**ASSIGNMENT 3**

**---------Database creation-----------**

create or replace database Demo\_db;

**---------Table creation-----------**

create or replace table PERSON\_NESTED(person\_data VARIANT);

SELECT \* FROM PERSON\_NESTED;

**----------File Format creation-----------**

create or replace file format DEMO\_DB.PUBLIC.my\_file\_format

type = 'JSON'

compression = 'AUTO'

STRIP\_OUTER\_ARRAY = True;

* Here we have used without storage integration. We can use both method we can create storage integration and then edit the trust policy in role and then add ARN number to stage code. Create stage and add S3 bucket location and add ARN role number.

**--------------Stage creation---------------**

CREATE OR REPLACE STAGE DEMO\_DB.PUBLIC.MY\_S3\_STAGE

URL='s3://sami-s3-bucket/Data/'

CREDENTIALS=(AWS\_KEY\_ID='AKIAXXXXXG6LEYTBRN2' AWS\_SECRET\_KEY='4gYj4mT2FozwYXXXXXrOQGHrJC0bPBs5iL1Tl5/6')

file\_format = my\_file\_format;

**------------to check files present in stage------------------**

list @DEMO\_DB.PUBLIC.MY\_S3\_STAGE;

**---------------create pipe ------------------------**

CREATE OR REPLACE PIPE DEMO\_DB.PUBLIC.person\_pipe AUTO\_INGEST = TRUE AS

COPY INTO PERSON\_NESTED

FROM (

SELECT $1

FROM @DEMO\_DB.PUBLIC.MY\_S3\_STAGE

)

FILE\_FORMAT = (FORMAT\_NAME = 'DEMO\_DB.PUBLIC.my\_file\_format')

ON\_ERROR = 'CONTINUE';

show pipes;

**-------command to refresh pipe ----------------------**

alter pipe person\_pipe refresh;

**------------create stream ----------------------**

create or replace stream DEMO\_DB.PUBLIC.PERSON\_STREAM on table PERSON\_NESTED;

select SYSTEM$PIPE\_STATUS('person\_pipe');

SELECT \* FROM PERSON\_STREAM;

DESC STREAM PERSON\_STREAM;

**-------------------create Task ---------------------**

CREATE OR REPLACE TASK DEMO\_DB.PUBLIC.new\_task

WAREHOUSE='compute\_wh'

SCHEDULE='1 minute'

WHEN SYSTEM$STREAM\_HAS\_DATA('DEMO\_DB.PUBLIC.PERSON\_STREAM') AS

CALL DEMO\_DB.PUBLIC.upload\_person\_data1();

**--------- resume task command. It is important to execute once task is created.--------------**

alter task DEMO\_DB.PUBLIC.new\_task resume;

**-------------------to check task history --------------------------**

SELECT \* FROM TABLE(INFORMATION\_SCHEMA.TASK\_HISTORY(TASK\_NAME=>'new\_task', RESULT\_LIMIT=>10));

select \* from table (information\_schema.  
copy\_history(table\_name=>'person\_nested',start\_time=>dateadd(hours,-1, current\_timestamp())));

**--------create stored procedure -------------------**

CREATE OR REPLACE PROCEDURE DEMO\_DB.PUBLIC.upload\_person\_data1()

RETURNS STRING

LANGUAGE JAVASCRIPT

EXECUTE AS OWNER

AS

$$

var rs = snowflake.execute({sqlText: `

MERGE INTO Demo\_db.public.PERSON\_MASTER F

USING (

SELECT

S.person\_data:ID::INT as ID,

S.person\_data:Name::STRING as Name,

S.person\_data:age::INT as age,

S.person\_data:location::STRING as location,

S.person\_data:zip::STRING as zip

FROM Demo\_db.public.PERSON\_STREAM S

WHERE S.METADATA$ACTION = 'INSERT' OR S.METADATA$ISUPDATE = 'TRUE'

) S

ON F.ID = S.ID

WHEN MATCHED THEN

UPDATE SET

F.Name = S.Name,

F.age = S.age,

F.location = S.location,

F.zip = S.zip

WHEN NOT MATCHED THEN

INSERT (ID, Name, age, location, zip)

VALUES (S.ID, S.Name, S.age, S.location, S.zip);

`});

return `DONE`;

$$;

select \* from PERSON\_NESTED;

A screenshot of a computer

Description automatically generated

select \* from PERSON\_MASTER;

A screenshot of a computer

Description automatically generated

**-------------create two tables--------------**

CREATE OR REPLACE TABLE PERSON\_AGE (

name STRING,

age INT

);

CREATE OR REPLACE TABLE PERSON\_LOCATION (

name STRING,

location STRING

);

**----------------create stored procedure ------------------**

CREATE OR REPLACE PROCEDURE PERSON\_MASTER\_PROCEDURE()

RETURNS STRING

LANGUAGE SQL

EXECUTE AS OWNER

AS

$$

BEGIN

-- Insert data into PERSON\_LOCATION table

INSERT INTO PERSON\_LOCATION (Name, Location)

SELECT Name, Location

FROM PERSON\_MASTER;

-- Insert data into PERSON\_AGE table

INSERT INTO PERSON\_AGE (Name, Age)

SELECT Name, Age

FROM PERSON\_MASTER;

RETURN 'Data load completed successfully';

END;

$$;

**-------------------create task ---------------**

create or replace task load\_person\_data\_task

WAREHOUSE='compute\_wh'

SCHEDULE='1 minute'

AS

CALL DEMO\_DB.PUBLIC.PERSON\_MASTER\_PROCEDURE( );

**--------------alter command to resume -------------------**

alter task DEMO\_DB.PUBLIC.load\_person\_data\_task resume;

**---------------to check task status----------------------**

SELECT \* FROM TABLE(INFORMATION\_SCHEMA.TASK\_HISTORY(TASK\_NAME=>'load\_person\_data\_task', RESULT\_LIMIT=>10));

Select \* from PERSON\_AGE;

A screenshot of a computer

Description automatically generated

SELECT \* FROM PERSON\_LOCATION;

A screenshot of a computer

Description automatically generated