▼ AWS account access

Open AWS console (us-east-1)

Get AWS CLI credentials

Exit event

Event ends in 5 hours 45 minutes.

Event dashboard > Lab3 - Query multiple data sources using Amazon Athena federated query >
Query multiple data sources using Athena Federated Query

Query multiple data sources using Athena Federated Query

In Lab 2, we stored taxi trip data in DynamoDB and replicated completed trip records to Aurora PostgreSQL using DynamoDB streams and AWS lambda function. Now, we will validate data accuracy and consistency of the trip record between Amazon DynamoDB and Amazon Aurora using Athena Federated query.

Querying data from Amazon DynamoDB and Amazon Aurora

1. Choose Query editor in the navigation bar of Athena AWS console.
2. Run some sample queries on the Aurora PostgreSQL database using the catalog name.

   Query to list the databases in *rdbcatalog* catalog:

   ```
   show databases in rdbcatalog;
   ```

   Query to list the tables in public schema of *rdbcatalog* catalog:

   ```
   show tables in rdbcatalog.public;
   ```

   Query to view the trip records in *rdbcatalog* catalog:

   ```
   select * from rdbcatalog.public.trips;
   ```

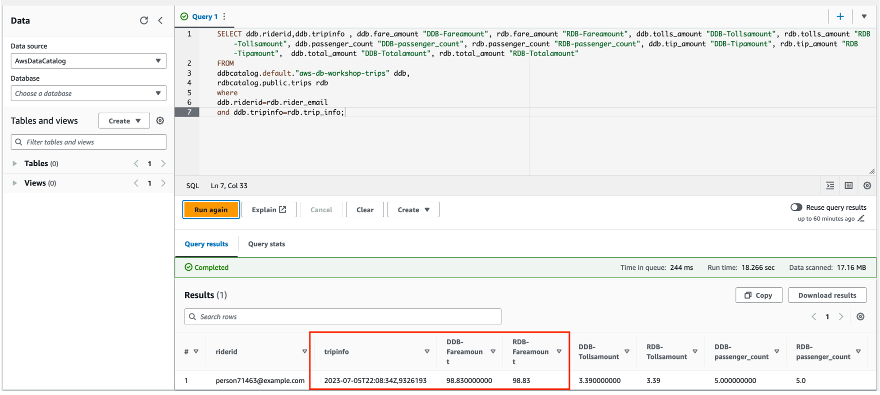
3. Run a sample query on DynamoDB using the catalog name.

   ```
   select * from ddbcatalog.default."aws-db-workshop-trips" where riderid='person6915
   ```

4. Run the query below which joins (Inner join) the trip data from Aurora PostgreSQL with DynamoDB. We have used the *riderid* attribute from DynamoDB to join with *rider_email* column of *trips_query* table in Aurora PostgreSQL. *trips_info* field is used as an additional join condition. The purpose of the query is to check data consistency of trip record between the two data stores.

   ```
   SELECT ddb.riderid,ddb.tripinfo , ddb.fare_amount "DDB-Fareamount", rdb.fare_amour
   FROM
   ddbcatalog.default."aws-db-workshop-trips" ddb,
   rdbcatalog.public.trips rdb
   where
   ddb.riderid=rdb.rider_email
   and ddb.tripinfo=rdb.trip_info;
   ```

   The query should take 10-20 seconds to complete and you should see an output similar to the following:



```
SELECT ddb.riderid,ddb.tripinfo , ddb.fare_amount "DDB-Fareamount", rdb.fare_amount "RDB-Fareamount", ddb.tolls_amount "DDB-Tollsamount", rdb.tolls_amount "RDB
    -Tollsamount", ddb.passenger_count "DDB-passenger_count", rdb.passenger_count "RDB-passenger_count", ddb.tip_amount "DDB-Tipamount", rdb.tip_amount "RDB
    -Tipamount",  ddb.total_amount "DDB-Totalamount", rdb.total_amount "RDB-Totalamount"
FROM
ddbcatalog.default."aws-db-workshop-trips" ddb,
rdbcatalog.public.trips rdb
where
ddb.riderid=rdb.rider_email
and ddb.tripinfo=rdb.trip_info;
```

Completed Time in queue: 244 ms Run time: 18.266 sec Data scanned: 17.16 MB

Results (1)

#	riderid	tripinfo	DDB-Fareamount	RDB-Fareamount	DDB-Tollsamount	RDB-Tollsamount	DDB-passenger_count	RDB-passenger_count
1	person71463@example.com	2023-07-05T22:08:34Z,9326193	98.830000000	98.83	3.390000000	3.39	5.000000000	5.0

5. Validate the consistency of the trip record between Amazon DynamoDB and Amazon Aurora by reviewing the output of the query.

Congratulations!! You have successfully completed Lab 3.

Previous Next

AWS DMS
Prepare the Environment
Creating Endpoints for Source and Target databases
Migrate data from Oracle source to DynamoDB target
Migrate data from Oracle source to Aurora PostgreSQL target
Final Validation of DMS Tasks
Lab2 - Data processing using Amazon DynamoDB and Amazon Aurora
Pre-requisites
Setup AWS Cloud 9 Environment
Enable Amazon DynamoDB Streams
Deploy AWS Lambda Function for DynamoDB Stream Integration
Deploy AWS Lambda Functions for Taxi Ride workflow
Create and Deploy API for Taxi Ride workflow
Taxi Ride Workflow
Lab3 - Query multiple data sources using Amazon Athena federated query
Pre-requisites
Prepare the Environment
Setup Athena Connectors and Catalogs
Query multiple data sources using Athena Federated Query
Lab4 - High performance and scale with Amazon DynamoDB
Lab5 - Integrating Amazon MemoryDB for GeoSpatial implementation
More Resources
Contributors & Revision History
AWS account access
Open AWS console (us-east-1)
Get AWS CLI credentials
Exit event