LUMENDATA

Data Sheet

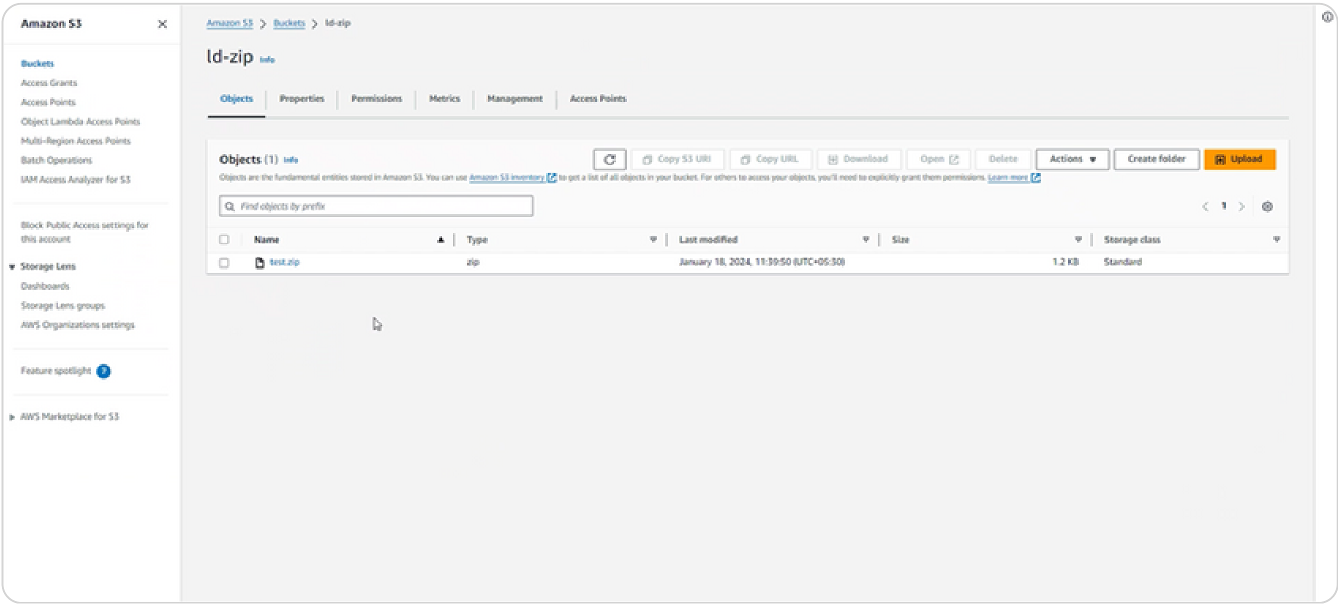
How to Unzip the Zipped folder in AWS S3 from Snowflake

Quick Steps for Seamless Data Access

5201 GREAT AMERICAN PARKWAY, SUITE 320
SANTA CLARA, CA 95054
Tel: (855) 695-8636
E-mail: info@lumendata.com
Website: www.lumendata.com

Let's consider a scenario where we get data for a couple of tables to AWS S3 daily in a zip folder. We will unzip the zipped folder using the Lambda function, which is invoked by an external function from Snowflake.

- Upload the zipped folder to AWS S3 as shown below.

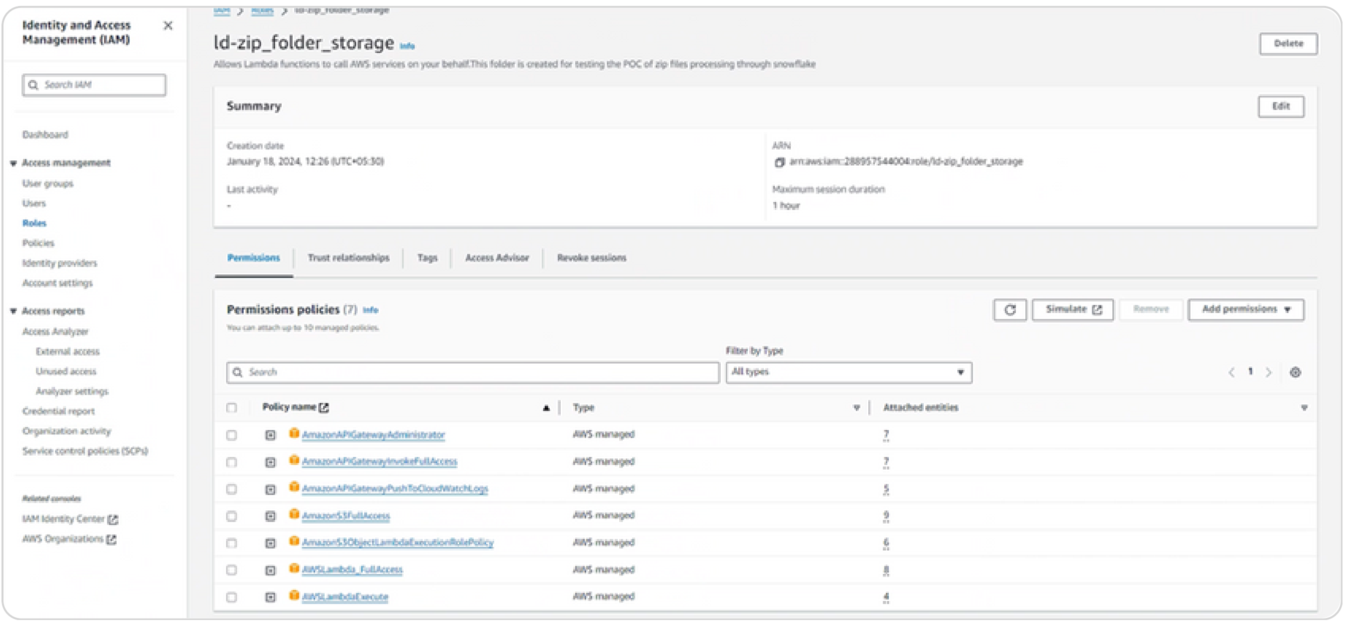


Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
test.zip	zip	January 18, 2024, 11:39:50 (UTC+05:30)	1.2 KB	Standard

- Create an IAM Role that can create the lambda function and execute it. We are creating the role named "**ld-zip_folder_storage**" and attaching the policies to it, as shown below.
  AmazonAPIGatwewayAdministrator
  AmazonAPIGatwewayInvokeFullAccess
  AmazonAPIGatwewayPushToCloudWatchLogs
  AmazonS3FullAccess
  Amazons3ObjectLambdaExecutionRolePolicy
  AWSLambda_FullAccess
  AWSLambdaExecute

LUMENDATA



ld-zip_folder_storage

Allows Lambda functions to call AWS services on your behalf.This folder is created for testing the POC of zip files processing through snowflake

Summary

Creation date: January 18, 2024, 12:26 (UTC+05:30)

ARN: arn:aws:iam::288957544004:role/ld-zip_folder_storage

Last activity: -

Maximum session duration: 1 hour

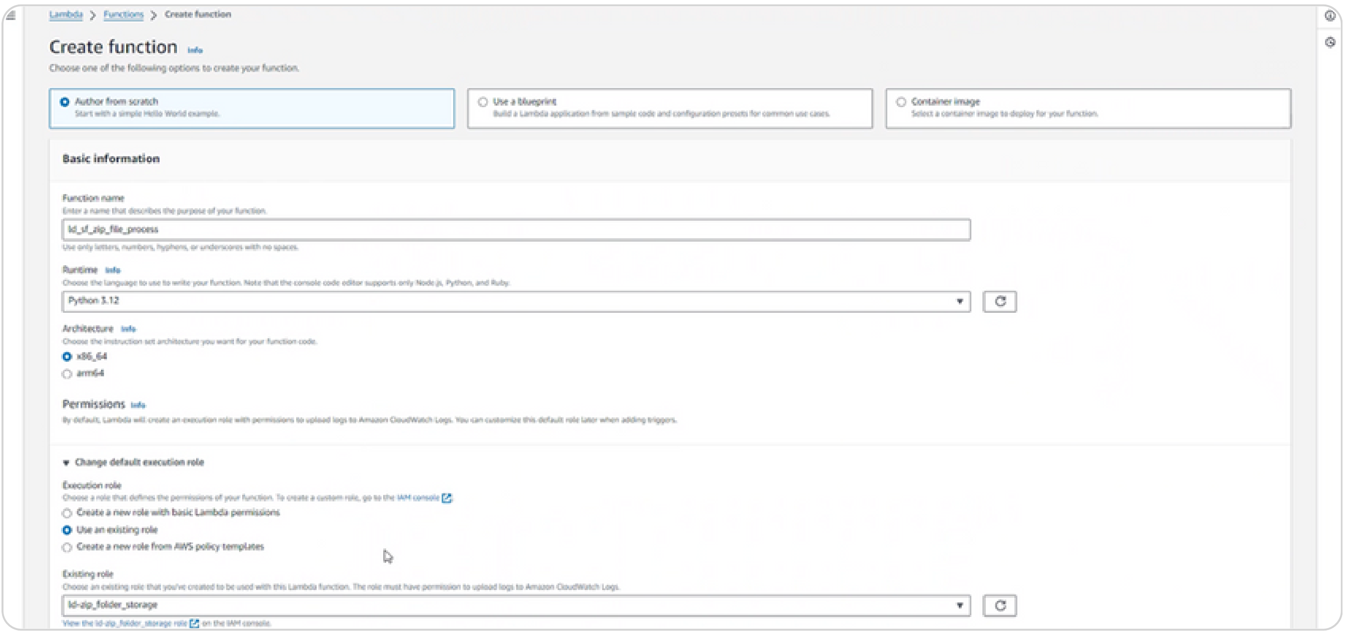
Permissions policies (7)

You can attach up to 10 managed policies.

Filter by Type: All types

Policy name	Type	Attached entities
AmazonAPIGatewayAdministrator	AWS managed	7
AmazonAPIGatewayInvokeFullAccess	AWS managed	7
AmazonAPIGatewayPushToCloudWatchLogs	AWS managed	5
AmazonS3FullAccess	AWS managed	9
AmazonS3ObjectLambdaExecutionRolePolicy	AWS managed	6
AWSLambda_FullAccess	AWS managed	8
AWSLambdaExecute	AWS managed	4

- Create a lambda function named “**ld_sf_zip_file_process**” in Python by using the role that we created above “**ld-zip_folder_storage**”.



Create function

Choose one of the following options to create your function.

[✓ Author from scratch] Use a blueprint / Container image

Basic information

Function name: ld_sf_zip_file_process

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime: Python 3.12

Architecture: [✓ x86_64] arm64

Permissions: By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Change default execution role

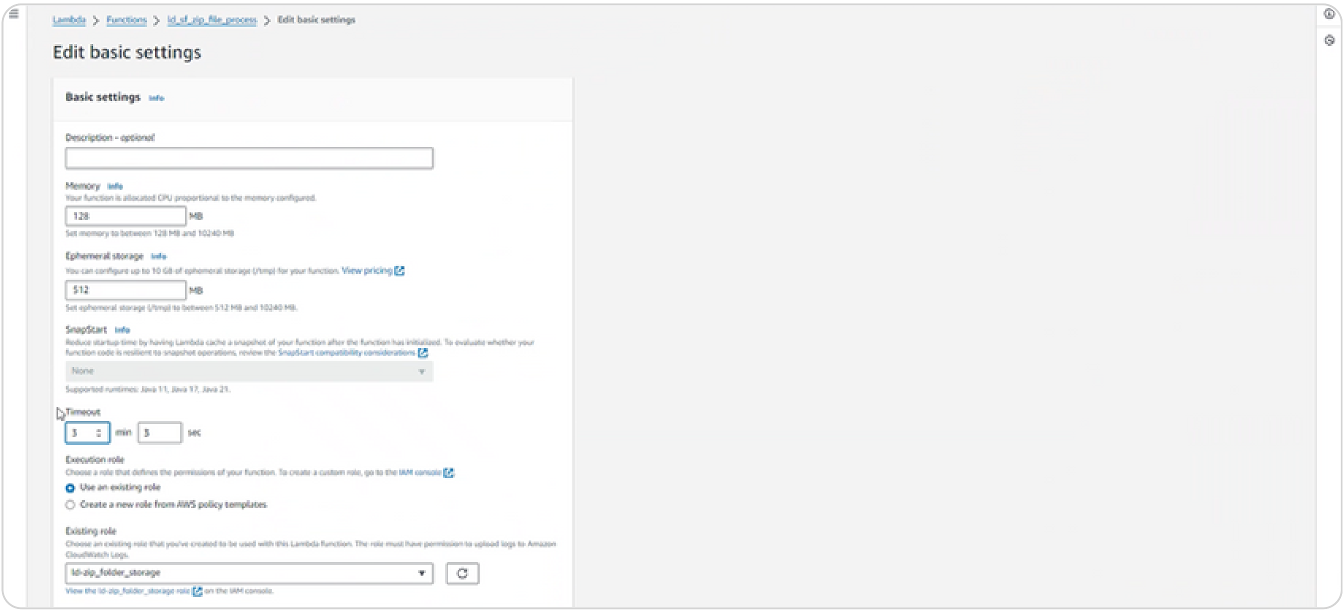
Execution role: Create a new role with basic Lambda permissions / [✓ Use an existing role] / Create a new role from AWS policy templates

Existing role: ld-zip_folder_storage

View the ld-zip_folder_storage role on the IAM console.

- Once we create the lambda function, we need to update the execution time for it.

LUMENDATA



Lambda > Functions > ld_sf_zip_file_process > Edit basic settings

Edit basic settings

Basic settings

Description - optional

Memory
Your function is allocated CPU proportional to the memory configured.
128 MB
Set memory to between 128 MB and 10240 MB

Ephemeral storage
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing
512 MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

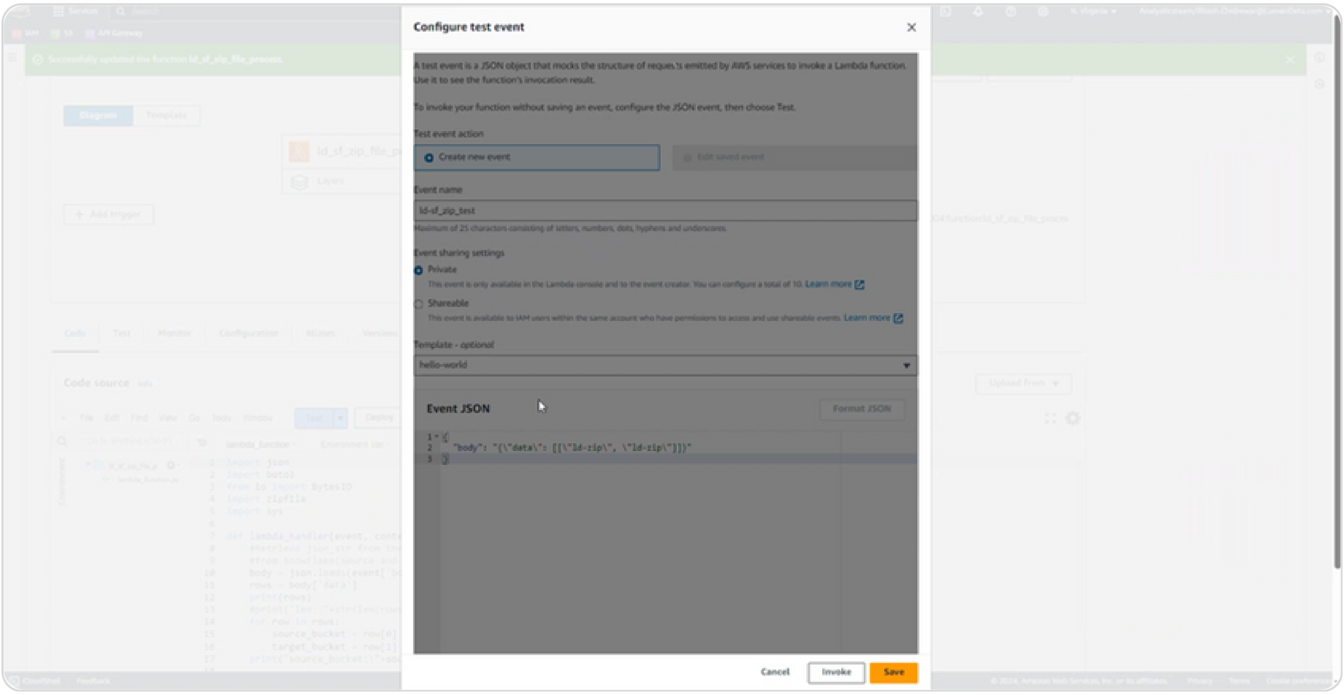
SnapStart
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations
None
Supported runtimes: Java 11, Java 17, Java 21.

Timeout
3 min 3 sec

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console
[✓ Use an existing role]
Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
ld-zip_folder_storage
View the ld-zip_folder_storage role on the IAM console.

- Create a test function to test the lambda function.



Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action
[✓ Create new event] Edit saved event

Event name
ld-sf_zip_test
Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings
[✓ Private]
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. Learn more
Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more

Template - optional
hello-world

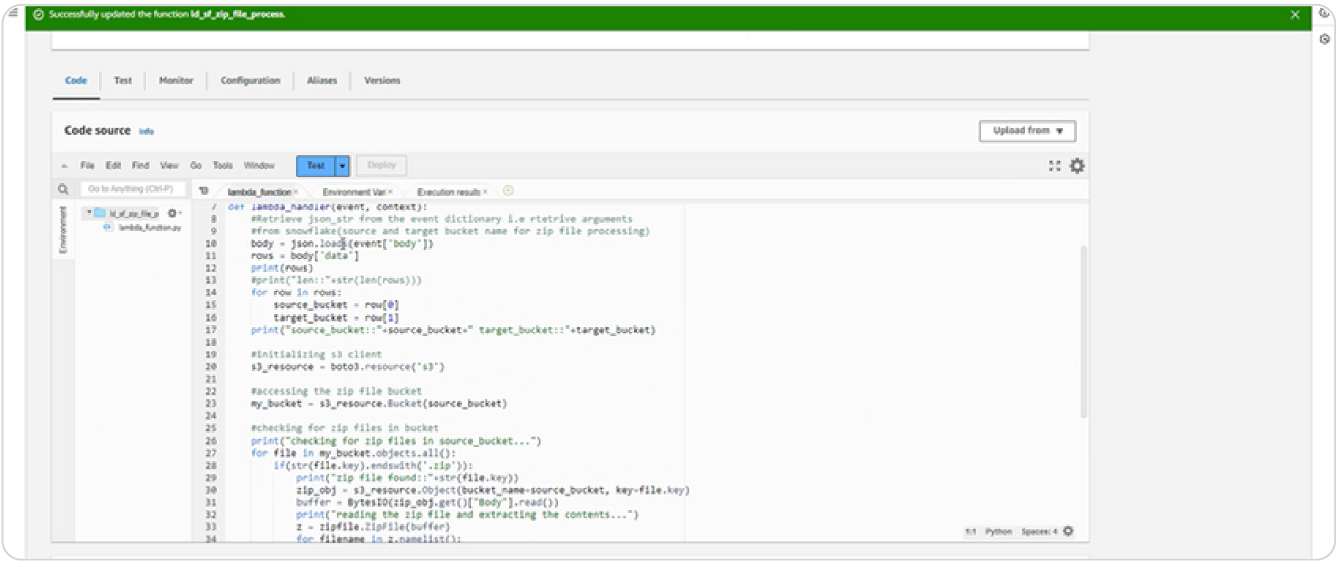
Event JSON

```
{
  "body": "{\"data\": [[\"ld-zip\", \"ld-zip\"]]}"
}
```

- Copy the code as shown below and deploy it:

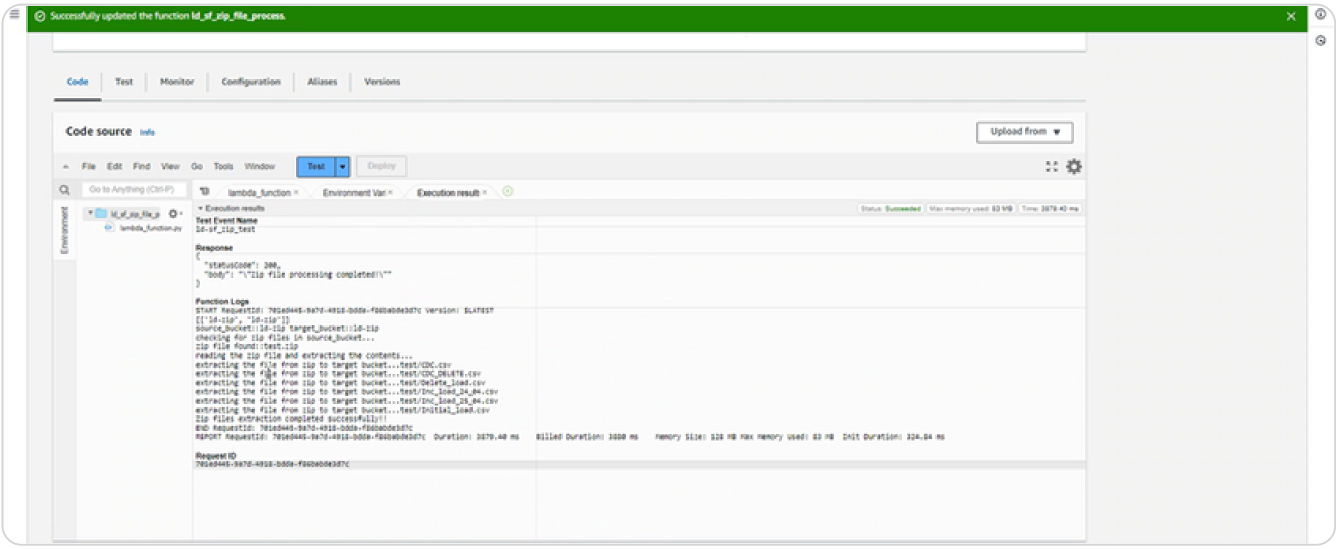
```
import json
import boto3
from io import BytesIO
import zipfile
import sys
```

```
def lambda_handler(event, context):
    #Retrieve json_str from the event dictionary i.e rtetrive arguments
    #from snowflake(source and target bucket name for zip file processing)
    body = json.loads(event['body'])
    rows = body['data']
    print(rows)
    #print("len::"+str(len(rows)))
    for row in rows:
        source_bucket = row[1]
        target_bucket = row[2]
    print("source_bucket::"+str(source_bucket)+" target_bucket::"+str(target_bucket))

    #initializing s3 client
    s3_resource = boto3.resource('s3')

#accessing the zip file bucket
    my_bucket = s3_resource.Bucket(source_bucket)

    #checking for zip files in bucket
    print("checking for zip files in source_bucket...")
    for file in my_bucket.objects.all():
        if(str(file.key).endswith('.zip')):
            print("zip file found::"+str(file.key))
            zip_obj = s3_resource.Object(bucket_name=source_bucket, key=file.key)
            buffer = BytesIO(zip_obj.get()["Body"].read())
            print("reading the zip file and extracting the contents...")
            z = zipfile.ZipFile(buffer)
            for filename in z.namelist():
                file_info = z.getinfo(filename)
                try:
                    print("extracting the file from zip to target bucket..."+filename)
                    response = s3_resource.meta.client.upload_fileobj(
                        z.open(filename),
                        Bucket=target_bucket,
                        Key=f'{filename}'
                    )
                except Exception as e:
                    print("error uplaoding the file to target bucket"+str(e))
            print("Zip files extraction completed successfully!!")
            json_compatible_string_to_return = json.dumps({"data":[[0,str("Zip file
processing completed!")]]})

        else:
            print(file.key+ ' is not a zip file.')

    return {
        'statusCode': 200,
        'body': json_compatible_string_to_return
    }
```



```
def lambda_handler(event, context):
    #Retrieve json_str from the event dictionary i.e rtetrive arguments
    #from snowflake(source and target bucket name for zip file processing)
    body = json.loads(event["body"])
    rows = body['data']
    print(rows)
    #print("len::"+str(len(rows)))
    for row in rows:
        source_bucket = row[0]
        target_bucket = row[1]
    print("source_bucket::"+source_bucket+" target_bucket::"+target_bucket)

    #initializing s3 client
    s3_resource = boto3.resource('s3')

    #accessing the zip file bucket
    my_bucket = s3_resource.Bucket(source_bucket)

    #checking for zip files in bucket
    print("checking for zip files in source_bucket...")
    for file in my_bucket.objects.all():
        if(str(file.key).endswith('.zip')):
            print("zip file found::"+str(file.key))
            zip_obj = s3_resource.Object(bucket_name=source_bucket, key=file.key)
            buffer = BytesIO(zip_obj.get()["Body"].read())
            print("reading the zip file and extracting the contents...")
            z = zipfile.ZipFile(buffer)
            for filename in z.namelist():
```

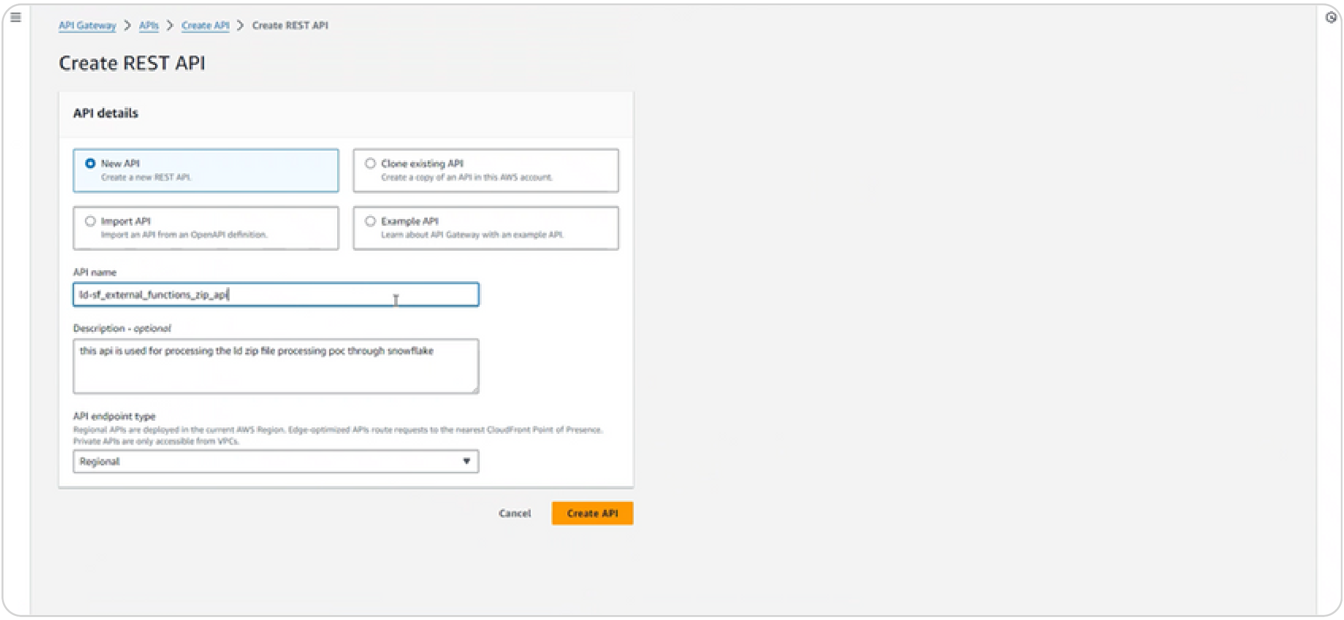
- Test the code in AWS Console to check whether the lambda function is working or not. In the snapshot below, we can see that the lambda code is working, and it has unzipped the zipped folder.



```
Test Event Name
ld-sf_zip_test

Response
{
  "statusCode": 200,
  "body": "\"Zip file processing completed!\""
}

Function Logs
START RequestId: 701ed445-9a7d-4918-bdda-f0dbabde3d7c Version: $LATEST
[['ld-zip', 'ld-zip']]
source_bucket::ld-zip target_bucket::ld-zip
checking for zip files in source_bucket...
zip file found::test.zip
reading the zip file and extracting the contents...
extracting the file from zip to target bucket...test/CDC.csv
extracting the file from zip to target bucket...test/CDC_DELETE.csv
extracting the file from zip to target bucket...test/Delete_load.csv
extracting the file from zip to target bucket...test/Inc_load_24_04.csv
extracting the file from zip to target bucket...test/Inc_load_25_04.csv
extracting the file from zip to target bucket...test/Initial_load.csv
Zip files extraction completed successfully!!
END RequestId: 701ed445-9a7d-4918-bdda-f0dbabde3d7c
REPORT RequestId: 701ed445-9a7d-4918-bdda-f0dbabde3d7c  Duration: 3879.40 ms    Billed Duration: 3880 ms    Memory Size: 128 MB Max Memory Used: 83 MB  Init Duration: 324.84 ms

Request ID
701ed445-9a7d-4918-bdda-f0dbabde3d7c
```

- Next, we create a **REST API** named "**ld-sf_external_functions_zip_api**".

LUMENDATA



API Gateway > APIs > Create API > Create REST API

Create REST API

API details

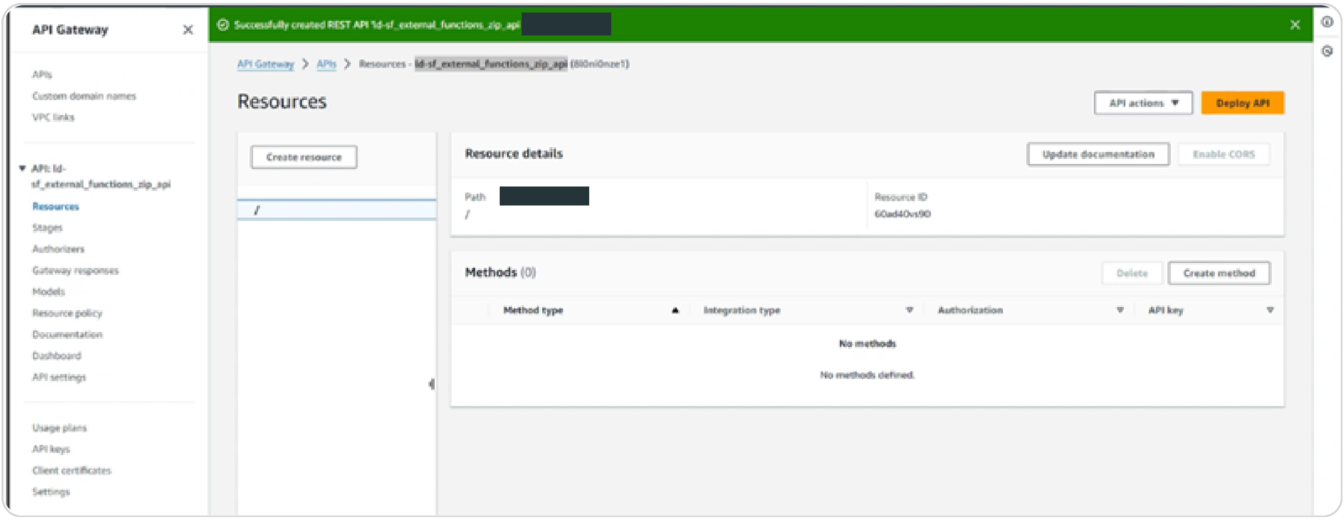
- [✓ New API] Create a new REST API.
- Clone existing API — Create a copy of an API in this AWS account.
- Import API — Import an API from an OpenAPI definition.
- Example API — Learn about API Gateway with an example API.

API name: ld-sf_external_functions_zip_api

Description - optional: this api is used for processing the ld zip file processing poc through snowflake

API endpoint type
Regional APIs are deployed in the current AWS Region. Edge-optimized APIs route requests to the nearest CloudFront Point of Presence. Private APIs are only accessible from VPCs.
Regional

- Create a resource for the **REST API**.



Successfully created REST API 'ld-sf_external_functions_zip_api [redacted]

API Gateway > APIs > Resources - ld-sf_external_functions_zip_api (8l0ni0nze1)

Resources

Resource details

Path [redacted]
/

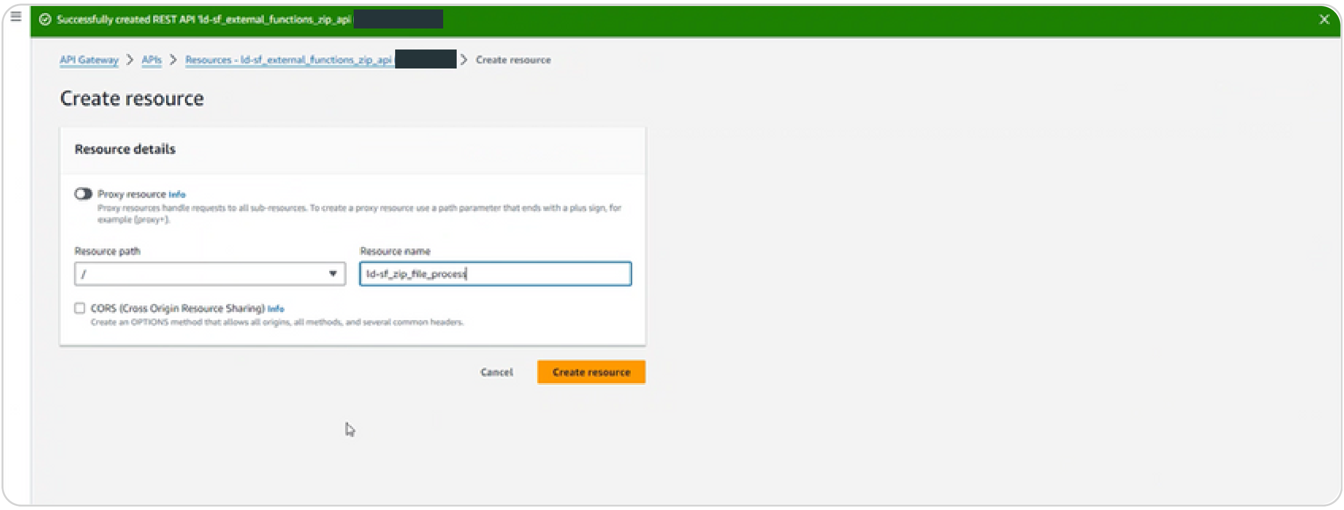
Resource ID
60ad40vs90

Methods (0)

Method type	Integration type	Authorization	API key

No methods

No methods defined.



Successfully created REST API 'ld-sf_external_functions_zip_api [redacted]

API Gateway > APIs > Resources - ld-sf_external_functions_zip_api [redacted] > Create resource

Create resource

Resource details

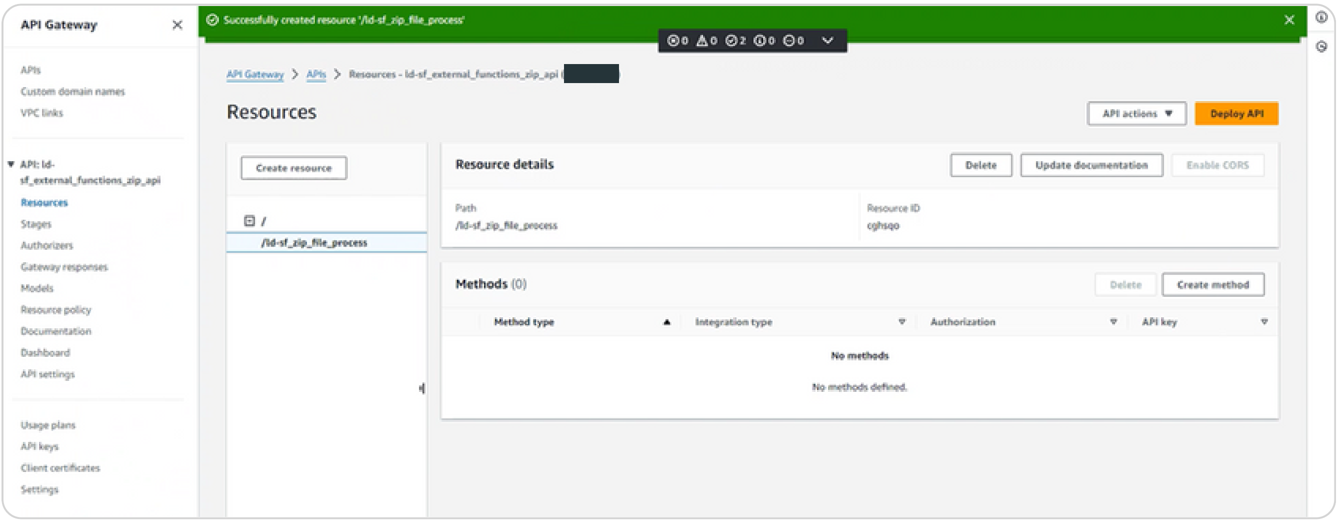
Proxy resource Info
Proxy resources handle requests to all sub-resources. To create a proxy resource use a path parameter that ends with a plus sign, for example {proxy+}.

Resource path: /

Resource name: ld-sf_zip_file_process

CORS (Cross Origin Resource Sharing) Info
Create an OPTIONS method that allows all origins, all methods, and several common headers.

- After creating the resource, create a **METHOD** where we associate the lambda function that we created.



Successfully created resource '/ld-sf_zip_file_process'

API Gateway > APIs > Resources - ld-sf_external_functions_zip_api ([redacted])

Resources

Resource details

Path: /ld-sf_zip_file_process

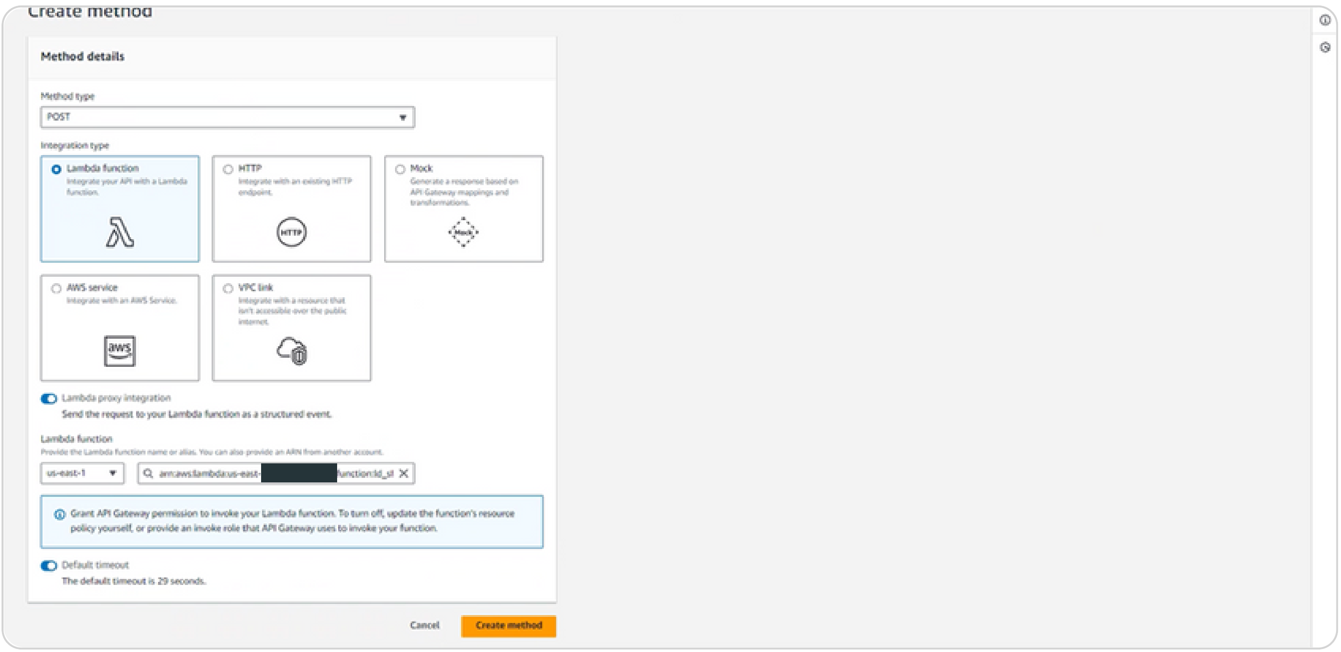
Resource ID: cghsqo

Methods (0)

Method type	Integration type	Authorization	API key

No methods

No methods defined.



Create method

Method details

Method type: POST

Integration type: [✓ Lambda function] Integrate your API with a Lambda function.

HTTP: Integrate with an existing HTTP endpoint.

Mock: Generate a response based on API Gateway mappings and transformations.

AWS service: Integrate with an AWS Service.

VPC link: Integrate with a resource that isn't accessible over the public internet.

Lambda proxy integration: on
Send the request to your Lambda function as a structured event.

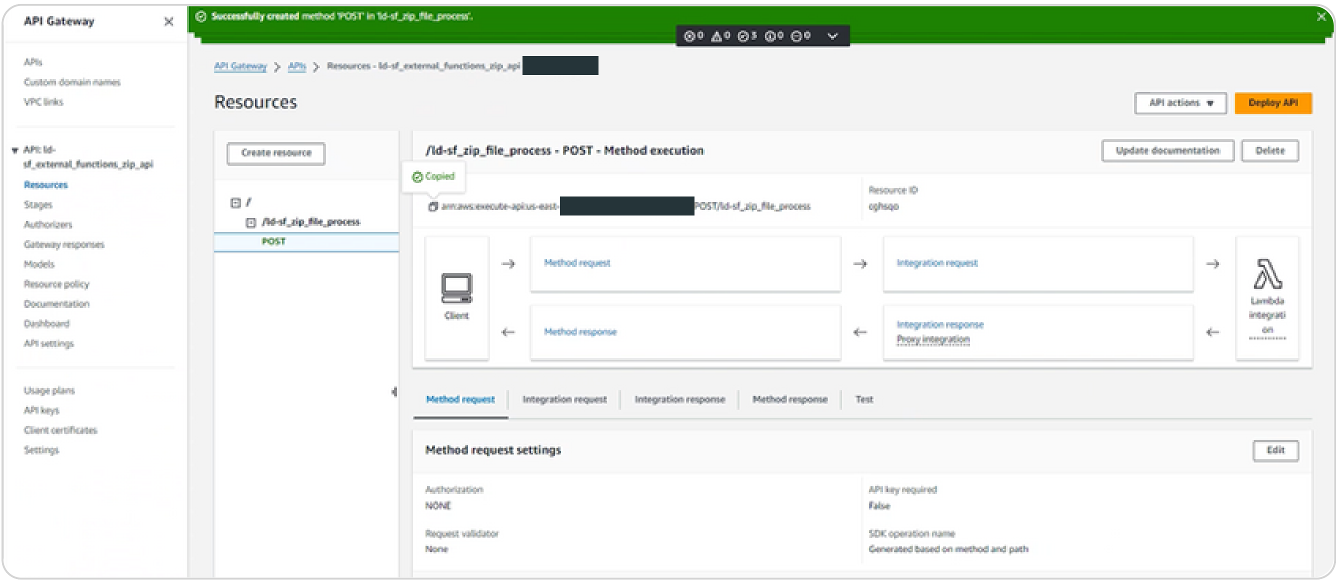
Lambda function
Provide the Lambda function name or alias. You can also provide an ARN from another account.

us-east-1 | arn:aws:lambda:us-east-[redacted]function:ld_sf

Grant API Gateway permission to invoke your Lambda function. To turn off, update the function's resource policy yourself, or provide an invoke role that API Gateway uses to invoke your function.

Default timeout: on
The default timeout is 29 seconds.

- Edit the **METHOD REQUEST** and apply the authorization.



Successfully created method 'POST' in 'ld-sf_zip_file_process'.

API Gateway > APIs > Resources - ld-sf_external_functions_zip_api [redacted]

Resources

/ld-sf_zip_file_process - POST - Method execution

arn:aws:execute-api:us-east-[redacted]POST/ld-sf_zip_file_process

Resource ID
cghsqo

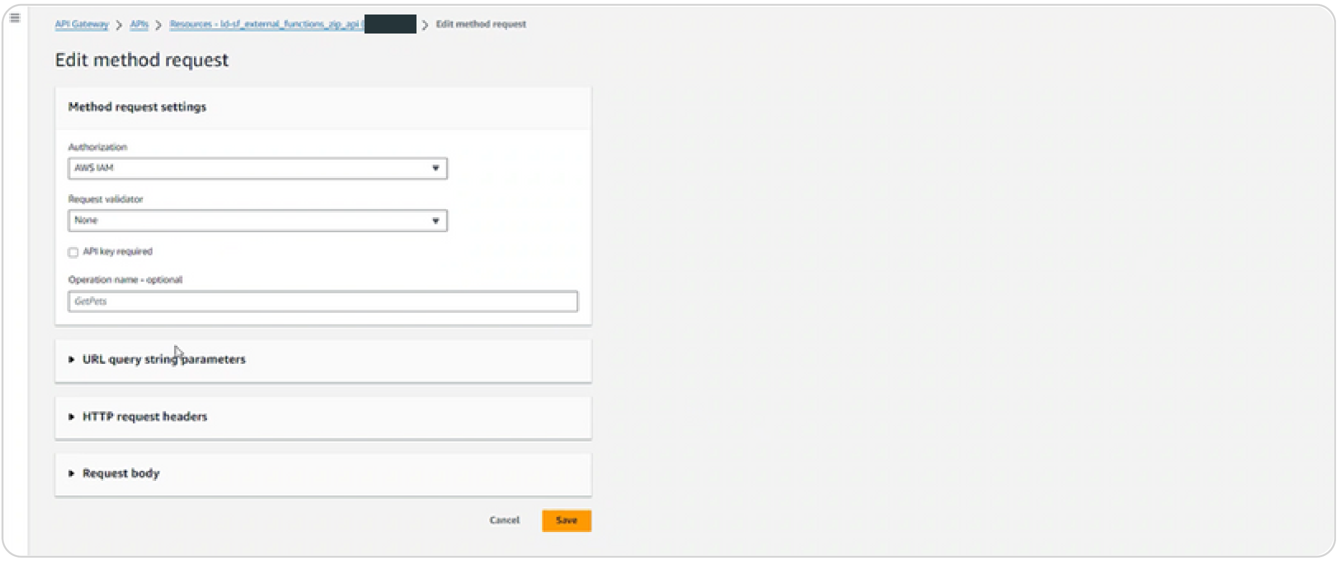
Method request settings

Authorization
NONE

API key required
False

Request validator
None

SDK operation name
Generated based on method and path



API Gateway > APIs > Resources - ld-sf_external_functions_zip_api [redacted] > Edit method request

Edit method request

Method request settings

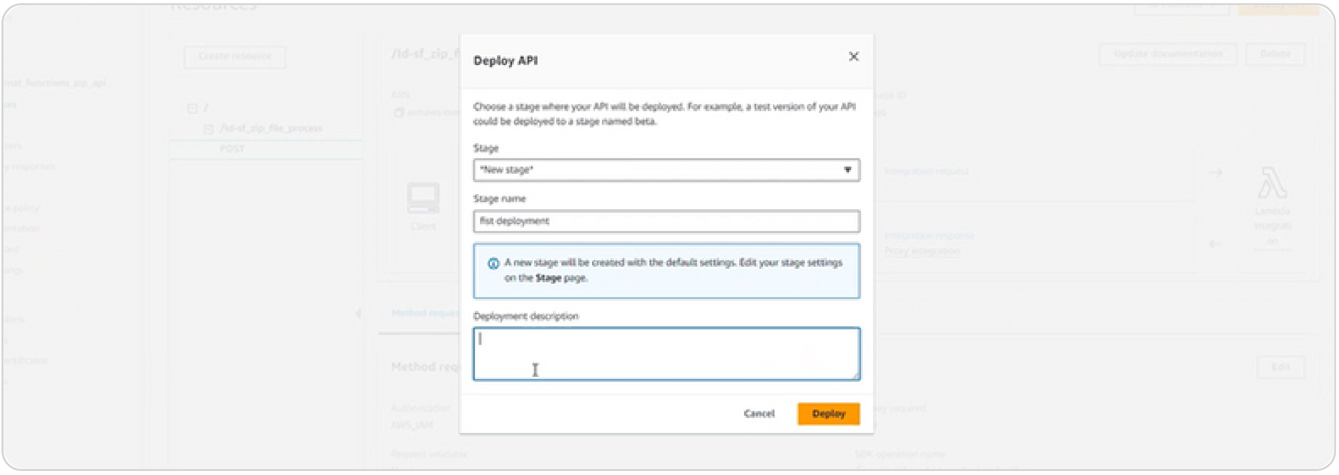
Authorization
AWS IAM

Request validator
None

☐ API key required

Operation name - optional
GetPets

- Create a stage for the **REST API** where we will get the endpoint URL which is supposed to invoke as shown below.



Deploy API

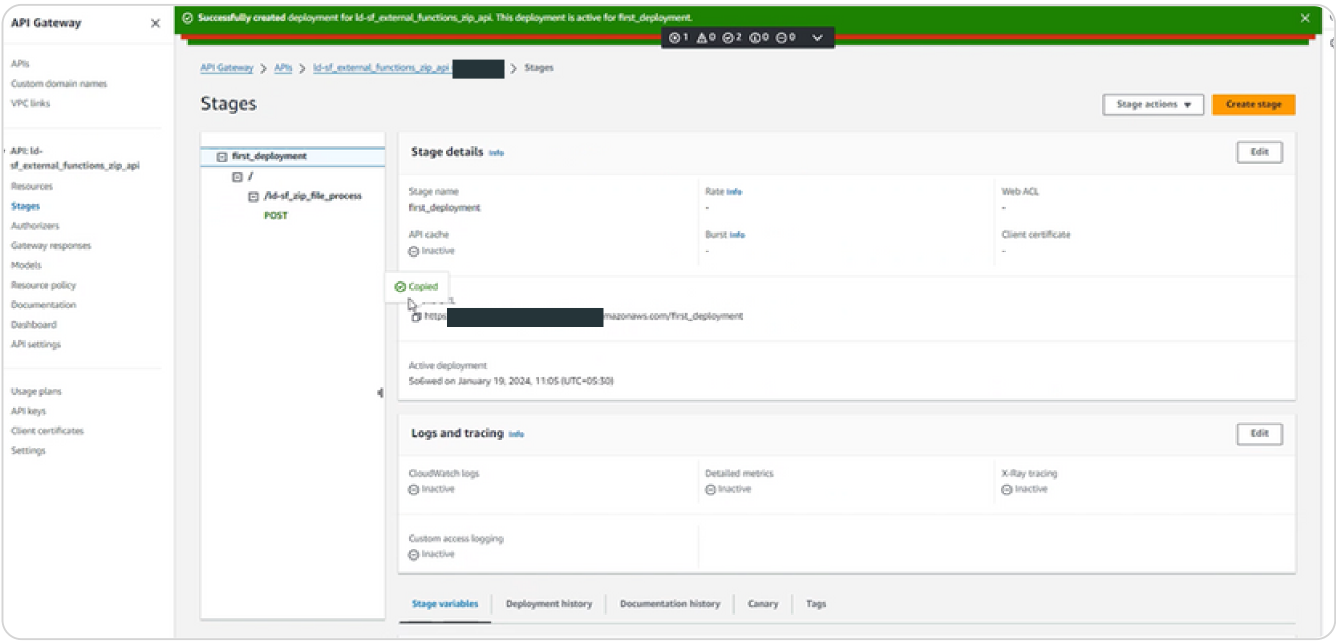
Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Stage
New stage

Stage name
fist deployment

A new stage will be created with the default settings. Edit your stage settings on the **Stage** page.

Deployment description


Successfully created deployment for ld-sf_external_functions_zip_api. This deployment is active for first_deployment.

API Gateway > APIs > ld-sf_external_functions_zip_api [redacted] > Stages

Stages

Stage details

Stage name: first_deployment
Rate: -
Web ACL: -
API cache: Inactive
Burst: -
Client certificate: -

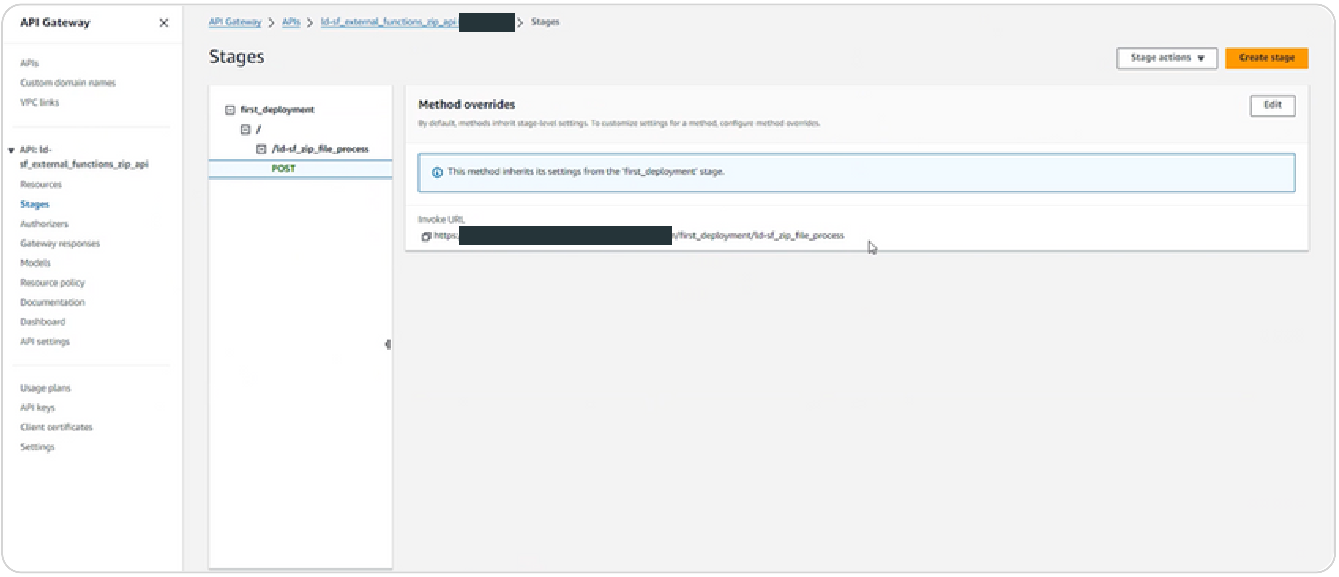
Copied

https[redacted]mazonaws.com/first_deployment

Active deployment
So6wed on January 19, 2024, 11:05 (UTC+05:30)

Logs and tracing

CloudWatch logs: Inactive
Detailed metrics: Inactive
X-Ray tracing: Inactive
Custom access logging: Inactive


API Gateway > APIs > ld-sf_external_functions_zip_api [redacted] > Stages

Stages

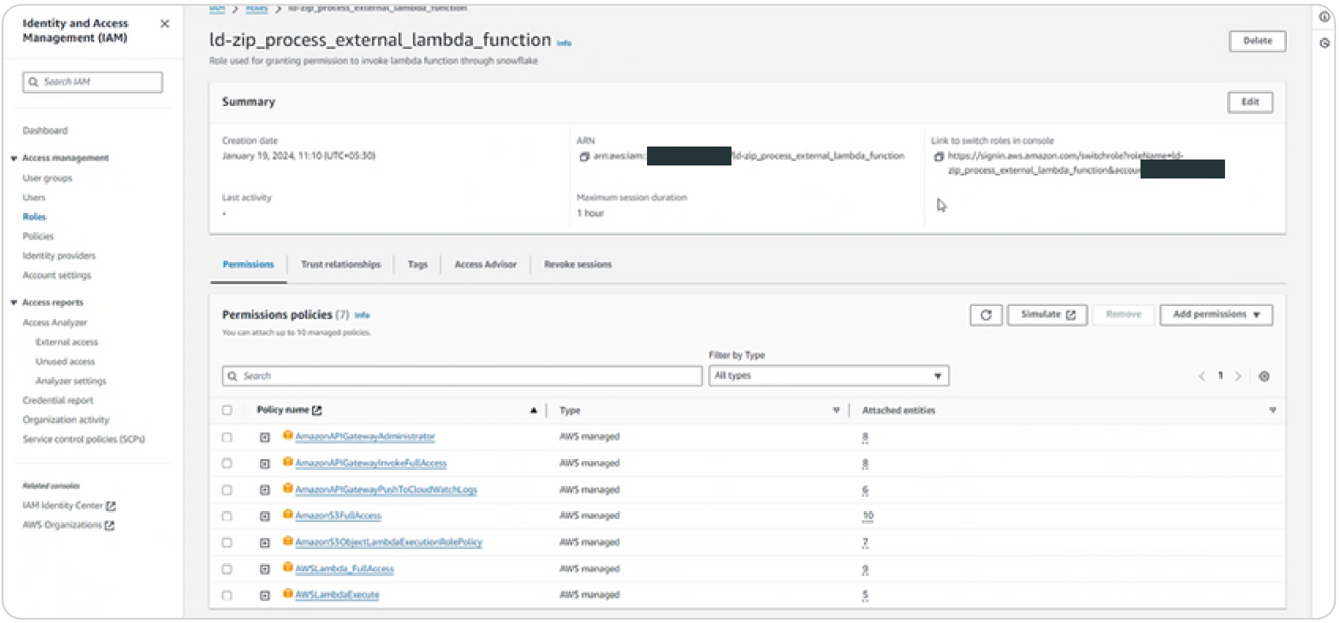
Method overrides

By default, methods inherit stage-level settings. To customize settings for a method, configure method overrides.

This method inherits its settings from the 'first_deployment' stage.

Invoke URL
https:[redacted]/first_deployment/ld-sf_zip_file_process

- So far, we have created the lambda function and **REST API** and associated the lambda function with the **REST API**.
- Now let's create an IAM Role named "**ld-zip_process_external_lambda_function**" .


Identity and Access Management (IAM)

Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
Service control policies (SCPs)
Related consoles
IAM Identity Center
AWS Organizations

IAM > Roles > ld-zip_process_external_lambda_function

ld-zip_process_external_lambda_function

Role used for granting permission to invoke lambda function through snowflake

Summary

Creation date: January 19, 2024, 11:10 (UTC+05:30)
ARN: arn:aws:iam:[redacted]ld-zip_process_external_lambda_function
Link to switch roles in console: https://signin.aws.amazon.com/switchrole?roleName=ld-zip_process_external_lambda_function&accou[redacted]
Last activity: -
Maximum session duration: 1 hour

Permissions | Trust relationships | Tags | Access Advisor | Revoke sessions

Permissions policies (7)

You can attach up to 10 managed policies.

Filter by Type: All types

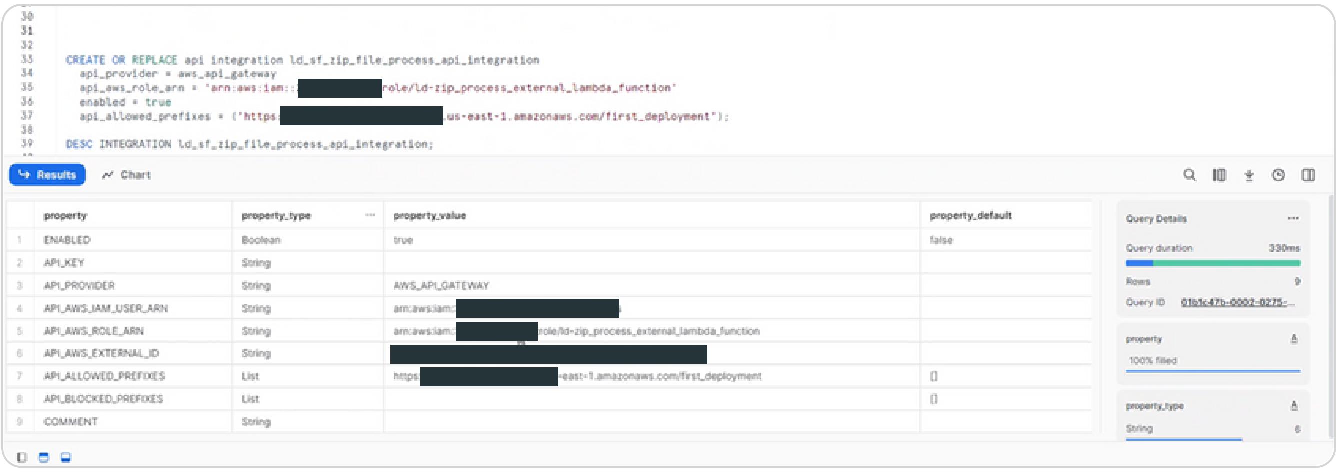
Policy name	Type	Attached entities
AmazonAPIGatewayAdministrator	AWS managed	8
AmazonAPIGatewayInvokeFullAccess	AWS managed	8
AmazonAPIGatewayPushToCloudWatchLogs	AWS managed	6
AmazonS3FullAccess	AWS managed	10
AmazonS3ObjectLambdaExecutionRolePolicy	AWS managed	7
AWSLambda_FullAccess	AWS managed	9
AWSLambdaExecute	AWS managed	5

- Create an **API INTEGRATION** in Snowflake named "ld_sf_zip_file_process_api_integration".

  CREATE OR REPLACE api integration ld_sf_zip_file_process_api_integration
  api_provider = aws_api_gateway
  api_aws_role_arn = 'arn:aws:iam::<AWS_ACCOUNTNAME>:role/ld-zip_process_external_lambda_function'
  enabled = true
  api_allowed_prefixes = ('https:[redacted]

- Describe the **API INTEGRATION** and get the **API_AWS_IAM_USER_ARN** and **API_AWS_EXTERNAL_ID**.

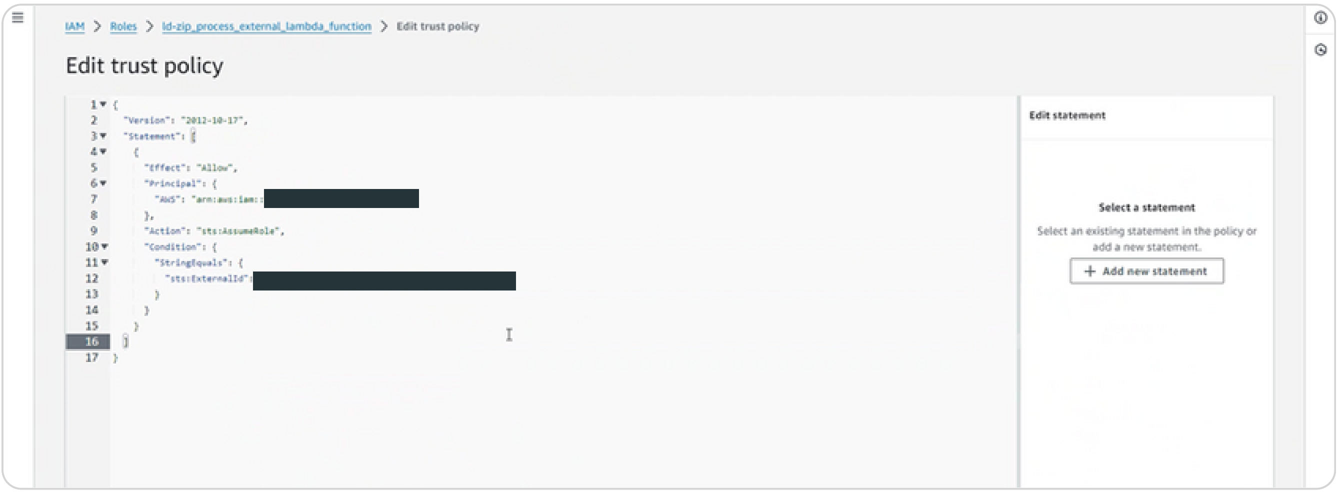
  DESC INTEGRATION ld_sf_zip_file_process_api_integration;


```
CREATE OR REPLACE api integration ld_sf_zip_file_process_api_integration
  api_provider = aws_api_gateway
  api_aws_role_arn = 'arn:aws:iam::[redacted]role/ld-zip_process_external_lambda_function'
  enabled = true
  api_allowed_prefixes = ('https:[redacted].us-east-1.amazonaws.com/first_deployment');

DESC INTEGRATION ld_sf_zip_file_process_api_integration;
```

	property	property_type	property_value	property_default
1	ENABLED	Boolean	true	false
2	API_KEY	String		
3	API_PROVIDER	String	AWS_API_GATEWAY	
4	API_AWS_IAM_USER_ARN	String	arn:aws:iam:[redacted]	
5	API_AWS_ROLE_ARN	String	arn:aws:iam:[redacted]role/ld-zip_process_external_lambda_function	
6	API_AWS_EXTERNAL_ID	String	[redacted]	
7	API_ALLOWED_PREFIXES	List	https[redacted]-east-1.amazonaws.com/first_deployment	[]
8	API_BLOCKED_PREFIXES	List		[]
9	COMMENT	String		

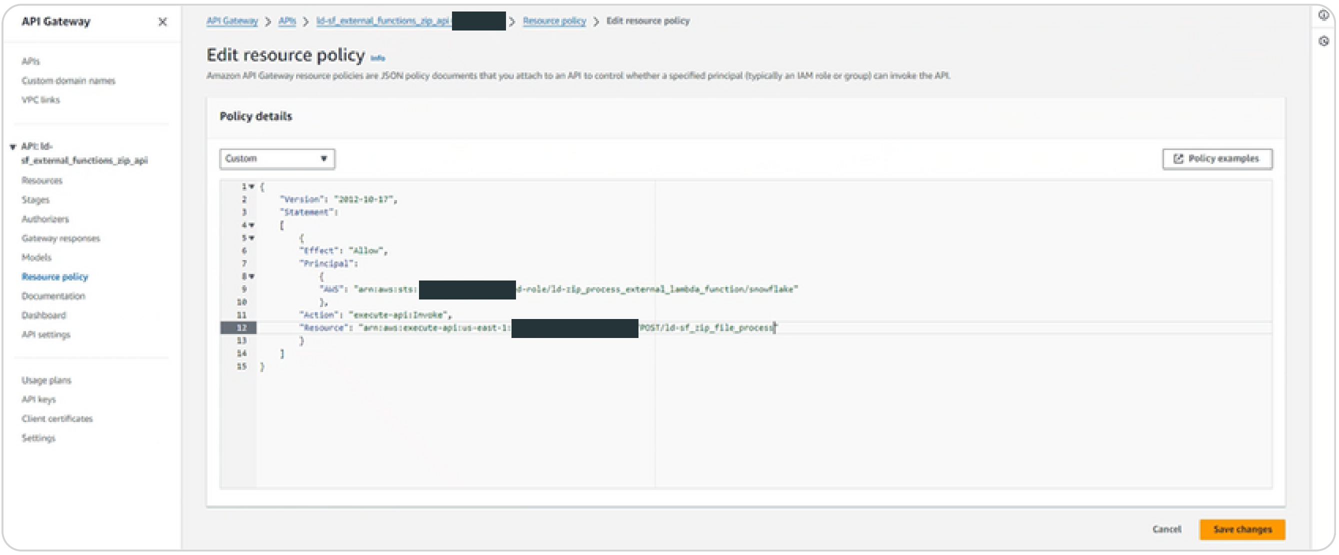
Query Details
Query duration 330ms
Rows 9
Query ID 01b1c47b-0002-0275-...

- Update the IAM Role policy with API_AWS_IAM_USER_ARN and API_AWS_EXTERNAL_ID with AWS and sts:ExternalId respectively.


IAM > Roles > ld-zip_process_external_lambda_function > Edit trust policy
Edit trust policy
Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
+ Add new statement

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::[redacted]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "sts:ExternalId":[redacted]
        }
      }
    }
  ]
}
```

- Update the resource policy for REST API as shown below such that Snowflake will be able to call the end point URL. Update the Snowflake IAM Role 'arn' to 'AWS' and 'API arn' to 'Resource'

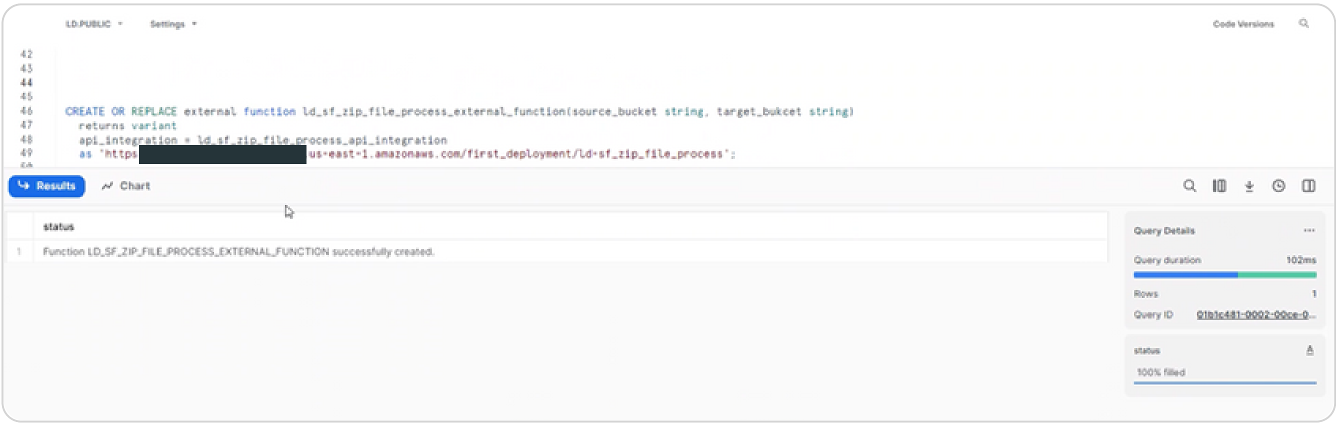


Amazon API Gateway resource policies are JSON policy documents that you attach to an API to control whether a specified principal (typically an IAM role or group) can invoke the API.

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
        "Effect": "Allow",
        "Principal":
            {
            "AWS": "arn:aws:sts:[redacted]d-role/ld-zip_process_external_lambda_function/snowflake"
            },
        "Action": "execute-api:Invoke",
        "Resource": "arn:aws:execute-api:us-east-1:[redacted]POST/ld-sf_zip_file_process"
        }
    ]
}
```

- Create an external function named "ld_sf_zip_file_process_external_function".

CREATE OR REPLACE external function ld_sf_zip_file_process_external_function(source_bucket string, target_bukcet string)
 returns variant
 api_integration = ld_sf_zip_file_process_api_integration
 as 'https:/[redacted]
1.amazonaws.com/first_deployment/ld-sf_zip_file_process';

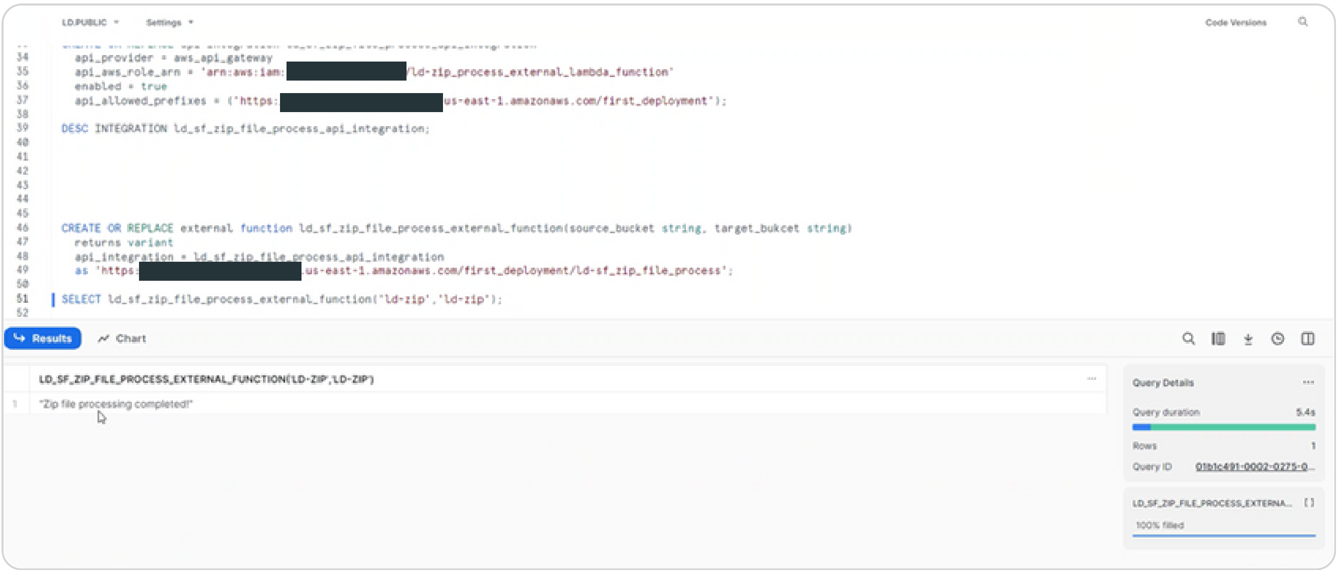


```
CREATE OR REPLACE external function ld_sf_zip_file_process_external_function(source_bucket string, target_bukcet string)
  returns variant
  api_integration = ld_sf_zip_file_process_api_integration
  as 'https[redacted]us-east-1.amazonaws.com/first_deployment/ld-sf_zip_file_process';
```

	status
1	Function LD_SF_ZIP_FILE_PROCESS_EXTERNAL_FUNCTION successfully created.

Query Details
Query duration 102ms
Rows 1
Query ID 01b1c481-0002-00ce-0...
status
100% filled

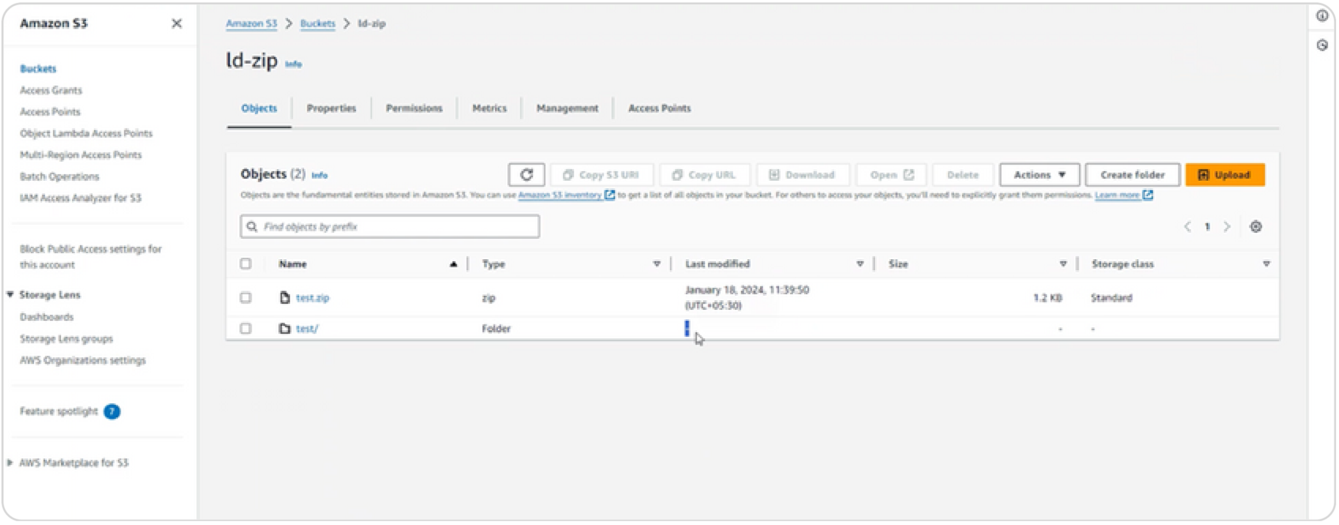
- Now, we call the external function which will unzip the folder in AWS S3.



```
  api_provider = aws_api_gateway
  api_aws_role_arn = 'arn:aws:iam:[redacted]/ld-zip_process_external_lambda_function'
  enabled = true
  api_allowed_prefixes = ('https:[redacted]us-east-1.amazonaws.com/first_deployment');

DESC INTEGRATION ld_sf_zip_file_process_api_integration;





CREATE OR REPLACE external function ld_sf_zip_file_process_external_function(source_bucket string, target_bukcet string)
  returns variant
  api_integration = ld_sf_zip_file_process_api_integration
  as 'https:[redacted].us-east-1.amazonaws.com/first_deployment/ld-sf_zip_file_process';

SELECT ld_sf_zip_file_process_external_function('ld-zip','ld-zip');
```

	LD_SF_ZIP_FILE_PROCESS_EXTERNAL_FUNCTION('LD-ZIP','LD-ZIP')
1	"Zip file processing completed!"

Query Details
Query duration 5.4s
Rows 1
Query ID 01b1c491-0002-0275-0...
LD_SF_ZIP_FILE_PROCESS_EXTERNA...
100% filled



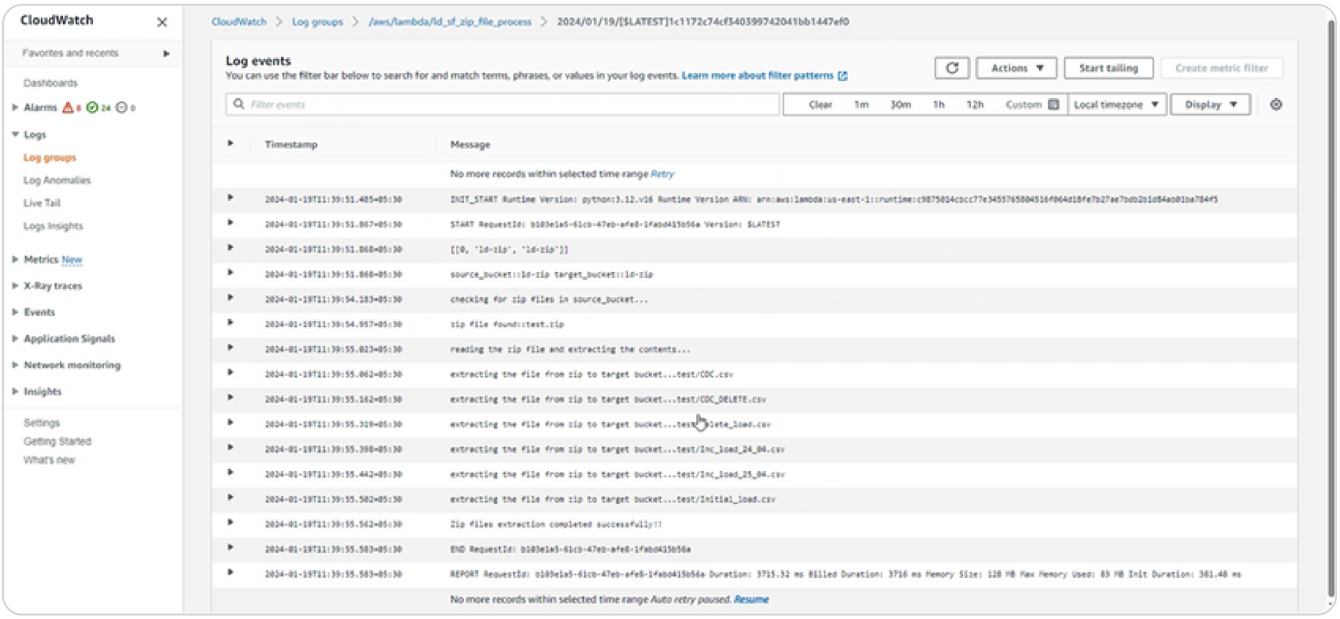
ld-zip

Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
test.zip	zip	January 18, 2024, 11:39:50 (UTC+05:30)	1.2 KB	Standard
test/	Folder		-	-

- In AWS, we can see the Lambda functions logs from "CLOUDWATCH", for debugging any issue.



You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                       Message
                                No more records within selected time range Retry
2024-01-19T11:39:51.485+05:30   INIT_START Runtime Version: python:3.12.v16 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:c9875014cbcc77e3455765804516f064d18fe7b27ae7bdb2b1d84a001ba784f5
2024-01-19T11:39:51.867+05:30   START RequestId: b103e1a5-61cb-47eb-afe8-1fabd415b56a Version: $LATEST
2024-01-19T11:39:51.868+05:30   [[0, 'ld-zip', 'ld-zip']]
2024-01-19T11:39:51.868+05:30   source_bucket::ld-zip target_bucket::ld-zip
2024-01-19T11:39:54.183+05:30   checking for zip files in source_bucket...
2024-01-19T11:39:54.957+05:30   zip file found::test.zip
2024-01-19T11:39:55.023+05:30   reading the zip file and extracting the contents...
2024-01-19T11:39:55.062+05:30   extracting the file from zip to target bucket...test/CDC.csv
2024-01-19T11:39:55.162+05:30   extracting the file from zip to target bucket...test/CDC_DELETE.csv
2024-01-19T11:39:55.319+05:30   extracting the file from zip to target bucket...test[illegible]elete_load.csv
2024-01-19T11:39:55.398+05:30   extracting the file from zip to target bucket...test/Inc_load_24_04.csv
2024-01-19T11:39:55.442+05:30   extracting the file from zip to target bucket...test/Inc_load_25_04.csv
2024-01-19T11:39:55.502+05:30   extracting the file from zip to target bucket...test/Initial_load.csv
2024-01-19T11:39:55.562+05:30   Zip files extraction completed successfully!!
2024-01-19T11:39:55.583+05:30   END RequestId: b103e1a5-61cb-47eb-afe8-1fabd415b56a
2024-01-19T11:39:55.583+05:30   REPORT RequestId: b103e1a5-61cb-47eb-afe8-1fabd415b56a Duration: 3715.32 ms Billed Duration: 3716 ms Memory Size: 128 MB Max Memory Used: 83 MB Init Duration: 381.48 ms
                                No more records within selected time range Auto retry paused. Resume
```

Authors

Ritesh Chidrewar
Senior Consultant

Sai Bharadwaja
Senior Consultant

About LumenData

LumenData is a leading provider of **Enterprise Data Management, Cloud & Analytics** solutions. We help businesses navigate their data visualization and analytics anxieties and enable them to accelerate their innovation journeys.
Founded in 2008, with locations in multiple countries, LumenData is privileged to serve over 100 leading companies. LumenData is **SOC2 certified** and has instituted extensive controls to protect client data, including adherence to GDPR and CCPA regulations.