**Assignment 2**

**PREPARATION**

create or replace database demo\_db;

CREATE OR REPLACE TRANSIENT TABLE DEMO\_DB.PUBLIC.CUSTOMER\_TEST

AS SELECT \* FROM "SNOWFLAKE\_SAMPLE\_DATA"."TPCDS\_SF100TCL"."CUSTOMER";

create or replace STORAGE INTEGRATION my\_s3\_integration

type = External\_stage

storage\_provider = s3

enabled = true

storage\_aws\_role\_arn = 'arn:aws:iam::975050369468:role/sami-new-role'

storage\_allowed\_locations = ('s3://customer-data-sami/Customer\_folder/');

desc STORAGE INTEGRATION my\_s3\_integration;

create or replace stage my\_s3\_stage

STORAGE\_INTEGRATION = my\_s3\_integration

URL = 's3://customer-data-sami/Customer\_folder/';

COPY INTO @DEMO\_DB.PUBLIC.my\_s3\_stage/Customer\_folder/

from

DEMO\_DB.PUBLIC.CUSTOMER\_TEST;

**QUERY DATA IN S3 FROM SNOWFLAKE**

create or replace file format MY\_CSV\_FORMAT

type = 'CSV';

SELECT $1 C\_CUSTOMER\_SK,

$2 C\_CUSTOMER\_ID ,

$3 C\_CURRENT\_CDEMO\_SK ,

$4 C\_CURRENT\_HDEMO\_SK ,

$5 C\_CURRENT\_ADDR\_SK,

$6 C\_FIRST\_SHIPTO\_DATE\_SK ,

$7 C\_FIRST\_SALES\_DATE\_SK ,

$8 C\_SALUTATION ,

$9 C\_FIRST\_NAME ,

$10 C\_LAST\_NAME,

$11 C\_PREFERRED\_CUST\_FLAG ,

$12 C\_BIRTH\_DAY ,

$13 C\_BIRTH\_MONTH ,

$14 C\_BIRTH\_YEAR,

$16 C\_LOGIN ,

$17 C\_EMAIL\_ADDRESS ,

$18 C\_LAST\_REVIEW\_DATE

FROM @DEMO\_DB.PUBLIC.my\_s3\_stage/Customer\_folder/

(file\_format => DEMO\_DB.PUBLIC.MY\_CSV\_FORMAT);

--- Filter data directly from s3,

SELECT $1 C\_CUSTOMER\_SK,

$2 C\_CUSTOMER\_ID ,

$3 C\_CURRENT\_CDEMO\_SK ,

$4 C\_CURRENT\_HDEMO\_SK ,

$5 C\_CURRENT\_ADDR\_SK,

$6 C\_FIRST\_SHIPTO\_DATE\_SK ,

$7 C\_FIRST\_SALES\_DATE\_SK ,

$8 C\_SALUTATION ,

$9 C\_FIRST\_NAME ,

$10 C\_LAST\_NAME,

$11 C\_PREFERRED\_CUST\_FLAG ,

$12 C\_BIRTH\_DAY ,

$13 C\_BIRTH\_MONTH ,

$14 C\_BIRTH\_YEAR,

$16 C\_LOGIN ,

$17 C\_EMAIL\_ADDRESS ,

$18 C\_LAST\_REVIEW\_DATE

FROM @DEMO\_DB.PUBLIC.my\_s3\_stage/Customer\_folder/

(file\_format => DEMO\_DB.PUBLIC.MY\_CSV\_FORMAT)

WHERE C\_CUSTOMER\_SK ='64596949';

--- Execute group by

SELECT $9 C\_FIRST\_NAME,$10 C\_LAST\_NAME,COUNT(\*)

FROM @DEMO\_DB.PUBLIC.my\_s3\_stage/Customer\_folder/

(file\_format => DEMO\_DB.PUBLIC.MY\_CSV\_FORMAT);

GROUP BY $9,$10;

**CREATE VIEW OVER S3 DATA**

CREATE OR REPLACE VIEW CUSTOMER\_DATA

AS

SELECT $1 C\_CUSTOMER\_SK,

$2 C\_CUSTOMER\_ID ,

$3 C\_CURRENT\_CDEMO\_SK ,

$4 C\_CURRENT\_HDEMO\_SK ,

$5 C\_CURRENT\_ADDR\_SK,

$6 C\_FIRST\_SHIPTO\_DATE\_SK ,

$7 C\_FIRST\_SALES\_DATE\_SK ,

$8 C\_SALUTATION ,

$9 C\_FIRST\_NAME ,

$10 C\_LAST\_NAME,

$11 C\_PREFERRED\_CUST\_FLAG ,

$12 C\_BIRTH\_DAY ,

$13 C\_BIRTH\_MONTH ,

$14 C\_BIRTH\_YEAR,

$16 C\_LOGIN ,

$17 C\_EMAIL\_ADDRESS ,

$18 C\_LAST\_REVIEW\_DATE

FROM @DEMO\_DB.PUBLIC.my\_s3\_stage/Customer\_folder/

(file\_format => DEMO\_DB.PUBLIC.MY\_CSV\_FORMAT);

Query data directly on view,

SELECT \* FROM CUSTOMER\_DATA;

1--Now we can directly query data from s3 through view. What is the disadvantage of using this approach ? Can you see partitions being scanned in the backend ?

Ans:

Querying Data directly from s3 might be have higher latency compared to querying data stored within Snowflake’s own optimized storage layer.

Cost might be more with data transfer between s3 to snowflake.

Now view might will also increase latency so here we might get issue in complex datasets.

Unable to see partition for select \* from view query.

Able to see only :

Scan progress

100.00%

External bytes scanned

5.48GB

Bytes written to result

5.36GB

**Create a sample snowflake table**

Create or replace transient table CUSTOMER\_SNOWFLAKE\_TABLE

AS

SELECT \* FROM CUSTOMER\_TEST limit 10000;

A screenshot of a computer

Description automatically generated

**View Joining with previous view.**

SELECT B.\*

FROM CUSTOMER\_SNOWFLAKE\_TABLE B

LEFT OUTER JOIN

CUSTOMER\_DATA A

ON

A.C\_CUSTOMER\_SK = B.C\_CUSTOMER\_SK;

2-- Now we successfully joined data in s3 with snowflake table. It may look simple but this approach has lot of potential. Can you mention few below/

Ans: Step 1 : First create demo\_db database.

Step 2: After creating database we need to create transient table named CUSTOMER\_TEST.

Step 3: Once it is done we need to create storage integration, stage and s3 bucket and in IAM role we need to change trust policy by using the ARN number and external id(5,7 property) of storage integration.

Step 4: Now using select first we can select data and create view for that.

Step5: Even we have uploaded data from snowflake table to s3 bucket.

A screenshot of a computer

Description automatically generated

How many partitions got scanned from snowflake table :355

COPY INTO @DEMO\_DB.PUBLIC.my\_s3\_stage/Customer\_joined\_data

from(

SELECT B.\*

FROM CUSTOMER\_SNOWFLAKE\_TABLE B

LEFT OUTER JOIN

CUSTOMER\_DATA A

ON

A.C\_CUSTOMER\_SK = B.C\_CUSTOMER\_SK);

A screenshot of a computer

Description automatically generated

Number of rows unloaded

10000

Number of bytes unloaded

0.58MB

Scan progress

100.00%

Bytes scanned

0.57MB

External bytes scanned

5.48GB

Percentage scanned from cache

100.00%

Partitions scanned

355

Partitions total

1

6 – Advantages and Disadvantages:

Advantages:

1. Snowflake will not use storage it will take data directly from S3 bucket.
2. Scalability – As for storage we are using S3 bucket and for compute we are using Snowflake so we can scale each component as we need.

Disadvantages:

1. It don’t have clustering key to this data.
2. Setup is complex
3. There must be cost included for this setup.