SNOWFLAKE

SNOWFLAKE AND AWS INTEGRATION:

PRIVATE BUCKET:

AWS >> BUCKETS>>>CREATE A BUCKET
>>>ENTER NAME OF BUCKET
>>>AWS REGION
>>>OBJECT ONWERSHIP------ ACL'S DISABLED
>>>BLOCK PUBLIC ACCESS
>>> BUCKET VERSIONING ----DISABLE
>>> ENCRYPTION KEY TYPE ------AMAZON S3-MANAGED KEYS (SSE-S3)
>>> BUCKET KEY------- ENABLE.

>>>OPEN CREATED BUCKET >>>>CHOOSE OBJECT>>> CREATE FOLDER IN OBJECT (CSV DUMPS)>>>OPEN FOLDER AND UPLOAD THE DOCUMENT/FILE IN OBJECT FOLDER
>>>COPY S3 URI (s3://[redacted]/) BY CHECKBOX OBJECT FOLDER

PUBLIC BUCKET:

AWS >> BUCKETS>>>CREATE A BUCKET
>>>ENTER NAME OF BUCKET
>>>AWS REGION
>>>OBJECT ONWERSHIP------ ACL'S DISABLED
>>> ALLOW PUBLIC ACCESS
>>> BUCKET VERSIONING ----DISABLE
>>> ENCRYPTION KEY TYPE ------AMAZON S3-MANAGED KEYS (SSE-S3)
>>> BUCKET KEY------- ENABLE.

>>>OPEN CREATED BUCKET >>>>CHOOSE OBJECT>>> CREATE FOLDER IN OBJECT>>>OPEN FOLDER AND UPLOAD THE DOCUMENT IN OBJECT FOLDER

CREDENTIALS TO ACCESS THE BUCKET/OBJECT FOLDER/ FILE:

AWS>>> I AM>>> ROLES>>>CREATE ROLE
>>>AWS ACCOUNT
>>>REQUIRED EXTERNAL ID---- 00000/0000
>>>PERMISSION POLICIES---SEARCH AMAZONS3FULLACCESS
>>>CHECKBOX AMAZONS3FULLACCESS
>>> ENTER ROLE NAME ----SF_CONNECT
>>>OPEN CREATED ROLE(SF_CONNECT)
>>> COPY ARN
(arn:aws:iam:[redacted])

SNOWFLAKE:

SNOWFLAKE>>> OPEN WORKSHEET>>> USE ACCOUNTADMIN ROLE

```
create database pk_new;
use database pk_new;

create schema pk_schema;
use schema pk_schema;

GRANT USAGE ON Database pk_new TO ROLE SYSADMIN;

CREATE STORAGE INTEGRATION PRIYA_INTIGRATION
TYPE = EXTERNAL_STAGE
STORAGE_PROVIDER = S3
ENABLED = TRUE
STORAGE_AWS_ROLE_ARN ='arn:aws:iam:[redacted]'
STORAGE_ALLOWED_LOCATIONS = ('s3://[redacted]');
```

```
7  DESC INTEGRATION PRIYA_INTIG
```

Objects | Editor | Results | Chart

	property	property_type	property_value	property_de
1	ENABLED	Boolean	true	false
2	STORAGE_PROVIDER	String	S3	
3	STORAGE_ALLOWED_LOCATIONS	List	[redacted]	[]
4	STORAGE_BLOCKED_LOCATIONS	List		[]
5	STORAGE_AWS_IAM_USER_ARN	String	[redacted]	
6	STORAGE_AWS_ROLE_ARN	String	[redacted]	
7	STORAGE_AWS_EXTERNAL_ID	String	[redacted]	
8	COMMENT	String		

Copy the STORAGE_AWS_IAM_USER_ARN and STORAGE_AWS_EXTERNAL_ID from output.

Go to aws>>I am>> role>> open selected role>> trust relationship>> edit trust policy.

```
{
        "Version": "2012-10-17",
        "Statement": [
```

```
                {
                        "Effect": "Allow",
                        "Principal": {
                                "AWS": "arn:aws:iam::232131462640:user/vqn40000-s"
                        },
                        "Action": "sts:AssumeRole",
                        "Condition": {
                                "StringEquals": {
                                        "sts:ExternalId":
"MC92663_SFCRole=2_kVbFnQwL4ELbCIXGq82kbzPQUrw="
                                }
                        }
                }
        ]
}
```

Update policy with replacing copied details of STORAGE_AWS_IAM_USER_ARN and STORAGE_AWS_EXTERNAL_ID from output.

```
--file format object
create or replace file format newformat
type = csv
field_delimiter = ','
skip_header = 1 -- skip the first row
--field_optionally_enclosed_by = ''
-- for multiple lines of data separated by comma engrossed inside''
empty_field_as_null = True;
```

Output: Successfully created file format NEWFORMAT

```
--CREATE TABLE
create or replace table flights_data
(S_no int,  airline varchar, flight varchar, source_city varchar, departure_time varchar, stops
varchar, arrival_time varchar, destination_city varchar, class varchar, duration float, days_left int,
price int);
```

Output: Successfully created FLIGHTS_DATA table

```
--CREATE STAGE
create or replace stage PK_STAGE
url = 's3://priya1privates3/flightscsv/'
storage_integration = PRIYA_INTIGRATION
file_format = newformat
```

Output: successfully created PK_STAGE stage

```
--VIEW THE STAGE
list @PK_STAGE

--COPY DATA FROM STAGE TO TABLE
copy into flights_data FROM @PK_STAGE
ON_ERROR = 'skip_file';
```

Output: successfully LOADED data

```
--VIEW TABLE
SELECT *FROM flights_data;
SELECT COUNT (*) FROM flights_data;
```

Output: 110

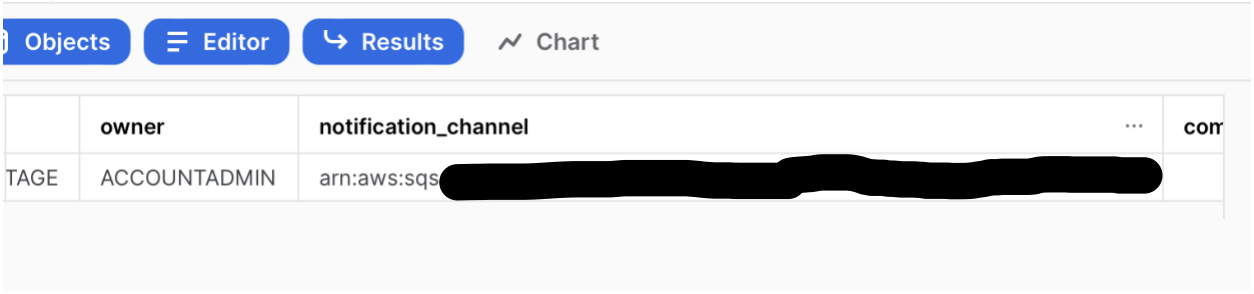
SNOWPIPELINE:

```
--CREATING SNOWPIPE QUERY
CREATE OR REPLACE pipe flightpipe
auto_ingest = TRUE
AS
COPY INTO flights_data
FROM @PK_STAGE.
```

Output: successfully created FLIGHTPIPE pipe

```
--DESCRIBE PIPENAME
DESC pipe flightpipe
```

```
--describe pipename
DESC pipe flightpipe
```



Copy the NOTIFICATION_CHANNEL
GO TO AWS S3 BUCKETS>>> OPEN THE SELECTED BUCKET>>>> PROPERTIES.

>>>> GO TO EVENT NOTIFICATION>>> CREATE EVENT NOTIFICATION
>>>>ENTER EVENT NAME
>>>>OBJECT CREATION
- ALL OBJECT CREATE EVENTS

>>>>DESTINATION --- SQS QUEUE
>>>> SPECIFY SQS QUEUE ---- Enter SQS queue ARN

ENTER THE COPIED NOTIFICATION_CHANNEL DETAIL IN SQS QUEUE ARN AND SAVE CHANGES
UPLOAD THE 2ND SET OF FLIGHTS_DATA (ROWS 108) INTO THE SELECTED S3 BUCKET'S OBJECT FOLDER

```
--VIEW TABLE
SELECT COUNT (*) FROM flights_data;
```
Output: 218

OBSERVATION: 2ND SET OF FLIGHTS_DATA MOVED TO FLIGHTS_DATA TABLE

PARQUET FILE

Jupitor notebook
Reading the parquet file using python
import pyarrow.parquet as pp
trips = pp.read_table ('userdataI.parquet)
trips = tips to.pandas()

>>saving the table as parquet file.
>>go to AWS s3 bucket upload the parquet file.
>>create a new role and copy ARN.
>>go to snowflake and use account admin.

--use database
use database pk_new;

--use schema
use schema pk_schema;

--grant permission
GRANT USAGE ON Database pk_new TO ROLE SYSADMIN;

-- Integration between snowflake and s3
CREATE or replace STORAGE INTEGRATION PK_INTIG
TYPE = EXTERNAL_STAGE
STORAGE_PROVIDER = S3
ENABLED = TRUE
STORAGE_AWS_ROLE_ARN ='arn:aws:iam[redacted]'
STORAGE_ALLOWED_LOCATIONS = ('s3://[redacted]
[redacted]');

Output: Successfully created PK_INTIG storage integration

--view the integration
desc INTEGRATION PK_INTIG ;

From output copy the STORAGE_AWS_IAM_USER_ARN and STORAGE_AWS_EXTERNAL_ID from output.

Go to aws>>I am>> role>> open selected role>> trust relationship>> edit trust policy.

Update policy with replacing copied details of STORAGE_AWS_IAM_USER_ARN and STORAGE_AWS_EXTERNAL_ID from output.

--file format object
create or replace file format newparquet
type = parquet

Output: Successfully created file format NEWPARQUET

--CREATE TABLE
CREATE OR REPLACE TABLE sample_data(registration int, id int, first_name varchar, last_name varchar, email varchar, gender varchar , ip_address varchar, cc varchar, country varchar, birthdate varchar, salary float, title varchar, comments varchar
)

Output: Successfully created SAMPLE_DATA table

--CREATE STAGE
create or replace stage parquet_STAGE
url = 's3://[redacted]'
storage_integration = PK_INTIG
file_format = newparquet

Output: successfully created PARQUET_STAGE stage

--view the stage
list @parquet_STAGE

--copy data from stage to table
copy into sample_data FROM @parquet_STAGE
ON_ERROR = 'skip_file'
match_by_column_name = case_insensitive;

Output: successfully LOADED data

```
select * from sample_data
```