AWS Data Analytics Speciality
Capstone Project (Solution)

About CloudThat

- CloudThat is the first company in India to Cloud Training & Consulting services for mid-market & enterprise clients around the world. With expertise in major Cloud platforms including Microsoft Azure, Amazon Web Services (AWS) and Google Cloud Platform (GCP). CloudThat is uniquely positioned to be the single technology source for organizations looking to utilize the flexibility and power Cloud Computing provides.

- CloudThat is focused on quickly empowering IT professionals and organizations with leveraging Cloud, Big Data & IoT. Founded by Bhavesh Goswami, an ex-Microsoft and ex-Amazonian who was part of the Microsoft and AWS product development teams.

- Till date we have trained more than 200,000 IT professionals and conducted corporate training for some of the fortune 500 companies which include Accenture, Infosys, Fidelity, HCL, Intuit, GE, TCS, HP, SAP, Oracle, Western Union, Philips, Flipkart, L&T and Samsung, just to name a few.

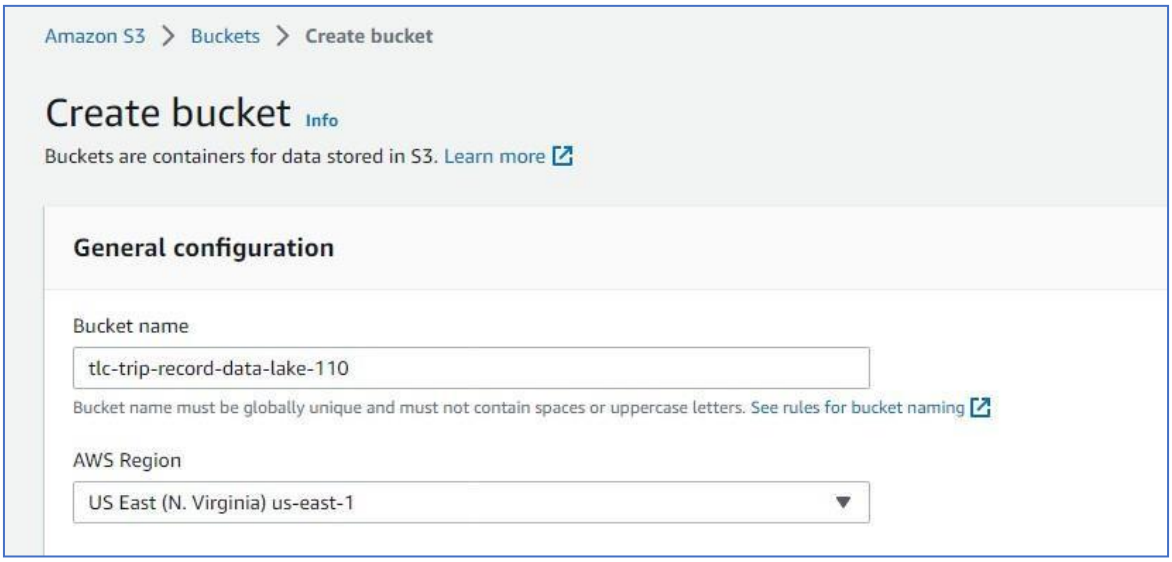
- We have presence in Bengaluru, USA & UK, but offer on-site and pre-scheduled public batches in different IT centric cities of India and Overseas.

- CloudThat is a Microsoft Gold Partner, Advanced AWS Consulting partner, Google Consulting Partner, Red Hat Certified Training Partner, MongoDB Ready Partner, and part of Pearson Testing Network.

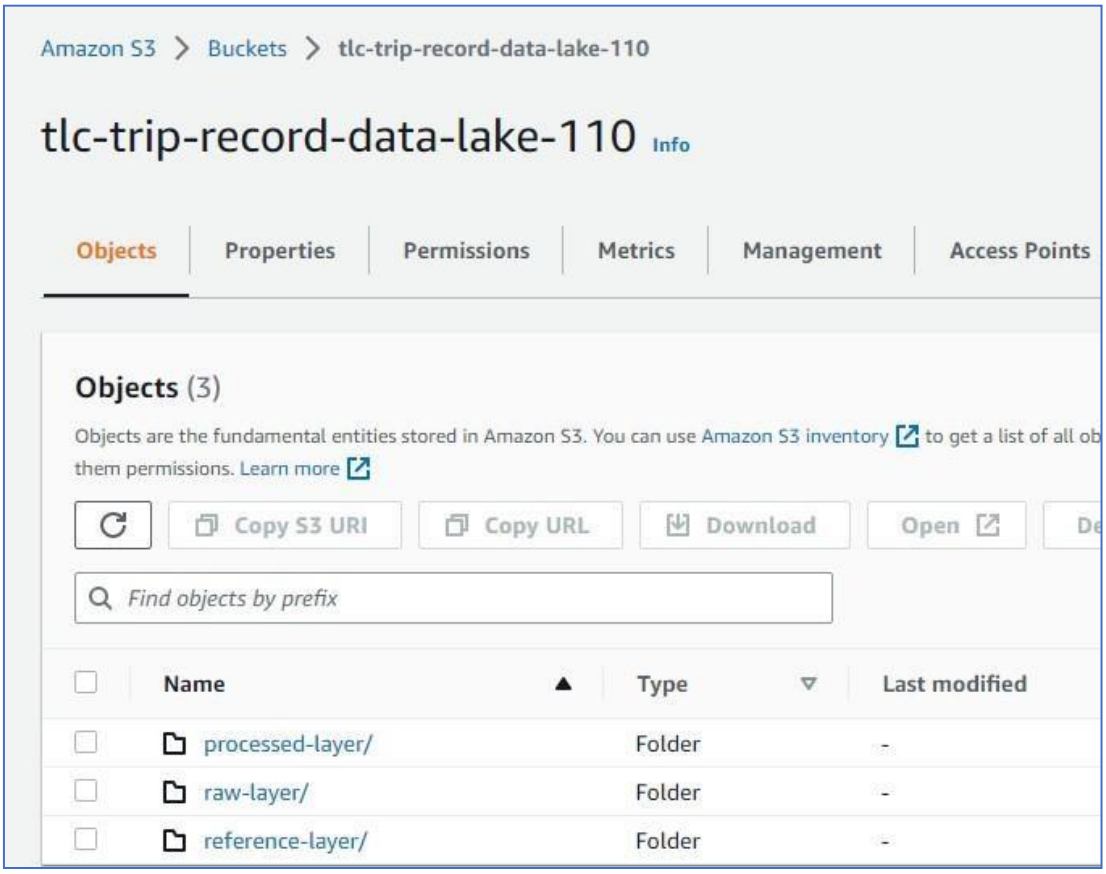
- Our current course offerings are on Azure, Dynamics 365, Microsoft Security Suite, AI & Machine Learning, Cloud Security, Analytics, Red Hat, IoT, DevOps, Chef, Docker, Ansible, Kubernetes, Blockchain, Big Data, etc. We are constantly adding more courses and more consulting offerings.

Phase-1: Ingestion and Storage

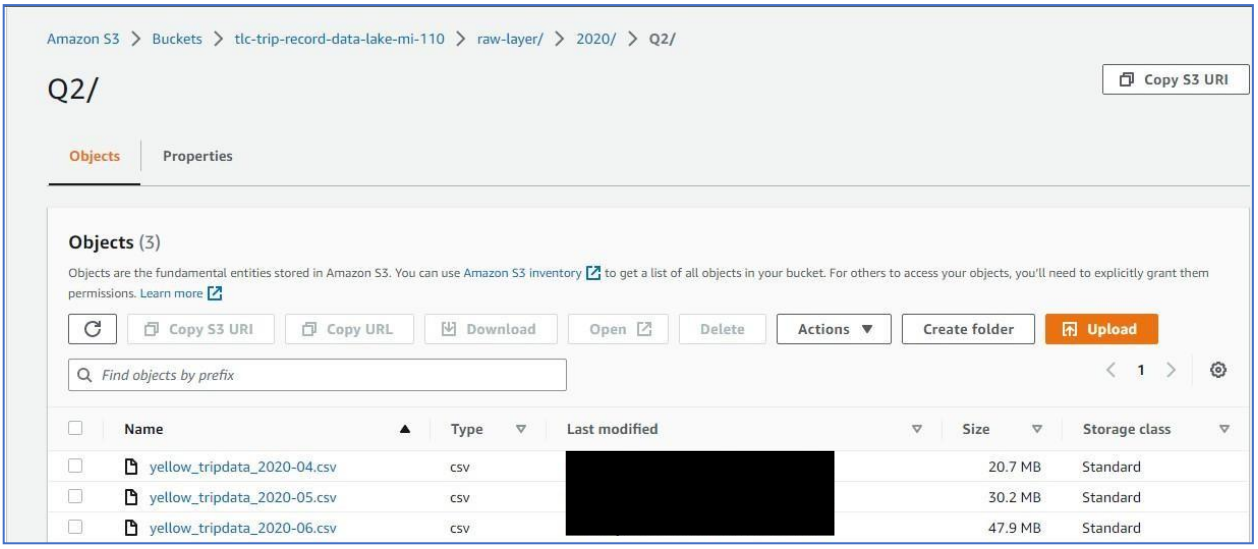
1. Create an S3 bucket to load the raw datasets. Give the name of the bucket as **tlc-trip-record-data-lake-<yourname-random number>** as shown in the figure below.

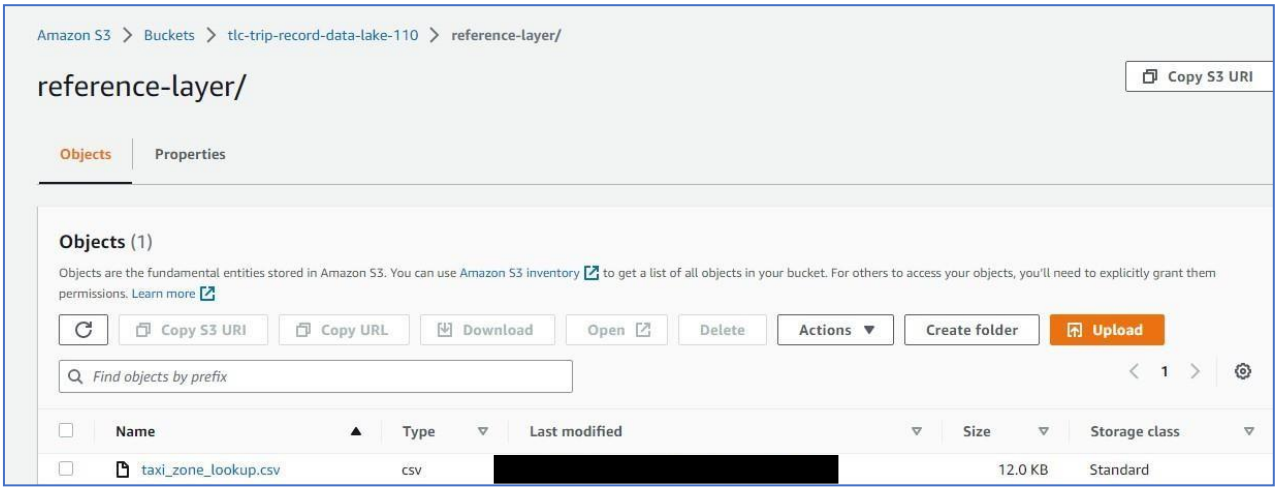


2. Create three folders within the bucket namely **raw-layer**, **processed-layer and reference-layer**.



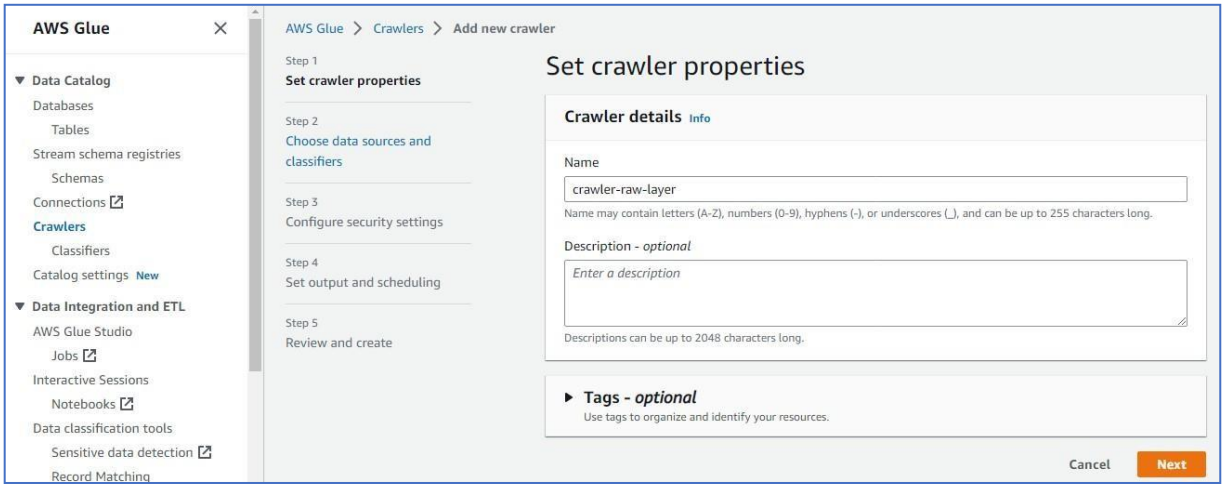
3. Load **raw data files for Q2-2020** in the **raw-layer** with the following hierarchy as **raw/2020/Q2** and **lookup file** in **reference-layer**, respectively.



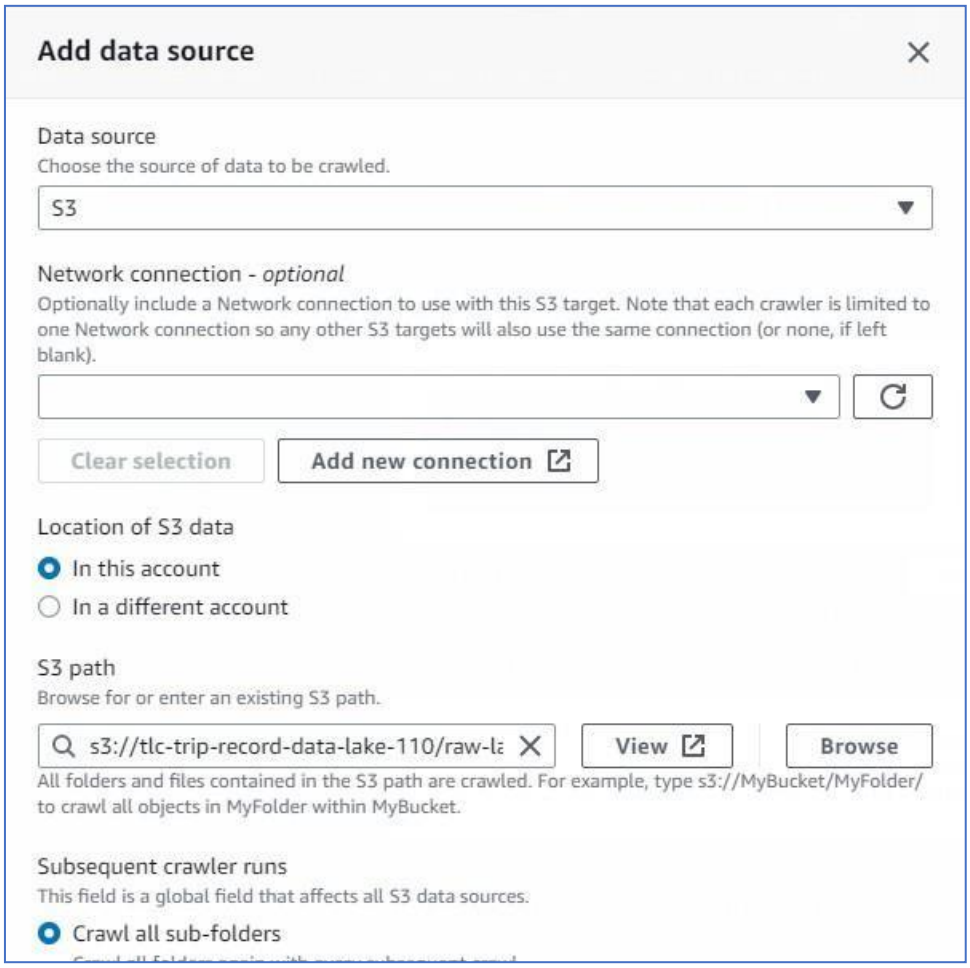


Phase-2: Cataloging

4. Now, let's create a crawler for raw-layer as **crawler-raw-layer-<yourname>**.
   Go to **AWS Glue>>Crawler>>Create Crawler**

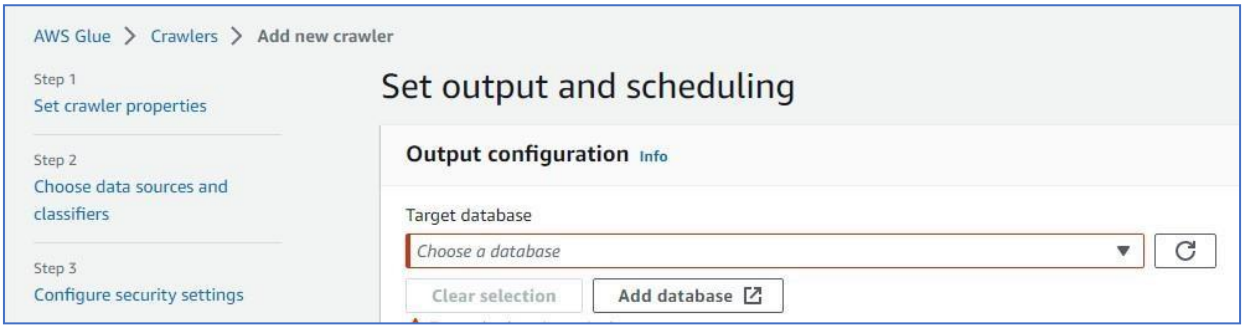


5. Select S3 as data source and choose path as ***s3://tlc-trip-record-data-lake-<yourname-random number>/raw-layer/2020/Q2/***

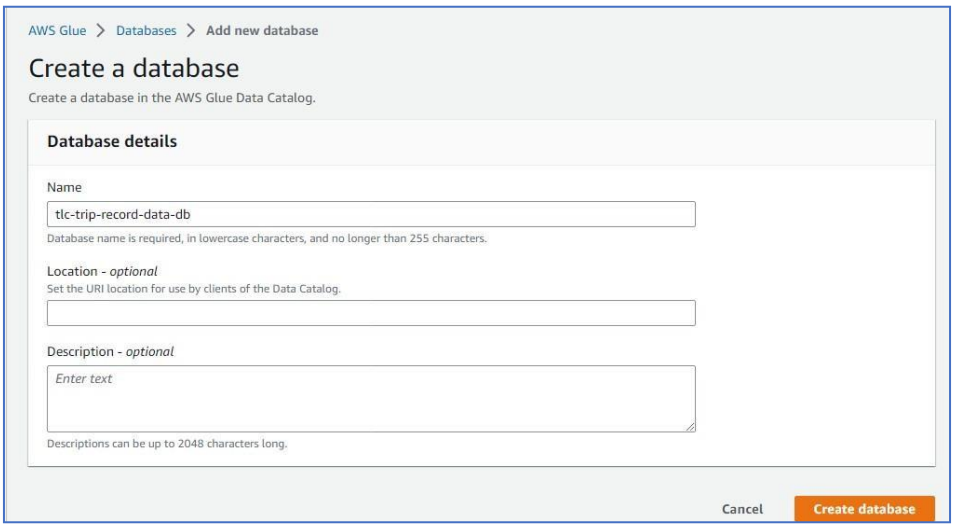


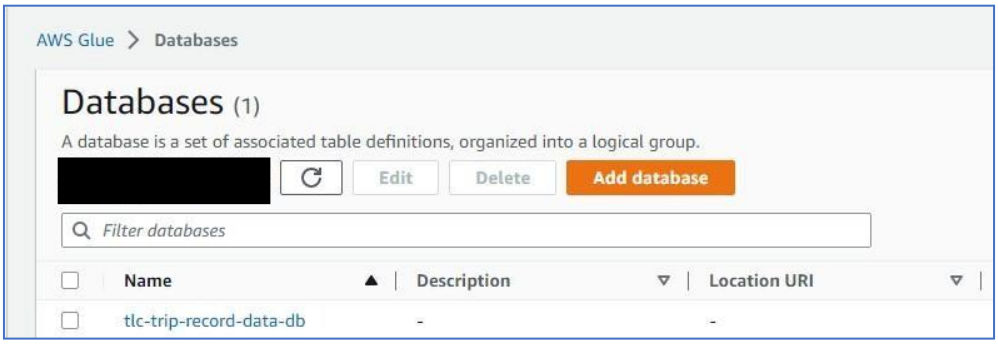
6. Select **GlueCapstoneRole** as IAM role.

7. Click on Add database.

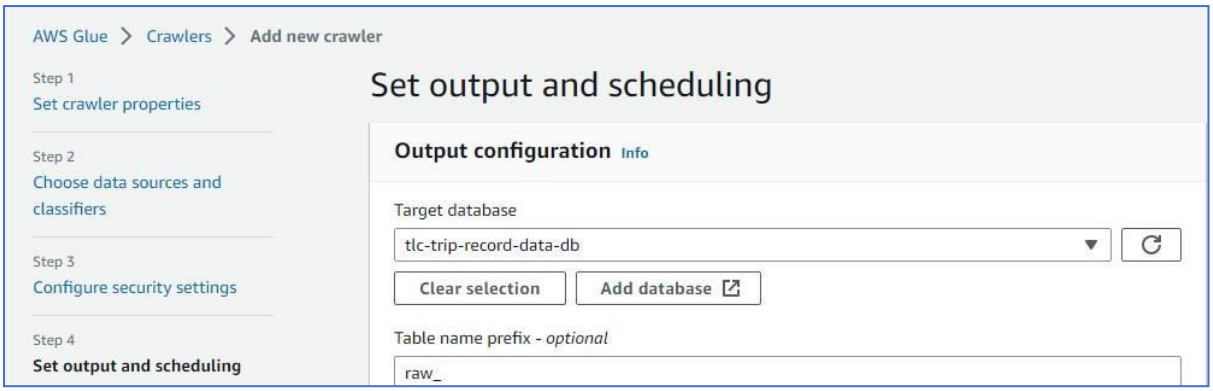


8. Name: **tlc-trip-record-data-db-<yourname>** and click on create database.

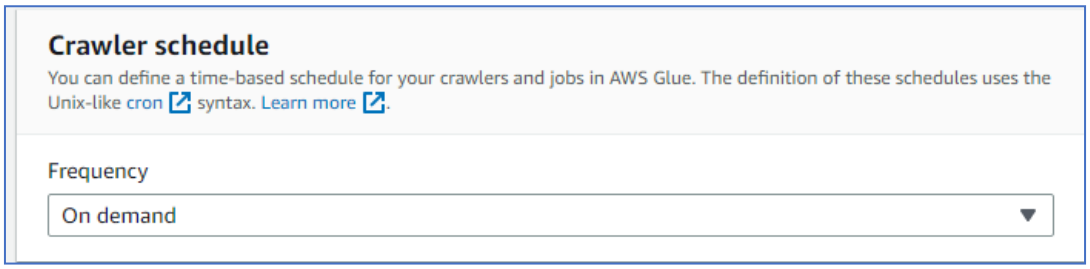




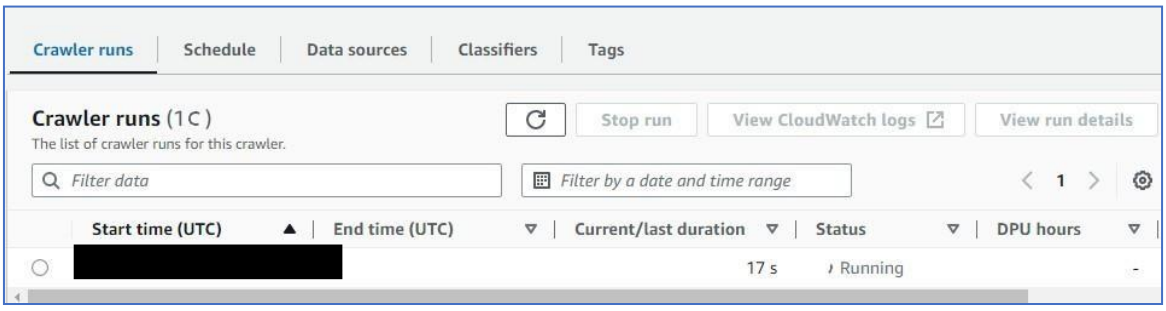
9. Now add the newly created database as Target database.

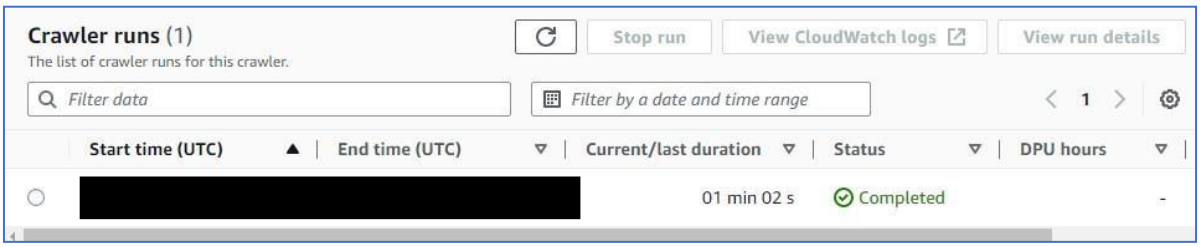


10. Keep crawler schedule frequency as On demand and create the crawler.

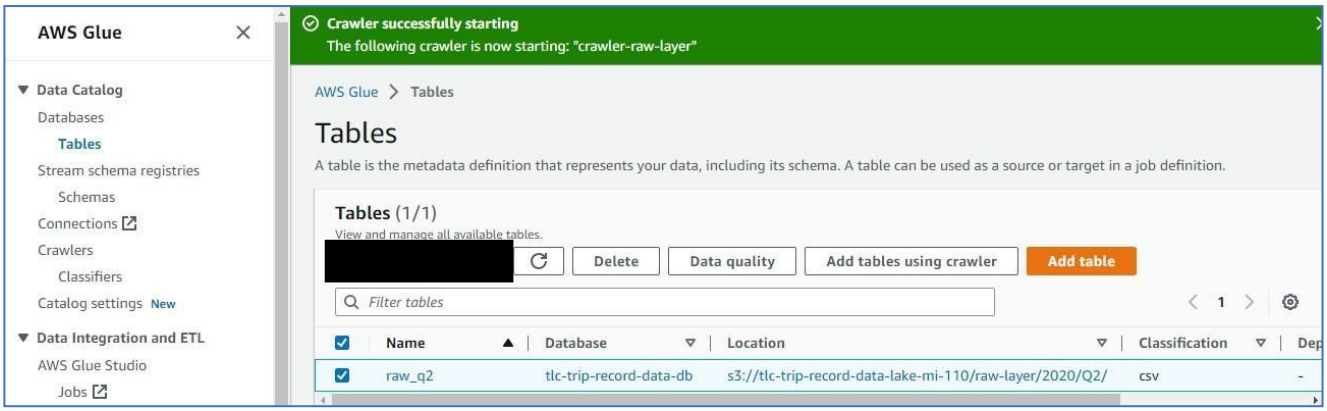


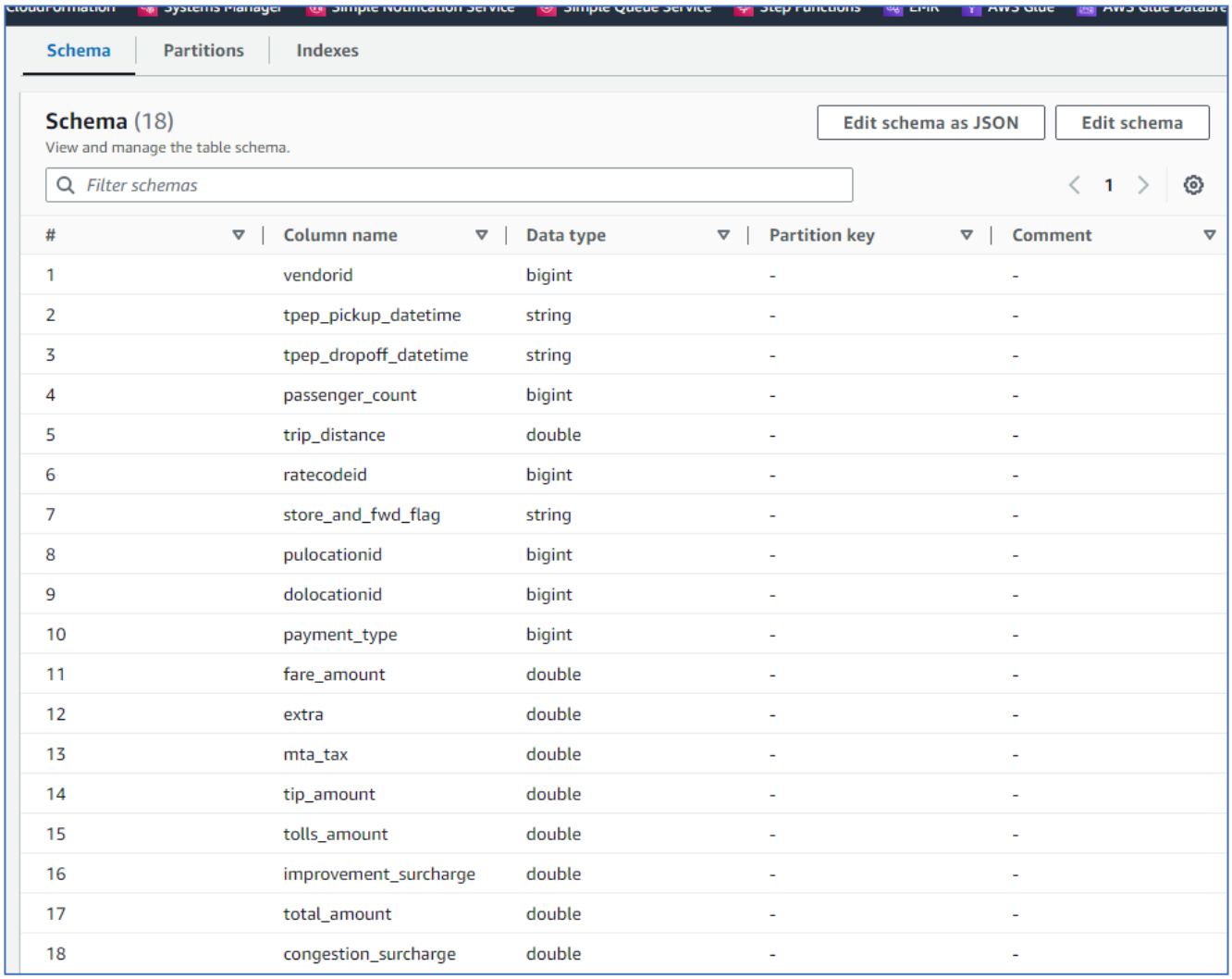
11. Once the crawler is created, run the crawler.





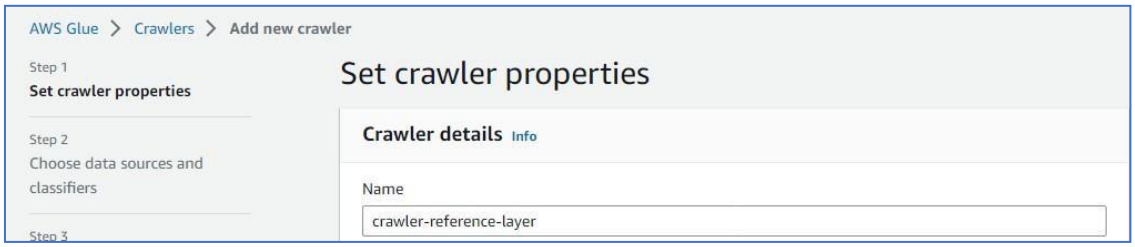
12. See the table got created for raw-layer and explore the metadata.



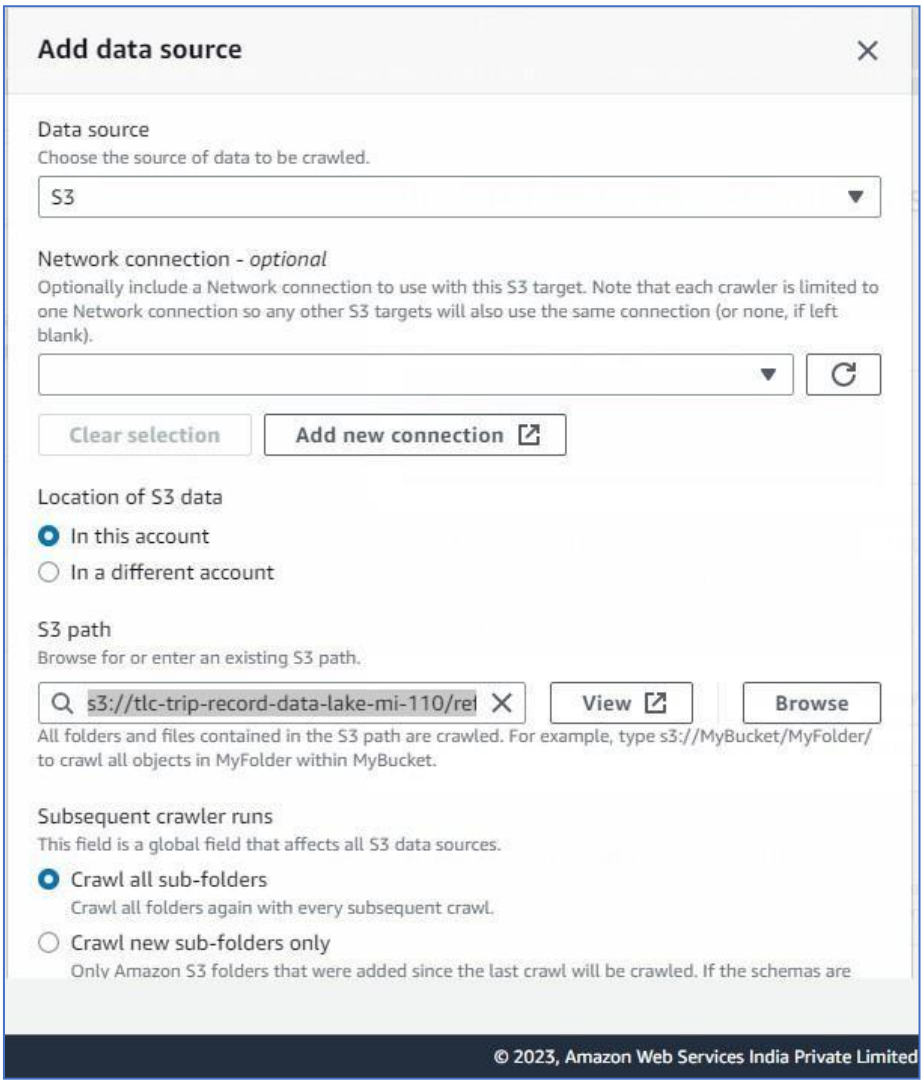


#	Column name	Data type	Partition key	Comment
1	vendorid	bigint	-	-
2	tpep_pickup_datetime	string	-	-
3	tpep_dropoff_datetime	string	-	-
4	passenger_count	bigint	-	-
5	trip_distance	double	-	-
6	ratecodeid	bigint	-	-
7	store_and_fwd_flag	string	-	-
8	pulocationid	bigint	-	-
9	dolocationid	bigint	-	-
10	payment_type	bigint	-	-
11	fare_amount	double	-	-
12	extra	double	-	-
13	mta_tax	double	-	-
14	tip_amount	double	-	-
15	tolls_amount	double	-	-
16	improvement_surcharge	double	-	-
17	total_amount	double	-	-
18	congestion_surcharge	double	-	-

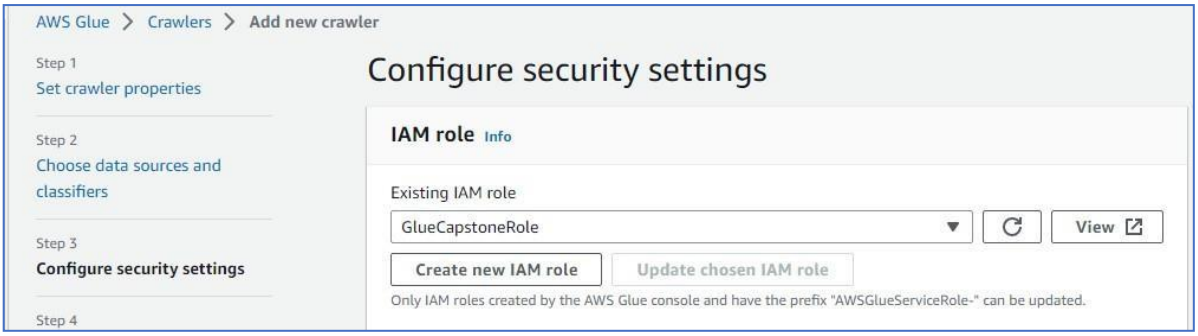
13. Now let's create the crawler for reference layer. Give the name to crawler as **crawler-reference-layer-<yourname>**



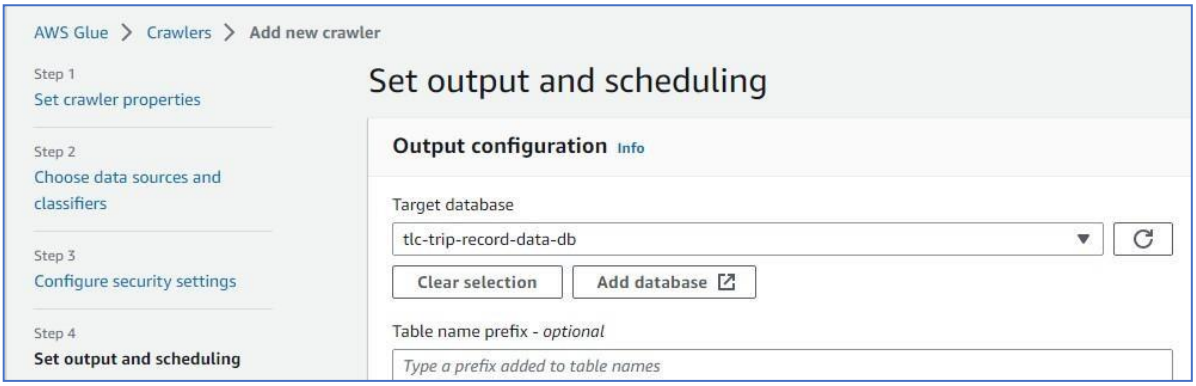
14. Choose S3 as data source and give path as ***s3://tlc-trip-record-data-lake-<yourname-random number>/reference-layer/***



15. Choose **GlueCapstoneRole** as an IAM Role.

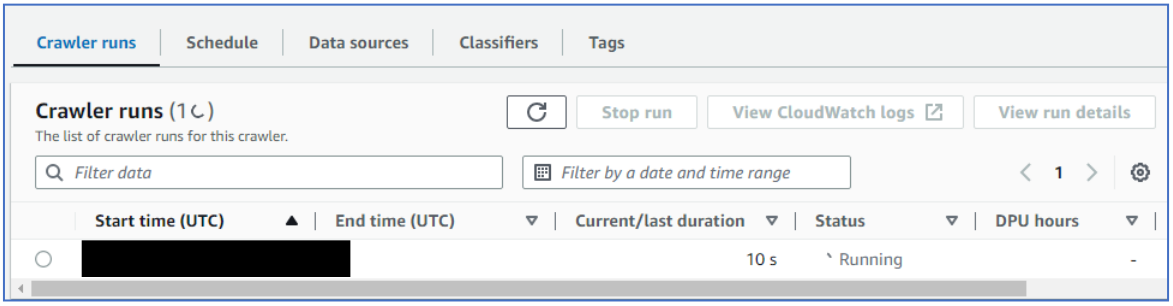


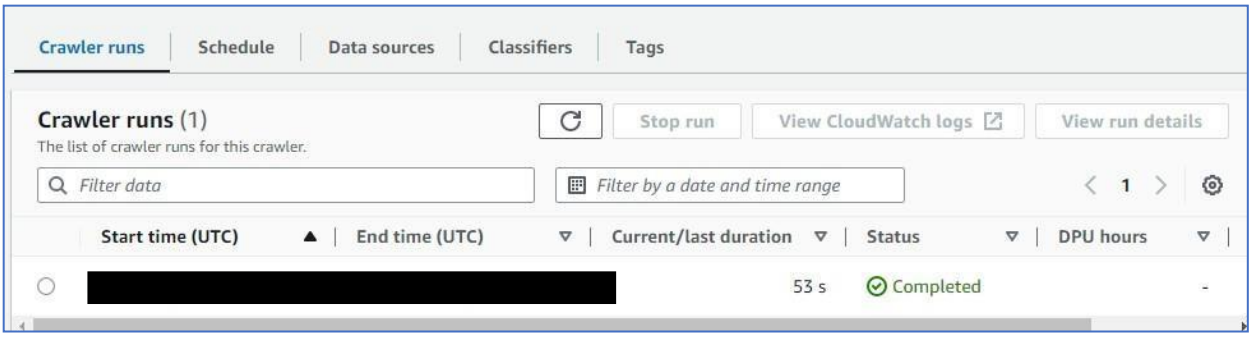
16. Select an existing database **tlc-trip-record-data-db-<yourname>** as target database.

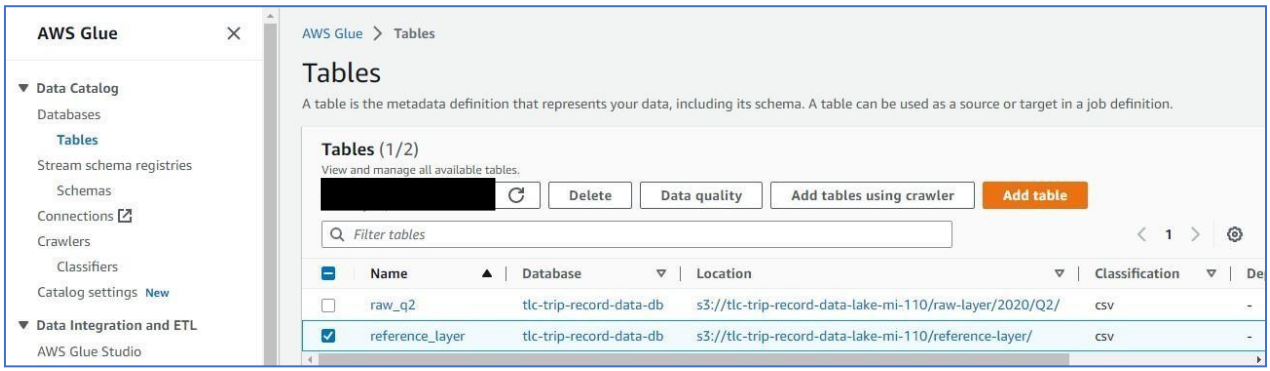


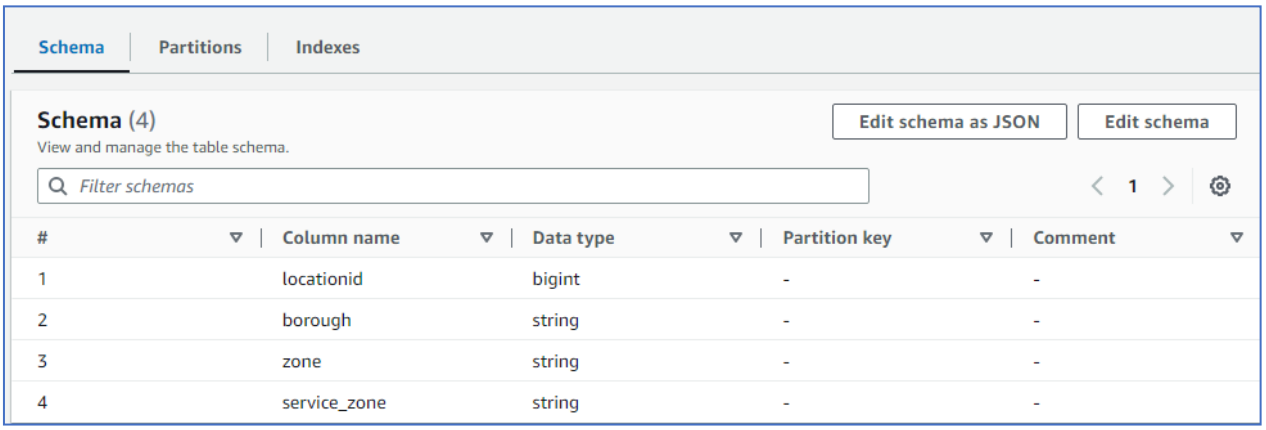
17. Keep crawler schedule frequency as On-Demand and create the crawler.

18. Once the crawler for reference layer is created, run it and check one table for reference layer is being created and explore the metadata.



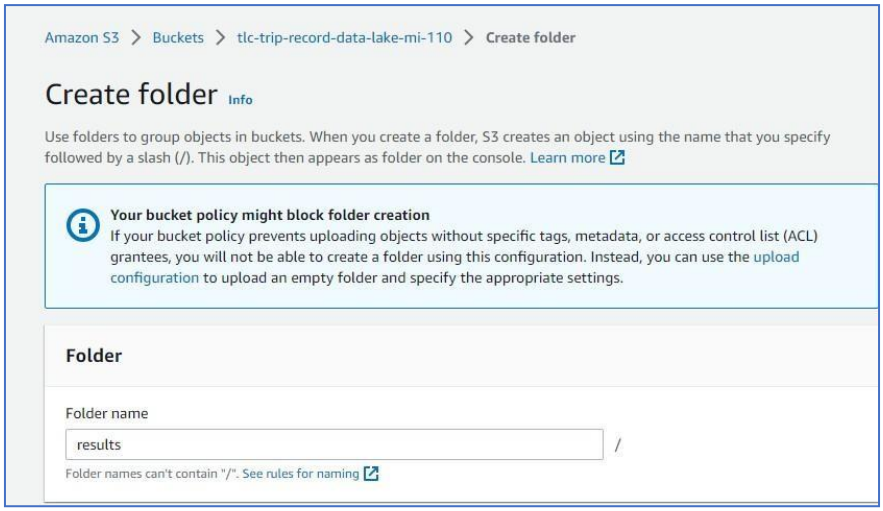

Crawler runs
Schedule
Data sources
Classifiers
Tags
Crawler runs (1)
The list of crawler runs for this crawler.
Stop run
View CloudWatch logs
View run details
Filter data
Filter by a date and time range
1
Start time (UTC)
End time (UTC)
Current/last duration
Status
DPU hours
[redacted]
53 s
Completed
-


AWS Glue
Data Catalog
Databases
Tables
Stream schema registries
Schemas
Connections
Crawlers
Classifiers
Catalog settings New
Data Integration and ETL
AWS Glue Studio
AWS Glue > Tables
Tables
A table is the metadata definition that represents your data, including its schema. A table can be used as a source or target in a job definition.
Tables (1/2)
View and manage all available tables.
[redacted]
Delete
Data quality
Add tables using crawler
Add table
Filter tables
1
Name
Database
Location
Classification
raw_q2
tlc-trip-record-data-db
s3://tlc-trip-record-data-lake-mi-110/raw-layer/2020/Q2/
csv
reference_layer
tlc-trip-record-data-db
s3://tlc-trip-record-data-lake-mi-110/reference-layer/
csv

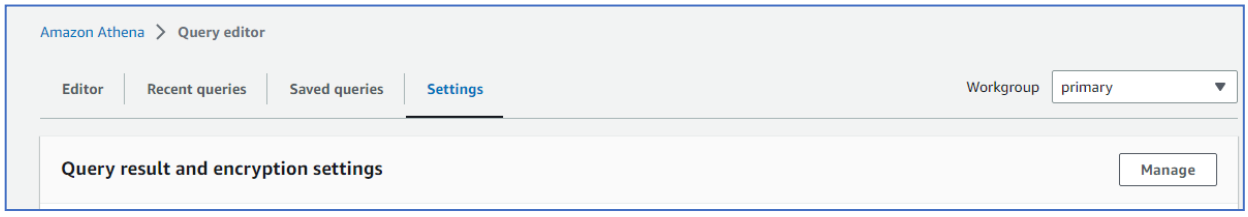

Schema
Partitions
Indexes
Schema (4)
View and manage the table schema.
Edit schema as JSON
Edit schema
Filter schemas
1
| Column name | Data type | Partition key | Comment
1 | locationid | bigint | - | -
2 | borough | string | - | -
3 | zone | string | - | -
4 | service_zone | string | - | -

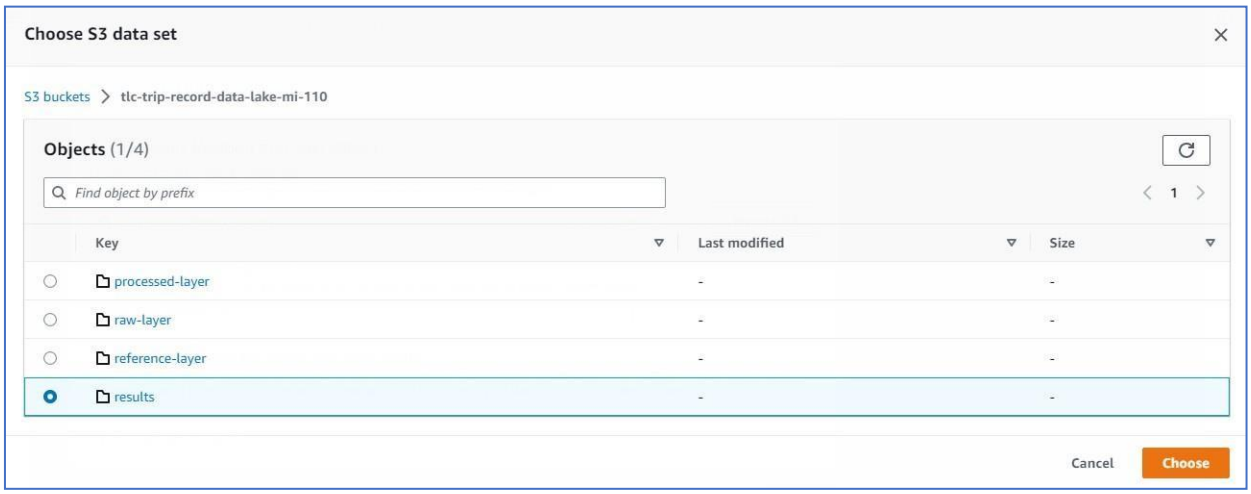
Phase-3: Ad-Hoc Exploration – SQL Analytics

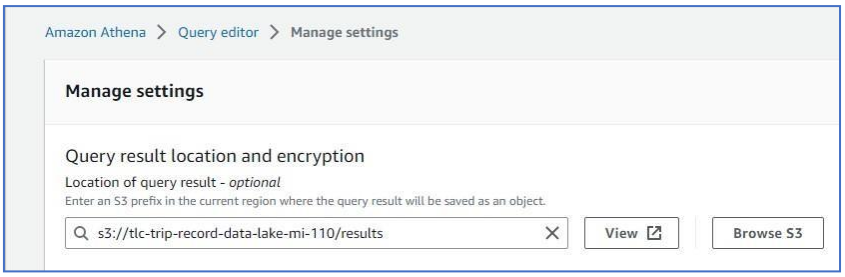
19. Go to your S3 bucket and create a new folder named as **results**.



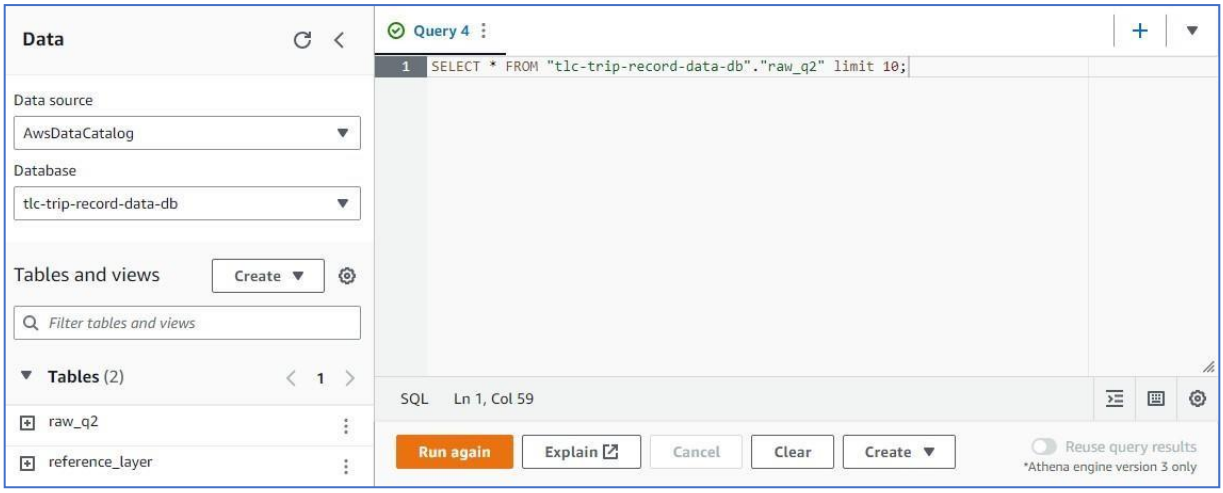
20. Go to **Athena>>Query Editor>>Settings>> Manage** and set the **Query Result location and encryption** as **s3://tlc-trip-record-data-<yourname-randomnumber>/results** and save.

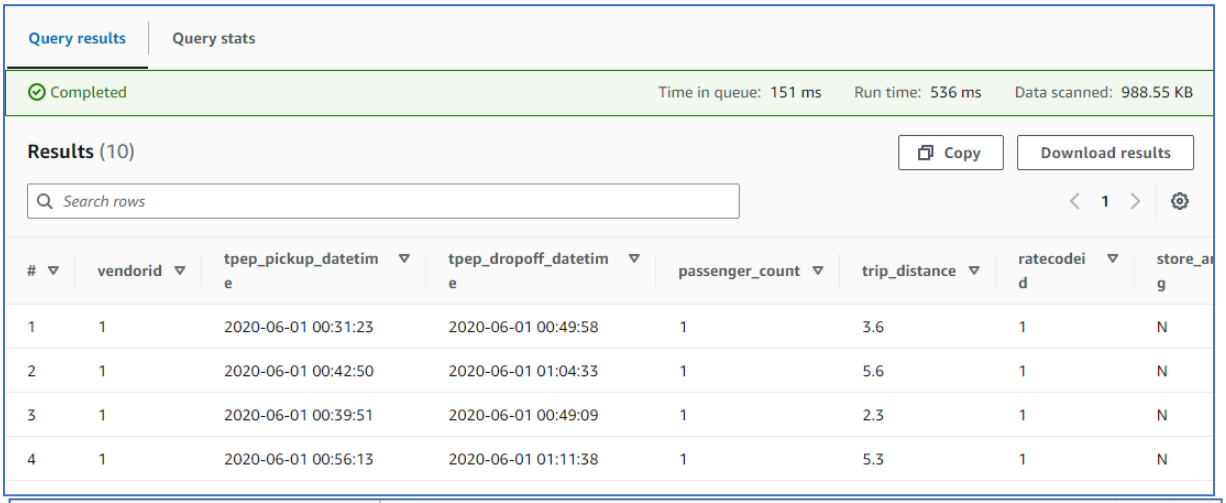




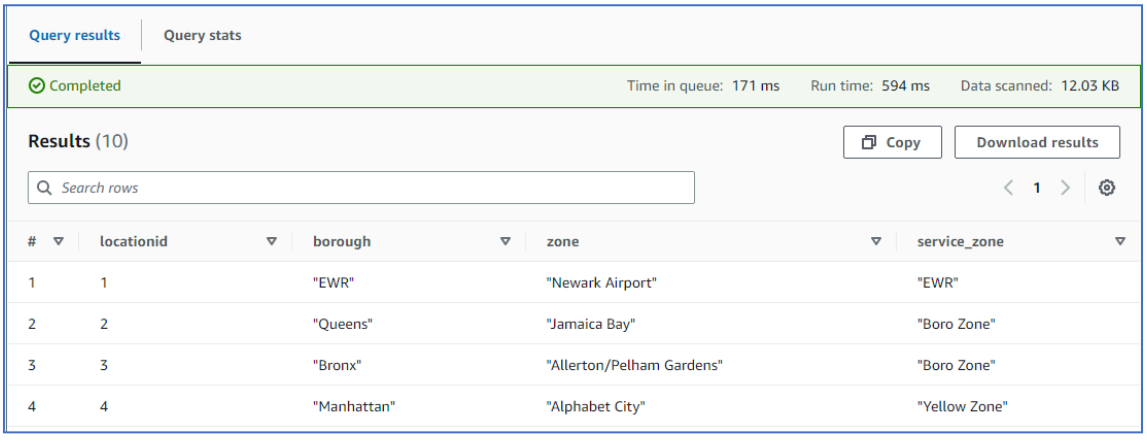


21. Now perform **ad-hoc SQL Analytics on raw table and reference table**.



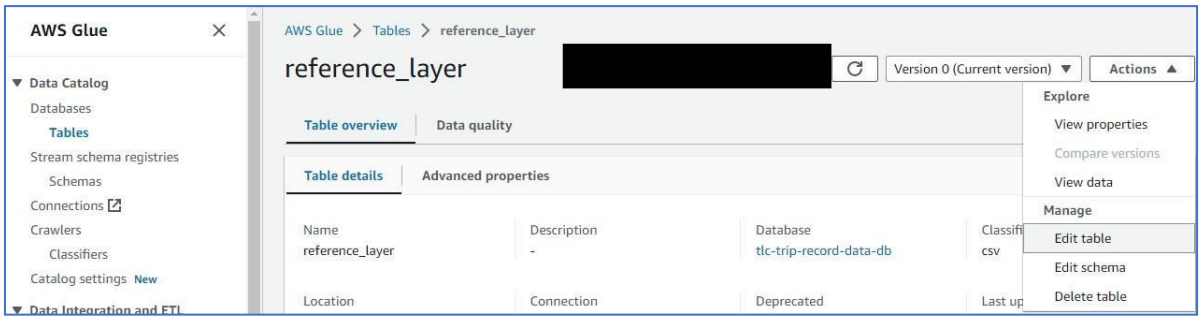


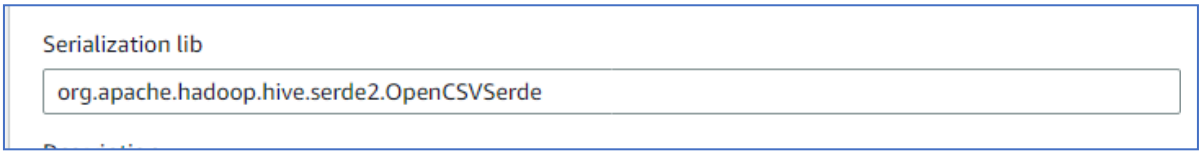
#	vendorid	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	ratecodeid	store_and_fwd_flag
1	1	2020-06-01 00:31:23	2020-06-01 00:49:58	1	3.6	1	N
2	1	2020-06-01 00:42:50	2020-06-01 01:04:33	1	5.6	1	N
3	1	2020-06-01 00:39:51	2020-06-01 00:49:09	1	2.3	1	N
4	1	2020-06-01 00:56:13	2020-06-01 01:11:38	1	5.3	1	N



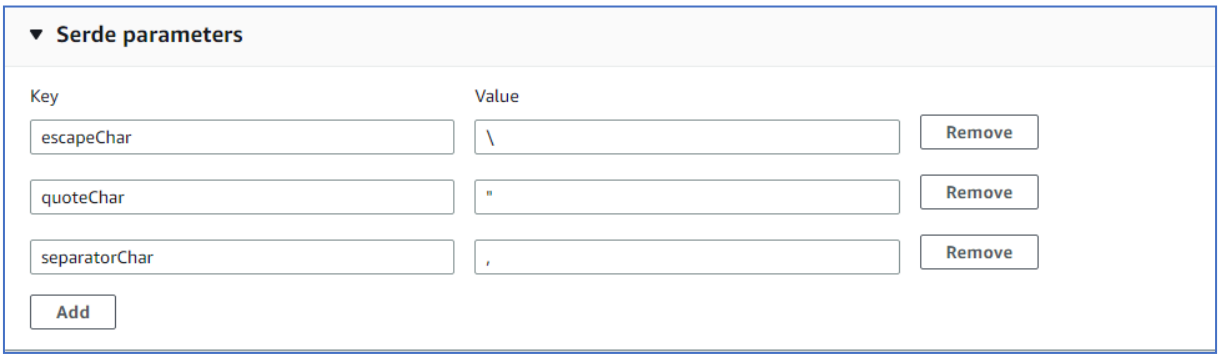
#	locationid	borough	zone	service_zone
1	1	"EWR"	"Newark Airport"	"EWR"
2	2	"Queens"	"Jamaica Bay"	"Boro Zone"
3	3	"Bronx"	"Allerton/Pelham Gardens"	"Boro Zone"
4	4	"Manhattan"	"Alphabet City"	"Yellow Zone"

22. Now let's get rid of these double quotes you saw in last query execution for reference data. Go to **AWS Glue >> Tables >> your reference layer table >> Actions >> Edit table >>** Under **Serialization lib**: replace the value with **org.apache.hadoop.hive.serde2.OpenCSVSerde**

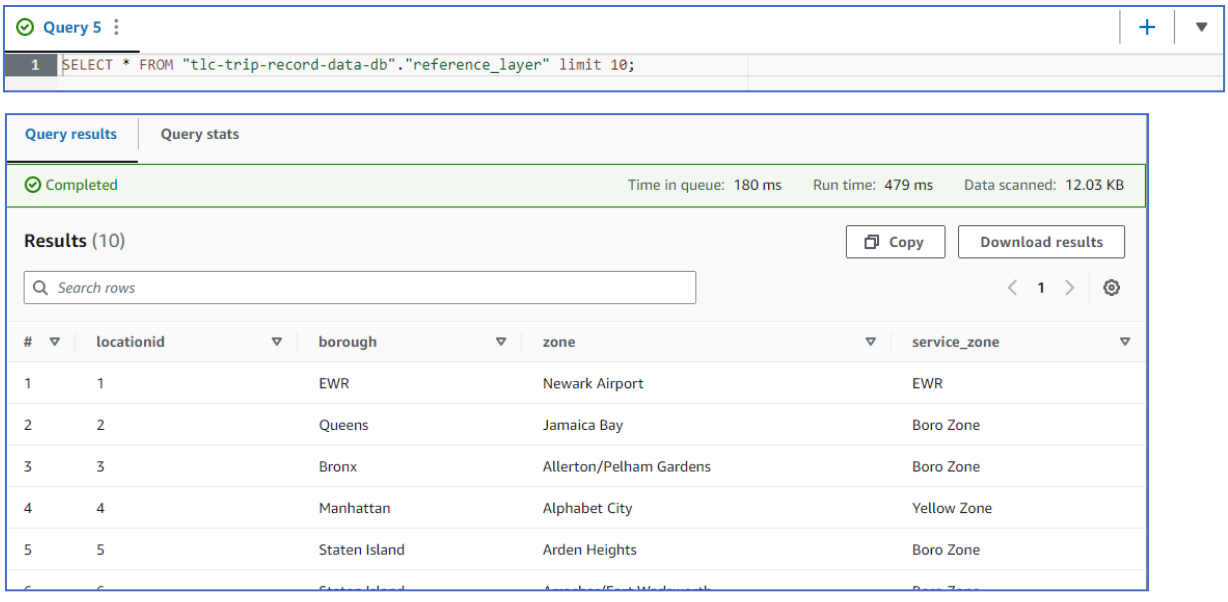




23. And update the **Serde parameters** as follows:

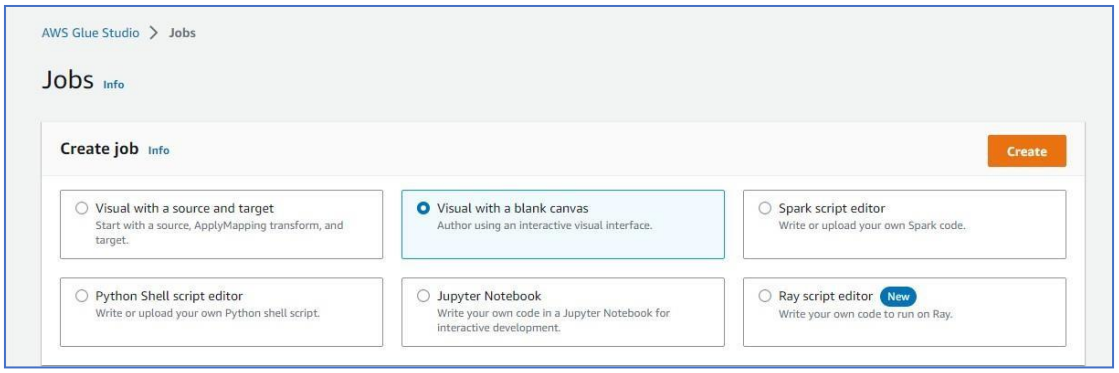


24. Now go back to **Athena and query your reference table and see the results**.



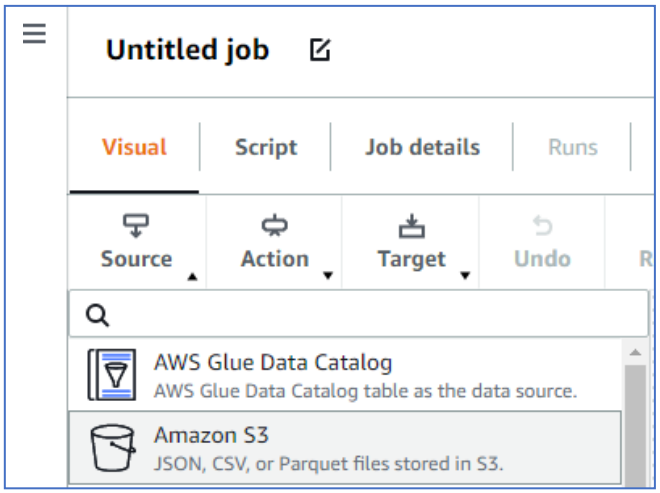
Phase-4: Processing - Let's Transform Taxi Ride Data

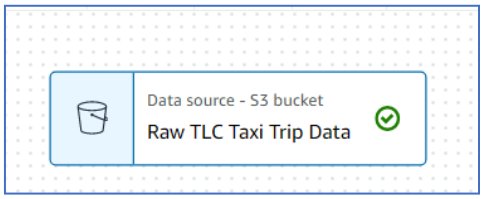
25. Go to **AWS Glue Studio >> Jobs >> Visual with a blank canvas >> Create**



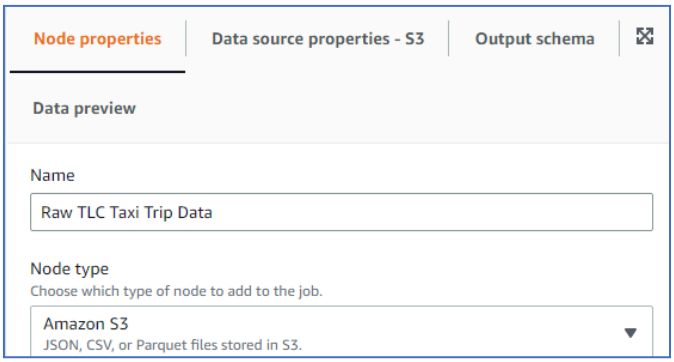
26. Now, let's start building our ETL workflow.

Select **Source>>S3**.

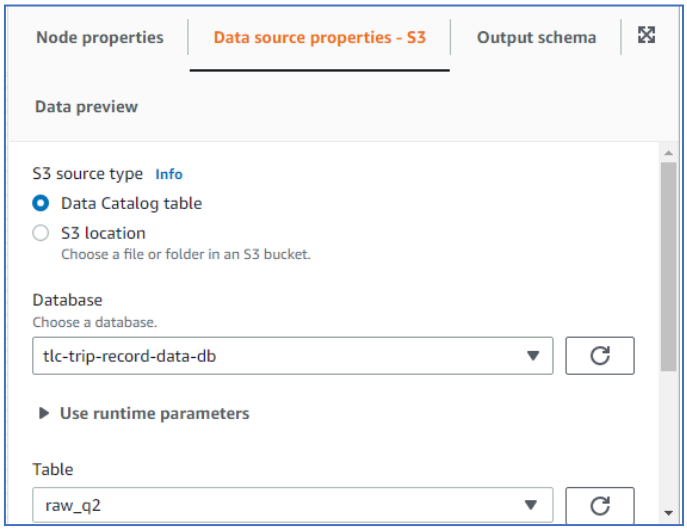




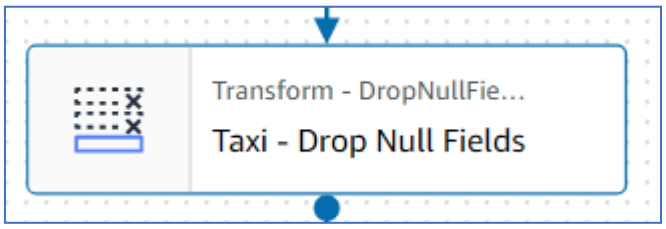
Node properties >> Name > Raw TLC Taxi Trip Data



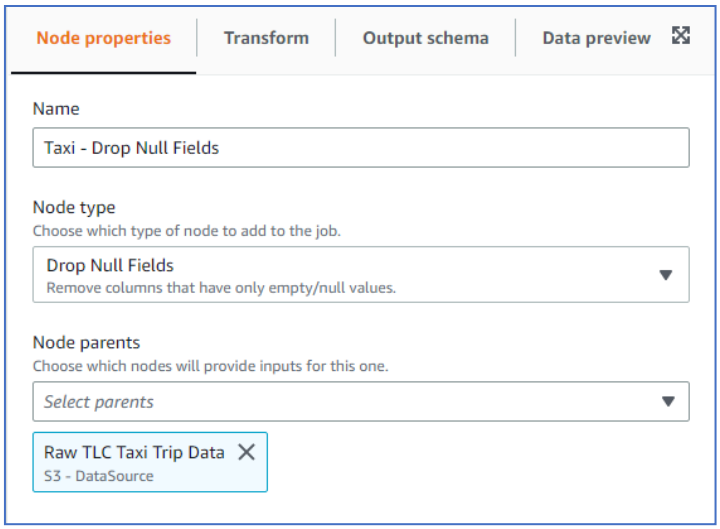
Data Source Properties – S3: Select **Data Catalog table** as **S3 source type**. Select **your database** that you created earlier in **Glue Data catalog**. Select **raw table** you got earlier.



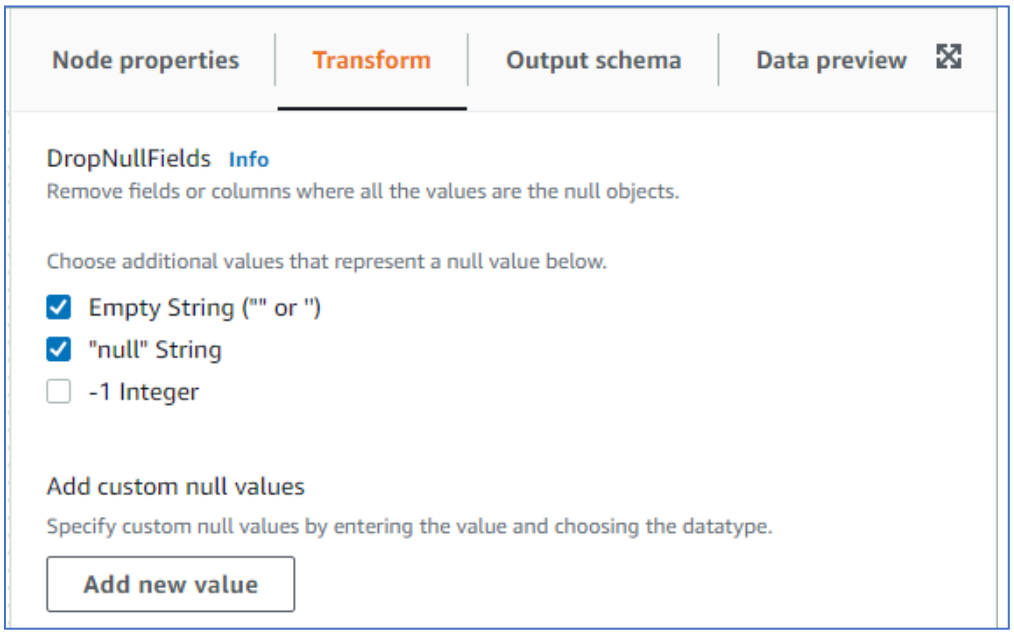
27. Now, let's add transformation after source node. From **actions >> choose DropNullFields**



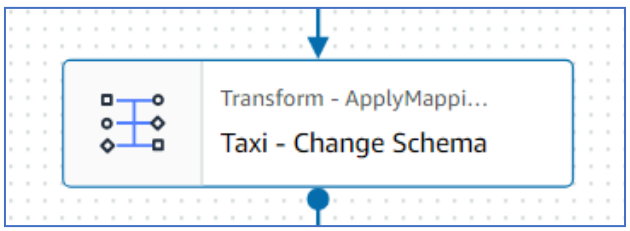
Node properties: Name>>Taxi – Drop Null Fields



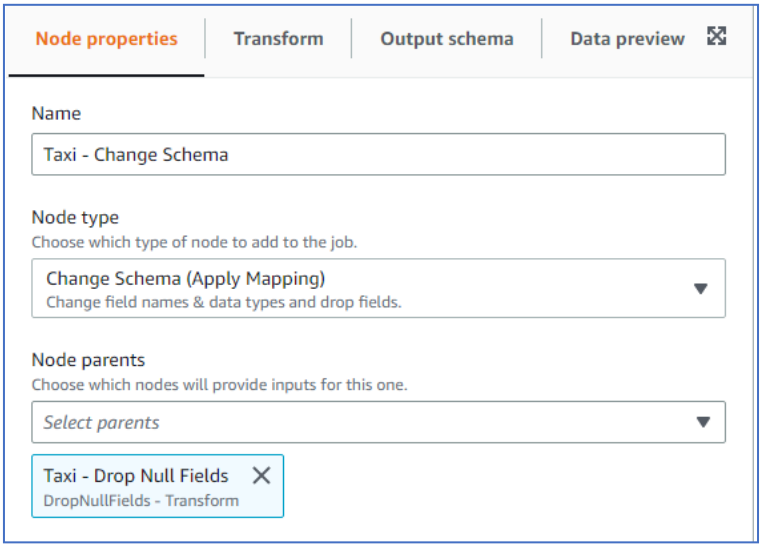
Transform: Select Empty String (“” or ‘’) and “null” String



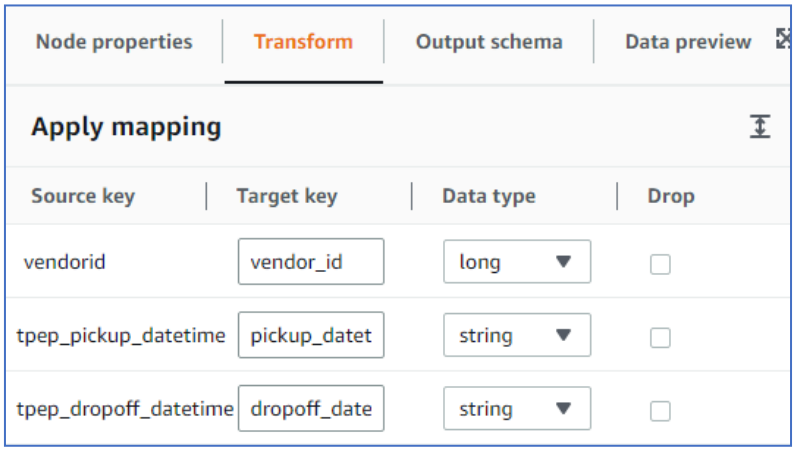
28. Add next transform, **actions >> ApplyMapping.**



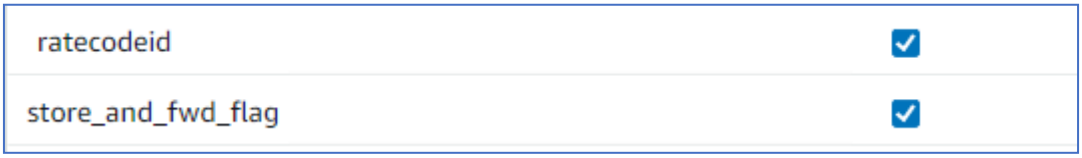
Node properties: Taxi – Change Schema



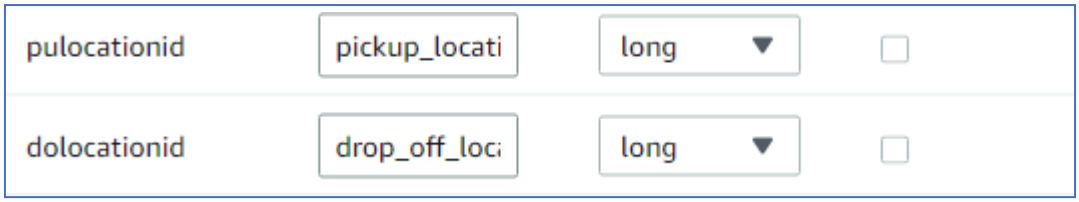
Transform:
Rename the columns as:
vendor_id, pickup_datetime, dropoff_datetime



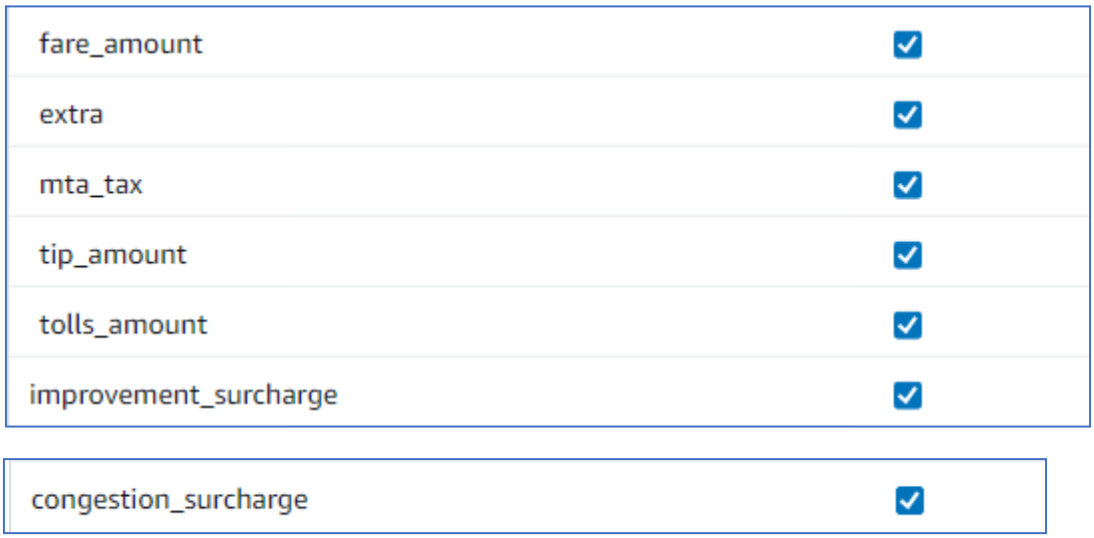
Drop ratecodeid and store_and_fwd_flag columns.



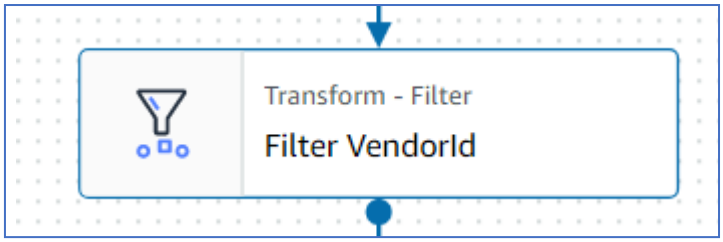
Rename: pickup_locationid and drop_off_locationid.



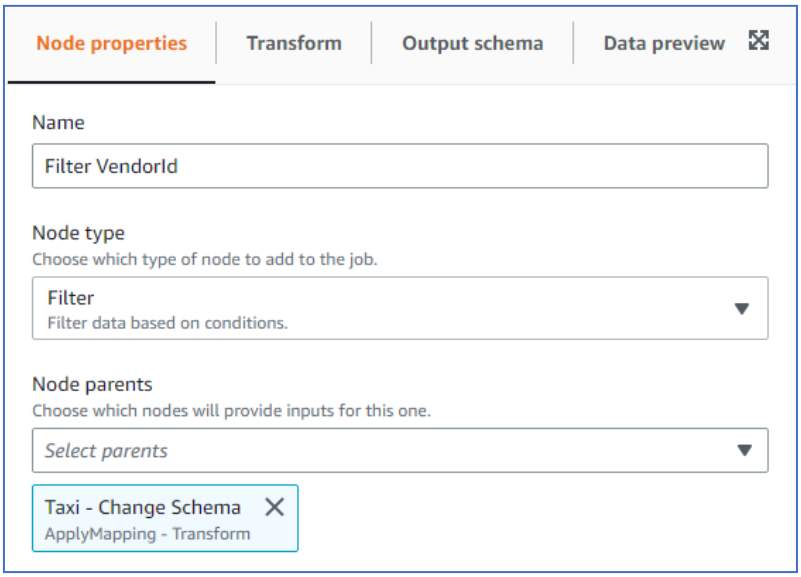
Drop the following columns.



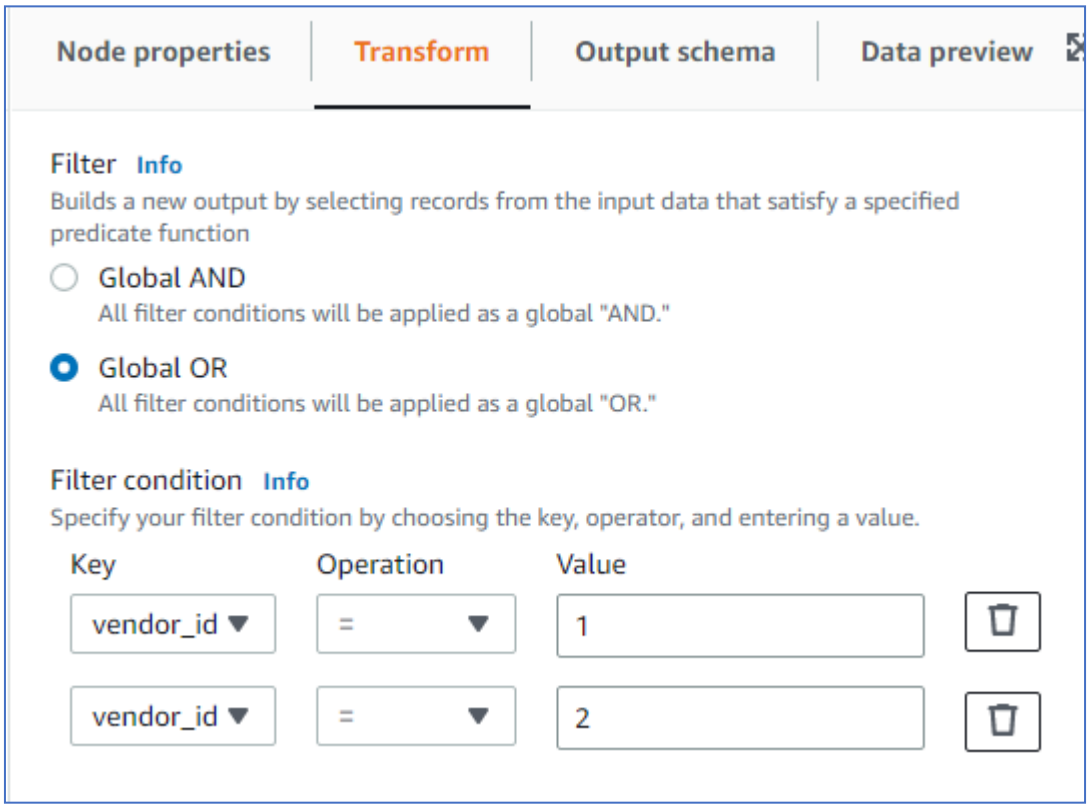
29. Let's add next transformation. **Actions>>Filter.**



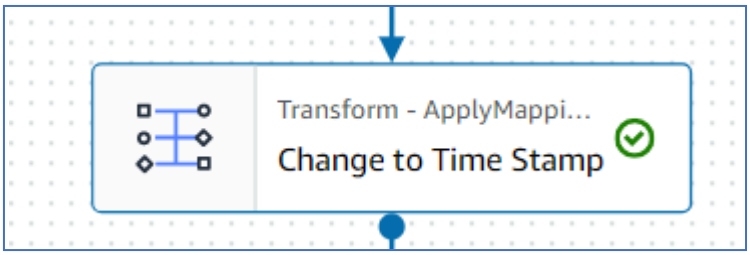
Node properties: Filter VendorId



Transform: Choose Global OR and in filter condition add filter condition as mentioned.



30. Add next transform. **Actions >> ApplyMapping.**

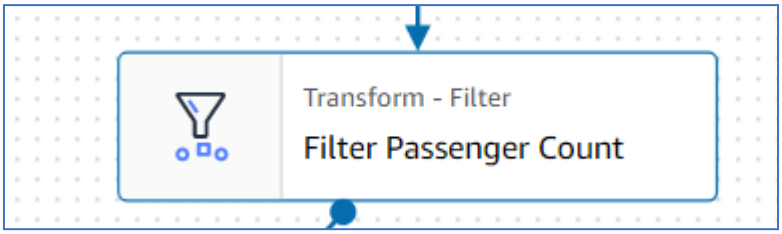


Node properties: Name as Change to Time Stamp

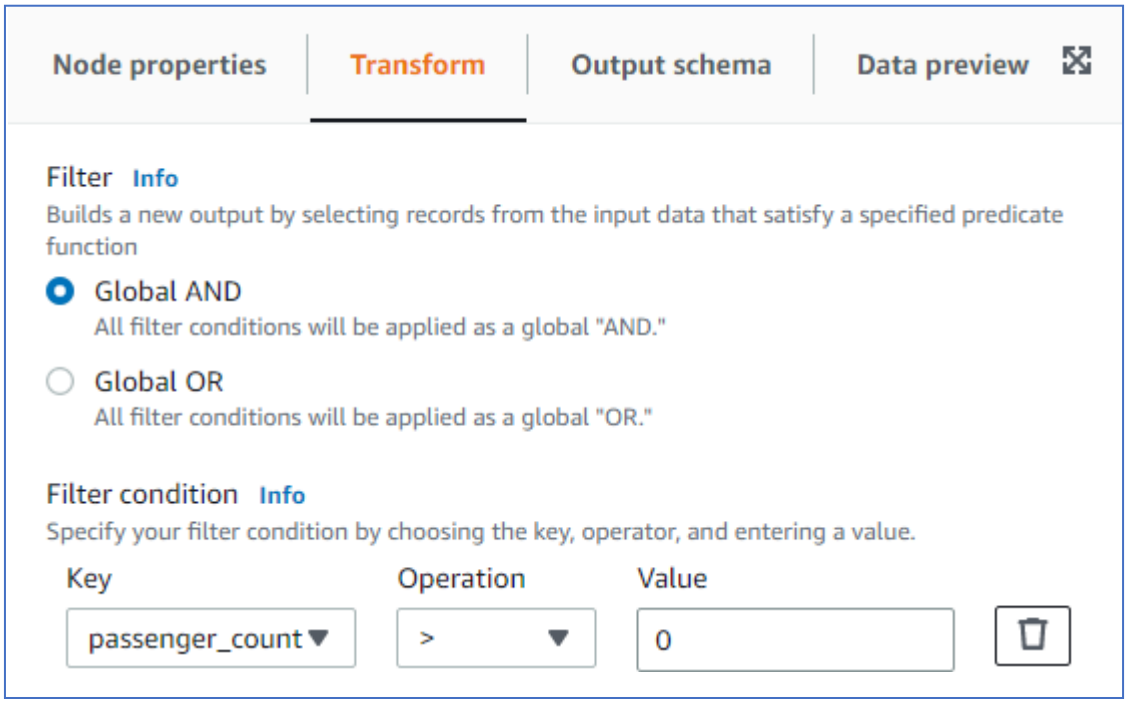
Transform: Change the data type of the two columns mentioned below to timestamp.

31. Add next transform. **Actions >> Filter.**

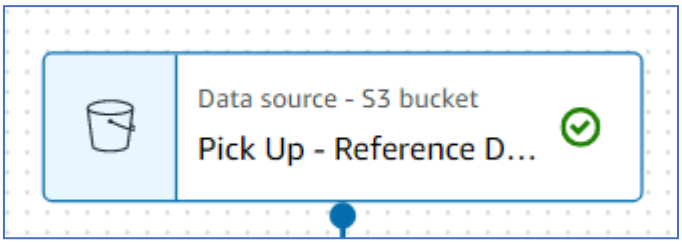
Node properties: Name as Filter Passenger Count



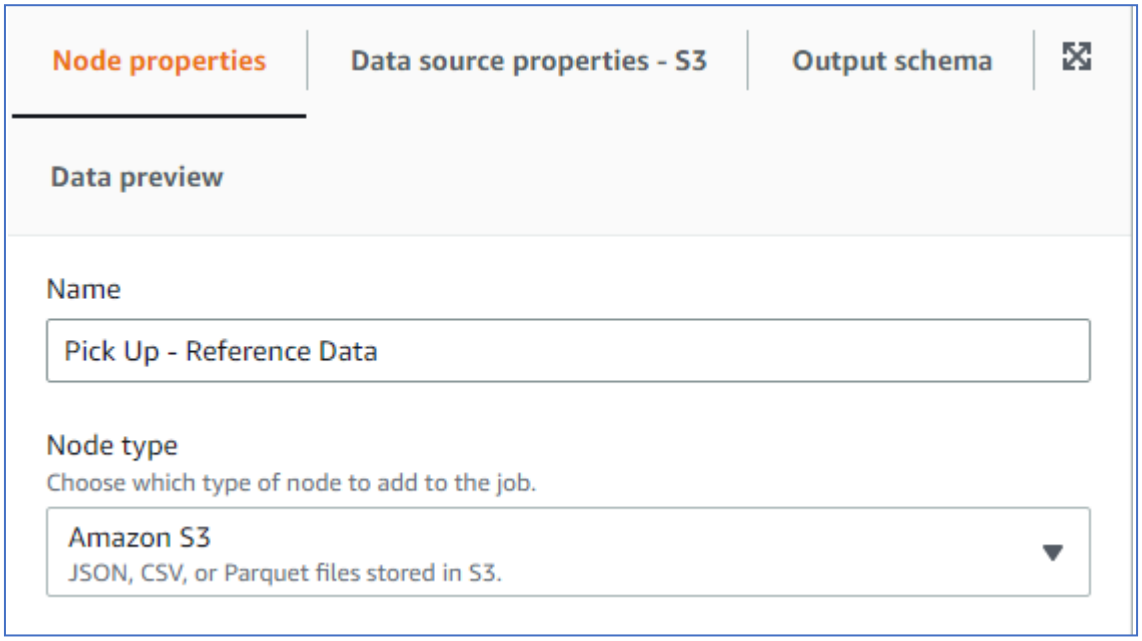
Transform: Select Global AND and add filter condition as shown below.



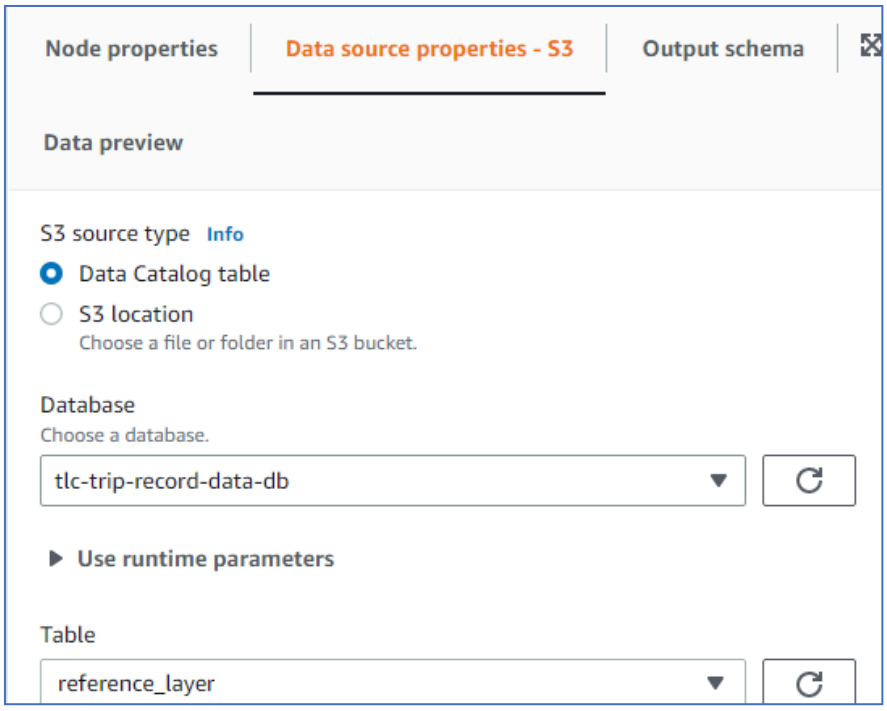
32. **Now click in the blank space on the canvas and add another S3 source.**



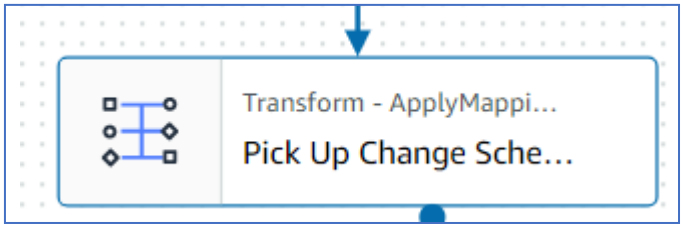
Node properties: Name as Pick Up - Reference Data



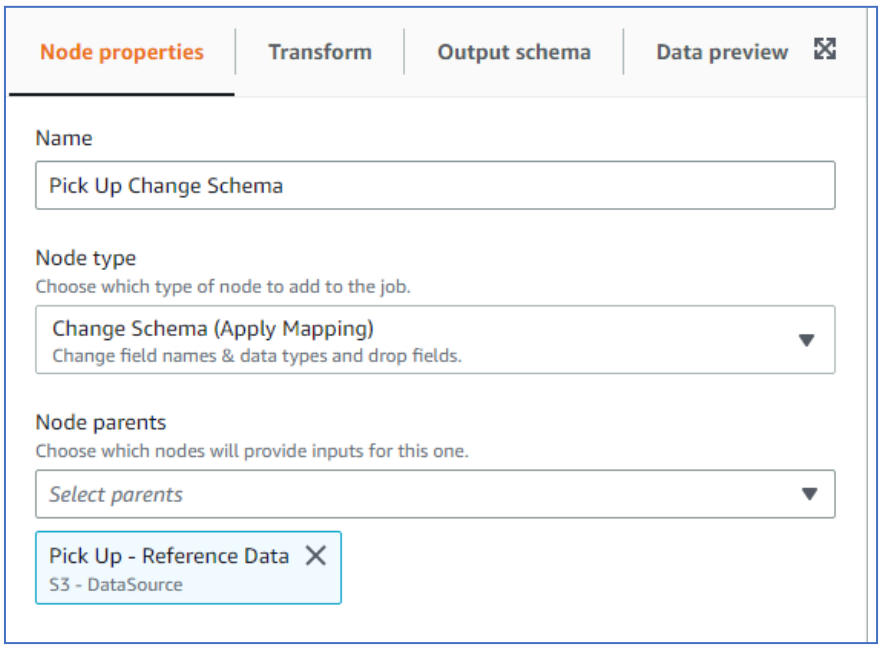
Data source properties: Choose Data Catalog table as S3 source. Choose your Glue Database you created earlier. Select reference table, this time.



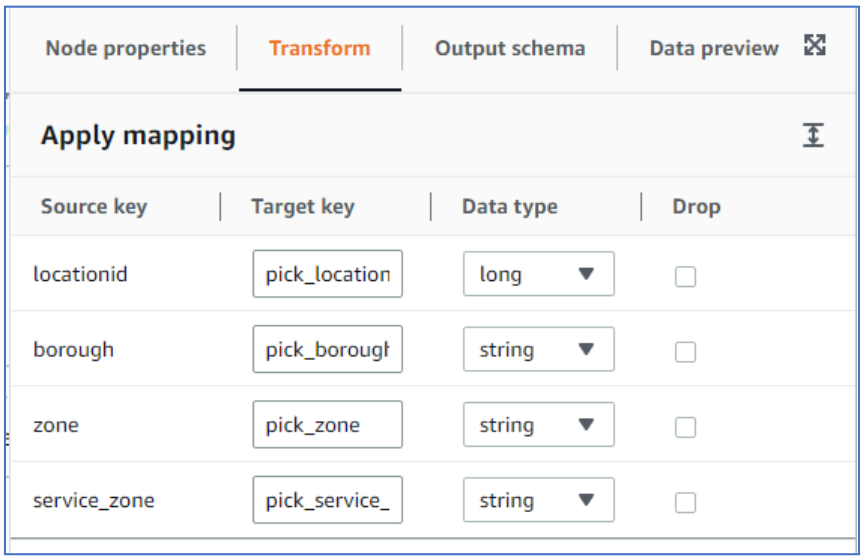
33. **Add an action next to your node “Pick Up – Reference Data” as Action >> ApplyMapping.**



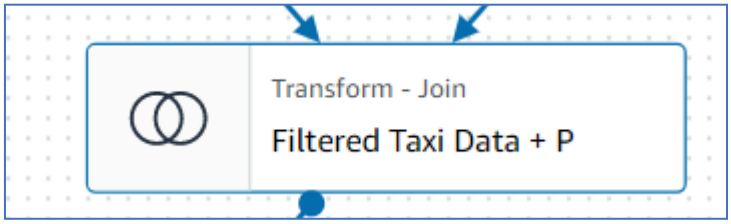
Node properties: Pick Up Change Schema



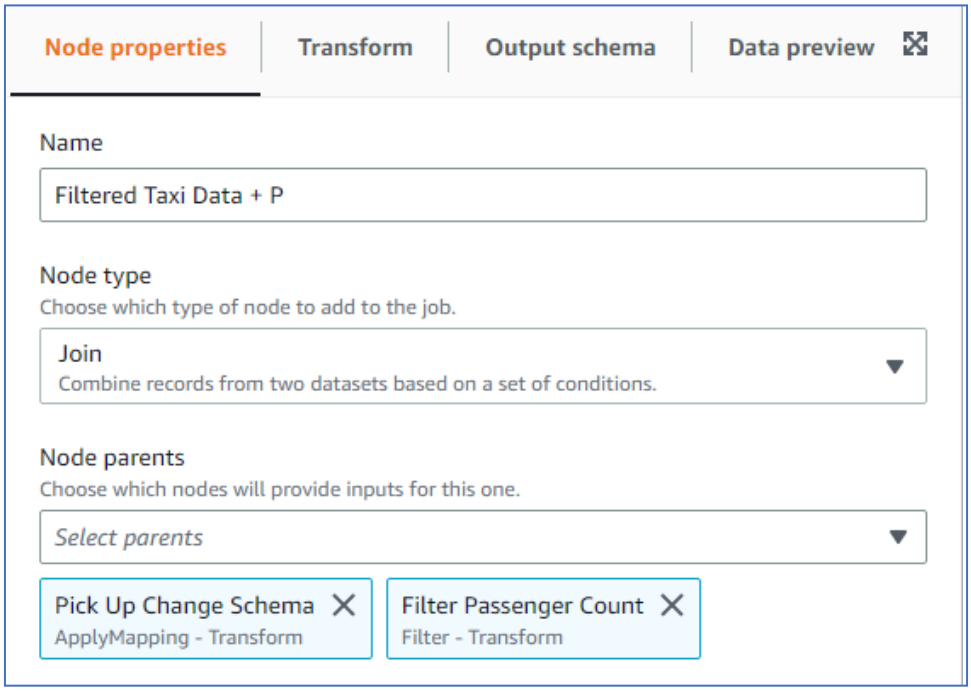
Transform: Rename the columns as follows: Append pick_ before every name in Target key.



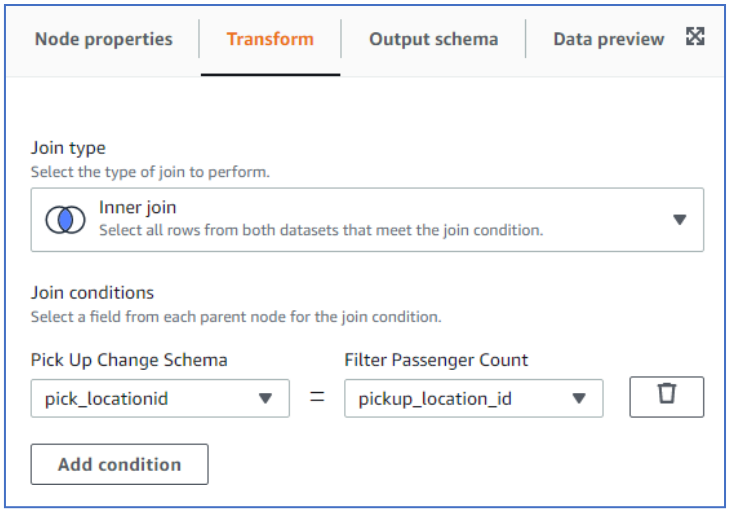
34. **Click in the blank space on canvas. Add Join transform.**



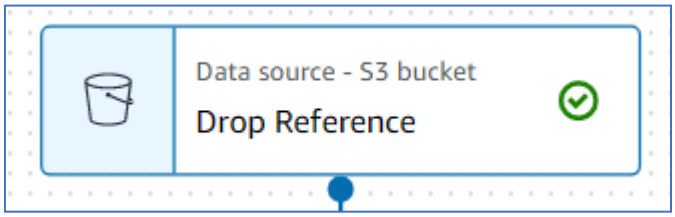
Node properties: Name as Filtered Taxi Data + P and select Pick Up Change Schema and Filter Passenger Count as node parents.



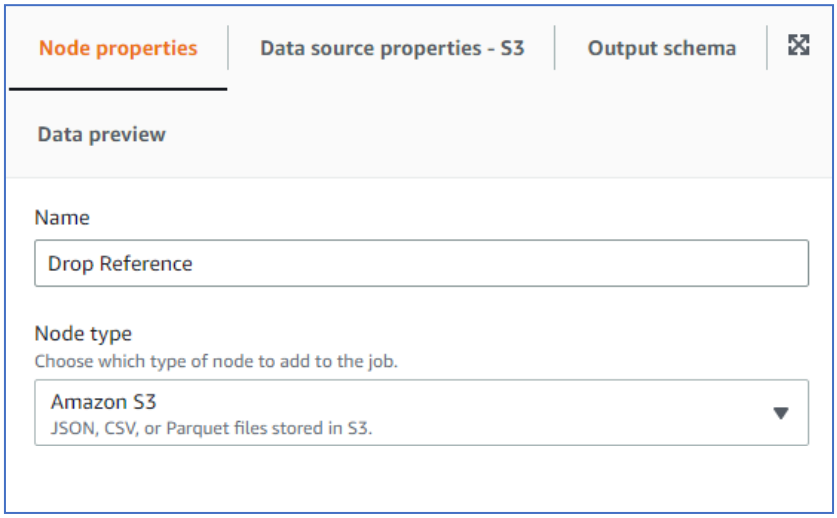
Transform: Apply join condition as follows.



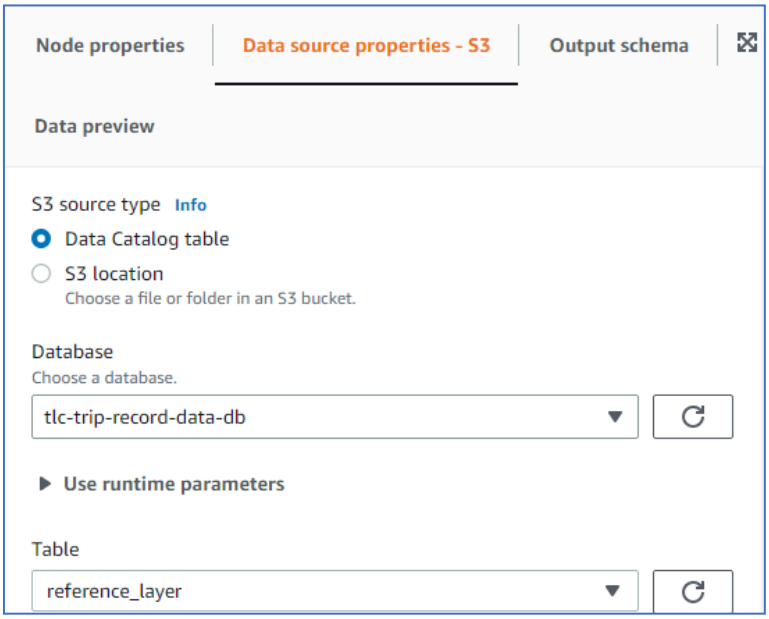
35. Click in blank space on canvas and one more S3 source.



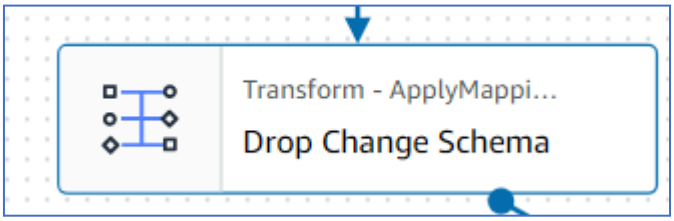
Node properties: Name as Drop Reference



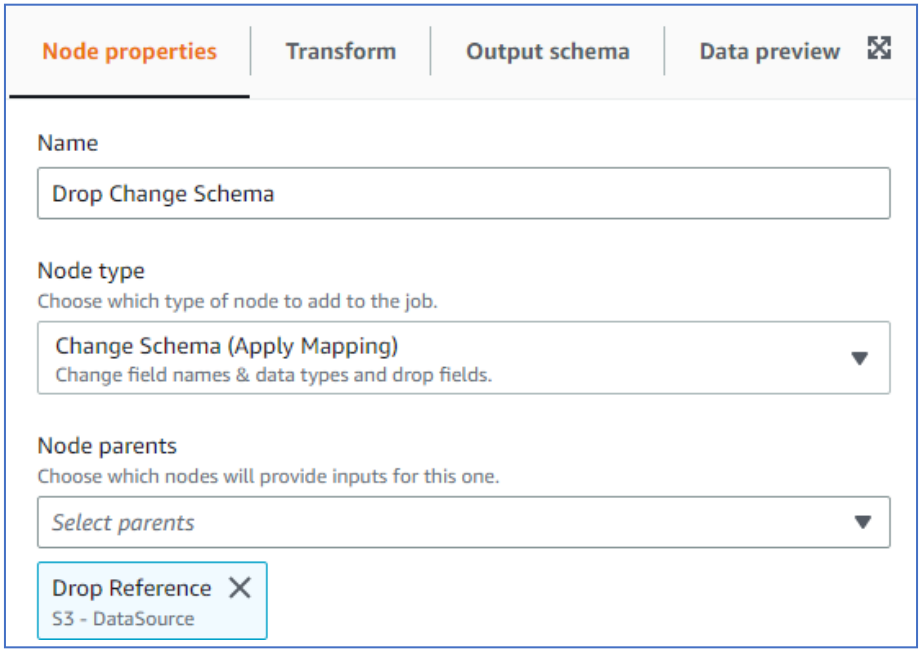
Data source: Select Data Catalog table as S3 source type. Select Glue Database created earlier and select reference table again.



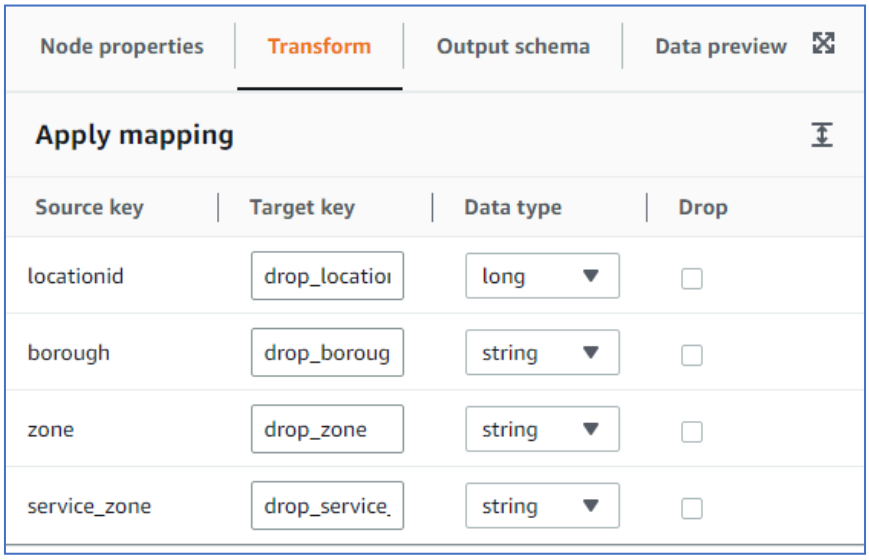
36. Add ApplyMapping transform to this new source.



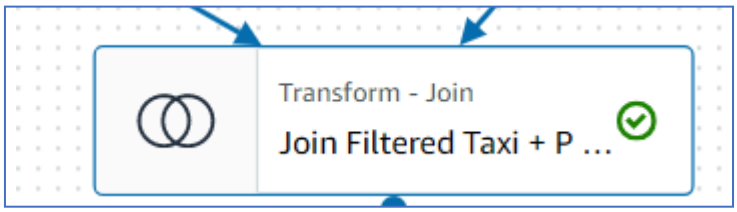
Node properties: Name as Drop Change Schema



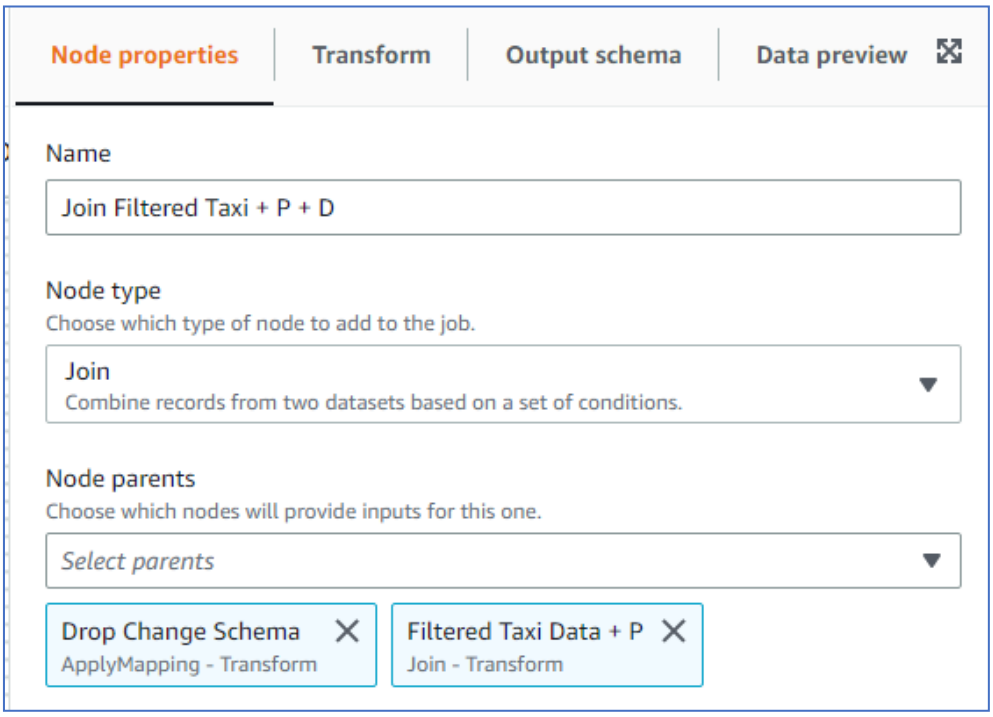
Transform: Simply append drop_ for each column in target key as shown below.



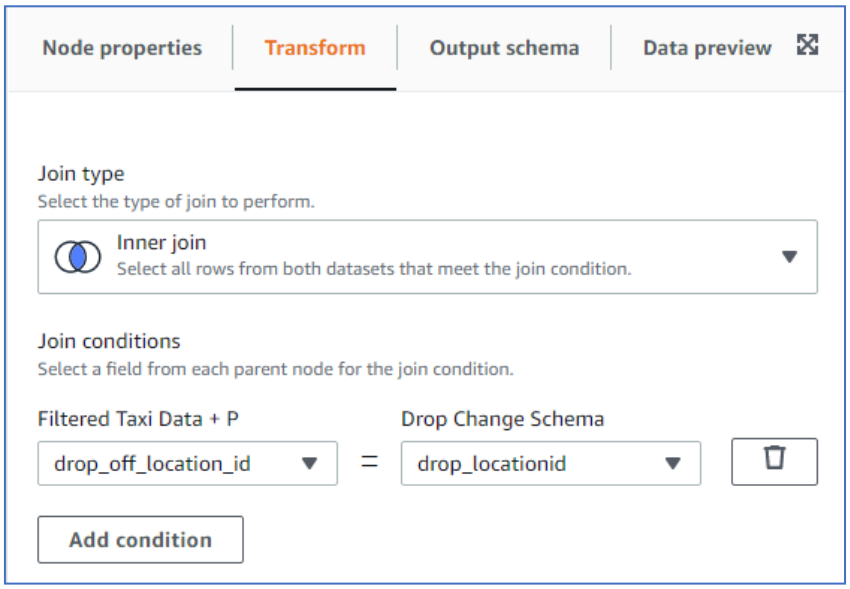
37. **Click in the blank space on the canvas. Add one more Join transform.**



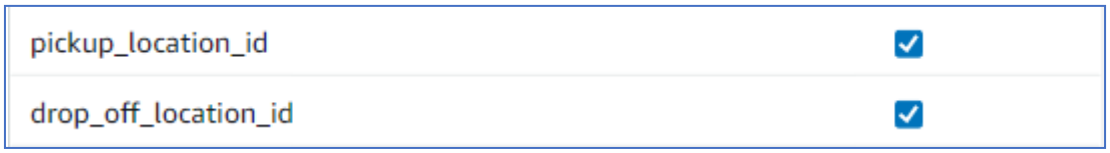
Node properties: Name as Join Filtered Taxi + P + D and select Drop Change Schema and Filtered Taxi Data + P as node parents.



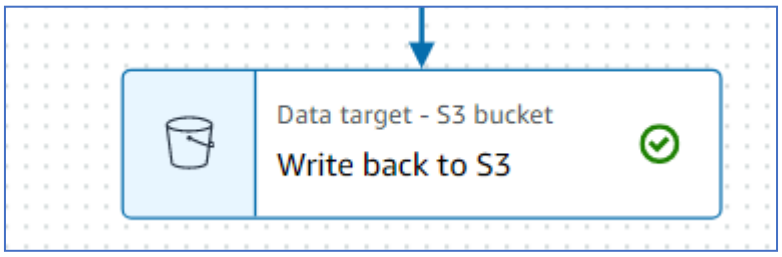
Transform: Use the join condition as shown below.



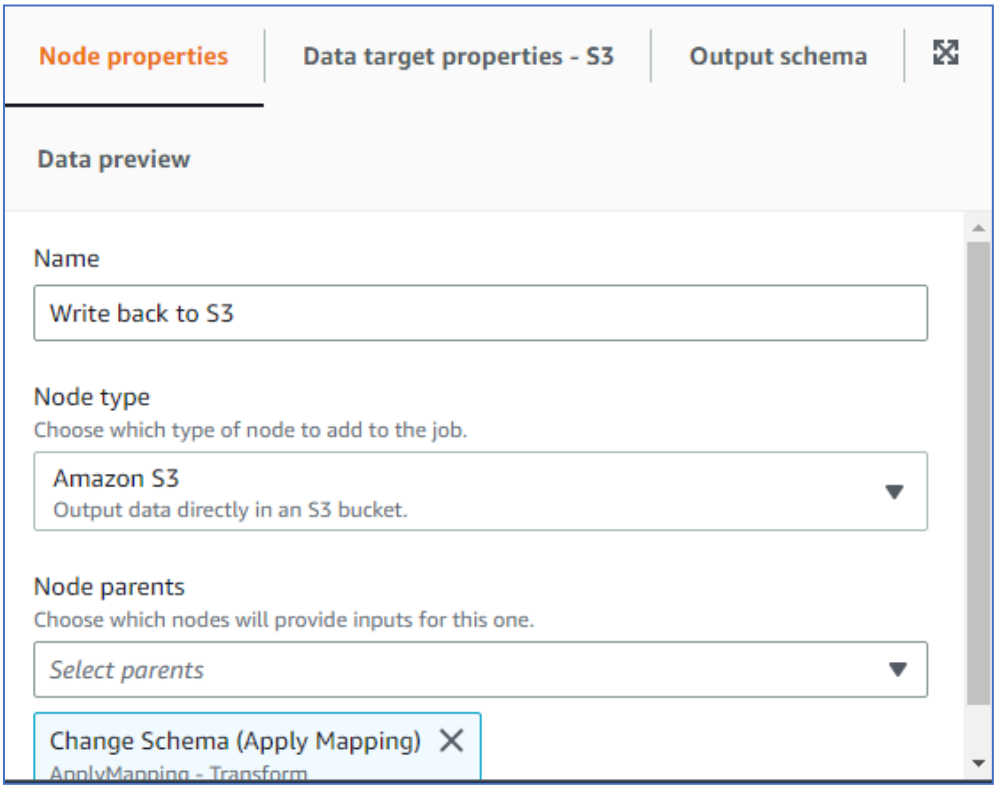
38. **Now after final joined node i.e. Join Filtered Taxi + P + D, add ApplyMapping and drop these two coloumns.**



39. **Last step, add target. Select S3.**

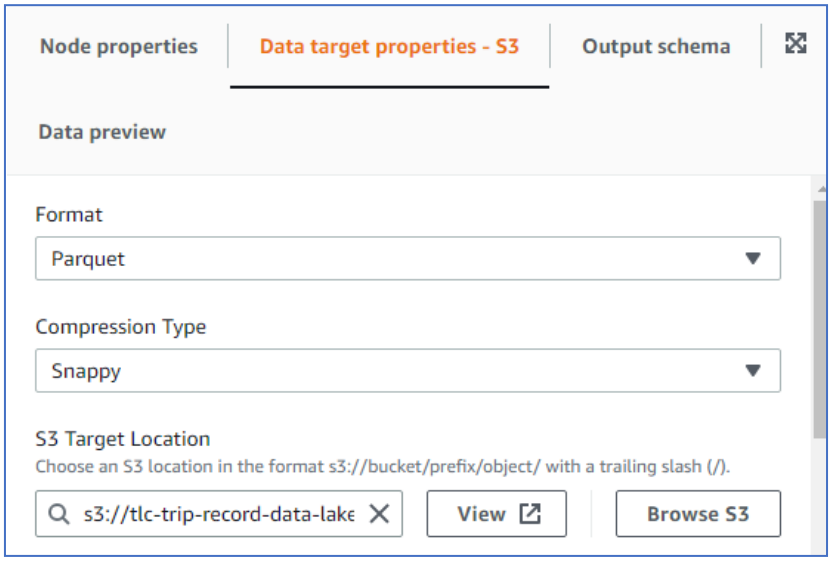


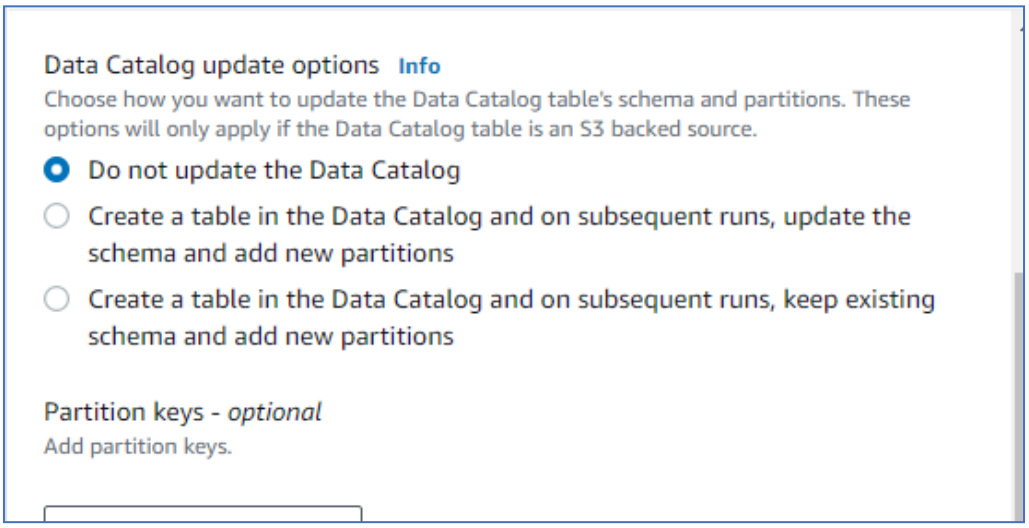
Node properties: Name as Write back to S3



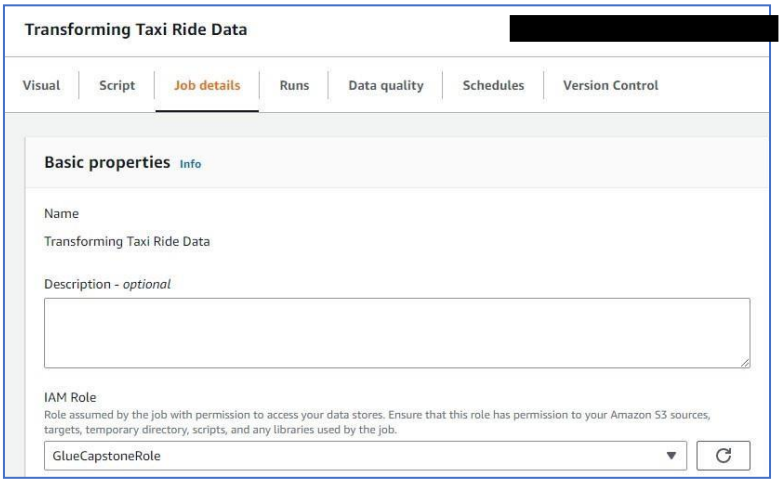
Data target: Format: Parquet, Compression: Snappy and target path as

s3://tlc-trip-record- data-lake-mi-<yourname-randownumber>/processed-layer/

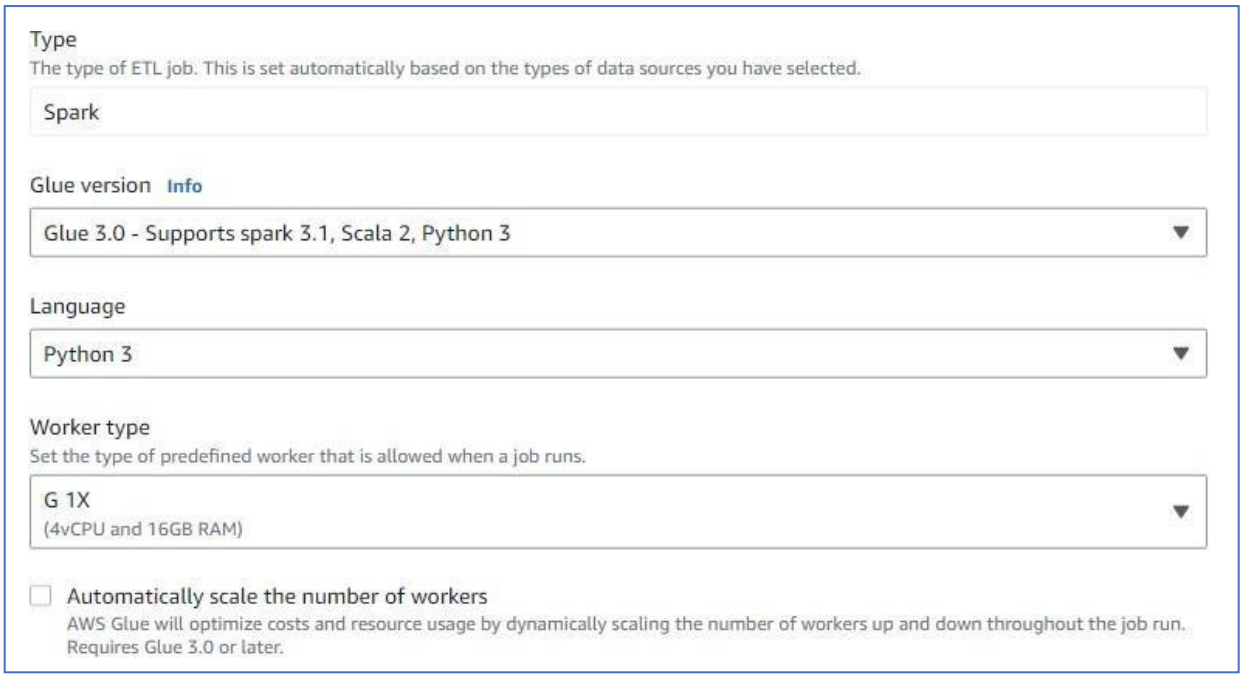




40. Go to **Job Details Tab: Name as Transforming Taxi Ride Data and select GlueCapstoneRole**



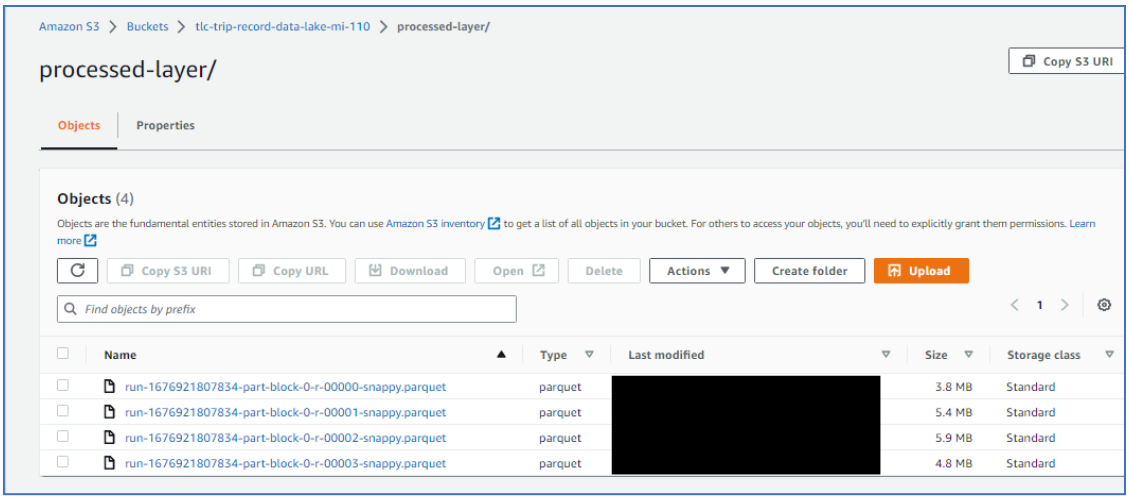
Make sure you go with the following configuration as shown below:



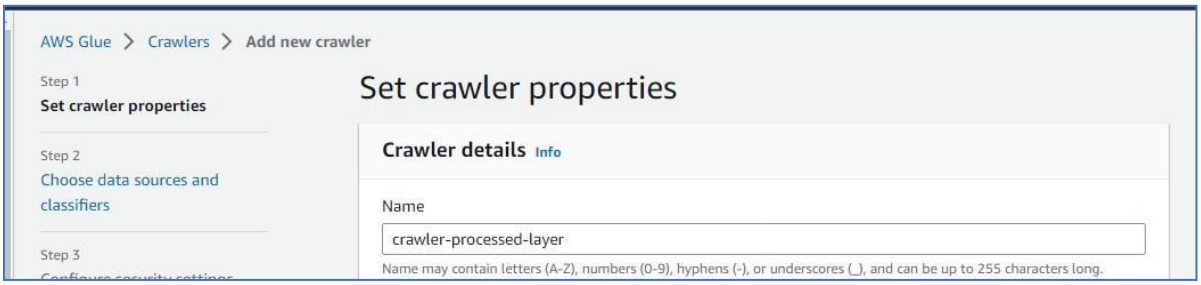
Give a unique name to your script. <unique name>.py

41. Save the job and run.

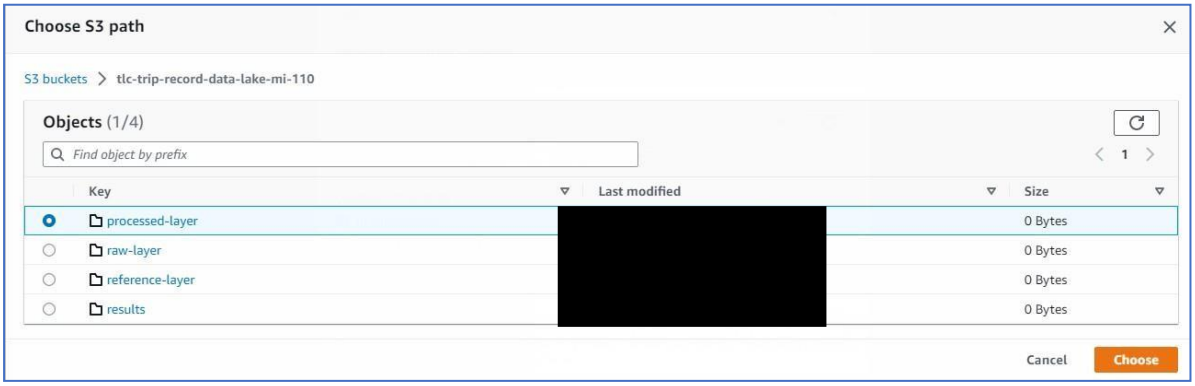
42. Watch for the output in processed layer in S3.

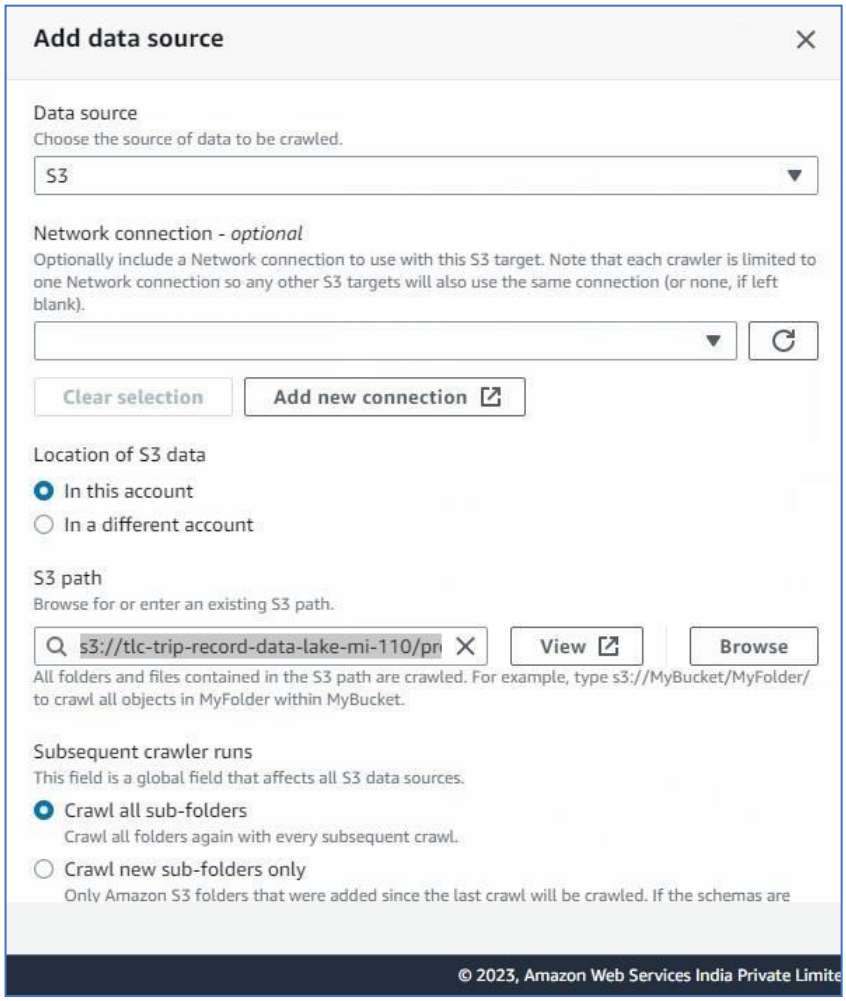


43. Go AWS Glue>> **Create a crawler for processed layer**. Name as **crawler-processed-layer-<yourname>**

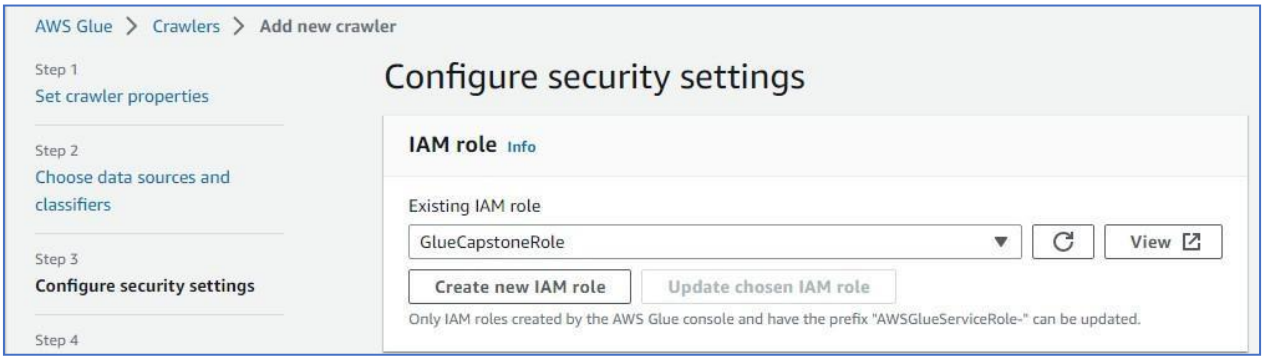


Choose S3 as data source.
Provide s3://tlc-trip-record-data-lake-<yourname-randomnumber>/processed-layer/ as S3 path

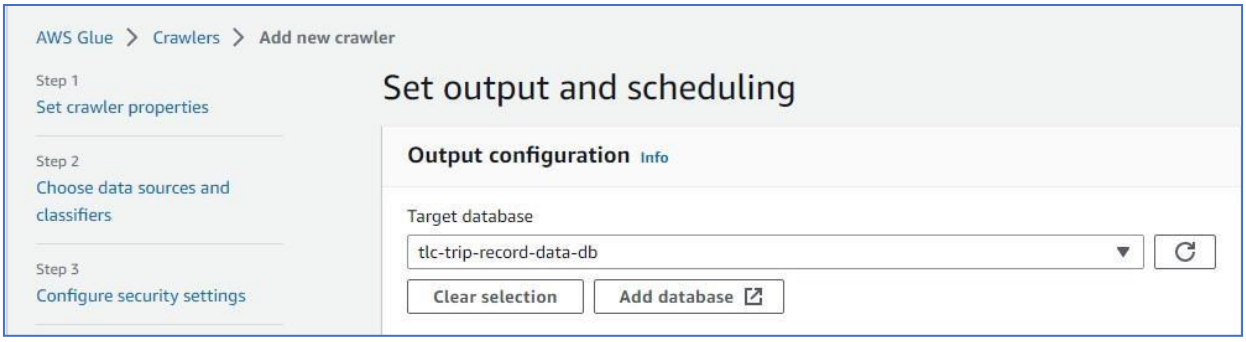




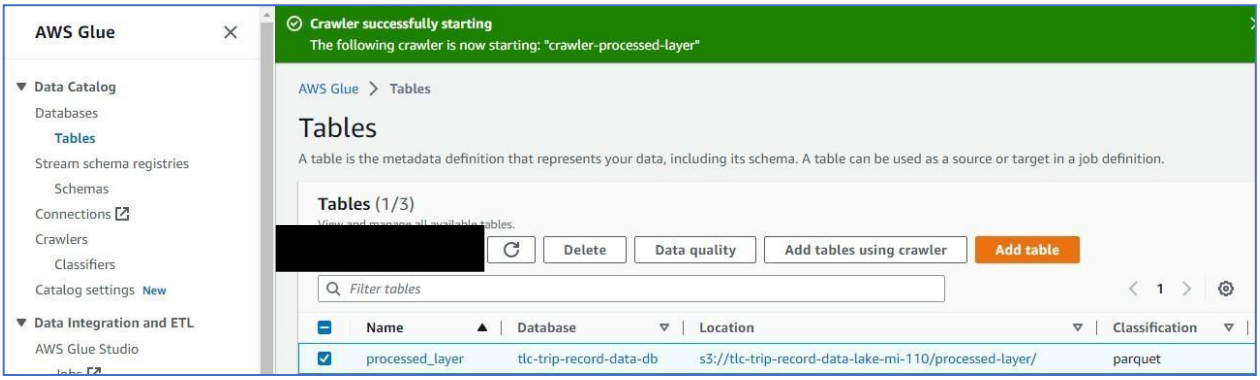
44. Select **GlueCapstone** as an IAM role.



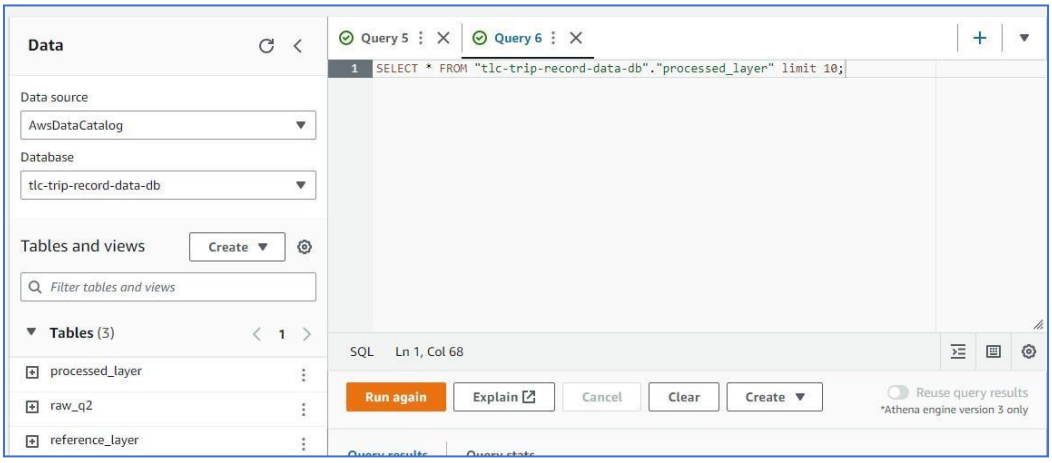
45. Select your existing Glue database as target database and create the crawler. Run it.

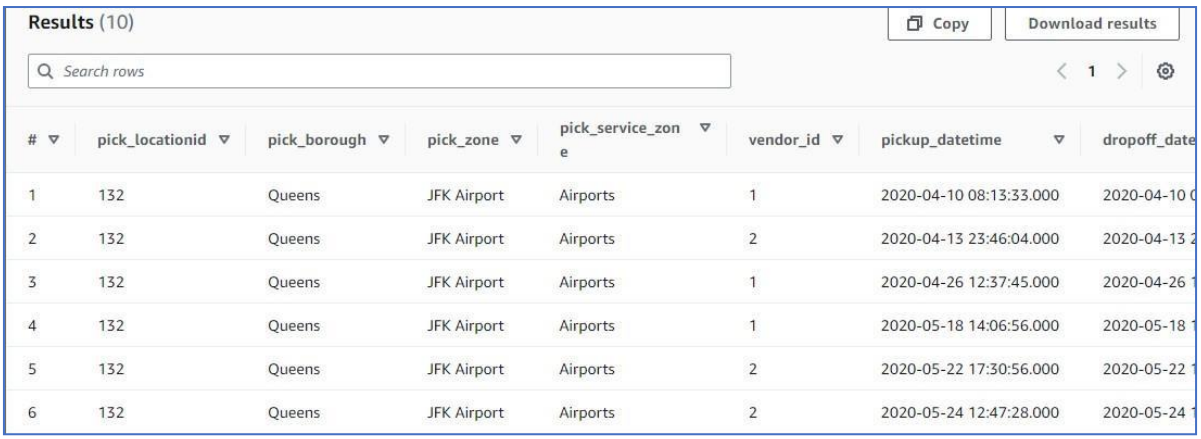


46. You will be able to see metadata table for processed layer as well.



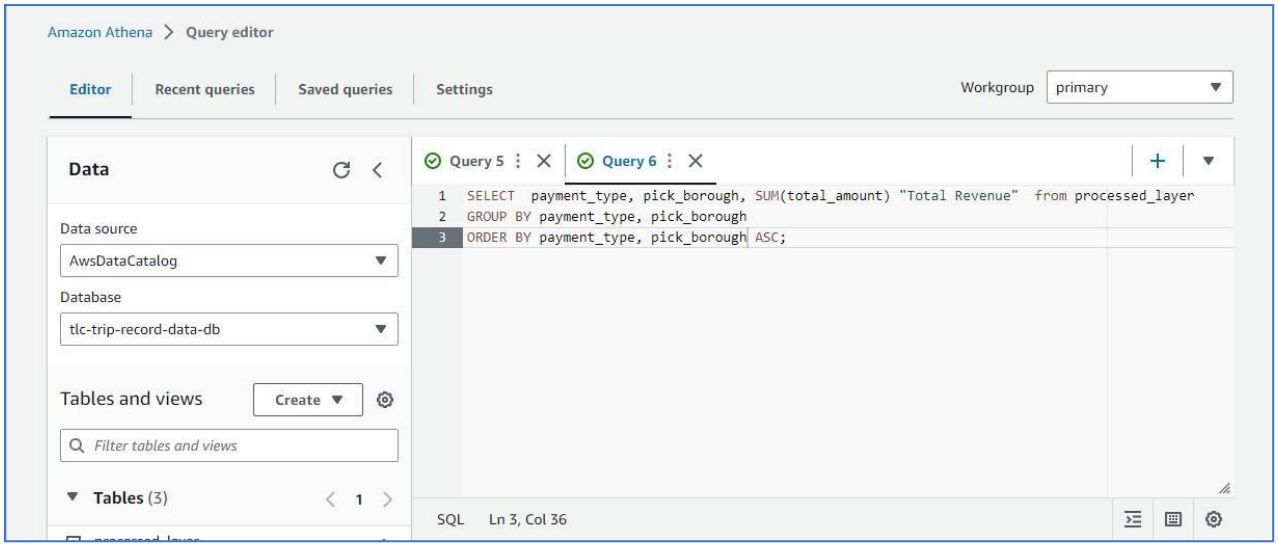
47. Now, let's perform SQL Analytics using Athena on processed layer.





#	pick_locationid	pick_borough	pick_zone	pick_service_zone	vendor_id	pickup_datetime	dropoff_date
1	132	Queens	JFK Airport	Airports	1	2020-04-10 08:13:33.000	2020-04-10 0
2	132	Queens	JFK Airport	Airports	2	2020-04-13 23:46:04.000	2020-04-13 2
3	132	Queens	JFK Airport	Airports	1	2020-04-26 12:37:45.000	2020-04-26 1
4	132	Queens	JFK Airport	Airports	1	2020-05-18 14:06:56.000	2020-05-18 1
5	132	Queens	JFK Airport	Airports	2	2020-05-22 17:30:56.000	2020-05-22 1
6	132	Queens	JFK Airport	Airports	2	2020-05-24 12:47:28.000	2020-05-24 1

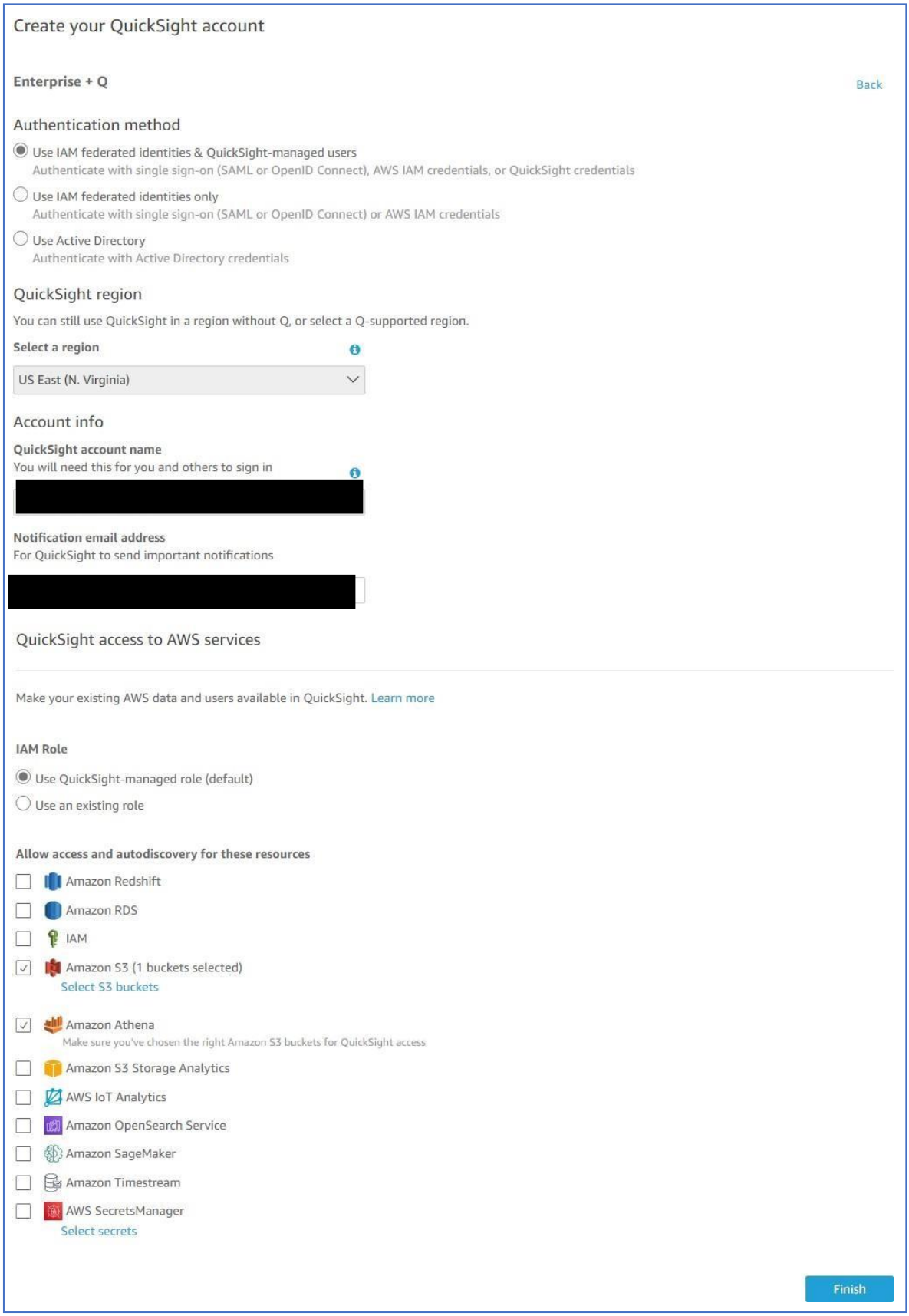
48. Do some more analysis.



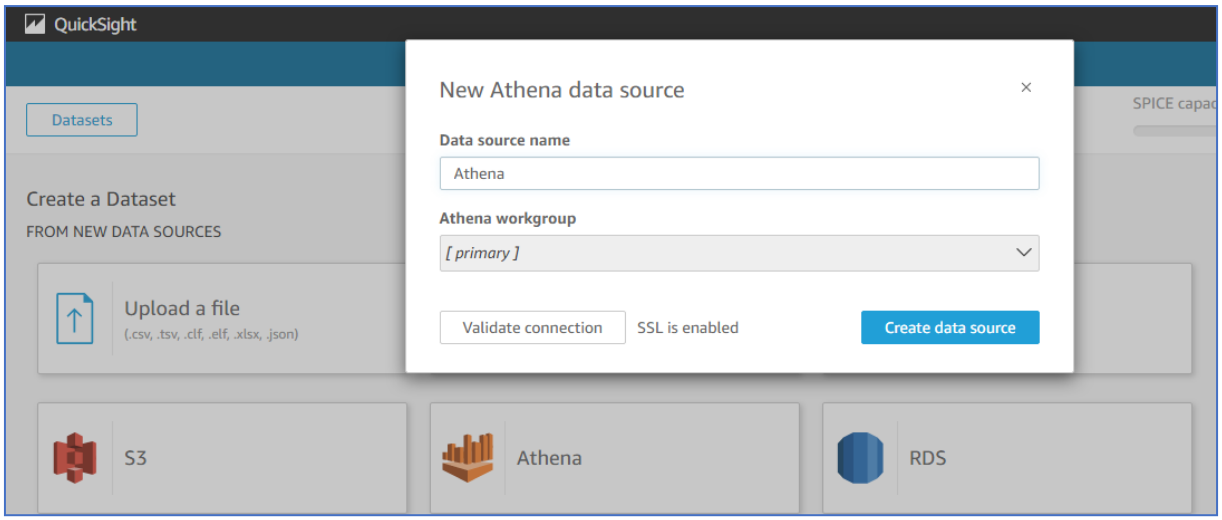
#	payment_type	pick_borough	Total Revenue
2	1	Brooklyn	249289.99000000206
3	1	EWR	2669.5
4	1	Manhattan	9386629.220000826
5	1	Queens	1096478.5399999865
6	1	Staten Island	12408.750000000002
7	1	Unknown	132535.98000000115
8	2	Bronx	49637.86999999938
9	2	Brooklyn	68562.0299999986
10	2	EWR	439.34999999999997
11	2	Manhattan	4229139.020006562
12	2	Queens	480157.6900000231

Stage-5: Visualisation (Optional)

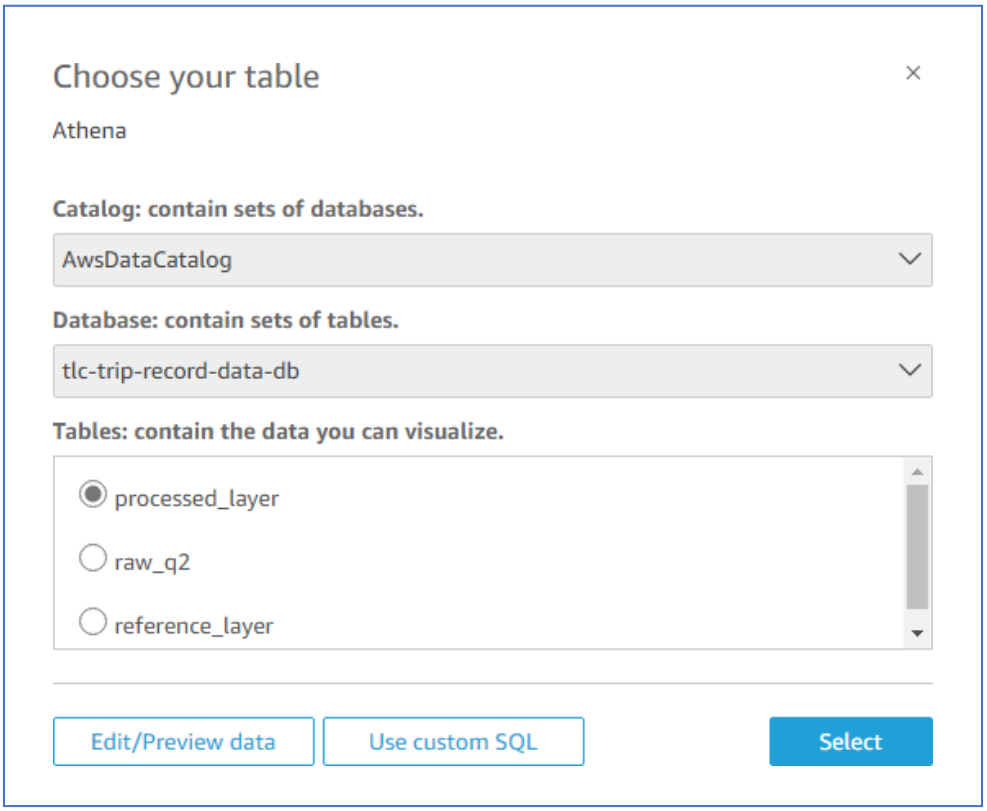
49. Sign up for Amazon QuickSight Account.



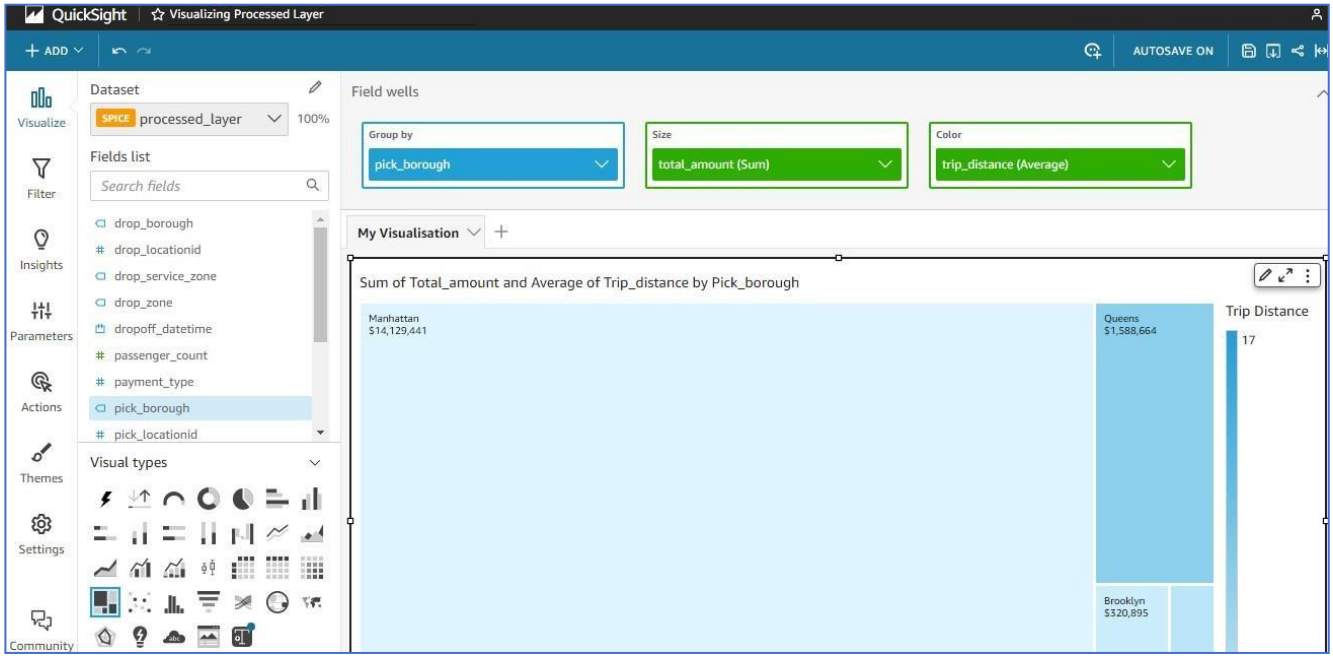
50. Once we will have an Amazon QuickSight Account, let's start adding new dataset. Choose **Athena as dataset source. Under Data Source name >. Athena. Create data source.**



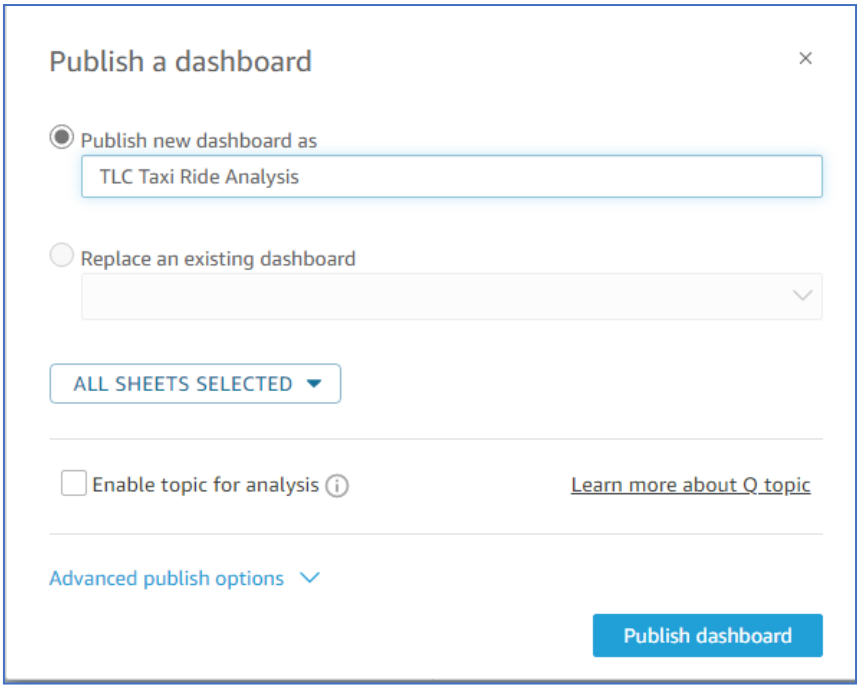
51. Under **Choose your table >> Select your Glue Database. Table>> processed table and click select.**

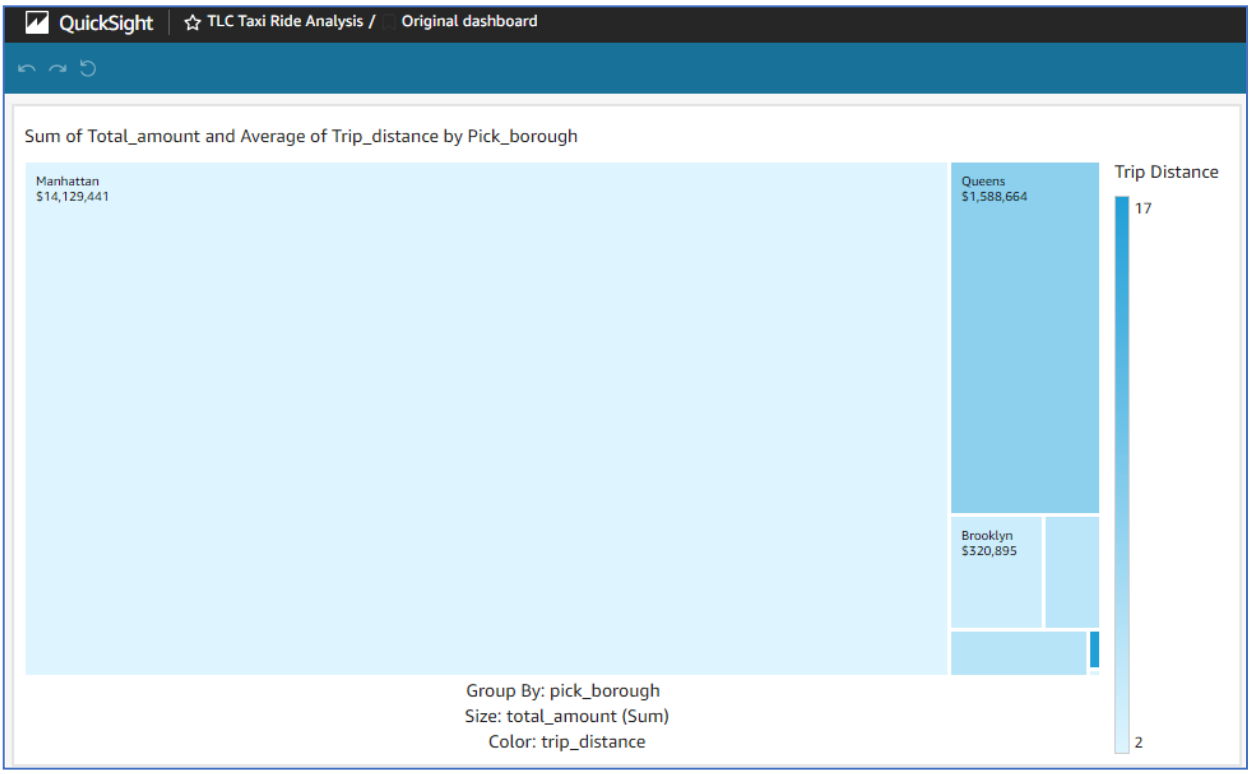


52. Once the dataset is imported into SPICE, create visualization as required. Here is one sample visualization for you.



53. Save it and publish as a Dashboard as following:





54. Once done, terminate your Amazon QuickSight Account.