

The Data Science team would like you to write a Glue job that does the following:

- Sanitize the Customer data (Trusted Zone) and create a Glue Table (Curated Zone) that only includes customers who have accelerometer data *and* have agreed to share their data for research called **customers\_curated**.

Finally, you need to create two Glue Studio jobs that do the following tasks:

- Read the Step Trainer IoT data stream (S3) and populate a Trusted Zone Glue Table called **step\_trainer\_trusted** that contains the Step Trainer Records data for customers who have accelerometer data and have agreed to share their data for research (customers\_curated).
- Create an aggregated table that has each of the Step Trainer Readings, and the associated accelerometer reading data for the same timestamp, but only for customers who have agreed to share their data, and make a glue table called **machine\_learning\_curated**.

Customer\_curated

## customer\_curated\_job

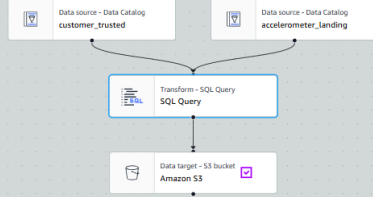
Last modified on 22/04/2025, 14:49:09

Actions

Save

Run

Visual Script Job details Runs Data quality Schedules Version Control



SQL Query

### Node parents

Choose which nodes will provide inputs for this one.

Choose one or more parent node

customer\_trusted

Catalog - DataSource

accelerometer\_landing

Catalog - DataSource

### Associate an alias with each input source

Edit the aliases used for the inputs to this node.

### Input sources

customer\_trusted

accelerometer\_landing

### SQL aliases

customer

accelerometer

### SQL query

Enter a SQL statement to add to your job.

```
1 select distinct c.customername,
2 c.email,
3 c.phone,
4 c.birthdav,
```

Data preview

Output schema

Data preview (200)

Info

READY



End session

Previewing 10 of 10 fields

Filter sample dataset

customername

email

phone

birthday

< \_landing\_sql : X accelerometer\_landi... : X steptrainer\_sql : X Query 16 : X Query 17 : X Query 18 : > ( + )

```
1 select * from customer_curated;
```

SQL Ln 1, Col 1

Run again

Explain

Cancel

Clear

Create

Reuse query results  
up to 60 minutes ago

Query results

Query status

Completed

Time in queue: 67 ms

Run time: 731 ms

Data scanned: 161.03 KB

Results (482)

Copy

Download results CSV

Query results

Query status

Completed

Time in queue: 76 msRun time: 1.203 secData scanned: 161.03 KB

Results (482)

Copy

Download results CSV

Search rows

<1>

#	customername	email	phone	birthday	serialnumber	registrationdate	lastupdatec
1	Angie Abram	Angie.Abram@test.com	8015551212	1539-01-01	0d30f1a9-7492-4cc1-95df-d9f5a159ed92	1655564403947	165556440
2	Bobby Davis	Bobby.Davis@test.com	8015551212	1138-01-01	db0fbb25-f82c-428c-823a-35274be4f259	1655564433827	165556443
3	Christina Mitra	Christina.Mitra@test.com	8015551212	1160-01-01	7414a106-ee97-4824-ab25-70bedeae1136	1655564432243	165556443
4	Craig Jefferson	Craig.Jefferson@test.com	8015551212	1666-01-01	f8e09680-6e0c-46b4-b7a7-eacd8372b079	1655564394456	165556439
5	Edward Abram	Edward.Abram@test.com	8015551212	1435-01-01	31cc254a-8982-4ace-a538-a556fdc2fec6	1655564411759	165556441
6	Eric Spencer	Eric.Spencer@test.com	8015551212	1002-01-01	9d2f683d-7588-41e9-b5af-704a191b2cec	1655564443889	165556444
7	John Wu	John.Wu@test.com	8015551212	1938-01-01	70cdc373-146e-4de9-80b3-941129bc7e42	1655564373109	165556437
8	Sarah Khatib	Sarah.Khatib@test.com	8015551212	1097-01-01	be0294fc-67e9-48dc-aa8a-3015fdcfe9cb	1655564436771	165556443
9	Suresh Jefferson	Suresh.Jefferson@test.com	8015551212	1219-01-01	8dc7645e-9700-4304-bba0-c5818b438806	1655564138779	165556413
10	Travis Abram	Travis.Abram@test.com	8015551212	1374-01-01	e2d30dec-120a-4f37-b571-7c27f49cbde8	1655562436181	165556249
11	Travis Clayton	Travis.Clayton@test.com	8015551212	988-01-01	c87d2224-b0f1-4392-9182-f9f7dcfd45c	1655564444965	165556444
12	Trevor Ahmed	Trevor.Ahmed@test.com	8015551212	1654-01-01	13ee02ac-37d3-40d2-af66-63c43d6ca079	1655564395318	165556439

Steptrainer trusted:

step\_trainer\_trusted

Last modified on 22/04/2025, 15:02:37

Actions

Save

Run

Visual

Script

Job details

Runs

Data quality

Schedules

Version Control

+

Data source - Data Catalog  
steptrainer

Data source - Data Catalog  
customer\_curated

Transform - SQL Query  
SQL Query

Data target - S3 bucket  
Amazon S3

Choose which nodes will provide inputs for this one.

Choose one or more parent node

steptrainer

customer\_curated

Associate an alias with each input source

Info

Edit the aliases used for the inputs to this node.

Input sources

SQL aliases

steptrainer

customer\_curated

steptrainer

customer\_curated

SQL query

Enter a SQL statement to add to your job.

```

1 select distinct s.sensorreadingtime,
2 s.serialnumber,s.distancefromobject
3 from steptrainer s
4 inner join customer_curated c
5 on s.serialnumber = c.serialnumber
6
7

```

Data preview

Output schema

Data preview (200)

Info

READY

End session

Previewing 3 of 3 fields

ng\_sql : X accelerometer\_landi... : X steptrainer\_sql : X Query 16 : X Query 17 : X Query 18 : X > ( + ) ▼

1 select \* from steptrainer\_trusted;

SQL Ln 1, Col 35

Run again Explain Cancel Clear Create ▼

Reuse query results up to 60 minutes ago

Query results Query status

Completed Time in queue: 73 ms Run time: 609 ms Data scanned: 1.59 MB

Results (14,460) Copy Download results CSV

Completed Time in queue: 73 ms Run time: 609 ms Data scanned: 1.59 MB

Results (14,460) Copy Download results CSV

Search rows

< 1 ... > ⚙

#	▼	sensorreadingtime	▼	serialnumber	▼	distancefromobject	▼
1		1655563987071		12ce2113-ab71-4761-89dc-ccef475d6da3		227	
2		1655563960709		12ce2113-ab71-4761-89dc-ccef475d6da3		234	
3		1655563941879		12ce2113-ab71-4761-89dc-ccef475d6da3		242	
4		1655563930581		12ce2113-ab71-4761-89dc-ccef475d6da3		233	
5		1655563919283		12ce2113-ab71-4761-89dc-ccef475d6da3		248	
6		1655563911751		12ce2113-ab71-4761-89dc-ccef475d6da3		297	
7		1655563904219		12ce2113-ab71-4761-89dc-ccef475d6da3		201	
8		1655563885389		12ce2113-ab71-4761-89dc-ccef475d6da3		279	
9		1655563983305		12ce2113-ab71-4761-89dc-ccef475d6da3		239	
10		1655563972007		12ce2113-ab71-4761-89dc-ccef475d6da3		220	
11		1655563956943		12ce2113-ab71-4761-89dc-ccef475d6da3		275	

Machinelearningcurated:



Completed

Time in queue: 104 ms

Run time: 1.586 sec

Data scanned: 8.23 MB

Results (43,681)

Search rows

< 1 ... >

⚙

Copy

Download results CSV

#	z	y	user	timestamp	x	serialnumber	distancefromobject
1	0.0	-1.0	Spencer.Jones@test.com	1655564392083	1.0	8a6ff576-1011-4abe-8889-e3c8a2a755f6	292
2	0.0	0.0	Spencer.Jones@test.com	1655564392083	0.0	8a6ff576-1011-4abe-8889-e3c8a2a755f6	292
3	-1.0	-1.0	Spencer.Jones@test.com	1655564392083	-1.0	8a6ff576-1011-4abe-8889-e3c8a2a755f6	292
4	0.0	1.0	Jane.Hansen@test.com	1655564447394	1.0	a7842856-e8a5-474d-8d2e-e825d6f5323f	296
5	0.0	-1.0	Jane.Hansen@test.com	1655564447394	1.0	a7842856-e8a5-474d-8d2e-e825d6f5323f	296
6	-1.0	1.0	Jane.Hansen@test.com	1655564447394	0.0	a7842856-e8a5-474d-8d2e-e825d6f5323f	296
7	0.0	0.0	Larry.Jones@test.com	1655564438781	1.0	f212d660-7970-4916-b6eb-e69e6719e800	219
8	-1.0	0.0	Larry.Jones@test.com	1655564438781	0.0	f212d660-7970-4916-b6eb-e69e6719e800	219
9	-1.0	-1.0	Jane.Huey@test.com	1655563991742	0.0	8b5957cd-e79f-4ba8-a030-e8f4928aef34	229
10	-1.0	1.0	Jane.Huey@test.com	1655563991742	0.0	8b5957cd-e79f-4ba8-a030-e8f4928aef34	229
11	-1.0	1.0	Jane.Huey@test.com	1655563991742	-1.0	8b5957cd-e79f-4ba8-a030-e8f4928aef34	229
12	1.0	1.0	Jon.Abram@test.com	1655564474488	0.0	275b4d51-0a56-48d1-8d4a-46e7a2664a4f	272

Counts :

- Landing
  - Customer: 956
  - Accelerometer: 81273
  - Step Trainer: 28680

Customer\_landing

1 select count(*) from customer_landing;	
SQL Ln 1, Col 39	
<div> <div>Run again</div> <div>Explain</div> <div>Cancel</div> <div>Clear</div> <div>Create</div> </div> <div> <div>Reuse query results up to 60 minutes ago</div> </div>	
<div> <div>Query results</div> <div>Query status</div> </div> <div> <div>Completed</div> <div>Time in queue: 109 ms</div> <div>Run time: 477 ms</div> <div>Data scanned: 286.89 KB</div> </div> <div> <div>Results (1)</div> <div>Copy</div> <div>Download results CSV</div> </div> <div> <div>Search rows</div> <div>&lt; 1 &gt;</div> <div>⚙</div> </div>	
#	_col0
1	956

## Accelerometer\_landing

The screenshot shows a SQL query execution interface. The query is: `select count(*) from accelerometer_landing;`. The interface includes a 'Run again' button, an 'Explain' link, and buttons for 'Cancel', 'Clear', and 'Create'. A status bar indicates 'Completed' with a green checkmark, 'Time in queue: 115 ms', 'Run time: 589 ms', and 'Data scanned: 6.55 MB'. Below the status bar, there are buttons for 'Copy' and 'Download results CSV'. A search bar is present with the text 'Search rows'. The results table has one column, '\_col0', and one row with the value 81273.

SQL Ln 1, Col 43

Run again Explain Cancel Clear Create

Reuse query results up to 60 minutes ago

Query results Query status

Completed Time in queue: 115 ms Run time: 589 ms Data scanned: 6.55 MB

Results (1) Copy Download results CSV

Search rows

#	_col0
1	81273

## Steptrainer\_landing

The screenshot shows a SQL query execution interface. The query is: `select count(*) from steptrainer_landing;`. The interface includes a 'Run again' button, an 'Explain' link, and buttons for 'Cancel', 'Clear', and 'Create'. A status bar indicates 'Completed' with a green checkmark, 'Time in queue: 116 ms', 'Run time: 556 ms', and 'Data scanned: 3.15 MB'. Below the status bar, there are buttons for 'Copy' and 'Download results CSV'. A search bar is present with the text 'Search rows'. The results table has one column, '\_col0', and one row with the value 28680.

SQL Ln 1, Col 42

Run again Explain Cancel Clear Create

Reuse query results up to 60 minutes ago

Query results Query status

Completed Time in queue: 116 ms Run time: 556 ms Data scanned: 3.15 MB

Results (1) Copy Download results CSV

Search rows

#	_col0
1	28680

- Trusted
  - Customer: 482
  - Accelerometer: 40981
  - Step Trainer: 14460

## customer\_trusted

1

select count(\*) from customer\_trusted

SQLLn 1, Col 38

Run again

Explain

Cancel

Clear

Create

Reuse query results  
up to 60 minutes ago

Query resultsQuery status

CompletedTime in queue: 68 msRun time: 1.328 secData scanned: 161.06 KB

Results (1)

Search rows

< 1 >

#

\_col0

1482

## Accelerometer\_trusted

1

select count(\*) from accelerometer\_trusted

SQLLn 1, Col 43

Run again

Explain

Cancel

Clear

Create

Reuse query results  
up to 60 minutes ago

Query resultsQuery status

CompletedTime in queue: 71 msRun time: 1.388 secData scanned: 3.30 MB

Results (1)

Search rows

< 1 >

#

\_col0

140981

## Steptrainer\_trusted



1

select count(\*) from steptrainer\_trusted;

SQLLn 1, Col 41

Run again

Explain

Cancel

Clear

Create

Reuse query results  
up to 60 minutes ago

Query resultsQuery status

CompletedTime in queue: 74 msRun time: 1.326 secData scanned: 1.59 MB

Results (1)

Search rows

<1>

Settings

#	_col0
1	14460

- Curated
  - Customer: 482
  - Machine Learning: 43681

Customer\_curated:

1

select count(\*) from customer\_curated;

SQLLn 1, Col 39

Run again

Explain

Cancel

Clear

Create

Reuse query results  
up to 60 minutes ago

Query resultsQuery status

CompletedTime in queue: 67 msRun time: 622 msData scanned: 161.03 KB

Results (1)

Search rows

<1>

Settings

#	_col0
1	482

Machinelearning\_curated

1

select count(\*) from machine\_learning\_curated

SQLLn 1, Col 46

Run again

Explain

Cancel

Clear

Create

Reuse query resu

up to 60 minutes ago

Query results

Query status

Completed

Time in queue: 83 ms

Run time: 1.465 sec

Data scanned: 8.23 MB

Results (1)

Copy

Download results CSV

Search rows

#

\_col0

1

43681

Not clear whether to paste script here or not : it is mention to upload in github

So pasting here :

Customer\_landing\_to\_trusted:

```

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 from awsgluedq.transforms import EvaluateDataQuality
8 from awsglue import DynamicFrame
9
10 def sparkSqlQuery(glueContext, query, mapping, transformation_ctx) -> DynamicFrame:
11     for alias, frame in mapping.items():
12         frame.toDF().createOrReplaceTempView(alias)
13         result = spark.sql(query)
14         return DynamicFrame.fromDF(result, glueContext, transformation_ctx)
15
16 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
17 sc = SparkContext()
18 glueContext = GlueContext(sc)
19 spark = glueContext.spark_session
20 job = Job(glueContext)
21 job.init(args['JOB_NAME'], args)
22
23 # Default ruleset used by all target nodes with data quality enabled
24 DEFAULT_DATA_QUALITY_RULESET = """
25 Rules = [
26     ColumnCount > 0
27 ]

```

Script (Locked) Info

[Download script](#)

[Edit script](#)

```

27 """
28
29 # Script generated for node AWS Glue Data Catalog
30 AWSGlueDataCatalog_node1745299750670 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="customer_landing", transformation_ctx="AWSGlueDataCatalog_node1745299750670")
31
32 # Script generated for node SQL Query
33 SqlQuery0 = '''
34 select customername ,
35 email,
36 phone,
37 birthday,
38 serialnumber,
39 registrationdate,
40 lastupdatedate,
41 sharewithresearchasofdate,
42 sharewithpublicasofdate,
43 sharewithfriendsasofdate
44
45
46
47 from customer_landing
48 where sharewithresearchasofdate is not null;
49 '''
50 SQLQuery_node1745299770123 = sparkSqlQuery(glueContext, query = SqlQuery0, mapping = {"customer_landing":AWSGlueDataCatalog_node1745299750670}, transformation_ctx = "SQLQuery_node1745299770123")
51
52 # Script generated for node Amazon S3
53

```

```

54
55 # Script generated for node Amazon S3
56 EvaluateDataQuality().process_rows(frame=SQLQuery_node1745299770123, ruleset=DEFAULT_DATA_QUALITY_RULESET, publishing_options={"dataQualityEvaluationContext": "EvaluateDataQuality_node1745299690147", "enabled":
57 AmazonS3_node1745299858767 = glueContext.getSink(path="s3://finalprojectsaleem/customer/trusted/", connection_type="s3", update_behavior="UPDATE_IN_DATABASE", partition_keys=[], enable_update_catalog=True, transfo
58 AmazonS3_node1745299858767.setCatalogInfo(catalogDatabase="finaldatabase", catalogTableName="customer_trusted")
59 AmazonS3_node1745299858767.setFormat("json")
60 AmazonS3_node1745299858767.writeFrame(SQLQuery_node1745299770123)
61 job.commit()

```

Accelerometer\_landing\_to\_trusted :

```

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  from awsgluedq.transforms import EvaluateDataQuality
8
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
16 # Default ruleset used by all target nodes with data quality enabled
17 DEFAULT_DATA_QUALITY_RULESET = """
18 Rules = [
19     ColumnCount > 0
20 ]
21 """
22
23 # Script generated for node CustomertrustedData
24 CustomertrustedData_node1745301464483 = glueContext.create_dynamic_frame_from_catalog(database="finaldatabase", table_name="customer_trusted", transformation_ctx="CustomertrustedData_node1745301464483")
25
26 # Script generated for node AccelerometerData
27
28
29 # Script generated for node Join
30 Join_node1745301213533 = Join.apply(frame1=AccelerometerData_node1745301165475, frame2=CustomertrustedData_node1745301464483, keys1=["user"], keys2=["email"], transformation_ctx="Join_node1745301213533")
31
32 # Script generated for node Select Fields
33 SelectFields_node1745301261258 = SelectFields.apply(frame=Join_node1745301213533, paths=["z", "user", "y", "x", "timestamp"], transformation_ctx="SelectFields_node1745301261258")
34
35 # Script generated for node Amazon S3
36 EvaluateDataQuality().process_rows(frame=SelectFields_node1745301261258, ruleset=DEFAULT_DATA_QUALITY_RULESET, publishing_options={"dataQualityEvaluationContext": "EvaluateDataQuality_node1745300140421", "ona
37 AmazonS3_node1745301294820 = glueContext.getSink(path="s3://finalprojectsaleem/accelerometer/trusted/", connection_type="s3", updateBehavior="UPDATE_IN_DATABASE", partitionKeys=[], enableUpdateCatalog=True, t
38 AmazonS3_node1745301294820.setCatalogInfo(catalogDatabase="finaldatabase", catalogTableName="accelerometer_trusted")
39 AmazonS3_node1745301294820.setFormat("json")
40 AmazonS3_node1745301294820.writeFrame(SelectFields_node1745301261258)
41 job.commit()

```

Customer\_curated :

```

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 from awsgluedq.transforms import EvaluateDataQuality
8 from awsglue import DynamicFrame
9
10 def sparkSqlQuery(glueContext, query, mapping, transformation_ctx) -> DynamicFrame:
11     for alias, frame in mapping.items():
12         frame.toDF().createOrReplaceTempView(alias)
13     result = spark.sql(query)
14     return DynamicFrame.fromDF(result, glueContext, transformation_ctx)
15 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
16 sc = SparkContext()
17 glueContext = GlueContext(sc)
18 spark = glueContext.spark_session
19 job = Job(glueContext)
20 job.init(args['JOB_NAME'], args)
21
22 # Default ruleset used by all target nodes with data quality enabled
23 DEFAULT_DATA_QUALITY_RULESET = ""
24 Rules = [
25     ColumnCount > 0
26 ]
27

```

```

27 """
28
29 # Script generated for node accelerometer_landing
30 accelerometer_landing_node1745312916301 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="accelerometer_landing", transformation_ctx="ac
31
32 # Script generated for node customer_trusted
33 customer_trusted_node1745312877151 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="customer_trusted", transformation_ctx="customer_tru
34
35 # Script generated for node SQL Query
36 SqlQuery0 = ""
37 select distinct c.customername,
38 c.email,
39 c.phone,
40 c.birthdate,
41 c.serialnumber,
42 c.registrationdate,
43 c.lastupdatedate,
44 c.sharewithresearchasofdate,
45 c.sharewithpublicasofdate,
46 c.sharewithfriendsasofdate
47 from customer c
48 join accelerometer a
49 on c.email = a.user
50 """
51 SqlQuery_node1745312941520 = sparkSqlQuery(glueContext, query = SqlQuery0, mapping = {"customer":customer_trusted_node1745312877151, "accelerometer":accelerometer_landing_n
52

```

```

52 """
53
54 # Script generated for node Amazon S3
55 EvaluateDataQuality().process_rows(frame=SqlQuery_node1745312941520, ruleset=DEFAULT_DATA_QUALITY_RULESET, publishing_options={"dataQualityEvaluationContext": "EvaluateData
56 AmazonS3_node1745313462883 = glueContext.getSink(path="s3://finalprojectsaleem/customer/curated/", connection_type="s3", updateBehavior="UPDATE_IN_DATABASE", partitionKeys=
57 AmazonS3_node1745313462883.setCatalogInfo(catalogDatabase="finaldatabase", catalogTableName="customer_curated")
58 AmazonS3_node1745313462883.writeFrame(SqlQuery_node1745312941520)
59 job.commit()

```

Steptrainer\_trusted:

```

1 import sys
2 from aws glue.transforms import *
3 from aws glue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from aws glue.context import GlueContext
6 from aws glue.job import Job
7 from aws glue dq.transforms import EvaluateDataQuality
8 from aws glue import DynamicFrame
9
10 def sparkSqlQuery(glueContext, query, mapping, transformation_ctx) -> DynamicFrame:
11     for alias, frame in mapping.items():
12         frame.toDF().createOrReplaceTempView(alias)
13     result = spark.sql(query)
14     return DynamicFrame.fromDF(result, glueContext, transformation_ctx)
15 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
16 sc = SparkContext()
17 glueContext = GlueContext(sc)
18 spark = glueContext.spark_session
19 job = Job(glueContext)
20 job.init(args['JOB_NAME'], args)
21
22 # Default ruleset used by all target nodes with data quality enabled
23 DEFAULT_DATA_QUALITY_RULESET = """
24     Rules = [
25         ColumnCount > 0
26     ]
27 """

```

```

28 # Script generated for node customer_curated
29 customer_curated_node1745313758346 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="customer_curated", transformation_ctx="customer_curated_node1745313758346")
30
31 # Script generated for node steptrainer
32 steptrainer_node1745313734418 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="steptrainer_landing", transformation_ctx="steptrainer_node1745313734418")
33
34 # Script generated for node SQL Query
35 SqlQuery0 = '''
36 select distinct s.sensorreadingtime,
37 s.serialnumber,s.distancefromobject
38 from steptrainer s
39 inner join customer_curated c
40 on s.serialnumber = c.serialnumber
41
42 '''
43
44 SqlQuery_node1745314070251 = sparkSqlQuery(glueContext, query = SqlQuery0, mapping = {"steptrainer":steptrainer_node1745313734418, "customer_curated":customer_curated_node1745313758346}, transformation_ctx =
45
46 # Script generated for node Amazon S3
47 EvaluateDataQuality().process_rows(frame=SqlQuery_node1745314070251, ruleset=DEFAULT_DATA_QUALITY_RULESET, publishing_options={"dataQualityEvaluationContext": "EvaluateDataQuality_node1745313648135", "enableD
48 AmazonS3_node1745314302198 = glueContext.getSink(path="s3://finalprojectsaaleem/step_trainer/trusted/", connection_type="s3", updateBehavior="UPDATE_IN_DATABASE", partitionKeys=[], enableUpdateCatalog=True, tr
49 AmazonS3_node1745314302198.setCatalogInfo(catalogDatabase="finaldatabase",catalogTableName="steptrainer_trusted")
50 AmazonS3_node1745314302198.setFormat("json")
51 AmazonS3_node1745314302198.writeFrame(SqlQuery_node1745314070251)
52
53 job.commit()

```

## Machinelearning\_curated:

```

1 import sys
2 from aws glue.transforms import *
3 from aws glue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from aws glue.context import GlueContext
6 from aws glue.job import Job
7 from aws glue dq.transforms import EvaluateDataQuality
8
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
16 # Default ruleset used by all target nodes with data quality enabled
17 DEFAULT_DATA_QUALITY_RULESET = """
18     Rules = [
19         ColumnCount > 0
20     ]
21 """
22
23 # Script generated for node steptrainer_trusted
24 steptrainer_trusted_node1745314537862 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="steptrainer_trusted", transformation_ctx="steptrainer_trusted_node1745314537862")
25
26 # Script generated for node accelerometer_trusted

```

```

21 """
22
23 # Script generated for node steptrainer_trusted
24 steptrainer_trusted_node1745314537862 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="steptrainer_trusted", transformation_ctx="steptrainer_trusted_node1745314537862")
25
26 # Script generated for node accelerometer_trusted
27 accelerometer_trusted_node1745314562737 = glueContext.create_dynamic_frame.from_catalog(database="finaldatabase", table_name="accelerometer_trusted", transformation_ctx="accelerometer_trusted_node1745314562737")
28
29 # Script generated for node Join
30 Join_node1745314589281 = Join.apply(frame1=steptrainer_trusted_node1745314537862, frame2=accelerometer_trusted_node1745314562737, keys1=["sensorreadingtime"], keys2=["timestamp"], transformation_ctx="Join_node1745314589281")
31
32 # Script generated for node Amazon S3
33 EvaluateDataQuality().process_rows(frame=Join_node1745314589281, ruleset=DEFAULT_DATA_QUALITY_RULESET, publishing_options={"dataQualityEvaluationContext": "EvaluateDataQuality_node1745314496747", "enableDataQuality": True})
34 AmazonS3_node1745314698522 = glueContext.getSink(path="s3://finalprojectsaleem/machine_learning_curated/", connection_type="s3", updateBehavior="UPDATE_IN_DATABASE", partitionKeys=[], enableUpdateCatalog=True)
35 AmazonS3_node1745314698522.setCatalogInfo(catalogDatabase="finaldatabase", catalogTableName="machine_learning_curated")
36 AmazonS3_node1745314698522.setFormat("json")
37 AmazonS3_node1745314698522.writeFrame(Join_node1745314589281)
38 job.commit()

```

- The customer\_landing data contains multiple rows with a blank shareWithResearchAsOfDate.

1 select \* from customer\_landing where sharewithresearchasofdate is null

SQL Ln 1, Col 67

Run again Explain Cancel Clear Create

Reuse query results up to 60 minutes ago

Query results Query status

Completed Time in queue: 69 ms Run time: 659 ms Data scanned: 286.89 KB

Results (474) Copy Download results CSV

Search rows

customername	email	phone	birthday	serialnumber	registrationdate	lastupdatedate
Frank Hansen	Frank.Hansen@test.com	8015551212	1323-01-01	454a7430-d4ff-47cf-8666-7428f8a9894d	1655564131452	165556413145

Results (474)							<a href="#">Copy</a>	<a href="#">Download results CSV</a>
Search rows							< 1 ... > ⚙	
number	registrationdate	lastupdatedate	sharewithresearchasofdate	sharewithpublicasofdate	sharewithfriendsasofdate			
30-d4ff-47cf-8666-7428f8a9894d	1655564131452	1655564131452			1655564131452			
35-4faf-47ab-96f7-ea79c6e87f0c	1655564435763	1655564435763						
ef-bc66-4128-85af-2c9d36c251b9	1655564387419	1655564387419		1655564387419	1655564387419			
36-5e19-4a70-9016-6e58f146d3f0	1655564397161	1655564397161			1655564397161			
13-8675-495b-a4b3-c5015f0ca3a2	1655564087815	1655564087815			1655564087815			
:7-e15e-48be-bcf7-6ab67028a72c	1655564424393	1655564424393			1655564424393			
4a-3018-416d-801c-cafd3bbe4b49	1655564408518	1655564408518		1655564408518				
78-d6df-4a1a-ace9-0e1c5773b7c7	1655564410194	1655564410194			1655564410194			
1b-70f2-41c1-9573-1a0ee31a1d7d	1655564141755	1655564141755						
8f-2d3f-43a6-868c-dc4697b0b7ff	1655562450053	1655562512396		1655562512396				
f-3756-4907-9b4d-dfc03a7e1c31	1655564138704	1655564138704		1655564138704				
d1-5578-410f-aa2d-b12f487ce65b	1655564426618	1655564426618						
:9-1aa1-4139-9dc8-8f91a6d40d48	1655564393958	1655564393958		1655564393958	1655564393958			
18-cd06-4d2b-9b87-2a89d2423140	1655564439832	1655564439832			1655564439832			

- The resulting customer\_trusted data has no rows where shareWithResearchAsOfDate is blank.

```
1 select * from customer_trusted where sharewithresearchasofdate is null
```

SQL Ln 1, Col 31

[Run again](#)
[Explain](#)
[Cancel](#)
[Clear](#)
[Create](#)

☐ Reuse query results up to 60 minutes ago

Query results

Query status

Completed
Time in queue: 102 ms
Run time: 515 ms
Data scanned: 161.06 KB

Results (0)
[Copy](#)
[Download results CSV](#)

Search rows

# | customername | email | phone | birthday | serialnumber | registrationdate | lastupdatedate | sharewithresearchasofdate | sha

No results

I have included everything only the order is mismatched , but have provided proper heading and references , since some parts were completed in morning sessions and some work was dependent on the other one to complete.