

Export historical NIFTY50 data in csv format

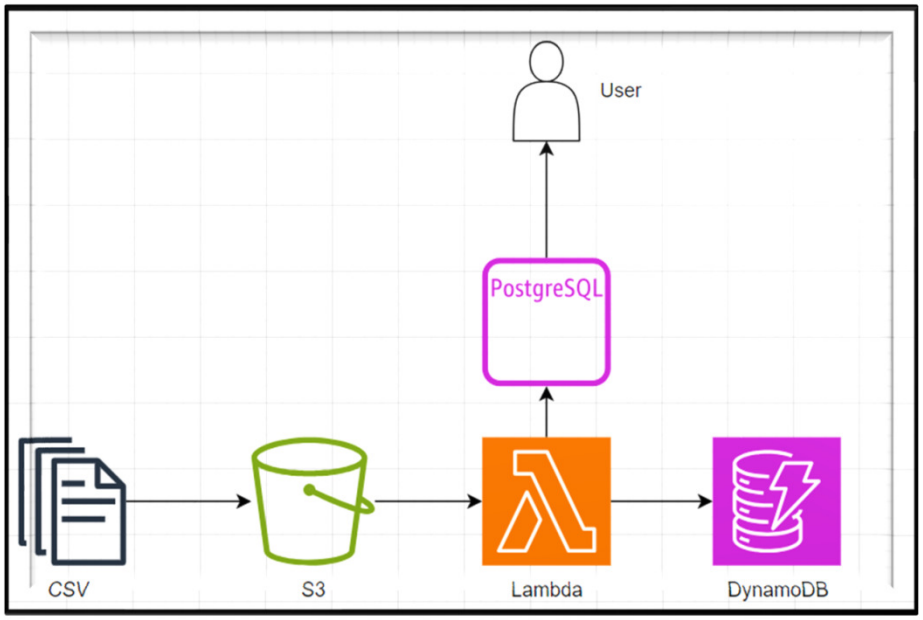
Upload the data into s3 manually.

Using Python lambda function, read data from s3 and process the same and write into DynamoDB and PostgreSQL.

Read data from PostgreSQL, using Store Procedure to calculate the moving average% to write into another report table for consumption. Mark/ flag DynamoDB data as processed.

ARCTITCTURE

Use Case Diagram



Now on, Let’s Solve!

AWS infrastructure Creation & Gathering History Data of Nifty50

Using the link given below, I exported the Nifty50 data into CSV format
Link: https://www.nseindia.com/reports-indices-historical-index-data

For Infra creation, signup/signin to AWS using email & password we have to follow the below slides

AWS IAM Role Creation

Steps:

1. Login to AWS console
2. Find the IAM service and click on it
3. Click on role and create role
4. Choose the AWS service and Lambda
5. Attach the required policy with permission to the role
6. Give the appropriate role name and create it

IAM Role Name: bse-nifty-service-role

AWS S3 Bucket and Folder Creation

Steps:

1. Login to AWS console
2. Find the S3 service and click on it
3. Create a Bucket
4. Created a folder within the bucket

Bucket Name: bse-nifty-bkt
Folder Name: nifty-src-bkt-src

After bucket and folder creation, the CSV Nifty50 file has been uploaded into it.

AWS RDS PostgreSQL DB creation

Steps:

1. Login to AWS console
2. Find the RDS service and click on it
3. Trying to create a free tier PostgreSQL database from the listed database option
4. Choose the PostgreSQL form the list and created a database
5. Provide the appropriate permission(VPC/Security Group/Subnet) to connect with Local and Lambda

Database Name: nifty50db

AWS DynamoDB Creation

Steps:

1. Login to AWS
2. Find the DynamoDB service and click on it
3. Click on Table and Create Table
4. Give the appropriate name of DynamoDB Table
5. Based on partition key and sort key create the table

DDB Table Name: nifty50-ddb

AWS Lambda Creation

Steps:

1. Login to AWS
2. Find the Lambda service and click on it
3. Click on Create function
4. Give the appropriate name of function
5. Choose Runtime(Python 3.9) and service role
6. Click on create function

Lambda Function Name: nifty50-lambda

Python Code: S3 File Processing Python script

```python
def s3_file_processing(bucket_name,file_name):

    print("Trying to get the object from s3")
    try:
        s3_response = s3_client.get_object(Bucket=bucket_name, Key=file_name)
        print("Objcet got successfully from the s3 bucket and processing started")
    except Exception as e:
        print("S3 get object error-->",e)
    try:

        data = s3_response["Body"].read().decode('utf')
        df = pd.read_csv(StringIO(data))
        df.rename(columns=lambda x: x.strip(), inplace=True)

        # Fill null values with strategy considering first and last rows
        def fill_nulls(df):
            df["Shares Traded"].fillna((df["Shares Traded"].ffill() + df["Shares Traded"].bfill()) / 2, inplace=True)
            df["Turnover"].fillna((df["Turnover"].ffill() + df["Turnover"].bfill()) / 2, inplace=True)
            return df

        df = fill_nulls(df.copy())

        csv_string = df.to_csv(index=False)
        csv_bytes = csv_string.encode('utf-8')
    except Exception as e:
        print("Data Processing error-->",e)

    target=f'target/niftymod_{datetime.datetime.now(ist_timezone).strftime("%Y%m%d%H%M%S")}.csv'
    try:

        s3_client.put_object(Body=csv_bytes, Bucket=bucket_name, Key=target)
        print("Nifty data processed successfully in S3 and uploaded to target folder")
        # calling the s3_to_ddb_file_processing function to insert data in DDB
        s3_to_ddb_file_processing(bucket_name,target)
    except Exception as e:
        print("S3 put object error-->",e)
```

Description: This script is used to process the CSV data within the S3 bucket where I have used boto3 s3 client to get and put object to process and store the csv data

For removing the null value from csv data, I have used panda library data frame to predict the null data using previous and after data.

Python Code: S3 File to DDB Python script

```python
def s3_to_ddb_file_processing(bucket_name,target):

    dynamodb = boto3.resource("dynamodb")
    table = dynamodb.Table(os.environ['table'])

    print("Trying to get the object from s3")
    resp = s3_client.get_object(Bucket=bucket_name,Key=target)
    print("Objcet got successfully from the s3 bucket and DDB processing started")

    data = resp['Body'].read().decode("utf-8")
    Stocks = data.split("\n")
    try:
        for stock in Stocks:
            if stock:
                stock_data = stock.split(",")
                table.put_item(
                    Item = {
                        "Id"         : stock_data[0],
                        "Date"       : stock_data[1],
                        "Open"    :    stock_data[2],
                        "High"   :     stock_data[3],
                        "Low"   :     stock_data[4],
                        "Close"   :     stock_data[5],
                        "Shares Traded"   :  stock_data[6],
                        "Turnover"   :     stock_data[7]
                    }
                )
        print("DynomoDb processing is done...")
        #calling the dddb_to_postgresql_processing function to insert the processed data
        dddb_to_postgresql_processing(bucket_name,target)
    except Exception as e:
        print("DDB processing error-->",e)
```

Description: This script is used to process the CSV data S3 to DDB where I have used boto3 s3 client to get object and DDB put object to process and store the csv data into DDB table

Python Code: S3 File to PostgreSQL Python script

```python
def dddb_to_postgresql_processing(bucket_name,file_key):

    table_name=os.environ['table_name']
    column_names=os.environ['column_names']

    session = boto3.session.Session()
    secrets_manager = boto3.client("secretsmanager")

    try:
        secret_response = secrets_manager.get_secret_value(
            SecretId=secret_name
        )

        secrets = json.loads(secret_response["SecretString"])
        psg_conn = secrets["rds_connection"]

        #geting the csv data from s3
        try:
            response = s3_client.get_object(Bucket=bucket_name, Key=file_key)
            csv_data = response['Body'].read().decode('utf-8')
            #reading the csv data in dataframe
            df = pd.read_csv(StringIO(csv_data))
        except Exception as e:
            print("CSV reading data error:",e)


        #arraning the data as per insert script.
        df2 = df[['Id','Date','Open','High','Low','Close','Shares Traded','Turnover']]
```

```python
        #arraning the data as per insert script.
        df2 = df[['Id','Date','Open','High','Low','Close','Shares Traded','Turnover']]

        tring to connect with db
        try:
            print("Trying to connect the DB...")
            connection = psycopg2.connect(psg_conn)
            cursor = connection.cursor()
            print("DB connected...")
        except Exception as e:
            print(f"DB connection error: {e}")

        # Executing the insert script
        try:
            #Inserting the new data
            query=f""" INSERT INTO {table_name} ({column_names})
            VALUES %s """ % ','.join([str(i) for i in list(df2.to_records(index=False))])
            print("query--->",query)
            cursor.execute(query)
            connection.commit()
            print("Data processing successfully completed from ddb to postgreql...")
        except Exception as e:
            print(f"DB processing error: {e}")
        finally:
            closing the DB connection and cursor.
            cursor.close()
            connection.close()
            print("DB connection closed...")
    except Exception as e:
            print(f"SQL processing of CSV data error: {e}")
```

Description: This script is used to process the CSV data S3 to PostgreSQL where I have used boto3 s3 client to get object and panda to convert the object in data frame after that I have used psycopg2 library to connect with PostgreSQL and form the insert script and store into my table.

Database: Main Table creation

```
CREATE TABLE public.nifty_data_main (
    nf_id int4 NOT NULL,
    "date" varchar(30) NULL,
    "open" numeric(38, 6) NULL,
    high numeric(38, 6) NULL,
    low numeric(38, 6) NULL,
    "close" numeric(38, 6) NULL,
    shares_traded numeric(38, 6) NULL,
    turnover numeric(38, 6) NULL,
    CONSTRAINT nifty_data_main_pkey PRIMARY KEY (nf_id)
);
```

Description: This ddl script is used to create the main table in PostgreSQL to store the processed data pulled by AWS Lambda.

Database: Report Table creation

```sql
CREATE TABLE public.nifty_data_report (
    nf_id int4 NOT NULL,
    "date" varchar(30) NULL,
    "open" numeric(38, 6) NULL,
    high numeric(38, 6) NULL,
    low numeric(38, 6) NULL,
    "close" numeric(38, 6) NULL,
    shares_traded numeric(38, 6) NULL,
    turnover numeric(38, 6) NULL,
    ma_20 numeric(38, 6) NULL,
    ma_50 numeric(38, 6) NULL,
    CONSTRAINT nifty_data_report_pkey PRIMARY KEY (nf_id)
);
```

Description: This ddl script is used to create the report table in PostgreSQL to store the processed data pulled by store procedure.

Database: Store procedure creation

```
CREATE OR REPLACE FUNCTION calculate_moving_average_test()
RETURNS VOID AS
$$
BEGIN
    -- Create a temporary table to store intermediate results
    --DROP TABLE IF EXISTS temp_ma_datas;
    CREATE TEMP TABLE temp_ma_datas (
    Nf_Id int primary key,
    date varchar(30),
    open decimal(38,6),
    high decimal(38,6),
    low decimal(38,6),
    close decimal(38,6),
    shares_traded decimal(38,6),
    turnover decimal(38,6),
    ma_20 decimal(38,6),
    ma_50 decimal(38,6)
    );

    -- Calculate the moving averages and store the results in the temporary table
    INSERT INTO temp_ma_datas (nf_id, "date", "open", high, low, "close", shares_traded, turnover, ma_20, ma_50)
    select
        nf_id,
        date,
        open,
        high,
        low,
        close,
        shares_traded,
        turnover,
        AVG(close) OVER (ORDER BY date ROWS BETWEEN 19 PRECEDING AND CURRENT ROW) AS ma_20,
        AVG(close) OVER (ORDER BY date ROWS BETWEEN 49 PRECEDING AND CURRENT ROW) AS ma_50
    FROM
        nifty_data_main;

    insert into public.nifty_data_report
    (nf_id, "date", "open", high, low, "close", shares_traded, turnover,ma_20,ma_50)
    select * from temp_ma_datas;

END;
$$
LANGUAGE plpgsql;
```

Description: This store procedure is used to transform the pulled data by AWS Lambda, In transforming the data we are calculating the moving average by 20 and 50 and storing into the temp table after that we are storing the data into report table using temp table.

CONCLUSIONS

As per the case study I have completed more than 95% of the cases .Due to lack of some logistic and knowledge, I have not able to complete some cases. The tasks that require the IAM permissions for service like create new user, PowerBI Reporting etc.

Overall, it’s a very good experience to do this case study.

REFERENCES

- https://docs.aws.amazon.com/s3/index.html
- https://docs.aws.amazon.com/dynamodb/
- https://docs.aws.amazon.com/lambda/
- https://docs.aws.amazon.com/rds/
- https://docs.aws.amazon.com/iam/
- https://www.postgresql.org/docs
- https://pypi.org/project/psycopg2/
- https://boto3.amazonaws.com/v1/documentation/api/latest/index.html

THANK YOU