

Cloudwick



Runbook for Amorphic Training

Day 2



LAB Session 1

Datasets:

The easiest way to create a Dataset is by cloning.

You can clone a Dataset in Amorphic. The Clone Dataset page auto-populates with the metadata from the existing dataset, so you only need to change the Dataset Name. You can edit any field before clicking Register, and a new dataset will appear in the Datasets page.

Below are some different types of Datasets that are already created for you to view and play with.

1. sample_redshift
2. sample_athena
3. sample_lakeformation
4. sample_s3

Cloning a Dataset.

1. Choose a Dataset you want to clone.
2. Click on the three dots on the right top corner of Dataset Page

The screenshot shows the Amorphic Datasets page. On the left, there's a sidebar with icons for Home, Catalog, Datasets, and Details. The main area displays a list of datasets: 'Immunization', 'Immunization_Staged', and 'sample_redshift'. Below this is a search bar with 'sample_redshift' and a 'Published' filter. To the right, the 'sample_redshift' dataset details are shown: Dataset ID 5101c62d..., Created By harsha a day ago, Last Modified By harsha a day ago. The description is 'Sample Redshift Table'. At the bottom, tabs for DETAILS, PROFILE, FILES, and RESOURCES are visible. On the far right, a context menu is open with options: 'Notification Settings' (disabled), 'Clone Dataset' (highlighted with a red box), 'Delete Dataset', 'Repair Dataset / Generate Dataset Report', and 'Cost Tags'.

3. Change the properties you are interested in.

For this demo lets change the name of a Dataset. Since a Datasets Name needs to be unique for a given Domain, change the Dataset Name to `sample_redshift_<yourname>`. Ex :
sample_redshift_harsha



Screenshot of the Amorphic Datasets interface showing the 'Clone Dataset' process. The 'Metadata' section is highlighted with a red box around the 'Dataset Name' field, which contains 'sample_redshift_yourname'. Other fields in this section include 'Description' (Sample Redshift Table), 'Domain' (Bronze Education (bronzeeducation)), and 'Keywords' (Owner: harsha). The 'Connection Configuration' section shows 'Ingestion Type' (File Upload), 'File Type' (csv), 'Target Location' (Redshift), 'Update Method' (Append), 'Skip File Header' (Yes), and 'Custom Delimiter' (,). The 'Advanced Configuration' section includes 'Enable Malware Detection' (No), 'Enable Data Profiling' (No), and 'Enable Metrics Collection' (No). A 'Next' button is visible at the bottom right.

4. Click on Next. Skip the Cost tags and Press on Continue.

5. Review the changes you made and hit on Clone.

Screenshot of the 'Clone Dataset' interface showing the 'Review and Submit' step. The 'JSON Payload' section displays the configuration details as a JSON object:

```
1 v {
2   "DatasetName": "sample_redshift_harsha",
3   "DatasetDescription": "Sample Redshift Table",
4   "Domain": "bronzeeducation",
5   "Keywords": [
6     "Owner: harsha"
7   ],
8   "ConnectionType": "api",
9   "FileDelimiter": ",",
10  "FileType": "csv",
11  "IsDataProfilingEnabled": false,
12  "TargetLocation": "redshift",
13  "MalwareDetectionOptions": {
14    "ScanForMalware": false,
15    "AllowUnscannableFiles": false
16  },
17  "SkipFileHeader": true,
18  "SkipLZProcess": false,
19  "TableUpdate": "append",
20  "DataMetricsCollectionOptions": {
21    "IsMetricsCollectionEnabled": false
22  },
23  "DatasetType": "internal",
24  "DatasetS3Path": "s3://nvd-us-west-2-816069158041-test-dlz/bronzeeducation/sample_redshift/"
25 }
```

A 'Clone' button is located at the bottom right of the review screen.



6. If Successful you would be taken to the Dataset page as below.

The screenshot shows the Amorphic Dataset Details page for a dataset named "sample_redshift_harsha". The top right corner displays a success message: "sample_redshift_harsha has been created. Dataset registration completed successfully." The main content area shows dataset metadata including Dataset Name, Dataset Type, Target Location, File Type, and Ingestion Type. A prominent yellow bar at the bottom indicates "Registration pending" with a "Click here" button to continue registration.

Dataset Registration:

1. Continue Registering the previous Dataset you created with a valid schema.

The screenshot shows the Amorphic Dataset Details page for a dataset named "sample_redshift_harsha". The top right corner displays a success message: "sample_redshift_harsha has been created. Dataset registration completed successfully." The main content area shows dataset metadata including Dataset Name, Dataset Type, Target Location, File Type, and Ingestion Type. A prominent yellow bar at the bottom indicates "Registration pending" with a "Click here" button to continue registration.

2. You are given different options to register a schema.

- upload a sample data file
- upload a JSON file containing the schema definition
- manually enter the schema fields.



We will choose option C for simplicity. Click on Continue.

sample_redshift_harsha

Define Schema >
Review and Publish >

Schema Entry Type ⓘ *
 Upload Sample Data File
 Upload Schema Definition JSON File
 Create Schema Manually

You have selected to manually create the schema

Continue

3. Click on Add Columns to fill column details.

Select appropriate DataType from the drop down. Set Sort Key Type to None. Leave everything else to Default and click on Publish Dataset.

Note: This page could slightly vary depending on the type of Dataset. Below screenshot is for the redshift Dataset.

sample_redshift_harsha

Define Schema >
Review and Publish >

Build Custom Schema

* required

Set all columns to VARCHAR(256) Type

1. > column1 Column Type* VARCHAR(256) X

2. > column2 Column Type* VARCHAR(256) X

3. > column3 Column Type* VARCHAR(256) X

Add Column

Primary Keys ⓘ Unique Columns ⓘ

Define Identity Column ⓘ

Sort Key Type ⓘ *
None

Distribution Type ⓘ *
Auto

Publish Dataset

You will be redirected to Dataset Page with Notification banner on the side with Success message.



The screenshot shows the Amorphic dataset management interface. A success message at the top right states: "Completed the registration process successfully". The dataset details page for "sample_redshift_harsha" is displayed, showing it was created by harsha a few seconds ago and last modified by harsha a few seconds ago. The description is "Sample Redshift Table". The "DETAILS" tab is selected, showing dataset metadata such as name (sample_redshift_harsha), type (internal), domain (bronzeeducation), target location (Redshift), file type (csv), ingestion type (File Upload), and S3 location (s3://nvd-us-west-2-816069158041-test-diz/bronzeeducation/sample_redshift_harsha/). The "File Options" and "Advanced Options" sections are also visible.

Congrats! Now you have learnt to clone the Dataset from the existing one.



LAB Session 2:

Views in Amorphic:

The easiest way to create a view is by cloning.

You can clone a view in Amorphic. The Clone View page auto-populates with the metadata from the existing view, so you only need to change the view name and update the view name in the view query.

Cloning a View.

1. Choose a View you want to clone.
2. In the top right corner, you can see the clone option.

This screenshot shows the 'Views' section of the Amorphic interface. A specific view named 'immunization_vitals_summary' is selected. In the top right corner of the view's detail card, there is a blue button labeled 'Clone View'. The rest of the page displays the view's ID, creation details, description, and various tabs like 'DETAILS', 'SCHEMA DETAILS', and 'VERSIONS'.

3. For this demo lets change the name of a View. Since a View Name needs to be unique for a given Domain, change the View Name to 'immunization_vitals_summary_<yourname>'.
- Ex : immunization_vitals_summary_harsha

This screenshot shows the 'Clone View' page. It has a sidebar on the left listing the original view 'immunization_vitals_summary'. The main area contains fields for cloning: 'View Name' is set to 'immunization_vitals_summary_yourname'; 'Description' is 'view provides a summary of immunization records,'; and 'Keywords' is 'Owner harsha'. There are also sections for 'Upload Data', 'Metadata' (which is expanded), 'View Configuration', and 'Review and Submit'. A 'Next' button is visible at the bottom right.

4. Click on Next.



5. Change the view name in the query, then click Next.

```
1. create view
2. goldhealth.immunization_vitals_summary_yourname
3. as (select vaccine_product_name,COUNT(vaccine_product_name) as deceased_count from goldhealth.immunization_standard
4. where client_id in (select case_id from goldhealth.vitals_standard where deceased = 'TRUE')
5. group by vaccine_product_name)
```

6. Review the changes you made and hit on Clone.

View Details

View Name	immunization_vitals_summary_yourname
Description	view provides a summary of immunization records, focusing specifically on deceased individuals.
Keywords	Owner:harsha
Data Classifications	-
View Type	Standard
Assume IAM role	no
Domain	goldhealth
Target Location	S3athena

Sql To Create View

```
create view goldhealth.immunization_vitals_summary_yourname as (select vaccine_product_name,COUNT(vaccine_product_name) as deceased_count from goldhealth.immunization_standard where client_id in (select case_id from goldhealth.vitals_standard where deceased = 'TRUE') group by vaccine_product_name)
```

JSON Payload

Congrats! Now you have learnt to clone the View from the existing one.



Connections:

Using a S3 Connection to Ingest data

1. A S3 connection **s3-connection-demo** is already created and shared for you. We will use this connection to ingest data from S3 bucket.

The screenshot shows the Amorphic Connections interface. On the left, there's a sidebar with a search bar and a list of connections: mysql-connection-demo and s3-connection-demo. The main area displays the details for the selected connection, "s3-connection-demo".
Connection Metadata:

- Connection Name: s3-connection-demo
- Connection Type: S3
- Version: 2.4
- Keywords: OH
- Estimated Cost: 0.01

S3 Bucket Metadata:

- S3 Bucket: cw-us-east-1-728673471265-cloudwick-demo
- S3 Bucket Region: us-east-1
- Access Type: bucket_policy

Buttons:

- View Bucket Policy
- View KMS Policy

A context menu is open on the right side of the screen, listing options: Edit Connection, Notification Settings, Clone Connection, Delete Connection, and Cost Tags.

Below picture shows the files present in S3. We will ingest data from the folder "*data/health/Vitals/*" in this demo.



Amazon S3 > Buckets > cw-us-east-1-728673471265-cloudwick-demo > data/ > health/

health/

Objects Properties

Objects (2) Info

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 must grant them permissions. [Learn more](#)

<input type="checkbox"/>	Name	Type	Last modified	Size
<input type="checkbox"/>	<input type="checkbox"/> Immunization/	Folder	-	
<input type="checkbox"/>	<input type="checkbox"/> Vitals/	Folder	-	

2. Head to the Dataset Page and search for Vitals. If you don't see it on your page. Increase the results per page displayed in the settings beside the reload button. If you still do not see a request for access.

Datasets

Home > Catalog > Datasets

+ New Dataset

Increase the Results Per Page

Showing 1 - 2 of 46 record(s)

Dataset Name	Domain	File Type	Target Location	Options
Vitals	bronzehealth	csv	s3athena	<input type="button"/> <input type="button"/> <input type="button"/> <input type="button"/>
vitals_standard	goldhealth	csv	s3athena	<input type="button"/> <input type="button"/> <input type="button"/> <input type="button"/>

Showing 1 - 2 of 46 record(s)

3. Click on Vitals Dataset and clone the Dataset by clicking on the three dots on the right top corner of Dataset Page



The screenshot shows the Amorphic Dataset Details page for a dataset named 'Vitals'. The top navigation bar includes 'Datasets', 'Home > Catalog > Datasets > Details', and a 'New Dataset' button. The main content area has tabs for 'DETAILS', 'PROFILE', 'FILES', and 'RESOURCES'. The 'DETAILS' tab is selected, showing 'Dataset Metadata' and 'Connection Details'. In the 'Dataset Metadata' section, fields include 'Dataset Name: Vitals', 'Dataset Type: internal', 'Domain: bronzehealth', 'Target Location: S3-Athena', 'File Type: csv', 'Ingestion Type: s3', 'Data Classification: -', 'Dataset S3 Location: s3://nvd-us-west-2-816069158041-test-diz/bronzehealth/Vitals/ (with a copy icon)', 'Keywords: OH', 'SerDe: OpenCSVSerde', and 'Delta Lake Table: No'. The 'Connection Details' section shows 'Connection Name: s3-connection-demo (with a copy icon)', 'Connection Id: 6a1fab51-effb-49d6-b817-4acf7018dd (with a copy icon)', and 'Source S3 Bucket Prefix: data/health/Vitals/ (with a copy icon)'. A context menu on the right side of the page lists options like 'Notification Settings', 'Clone Dataset', 'Delete Dataset', 'Repair Dataset / Generate Dataset Report', 'Cost Tags', and 'File Options'.

- Download the schema from Profiles tab. A json file will be downloaded which will contain the schema of the Dataset. (which you will use to register Dataset)

The screenshot shows the Amorphic Dataset Details page for a dataset named 'Vitals'. The top navigation bar includes 'Datasets', 'Home > Catalog > Datasets > Details', and a 'Download' button. The main content area has tabs for 'DETAILS', 'PROFILE', 'FILES', and 'RESOURCES'. The 'PROFILE' tab is selected, showing the 'Schema' section. The schema table lists three columns: 'Name' (case_id, case_file_number, local_file_number), 'Data Type' (varchar(256), varchar(256), varchar(256)), and 'Description' (empty). To the right of the table, a large JSON code block displays the schema definition, starting with:

```
1 v [  
2 v {  
3 v   "columnName": "case_id",  
4 v   "columnType": "varchar(256)",  
5 v   "description": ""  
6 v },  
7 v {  
8 v   "columnName": "case_file_number",  
9 v   "columnType": "varchar(256)",  
10 v  "description": ""  
11 v },  
12 v {  
13 v   "columnName": "local_file_number",  
14 v   "columnType": "varchar(256)",  
15 v   "description": ""  
16 v },  
17 v {  
18 v   "columnName": "state_file_number",  
19 v   "columnType": "varchar(256)",  
20 v   "description": ""  
21 v },  
22 v {  
23 v   "columnName": "first_name",  
24 v   "columnType": "varchar(256)",  
25 v   "description": ""  
26 v }.
```

- Take a look at all the properties. Especially the Connection and Directory Path. The connection **s3-connection-demo** has already been created. Directory path is the path where the data resides in source S3. In this case we are ingesting from *data/health/Vitals/* to the Dataset we are creating.



6. Change the name of Dataset to *Vitals_S3_<yourusername>* Ex: *Vitals_S3_Harsha* and click on Next.
Skip the cost tags and Clone the Dataset.

Clone Dataset

Metadata >
Cost Tags >
Review and Submit >

JSON Payload

```
1 v {  
2   "DatasetName": "Vitals_S3_Harsha",  
3   "DatasetDescription": "Vitals | S3 Ingestion",  
4   "Domain": "bronzehealth",  
5 v   "Keywords": [  
6     "Owner: harsha"  
7   ],  
8   "ConnectionType": "s3",  
9   "ConnectionId": "6a1fab51-fffb-49d6-b817-4acfccf7018dd",  
10  "IsDataValidationEnabled": false,  
11  "SerDe": "OpenCSVSerde",  
12  "DirectoryPath": "data/health/Vitals/",  
13  "FileDelimiter": ",",  
14  "FileType": "csv",  
15  "IsDataCleanupEnabled": false,  
16  "IsDataProfilingEnabled": false,  
17  "LifeCyclePolicyStatus": "Disabled",  
18  "TargetLocation": "s3@athena",  
19  "SkipFileHeader": true,  
20 v  "SkipRowCount": {  
21    "header": 1,  
22    "footer": 0  
23  },  
24  "SkipLZProcess": false,  
25  "TableUpdate": "append",  
26 v  "DataMetricsCollectionOptions": {  
27    "IsMetricsCollectionEnabled": false  
28  },  
29 v  "IcebergTableOptions": {  
30    "IsIcebergTable": false,  
31    "TableProperties": {}  
32  },  
33  "DatasetType": "internal",  
34  "DatasetS3Path": "s3://nvd-us-west-2-816069158041-test-dlz/bronzehealth/Vitals/"  
35 }
```

Clone

7. Continue to register the Dataset using the schema you downloaded from step 4. Upload the schema and wait for the extraction to complete.

← Vitals_S3_Harsha No Schema

Dataset Id a4da0c54... | Created By harsha a few seconds ago | Last Modified By harsha a few seconds ago

Description Vitals | S3 Ingestion

Registration pending
Click here to continue Dataset registration

DETAILS

Dataset Metadata

Dataset Name Vitals_S3_Harsha	Dataset Type internal	Domain bronzehealth
Target Location S3-Athena	File Type csv	Ingestion Type s3
Data Classification -	Dataset S3 Location s3://nvd-us-west-2-816069158041-test-dlz/bronzehealth/Vitals_S3_Harsha/	
Keywords OH	Serde OpenCSVSerde	Delta Lake Table No

File Options

Advanced Options



Define Schema >
Review and Publish >

Schema Entry Type

 Upload Sample Data File Upload Schema Definition JSON File Create Schema Manually[Click here](#) - OR - drag and drop file here

Accepted file type - json

Schema definition JSON Template:

```
1 [ [  
2 {  
3   "columnName": "<Column_Name_1>",  
4   "columnType": "<Integer | Double Precision | Boolean | Varchar(65535) | Varchar(256) | Numeric | Date Time | Date  
5   "description": "<String>"  
6 },  
7 {  
8   "columnName": "<Column_Name_2>",  
9   "columnType": "<Integer | Double Precision | Boolean | Varchar(65535) | Varchar(256) | Numeric | Date Time | Date  
10  "description": "<String>"  
11 },  
12 {  
13   "columnName": "<Column_Name_n>",  
14   "columnType": "<Integer | Double Precision | Boolean | Varchar(65535) | Varchar(256) | Numeric | Date Time | Date  
15   "description": "<String>"  
16 }  
17 ]
```

[Continue](#)



8. Once the Extraction is complete, Scroll to the bottom and publish the Dataset.

Vitals_S3_Harsha

34.	> method_of_disposition	Column Type* VARCHAR(256)	<input type="button" value="X"/>
35.	> cemetery_name	Column Type* VARCHAR(256)	<input type="button" value="X"/>
36.	> cemetery_address	Column Type* VARCHAR(256)	<input type="button" value="X"/>
37.	> informant_name	Column Type* VARCHAR(256)	<input type="button" value="X"/>
38.	> informant_relationship	Column Type* VARCHAR(256)	<input type="button" value="X"/>

Custom Partition Options

You have now successfully created a Dataset!

Scheduling Ingestion:

We can schedule the ingestion from S3 either on-demand or time based. Lets create a Schedule.

1. Navigate to schedules from the Home Page.



Amorphic

+ New Schedule

Job Name	Description	Job Type	Last Modified	Schedule Type	Options
SCH_S3_Vitals_Ingestion	Schedule to Ingest Vitals data	ingestion	6 hours ago	On-Demand	
Ingestion	Schedule to Ingest Vitals data	ingestion	3 days ago	On-Demand	

Schedules

Home > Schedules

+ New

- Clone the Schedule [SCH_S3_Vitals_Ingestion](#) and rename the schedule to SCH_S3_Vitals_Ingestion_<yourname> Ex: SCH_S3_Vitals_Ingestion_harsha

Schedules

Home > Schedules

+ New

Showing 1 - 2 of 2 record(s)

Job Name	Description	Job Type	Last Modified	Schedule Type	Options
SCH_JDBC_Data_Ingestion	Schedule to Ingest Vitals data	ingestion	6 hours ago	On-Demand	
SCH_S3_Vitals_Ingestion	Schedule to Ingest Vitals data	ingestion	3 days ago	On-Demand	

Showing 1 - 2 of 2 record(s)

Clone Schedule



3. Update the Dataset to the one you created earlier and click on continue.

Clone Schedule

(X)

<input type="checkbox"/> Upload Data <input checked="" type="checkbox"/> Metadata <input type="checkbox"/> Cost Tags <input type="checkbox"/> Review & Submit	<div style="border: 2px solid black; padding: 5px; margin-bottom: 10px;"> Schedule Name ⓘ * SCH_S3_Vitals_Ingestion_harsha </div> <div> Description ⓘ Schedule to Ingest Vitals data </div>	* Required Job Type ⓘ * Data Ingestion
		Keywords <div style="border: 1px solid #ccc; padding: 2px; display: inline-block;"> Owner: harsha </div>
Schedule Type ⓘ * On Demand		
<div style="border: 2px solid black; padding: 5px; margin-bottom: 10px;"> Select Dataset ⓘ * Vitals_S3_Harsha s3 </div>		
Max Capacity ⓘ		
New Arguments ⓘ <div style="border: 1px solid #ccc; padding: 2px; display: inline-block;"> New Arguments </div>		

Continue

4. Skip Cost tags > Review > Clone

Clone Schedule

(X)

- Upload Data >
- Metadata >
- Cost Tags >
- Review & Submit >**

Schedule Details	
Job Name	SCH_S3_Vitals_Ingestion_harsha
Job Type	ingestion
Description	Schedule to Ingest Vitals data
Keywords	Owner: harsha
Resource	a4da0c54-ccf4-4c40-8d7b-4c30d73d21bb
Schedule Type	none
Arguments	<input type="text" value="1"/> []
Cost Tags	

JSON Payload

Up Down Copy

Clone

The schedule creation is now complete.



5. Let's run the schedule. To trigger the S3 Ingestion.

The screenshot shows the Amorphic interface for managing schedules. The top navigation bar has a 'New Schedule' button. Below it, a schedule named 'SCH_S3_Vitals_Ingestion_harsha' is displayed as 'ENABLED'. The schedule details include:

- Schedule Name: SCH_S3_Vitals_Ingestion_harsha
- Created By: harsha (a few seconds ago)
- Last Modified By: harsha (a few seconds ago)
- Description: Schedule to Ingest Vitals data
- Job Type: ingestion
- Schedule Type: on-demand
- Status: ENABLED
- Dataset: bronzehealth: Vitals_S3_Harsha
- Keywords: OH

The 'DETAILS' tab is selected. Below it, the 'Additional Metadata' section shows arguments: datasetid (a4da0c54-ccf4-4c40-8d7b-4c30d73d21bb) and MaxCapacity (0.0625).

6. Once the execution is complete, your data would have been loaded to the Dataset you created.

The screenshot shows the Amorphic interface displaying the execution results of the 'SCH_S3_Vitals_Ingestion_harsha' schedule. The results table shows one record:

Start Time	End Time	Job Type	Message	Log Status	Options
2 minutes ago	a minute ago	glue	Copied 1 number of files. For more details, download logs.	-	

The 'EXECUTIONS' tab is selected. A search bar at the top of the results table is visible.



7. You can go back to the Dataset you created and confirm that file has been loaded.

The screenshot shows the Amorphic Catalog interface. On the left, there's a sidebar with icons for Home, Ingestion, Catalog, Transformation, Analytics (which is selected), Workflows, and Schedules. The main area is titled 'Datasets' and shows a list of datasets: 'PH_Vitals', 'Vitals', 'Vitals_S3_Harsha' (which is highlighted), and 'vitals_standard'. Below this, under 'Vitals_S3_Harsha', is a 'FILES' tab. The 'FILES' tab displays a table with one record:

File Name	Last Modified	Message	Tags	Options
...981962_PH_Vitals.csv	harsha 3 minutes ago	Data load completed.	-	

Querying the Ingested S3 Data:

1. Lets query the data we just ingested. Navigate to Analytics > Query Engine and click on it to open.

The screenshot shows the Amorphic Analytics interface. The left sidebar has sections for Home, Ingestion, Catalog, Transformation, Analytics (selected), Workflows, and Schedules. The main area is titled 'Query Engine' and shows a 'New Query' input field with the query 'select * from health.Vitals'. Below the input field are buttons for 'Athena' and 'Primary'. To the right of the input field are 'Clear' and 'Run Query' buttons. The results section below is currently empty.

2. From the Schema Explorer on the left right click on the Dataset you ingested and Click on "Generate Sample Query".



Query Engine

Home > Analytics > Query Engine

SCHEMA EXPLORER HISTORY

Target Location: Athena

System Datasets (nvdtestsystem)

Bronze Education (bronzeducation)

Bronze Health (bronzehealth)

- Datasets (5)
 - gold (gold)
 - Gold Health (gold)
 - Silver Health (silverhealth)
- Immunization
- PH_Immunization
- PH_Vitals
- Vitals
- Vitals_S3_Har...

View Schema Add to Query Generate Sample Query

New Query X New Query X +

1 Select * from ... limit 50

Athena Primary

RESULT

No Results

Clear Run Query

The screenshot shows the Amorphic Query Engine interface. On the left, the 'SCHEMA EXPLORER' panel displays a tree view of datasets and tables under the 'Bronze Health' category. A red arrow points to the 'gold (gold)' dataset. The main area shows a query editor with a single line of code: '1 Select * from ... limit 50'. Below it, the 'RESULT' section says 'No Results'. There are 'Athena' and 'Primary' tabs at the bottom of the query editor.

3. The query will be autopopulated to you. Run the query and wait for it to complete. You will have the results displayed.

Query Engine

Home > Analytics > Query Engine

SCHEMA EXPLORER HISTORY

Target Location: Athena

System Datasets (nvdtestsystem)

Bronze Education (bronzeducation)

Bronze Health (bronzehealth)

- Datasets (5)
 - gold (gold)
 - Gold Health (goldhealth)
 - Silver Health (silverhealth)
- Immunization
- PH_Immunization
- PH_Vitals
- Vitals
- Vitals_S3_Har...

View Schema Add to Query Generate Sample Query

New Query X New Query X +

```
1 select * from
2 bronzehealth.Vitals_S3_Harsha
3 limit 50
```

Athena Primary

RESULT

Query Results:

case_id	case_file_number	local_file_number	state_file_number	first_name	last_name	...
00901a88-59b2-4acc-a...	CF-356	LF-188	SF-238	Saltzman327	Elaine169	J...
03b86e36-59f5-4db3-a...	CF-478	LF-346	SF-783	Washington743	Darius291	N...
069335e0-addd-4969-b...	CF-159	LF-277	SF-154	Johnson823	Terrell657	T...
06e441e3-2ed3-4f08-b...	CF-656	LF-909	SF-895	Nguyen378	Minh621	J...

Clear Run Query

The screenshot shows the Amorphic Query Engine interface after running the query. The 'RESULT' section now displays a table of data from the 'Vitals_S3_Harsha' table. The table has columns: case_id, case_file_number, local_file_number, state_file_number, first_name, last_name, and an ellipsis. There are five rows of data, each with a red arrow pointing to the 'case_id' column. The 'Athena' and 'Primary' tabs are at the bottom of the query editor.

Congrats! You have learnt to Ingest the S3 Data and Query it on Amorphic!



Creating a JDBC Connection

- Clone the JDBC Connection: mysql-connection-demo by clicking on 3 dots and click on Clone Connection.

The screenshot shows the Amorphic Connections interface. On the left is a sidebar with various icons. The main area displays a list of connections: 'mysql-connection-demo' and 's3-connection-demo'. Below this is a detailed view of 'mysql-connection-demo'. The 'DETAILS' tab is selected, showing the following information:

- Connection Name:** mysql-connection-demo
- Connection Type:** JDBC
- Estimated Cost:** 0.14
- Keywords:** DM, OH

Under the 'Jdbc Connection Metadata' section, the following details are shown:

- Data Load Type:** bulkdataoadv1
- Username:** admin
- Publicly Accessible:** yes

The JDBC Connection URL is listed as: `jdbc:mysql://database-1-instance-1.cozbikcm76r.us-east-1.rds.amazonaws.com:3306/demo`.

- Leave the ConnectionType as JDBC and click on Continue.

The screenshot shows the 'Clone Connection' dialog box. On the left is a sidebar with navigation links: Upload Data, Connection Type, Metadata, Connection Configuration, Cost Tags, and Review & Submit. The 'Connection Type' link is highlighted. The main area has two sections:

- Database Type:** A row of icons for various databases: Oracle, MySQL, PostgreSQL, Microsoft SQL Server, IBM Db2 (LUW), Amazon Redshift, Amazon Aurora, Maria DB, and Aurora MySQL. The MySQL icon is selected.
- Connection Type:** A row of icons for S3, External API, and Email. The JDBC icon is selected.

A red asterisk (*) is present next to the 'Required' field for the Database Type and Connection Type sections. At the bottom right is a 'Continue' button.

- Change the name of connection from mysql-connection-demo to mysql-connection-demo-<yourusername> Ex: *mysql-connection-demo-harsha*



Fill the password: Cloudwick#123 and click on Continue.

Clone Connection

Upload Data >

Connection Type >

Metadata >

Connection Configuration > * Required

Cost Tags >

Review & Submit >

Data Load Type ⓘ * Bulk Data Load

Username ⓘ * admin

JDBC Connection URL ⓘ * jdbc:mysql://database-1-instance-1.cozpbikcm76r.us-east-1.rds.amazonaws.com:3306/demo

Click here ⓘ for pre-reqs

Connection Accessibility ⓘ

Private Connection

Publicly Accessible Connection

Continue

4. Skip Cost Tags, Click on Review and Proceed to Next page and click on Clone.

Clone Connection

Upload Data >

Connection Type >

Metadata >

Connection Configuration >

Cost Tags >

Review & Submit >

Connection Details

Connection Type	jdbc
Connection Name	mysql-connection-demo-harsha
Description	Connection of MySQL Database Ingestion
Keywords	DB : MySQL, Owner: harsha
Public Accessibility	yes
Data Load Type	bulkdataloadv1
Password	***** ⓘ
JdbcURL	jdbc:mysql://database-1-instance-1.cozpbikcm76r.us-east-1.rds.amazonaws.com:3306/demo
Cost Tags	

JSON Payload

Clone



5. You can test the connection to see its working as below by clicking on test connection

mysql-connection-demo-harsha

Connection Id 90543a1d... | Created By harsha a few seconds ago | Last Modified By harsha a few seconds ago

Description Connection of MySQL Database | Ingestion

DETAILS TASKS INSTANCES RESOURCES

Connection Metadata

Connection Name	mysql-connection-demo-harsha	Connection Type	JDBC
Estimated Cost	N/A	Keywords	DM OH

Jdbc Connection Metadata

Data Load Type	bulkdataloadv1	Username	admin	Publicly Accessible
JDBC Connection URL	jdbc:mysql://database-1-instance-1.cozbikcm76r.us-east-1.rds.amazonaws.com:3306/demo			

yes

6. You have successfully created a mysql connection.

Connections

mysql-connection-demo-harsha

Connection Id 90543a1d... | Created By harsha a few seconds ago | Last Modified By harsha a few seconds ago

Description Connection of MySQL Database | Ingestion

DETAILS TASKS INSTANCES RESOURCES

Connection Metadata

Connection Name	mysql-connection-demo-harsha	Connection Type	JDBC
Estimated Cost	N/A	Keywords	DM OH

Jdbc Connection Metadata

Data Load Type	bulkdataloadv1	Username	admin	Publicly Accessible
JDBC Connection URL	jdbc:mysql://database-1-instance-1.cozbikcm76r.us-east-1.rds.amazonaws.com:3306/demo			

Connection created Successfully created connection



Tasks:

Tasks automate the data ingestion process in Amorphic.

1. To create a task navigate to Tasks tab and click on Create Task

The screenshot shows the Amorphic interface for managing connections. On the left, there's a sidebar with various icons. The main area is titled 'Connections' and shows a list of connections: 'mysql-connection-demo', 'mysql-connection-demo-harsha' (which is selected), and 's3-connection-demo'. Below the list, there are tabs for 'DETAILS', 'TASKS' (which is active), 'INSTANCES', and 'RESOURCES'. A large central panel displays the message 'No Tasks Found!' and features a cartoon illustration of two people interacting with a computer screen. At the bottom of this panel are three buttons: 'Reload', 'Create new Task', and 'Notification Settings'.

2. Fill the Details for Task Name, Migration Type, Target Location as below and click on continue.
 - A. Task Name: task-a-<yourusername>
 - B. Migration Type: Full Load
 - C. Target Location: s3athena
 - D. Data Format: parquet

The screenshot shows the 'New Task' configuration page. On the left, a sidebar titled 'Task Specs' lists four sections: 'Replication Configuration', 'Table Selection', 'Ingestion Configuration', and 'Preview'. The main form area contains several input fields:

- 'Task Name' is set to 'task-a-harsha'.
- 'MigrationType' is set to 'Full Load'.
- 'Target Location' is set to 'redshift'.
- 'Sync To S3' is set to 'Yes'.
- 'Data Format' is set to 'parquet'.

At the bottom right of the form is a 'Continue' button. The top of the page shows the navigation path: Home > Ingestion > Connections > mysql-connection-demo-harsha > New Task.



3. Proceed to Replication Configuration.

For this demo we will create an on demand instance instead of a shared instance. Hence Choose No for Shared Instance and Serverless Instance.

The right size and storage will be automatically configured based on the table size.
Hence you can skip filling the details for replication instances.

Task Specs

Replication Configuration

Table Selection

Ingestion Configuration

Preview

Use Serverless Replication ⓘ
No

Use Shared Instance ⓘ

Allocated Storage ⓘ

OR

Instance AZ, Instance class & Allocated storage are all optional, however, if one is provided you must provide the other values as well.

Continue

4. Choose the table you want to Ingest and click on continue.

Let's ingest one table for the purpose of this demo. Choose the tables PH_Immunization and PH_Vitals and Click on Continue.

Task Specs

Replication Configuration

Table Selection

Ingestion Configuration

Preview

Filter Schemas

demo

Select All

Filter By Table Name

Courses

Graduated

PH_Vitals

defendents

shots_fired

Enrollments

PH_Immunization

Students

pet

Continue

5. Some info is already prefilled for you in the next page.

Let's put the data into the correct Domain and Change the Dataset Name using Bulk Edit.

Click on Bulk Edit > Dataset Name > Update the Prefix and Suffix as below.



Remove the Dataset Prefix and add _<yourusername> to Dataset Suffix and Update All Dataset Names.

Bulk Update Amorphic Dataset Names

Dataset Name Prefix ⓘ

Update All Dataset Names

Similarly lets update Domain Name:

Bulk Edit > Domains > Choose the appropriate Domain > Update All Domains.

- Once done, review the changes on the page to see if the updates are reflected and Click on Continue.

< New Task for mysql-connection-demo-harsha

Home > Ingestion > Connections > mysql-connection-demo-harsha > New Task

Task Specs >

Replication Configuration >

Table Selection >

Ingestion Configuration > **Selected**

Preview >

Bulk Edit ▾

Filter by Schema Name

demo

PH_Vitals >

PH_Immunization >

Basic Table Options

Dataset Name ⓘ PH_Vitals_harsha

Domain Name ⓘ Bronze Health (bronzehealth)

Approx Table Size ⓘ Small (< 2GB)

Keywords ⓘ Owner:harsha

Enable Filters, Transforms

Continue

Note: You can also Toggle on Enable Filters, Transforms to filter your data that you are ingesting and Choose any transforms from a list of available [filter and transformation rules](#). (We are skipping this for this demo)



< New Task for mysql-connection-demo-harsha

Home > Ingestion > Connections > mysql-connection-demo-harsha > New Task

Task Specs >

Replication Configuration >

Table Selection >

Ingestion Configuration >

Preview >

Bulk Edit <

Filter by Schema Name demo PH_Immunication

Basic Table Options

Dataset Name **demo** PH_Immunication_harsha

Domain Name **Bronze Health (bronzehealth)**

Dataset Description This dataset is loaded as part of connection task from PH_Immunication table in demo schema

Approx Table Size Small (< 2GB)

Keywords Owner: harsha

Sort and Distribution

Sort Key Type

Distribution type

Filters

Filter By Column

Add New Filter

Transforms

Transform Column

Add New Transform Rule

Continue >

7. Preview the changes and click on Create.

< New Task for mysql-connection-demo-harsha

Home > Ingestion > Connections > mysql-connection-demo-harsha > New Task

Task Specs >

Replication Configuration >

Table Selection >

Ingestion Configuration >

Preview >

Task Details

Task Name task-a

Migration Type full-load

Target Location redshift

Task S3Payload no

Shared Instance no

Instance AZ us-west-2b

Instance Class dms.t3.medium

DMS Version 3.5.3

Allocated Storage 50

Sync To S3 yes

Data Format parquet

Target Table Prep Mode -

Datasets Count 1

Rules Count 3

JSON Payload

Create >



8. Back to the main page you will see the Task Creation has been successfully completed after a min.

The screenshot shows the Amorphic interface under the 'Connections' section. A connection named 'mysql-connection-demo-harsha' is selected. The 'TASKS' tab is active, displaying a single task entry:

Task Name	Migration Type	Shared Instance	Target Location	Message	Options
task-a-harsha	full-load	no	redshift	Task datasets registration completed	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="checkbox"/>

A tooltip 'Showing 1 - 1 of 1 record(s)' is visible below the table.

9. You can click on it to expand and see the properties you configured.

The screenshot shows the expanded details for the task 'task-a-harsha'. The 'DETAILS' tab is active, displaying the following configuration:

Created By	harsha	a minute ago
Last Modified By	harsha	a minute ago
Migration Type	full-load	
Status	ready	
Target Location	redshift	
Sync to S3	yes	
Data Format	parquet	
Target Table Prep Mode	truncate	
Allocated Storage	-	

Below the configuration, there are five icons with labels: 'Migration Overview', 'Table Statistics', 'Logs', 'Payload', and 'Runs'.

10. Go back to the Tasks and start the task to ingest data.

The screenshot shows the task 'task-a-harsha' again. The 'DETAILS' tab is active, and the task status is now 'ready'. The 'Start Task' button is highlighted with a black border.



This process would take some time depending on the size of the tables.

11. You have now learnt to create JDBC connections and Bulk Load Tables using DMS Task on Amorphic.
Cheers!



LAB Session 3:

This is focused for Data Engineers

Paramstore:

In today's session let's start by creating a Parameters Store key.

1. Create a new parameter in the Parameter Store by navigating to the Parameter Store section under Transformations.(Here, you'll also find some default system-generated parameters, such as bucket names, AWS region, etc.)

The screenshot shows the AWS Lambda Parameters Store interface. The top navigation bar includes the Amorphic logo, a search bar, and a 'Create Parameters Store' button. The main content area displays a table with the following columns: Parameters Store Key, Scope, Parameters Store Type, and Options. The table lists several parameters, including a user-defined parameter 'nvd-sample-param-adityabhat' and various system-generated parameters like 'SYSTEM.S3BUCKET.ATHENA' and 'SYSTEM.AWSREGION'. Each row in the table has a set of edit icons (pencil, delete, etc.) in the 'Options' column.

Parameters Store Key	Scope	Parameters Store Type	Options
nvd-sample-param-adityabhat	global	SecureString	
SYSTEM.S3BUCKET.ATHENA	global	String	
SYSTEM.S3BUCKET TEMP	global	String	
SYSTEM.S3BUCKET.LZ	global	String	
SYSTEM.S3BUCKET.DLZ	global	String	
SYSTEM.NOTIFICATIONS.QUEUE	global	String	
SYSTEM.AWSREGION	global	String	
SYSTEM.ENVIRONMENT	global	String	
SYSTEM.S3BUCKET.ETL	global	String	

2. Name the parameter **sample-param-<yourusername>**. Ex: **nvd-sample-param-adityabhat**. Choose Parameters Store Type as Secure String, Scope as Global, Fill in Parameter Store Value.



3. Once you've filled in the details, click on **Add Parameter Store**.

The screenshot shows the Amorphic Parameters Store interface. On the left, there is a list of existing parameters with their keys and scopes. On the right, a modal window titled "Add New Parameters Store" is open, prompting for new parameter details.

Parameters Store Key	Scope
nvd-sample-parm-adityabhat	global
SYSTEM.S3BUCKET.ATHENA	global
SYSTEM.S3BUCKET.TEMP	global
SYSTEM.S3BUCKET.LZ	global
SYSTEM.S3BUCKET.DLZ	global
SYSTEM.NOTIFICATIONS.QUEUE	global
SYSTEM.AWSREGION	global

Add New Parameters Store

Tenant <small>(*)</small>	Default tenant
Parameters Store Type <small>(*)</small>	Secure String
Scope <small>(*)</small>	Global
Description <small>(*)</small>	A secure string from the parameter store, created for a training session to store sensitive information.

Add Parameters Store

Congratulations! You have now learned how to create a parameter in the Parameter Store.

ETL Library:

1. Navigate to the ETL Library page. Transformations > ETL Library
2. Let's take a look at ETL Library as a reference.
 - A. amorphicutils-v0-3-1-pyspark
 - B. amorphicutils-v0-3-1-python

The screenshot shows the Amorphic ETL Library interface. It displays a list of two ETL libraries, each with its name, creation time, last modified time, access type, packages, jobs attached, and created by user.

ETL Library Name	Creation Time	Last Modified Time	Access Type	Packages	Jobs Attached	Created By
amorphicutils-v0-3-1-pyspark	6 hours ago	6 hours ago	owner	1	1	harsha
amorphicutils-v0-3-1-python	7 hours ago	7 hours ago	owner	1	1	harsha



3. Let's download the packages for both the libraries (pyspark and python).

ETL Library

Home > Transformation > ETL Library > Details

← amorphicutils-v0-3-1-pyspark ✓ Update Complete

ETL Library Id fe4b4d47... | Created By harsha 6 hours ago | Last Modified By harsha 6 hours ago

Description Amorphicutils v.0.3.1 for Pyspark | Docs: <https://www.docs.amorphicdata.io/utils-docs/category/version-030/>

amorphicutils-v0-3-1-pyspark

amorphicutils-v0-3-1-python

DETAILS RESOURCES

ETL Library Metadata

ETL Library Name amorphicutils-v0-3-1-pyspark ETL Library Status update_complete Keywords OH

Packages

Package Name	Actions
/libs/python/amorphicutils-0.3.1.zip	

Download Package

Once downloaded we can use them to create our own Shared ETL library.

4. Click on create ETL Library. Name it as **amorphicutils-v0-3-1-python**. Example: **amorphicutils-v0-3-1-python-adityabhat**.

Create ETL Library

Create ETL Library >

Create Package >

ETL Library Name * amorphicutils-v0-3-1-python-adityabhat

Description

Keywords Owner: harsha

Next

5. Click on next and then select the package from local to add the package you downloaded.

Note that in this example we are creating a python library hence uploading the **.whl** file (to be used in pythonshell job) Follow below screenshots.

Create ETL Library

Create ETL Library >

Create Package >

ETL Library amorphicutils-v0-3-1-python-adityabhat created. Select the files to be added to the library

Select & Upload

Add package(s) to ETL Library



Create ETL Library

Create ETL Library >

Create Package >

Select & Upload

ETL Library amorphicutils-v0.3-1-python-adityabhat1 created. Select the files to upload

Select files to upload

Please select upto 100 files (<5GB each) to upload

Create ETL Library

Create ETL Library >

Create Package >

Select files to upload

Favourites

Recent

Applications

- Desktop
- Documents
- Downloads

iCloud

- iCloud Drive
- Shared

Locations

- MySQL...
- Google Dri...

Tags

demo-nvd

Name	Size	Kind	Date Added
amorphicutils-0.3.1-py3-none-any.whl	53 KB	Document	Today at 9:04 PM
amorphicutils-0.3.1.zip	254 KB	ZIP archive	Today at 9:04 PM

Show Options

Cancel

Open

Please select upto 100 files (<5GB each) to upload

6. Click on Upload Selected Files.

Create ETL Library

Create ETL Library >

Create Package >

Select & Upload

ETL Library amorphicutils-v0.3-1-python-adityabhat1 created. Select the files to upload

Select files to upload

Upload Selected Files

amorphicutils-0.3.1-py3-none-any.whl

51.38 kB

Upload Selected Files



7. Proceed further by clicking on Add packages(s) to ETL Library.

The screenshot shows a user interface for creating an ETL library. On the left, there's a sidebar with 'Create ETL Library' and 'Create Package' options. The main area displays a message: 'ETL Library amorphicutils-v0-3-1-python-adityabhat1 created. Select the files to be added to the library'. Below this is a file list containing 'amorphicutils-0.3.1-py3-none-any.whl'. At the bottom are two buttons: 'Select & Upload' and 'Add package(s) to ETL Library'.

8. Once successful you are taken back to the main page and you can see the status of the library as Update Complete. (If you don't see it reload the page)

The screenshot shows the 'ETL Library' details page. The top navigation bar includes 'Create ETL Library'. The main content area shows an ETL library named 'amorphicutils-v0-3-1-python-adityabhat' with an 'Update Complete' status indicator. Below this, there are sections for 'DETAILS' and 'RESOURCES'. Under 'DETAILS', the ETL Library Name is listed as 'amorphicutils-v0-3-1-python-adityabhat'. Under 'RESOURCES', there's a table titled 'Packages' with one entry: 'Package Name' (amorphicutils-0.3.1-py3-none-any.whl) and 'Actions' (trash bin icon).

You have now successfully created the ETL Library.

Repeat the steps from 4 to 8 for creating the pyspark library. Use **amorphicutils-0.3.1.zip** file for upload.

ETL Job:

Let's create an ETL job to Read from Dataset, Apply simple Transformations and Write to Dataset.

Pre- Requisites:

Make sure you have access to the below Datasets.



- a. Immunization (Domain: bronzehealth, Read Access)
- b. Immunization Staged (Domain: silverhealth , Write Access)

Procedure:

1. Navigate to ETL Jobs from the left pane. Transformation > Jobs

The screenshot shows the Amorphic platform's user interface. On the left, there is a vertical navigation bar with icons and labels: Home, Ingestion, Catalog, Transformation (which is highlighted in blue), and Analytics. To the right of this is a main content area titled "Jobs". Below the title, there are two sub-options: "ETL Library" and "Parameters Store". The overall background is light gray.

2. You should already see jobs **sample_pyspark_job** and **sample_python_job** shared with you.

The screenshot shows the "Jobs" page within the Amorphic interface. At the top, there is a search bar and a message indicating "Showing 1 - 3 of 3 record(s)". Below this is a table with the following data:

Job Name	Description	Last Modified	Job Type	Job Bookmark	Options
ETI_Immunization_Cleanse ETL to Clean Immunization Data	ETL to Clean Immunization Data	7 days ago	pythonshell	Disabled	
sample_pyspark_job Sample script to read and write using amorphic utils - Pyspark	Sample script to read and write using amorphic utils - Pyspark	5 hours ago	spark	Disabled	
sample_python_job Sample script to read and write using amorphic utils - Pythonshell	Sample script to read and write using amorphic utils - Pythonshell	5 hours ago	pythonshell	Disabled	

3. Let's copy the script for our reference which we can use later in our lab.
Select the job **sample_python_job** and click on Edit Script.



+ New Job

← sample_python_job ⏱

ID c60e6458... | Created By harsha ⏱ 6 hours ago | Last Modified By harsha ⏱ 5 hours ago

Description Sample script to read and write using amorphic utils - Pythonshell

DETAILS EXECUTIONS SCHEDULES

Edit Job Script

4. Deselect the toggle for Read Mode On, Select all and copy it into your editor.

sample_python_job

Script Editor ⏱

Read Mode On

```
1 """
2 Description : Sample script to read and write using amorphic utils - Pythonshell
3
4 Date      : 12/Nov/2024
5 Author    : aditya.bhat@cloudwick.com
6 Version   : 1.0
7 Dependencies : amorphicutils
8 Glue Version : 3
9 Python Version : 3
10 Language : Python
11 ParamStore : SYSTEM.S3BUCKET.LZ , SYSTEM.S3BUCKET.LZ
12
13 Update History :
14 Date          Version      Author           Description
15 12/Nov/2024    1.0          aditya.bhat@cloudwick.com  First version
16
17 """
18
19
20 import time
21 import pandas as pd
22 import logging
23 from amorphicutils.common import read_param_store
24 from amorphicutils.python import write
25 from amorphicutils.python import read
26
27 logging.basicConfig(level="INFO")
28 LOGGER = logging.getLogger()
29 LOGGER.setLevel(logging.INFO)
30
31
32 def get_param_value(param_store_key, secure=False):
33     """Function to read parameters from parameters store"""
34     LOGGER.info(
35         "In get param value - Fetching values stored in %s - %s" % (secure, param_store_key))
```

5. You need to make a few updates to the script.

- Replace the user_id with your login user id. (Line no 62 from below SS)
- Replace the param_store_key with the one you created. (Line no 81 from below SS).
- You can replace your read/write domain and read/write dataset if you wish. Let's skip this for this demo.



```
55 def main():
56     """
57     Main
58     """
59
60     LOGGER.info("In Main,")
61
62     user_id = "adityabhat" # Your Login UserId with access to write
63
64     # Replace below if you have different Domain to Read from
65     r_domain = "bronzehealth"
66     # Replace this if you have different Domain to write into.
67     w_domain = "silverhealth"
68     # Replace this if you created new Dataset to Read from.
69     r_dataset = "Immunization"
70     # Replace this if you created new Dataset to write into.
71     w_dataset = "Immunization_Staged"
72
73     # LZ Bucket is Output Bucket
74     # lz_bucket = "nvd-us-west-2-816069158041-test-lz"
75     # DLZ Bucket is Input Bucket
76     # dlz_bucket = "nvd-us-west-2-816069158041-test-dlz"
77
78     lz_bucket = get_param_value(param_store_key="SYSTEM.S3BUCKET.LZ")
79     dlz_bucket = get_param_value(param_store_key="SYSTEM.S3BUCKET.DLZ")
80     my_param = get_param_value(
81         param_store_key="nvd-sample-param-adityabhat", secure=True
82     ) # Replace with your paramstore name
83     print(my_param)
84
```

Fig: Sample Snippet of the ETL Code.

6. Back to Amorphic UI. Clone the **sample_python_job** by clicking on three dots on the right corner of the page and clone.

The screenshot shows the Amorphic UI Jobs page. On the left, there's a sidebar with icons for Home, Transformation, and Jobs. Under 'Jobs', there are three items: 'ETL_Immunization_Cleanse', 'sample_pyspark_job', and 'sample_python_job'. The 'sample_python_job' item is selected. The main panel shows its details: ID (c60e645b...), Created By (harsha) 5 hours ago, Last Modified By (harsha) 4 hours ago, and a Description: 'Sample script to read and write using amorphic utils - Pythonshell'. Below this are tabs for DETAILS, EXECUTIONS, and SCHEDULES. A context menu is open on the right, with 'Clone Job' highlighted. Other options in the menu include Edit Job, Notification Settings, Delete Job, Update Extra Resources, Manage External Libraries, and Cost Tags.

7. Update the **Job Name** with **sample_python_job_<yourusername>**. Example **sample_python_job_adityabhat**

Remove the existing shared library and update it with the one you created for python.

Add the new paramstore key you created to the Job.

Refer below screenshot for more info.



Clone Job

Upload Data >

Metadata > * Required

Job Name (1) * sample_python_job_adityabhat

Description (1) Sample script to read and write using amorphic u

Job Type (1) * Python Shell

Advanced Parameters >

Cost Tags >

Review & Submit >

Network Configuration (1) * Public

Bookmark (1) * Disable

Keywords Owner: harsha

Resource Access Configuration

Parameters Store Access (1) nvd-sample-parm-adityabhat SYSTEM.S3BUCKET.DLZ

Datasets Write Access (1) silverhealth : Immunization_Staged

Datasets Read Access (1) bronzehealth : Immunization

Shared Libraries (1) amorphicutils-v0-3-1-python-adityabhat

Domains Write Access (1)

Domains Read Access (1)

Next

8. Skip the Advanced Configuration and Cost Tags. Review the Job and click on create.

Clone Job

Upload Data >

Metadata >

Advanced Parameters >

Cost Tags >

Review & Submit >

Network Configuration	general-public-network
Job Bookmark Option	disable
Keywords	Owner: harsha
Max Concurrent Runs	4
Max Retries	-
Notify Delay After	-
Timeout	20
Is Data Lineage Enabled	no
Is Auto Scaling Enabled	false
Parameter Access	SYSTEM.S3BUCKET.DLZ, SYSTEM.S3BUCKET.LZ, nvd-sample-parm-adityabhat
Shared Libraries	amorphicutils-v0-3-1-python-adityabhat
Cost Tags	

JSON Payload

Create

9. You will be taken to the script editor. Deselect the Read Mode On toggle and update your script with the one you made changes previously. Once done click on **Save & Exit**



sample_python_job_adityabhat

Script Editor ⓘ

Read Mode On

```
1 import sys
2 from awsglue.transforms import *
3 from awsglue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from awsglue.context import GlueContext
6 from awsglue.job import Job
7
8 ## @params: [JOB_NAME]
9 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
10 sc = SparkContext()
11 glueContext = GlueContext(sc)
12 spark = glueContext.spark_session
13 job = Job(glueContext)
14 job.init(args['JOB_NAME'], args)
15 job.commit()
```

Save & Publish

10. You have now successfully created the pythonshell Job.

[← sample_python_job_adityabhat](#) 

Id 7bb18e73... | **Created By** adityabhat  a few seconds ago | **Last Modified By** adityabhat  a few seconds ago

Description Sample script to read and write using amorphic utils - Pythonshell

DETAILS **EXECUTIONS** **SCHEDULES**

Basic Metadata 

Job Name	Job Type	Registration Status
sample_python_job_adityabhat	Pythonshell	completed

Keywords 

Advanced Parameters Metadata 

Resource Access Metadata 

Extra Resources 

11. Let's run the ETL. Click on Run button on the right corner of ETL page for your selected Job.

[← sample_python_job_adityabhat](#)

[Id](#) 7bb18e73... | [Created By](#) adityabhat a few seconds ago | [Last Modified By](#) adityabhat a few seconds ago

Description Sample script to read and write using amorphic utils - Pythonshell

[DETAILS](#) [EXECUTIONS](#) [SCHEDULES](#)



You will be taken to the **Execute Job** page. Click on Continue and then Click Run Job.

Execute Job



Upload Data >	Old Job RunId ⓘ	* Required
Execute Job >	Job Type ⓘ	pythonshell
Review & Submit >	Max Capacity ⓘ	1
	Job Bookmark ⓘ	disable
	Worker Type ⓘ	▼
	Notify Delay After ⓘ	
	Job Parameters ⓘ	
	Add Parameter	
		Continue



Execute Job



Upload Data >

Execute Job >

Review & Submit >

Job Details

Old Job Run Id -

Max Capacity 1

Timeout 20

Notify Delay After -

Glue Version 3.0

Job Bookmark Option disable

Max Retries -

Network Configuration general-public-network

Python Version 3.9

Is Auto Scaling Enabled false

JSON Payload

Run Job

12. You can track your execution and preview logs.

← sample_python_job_adityabhat

Id 7bb18e73... | Created By adityabhat a few seconds ago | Last Modified By adityabhat a few seconds ago

Description Sample script to read and write using amorphic utils - Pythonshell

DETAILS EXECUTIONS SCHEDULES

Search in this page

Showing 1 - 1 of 1 record(s)

Started On	Completed On	Trigger Source	Error	Log Status	Options
a few seconds ago	-	Self	N/A	Trigger logs download to view.	

Showing 1 - 1 of 1 record(s)



13. Once the Job is successfully complete you can navigate to the Dataset you are writing to from the Details page by expanding Resource Access Metadata and clicking on the Dataset which will open in a new tab.

The screenshot shows the 'Basic Metadata' section with 'Job Name: sample_python_job_adityabhat', 'Job Type: Pythonshell', and 'Registration Status: completed'. Below it is the 'Advanced Parameters Metadata' section. The 'Resource Access Metadata' section is expanded, showing 'Domain Access' with 'Write Access: 00' and 'Read Access: 00'. The 'Dataset Access' section is also expanded, showing 'Write Access: 01' and 'Read Access: 01'. A specific dataset entry, 'silverhealth:Immunization_Staged', is highlighted with a red box. At the bottom right of the page is a blue circular icon with a white gear symbol.

14. Now navigate to the Files tab of the Dataset and search for your login user id to see if the file was successfully loaded.

The screenshot shows the 'FILES' tab of the dataset 'Immunization_Staged'. A search bar at the top contains 'adityabhat'. Below it, a table lists one record: '...abhat_1731422104.csv' with 'Last Modified: adityabhat 4 minutes ago' and 'Message: Data load completed.'. The table has columns: File Name, Last Modified, Message, Tags, and Options. A red box highlights the search bar and the first table row.

Congrats! You were successfully able Read and Write Data using ETL Job.
Repeat the same steps as above to create and execute PySpark Job by using **sample_pyspark_job** as a reference.



LAB Session 4:

Workflows:

Lets create a workflow by cloning the existing one.

Purpose:

Setup a workflow to trigger an ETL, configure success and failure nodes to receive email communication

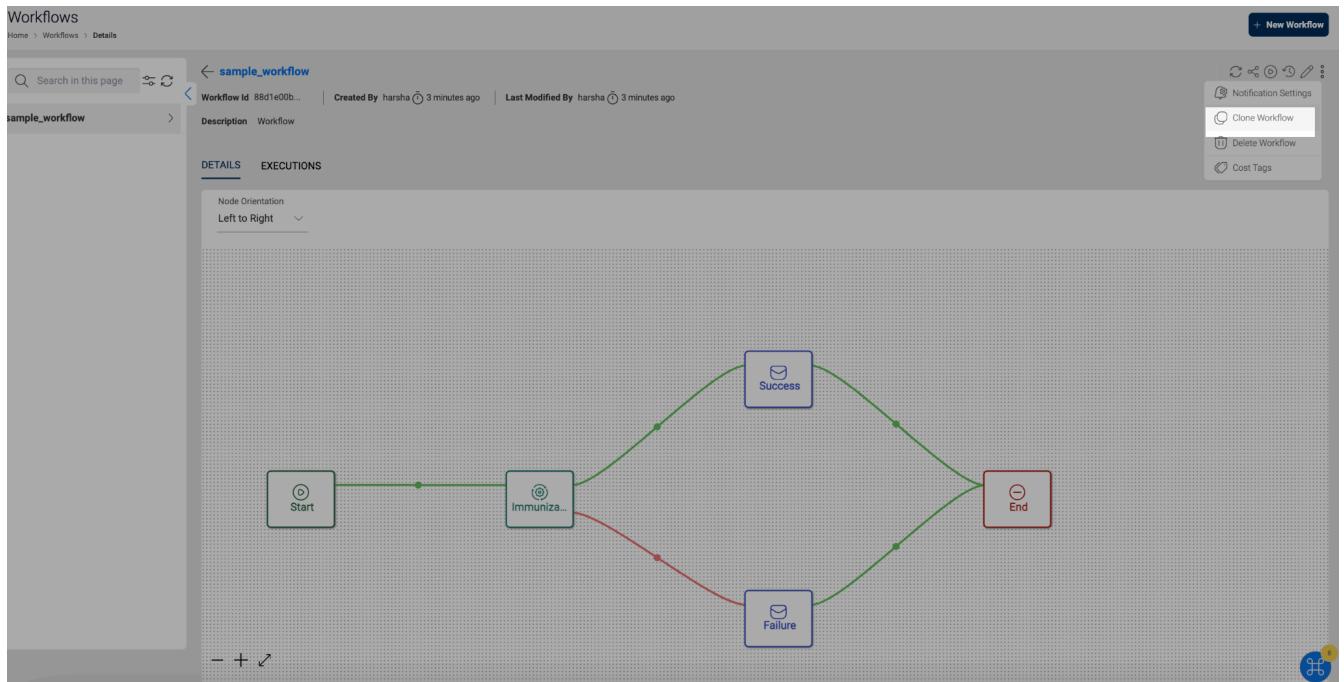
Procedure:

1. Navigate to Workflows from the side pane. You should already see one workflow that has been shared to you.

The screenshot shows the Amorphic Workflows interface. At the top, there is a navigation bar with icons for Home, Workflows, and a New Workflow button. Below the navigation bar, there is a search bar labeled "Search in this page". The main area displays a table with one record. The table has columns for "Workflow Name", "Description", "Last Modified Time", and "Options". The single record shown is "sample_workflow" (Workflow), last modified "a minute ago", and has three options: a copy icon, a edit icon, and a delete icon. At the bottom of the table, it says "Showing 1 - 1 of 1 record(s)".

Workflow Name	Description	Last Modified Time	Options
sample_workflow Workflow	Workflow	a minute ago	

2. Click on it to see the details and click on three dots in the right corner to see the clone option and hit clone.



3. Rename the workflow as `sample_workflow_<yourname>` Ex: `sample_workflow_adityabhat`
Under Default Execution Properties you will see a few key value pairs, these can be used in your workflow nodes.

In our workflow we will be using it in a [Email Node](#)

Update the '**to**' key with the list of **values** of the email address of people who you want to send email to. For this demo use your own mail_id that you used to register on the platform.
Click on **Next** to continue.

Workflow Name: sample_workflow_adityabhat

Description: Workflow

Keywords: Owner: harsha

Default Execution Properties:

- Key: s_body Value: Workflow Execution Successfully Completed. Please check the target datasets.
- Key: sub Value: Health | Nevada | Immunization | Workflow Status
- Key: to Value: [aditya.bhat@cloudwick.com]
- Key: f_body Value: Workflow Execution Failed. Please check workflow execution logs in Amorphic for n

Add Default Execution Properties

Continue



4. Skip Cost Tags and Continue.

< Clone Workflow

Home > Workflows > Clone Workflow

Select Cost Tags ⓘ

You don't have enough permissions to use cost tags! Please contact your administrator.

Go Back Continue

5. You can now design your workflow. Hover on any node that you want to edit and click on the pencil icon to edit/view the details of the node.

< Clone Workflow

Home > Workflows > Clone Workflow

Design your Workflow

+ Add a node

```
graph LR; Start((Start)) --> ImmunizationET[ImmunizationET_L_Clean]; ImmunizationET --> Success[Success]; ImmunizationET --> Failure[Failure]; Success --> End((End)); Failure --> End;
```

Go Back Continue

6. Let's first edit the ETL node. Change the ETL Job to the one you created in session 3. Update the node name if required and click on **Update Node**



Configure ETL Job



ETL Job instance ⓘ *

sample_python_job_adityabhat



Node name ⓘ *

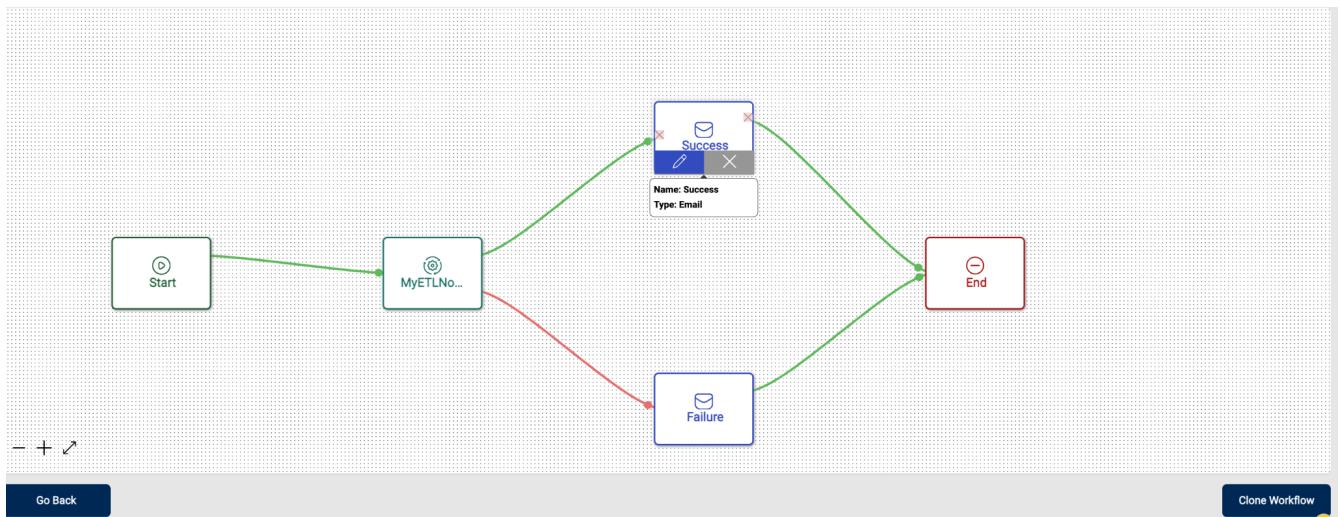
MyETLNode

Arguments ⓘ

Add Argument

Update Node

7. Lets inspect what's inside an Email Node. Click on Edit on any one of the email node (Success/Failure)



Notice that the arguments you passed earlier are now visible in the drop down. Leave the Configure Email page by clicking anywhere outside.



Configure Email



Node name ⓘ *

Success

Email recipient ⓘ *

to



s_body

sub

to

f_body

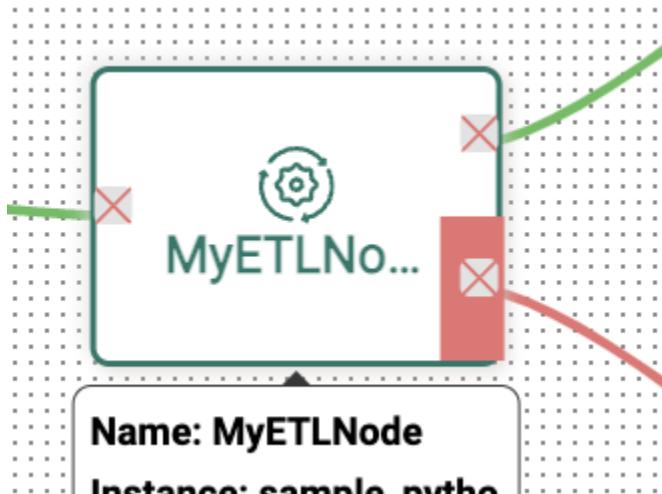
.....

Arguments ⓘ

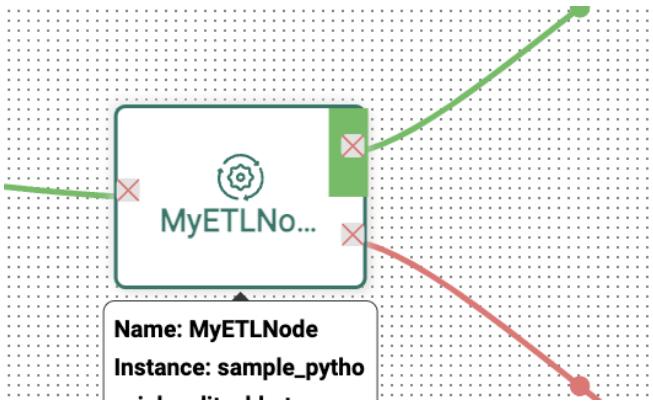
Add Argument

Update Node

8. Notice the red bar in the node, you can connect it to any other node which would then execute on failure of the previous node. (which in this case is a failure node)



9. Notice the green bar on the node, you can connect it to any other node which would then execute on the success of the previous node. (which in this case is a success node)



10. You can also add any of the available nodes by clicking on Add Node. You can get to know more about each node in the [documentation](#)

Select Node Type

ML Model	ETL Job	Email	Textract
Translate	Comprehend	Medical Comprehend	Transcribe
Medical Transcribe	Rekognition	Workflow	File Load Validation
Sync To S3	Connection		

Continue

11. Let's now simply clone the workflow.

You have successfully created a workflow.



Executing a workflow:

1. Select the workflow you want to run and click on the run workflow button on the left. Confirm if you see a popup.

The screenshot shows the 'Workflows' section of the Amorphic interface. A specific workflow named 'sample_workflow_adityabhat' is selected. At the top right, there is a 'Run Workflow' button. Below it, the workflow details are shown: Workflow Id (b4117cd9...), Created By (adityabhat), Last Modified By (adityabhat), and a description of 'Workflow'. There are tabs for 'DETAILS' and 'EXECUTIONS', with 'DETAILS' currently selected. A dropdown for 'Node Orientation' is set to 'Left to Right'. On the far right, there are several small icons for managing the workflow.

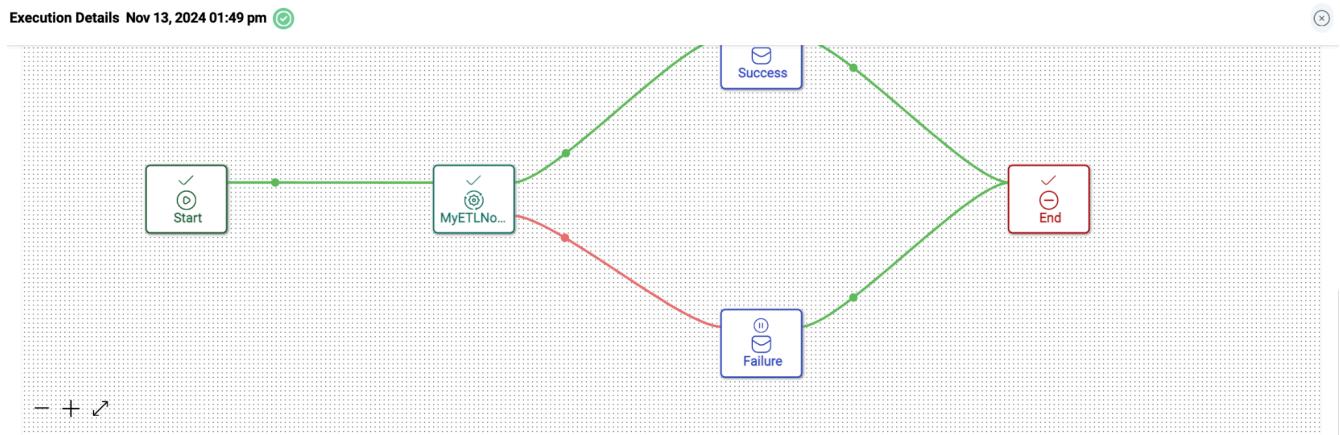
2. Go to the Execution tab to see the progress.

The screenshot shows the 'EXECUTIONS' tab for the same workflow. It displays a single execution record. The table has columns: Started On, Completed On, Node Count, Failed Nodes, and Options. The data shows: Started On (Nov 13, 2024 01:49 pm), Completed On (-), Node Count (5), Failed Nodes (0), and Options (a small icon). Below the table, it says 'Showing 1 - 1 of 1 record(s)'. At the bottom right, there is a 'Execution Details' button.

Started On	Completed On	Node Count	Failed Nodes	Options
Nov 13, 2024 01:49 pm	-	5	0	



3. You can click on the Options to see more info of progress of each node.



4. You can also see the logs of your ETL run straight from this Execution Details page.

The diagram and table are identical to the previous section, showing a successful workflow execution. The 'Failure' node has an 'Error Logs' button next to its 'More Details' button.

Node Name	Execution Details	Execution Time (sec)	Attempts	Options
Start	SUCCEEDED	-	-	
Success	SUCCEEDED	15	-	
MyETLNode	SUCCEEDED	33	-	
Failure	NOT_STARTED	N/A	-	Error Logs
End	SUCCEEDED	-	-	

5. Back at the Execution tab, you can see the workflow has now completed successfully. The summary of success and failed nodes is also displayed. In this case all the five are completed with success status.



← sample_workflow_adityabhat

Workflow Id b4117cd9... | Created By adityabhat (a few seconds ago) | Last Modified By adityabhat (a few seconds ago)

Description Workflow

DETAILS EXECUTIONS

Search in this page

Showing 1 - 1 of 1 record(s)

Started On	Completed On	Node Count	Failed Nodes	Options
Nov 13, 2024 01:49 pm	Nov 13, 2024 01:51 pm	5	0	⋮

Showing 1 - 1 of 1 record(s)

6. Let's check if we received mail on the success of this workflow.

Health | Nevada | Immunization | Workflow Status

Amorphic Data Services noreply@cloudwick.com via amazoneses.com
to me ▾

1:52PM (5 minutes ago) ⌂ ⌂ ⌂ ⌂ ⌂

Be careful with this message.
The sender hasn't authenticated this message so Gmail can't verify that it actually came from them. Avoid clicking links, downloading attachments, or replying with personal information.

Looks safe ⓘ

Health | Nevada | Immunization | Workflow Status

EventType	Generic Email
Message	Workflow Execution Successfully Completed. Please check the target datasets
AdditionalDetails	{}

Congrats you have now successfully created and executed a workflow.



Notebook Lifecycle Configurations:

A lifecycle configuration provides shell scripts that run only when you create the notebook instance or whenever you start one

Procedure:

1. Navigate to Analytics > Lifecycle Configurations.
2. You will already see list of system created Lifecycle Configs.

Notebooks Lifecycle Configurations

Home > Analytics > Notebooks Lifecycle Configurations

+ Create Lifecycle Configuration

Showing 1 - 4 of 4 record(s)

Lifecycle Name	Last Modified Time	Created By	Options
auto-stop-idle-notebook This script stops a SageMaker notebook once it's idle for more than 1 hour (default time). Enabling internet access is mandatory for this lif...	10 days ago	System	
disable-uninstall-ssm-agent This script disables and uninstalls the SSM agent which is present by default in Notebook Instances	10 days ago	System	
enable-code-whisperer This script enables the code whisperer extension in Jupyter lab which can be used for generating code recommendations	10 days ago	System	
enable-glue-sessions This lifecycle configuration can be attached to notebooks which have glue-sessions enabled. Enabling internet access is mandatory for thi...	10 days ago	System	

Showing 1 - 4 of 4 record(s)

3. Lets create a create custom LC that combines enabling glue session and auto stopping the idle instance. Clone the [enable-glue-sessions](#)

enable-glue-sessions
This lifecycle configuration can be attached to notebooks which have glue-sessions enabled. Enabling internet access is mandatory for thi...

Showing 1 - 4 of 4 record(s)

4. Rename it to custom-enable-glue-sessions-<yourusername> Ex: custom-enable-glue-sessions-harsha and click continue.

Clone Lifecycle Configuration

Upload Data >

Metadata > * Required

On-Start Script >

On-Create Script >

Review & Submit >

Lifecycle Configuration Name ⓘ *

custom-enable-glue-sessions-harsha

Description ⓘ

This lifecycle configuration can be attached to notebooks which have glue-

Keywords



5. Replace the On-Start Script from the script given below.

<https://github.com/cwkadityabhat/nevada-public-datalake/blob/main/Lab4/custom-enable-glue-sessions-lc.sh>

Clone Lifecycle Configuration

Upload Data >

Metadata >

On-Start Script > **Selected**

On-Create Script >

Review & Submit >

Script will be converted to Base64

```
1 #!/bin/bash
2 set -ex
3 [ -e /home/ec2-user/glue_ready ] && exit 0
4 sudo -u ec2-user -i <<'EOF'
5
6 ANACONDA_DIR=/home/ec2-user/anaconda3
7
8 # Create and Activate Conda Env
9 echo "INFO: Creating glue_pyspark conda environment"
10 conda create --name glue_pyspark ipykernel jupyterlab pandas=1.5.3 -y
11
12 echo "INFO: Activating glue_pyspark"
13 source activate glue_pyspark
14
15 # Initialize Latest Python Path
16 echo "INFO: Getting site packages directory of latest python version"
17 SITE_PACKAGES_DIR=$(python -c 'import site; print(site.getsitepackages()[0])')
18 echo "INFO: Site package directory is --- $SITE_PACKAGES_DIR"
19
20 # Install Glue Sessions to Env
21 echo "INFO: Installing AWS Glue Sessions with pip"
22 pip install aws-glue-sessions==1.0.0
23
```

Continue

6. Leave the On-Create Script unchanged and Proceed to Review. Clone the LifeCycle Configuration

Clone Lifecycle Configuration

Upload Data >

Metadata >

On-Start Script >

On-Create Script >

Review & Submit > **Selected**

Lifecycle Configuration Details

Lifecycle Name	custom-enable-glue-sessions-harsha
Description	This lifecycle configuration can be attached to notebooks which have glue-sessions enabled. Enabling internet access is mandatory for this lifecycle configuration to work
Keywords	-

JSON Payload

Clone Lifecycle Configuration



7. You will be redirected to the main page upon successful creation.

Lifecycle Configuration 41358962... | Created By harsha a few seconds ago | Last Modified By harsha a few seconds ago

Description This lifecycle configuration can be attached to notebooks which have glue-sessions enabled. Enabling internet access is mandatory for this lifecycle configuration to work

DETAILS RESOURCES

Metadata

Lifecycle Configuration Name: custom-enable-glue-s... | Keywords: -

On-Start Script | Show On-Start Script

Show On-Create Script

You have successfully created a custom life cycle configuration. We will use this to attach it to our Notebook in next part.

Code Repository:

The code repository has already been created for you **sample-public-repo**. You will be using this to attach it to a notebook.

Only a unique code repository can exist in amorphic with a particular repository url. Hence you do not need to create any repo for this demo

Notebooks:

Procedure:

1. Navigate to Analytics > Notebooks from the left pane. A sample notebook is shared with you [sample-notebook](#)
2. Clone the notebook from notebooks page under options of the notebook that you want to clone.

Notebook Name	Description	Instance Type	Volume Size In Gb	Sessions Enabled	Options
sample-notebook Sample Notebook	Sample Notebook	ml.t2.medium	10	Enabled	

3. Rename the notebook from sample-notebook to sample-notebook-<username>. Ex: sample-notebook-adityabhat. Optionally you can update the keywords. Click on **continue**.



Clone Notebook

Upload Data >

MetaData > * Required

Instance MetaData >

Code Repository >

Cost Tags >

Review & Submit >

Notebook Name ⓘ * sample-notebook-adityabhat

Description ⓘ Sample Notebook

Keywords
Owner: adityabhat

Continue

4. You might have to update below fields.

Dataset Read Access: [Immunization](#) (Domain: bronzehealth)

Dataset Write Access: [Immunization_Staged](#) (Domain: silverhealth)

Parameter Store: The paramstore value that you created in session 3.

(nvd-sample-parm-adityabhat)

Shared Libraries: The pyspark library that you created in session 3. Use [amorphicutils-v0-3-1-pyspark](#) if you havent.

Lifecycle Configuration: enable-glue-sessions

Auto Stop Time: Select this time that you want instance to be stopped to stop incurring additional cost.

Clone Notebook

Upload Data >

MetaData > * Required

Instance MetaData >

Code Repository >

Cost Tags >

Review & Submit >

Instance Type ⓘ * ml.t2.medium USD 0.0464/hour

Volume Size(n GB) ⓘ * 10

Sessions ⓘ * Enabled

Parameters Store Access ⓘ SYSTEM.S3BUCKET.LZ SYSTEM.S3BUCKET.DLZ nvd-sample-parm-harsha SYSTEM.S3BUCKET.ATHENA

Datasets Write Access ⓘ stage

DOMAIN: SILVERHEALTH silverhealth: Immunization_Staged

Shared Libraries ⓘ amorphicutils-v0-3-1-pyspark-adityabhat

Datasets Read Access ⓘ bronzehealth: Immunization

Domains Read Access ⓘ

Lifecycle Configuration ⓘ enable-glue-sessions

Auto Stop ⓘ * Yes

View Access ⓘ

Auto Stop Time ⓘ * 2024-11-14 02:00:00

Continue



Click on continue after updating all of the above mentioned fields.

5. You can setup code-repository to work/ test the code.

Under Default Code Repository choose from drop down sample-public-repo

Repor Used: <https://github.com/cwkadityabhat/nevada-public-datalake.git>

You can click on the above link to check the public repo hosted in GitHub.

Edit Notebook

MetaData >

Instance MetaData >

Code Repository >

Review & Submit >

Switch to either select an existing code repository or enter a repository link

Default Code Repository –

Select an Existing Code Repository ⓘ

sample-public-repo

Additional Code Repositories –

Continue

Click on **Continue**.

6. Skip Cost tags. Review the configurations and hit on Clone Notebook at the bottom

Clone Notebook

Upload Data >

MetaData >

Instance MetaData >

Code Repository >

Cost Tags >

Review & Submit >

Direct Internet Access	Enabled
Owner	-
Read Only	-
GlueSessionsEnabled	true
Owner Dataset Access	Immunization_Staged
ReadOnly Dataset Access	Immunization
Parameter Access	SYSTEM.S3BUCKET.LZ, SYSTEM.S3BUCKET.DLZ, SYSTEM.S3BUCKET.ATHENA, nvd-sample-parm-adityabhat
Shared Libraries	amorphicutils-v0-3-1-pyspark-adityabhat
View Access	-
Default Code Repository	-
Additional Code Repositories	https://github.com/cwkadityabhat/nevada-public-datalake.git
Cost Tags	-
JSON Payload	

Clone Notebook



7. A notebook is created for you. The status of the notebook will be in **pending** Status for approx 10-15 mins. Once the notebook instance is up you will see the Status **In-Service**

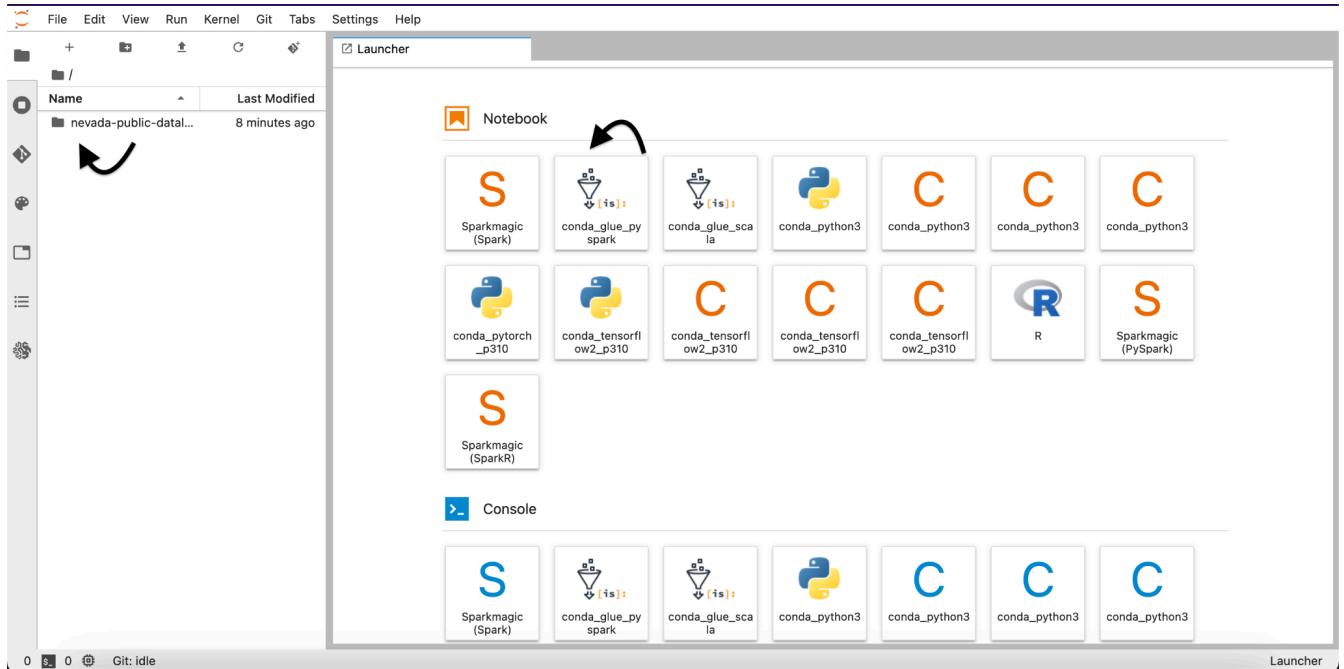
The screenshot shows the Amorphic interface for managing notebooks. On the left, there's a sidebar with a search bar and a tree view showing 'sample-notebook-adityabhat' and 'sample-notebook'. The main content area is titled 'sample-notebook-adityabhat'. It displays the following details:

- Metadata:**
 - Notebook Name: sample-notebook-adit...
 - Instance Type: ml.t2.medium
 - Notebook Status: InService
 - Volume Size(in GB): 10 GB
 - Estimated Cost: \$0.00
 - Sessions: Enabled
- Keywords:** OA
- Actions:** Go to Notebook, Go to Notebook Lab
- Network Options:** Network Configuration: App-Private, Internet Access: Enabled
- Additional Metadata:** (Collapsible section)
- Linked Resources:** (Collapsible section)
- Extra Resources:** (Collapsible section)
- Code Repositories:**
 - Additional Code Repositories: 01

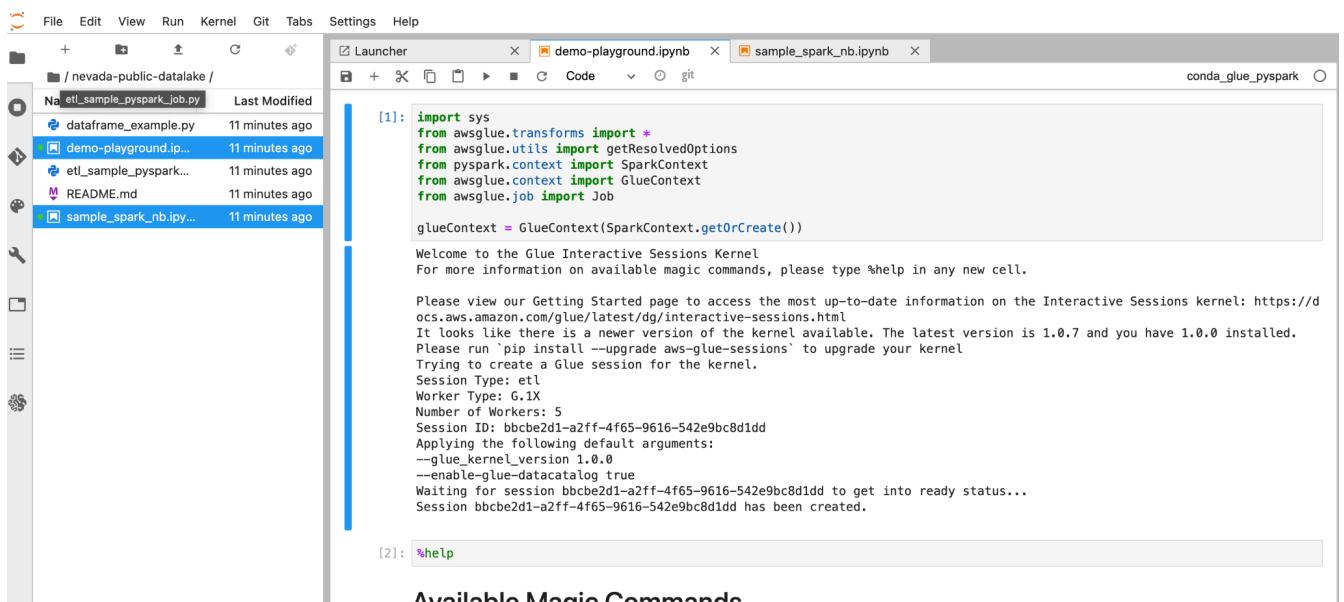
You have successfully created a Notebook.

Using a Notebook:

1. From the notebook you have created click on Go to Notebook Lab. This will take you to the lab where you can create a notebook, use a variety of consoles also notice that the git repository has been cloned for you.



2. Navigate inside the repo and you can make use of the notebook that's already been created for you.



demo-playground.ipynb: Use this to play around

sample_spark_nb.ipynb: This is the same code you ran on ETL yesterday using amorphicutils.

3. Add the path for pyspark library to be used while using **sample_spark_nb.ipynb**



Description :

Sample Notebook to read and write using amorphic utils -Pyspark

Load the pysaprk amorphic utils you created

copy the path from "Home -> Transformation -> ETL Library -> Details -> (hover over package path and click copy - use the version you need)"

You can navigate to Home -> Transformation -> ETL Library -> Details -> (hover over package path and click copy - use the version you need)

Note: Your session will fail if you dont copy the correct path/ have missed attaching your lib to instance.

4. Use Shift + Enter to run each cell.

5. Stop the notebook after your use.

Congrats you have successfully learnt to Create, Use and Stop the Notebook.



LAB Session 5:

Amorphic BI:

Creating a Quicksight Dashboard:

1. Navigate to Apps > Click on BI

The screenshot shows the Amorphic application interface. On the left, there is a sidebar with various icons and labels: Home, Ingestion, Catalog, Transformation, Analytics, Workflows, Schedules, and Apps. The 'Apps' icon is highlighted with a blue border. To the right of the sidebar, there is a main panel titled 'List Apps'. Inside this panel, there is a single item labeled 'BI' with a small icon next to it.

You will be redirected to Quicksight Dashboard

2. Click on Dataset on the left pane and then click on 'New Dataset' to add new Dataset from source.

The screenshot shows the QuickSight interface. On the left, there is a sidebar with the following options: Favorites, Recent, My folders, Shared folders, Dashboards (which is selected and highlighted in light blue), Data stories, Analyses, Datasets (which is also selected and highlighted in light blue), Topics, and Community. An arrow points from the 'Datasets' label in the sidebar to the 'Datasets' label in the main content area. The main content area is titled 'Datasets' and contains a message: 'No datasets' and 'Import or create a new dataset to start an analysis.' A blue button labeled 'New dataset' with a white arrow is located in the top right corner of this area.



3. Scroll to the bottom and use the Existing Data Source created for your user_id

The screenshot shows the QuickSight Datasets page. At the top, there's a header bar with the QuickSight logo and a message about SPICE capacity. Below the header is a grid of 20 data source icons, each with its name and a small description. The sources include RDS, Redshift (Auto-discovered and Manual connect), MySQL, PostgreSQL, ORACLE, SQL Server, Aurora, MariaDB, Presto, Spark, Teradata (Provided by Teradata), Snowflake, Google BigQuery, AWS IoT Analytics, Amazon OpenSearch Service (Successor to Amazon Elasticsearch Service), Timestream, Exasol, Databricks, Trino, Starburst, GitHub, Jira, ServiceNow, and Adobe Analytics. Below the grid, a section titled "FROM EXISTING DATA SOURCES" contains a single item: "harsha data source" with a timestamp "Updated 3 hours ago".

4. Click on Create Dataset

The screenshot shows a modal dialog titled "harsha data source". It has a close button ("X") in the top right corner. Inside the dialog, there is a section labeled "Database name ATHENA". At the bottom right of the dialog is a blue button labeled "Create dataset".



5. Choose the table to use for analysis. Lets use [goldhealth.immunization_standard](#) for this demo.

Choose your table ×

harsha data source

Catalog: contain sets of databases.
AwsDataCatalog ▼

Database: contain sets of tables.
goldhealth ▼

Tables: contain the data you can visualize.

immunization_standard
 immunization_standard_harsha
 vitals_standard

Edit/Preview data Use custom SQL Select

6. Click on Visualise

Finish dataset creation ×

Table: immunization_standard
Data source: harsha data source
Schema: goldhealth

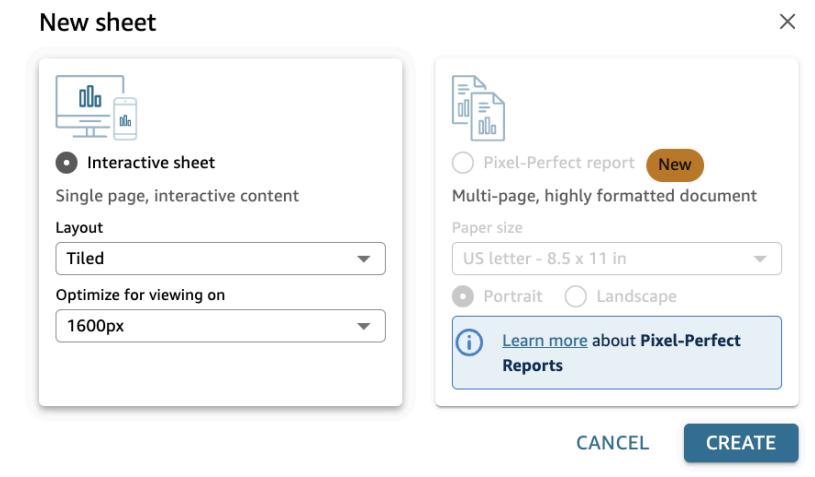
Import to SPICE for quicker analytics ✓ 30GB available SPICE
 Directly query your data

Email owners when a refresh fails

Edit/Preview data Augment with SageMaker Visualize



7. Click on Create to create a new sheet for analysis.



8. Let's create the first visualization.

A. **Vaccination Distribution:** To analyze the number of different vaccinations administered to citizens.

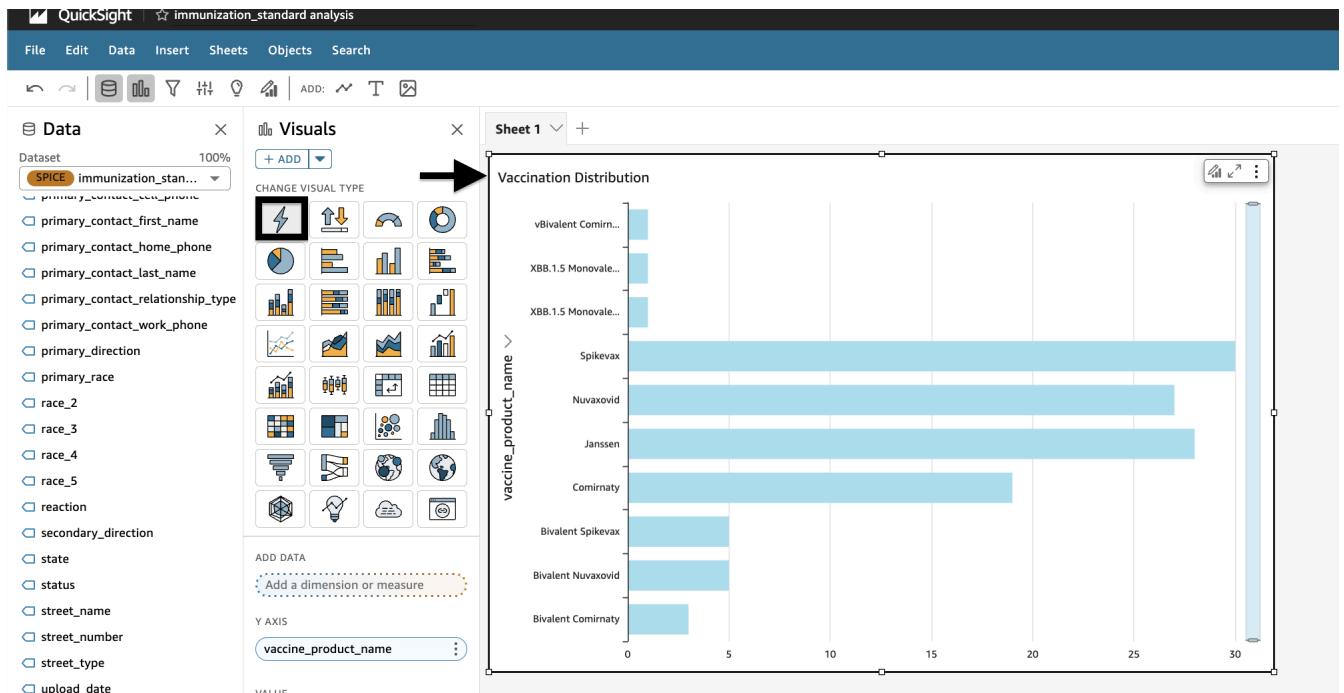
Drag the vaccine_product_name to build the visual graph on the sheet.

The screenshot shows the QuickSight interface with the following components:

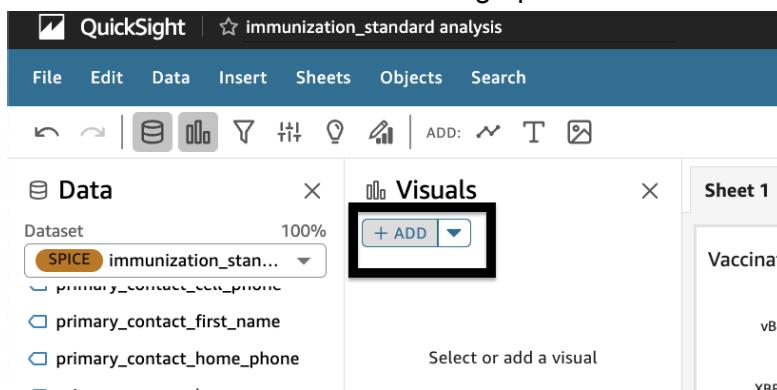
- Data Panel:** On the left, it lists the dataset "SPICE_immunization_stan..." with various fields: primary_contact_email, primary_contact_first_name, primary_contact_home_phone, primary_contact_last_name, primary_contact_relationship_type, primary_contact_work_phone, primary_direction, primary_race, race_2, race_3, race_4, race_5, reaction, secondary_direction, state, status, street_name, street_number, street_type, upload_date, vaccination_cvx_code, vaccination_date, vaccine_manufacturer, vaccine_product_name, and vaccine_type.
- Visuals Panel:** In the center, it shows a grid of visualization icons. A red arrow points from the "vaccine_product_name" field in the Data panel to the "AutoGraph" icon in the Visuals panel.
- Sheet 1:** On the right, a large empty area is labeled "Sheet 1". It has a title "AutoGraph" and a placeholder text "Add 1 or more fields to build a visual."

The AutoGraph will auto select the type of graph for you based on the dimension.

Update the title by double clicking on the title: Give it an appropriate name 'Vaccination Distribution' and Save.

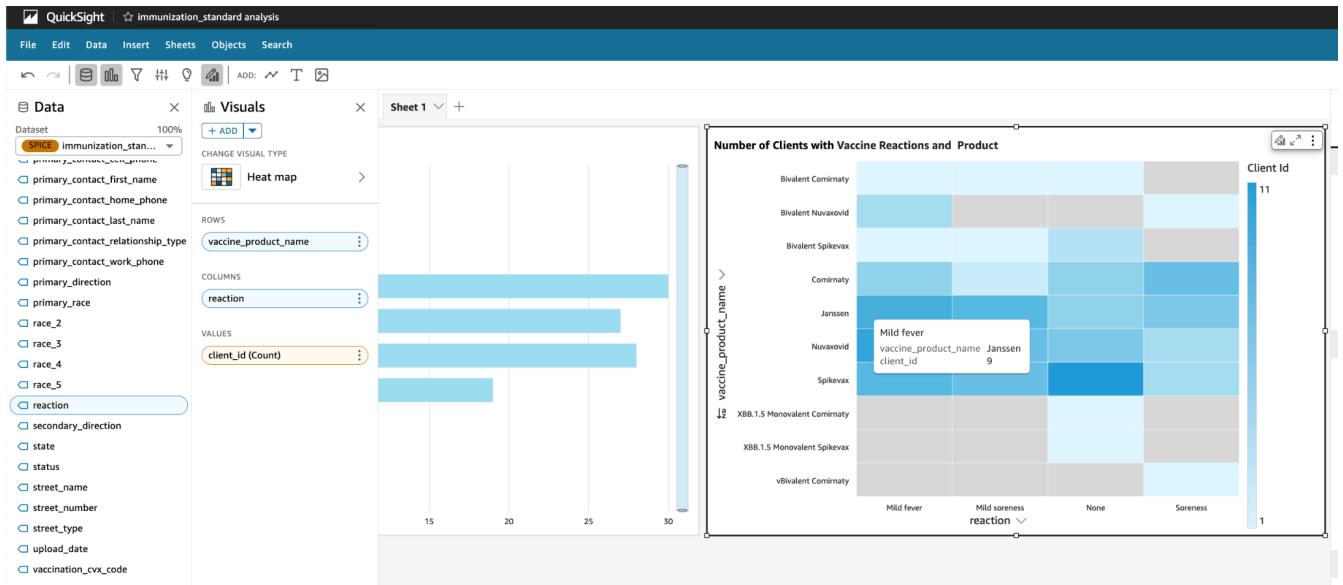


Click on Add Visuals to add another graph



B. Number of Clients with Vaccine Reactions and Product: Count of clients who had a reaction categorized for each product.

Let's create a heat map. Fill the row, columns and values as vaccine_product_name,reaction and client_id respectively.(See below SS for more info.) Update the title.



Click on Publish Dashboard to create a dashboard with the visuals you created.



Give an appropriate name and click on the publish dashboard.

Publish a dashboard

Publish new dashboard as

Vaccination Analysis

Replace an existing dashboard

ALL SHEETS SELECTED ▾

Data story

Allow sharing data stories ⓘ

Generative capabilities

Allow executive summary ⓘ

⚠ No topic linked yet. [Link topic](#)

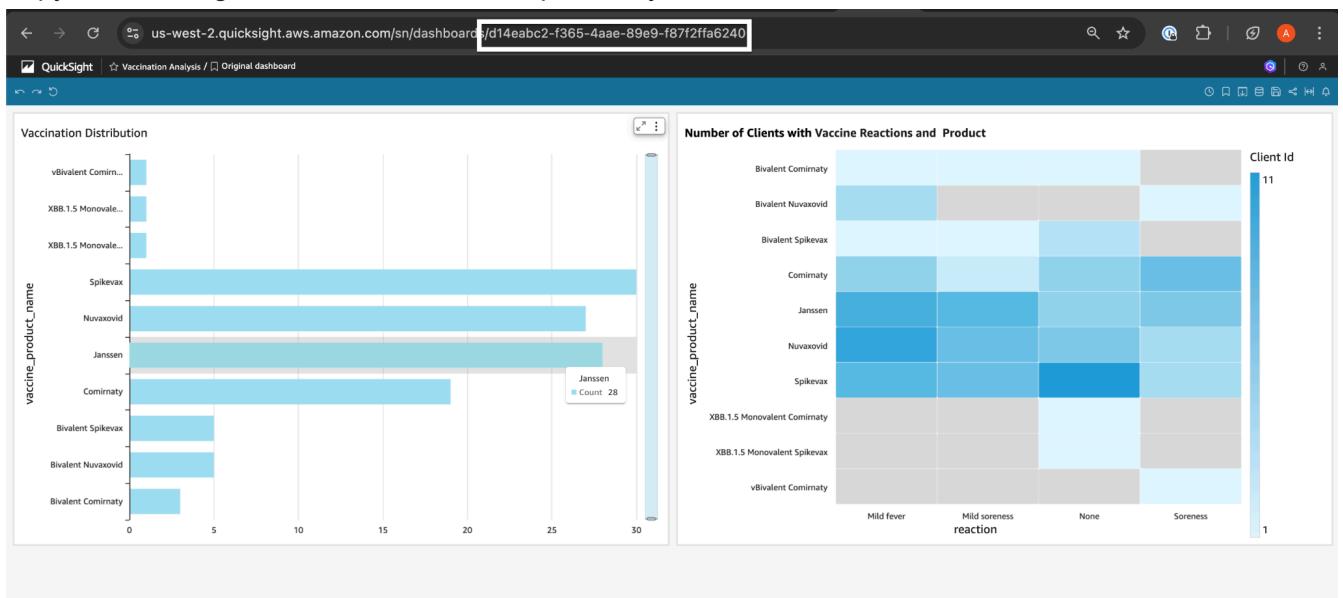
Advanced publish options ▾

Publish dashboard

Your dashboard is now created.



9. Copy the Quicksight Dashboard ID and keep it handy.



Sync Dashboard to Amorphic.

1. Back at the Amorphic UI. Navigate to Apps > List Apps. Click on the BI application from the list.

The screenshot shows the Amorphic 'List Apps' page with the following details:

App Name	App Type	Last Modified Time	Options
BI	BI	7 hours ago	(Edit)

2. Click on Sync Dashboards to sync the dashboard to Amorphic.

You will see a notification that says "Successfully triggered dashboard sync process"

The screenshot shows the Amorphic 'nvd-bi' app details page with the following details:

App Metadata

App Name	App Type	App Url
NVD-BI	BI	Go To BI App

Sync Dashboards button is highlighted.



Amorphic Dashboard:

1. Navigate to Analytics > Dashboard. A sample dashboard has been added. Clone it.

Dashboard Name	Description	Dashboard Type	Last Modified Time	Options
VaccineAnalyticsDashboard Vaccine Analytics	Vaccine Analytics	Quicksight	a few seconds ago	Clone Dashboard

2. Update the Dashboard name (prefix it with your name) and QuickSight Dashboard Id that you copied in the previous step and click on Next

Clone Dashboard

Metadata

Dashboard Name: VaccineAnalyticsDashboard_harsha

Description: Vaccine Analytics

Dashboard Type: Quicksight

Quicksight Dashboard Id: d14eabc2-f365-4aae-89e9-f87f2ffa6240

Keywords: Owner: harsha

Next

3. Finally click on Create to create the dashboard.

Clone Dashboard

Metadata

Connection Details

Keywords: Owner: harsha

Dashboard Name: VaccineAnalyticsDashboard_harsha

Dashboard Type: Quicksight

Quicksight Dashboard Id: d14eabc2-f365-4aae-89e9-f87f2ffa6240

Description: Vaccine Analytics

JSON Payload

Create

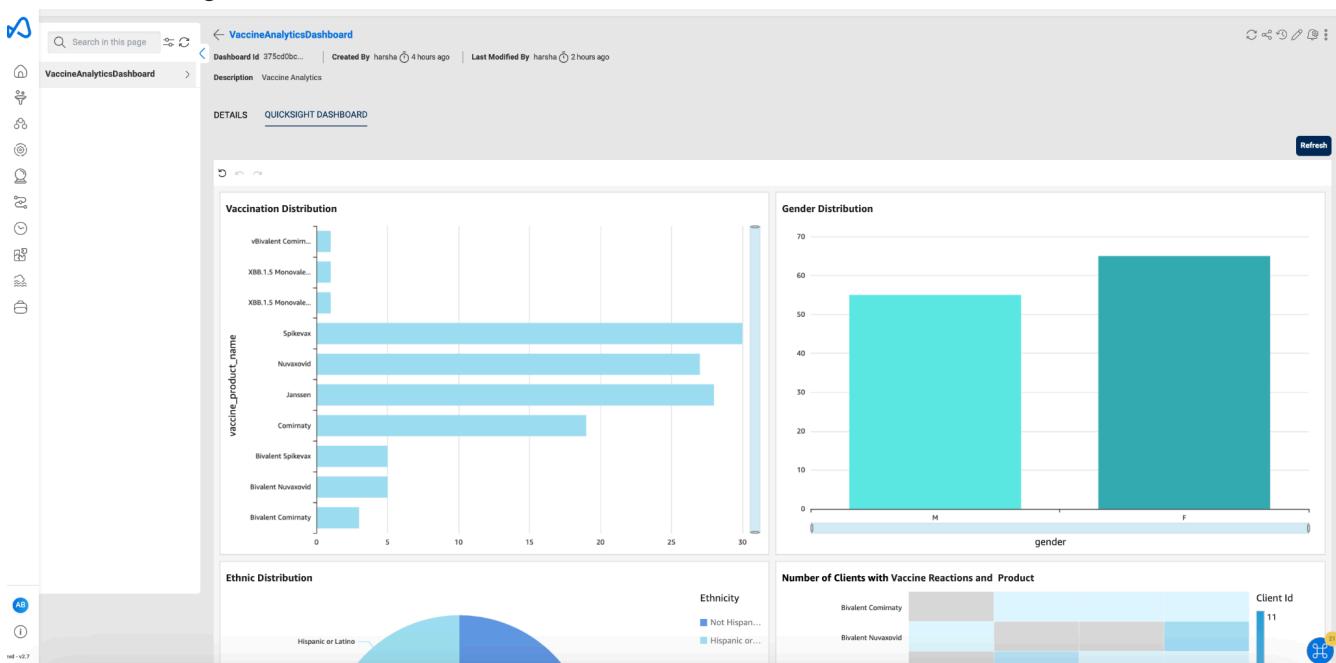


4. Dashboard will be created

The screenshot shows the Quicksight dashboard creation interface. At the top right, there is a green checkmark icon and a message: "Dashboard registration completed successfully". The dashboard details are as follows:

- Dashboard Name: VaccineAnalyticsDashboard
- Dashboard Type: Quicksight
- Keywords: OH

5. Click on Quicksight Dashboard to view the Dashboard.





Adding Dashboard as widget:

1. Navigate to home and click on widget registry to open it

The screenshot shows the Amorphic home interface. At the top, there's a navigation bar with the Amorphic logo and a "Home" link. On the far right of the header, there's a "Widget Registry" button. Below the header, the main content area is currently empty, represented by a large grey box.

Click on create widget group.

The screenshot shows the "Widget Registry" page. The left sidebar has a "Widget Groups" section with a message "No records found". On the right side, there's a large icon of a calendar or dashboard with a blue "0" on it, and a "Create Widget Group" button with a plus sign.

2. Give the widget a name and toggle the button to make this default and click on Create.

The screenshot shows the "Create Widget Group" form. It has a file upload section with "Click here - OR - drag and drop file here" and "Accepted file type - .json". Below that is a "Widget Group Name" field with a red asterisk indicating it's required, containing the value "MyWidget". There's also a "Description" field and a "Make this Group as Default" toggle switch which is turned on (green). At the bottom is a prominent "Create" button.

Widget Group Name *

MyWidget

Description

Make this Group as Default

Create



3. Choose a widget you want to add. In this case a quicksight Dashboard. Click on the corresponding + icon.

The screenshot shows the Amorphic Widget Registry interface. On the left, there's a sidebar titled "Widget Groups" with a "+ Add" button. Below it, under "MyWidget", there's a small preview icon and a "View Details" button. To the right, there's a search bar with a magnifying glass icon and the placeholder "Filter by title or description". Above the main content area, there are several icons: a circular arrow, a downward arrow, a refresh symbol, and a trash bin. The main area is titled "Available Widgets" and contains a table with columns: Name, Description, and Options. The table lists eight widgets:

Name	Description	Options
Access Requests	List of Access requests for datasets access, both sent and received	(i) +
Dataset Lineage	To check the dataset journey	(i) +
FlexiQuery	Quick query to run	(i) +
Insights Details	Watch list of the selected insight	(i) +
Latest Connection Tasks Status	Displays the latest tasks and their statuses for given connection	(i) +
Latest schedule executions Status	Displays the latest executions and their statuses for given schedule	(i) +
My Modules	List of all services/resources user has access to	(i) +
Quick Sight Dashboard	Rended Quicksight dashboard from list of user dashboards	(i) +

A black button at the bottom right of the table says "Add this Widget".

4. Choose from the available dashboard and click on update.

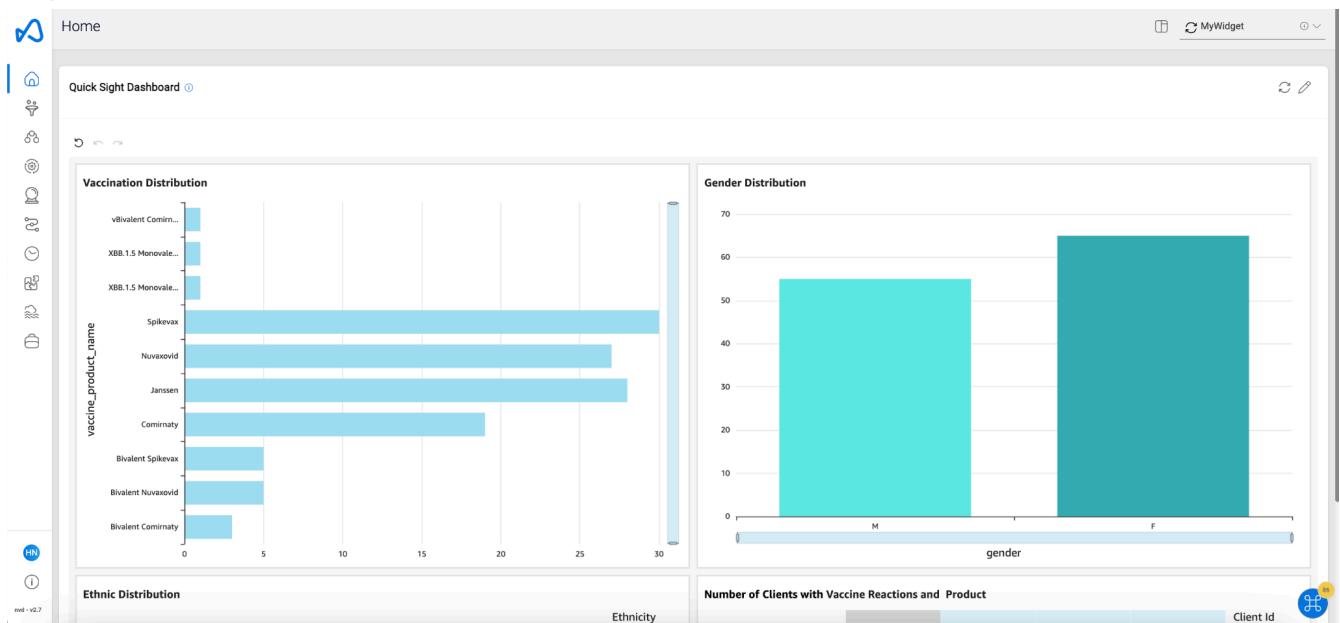
The screenshot shows the "Edit Widget Configuration" form. At the top, there's a large input field with a placeholder "Click here - OR - drag and drop file here" and a note "Accepted file type - .json". Below this, there are several configuration sections:

- Title (i)**: Quick Sight Dashboard
- Description (i)**: Rended Quicksight dashboard from list of user dashboards
- Dashboard Id (i) ***: VaccineAnalyticsDashboard
- Dashboard Preview**: A list box containing "VaccineAnalyticsDashboard" and "VaccineAnalyticsDashboard_harsha".
- Full Page**: A dropdown menu currently set to "Full Page".

At the bottom right, there's a large blue "Update" button.



5. Go back to the Home Page and you should now see a QuickSight Dashboard you created as a widget.





LAB Session 6:

Studios:

Amazon SageMaker Studio is an integrated development environment(IDE) that provides a single web-based visual interface where you can access purpose-built tools to perform all machine learning (ML) development steps, from preparing data to building, training, and deploying your ML models.

Note: For the purpose of this demo we will only see the capability of the studio and will not be creating any models as it would take some time to train and procure the data. You can choose to experiment.

Procedure:

1. Navigate to Analytics > Studio
2. A studio has been shared with you. Let's create a new studio by cloning the existing ones. Click on the **Clone Studio** button to begin

Studio Name	Last Modified Time	Created By	Options
sample-studio Sample Studio	an hour ago	harsha	

3. Give a studio a name. Rename **sample-studio** to **sample-studio-<yourname>** Ex:
sample-studio-adityabhat



4. Take a look at the properties in Instance Metadata. Give access to the Dataset you want to read and Parameter Access that needs to be accessed in the notebook. Click on **Continue**.

Clone Studio

Upload Data >	Allowed Instances List ⓘ * ml.t3.medium USD 0.0500/hour ml.t3.large USD 0.1000/hour	Direct Internet Access ⓘ * Yes
Metadata >	Volume Size (GB) ⓘ * 50	Max Volume Size (GB) ⓘ 100
Instance Metadata >	RStudio Access ⓘ * Disabled	Parameter Access ⓘ SYSTEM.S3BUCKET.LZ SYSTEM.S3BUCKET.DLZ
Cost Tags >	Dataset Write Access ⓘ	Dataset Read Access ⓘ bronzehealth: Immunization
Review & Submit >	Shared Libraries ⓘ	Jupyter Lab Instance Type ⓘ ml.t3.medium USD 0.0500/hour

Continue

5. Skip Cost Tags and head to *Review and Submit*. Click on Clone Studio.

Clone Studio

Upload Data >	Description Sample Studio
Metadata >	Keywords Owner: adityabhat
Instance Metadata >	Direct Internet Access Enabled
Cost Tags >	Volume Size In GB 50
Review & Submit >	Max Volume Size In GB 100
	RStudio Access Status Disabled
	Jupyter Lab Instance Type ml.t3.medium
	Owner Dataset Access -
	ReadOnly Dataset Access Immunization
	Parameter Access SYSTEM.S3BUCKET.LZ SYSTEM.S3BUCKET.DLZ
	Shared Libraries -
	Instance List ml.t3.medium ml.t3.large
	Cost Tags
	JSON Payload

Clone Studio

6. Wait for a few minutes for the studio instance to spin up. Your studio will be ready.



7. Once the instance is up click on the User Profile URL to jump directly to the studio.

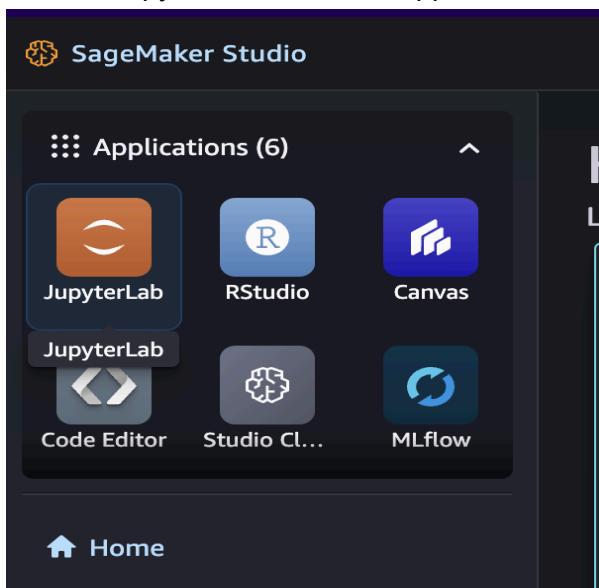
The screenshot shows the 'Studios' section of the AWS console. A specific studio named 'sample-studio-adityabhat' is selected. On the right, the 'User Profile Details' panel is expanded, showing the 'User Profile URL' field which contains the value 'user-adityabhat'. Other details like 'User Profile Name' and 'User Profile Status' are also visible.

8. Take a tour of the studio if you would like to else skip tour

The screenshot shows the 'Welcome to the new SageMaker Studio' tour modal. It features a central icon of a brain-like shape with lines, followed by the text 'Welcome to the new SageMaker Studio'. Below it says 'We've built a new experience to empower you and your work.' There are two buttons at the bottom: 'Skip Tour for now' and 'Take a quick tour' (which is highlighted with a blue background). A message at the bottom asks if the user is an existing Studio Classic user and provides a link to learn how to migrate.



9. Choose JupyterLab under the Applications from the left pane.



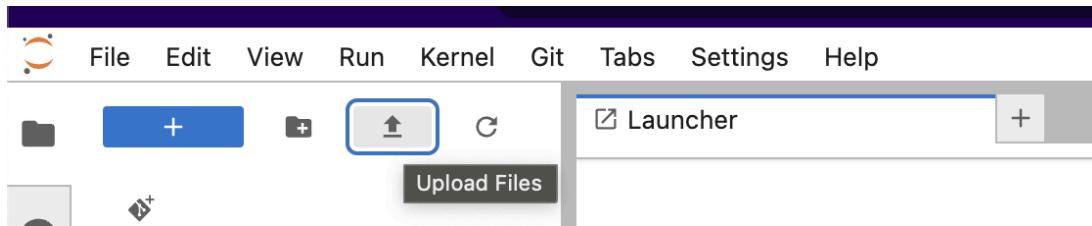
10. Open the JupyterLab.

A screenshot of the JupyterLab page within the SageMaker Studio. The top navigation bar shows "SageMaker Studio > Jupyterlab". The main content area has a title "JupyterLab" with a "Create JupyterLab space" button. Below it is an "About" section with a description of JupyterLab as a web-based IDE. There are "See features" and "Quick start guide" links, a search bar, and a filter for "Running" spaces. A table lists one running JupyterLab space: "Name: Jupyter-Lab-Space, Application: JupyterLab, Status: Running, Type: Shared, Last modified: 5 minutes ago". At the bottom are "Go to page 1" and "Page 1 of 1" buttons.

11. For this lab we will only be reading from the Dataset. Download the script from the URL below.
<https://github.com/cwkadityabhat/nevada-public-datalake/blob/main/Lab6/sample-read-nb.ipynb>

A screenshot of a GitHub repository page for "nevada-public-datalake / Lab6 / sample-read-nb.ipynb". The file was uploaded by "cwkadityabhat" 42 minutes ago. The file size is 12.6 KB and contains 340 lines (340 loc). There are "Preview", "Code", and "Blame" buttons at the bottom. A note says "Code 55% faster with GitHub Copilot". There are download and copy buttons at the bottom right.

12. Lets upload the notebook downloaded to our JupyterLab. Back in the studio on the left hand pane on the top hand side you will see an upload button.



Click on it to choose the file you downloaded (sample-read-nb.ipynb).

13. You can now see the notebook uploaded. Double click on the notebook to open in the new tab.

```

[1]: import boto3
import pandas as pd

[2]: s3 = boto3.client('s3', region_name="us-west-2")
ssm = boto3.client("ssm", region_name="us-west-2")

[3]: dlz_bucket = ssm.get_parameter(Name="SYSTEM.S3BUCKET.DLZ", WithDecryption=False)["Parameter"]["Value"]
l2_bucket = ssm.get_parameter(Name="SYSTEM.S3BUCKET.L2", WithDecryption=False)["Parameter"]["Value"]
prefix = 'bronzehealth/Immunization/' # prefix of Dataset you have access to

[4]: output_df = pd.DataFrame()
s3_obj_list = s3.list_objects(Bucket = dlz_bucket, Prefix = prefix)['Contents']
print(s3_obj_list)

[5]: for obj in s3_obj_list:
    data = s3.get_object(Bucket=dlz_bucket, Key=obj.get('Key'))
    s3_data = data['Body']
    df = pd.read_csv(s3_data, header=0)
    output_df = pd.concat([output_df, df], ignore_index=True)

[6]: output_df.head(5)

```

immunization_id	client_id	client_first_name	client_last_name	client_suffix	client_date_of_birth	client_age	client_alias_last_name	cli
b4084d37-20de-	00901a88-							

14. Run each cell by clicking Shift + Enter or Run Icon on the top.

15. Once done to stop incurring any cost, Delete the studio.

Congrats!! You have successfully created the Studio ,Run a notebook and Delete a Studio.

Feel free to explore other great functionality of SageMaker studio

<https://aws.amazon.com/sagemaker-ai/studio/>

<https://docs.aws.amazon.com/sagemaker/latest/dg/studio-updated.html>

