CELIA MURIEL

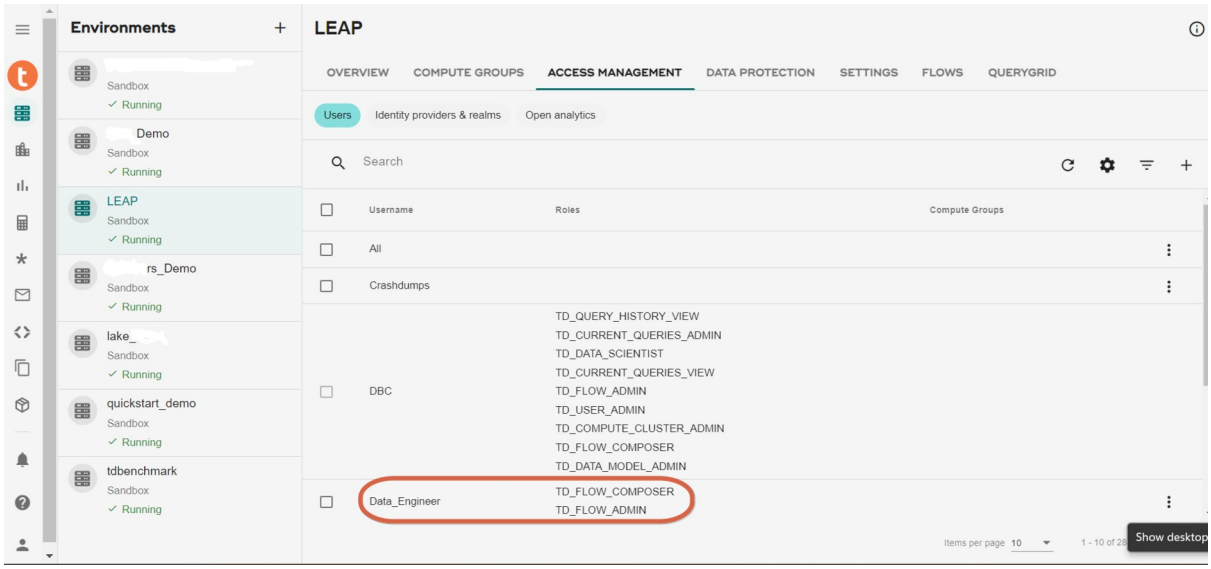
Flow in VantageCloud Lake: Cookbook to Make it Work in AWS

I wrote this document in February 2024. At the time, Teradata had online documentation about Flow. It was the first I was going to use the service, I needed to quickly upload a file into a database and I got stuck. So, I noted everything I did so I could repeat my steps. Here you have my lessons learnt.

Table of Contents

Set up the Flow database user – One-time task

Flow needs a database user to run the jobs. To that end, I created the user Data_Engineer in my environment and assigned the TD_FLOW_COMPOSER and TD_FLOW_ADMIN roles to it.



Additionally, I must grant permissions to the Flow database user (Data_Engineer, in my case) to load the target database.

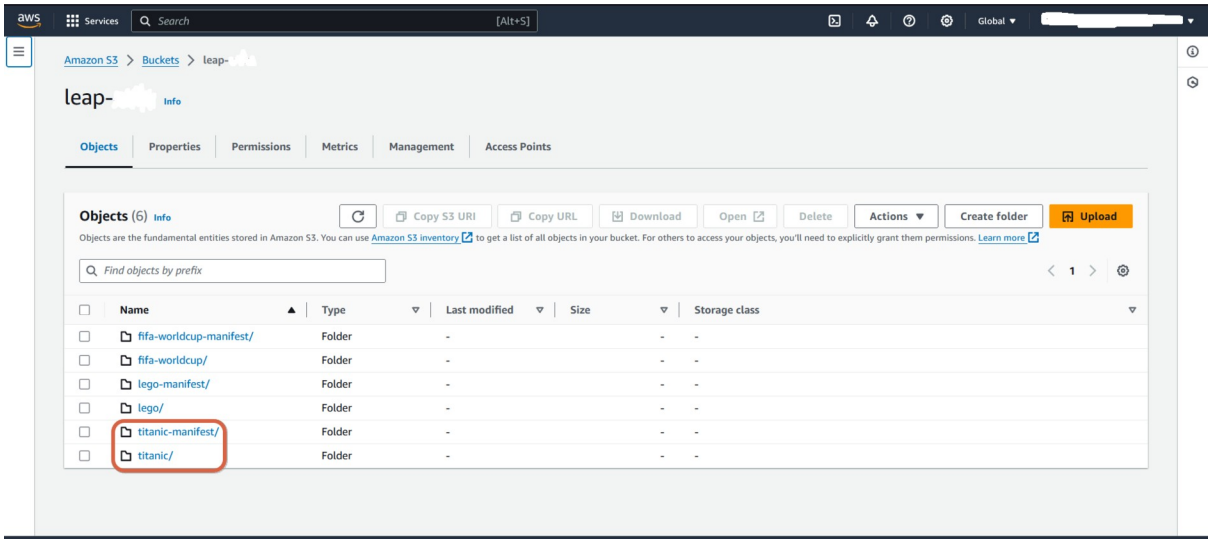
```
grant SELECT, UPDATE, DELETE, INSERT, CREATE TABLE
on <target-database>[1]
to <flow-database-user>;

grant EXECUTE
on <flow-database-user>
to <target-database>
with grant option;
```

Grant permissions to Lake on the AWS S3 bucket where I keep my files – One time task

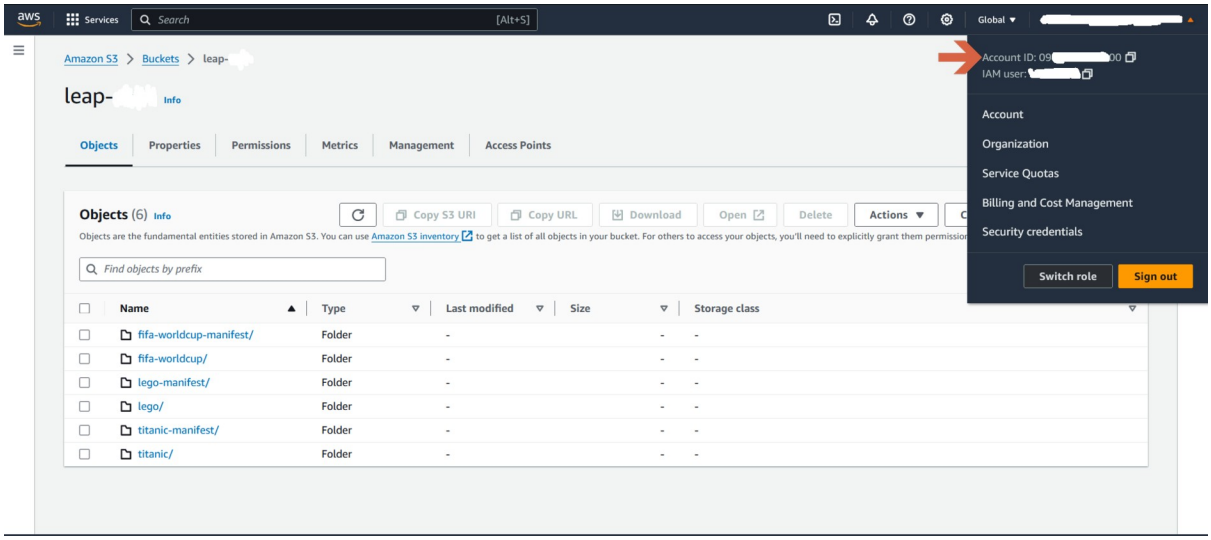
I have one bucket in AWS S3 where I have all the files I need to upload once or recurrently. I must configure the security to allow Lake to read the files in any folders when needed. I only need to configure the security in the bucket once, and then I'll keep my different workloads in separate folders. I used the titanic folder and its content to write this document. Note: I have also created a separate manifest folder to store a file Flow uses to select the files to read.

[1] In this document, all code is in Courier New. When highlighted in green, you should replace it with the appropriate value for your case.



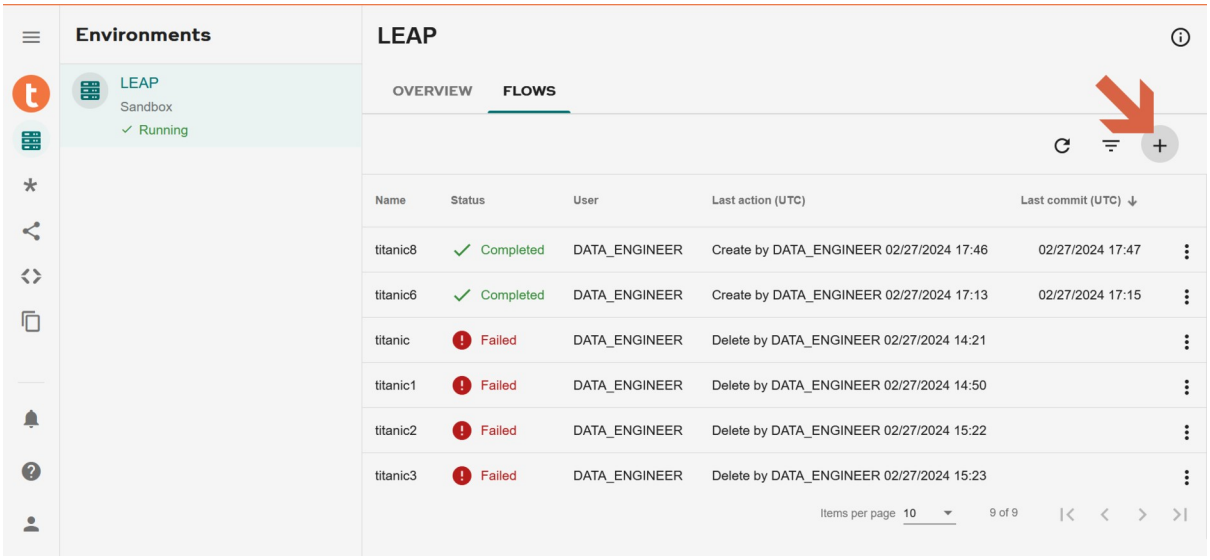
AWS Account ID where my AWS S3 bucket is

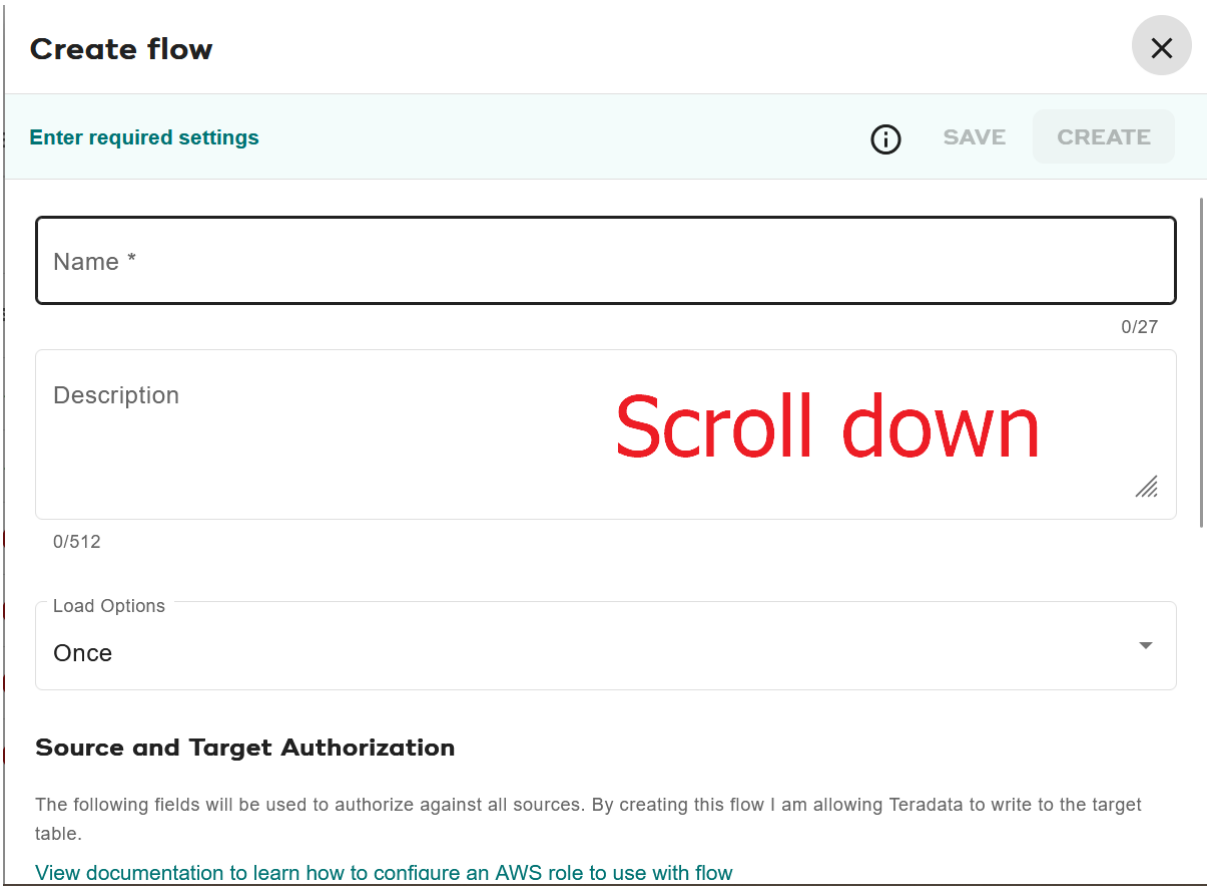
Before granting permissions to Lake on the AWS S3 bucket, I need to annotate and keep handy the AWS Account ID of the account where I keep my AWS S3 bucket. See below where you can find it.

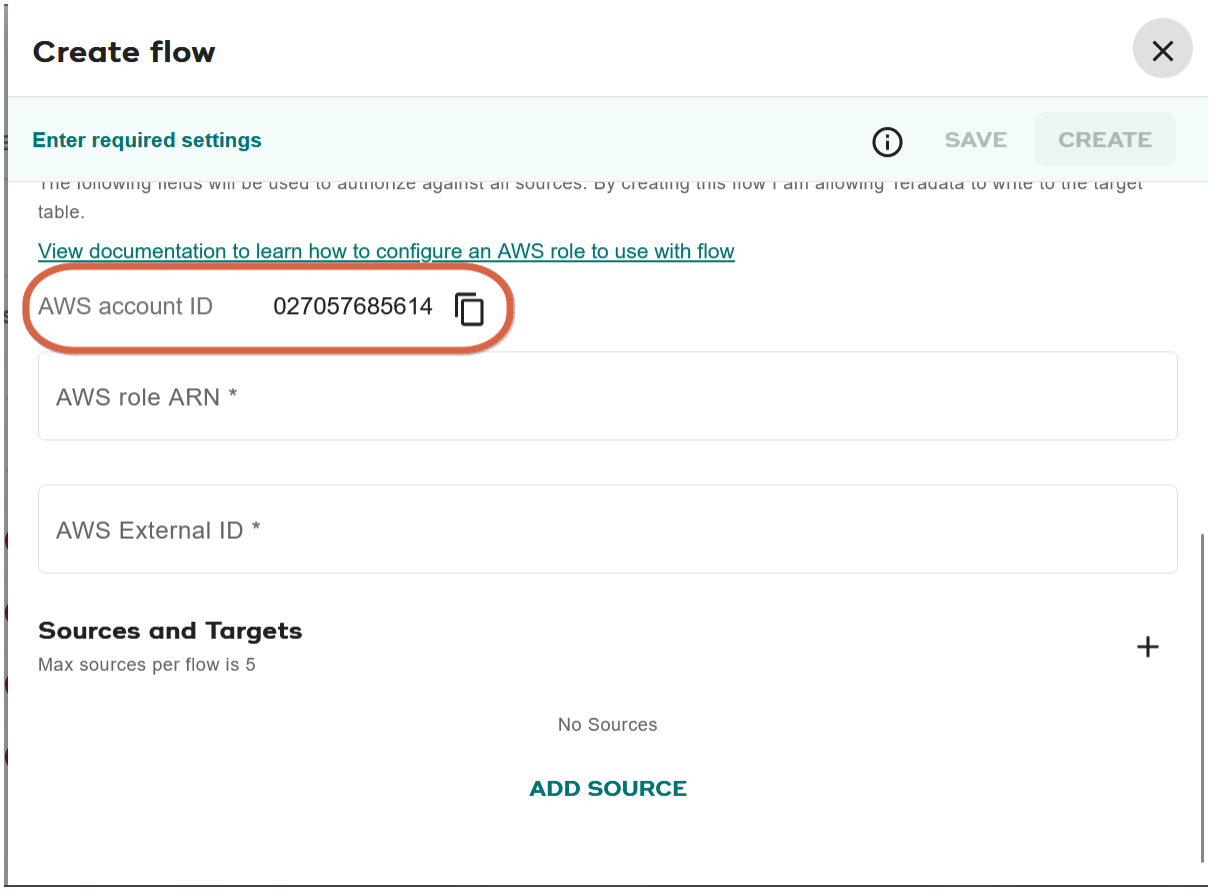


AWS Account ID where my Lake environment is

I also need to know the AWS Account ID of my Lake environment. You can find it in the “Create Flow” screen.


Environments
LEAP
Sandbox
Running
LEAP
OVERVIEW
FLOWS
Name
Status
User
Last action (UTC)
Last commit (UTC)
titanic8 Completed DATA_ENGINEER Create by DATA_ENGINEER 02/27/2024 17:46 02/27/2024 17:47
titanic6 Completed DATA_ENGINEER Create by DATA_ENGINEER 02/27/2024 17:13 02/27/2024 17:15
titanic Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 14:21
titanic1 Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 14:50
titanic2 Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 15:22
titanic3 Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 15:23
Items per page 10
9 of 9

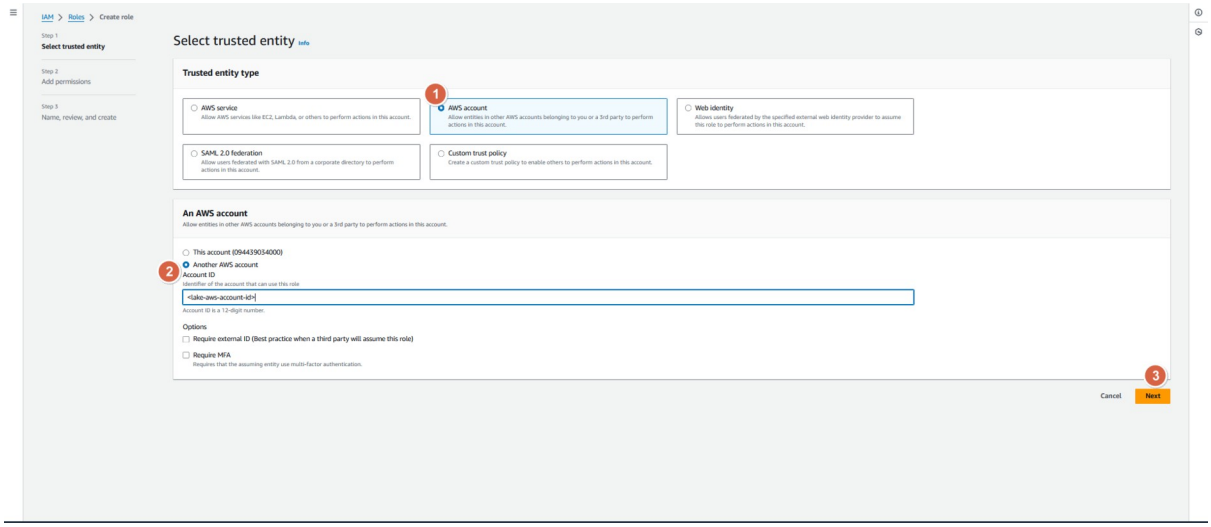

Create flow
Enter required settings
SAVE
CREATE
Name *
0/27
Description
Scroll down
0/512
Load Options
Once
Source and Target Authorization
The following fields will be used to authorize against all sources. By creating this flow I am allowing Teradata to write to the target table.
View documentation to learn how to configure an AWS role to use with flow



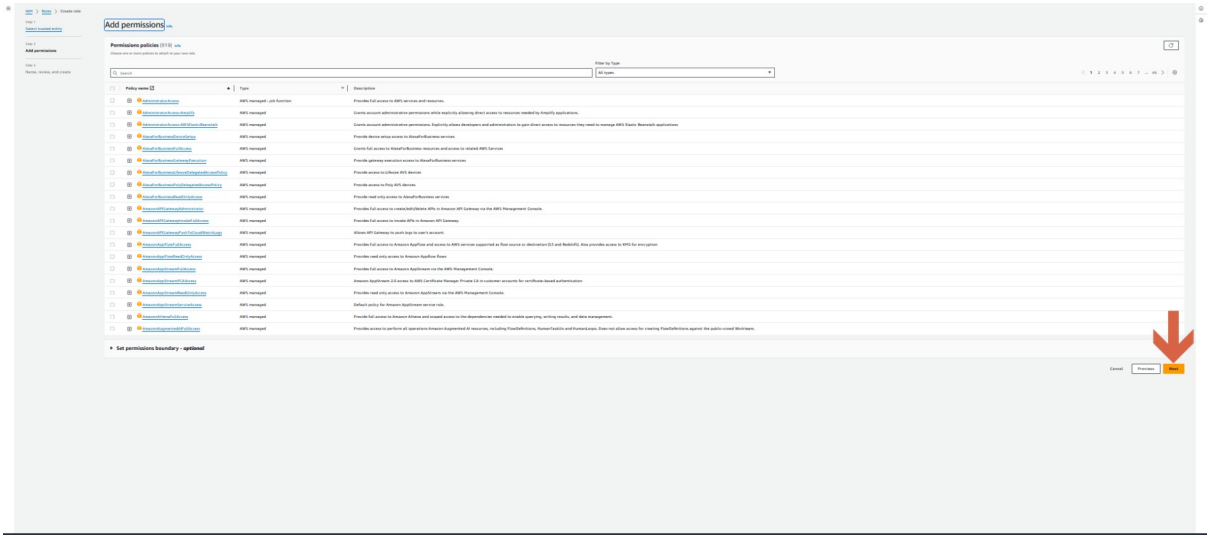
Create an AWS IAM role in the account where you have your bucket

The documentation explains how to create this role, but I'll do it step by step in this section.

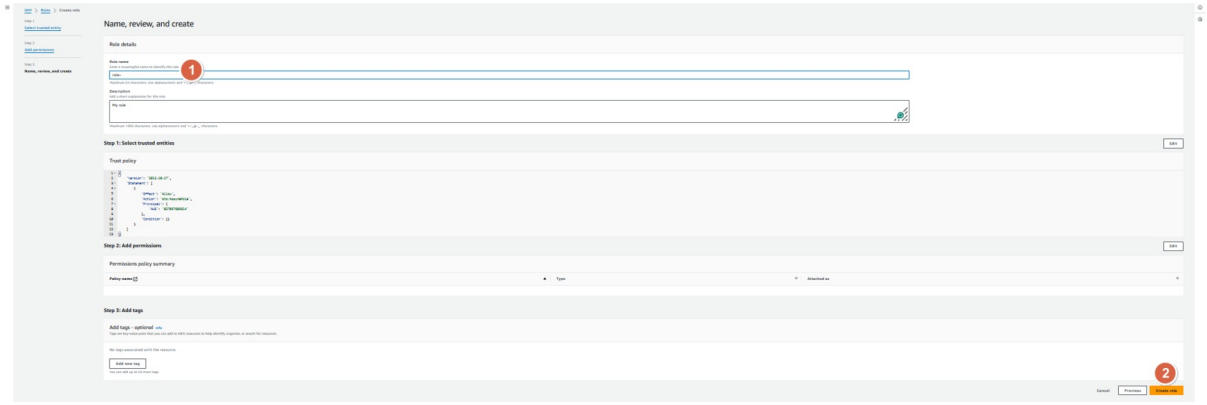
In the AWS Console, go to IAM and create a role as follows:



You don’t need to choose any policy in the next screen. Just click on “Next”.

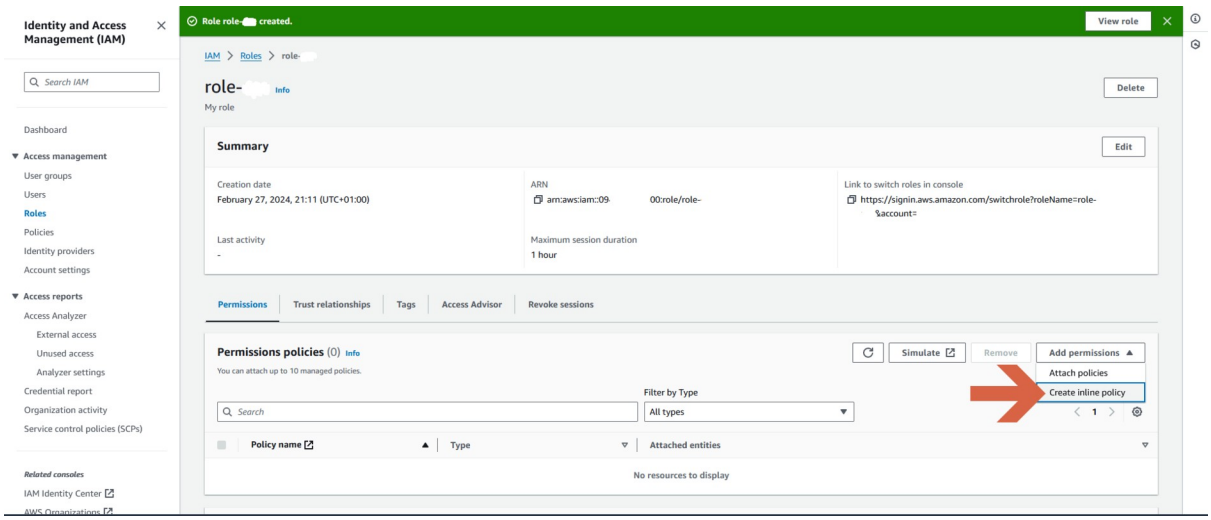


On the following screen, name the role and click on “Create role”.

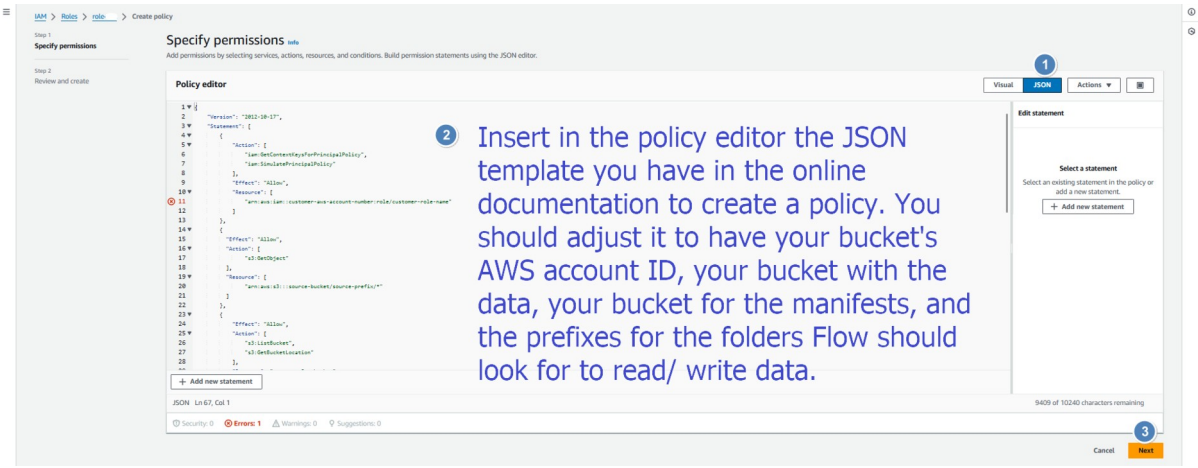


Now, go to the list of roles, search for the one you have just created, and open it.

Then in the Permissions tab, open the drop-down menu “Add permissions” and click on “Create inline policy”.



On the following screen, choose the JSON format, copy the JSON template you will find in the online documentation to create a policy, and correct it with the details of your account.



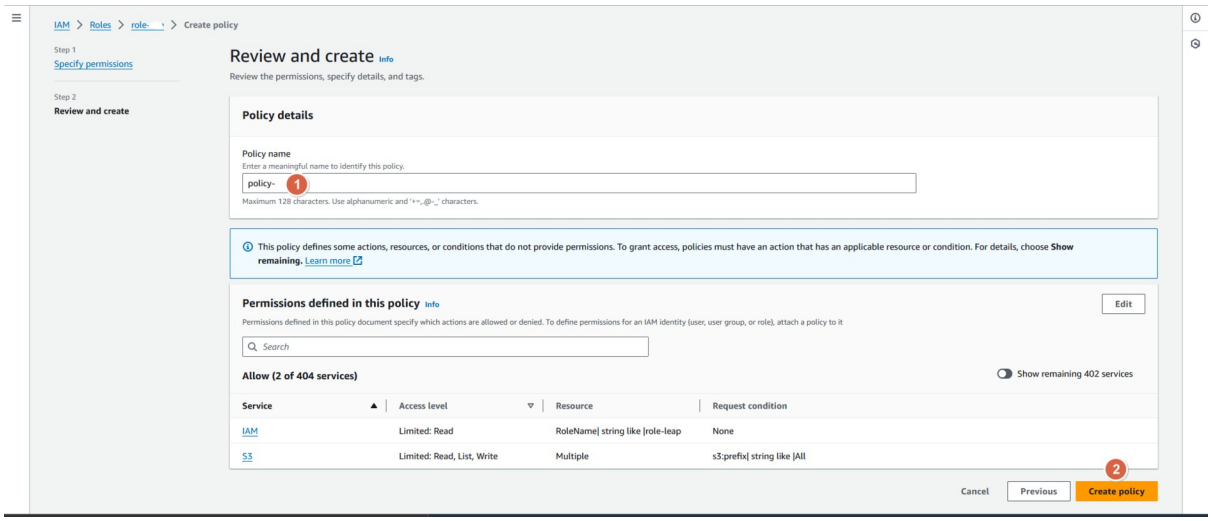
In my case, I want to have one bucket with several folders. In each folder, I store the data needed for every one of my workloads. Additionally, I'll have separate folders for the manifests. So, I use the same bucket name for the data and the manifests, and I don't use prefixes but "*" to include all folders within my bucket. You have my policy.

```
{
                    "Version":          "2012-10-17",
                        "Statement":            [
                                                {
                                    "Action":     [
                "iam:GetContextKeysForPrincipalPolicy",
                            "iam:SimulatePrincipalPolicy"
                                                ],
```

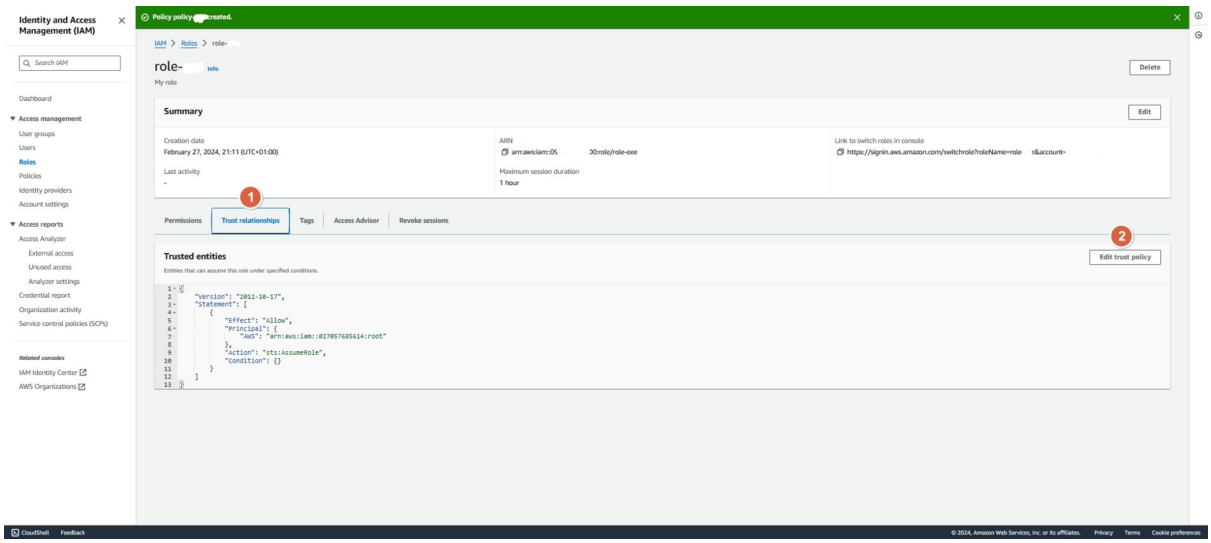
```
                                                "Effect":    "Allow",
                                                    "Resource":     [
                    "arn:aws:iam::<bucket-aws-account-id>:role/<your-
role>"
                                                                    ]
                                                                   },
                                                                    {
                                                "Effect":    "Allow",
                                                       "Action":     [
                                                        "s3:GetObject"
                                                                    ],
                                                     "Resource":     [
                                              "arn:aws:s3:::<your-bucket>/*"
                                                                    ]
                                                                   },
                                                                    {
                                                "Effect":    "Allow",
                                                       "Action":     [
                                                     "s3:ListBucket",
                                                "s3:GetBucketLocation"
                                                                    ],
                                  "Resource":   "arn:aws:s3:::<your-bucket>",
                                                     "Condition":     {
                                                     "StringLike":    {
                                                      "s3:prefix":    [
                                                                   "*"
                                                                    ]
                                                                    }
                                                                    }
                                                                   },
                                                                    {
                                                "Effect":    "Allow",
                                                       "Action":     [
                                                      "s3:PutObject",
                                                   "s3:DeleteObject",
                                                       "s3:GetObject"
                                                                    ],
                                                     "Resource":     [
                                              "arn:aws:s3:::<your-bucket>/*"
                                                                    ]
                                                                   },
                                                                    {
                                                "Effect":    "Allow",
                                                       "Action":     [
                                                     "s3:ListBucket",
                                                "s3:GetBucketLocation"
                                                                    ],
                                  "Resource":   "arn:aws:s3:::<your-bucket>",
```

```
                "Condition":    {
                "StringLike":   {
                  "s3:prefix":   [
                                   "*"
                                 ]
                                 }
                                 }
                                 }
    ]
```

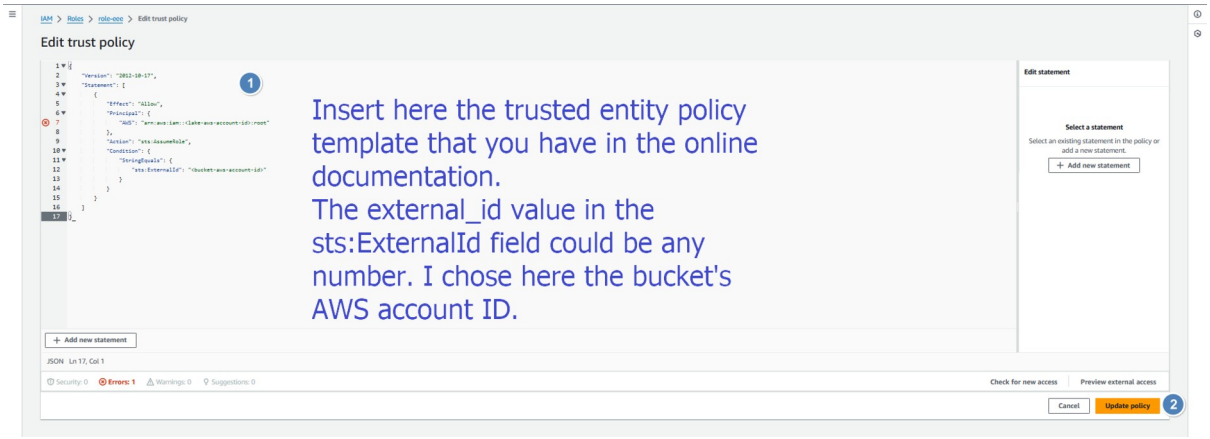
Name the policy and click on “Create policy”.



Back on the role screen, go the “Trusted relationships” tab and click on “Edit trust policy”.



Insert here the [trusted entity policy template] that you have in the online documentation.

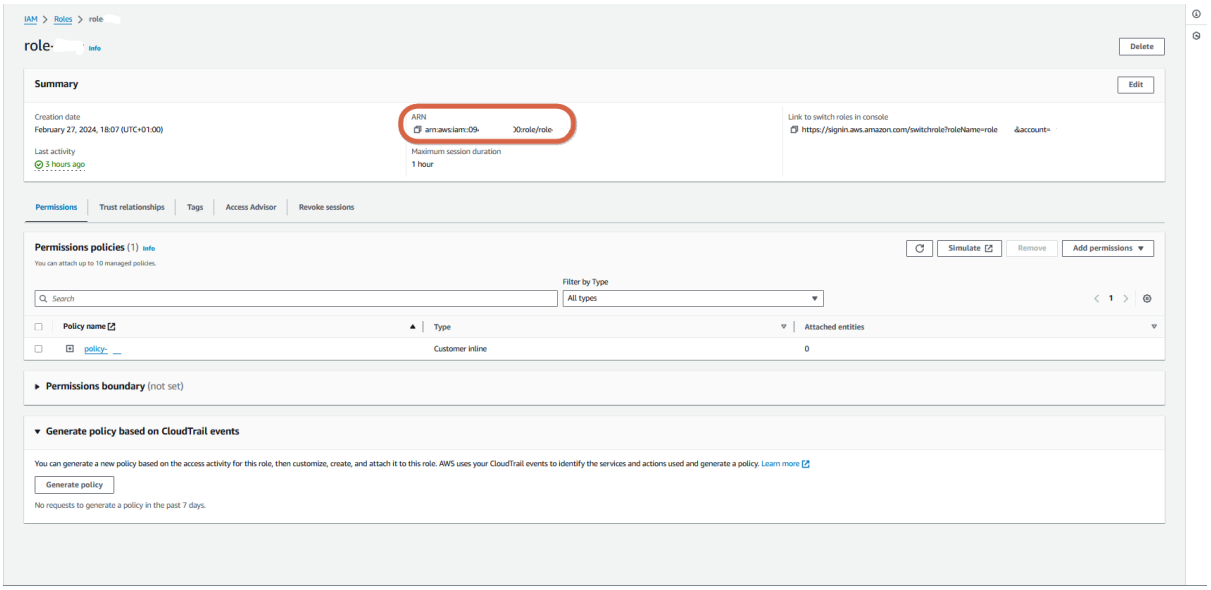


Below you have the trusted entity policy I used.

```
{
                                        "Version":          "2012-10-17",
                                                "Statement":            [
                                                                        {
                                                    "Effect":    "Allow",
                                                        "Principal":     {
                        "AWS":  "arn:aws:iam::<lake-aws-account-id>:root"
                                                                        },
                                                "Action":    "sts:AssumeRole",
                                                        "Condition":     {
                                                    "StringEquals":    {
                                "sts:ExternalId":  "<bucket-aws-account-id>"
                                                                        }
                                                                        }
                                                                        }
                                                                        ]
}
```

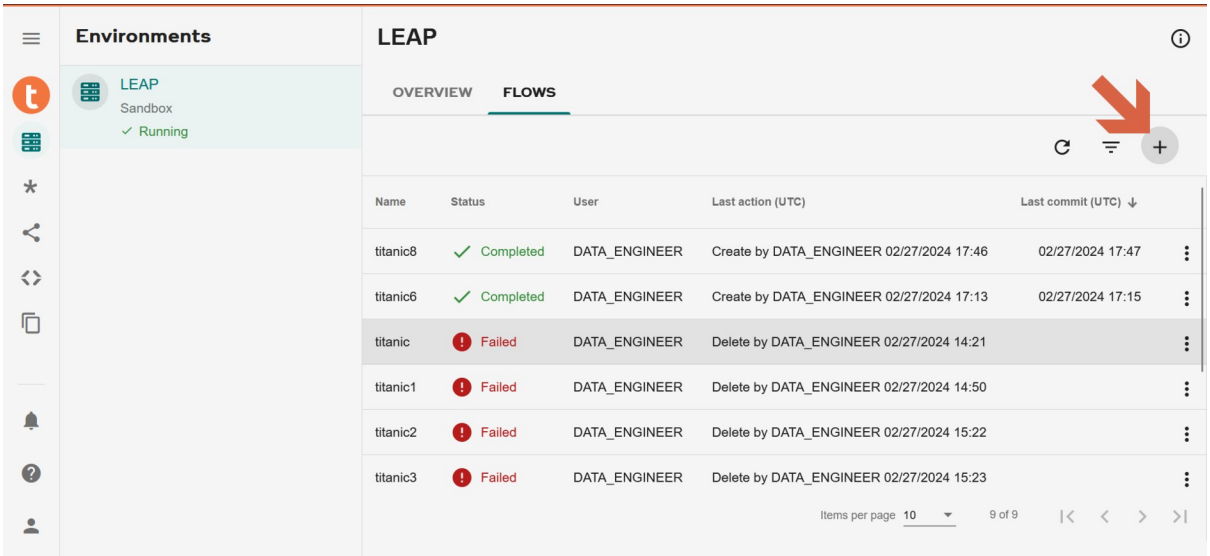
Role ARN

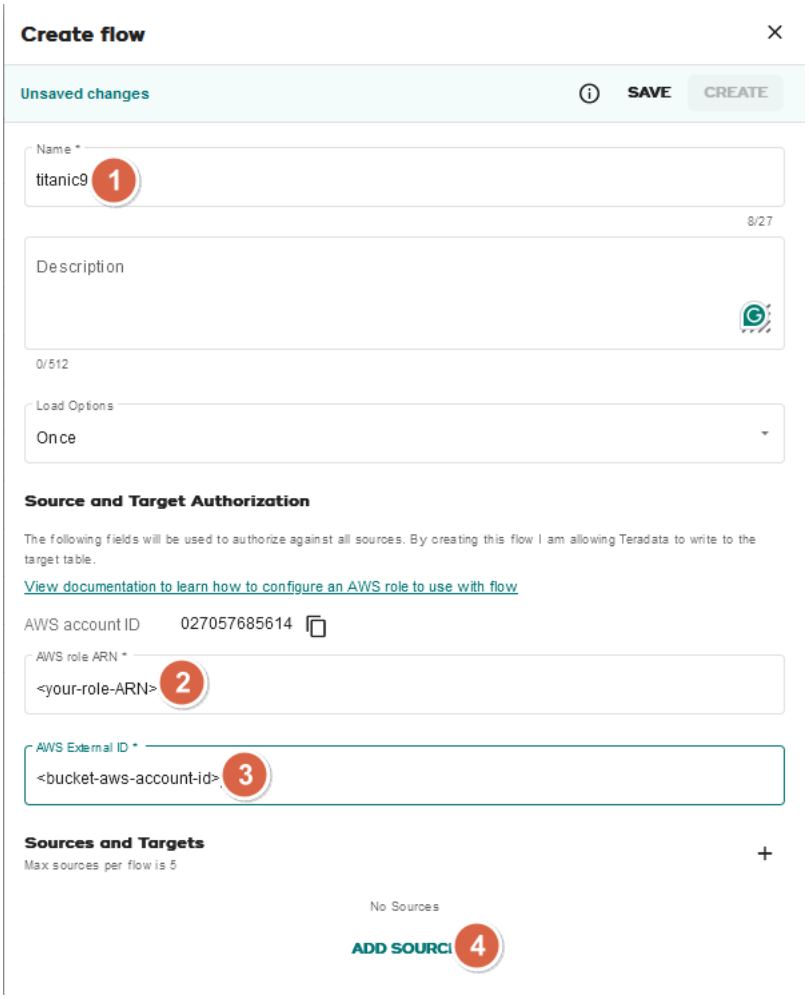
Take a note on the role ARN name you have just created, as you will need it to create flows.

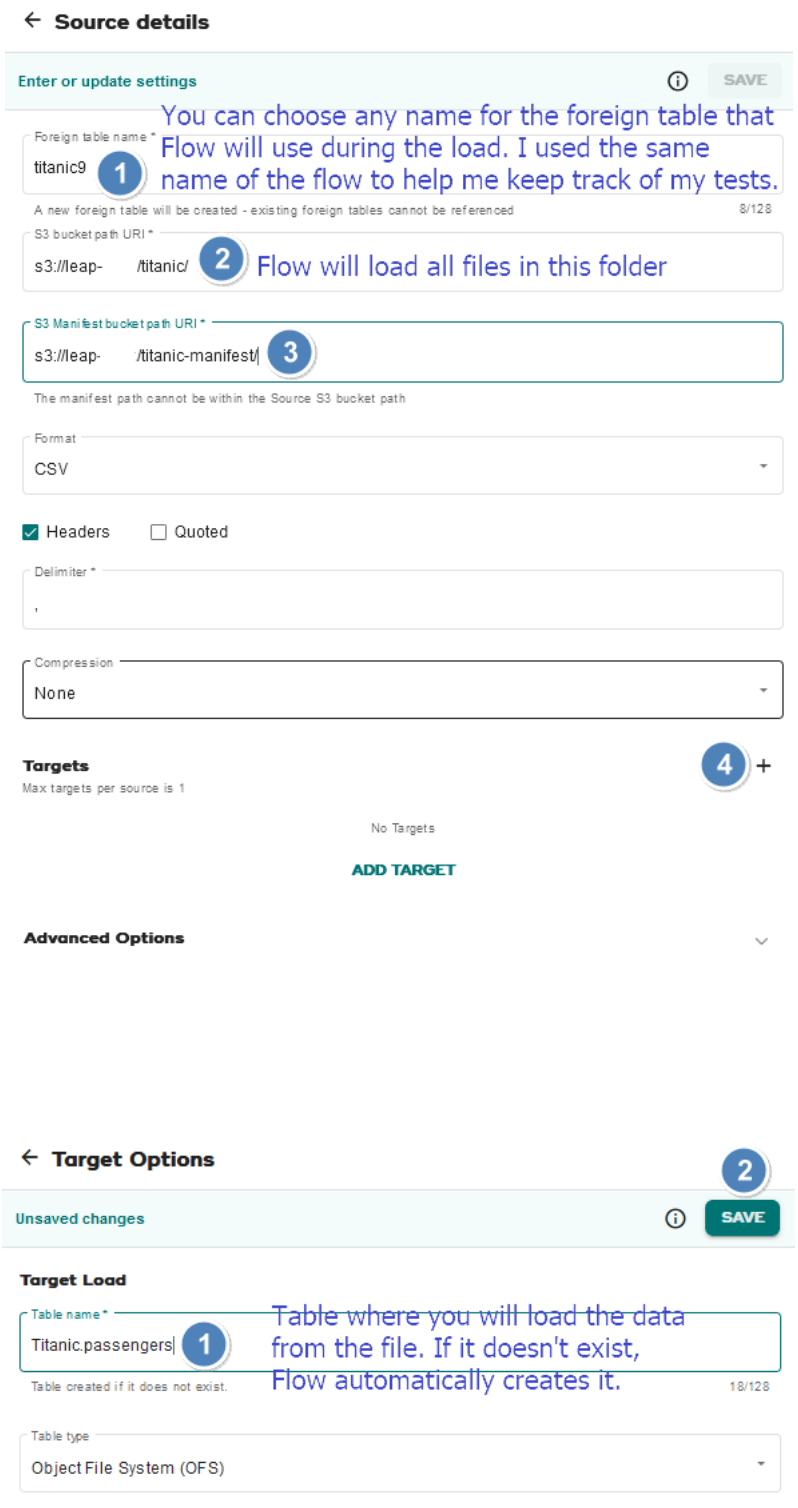


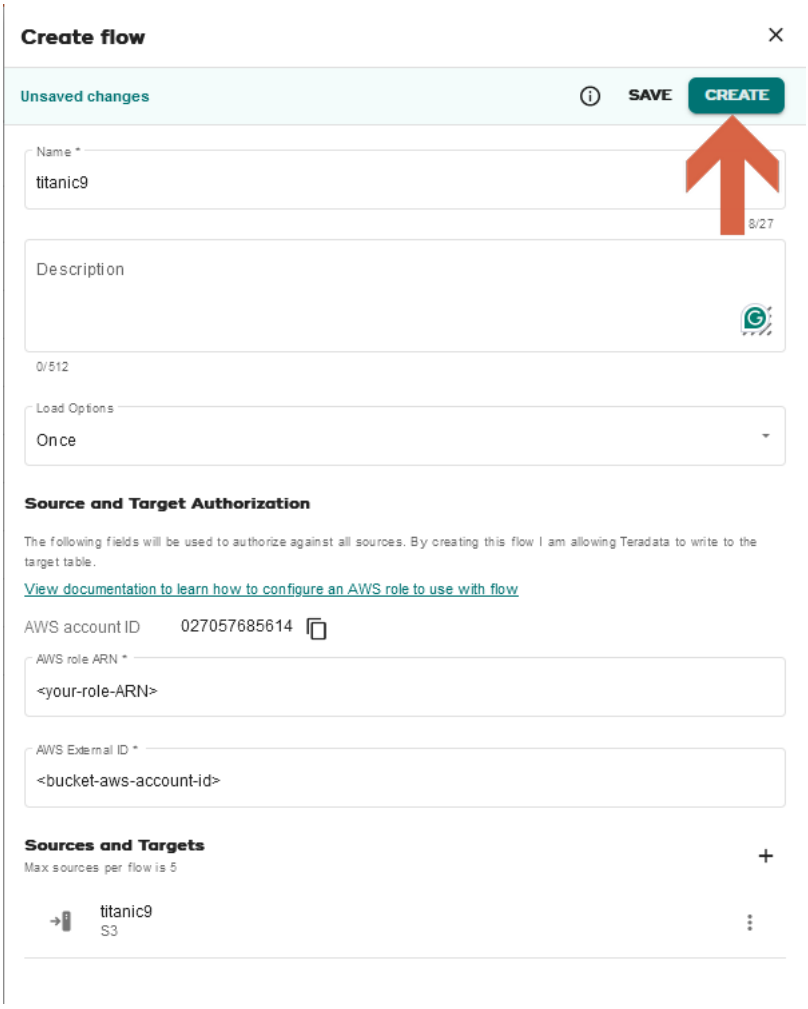
Create a flow – Whenever you need it

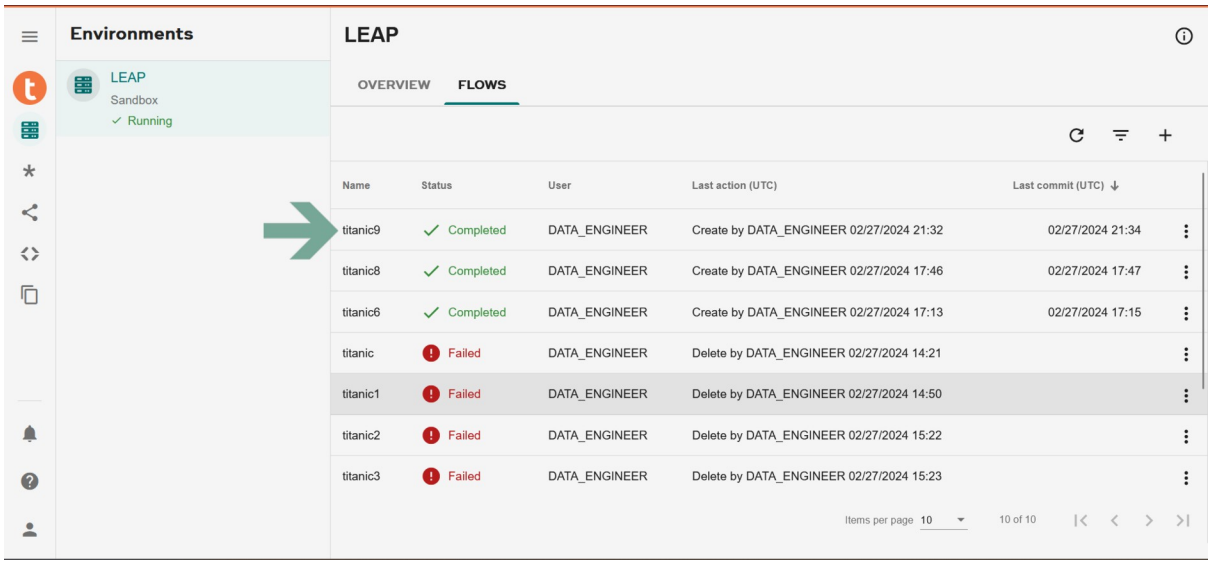
Teradata online documentation explains in detail how to create a flow and what every field means. I'll review the process here as an example.

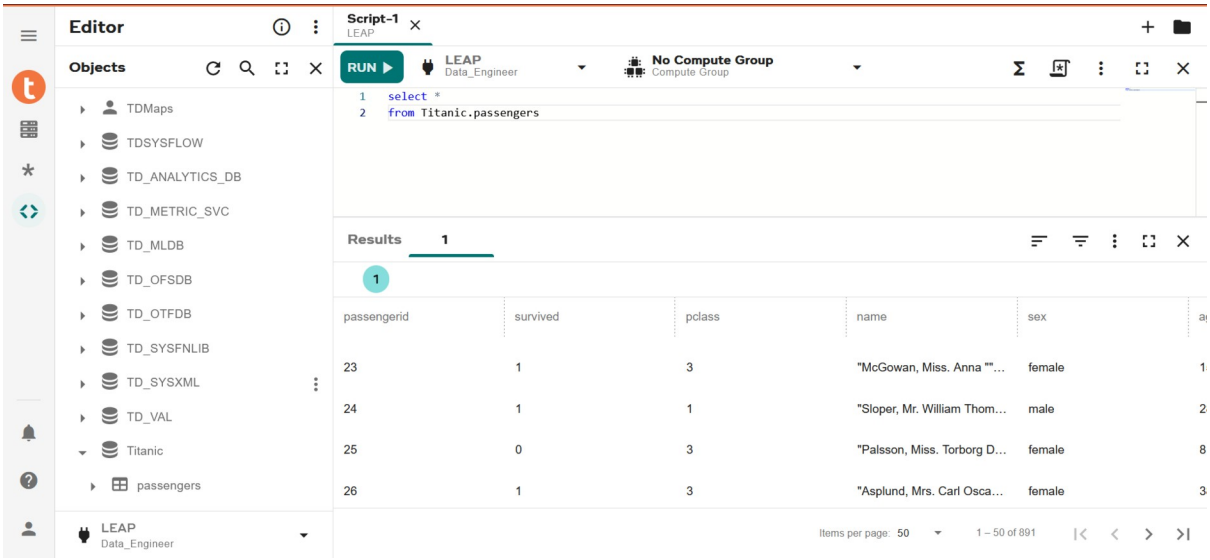



Create flow
Unsaved changes
SAVE
CREATE
Name *
titanic9
1
8/27
Description
0/512
Load Options
Once
Source and Target Authorization
The following fields will be used to authorize against all sources. By creating this flow I am allowing Teradata to write to the target table.
View documentation to learn how to configure an AWS role to use with flow
AWS account ID
027057685614
AWS role ARN *
<your-role-ARN>
2
AWS External ID *
<bucket-aws-account-id>
3
Sources and Targets
Max sources per flow is 5
No Sources
ADD SOURCE
4


Source details
Enter or update settings
SAVE
You can choose any name for the foreign table that Flow will use during the load. I used the same name of the flow to help me keep track of my tests.
Foreign table name *
titanic9
1
A new foreign table will be created - existing foreign tables cannot be referenced
8/128
S3 bucket path URI *
s3://leap- /titanic/
2
Flow will load all files in this folder
S3 Manifest bucket path URI *
s3://leap- /titanic-manifest/
3
The manifest path cannot be within the Source S3 bucket path
Format
CSV
Headers
Quoted
Delimiter *
,
Compression
None
Targets
Max targets per source is 1
4
No Targets
ADD TARGET
Advanced Options
Target Options
2
Unsaved changes
SAVE
Target Load
Table name *
Titanic.passengers
1
Table where you will load the data from the file. If it doesn't exist, Flow automatically creates it.
Table created if it does not exist.
18/128
Table type
Object File System (OFS)


Create flow
Unsaved changes
SAVE
CREATE
Name *
titanic9
8/27
Description
0/512
Load Options
Once
Source and Target Authorization
The following fields will be used to authorize against all sources. By creating this flow I am allowing Teradata to write to the target table.
View documentation to learn how to configure an AWS role to use with flow
AWS account ID
027057685614
AWS role ARN *
<your-role-ARN>
AWS External ID *
<bucket-aws-account-id>
Sources and Targets
Max sources per flow is 5
titanic9
S3


Environments
LEAP
Sandbox
Running
LEAP
OVERVIEW
FLOWS
Name
Status
User
Last action (UTC)
Last commit (UTC)
titanic9 Completed DATA_ENGINEER Create by DATA_ENGINEER 02/27/2024 21:32 02/27/2024 21:34
titanic8 Completed DATA_ENGINEER Create by DATA_ENGINEER 02/27/2024 17:46 02/27/2024 17:47
titanic6 Completed DATA_ENGINEER Create by DATA_ENGINEER 02/27/2024 17:13 02/27/2024 17:15
titanic Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 14:21
titanic1 Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 14:50
titanic2 Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 15:22
titanic3 Failed DATA_ENGINEER Delete by DATA_ENGINEER 02/27/2024 15:23
Items per page 10
10 of 10


Editor
Objects
TDMaps
TDSYSFLOW
TD_ANALYTICS_DB
TD_METRIC_SVC
TD_MLDB
TD_OFSDB
TD_OTFDB
TD_SYSFNLIB
TD_SYSXML
TD_VAL
Titanic
passengers
LEAP
Data_Engineer
Script-1
LEAP
RUN
LEAP
Data_Engineer
No Compute Group
Compute Group
1 select *
2 from Titanic.passengers
Results
1
passengerid
survived
pclass
name
sex
23
1
3
"McGowan, Miss. Anna ""...
female
24
1
1
"Sloper, Mr. William Thom...
male
25
0
3
"Palsson, Miss. Torborg D...
female
26
1
3
"Asplund, Mrs. Carl Osca...
female
Items per page: 50
1 – 50 of 891