LinkedIn

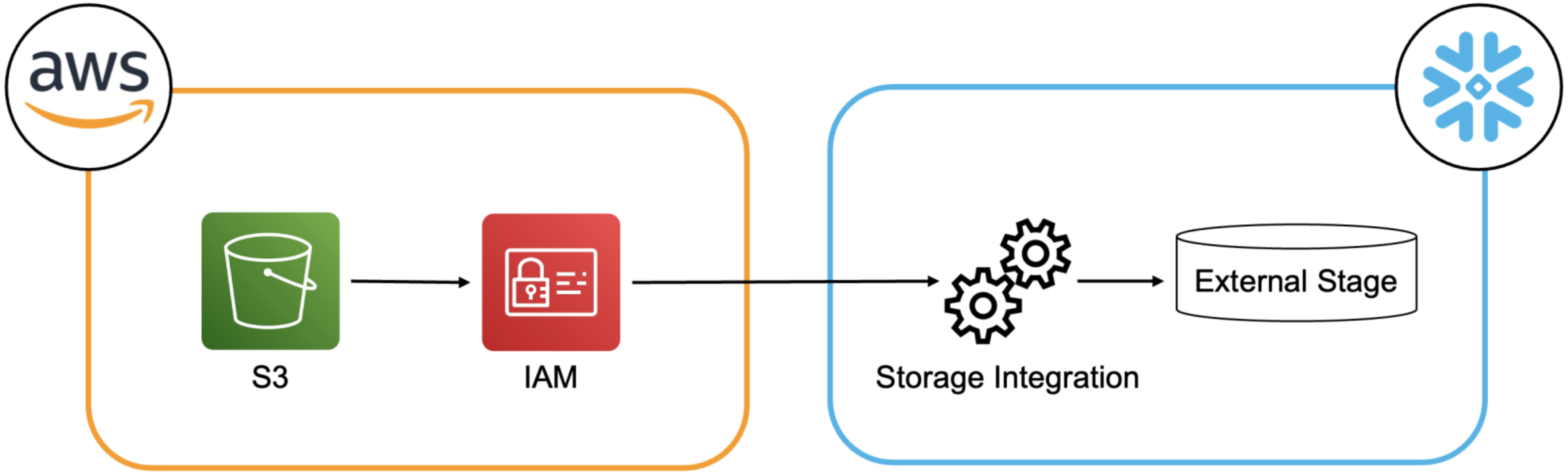
Snowflake: Integration with AWS S3

AWS S3 and **Snowflake** Unite for Smarter Insights

aws | S3 → snowflake

Created By :- Vishal R Nivangune

Snowflake Integration Flow with S3



Requirements

Let's assumes that the following objects are already configured:

- An **AWS S3** bucket with data in it
- A Snowflake user with ACCOUNTADMIN role or another role with granted **CREATE STAGE** privileges

Create an AWS IAM User

1. Go to the AWS **IAM** (Identity and Access Management) console (Type IAM in the search box).
2. Click on **"Users"** in the left navigation pane.
3. Click the **"Create user"** button.
4. Give a name to the user profile and click **"Next"**
5. In the **"Set permissions"** window, click on **"Attach policies directly".**
6. Scroll down and in the **"Permission policies"** tab, select **"AmazonS3FullAccess"** then Click **"Next".**
7. Review the user details, and if everything looks correct, click **"Create user".**
8. Your User is created with permission to access S3 bucket.

Access Key for Snowflake Stage

1. After the user is created, click on the **user's profile**, Which you just created.
2. Scroll down to the **"Security credentials"** tab and click on it.
3. Scroll Down to **"Access keys"** , click Create **"access key"** .
4. Choose a use case for your access key **("Other")** and click "Next".
5. Provide a description for your access key and click **"Create access key".**
6. You will see an option to download the **access key as a CSV file**. Download it, as you will need it when configuring the Snowflake stage.

Let's focus on creating the Snowflake Stage

Stages are a crucial component for managing data loading and unloading processes. They serve as a location where data files can be stored and accessed, facilitating the movement of data between Snowflake and external storage systems.

Snowflake supports two main types of stages:

1. **Internal stages**: - Internal stages store the files internally within Snowflake. Internal stages can be either permanent or temporary.
2. **External stages**: - External stages reference data files stored in a location outside of Snowflake. Currently, Amazon S3 buckets, Google Cloud Storage buckets and Microsoft Azure containers are supported.

AWS S3 Stage Creation In Snowflake Using SQL

Syntax:

```
CREATE STAGE   <stage_name>

URL = 'Copy URL from your S3 bucket'

CREDENTIALS = ( AWS_KEY_ID = ‘<Enter your AWS Access Key ID >’

          AWS_SECRET_KEY = ‘<Enter your AWS Secret Access Key >’  );
```

Step 1: Replace <stage_name> with the desired name for your Snowflake stage.

Step 2: Obtain the URL from your S3 bucket:

- Go to your AWS S3 bucket. Open bucket which contain the files.
- Select the desired file you want to stage in Snowflake.
- Click on the "Copy URL" option

Step 3: Replace 'Copy URL from your S3 bucket' in the SQL statement with the URL you copied in Step 2.

Step 4: Replace 'Enter your AWS access key ID' with the AWS access key ID you obtained when creating the IAM user and key pair in the AWS IAM console.

Step 5: Replace 'Enter your AWS secret access key' with the AWS secret access key that corresponds to the access key ID you entered in Step 4.

Step 5: Execute the SQL statement to create the Snowflake stage. Your stage is now created and ready to use

Table Creation For Stage Data

Step 1: For determining the structure of the data in your Snowflake stage for table creation. You can use the following SQL command to see the rows and columns of the data:

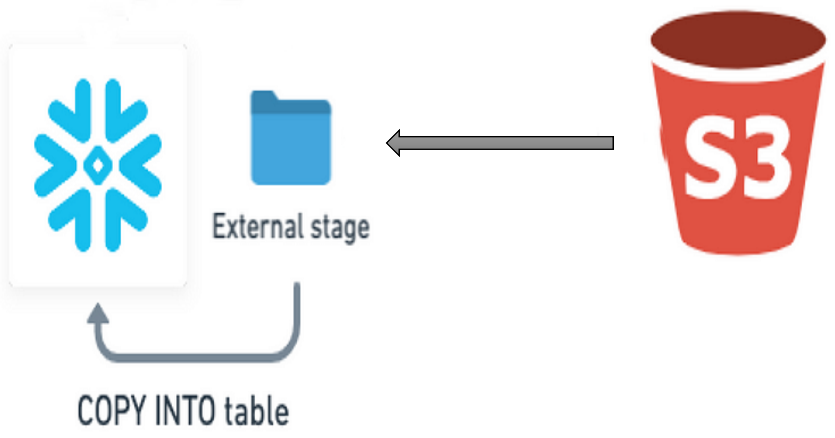
```
SELECT $1, $2, $3……  FROM @<stage_name>;
```

(The $1, $2, etc. notation is used to refer to columns when the actual column names are unknown. This is useful when querying external data sources where you don't have visibility of the actual column names.)

This command will give you a preview of the data and help you decide on the appropriate data types for your table columns based on the observed data.

Step 2: Create a table in Snowflake to store the data. You should use the appropriate data types for each column based on your observations in Step 1.

Step 3: After creating the table, you can use the COPY INTO command to load data from your Snowflake stage into the table.



Syntax:

```
COPY INTO my_table FROM @<stage_name>
FILE_FORMAT = (
  TYPE = CSV
  FIELD_DELIMITER = ','
  SKIP_HEADER = 1
);
```

Let's break down the components of the command:

1. **COPY INTO my_table**: This part of the command specifies the target table where you want to load the data. my_table should be replaced with the name of the Snowflake table where you want to insert the data.
2. **FROM @<stage_name>:** This part of the command specifies the source of the data. @<stage_name> refers to the Snowflake stage you created earlier, from which you want to copy data. <stage_name> should be replaced with the name of your stage.
3. **FILE_FORMAT:** This part of the command allows you to specify how the data in the source files (in the Snowflake stage) is formatted. It consists of several options:

a) **TYPE = CSV :** Indicates that the source files are in CSV (Comma-Separated Values) format. Snowflake will parse the files accordingly.

b) **FIELD_DELIMITER = ‘,’** : Specifies that the comma , character is used as the field delimiter in the CSV files. This means that columns in the CSV files are separated by commas.

c) **SKIP_HEADER = 1** : Indicates that the first row of the CSV files should be skipped during the loading process. This is common when the first row contains headers or metadata, and you want to start loading data from the second row

Putting it all together, this COPY INTO command is essentially instructing Snowflake to take the data from the specified Snowflake stage, interpret it as CSV, use a comma as the field delimiter, and skip the first row when loading it into the specified Snowflake table (my_table).

After executing this command, data from the CSV files in your Snowflake stage will be loaded into the target table, and you'll be able to query and work with it within Snowflake.

Terminologies Used

- **AWS S3 Bucket:** A storage container to store objects in the cloud
- **IAM:** AWS Identity and Access Management (IAM) center is a web-based service that controls access to AWS resources.
- **IAM Policy:** Mechanism to define and enforce permissions on identities such as users, groups, and roles
- **Roles:** Set of permissions that can be delegated to users
- **Users:** An entity that can be used to interact with AWS
- **External Stage**: Use to store metadata about external data files such as S3 bucket URL and file format.

Congratulations!

You've successfully learned how to create a Snowflake stage for AWS S3, examine the data within it, create a Snowflake table, and load data from the S3 stage into the table. With these skills, you're well-equipped to manage and analyze data stored in AWS S3 using Snowflake.

Created By :- Vishal R Nivangune