L3 Exercise 3 - Parallel ETL

January 14, 2023

1 Exercise 3: Parallel ETL

```
In [34]: %load_ext sql
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
```

```
In [35]: import boto3
         import configparser
         import matplotlib.pyplot as plt
         import pandas as pd
         from time import time
```

2 STEP 1: Get the params of the created redshift cluster

- We need:
  - The redshift cluster endpoint
  - The IAM role ARN that give access to Redshift to read from S3

```
In [36]: config = configparser.ConfigParser()
         config.read_file(open('dwh.cfg'))

         REGION_NAME = config.get('default', 'region_name')
         KEY=config.get('AWS','key')
         SECRET= config.get('AWS','secret')

         DWH_DB= config.get("DWH","DWH_DB")
         DWH_DB_USER= config.get("DWH","DWH_DB_USER")
         DWH_DB_PASSWORD= config.get("DWH","DWH_DB_PASSWORD")
         DWH_PORT = config.get("DWH","DWH_PORT")
```

```
In [37]: # FILL IN THE REDSHIFT ENPOINT HERE
         # e.g. DWH_ENDPOINT="redshift-cluster-1.csmamz5zxmle.us-west-2.redshift.amazonaws.com"
         DWH_ENDPOINT="dwhcluster.ceok3cz80k9u.us-west-2.redshift.amazonaws.com"

         #FILL IN THE IAM ROLE ARN you got in step 2.2 of the previous exercise
         #e.g DWH_ROLE_ARN="arn:aws:iam::988332130976:role/dwhRole"
         DWH_ROLE_ARN="arn:aws:iam::996813506119:role/dwhRole"
```

3 STEP 2: Connect to the Redshift Cluster

```
In [40]: conn_string="postgresql://{}:{}@{}:{}/{}".format(DWH_DB_USER, DWH_DB_PASSWORD, DWH_ENDP
         print(conn_string)
         %sql $conn_string
```

```
postgresql://dwhuser:Passw0rd@dwhcluster.ceok3cz80k9u.us-west-2.redshift.amazonaws.com:5439/dwh
```

```
Out[40]: 'Connected: dwhuser@dwh'
```

```
In [41]: print(REGION_NAME)
```

```
us-west-2
```

```
In [43]: s3 =  boto3.resource('s3', region_name=REGION_NAME,
                               aws_access_key_id=KEY,
                               aws_secret_access_key=SECRET)

         sampleDbBucket =  s3.Bucket('udacity-labs')
```

```
In [44]: for obj in sampleDbBucket.objects.filter(Prefix="tickets"):
             print(obj)
```

```
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/')
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/full/')
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/full/full.csv.gz')
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/')
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00000-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00001-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00002-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00003-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00004-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00005-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00006-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00007-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00008-d33afb94-b8af-407d-ab
s3.ObjectSummary(bucket_name='udacity-labs', key='tickets/split/part-00009-d33afb94-b8af-407d-ab
```

4 STEP 3: Create Tables

```
In [46]: %%sql
         DROP TABLE IF EXISTS "sporting_event_ticket";
         CREATE TABLE "sporting_event_ticket" (
             "id" double precision DEFAULT nextval('sporting_event_ticket_seq') NOT NULL,
             "sporting_event_id" double precision NOT NULL,
             "sport_location_id" double precision NOT NULL,
```

```
            "seat_level" numeric(1,0) NOT NULL,
            "seat_section" character varying(15) NOT NULL,
            "seat_row" character varying(10) NOT NULL,
            "seat" character varying(10) NOT NULL,
            "ticketholder_id" double precision,
            "ticket_price" numeric(8,2) NOT NULL
        );
```

```
 * postgresql://dwhuser:***@dwhcluster.ceok3cz80k9u.us-west-2.redshift.amazonaws.com:5439/dwh
   postgresql://dwhuser:***@dwhcluster.chkkpnquq5ee.us-east-1.redshift.amazonaws.com:5439/dwh
Done.
Done.
```

Out[46]: []

5 STEP 4: Load Partitioned data into the cluster

Use the COPY command to load data from s3://udacity-labs/tickets/split/part using your iam role credentials. Use gzip delimiter ;.

In [48]:
```
%%time
qry = """
    COPY sporting_event_ticket FROM 's3://udacity-labs/tickets/split/part'
    CREDENTIALS 'aws_iam_role={}'
    gzip delimiter ';' compupdate off region 'us-west-2';
""".format(DWH_ROLE_ARN)

%sql $qry
```

```
 * postgresql://dwhuser:***@dwhcluster.ceok3cz80k9u.us-west-2.redshift.amazonaws.com:5439/dwh
   postgresql://dwhuser:***@dwhcluster.chkkpnquq5ee.us-east-1.redshift.amazonaws.com:5439/dwh
Done.
CPU times: user 3.79 ms, sys: 0 ns, total: 3.79 ms
Wall time: 12.1 s
```

6 STEP 5: Create Tables for the non-partitioned data

In [49]:
```
%%sql
DROP TABLE IF EXISTS "sporting_event_ticket_full";
CREATE TABLE "sporting_event_ticket_full" (
    "id" double precision DEFAULT nextval('sporting_event_ticket_seq') NOT NULL,
    "sporting_event_id" double precision NOT NULL,
    "sport_location_id" double precision NOT NULL,
    "seat_level" numeric(1,0) NOT NULL,
    "seat_section" character varying(15) NOT NULL,
```

```
            "seat_row" character varying(10) NOT NULL,
            "seat" character varying(10) NOT NULL,
            "ticketholder_id" double precision,
            "ticket_price" numeric(8,2) NOT NULL
        );
```

```
 * postgresql://dwhuser:***@dwhcluster.ceok3cz80k9u.us-west-2.redshift.amazonaws.com:5439/dwh
   postgresql://dwhuser:***@dwhcluster.chkkpnquq5ee.us-east-1.redshift.amazonaws.com:5439/dwh
Done.
Done.
```

Out[49]: []

7 STEP 6: Load non-partitioned data into the cluster

Use the COPY command to load data from `s3://udacity-labs/tickets/full/full.csv.gz` using your iam role credentials. Use gzip delimiter ;.

- Note how it's slower than loading partitioned data

In [50]:
```
%%time

qry = """
    COPY sporting_event_ticket_full FROM 's3://udacity-labs/tickets/full/full.csv.gz'
    CREDENTIALS 'aws_iam_role={}'
    gzip delimiter ';' compupdate off region 'us-west-2';
""".format(DWH_ROLE_ARN)

%sql $qry
```

```
 * postgresql://dwhuser:***@dwhcluster.ceok3cz80k9u.us-west-2.redshift.amazonaws.com:5439/dwh
   postgresql://dwhuser:***@dwhcluster.chkkpnquq5ee.us-east-1.redshift.amazonaws.com:5439/dwh
Done.
CPU times: user 3.99 ms, sys: 0 ns, total: 3.99 ms
Wall time: 22.8 s
```

In [ ]: