NOTIFICATION USING SNS-SQS

Why notifications?

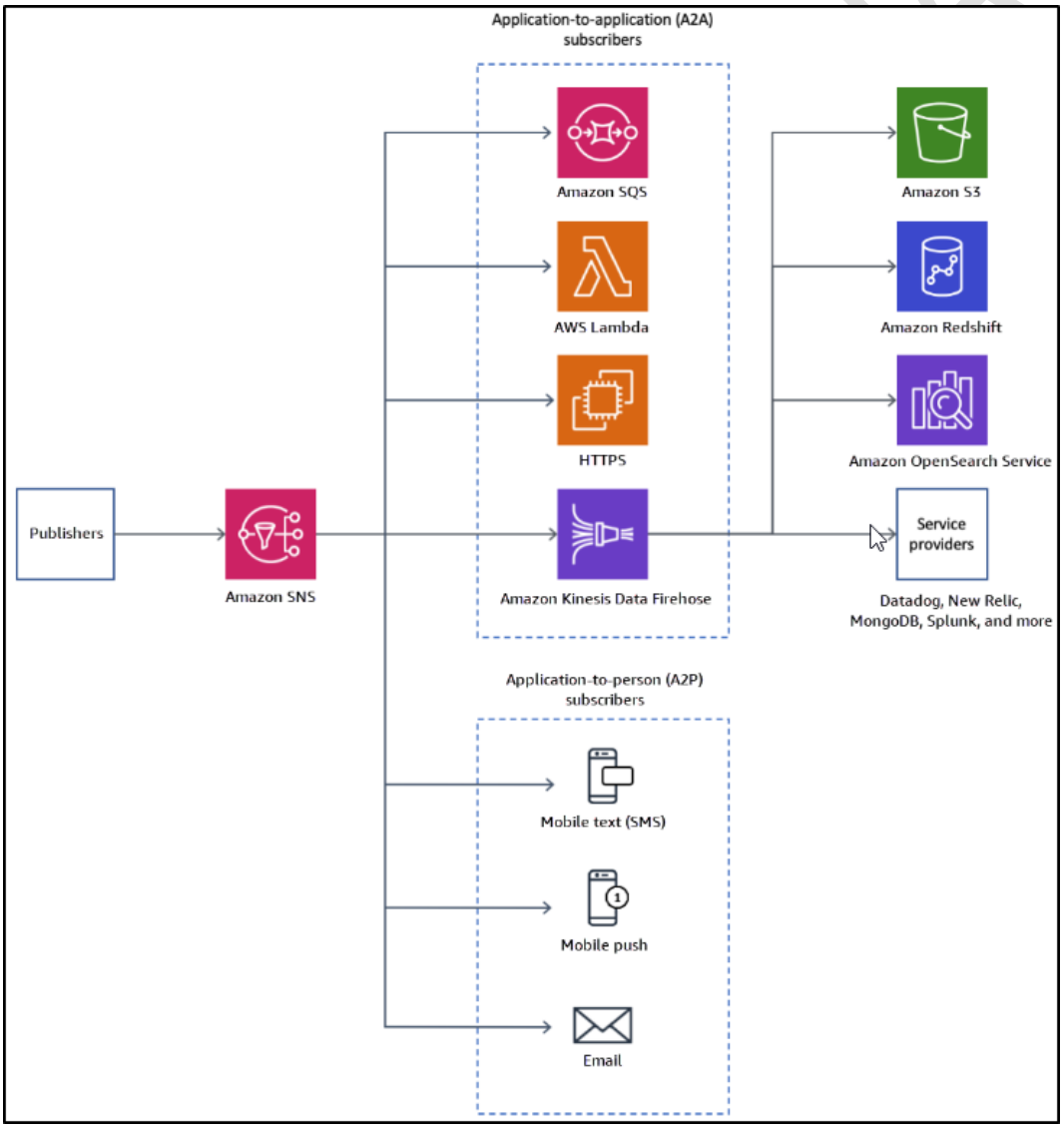
- To know that jobs have run successfully or failed. If failed, details of the failure.

AWS SNS & SQS INTRO: #detailed in AWS

What is SNS?

- Simple notification service.
- It's an alerting service.

AWS official documentation: https://docs.aws.amazon.com/sns/latest/dg/welcome.html



SNS topics are used to enable communication:

- Producers publish messages to topics, and

- Consumers subscribe to these topics to receive messages.

What is SQS?

- Simple Queue Service.
- AWS SQS is **queueing** service used for communication between applications, micro services, and distributed systems. # in short, software components

- Components:
  - **Producers** (components that send messages to the queue).
  - **Queue** (which stores messages).
  - **Consumers** (other components that receive messages from the queue).

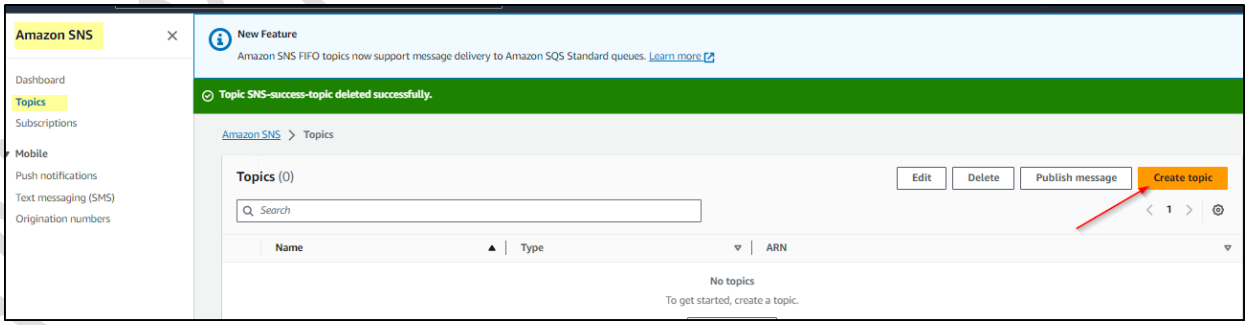
Differences:

SNS	SQS
SNS supports **A2A and A2P** communication	SQS supports only **A2A** communication
➢ SNS is a pub/sub system ➢ Typically use SNS to send the same message to multiple consumers via topics.	➢ SQS is a queueing system. ➢ Each message in an SQS queue is processed by only one consumer.
➢ SNS does not persist messages - it delivers them to subscribers that are present, and then deletes them.	➢ SQS can persist messages (from 1 minute to 14 days).

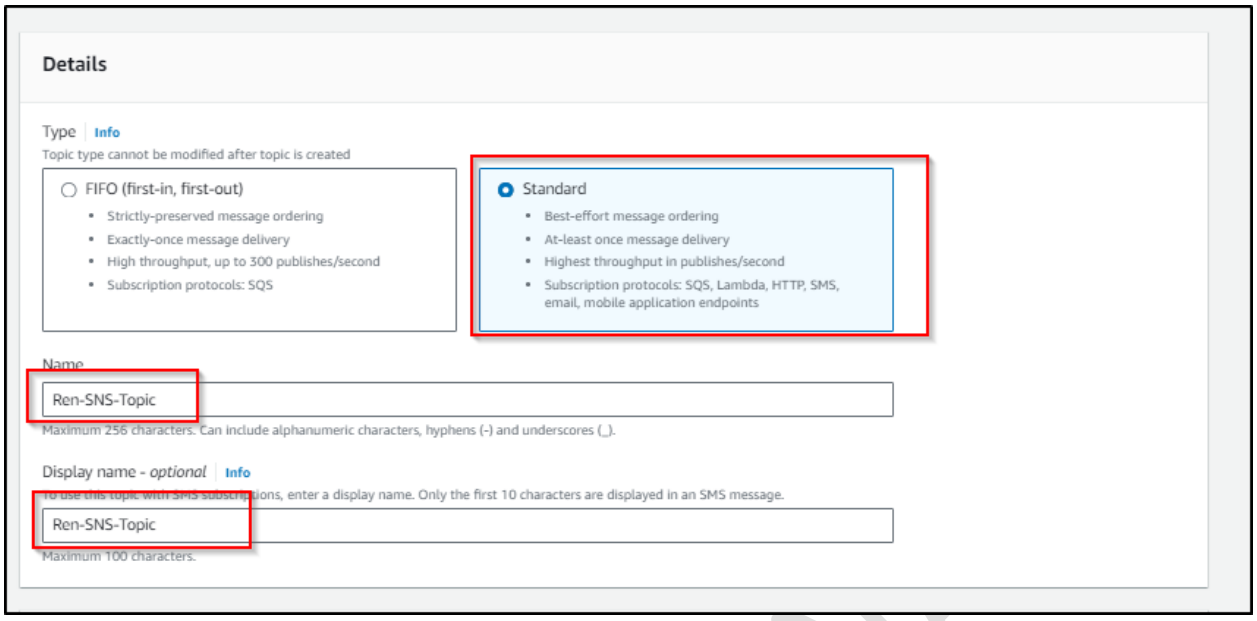
AWS SNS JOB:

Email and SMS alert using SNS:

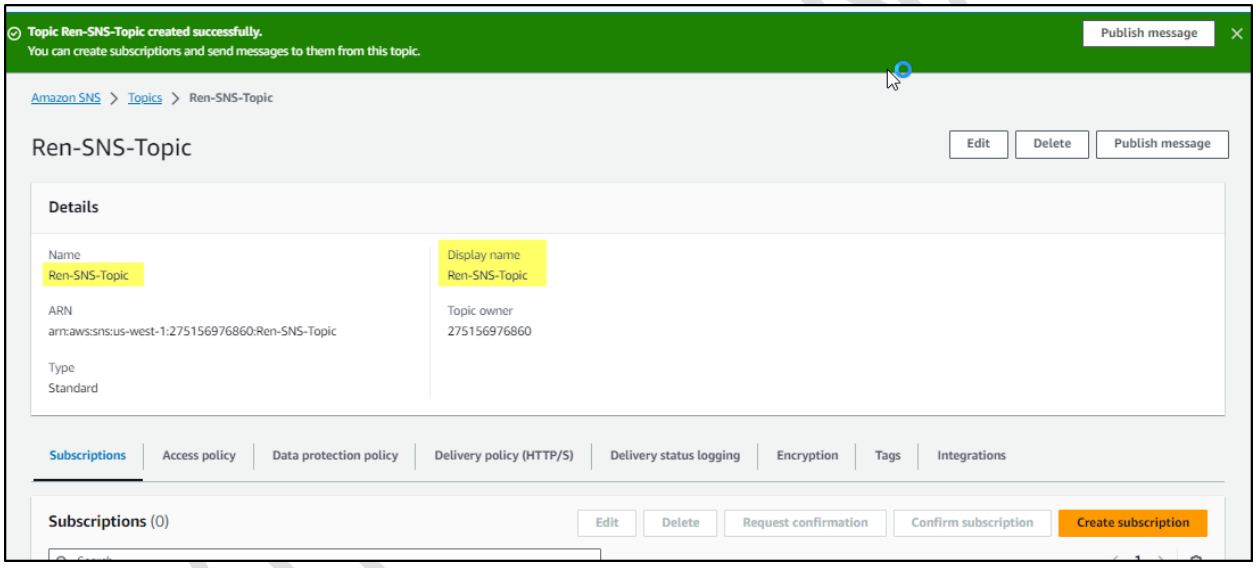
1. In AWS:
   1.1. Create an SNS topic in AWS
      1.1.1. Go to Amazon SNS and click Create topic



      1.1.2. Select the 'Standard' and update the name fields highlighted below. Let the defaults be and click on *Create topic*

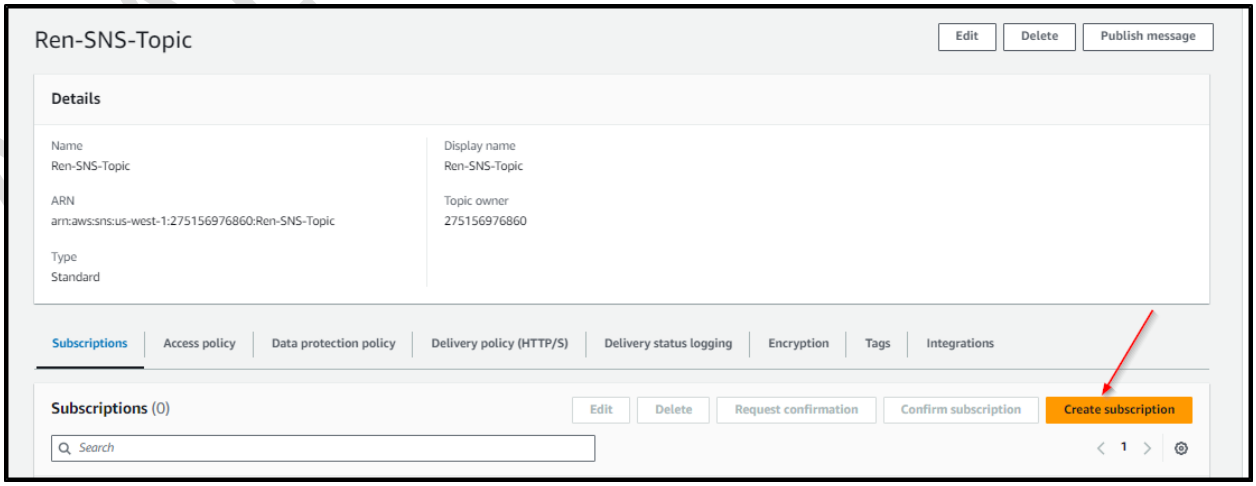


Topic is created:

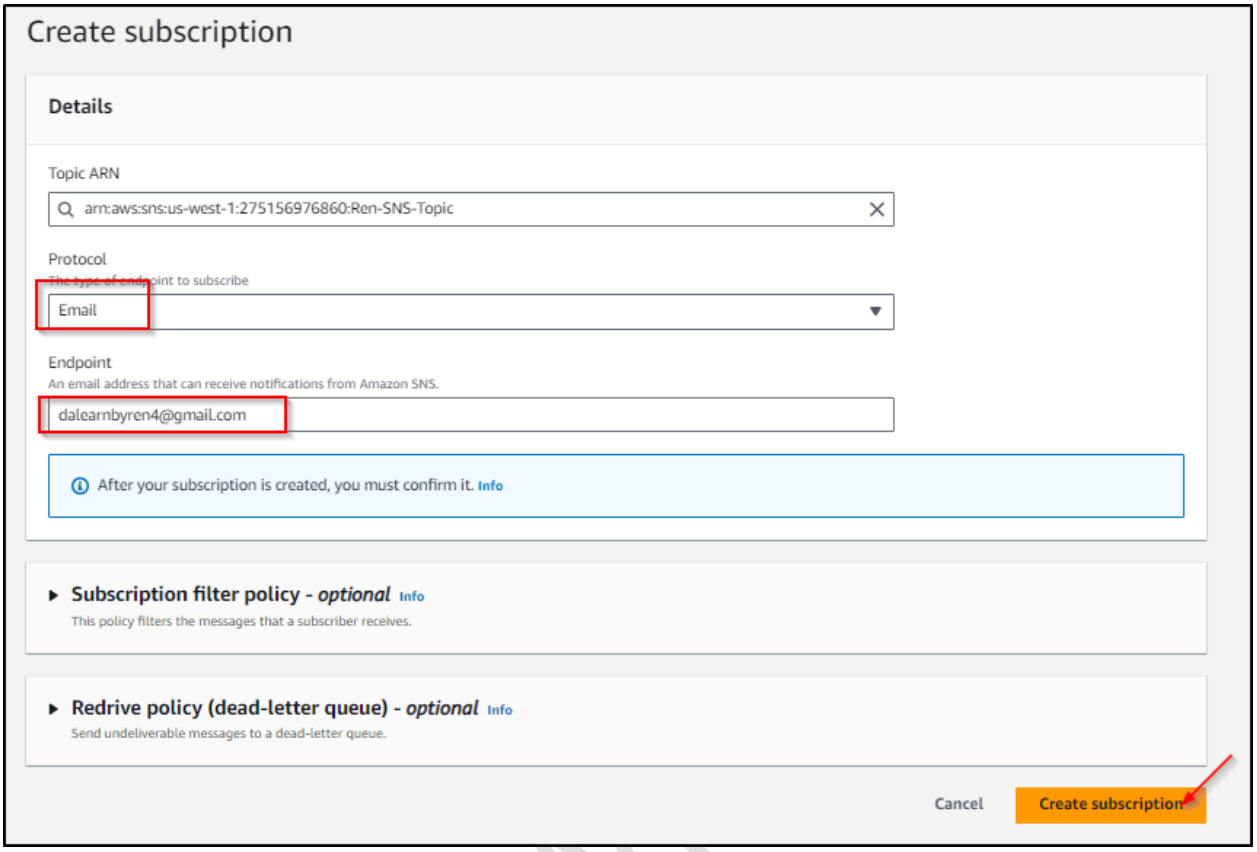


1.2. Create a subscription to the topic with Email alerts and SMS

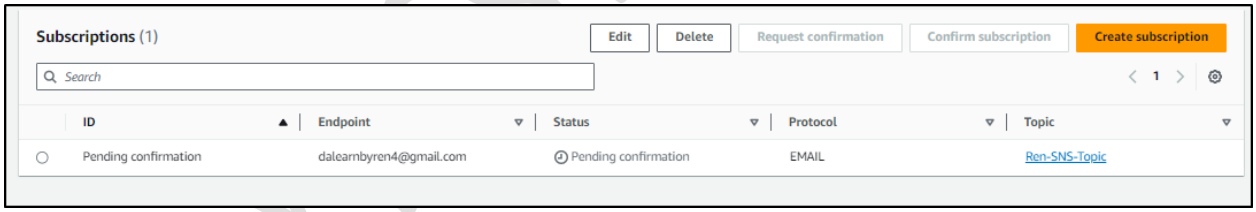
1.2.1. Click on create subscriptions in the same page



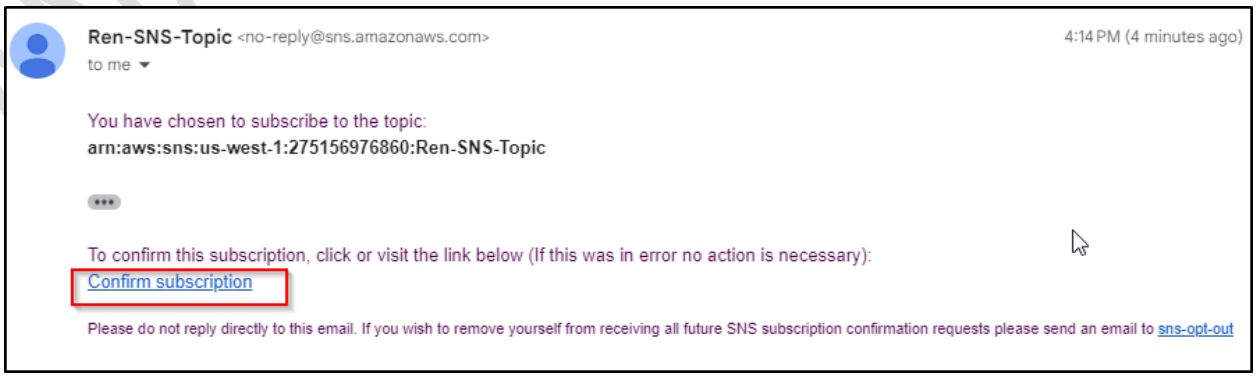
1.2.2.Add protocol as '*Email*' , endpoint click 'create subscription'

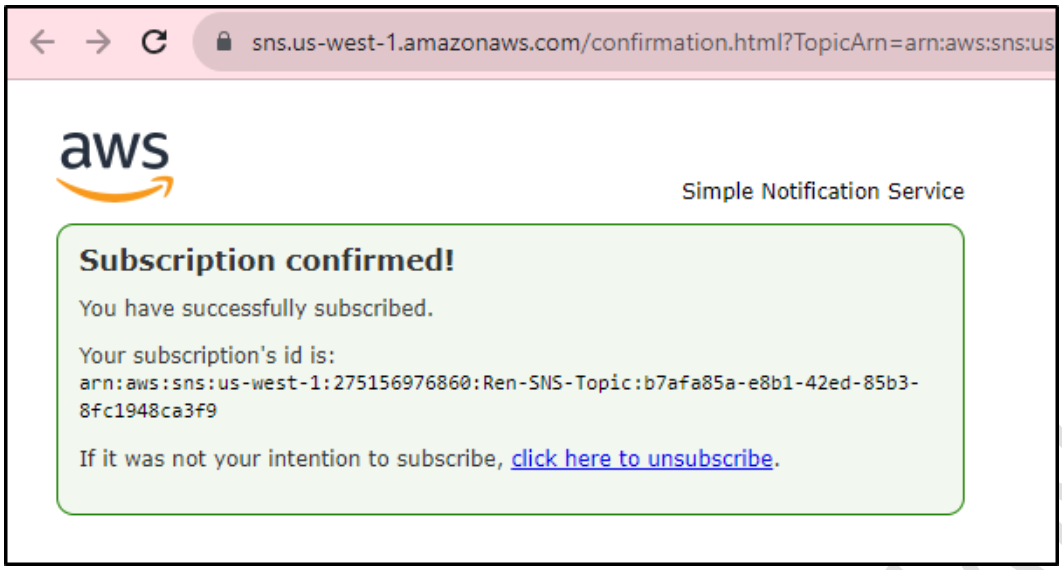


1.2.2.1. Displays as pending confirmation

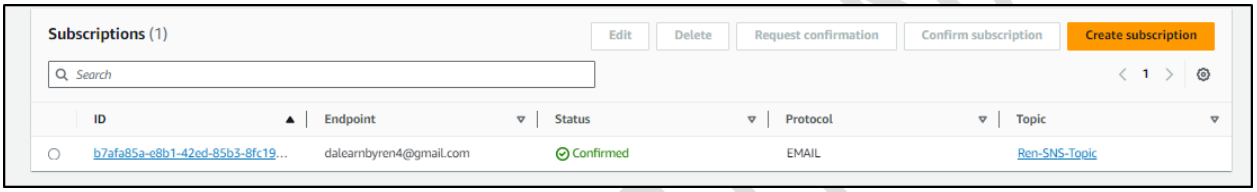


1.2.3.Check your mailbox for the confirmation email (could be in Spam as well) and click 'confirm subscriptions'.

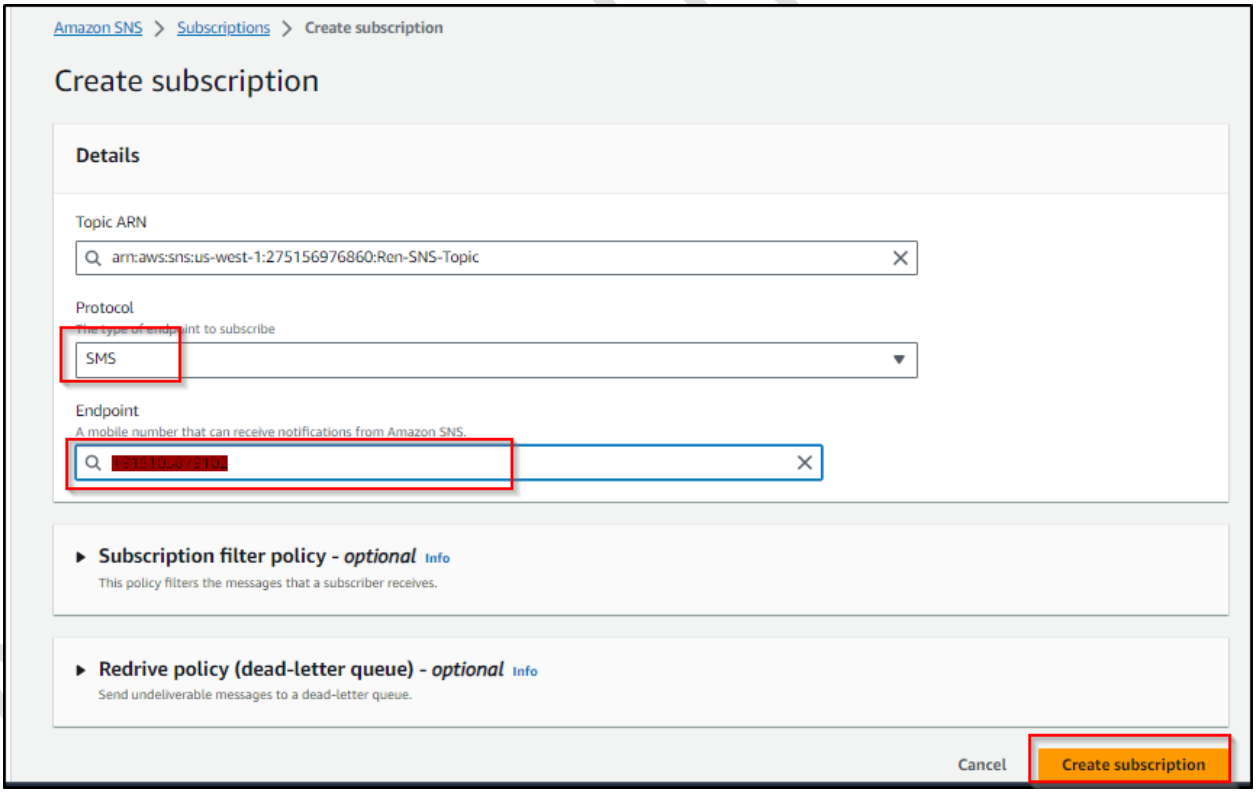




1.3. Refresh your page in Amazon SNS . Status will be confirmed.



1.4. Similarly create a subscription for SMS by adding your phone number



2. In matillion,
   2.1. Create an orchestration project
   2.2. Add python script component

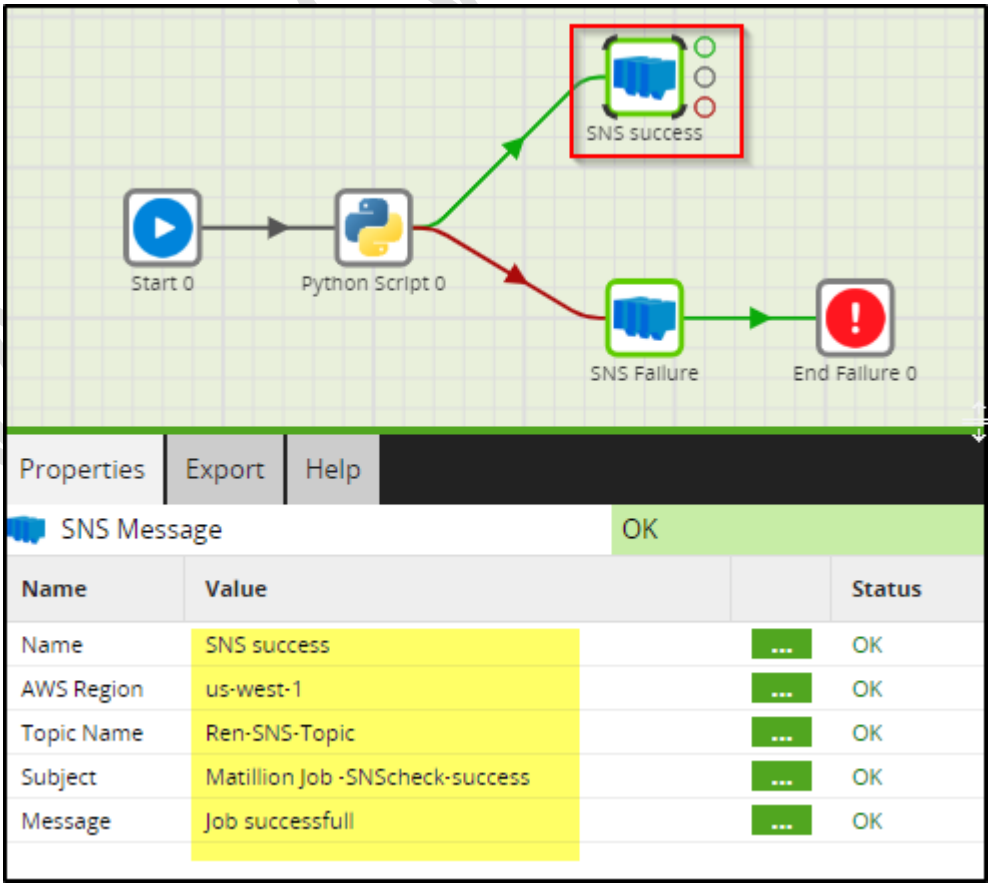
```
###
# Variables are directly accessible:
#   print (myvar)
# Updating a variable:
#   context.updateVariable('myvar', 'new-value')
# Grid Variables are accessible via the context:
#   print (context.getGridVariable('mygridvar'))
# Updating a grid variable:
#   context.updateGridVariable('mygridvar', [['list','of
# A database cursor can be accessed from the context (Jy
#   cursor = context.cursor()
#   cursor.execute('select count(*) from mytable')
#   rowcount = cursor.fetchone()[0]
###
print("running python job - Ren")
```

Run | Python 3 | Timeout (seconds):

```
running python job - Ren
Script duration: 0.1s.
```

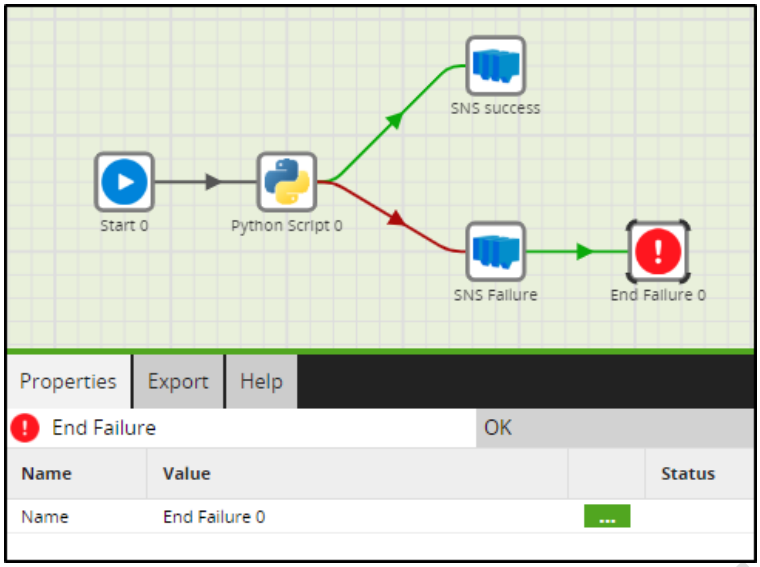
   2.3. Add SNS Message component to your job and configure it for Job Success
      2.3.1.Details :

Name: Give a name for the component
AWS Region: Select the region where we created the SNS topic in AWS
Topic Name: Select the topic name. It should come in the drop-down.
Subject: Give the Email Alert a Subject line.
Message: Configure your message using the Component Metadata and Environment variables.



3. Similarly, Add SNS Message component to your job and configure it for Job Failure

4. Add an End Failure component for the job (optional)
   4.1. The overall status of the job will be failure, even if it would have been otherwise successful.
   4.2. End Failure component

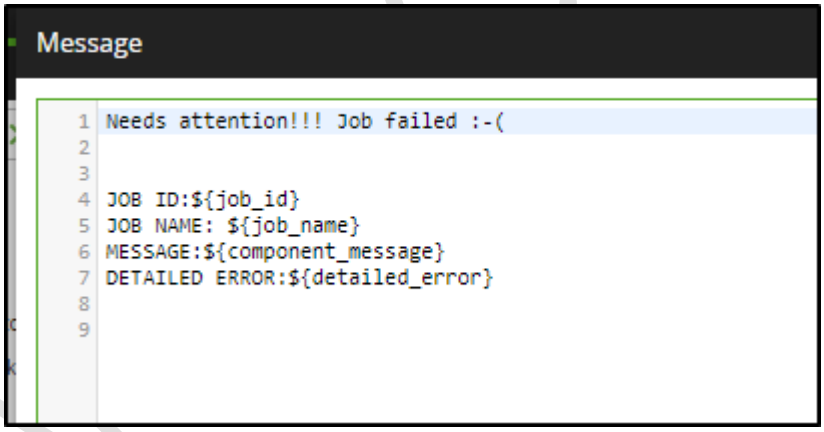


SNS Alert with Job details using automatic variables:

Automatic variables – Already created in the Matillion account by default. It holds the metadata.

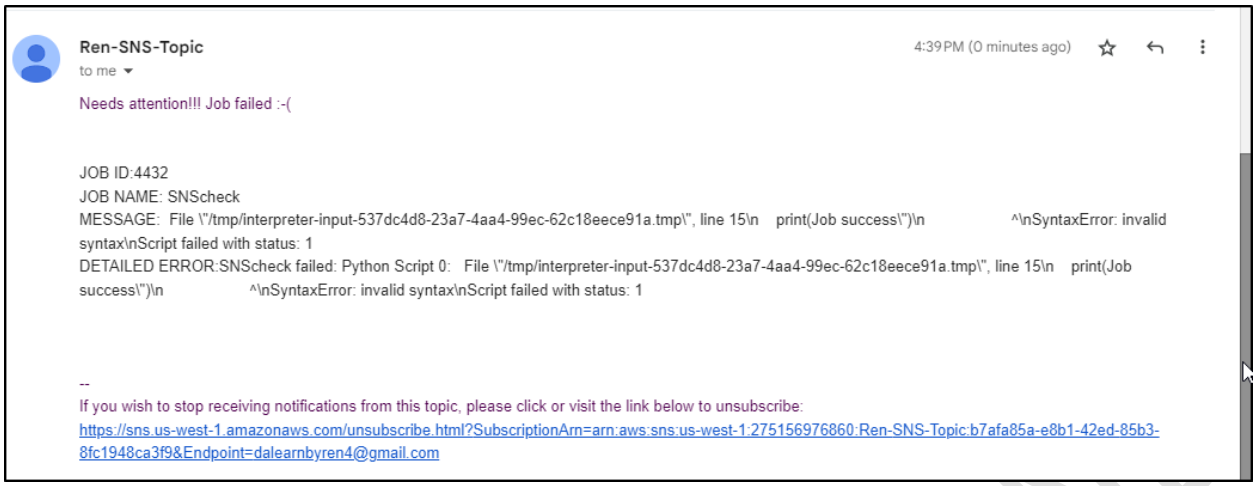
- In the SNS component > message, add below variables

  JOB ID:${job_id}
  JOB NAME: ${job_name}
  MESSAGE:${component_message}
  DETAILED ERROR:${detailed_error}



- Run Job

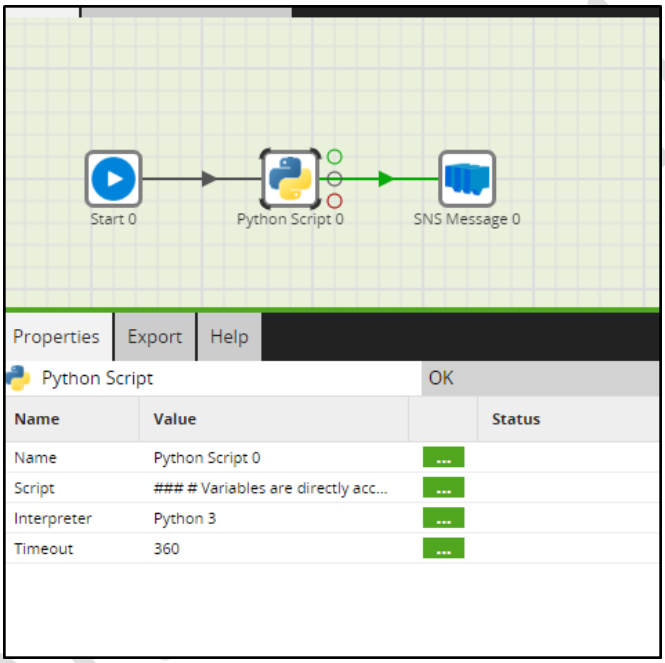
  Email received on failure:



SQS Notification:

Run a Job B based on result in Job A:

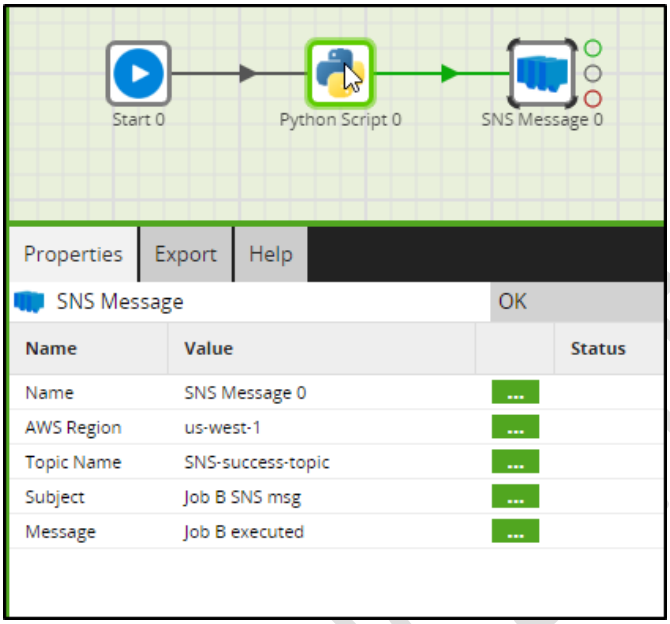
1. We will continue from the Job that we already created. We consider it as Job A
2. Create a simple job B
   2.1. Add a python script component with simple script .
   2.2. Added a SNS message component to receive a email on completion (optional)



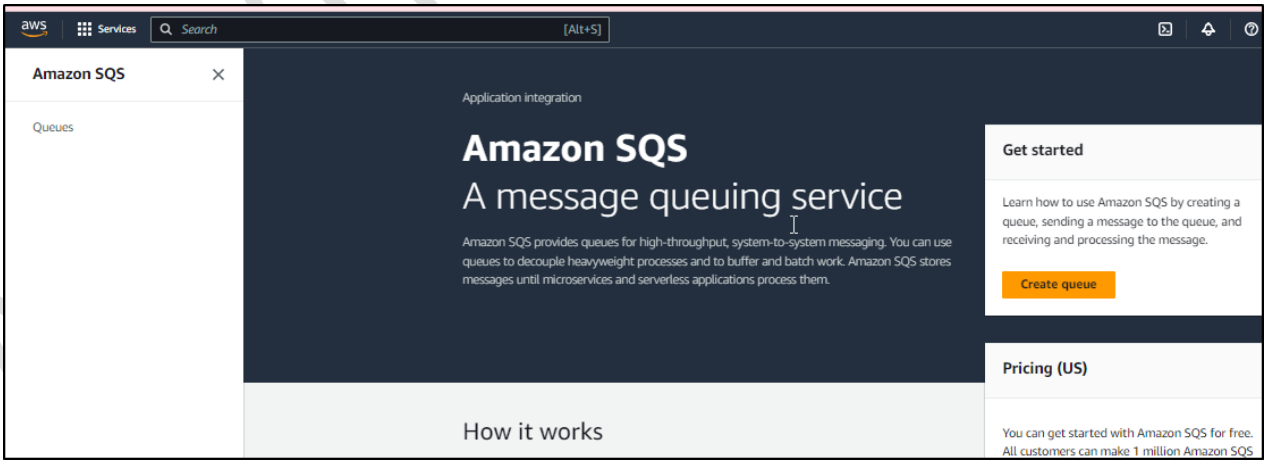
```
###
# Variables are directly accessible:
#   print (myvar)
# Updating a variable:
#   context.updateVariable('myvar', 'new-value')
# Grid Variables are accessible via the context:
#   print (context.getGridVariable('mygridvar'))
# Updating a grid variable:
#   context.updateGridVariable('mygridvar', [['list
# A database cursor can be accessed from the contex
#   cursor = context.cursor()
#   cursor.execute('select count(*) from mytable')
#   rowcount = cursor.fetchone()[0]
###
print("JobB ran sucessfully")
```

Run | Python 3 | Timeout (secor

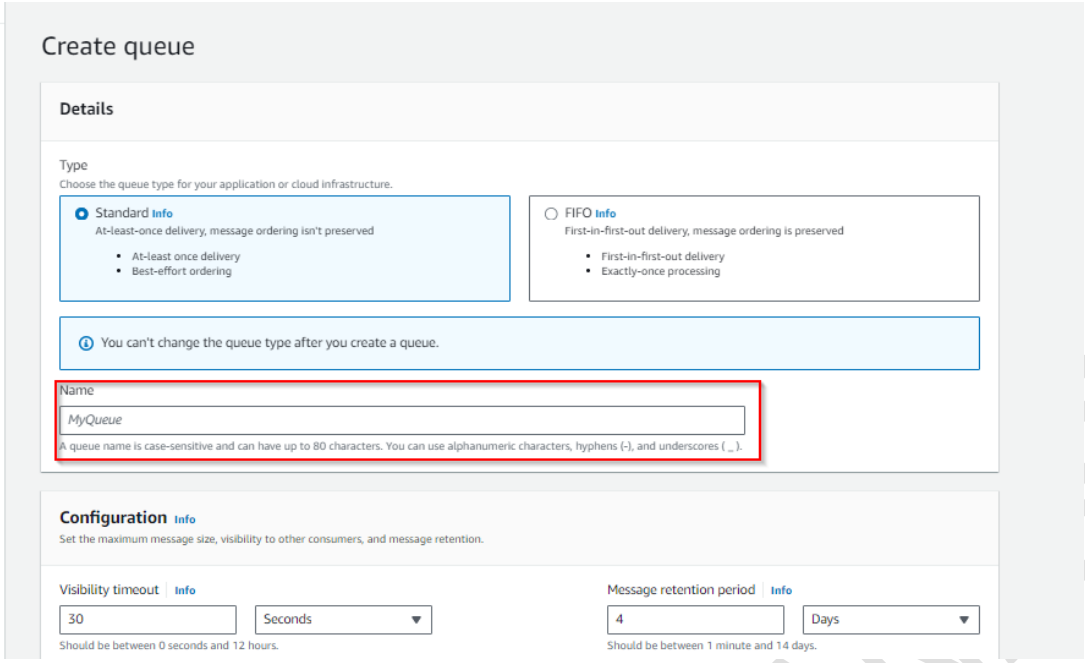
Press "Run" to test script execution.



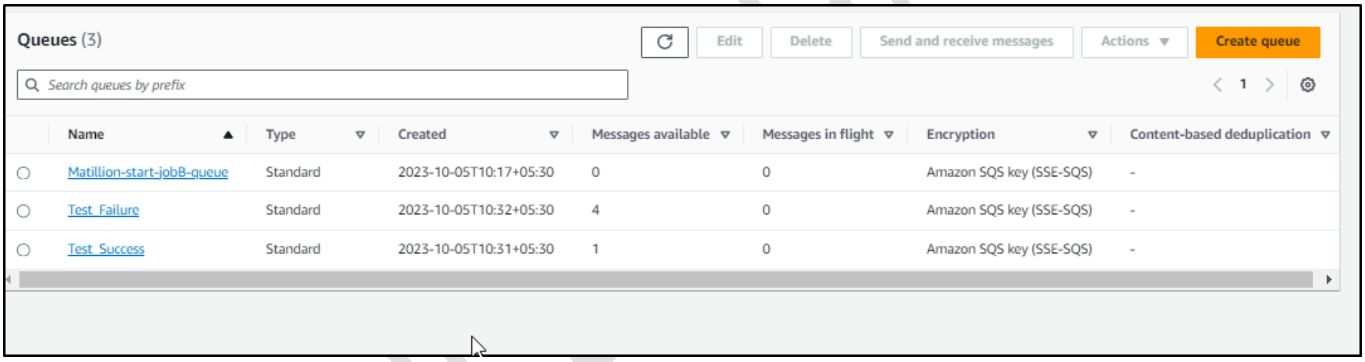
3. Create queue:
   3.1. Go to SQS in AWS, click on create queue



   3.2. Write the name and click on ‘create queue’, We will take this as ‘listen queue’.



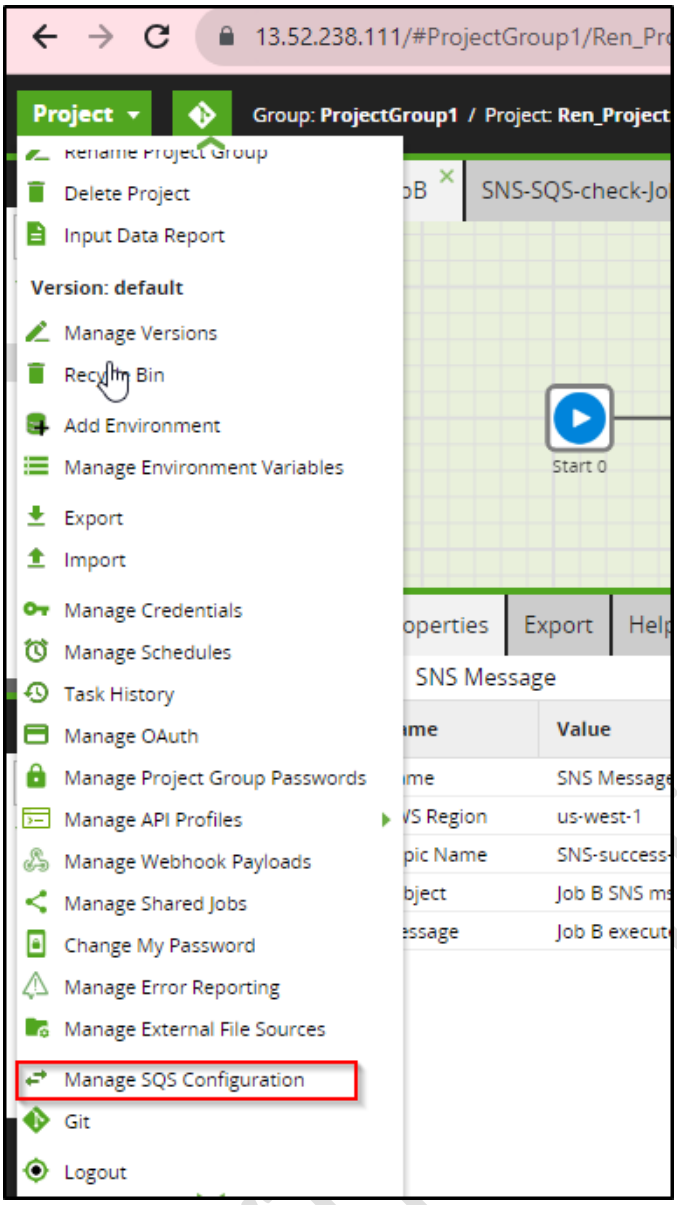
3.3. Similarly create 2 more queues for Success and Failure



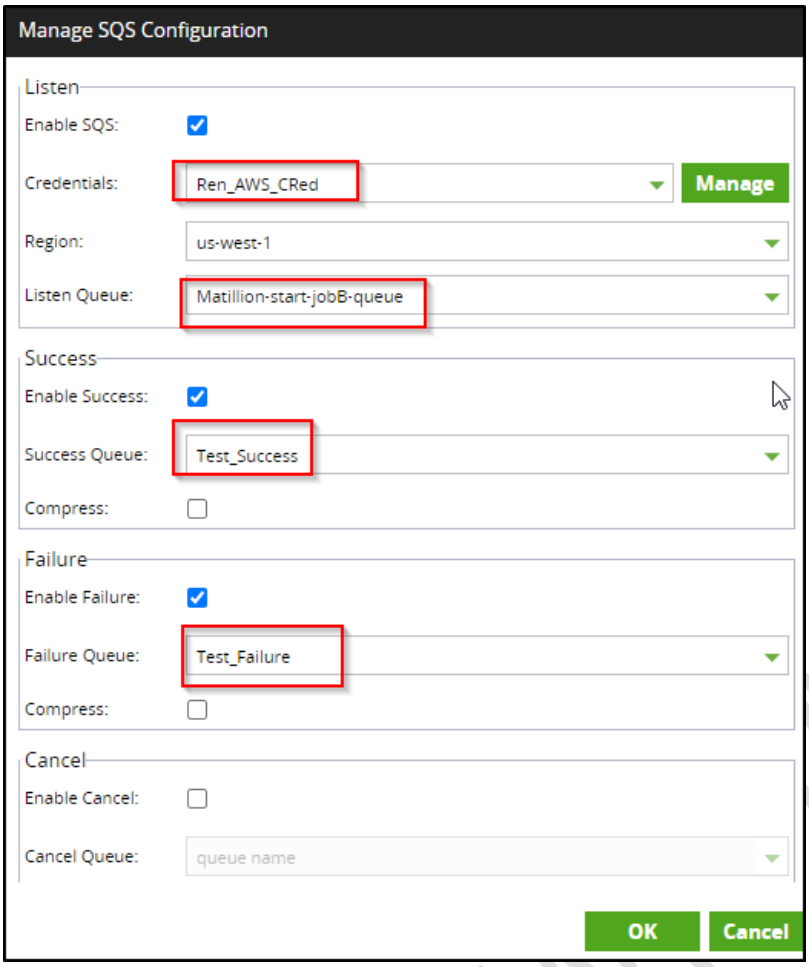
4. In Matillion,

   We need to Configure in ‘Manage SQS Credentials ‘:

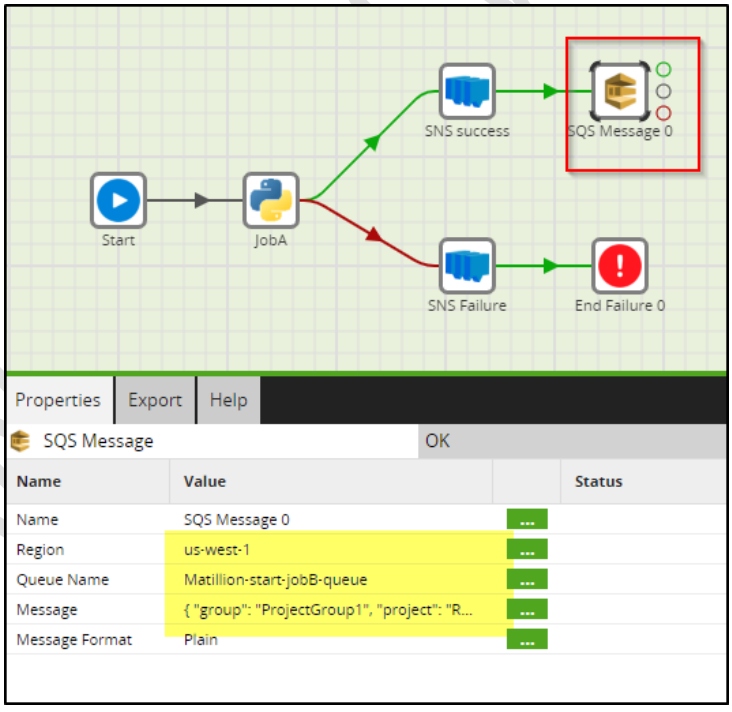
   Go t Project > ‘Manage SQS Credentials ‘



Fill in the details



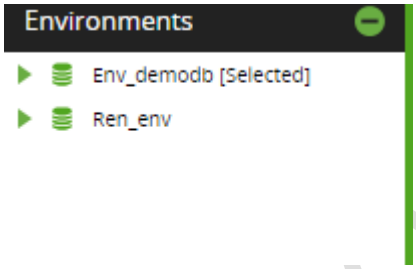
5. Add SQS message component:
   5.1. To our SNS project that we created, add SQS MESSAGE component for success.



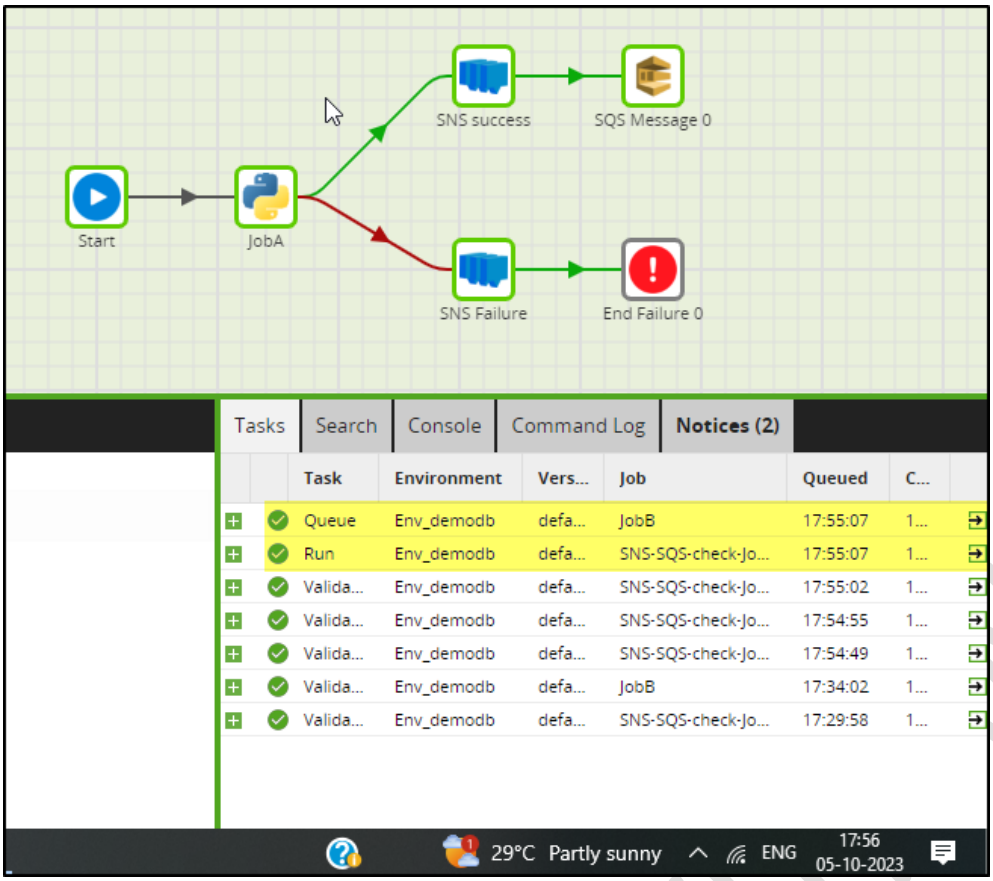
Region – select you AWS region
Queue Name: this is the SQS queue we created as a listener
Message: Add below script and update accordingly

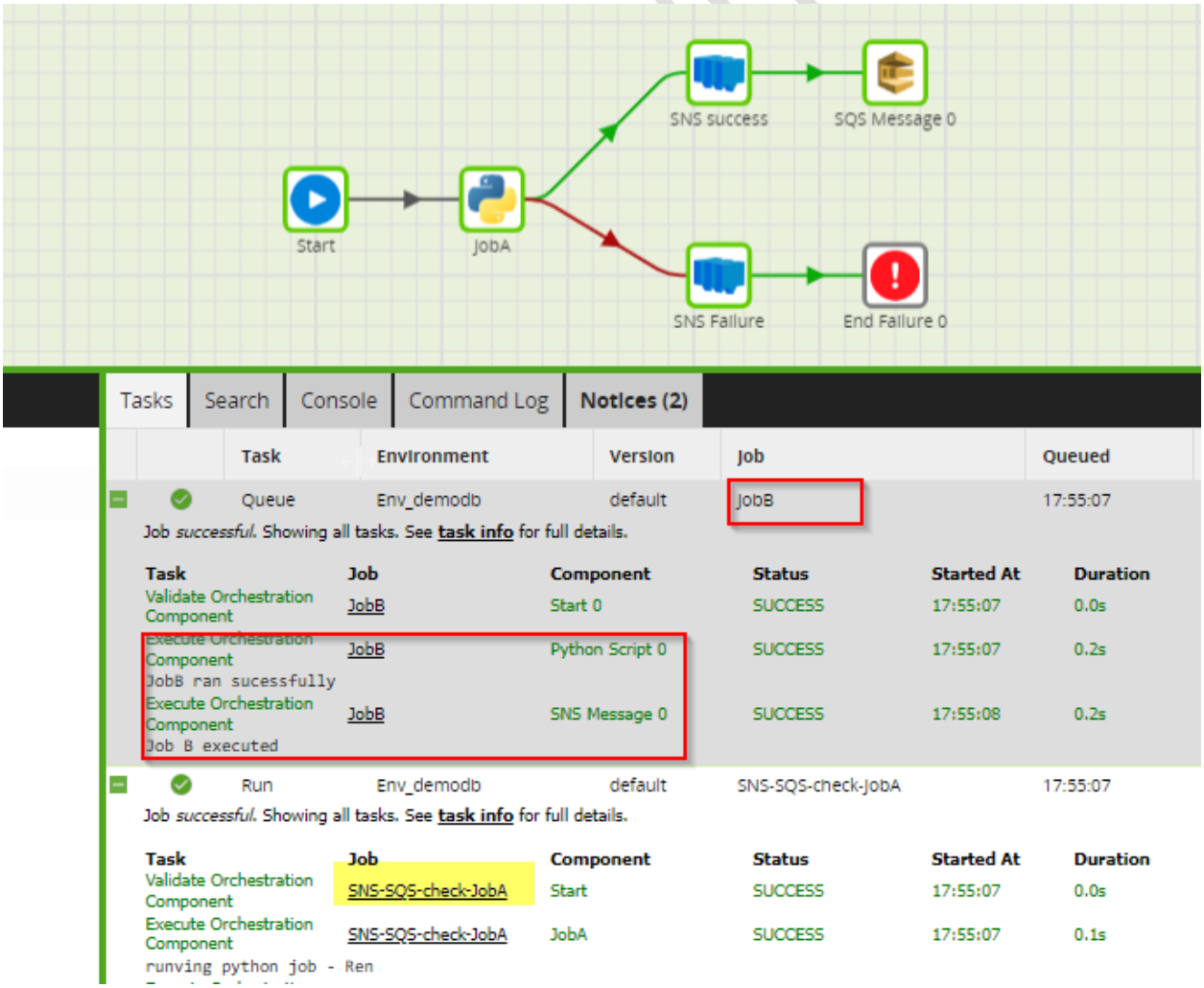
```
{
   "group": "ProjectGroup1",
   "project": "Ren_Project",
   "version": "default",
   "environment": "Env_demodb",
   "job": "JobB"
}
```

Environment is the name of your matiliion environment

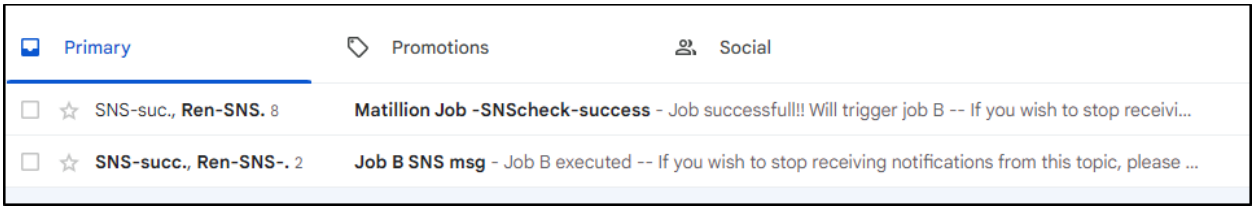


6. Now run Job:


SNS success
SQS Message 0
Start
JobA
SNS Failure
End Failure 0
Tasks
Search
Console
Command Log
Notices (2)
Task
Environment
Vers...
Job
Queued
C...
Queue
Env_demodb
defa...
JobB
17:55:07
1...
Run
Env_demodb
defa...
SNS-SQS-check-Jo...
17:55:07
1...
Valida...
Env_demodb
defa...
SNS-SQS-check-Jo...
17:55:02
1...
Valida...
Env_demodb
defa...
SNS-SQS-check-Jo...
17:54:55
1...
Valida...
Env_demodb
defa...
SNS-SQS-check-Jo...
17:54:49
1...
Valida...
Env_demodb
defa...
JobB
17:34:02
1...
Valida...
Env_demodb
defa...
SNS-SQS-check-Jo...
17:29:58
1...
29°C Partly sunny
ENG
17:56
05-10-2023


SNS success
SQS Message 0
Start
JobA
SNS Failure
End Failure 0
Tasks
Search
Console
Command Log
Notices (2)
Task
Environment
Version
Job
Queued
Queue
Env_demodb
default
JobB
17:55:07
Job successful. Showing all tasks. See task info for full details.
Task
Job
Component
Status
Started At
Duration
Validate Orchestration Component
JobB
Start 0
SUCCESS
17:55:07
0.0s
Execute Orchestration Component
JobB ran sucessfully
JobB
Python Script 0
SUCCESS
17:55:07
0.2s
Execute Orchestration Component
Job B executed
JobB
SNS Message 0
SUCCESS
17:55:08
0.2s
Run
Env_demodb
default
SNS-SQS-check-JobA
17:55:07
Job successful. Showing all tasks. See task info for full details.
Task
Job
Component
Status
Started At
Duration
Validate Orchestration Component
SNS-SQS-check-JobA
Start
SUCCESS
17:55:07
0.0s
Execute Orchestration Component
runving python job - Ren
SNS-SQS-check-JobA
JobA
SUCCESS
17:55:07
0.1s

Received emails for Jobs executed.



Congratulations you have successfully configured notification for Job status in matillion using SNS and SQS.