

Experiment: Data Engineering Pipeline

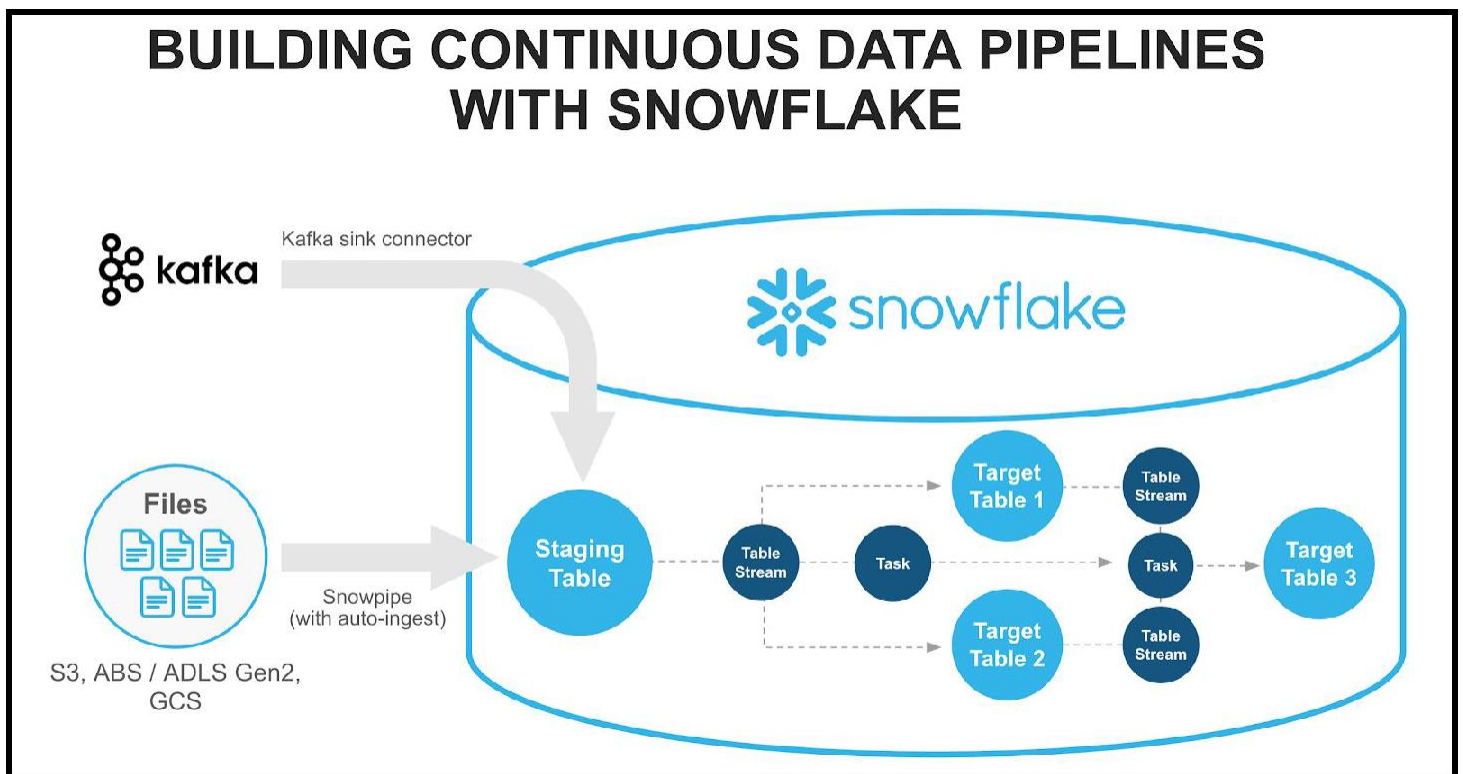
Overview

This experiment demonstrates Snowflake features specifically aligned to “Data Engineering” workload to build modern data pipelines. Data pipelines automate many of the manual steps involved in transforming and optimizing continuous data loads. Frequently, the “raw” data is first loaded temporarily into a staging table used for interim storage and then transformed using a series of SQL statements before it is inserted into the destination reporting tables. The most efficient workflow for this process involves transforming only data that is new or modified.

Snowflake provides the following features to enable continuous data pipelines:

- Continuous data loading using Snowpipe, Snowflake Connector for Kafka, or Third-party data integration tools.
- Change data tracking using Streams.
- Recurring tasks.

The below picture illustrates how to build continuous data pipelines with Snowflake.



Snowflake Connector for Kafka makes it fast and easy to reliably publish continuous streams of records from Kafka to your Snowflake instance for storage and analysis. The Streams and Tasks features enable you to build data pipelines and turn Snowflake into a nimble data transformation engine in addition to a powerful data warehouse.

What you'll learn

This hands-on lab will demonstrate using the following features of the platform:

1. Create a Stage and configure it to automatically call **Snowpipe** using S3 event notifications.
NOTE: This will require the user to create an external AWS S3 bucket in order to demonstrate.
2. **Streams** - Abstraction over staging table used by Snowpipe to allow one time processing of ingested data.
3. **Tasks** (in conjunction with Stored Procedures) - Regularly scheduled execution of transformation logic over data in the stream.

Environment

- We will use our Snowflake free 30-day trial environment.
- AWS Account provided by our session leader.

Experiment 21: Prepare Environment

21.1 Create AWS Account

21.1.1 If you were to do this lab in the future you would have to create or have access to an AWS account. Please create a new account using this link -[Create AWS Account](#). Once logged in to your account select an AWS region closest to your Snowflake account, US-WEST-2 (Oregon), US-EAST-2 (Ohio) and US-EAST-1 (N. Virginia) are good choices.

21.2 Download Snowflake SQL files to your local machine

21.2.1 SQL file **ModernDataPipelines_ExperimentDataSetup.sql** - to create database, warehouse, and data setup for the labs.

21.2.2 SQL file **ModernDataPipelines_ExperimentStoredProcedures.sql** - to create stored procedures for
(1) mimicking periodically arriving data and (2) for cleanup of files that are successfully loaded.

21.2.3 SQL file **ModernDataPipelines_Experiment.sql** - to configure/create Snowpipe, Streams, Tasks, and execute the end-to-end data pipeline.

21.2.4 SQL file **ModernDataPipelines_ExperimentCleanup.sql** - to remove all the Snowflake data/objects.

21.3 Data Setup in Snowflake for the labs

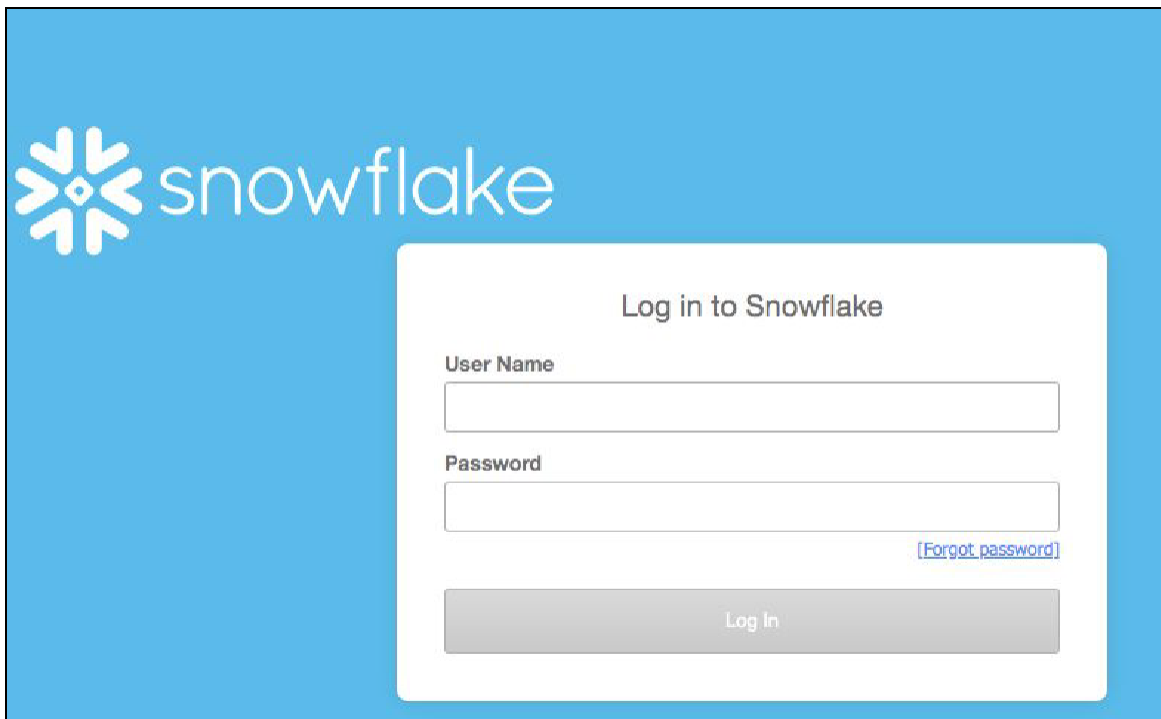


About the screen captures, sample code, and environment

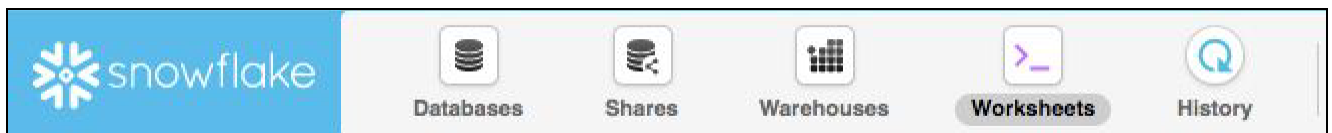
Screen captures in this experiment depict examples and results that may slightly vary from what you may see when you complete the

21.3.1 Open a browser window and enter the URL of your Snowflake 30-day trial environment.

21.3.2 You should see the login screen below. Enter your unique credentials to log in.



The top menu allows you to switch between the different areas of Snowflake:



21.3.3 The **Warehouses** tab is where you set up and manage compute resources (virtual warehouses) to load or query data in Snowflake. Note a warehouse called "COMPUTE_WH (XL)" already exists in your environment.

21.3.4 If you click on the top right of the UI where your username appears, you will see that here you can do things like change your password, roles, or preferences. Snowflake has several system defined roles. You are currently in the default role of SYSADMIN and we will Switch the role to ACCOUNTADMIN for these labs.

ACCOUNTADMIN

For most in these experiment you will remain in the ACCOUNTADMIN role which has privileges to create warehouses and databases and other objects in an account.

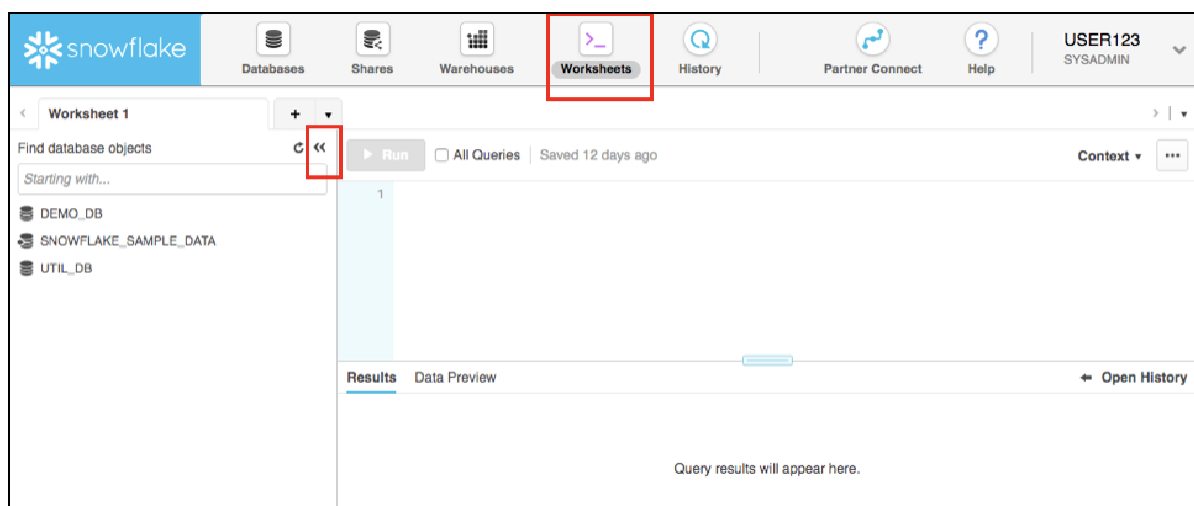


In a real-world environment, you would use different roles for the tasks in this experiment, and assign the roles to your users. More on access control in Snowflake is in towards the end of this experiment and also at <https://docs.snowflake.net/manuals/user-guide/security-access-control.html>

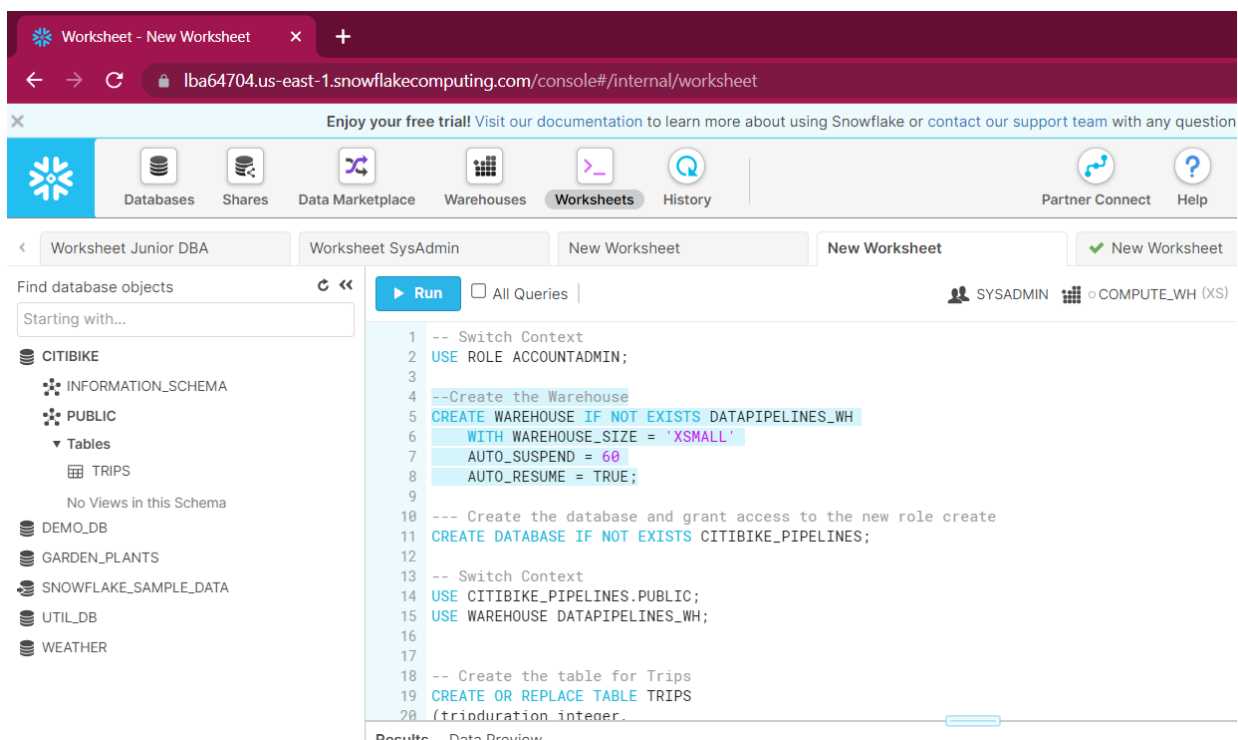
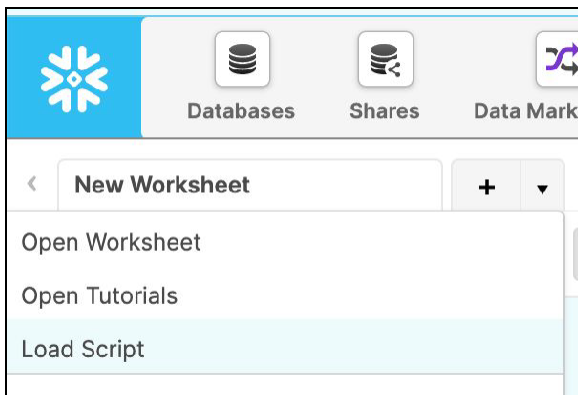
21.3.5 The **Worksheets** tab provides an interface for submitting SQL queries, performing DDL and DML operations and viewing results as your queries/operations complete. The default “Worksheet 1” appears.

In the left pane is the database objects browser which enables users to explore all databases, schemas, tables, and views accessible by the role selected for a worksheet. The bottom pane shows results of queries and operations.

The various windows on this page can be resized by moving the small sliders on them. And if during the lab you need more room to work in the worksheet, collapse the database objects browser in the left pane. Many of the screenshots in this guide will have this database objects browser closed.



21.3.6 At the top left of the default “Worksheet 1,” just to the right of the worksheet tab, click on the small, downward facing arrow, select “Load Script”, then browse to the “**ModernDataPipelines_ExperimentDataSetup.sql**” file you downloaded in the prior section and select “Open”.



Warning - Do Not Copy/Paste SQL From This PDF to a Worksheet

Copy-pasting the SQL code from this PDF into a Snowflake worksheet will result in formatting errors and the SQL will not run correctly. Make sure to use the “Load Script” method just covered above.



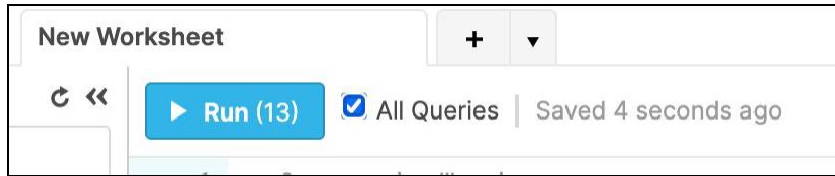
On older or locked-down browsers, this “load script” step may not work as the browser will prevent you from opening the .sql file. If this is the case, open the .sql file with a text editor and then copy/paste all the text from the .sql file to the “Worksheets”



Worksheets vs the UI

Much of the configurations in this lab will be executed via this pre-written SQL in the Worksheets in order to save time. These configurations could also be done via the UI in a less technical manner but would take more time.

21.3.7 All of the SQL commands for the lab data setup will now appear on the new worksheet. **Click the checkbox before “All Queries”** inside the worksheet and execute by clicking on Run.



21.3.8 Ensure the database, tables and data is loaded by reviewing the results.

The screenshot shows the Snowflake web interface. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets, and History. The main area displays a worksheet titled 'New Worksheet' with the following SQL commands:

```
1 -- Switch Context
2 USE ROLE ACCOUNTADMIN;
3
4 --Create the Warehouse
5 CREATE WAREHOUSE IF NOT EXISTS DATAPIPelines_WH
6   WITH WAREHOUSE_SIZE = 'XSMALL'
7   AUTO_SUSPEND = 60
8   AUTO_RESUME = TRUE;
9
10 --- Create the database and grant access to the new role create
11 CREATE DATABASE IF NOT EXISTS CITIBIKE_PIPELINES;
12
13 -- Switch Context
14 USE CITIBIKE_PIPELINES.PUBLIC;
15 USE WAREHOUSE DATAPIPelines_WH;
16
17
18 -- Create the table for Trips
19 CREATE OR REPLACE TABLE TRIPS
20 (tripduration integer);
```

The left sidebar shows the database structure under 'CITIBIKE' and 'PUBLIC' schemas, including tables like 'TRIPS'. The bottom section shows the 'Results' tab, which is currently empty.

21.3.9 At this point the data setup for the labs is complete.

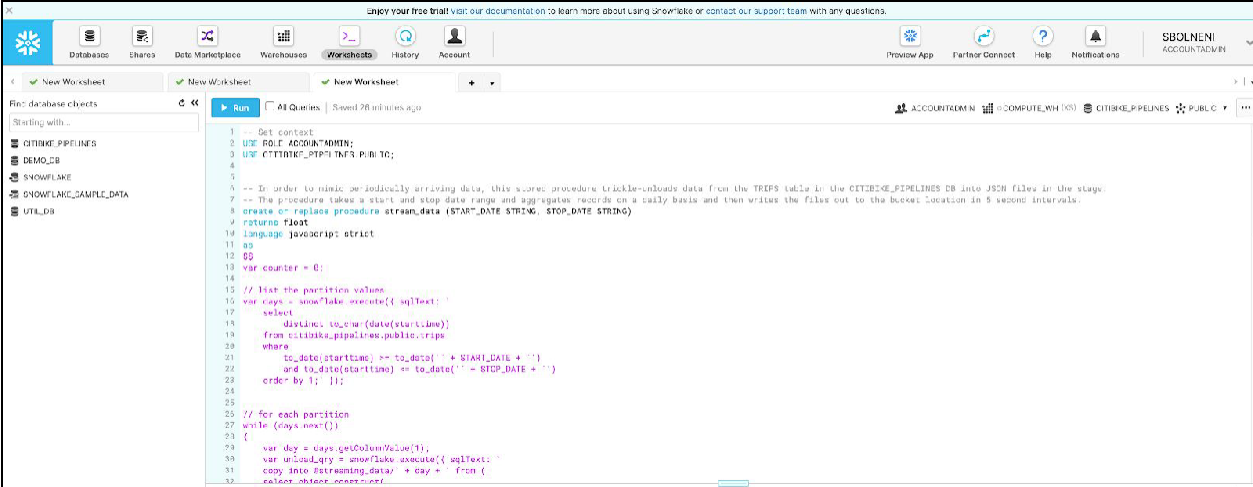
21.4 Stored Procedures for Data Generator & Cleaning Up Stage Files after Successful Load

21.4.1 In this section we will create two new stored procedures for data generation and cleaning up of stage files after successful load.

21.4.2 In order to mimic periodically arriving data, stored procedure 'STREAM_DATA' trickle-unloads data from the TRIPS table in the CITIBIKE_PIPELINES database into JSON files in the external stage which we will create in the next step. The procedure takes a start and stop date range and aggregates records on a daily basis and then writes the files out to the bucket location in 5 second intervals. When it is run, it will produce a steady flow of data arriving in your defined STAGE object.

21.4.3 In this lab, data being written to the External STAGE by the STREAM_DATA stored procedure will be automatically loaded into a table via Snowpipe. The 'PURGE_FILES' stored procedure will be used to clean up files that are successfully loaded. It will compare records entries in the information_schema.copy_history view with the file names still present in the stage. Files that were found to have been loaded are safely deleted by the procedure.

21.4.4 Similar to step 21.4.10 in the worksheets tab, click on the small, downward facing arrow, select "Load Script", then browse to the **"ModernDataPipelines_ExperimentStoredProcedures.sql"** file you downloaded in section 21.3 and select "Open".



The screenshot shows the Snowflake Worksheets interface. On the left, the 'New Worksheet' tab is active. The 'Run' checkbox is checked, and the 'All Queries' checkbox is also checked. The SQL script in the main editor is as follows:

```
1 -- Get context
2 USE ROLE ACCOUNTADMIN;
3 USE CITIBIKE_PIPELINES.PUBLIC;
4
5
6 -- In order to mimic periodically arriving data, this stored procedure trickle-unloads data from the TRIPS table in the CITIBIKE_PIPELINES DB into JSON files in the stage.
7 -- The procedure takes a start and stop date range and aggregates records on a daily basis and then writes the files out to the bucket location in 5 second intervals.
8 create or replace procedure stream_data (START_DATE STRING, STOP_DATE STRING)
9 returns float
10 language javascript strict
11 as
12 $$
13 var counter = 0;
14
15 // List the partition values
16 var cpts = snowflake.execute({ sqlText: '
17 select
18   distinct, to_char(date(starttime))
19   from citibike.pipelines.public.trips
20   where
21     to_date(starttime) >= to_date(' || START_DATE || ')
22     and to_date(starttime) <= to_date(' || STOP_DATE || ')
23   order by 1; });
24
25 // for each partition
26 while (cpts.next())
27 {
28   var day = cpts.getColumnValue(1);
29   var sqlQuery = snowflake.execute({ sqlText: '
30 copy into $streaming_data/ > day = ' from (
31   select, object_construct(
```

21.4.5 All of the SQL commands for setting the context and creating the stored procedures - 'STREAM_DATA', 'PURGE_FILES' will now appear on the new worksheet. **Click the checkbox before "All Queries"** inside the worksheet and execute by clicking on Run.

21.4.6 Ensure the stored procedures are created successfully by reviewing the results of 'show procedures' sql.

The screenshot shows the Snowflake console interface. The top navigation bar includes a 'Run (5)' button, a status bar for 'All Queries', and the current user 'ACCOUNTADMIN' with warehouse 'COMPUTE_WH (XS)' and database 'CITIBIKE_PIPELINES'. The left sidebar displays a tree view of database objects, including 'CITIBIKE', 'CITIBIKE_PIPELINES', and 'SNOWFLAKE'. The main area shows a SQL query being executed. The query sets the context to 'ACCOUNTADMIN' and 'CITIBIKE_PIPELINES.PUBLIC', then creates or replaces a stored procedure named 'stream_data'. The procedure is written in JavaScript and is designed to periodically trickle-unload data from the 'TRIPS' table in the 'CITIBIKE' database. The results section at the bottom shows a 'Data Preview' of the 'show procedures' query, displaying two rows of information about the created procedures.

```

1 -- Set context
2 USE ROLE ACCOUNTADMIN;
3 USE CITIBIKE_PIPELINES.PUBLIC;
4
5
6 -- In order to mimic periodically arriving data, this stored procedure trickle-unloads data from the TRIPS table in the CITIBIKE
7 -- The procedure takes a start and stop date range and aggregates records on a daily basis and then writes the files out to the
8 create or replace procedure stream_data (START_DATE STRING, STOP_DATE STRING)
9 returns float
10 language javascript strict
11 as
12 $$
13 var counter = 0;
14
15 // list the partition values
16 var days = snowflake.execute({ sqlText:
17   select
18     distinct to_char(date(starttime))
19   from citibike_pipelines.public.trips
20
21

```

Results Data Preview

✓ Query ID SQL 41ms 2 rows

Filter result...

| Row | created_on | name | schema_name | is_builtin | is_aggregate | is_ansi | min_num_argum | max_num_argun | arguments |
|-----|-----------------|--------------|-------------|------------|--------------|---------|---------------|---------------|---------------|
| 1 | 2021-11-22 1... | PURGE_FILES | PUBLIC | N | N | N | 2 | 2 | PURGE_FILE... |
| 2 | 2021-11-22 1... | STREAM_DA... | PUBLIC | N | N | N | 2 | 2 | STREAM_DA... |

21.4.7 At this point the creation of stored procedures is complete.

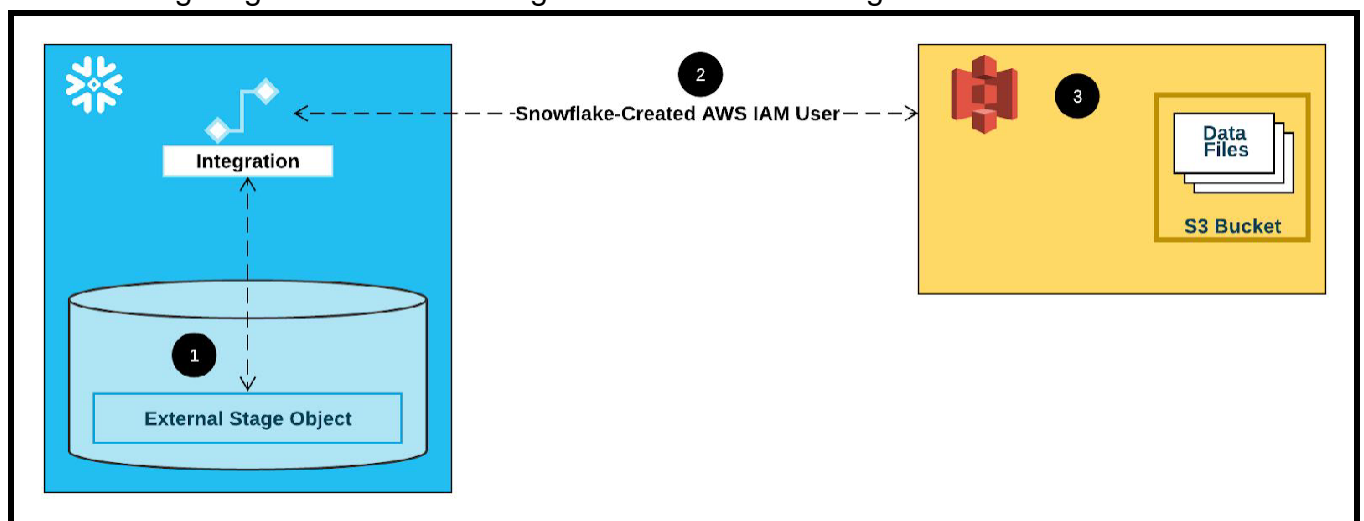
21.5 Configuring Secure Access to Amazon S3

This section describes how to use storage integrations to allow Snowflake to read data from and write data to an Amazon S3 bucket referenced in an external (i.e. S3) stage. Integrations are named, first-class Snowflake objects that avoid the need for passing explicit cloud provider credentials such as secret keys or access tokens. Integration objects store an AWS identity and access management (IAM) user ID.

Note:

Completing the instructions in this section requires permissions in AWS to create and manage IAM policies and roles. If you are not an AWS administrator, ask your AWS administrator to perform these tasks.

The following diagram shows the integration flow for a S3 stage:



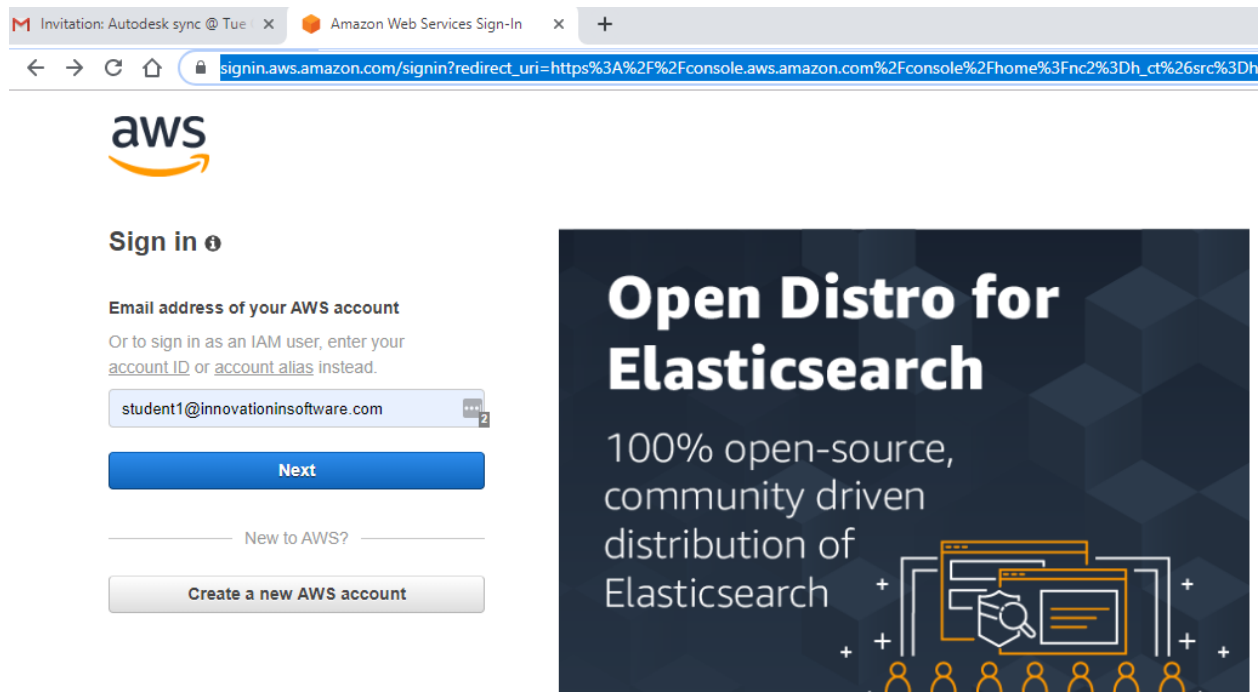
- An external (i.e. S3) stage references a storage integration object in its definition.
- Snowflake automatically associates the storage integration with a S3 IAM user created for your account. Snowflake creates a single IAM user that is referenced by all S3 storage integrations in your Snowflake account.
- An AWS administrator in your organization grants permissions to the IAM user to access the bucket referenced in the stage definition. Note that many external stage objects can reference different buckets and paths and use the same storage integration for authentication.

High level steps involved in this Section:

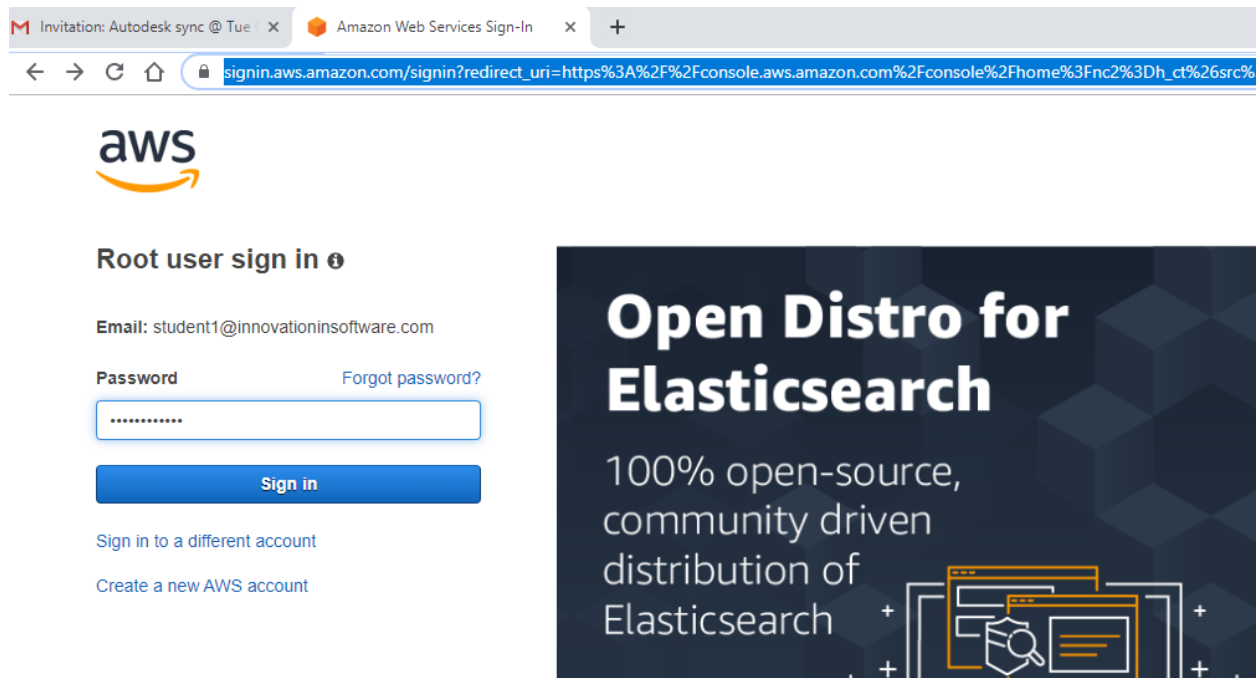
- Step 1: Configure Access Permissions for the S3 Bucket
 - AWS Access Control Requirements
 - Creating an IAM Policy
- Step 2: Create the IAM Role in AWS
- Step 3: Create a Cloud Storage Integration in Snowflake
- Step 4: Retrieve the AWS IAM User for your Snowflake Account
- Step 5: Grant the IAM User Permissions to Access Bucket Objects

21.5.1 Log into the AWS Management Console

https://signin.aws.amazon.com/signin?redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3Fnc2%3Dh_ct%26src%3Dheader-signin%26state%3DhashArgs%2523%26isauthcode%3Dtrue&client_id=arn%3Aaws%3Aiam%3A%3A015428540659%3Auser%2Fhomepage&forceMobileApp=0

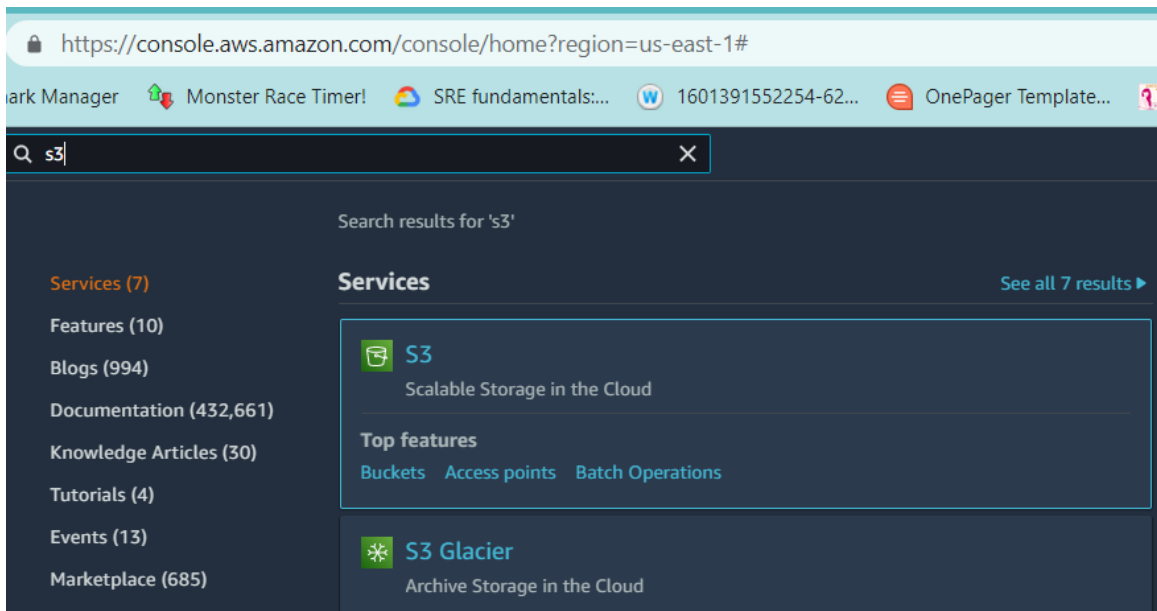


The screenshot shows the AWS Sign-in page in a web browser. The browser's address bar displays the URL: `signin.aws.amazon.com/signin?redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3Fnc2%3Dh_ct%26src%3Dheader-signin%26state%3DhashArgs%2523%26isauthcode%3Dtrue&client_id=arn%3Aaws%3Aiam%3A%3A015428540659%3Auser%2Fhomepage&forceMobileApp=0`. The page features the AWS logo at the top left. Below it, the heading "Sign in" is followed by the instruction "Email address of your AWS account". A subtext reads: "Or to sign in as an IAM user, enter your account ID or account alias instead." A text input field contains the email address "student1@innovationinsoftware.com". Below the input field is a blue "Next" button. Further down, there is a link "New to AWS?" and a button "Create a new AWS account". On the right side of the page, there is a promotional banner for "Open Distro for Elasticsearch" with the text "100% open-source, community driven distribution of Elasticsearch" and an illustration of server racks and a magnifying glass.

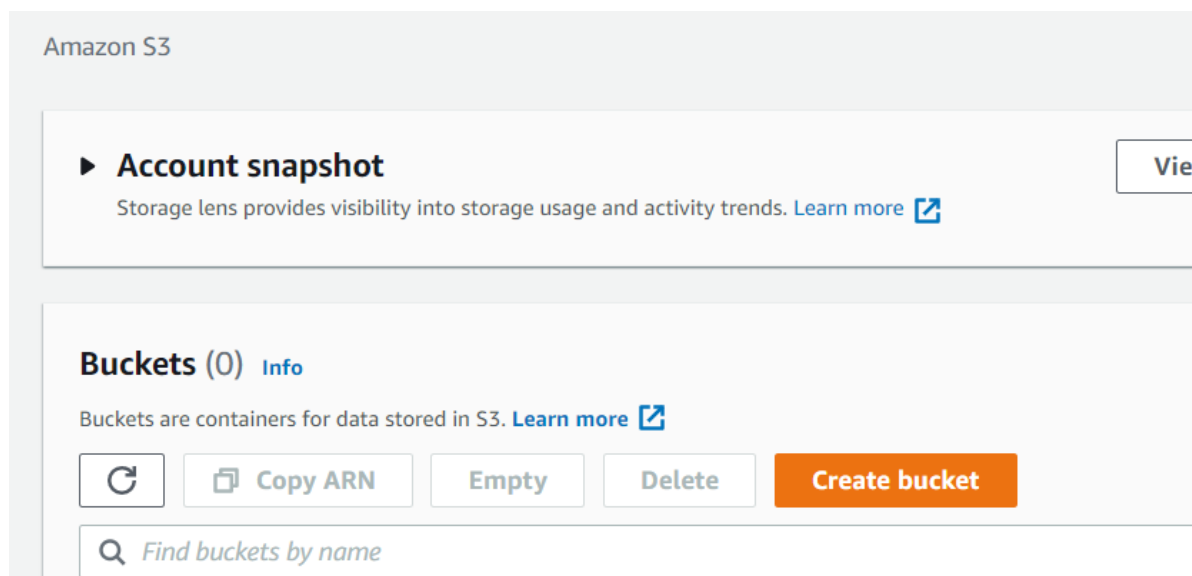


The screenshot shows the AWS Root user sign-in page. The browser's address bar displays the URL: `signin.aws.amazon.com/signin?redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3Fnc2%3Dh_ct%26src%3Dheader-signin%26state%3DhashArgs%2523%26isauthcode%3Dtrue&client_id=arn%3Aaws%3Aiam%3A%3A015428540659%3Auser%2Fhomepage&forceMobileApp=0`. The page features the AWS logo at the top left. Below it, the heading "Root user sign in" is followed by the instruction "Email: student1@innovationinsoftware.com". A subtext reads: "Password". A text input field contains a masked password "*****". Below the input field is a blue "Sign in" button. Further down, there are links "Sign in to a different account" and "Create a new AWS account". On the right side of the page, there is a promotional banner for "Open Distro for Elasticsearch" with the text "100% open-source, community driven distribution of Elasticsearch" and an illustration of server racks and a magnifying glass.

21.5.2 Search for the S3 service and select S3



21.5.3 Create an AWS S3 bucket and a folder to be used as an external stage for Snowflake. Choose **Create bucket**.



21.5.4 Make a note of the S3 bucket/folder which will be used in the below steps. Bucket uniqueness is required globally, so we'll use a simple naming like **snowflake-data-pipeline-session1-george** or similar

Amazon S3 > Create bucket

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

21.5.5 Scroll to the bottom, leaving the remaining options default, and choose **Create bucket**

21.5.6 Select the bucket name that you created

| | Name ▼ | AWS Region ▼ | Access ▼ |
|--|---|------------------------------------|----------------------------------|
| | snowflake-data-pipeline-session1-george | US East (N. Virginia) us-east-1 | Bucket and objects not public |

21.5.7 Choose Create folder

Objects (0)

Objects are the fundamental entities stored in Amazon S3. You can use others to access your objects, you'll need to explicitly grant them permi

Copy S3 URI Copy URL

Actions ▼

Create folder

Upload

Find objects by prefix

21.5.8 Enter the bucket name as **citibike-pipeline**

Folder

Folder name

citibike-pipeline

Folder names can't contain "/". [See rules for naming](#)

21.5.9 Choose **Create folder**

21.5.10 Configure Access Permissions for the S3 Bucket. From the service search enter **IAM**

21.5.11 Choose **Account settings** from the left-hand navigation pane.

21.5.12 Expand the **Security Token Service Regions** list, find the AWS region corresponding to the region where your account is located, and choose **Activate** if the status is Inactive. If we chose the US-EAST-1 region it should already be “Always Active”

▼ Security Token Service (STS)

Session Tokens from the STS endpoints

AWS recommends using regional STS endpoints to reduce latency. Session tokens from regional STS endpoints are v

Session tokens from the global STS endpoint (<https://sts.amazonaws.com>) are valid only in AWS Regions that are enal
use session tokens from regional STS endpoints or activate the global STS endpoint to issue session tokens that are v

| Endpoints | Region compatibility of session tokens | Act |
|--------------------|--|----------------------|
| Global endpoint | Valid only in AWS Regions enabled by default | Edit |
| Regional endpoints | Valid in all AWS Regions | |

Endpoints

You can enable additional endpoints from which you can request temporary credentials. Activate only endpoints you int

| Region name | Endpoint | STS status |
|-----------------------|---|-----------------|
| Global Endpoint | https://sts.amazonaws.com | Always active ⓘ |
| US East (N. Virginia) | https://sts.us-east-1.amazonaws.com | Always active ⓘ |

21.5.13 Choose **Policies** from the left-hand navigation pane.

21.5.14 Choose **Create Policy**

21.5.15 Select the JSON tab

21.5.16 Copy and paste the contents of **s3-iam-role.json** into the policy content area. Choose
Next:Tags

```
Visual editor  JSON
6 ACTION : [
7   "s3:PutObject",
8   "s3:GetObject",
9   "s3:GetObjectVersion",
10  "s3:DeleteObject",
11  "s3:DeleteObjectVersion"
12 ],
13 "Resource": "arn:aws:s3:::<bucket>/<prefix>/*"
14 },
15 {
16   "Effect": "Allow",
17   "Action": [
18     "s3:ListBucket",
19     "s3:GetBucketLocation"
20   ],
21   "Resource": "arn:aws:s3:::<bucket>",
22   "Condition": {
23     "StringLike": {
24       "s3:prefix": [
25         "<prefix>/*"
26       ]
27     }
28   }
29 }
30 ]
31 }
```

Security: 0 Errors: 0 Warnings: 0 Suggestions: 0

Make sure to replace `bucket` and `prefix` with your actual bucket name and folder path prefix. For the example `bucket = snowflake-data-pipeline-session1-george` and the `prefix = citibike-pipeline`

```
Visual editor  JSON
6 ACTION : [
7   "s3:PutObject",
8   "s3:GetObject",
9   "s3:GetObjectVersion",
10  "s3:DeleteObject",
11  "s3:DeleteObjectVersion"
12 ],
13 "Resource": "arn:aws:s3:::snowflake-data-pipeline-session1-george/citibike-pipeline/*"
14 },
15 {
16   "Effect": "Allow",
17   "Action": [
18     "s3:ListBucket",
19     "s3:GetBucketLocation"
20   ],
21   "Resource": "arn:aws:s3:::snowflake-data-pipeline-session1-george",
22   "Condition": {
23     "StringLike": {
24       "s3:prefix": [
25         "citibike-pipeline/*"
26       ]
27     }
28   }
29 }
30 ]
31 }
```


- 21.5.17 Choose **Next: Review**
- 21.5.18 Enter **snowflake_access** as the Policy Name and choose **Create policy**

Create policy 1 2 3

Review policy

Name*
Use alphanumeric and '+-=,@-_' characters. Maximum 128 characters.

Description
Maximum 1000 characters. Use alphanumeric and '+-=,@-_' characters.

Summary

This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition. For details, choose [Show remaining](#). [Learn more](#)

| Service ▾ | Access level | Resource | Request condition |
|--|----------------------------|----------|--------------------------------------|
| Allow (1 of 302 services) Show remaining 301 | | | |
| S3 | Limited: List, Read, Write | Multiple | s3.prefix string like <prefix>/* |

Tags

| Key | Value |
|---------------------------------------|-------|
| No tags associated with the resource. | |

* Required Cancel Previous Create policy

The policy [snowflake_access](#) has been created.

[IAM](#) > Policies

Policies (900) [Info](#)

A policy is an object in AWS that defines permissions.

| | Policy name | Type |
|-----------------------|--|------------------|
| <input type="radio"/> | AWSServiceCatalogPolicy-d39wnz205qeyrk | Customer managed |
| <input type="radio"/> | AWSLambdaBasicExecutionRole-fe87899e-be72-46a6-adfe-481f43dca2e8 | Customer managed |
| <input type="radio"/> | snowflake_access | Customer managed |

Make sure to replace `bucket` and `prefix` with your actual bucket name and folder path prefix. For the example bucket = `snowflake-data-pipeline-session1-george` and the prefix = `citibike-pipeline`

- 21.5.19 Choose Roles from the left-hand navigation pane.

21.5.20 Click the **Create role** button.

21.5.21 Select **Trusted Entity** Page in AWS Management Console


21.5.22 Select **Another AWS account** as the trusted entity type.


In the Account ID field, enter your own AWS account ID temporarily. Later, you will modify the trusted relationship and grant access to Snowflake.


Create role


1234

Select type of trusted entity

**AWS service**
EC2, Lambda and others

**Another AWS account**
Belonging to you or 3rd party

**Web identity**
Cognito or any OpenID provider

**SAML 2.0 federation**
Your corporate directory

Allows entities in other accounts to perform actions in this account. [Learn more](#)

Specify accounts that can use this role

Account ID*

664837035227

i

Options

☒ Require external ID (Best practice when a third party will assume this role)

You can increase the security of your role by requiring an optional external identifier, which prevents "confused deputy" attacks. This is recommended if you do not own or have administrative access to the account that can assume this role. The external ID can include any characters that you choose. To assume this role, users must be in the trusted account and provide this exact external ID. [Learn more](#)

External ID

00000000

Important: The console does not support using an external ID with the Switch Role feature. If you select this option, entities in the trusted account must use the API, CLI, or a custom federation proxy to make cross-account iam:AssumeRole calls. [Learn more](#)

* Required

Cancel

Next: Permissions

21.5.23 Enter a dummy ID such as **0000**. Later, you will modify the trusted relationship and specify the external ID for your Snowflake stage. An external ID is required to grant access to your AWS resources (i.e. S3) to a third party (i.e. Snowflake).

21.5.24 Click the **Next** button.

21.5.25 Locate the policy you created in Step 1: Configure Access Permissions for the S3 Bucket (in this topic), and select this policy.

Create role

1 2 3 4

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy 

Filter policies ▼

Q snowflake

Showing 1 result

| | Policy name ▼ | Used as |
|-------------------------------------|------------------|---------|
| <input checked="" type="checkbox"/> | snowflake_access | None |

* Required

Cancel

Previous

Next: Tags

21.5.26 Click the **Next:Tags** button. Choose **Next: Review**

21.5.27 Review Page in AWS Management Console

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name*
Use alphanumeric and '+-=,@-_' characters. Maximum 64 characters.

Role description
Maximum 1000 characters. Use alphanumeric and '+-=,@-_' characters.

Trusted entities The account 664837035227

Policies [snowflake_access](#) 

Permissions boundary Permissions boundary is not set

* Required

Cancel

Previous

Create role

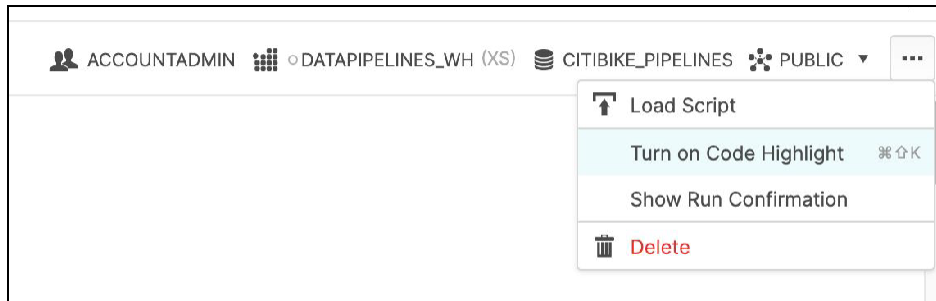
21.5.28 Enter a name **mysnowflakerole** and description for the role, and click the **Create role** button.

You have now created an IAM policy for a bucket, created an IAM role, and attached the policy to the role.

21.5.29 Create a Cloud Storage Integration in Snowflake. [SNOWFLAKE]

Similar to step 21.4.10 in the worksheets tab, click on the small, downward facing arrow, select “Load Script”, then browse to the “**ModernDataPipelines_Experiment.sql**” file you downloaded in section 21.3 and select “Open”.

Before we start using SQL in Worksheets we will turn on Code Highlight by clicking on the 3 dots on the top right hand corner of the worksheet, and then clicking on **Turn on Code Highlight**. This will make it easier to identify the SQL that will be executed.



Set the context for the worksheet by executing the below sql.

```
-- Set the context for the worksheet.
```

```
USE ROLE ACCOUNTADMIN;  
USE CITIBIKE_PIPELINES.PUBLIC;  
USE WAREHOUSE DATAPIPelines_WH;
```

Run the queries by placing your cursor anywhere in the command and clicking the blue “Run” button at the top of the page or by hitting Ctrl/Cmd+Enter on your keyboard



Warning

From here on in the labs, never check the “All Queries” box at the top of the worksheet. We want to run SQL queries one at a time in a specific order; not all at once.

Create Cloud Storage Integration in Snowflake using the below sql. Use the ARN of the IAM role created in step 1.6.3

```
-- Create Cloud Storage Integration.
create or replace storage integration
citibike_snowpipetype = external_stage
storage_provider
= s3enabled =
true
storage_aws_role_arn =
'arn:aws:iam::664837035227:role/mysnowflakerole '
storage_allowed_locations = ('s3://snowflake-
data-pipeline-session1-george/citibike-
pipeline');
```

Note the Storage Integration name created in Snowflake. We need the name in the following sections.

21.5.30 Retrieve the AWS IAM User for your Snowflake Account. **[SNOWFLAKE]**
Follow the instructions outlined in the Snowflake documentation using the link below.
<https://docs.snowflake.com/en/user-guide/data-load-s3-config-storage-integration.html#step-4-retrieve-the-aws-iam-user-for-your-snowflake-account>

```
desc integration citibike_snowpipe;
```

```
desc integration citibike_snowpipe;
```

Data Preview

Query ID: SQL 126ms 8 rows

result...



Copy

| Row | property | property_type ↑ | property_value |
|-----|---------------------------|-----------------|---|
| 4 | STORAGE_BLOCKED_LOCATIONS | List | |
| 2 | STORAGE_PROVIDER | String | S3 |
| 5 | STORAGE_AWS_IAM_USER_ARN | String | arn:aws:iam::941093095223:user/92aj-s-v2sc2810 |
| 6 | STORAGE_AWS_ROLE_ARN | String | arn:aws:iam::664837035227:role/mysnowflakerole |
| 7 | STORAGE_AWS_EXTERNAL_ID | String | LBA64704_SFCRole=3_G31c29cYbWdAYi71EDegGbpEMvM= |

Record the following values from the output of the above sql statement:

STORAGE_AWS_IAM_USER_ARN = arn:aws:iam::941093095223:user/92aj-s-v2sc2810

STORAGE_AWS_EXTERNAL_ID =
LBA64704_SFCRole=3_G31c29cYbWdAYi71EDegGbpEMvM=

21.5.31 Grant the IAM User Permissions to Access Bucket Objects [AWS]


Follow the instructions outlined in the Snowflake documentation using the link below.

<https://docs.snowflake.com/en/user-guide/data-load-s3-config-storage-integration.html#step-5-grant-the-iam-user-permissions-to-access-bucket-objects>

Complete only the instructions for Step5 in the documentation link. After completing step 5, come back to this guide.

Edit the role that we created **mysnowflakerole**. Choose **Trust relationships**

Summary

| | |
|---|---|
| Role ARN | arn:aws:iam::664837035227:role/mysnowflake |
| Role description | Role for Data Pipeline Edit |
| Instance Profile ARNs |  |
| Path | / |
| Creation time | 2021-11-22 21:03 CST |
| Last activity | Not accessed in the tracking period |
| Maximum session duration | 1 hour Edit |
| Give this link to users who can switch roles in the console | https://signin.aws.amazon.com/switchrole?role= |

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

▼ Permissions policies (1 policy applied)

21.5.32 Choose **Edit trust relationship**

Permissions

Trust relationships

Tags

You can view the trusted entities that can assume the

Edit trust relationship

Trusted entities

21.5.33 Paste the updated policy replacing the JSON

Edit Trust Relationship

You can customize trust relationships by editing the following access control policy document.

Policy Document

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "",  
6       "Effect": "Allow",  
7       "Principal": {  
8         "AWS": "arn:aws:iam::941093095223:user/92aj-s-v2sc2810"  
9       },  
10      "Action": "sts:AssumeRole",  
11      "Condition": {  
12        "StringEquals": {  
13          "sts:ExternalId": "LBA64704_SFCRole*"  
14        }  
15      }  
16    }  
17  ]  
18 }
```

21.5.34 Choose **Update Trust Policy**

We just completed configuring a Snowflake storage integration object to delegate authentication responsibility for cloud storage to a Snowflake identity and access management (IAM) entity.

21.6 Creating an External Stage for Snowpipe

21.6.1 Create the streaming_data EXTERNAL STAGE object where the data will land. In the worksheet, execute the below sql statement to create the External Stage object.

```
-- Creating an External Stage.  
-- Replace the s3 bucketname and the folder name as highlighted  
below.
```

```
create or replace stage  
  streaming_data url =  
    's3://<s3bucketname>/<folder>/'  
  storage_integration =  
    citibike_snowpipe  
  file_format=(type=json) ;
```

21.7 Finalizing Setup

21.7.1 Pause to take stock for what has been built so far. An external stage pointing to an S3 bucket was created.

```
show stages like '%STREAMING%';
```