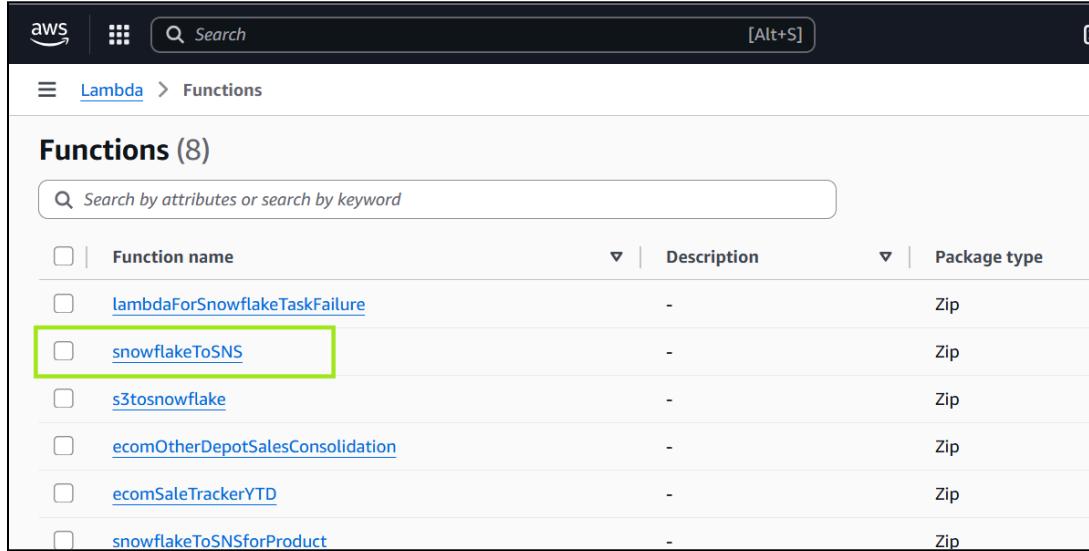


# Snowflake to SNS Notification Pipeline

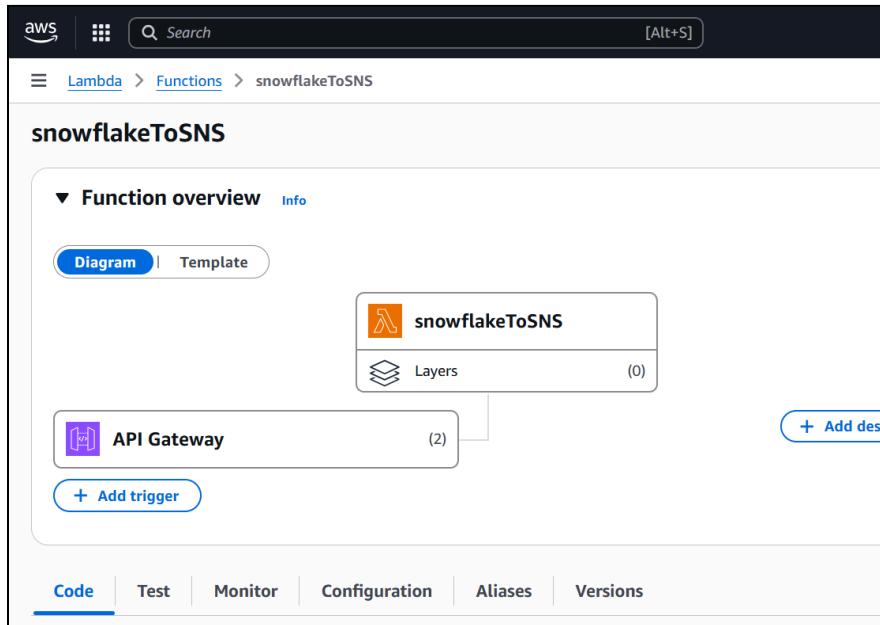
Distributor & Product Addition

## AWS Set-up:

- Navigate to AWS Lambda Console. Function name: 'snowflakeToSNS'.



Function name	Description	Package type
lambdaForSnowflakeTaskFailure	-	Zip
snowflakeToSNS	-	Zip
s3tosnowflake	-	Zip
ecomOtherDepotSalesConsolidation	-	Zip
ecomSaleTrackerYTD	-	Zip
snowflakeToSNSforProduct	-	Zip



This function contains python code in the default code editor. The purpose of the code/lambda function is to receive messages from snowflake external function (mentioned later in this document), parse it and forward it to AWS SNS topic (in next step).

- Open the AWS SNS console. Go to topic '[NewDistributorTopic](#)' or '[NewProductTopic](#)'

The screenshot shows the AWS SNS Topics page. On the left, there's a sidebar with 'Amazon SNS' at the top, followed by 'Dashboard', 'Topics' (which is selected and highlighted in blue), 'Subscriptions', and 'Mobile' which includes 'Push notifications' and 'Text messaging (SMS)'. The main area is titled 'Topics (4)' and lists the following:

Name	Type	ARN
<a href="#">Agastya-taskEdit</a>	Standard	arn:aws:sns:ap-south-1:123456789012:Agastya-taskEdit
<a href="#">NewDistributorTopic</a>	Standard	arn:aws:sns:ap-south-1:123456789012:NewDistributorTopic
<a href="#">NewProductTopic</a>	Standard	arn:aws:sns:ap-south-1:123456789012:NewProductTopic
<a href="#">SnowflakeTaskFailure</a>	Standard	arn:aws:sns:ap-south-1:123456789012:SnowflakeTaskFailure

This topic is of type: “*Standard*” (not FIFO), everything else is default. Subscriptions are created which are subscribed to this topic. Below are the subscriptions (of type Email) which are subscribed to this topic:

The screenshot shows the 'Subscriptions' tab for the 'NewDistributorTopic' topic. At the top, there are tabs for 'Subscriptions', 'Access policy', 'Data protection policy', 'Delivery policy (HTTP/S)', 'Delivery status logging', and 'Encryption'. The 'Subscriptions' tab is selected and highlighted in blue. Below the tabs, there's a search bar and a table with the following data:

ID	Endpoint	Status	Protocol
<a href="#">3af9bf6b-b949-46d9-9918-b...</a>	ketan.vaidya@niine.com	Confirmed	EMAIL
<a href="#">cd1fe0bb-5b5b-40ae-b1b1-5...</a>	Shreemala.dhayal@niine.com	Confirmed	EMAIL
<a href="#">e7cc4150-b515-4296-8b8a-e...</a>	ayush.narayan@niine.com	Confirmed	EMAIL

All the subscribed people will receive the mail when the pipeline is triggered from snowflake.

- Now navigate to AWS API Gateway console, specifically to '[ApiForSNS](#)' API.

APIs (2/2)

Name	Description	ID	Protocol	API endpoint type	Created	Security policy
ApiForSNS		o5vo9rus4f	HTTP	Regional	2025-01-16	-
GetDetailsForUPC		gbleujnd7c	REST	Regional	2025-01-29	TLS_1_0

Go to the “Stages” section, stage is where the base url is present.

Stages

Name
\$default
<b>prod</b>

**Stage details**

**Details**

Name	Created	Last updated
prod	January 16, 2025 5:52 PM	April 28, 2025 5:20 PM
Invoke URL		
<a href="https://o5vo9rus4f.execute-api.ap-south-1.amazonaws.com/prod">https://o5vo9rus4f.execute-api.ap-south-1.amazonaws.com/prod</a>		

**Description**  
None

**Attached deployment**  
**Automatic Deployment**  
 Enabled

Deployment ID	Deployment created
lxmk9k	April 28, 2025 5:20 PM

Then in the routes section, further url and API method (POST) is defined.

Routes

Path	Method
/send-notification	POST

**Route details**

**ARN**  
`arn:aws:apigateway:ap-south-1::apis/o5vo9rus4f/routes/5368v97`

**Authorization**  
Authorizers protect your API against unauthorized requests. Routes with no authorization attached are open.

No authorizer attached to this route. [Attach authorization](#)

**Integration**  
The integration is the backend resource that this route calls when it receives a request.

681ne1t [Configure](#)

After routes, integration is attached to the route, which in this case is Lambda integration (the ‘snowflakeToSNS’ lambda function in above steps). When a request is made to the complete url of this API (<https://o5vo9rus4f.execute-api.ap-south-1.amazonaws.com/prod/send-notification>), it will call the attached lambda function.

The screenshot shows the AWS API Gateway Integrations page for an API named 'ApiForSNS...'. On the left, there's a sidebar with sections for Develop (Routes, Authorization, Integrations, CORS, Reimport, Export), Deploy (Stages), and Monitor (Metrics, Logging). The main area is titled 'Integrations' with tabs for 'Attach integrations to routes' (selected) and 'Manage integrations'. Under 'Routes for ApiForSNS', a search bar shows '/send-notification' with a 'POST' method and 'AWS Lambda' selected as the provider. To the right, 'Integration details for route' are displayed, showing a POST method to '/send-notification' with a Lambda function named 'snowflakeToSNS (ap-south-1)'. The Integration ID is 681ne1t. A note about payload format version 2.0 (interpreted response format) is also present.

AWS Set-up is complete, now snowflake needs to be configured to be able to call ‘snowflakeToSNS’ lambda function so that it can forward the message from snowflake to subscribed users of SNS Topic.

## Snowflake Set-up:

- API Integration is created with API provider as *aws\_api\_gateway* and base url of our API as allowed prefix. Follow this [documentation](#).

The screenshot shows a Snowflake Worksheet titled 'DistributorNotification Worksheet'. It contains the following SQL code:

```

CREATE OR REPLACE API INTEGRATION aws_sns_integration
API_PROVIDER = 'aws_api_gateway'
API_AWS_ROLE_ARN = 'arn:aws:iam::339712928391:role/RoleForSNSDistributorAddition'
API_ALLOWED_PREFIXES = ('https://o5vo9rus4f.execute-api.ap-south-1.amazonaws.com/prod')
ENABLED = TRUE;
CREATE OR REPLACE EXTERNAL FUNCTION send_sns_notification(message STRING)

```

- External function is then created with String type parameter and complete url of our AWS API route.

```

My Workspace
Worksheets
Search for files
DeadOutlets-L3M
DistributorNotification Worksheet
Distributors_Product_Update

CREATE OR REPLACE EXTERNAL FUNCTION send_sns_notification(message STRING)
RETURNS VARIANT
API_INTEGRATION = aws sns integration
AS 'https://o5vo9rus4f.execute-api.ap-south-1.amazonaws.com/prod/send-notification';

```

- Snowflake query is written as such that it executes the above function and the query for new distributor addition is passed as parameter to the function.

```

-- 2) Distributor Update:
WITH sns_result AS (
    SELECT send_sns_notification(
        -- Concatenate the DISTINCT result into a single string
        (SELECT LISTAGG(
            customer_data.CUSTOMER_CODE || ',' || customer_data.customer_name || ',' ||
            customer_data.customer_state_name || ',' || customer_data.GROUPNAME, ';'
        ) WITHIN GROUP (ORDER BY customer_data.CUSTOMER_CODE) -- Order by CUSTOMER_CODE
        FROM (
            SELECT DISTINCT a.CUSTOMER_CODE, a.customer_name, a.customer_state_name, a.G
            FROM S3_DATA_NIINE.SAP_PRIMARY.a
            LEFT JOIN S3_DATA_NIINE.NIINE_DATA.DB_MASTER b
                ON a.CUSTOMER_CODE = b.DISTRIBUTORERPID
            WHERE b.DISTRIBUTORERPID IS NULL
        ) AS customer_data
    )
)
SELECT
    result
FROM sns_result;

```

- Snowflake task is then created for this query to be executed at a specific schedule.

```

100
101      -----Task for Email Update-----
102
103      CREATE OR REPLACE TASK send_sns_notification_task
104          WAREHOUSE = COMPUTE_WH
105          SCHEDULE = 'USING CRON 30 7 * * * UTC'
106          -- SCHEDULE = '15 MINUTE'
107          -- SCHEDULE = '60 MINUTE'
108          -- SCHEDULE = '3 HOURS'
109
110      AS
111          WITH sns_result AS (
112              SELECT send_sns_notification(
113                  -- Concatenate the DISTINCT result into a single string
114                  (SELECT LISTAGG(
115                      customer_data.CUSTOMER_CODE || ',' || customer_data.customer_name || ',' ||
116                      customer_data.customer_state_name || ',' || customer_data.GROUPNAME, ','
117                  ) WITHIN GROUP (ORDER BY customer_data.CUSTOMER_CODE) -- Order by CUSTOMER_CODE
118                  FROM (
119                      SELECT DISTINCT a.CUSTOMER_CODE, a.customer_name, a.customer_state_name, a.GROUPNAME
120                          FROM S3_DATA_NINE.SAP_PRIMARY.SAP_PRIMARY a
121                          LEFT JOIN S3_DATA_NINE.NINE_DATA.DB_MASTER b
122                              ON a.CUSTOMER_CODE = b.DISTRIBUTORERPID
123                          WHERE b.DISTRIBUTORERPID IS NULL
124                  ) AS customer_data)
125          )
126          SELECT
127              result
128          FROM sns_result;

```

For pipeline flow, view the “Snowflake to SNS Pipeline.drawio” diagram.

– By Agastya Singh