Lab 2: Glue, Athena, QuickSight

This lab is divided into several parts to help you get familiar with **Amazon Glue**, **Athena** and **QuickSight**. We are going to use the files that **Kinesis Firehose** stored into our **S3** bucket coming from our simulator running in the **EC2** instance. First, we are going to use **Glue** to discover the data. A **Glue crawler** will infer the data structure and create a **Hive** external table into the **Amazon Data Catalog**. Next we are going to run a **Glue ETL** job using **pySpark** in order to transform the input files into **Parquet** format which is ideal for queries. Next, we will query the data in **parquet** format stored in **S3** using **Athena** and finally will visualize the data with **QuickSight** which will query **Athena**.

Automating Table Creation

Creating a Service Role for Glue

Note: Since we are using the same account, only one student needs to do this step.

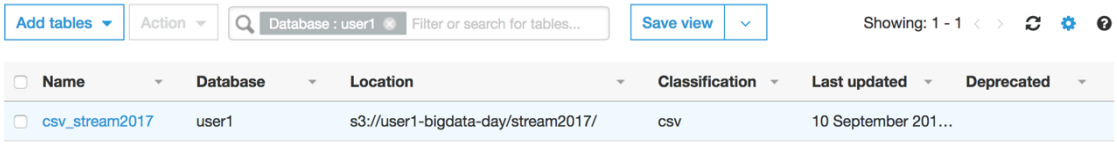
1. Access the **IAM** console and select **Roles** and create a new role
2. Select Role type: **AWS Service -> Glue**
3. Click: **Next Permissions**
4. From the list of managed policies, attach the following:
    1. AWSGlueServiceRole
    2. AWSGlueServiceNotebookRole
    3. AmazonS3FullAccess
5. Name it **AWSGlueServiceRole**. If you choose a different name you will need to manually create a new policy.

Creating an Athena Table using Glue Crawler

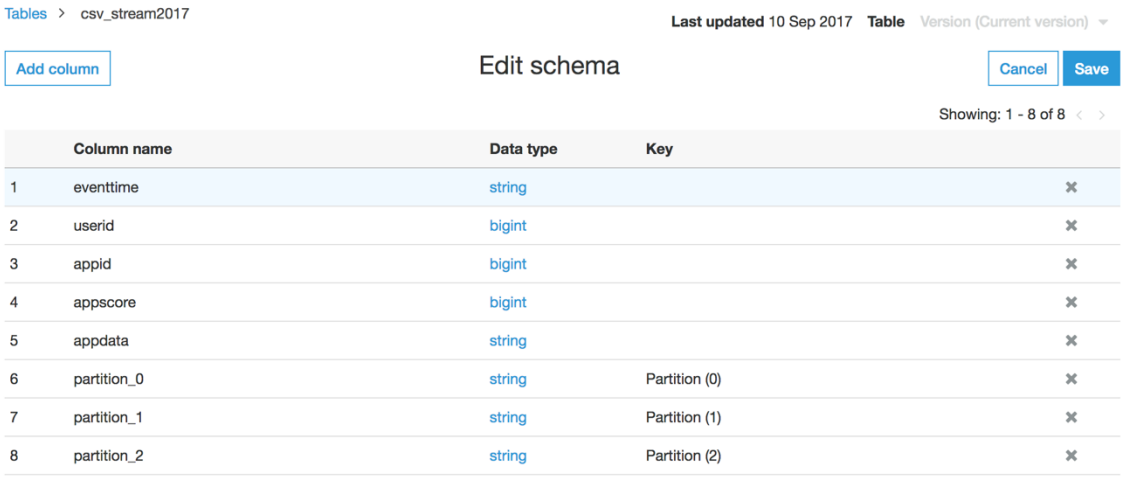
1. Choose the **North Virginia** Region
2. Open the **AWS Glue** console
3. From the **Crawlers** pane on the left hand side, click **Add Crawler**
4. Crawler name: **<user>_stream_crawler**
5. Click **Next**.
6. Select the **Specify path in my account** radio button and enter **s3://<user>-bigdata-day/stream/** for the **S3** path. Click **Next**.
7. Do **not** add another data source and click **Next**.
8. Sthe IAM role we created in the previous section: **AWSGlueServiceRole**
9. For frequency leave as **Run on Demand** and click **Next**.
10. Click **Add Database** button and give your database a name, say **<user>**
11. In order to avoid table name collision **Glue** generates a unique table name so we'll need to provide a prefix, say **csv_** (include the underscore)
12. Click **Finish**
13. Check the box next to your newly created crawler and click **Run Crawler**. It should take a few minutes to run and create one table.

Exploring Glue Data Catalog

1. On the left hand side, click **Databases**
2. Find the **<user>** database and click on it
3. Click **Tables in <user>** to view our newly created table



4. Click the table name and explore.
5. Click **Edit Schema**
6. Name the columns of the table generates



7. Click **Save**

Querying the Data with Athena

1. Switch to the **Athena** console
2. In **Query Editor**, choose the database: **<user>**
3. We should see a table **csv_stream2019** available. Please note that this table is partitioned.

4. Enter `SHOW PARTITIONS csv_stream2019` to verify all partitions were automatically added. The partitions are added based on the directories where the files are stored in S3 (Firehose creates a month/day/hour structure)

5. You can try the following SQL statements below to explore the data is S3 in CSV format.

```
SELECT * from csv_stream2019 LIMIT 10;

SELECT COUNT(*)from csv_stream2019;

SELECT * from csv_stream2019 ORDER BY eventtime DESC;

SELECT SUM(appscore) as count_appscore from csv_stream2019;
```

Transforming the Data

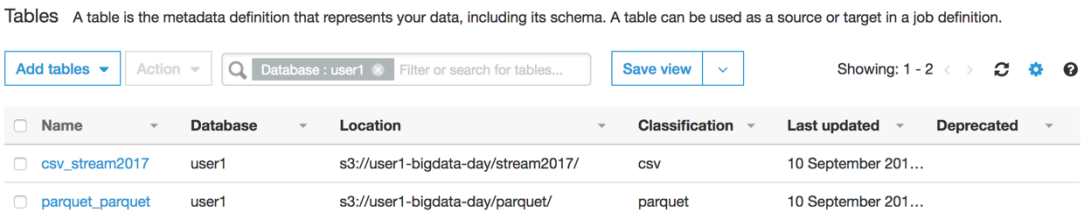
1. Go to **Glue** Console
2. From the Glue ETL menu select **Jobs** and click **Add Job**
3. Give your job a name, **<user>_stream_parquet_etl** and select our service role **AWSGlueServiceRole**.
4. This job runs: **A proposed script generated by AWS Glue**
5. Script file name: **<user>_stream_parquet**
6. S3 path where the script is stored (prefilled, od not change): **s3://aws-glue-scripts-<AWS ACCOUNT>-us-east-1/<user>**
7. Temporary directory (prefilled, od not change):: **s3://aws-glue-scripts-<AWS ACCOUNT>-us-east-1/<user>/tmp**
8. Click **Next**
9. From the Data Sources list select the table **csv_stream2019** from the **<user>** database
10. Click **Next**
11. When choosing our data targets select **Create tables in your data target**.
12. Data Store: **Amazon S3**
13. Format: **Parquet**
14. Target path: **s3://<user>-bigdata-day/parquet/**
15. Click: **Next**
16. In this step we can change the source to target column mapping. We will leave by default. Click **Next**
17. Click **Finish** to complete creating our ETL

As you can see, AWS Glue created a script for you to get started. If we didn't need to do anything else, this script simply converts our CSV data to Parquet.

18. Click: **Save**
19. Click: **Run Job**
20. The job will take a few minutes to complete.

One thing to note is that we're writing out a new set of data to our own S3 bucket. We need to configure a crawler to scan this new bucket and create appropriate tables in the Data Catalog so we can query them with Athena.

1. From the **Crawlers** pane on the left hand side, click **Add Crawler**
2. Crawler name: **<user>_parquet_crawler**
3. Click **Next**.
4. Select the **Specify path in my account** radio button and enter **s3://<user>-bigdata-day/parquet/** for the S3 path. Click **Next**.
5. Do **not** add another data source and click Next.
6. Select the IAM role we created in the previous section: **AWSGlueServiceRole**
7. For frequency leave as **Run on Demand** and click Next.
8. Choose **Database** with name **<user>**
9. In order to avoid table name collision Glue generates a unique table name so we'll need to provide a prefix, say **parquet_** (include the underscore)
10. Click **Finish**
11. Check the box next to your newly created crawler and click **Run Crawler**. It should take a few minutes to run and it should create our table if everything goes ok.



Go ahead and explore the data. Open the **Athena** console. You will see now 2 tables in the <user> database, one in **csv** format and another one in **parquet** format.

Run the same queries than before but on the parquet table:

```
SELECT * from parquet_parquet LIMIT 10;

SELECT COUNT(*)from parquet_parquet;

SELECT * from parquet_parquet ORDER BY eventtime DESC;

SELECT SUM(appscore) as count_appscore from parquet_parquet;
```

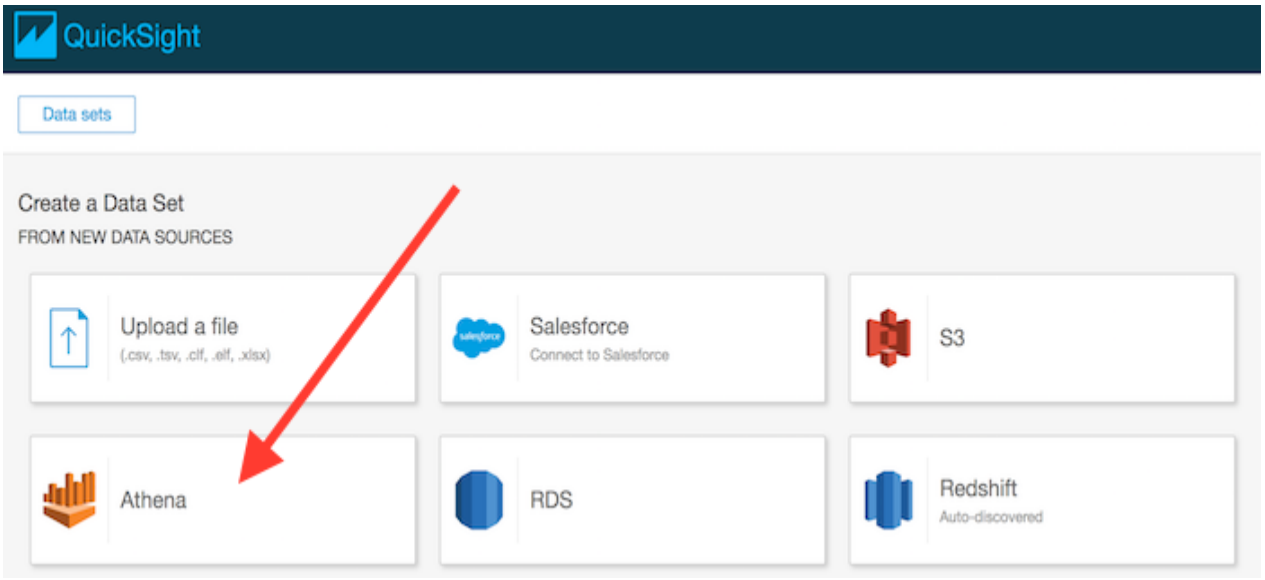
The parquet database is not partitioned because the ETL job has not created a subdirectory structure by default, although the pySpark script could have been updated to do so.

Connecting with Amazon QuickSight

To explore the data we previously created we need to login to **Amazon QuickSight** and create a **data set** directly from **Amazon Athena**.

1. Open the **QuickSight** console
2. If it's your first time you'll be asked to enter an email address to create an account. If you've accessed QuickSight before you'll land in the main window. If you use QuickSight for the first time, click through the wizard and **connect Athena and S3** (with your lab bucket)

3. Click the **New Analysis** button at the top left of the screen
4. Click **New Data Set** and select **Athena** from the list of sources

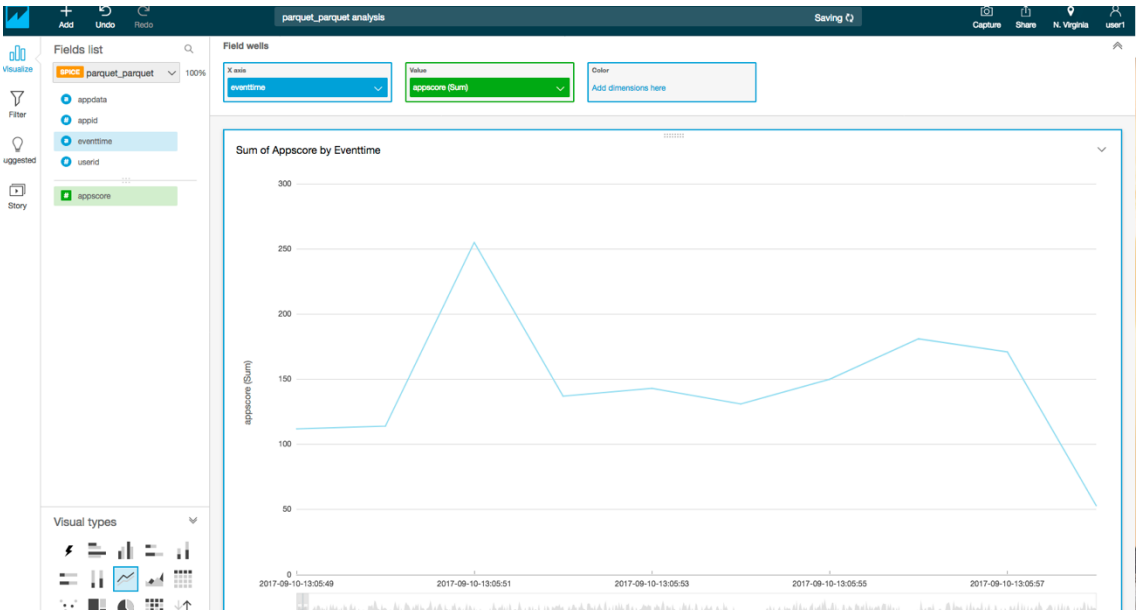


5. Give your data source a name and click **Create data source**
6. Select the **<user>** database we created earlier in this lab. Click **Validate Connection**
7. Select the database <user> and the table **`parquet_parquet`** as a starting point for us to explore. Click **Select**.

8. Select to **Import to Spice to visualize your data** and click **Visualize**

At this point you are presented with a blank graph so feel free to play around and build interesting visualizations.

For example, the following visualization shows the sum of AppScores in each timestamp.



Even if the data set that we are using in this lab is not very rich, you can try to create other visualizations (i.e. Charts, etc).