ECV-BD-311

Building a Data Lake with AWS Glue and Amazon S3

2018.02.23

Version 1.1

ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD. CONFIDENTIAL. THE INFORMATION IN THIS DOCUMENT MAY NOT BE DISCLOSED WITHOUT THE PRIOR WRITTEN CONSENT OF ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD.

Agenda

About this lab

Scenario

The following procedures help you set up a data lake that could store and analyze data that addresses the challenges of dealing with massive volumes of heterogeneous data. A data lake allows organizations to store all their data—structured and unstructured—in one centralized repository. Because data can be stored as-is, there is no need to convert it to a predefined schema. This tutorial walks you define a database, configure a crawler to explore data in an Amazon S3 bucket, create a table, transform the CSV file into Parquet, create a table for the Parquet data, and query the data with Amazon Athena.

AWS Glue introduction

What is AWS Glue?

AWS Glue is a fully managed data catalog and ETL (extract, transform, and load) service that simplifies and automates the difficult and time-consuming tasks of data discovery, conversion, and job scheduling. AWS Glue crawls your data sources and constructs a data catalog using pre-built classifiers for popular data formats and data types, including CSV, Apache Parquet, JSON, and more. It is significantly reducing the time and effort that it takes to derive business insights quickly from an Amazon S3 data lake by discovering the structure and form of your data. Also automatically crawls your Amazon S3 data, identifies data formats, and then suggests schemas for use with other AWS analytic services.

Amazon S3 introduction

What is Amazon S3?

Amazon S3 is a highly durable, cost-effective object start that supports Open Data Formats while decoupling storage from compute, and it works with all the AWS analytic services.

The workshop's region will be in 'Virginia'

ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD. CONFIDENTIAL. THE INFORMATION IN THIS DOCUMENT MAY NOT BE DISCLOSED WITHOUT THE PRIOR WRITTEN CONSENT OF ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD.

Prerequisites

- Sign-in a AWS account, and make sure you have select N.Virginia region.

Lab tutorial

1.1 Create IAM role

1.1. On the **service** menu, click **IAM**.

1.2. In the navigation pane, choose **Roles.**

1.3. Click **Create role**.

1.4. For role type, choose **AWS Service**, find and choose **Glue**, and choose **Next: Permissions**.

1.5. On the **Attach permissions policy** page, search and choose **AWSS3FullAccess, AWSGlueServiceRole**, and choose **Next: Review**.

1.6. On the **Review** page, enter the following detail:

- **Role name: AWSGlueServiceRoleDefault**

1.7. Click **Create role.**

1.8. Choose **Roles** page, select the role **AWSGlueServiceDefault** you just created.

1.9. On the **Permissions** tab, choose the link **add inline policy** to create an inline policy.

1.10. On the JSON tab, paste in the following policy:

```
{
 "Version": "2012-10-17",
 "Statement": [
   {
     "Effect": "Allow",
     "Action": [
       "logs:CreateLogGroup",
       "logs:CreateLogStream",
       "logs:PutLogEvents",
       "logs:DescribeLogStreams"
   ],
     "Resource": [
       "arn:aws:logs:*:*:*"
   ]
  }
]
```

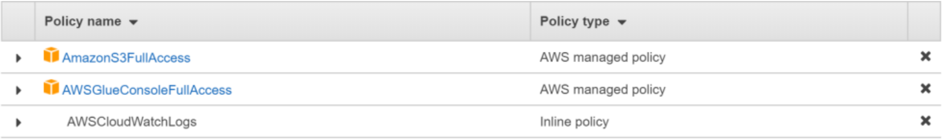
}

1.11. Click **Review policy**.

1.12. On the Review policy, enter policy name: **AWSCloudWatchLogs**.

1.13. Click **Create policy**.

1.14. Now confirm you have policies as below figure.



Policy name	Policy type	
AmazonS3FullAccess	AWS managed policy	✖
AWSGlueConsoleFullAccess	AWS managed policy	✖
AWSCloudWatchLogs	Inline policy	✖

Figure1: IAM role policies

1.2 Add Crawler

1.15. On the **Services** menu, click **AWS Glue**.

1.16. In the console, choose **Add database**. In the **Database name**, type **nycitytaxi**, and choose **Create**.

1.17. Choose **Crawlers** in the navigation pane, choose **Add crawler**. Add type Crawler name **nytaxicrawler**, and choose **Next**.

1.18. On the **Add a data store** page, choose **S3** as data store.

1.19. Select **Specified path in my account**.

1.20. Enter data store path **s3://aws-bigdata-blog/artifacts/glue-data-lake/data/**, and choose **Next**.

1.21. On **Add another data store** page, choose **No**, and choose **Next**.

1.22. Select Choose an existing IAM role, and choose the role **AWSGlueServiceRoleDefault** you just created in the drop-down list, and choose **Next**.

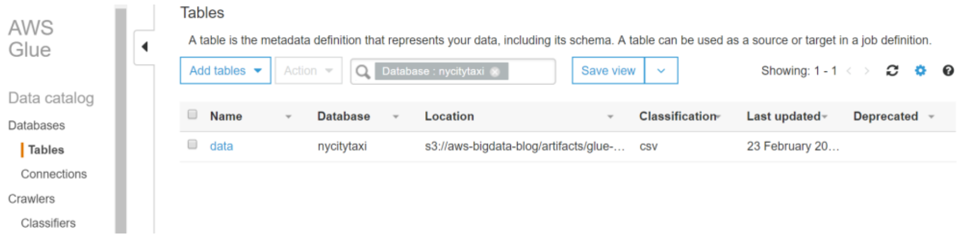
1.23. For **Frequency**, choose **Run on demand**, and choose **Next**.

1.24. For **Database**, choose **nycitytaxi**, and choose **Next**.

1.25. Review the steps, and choose **Finish**.

1.26. The crawler is ready to run. Choose **Run it now**.

1.27. When the crawler has finished, one table has been added. Choose **Tables** in the left navigation pane, and then choose **data** to confirmed.

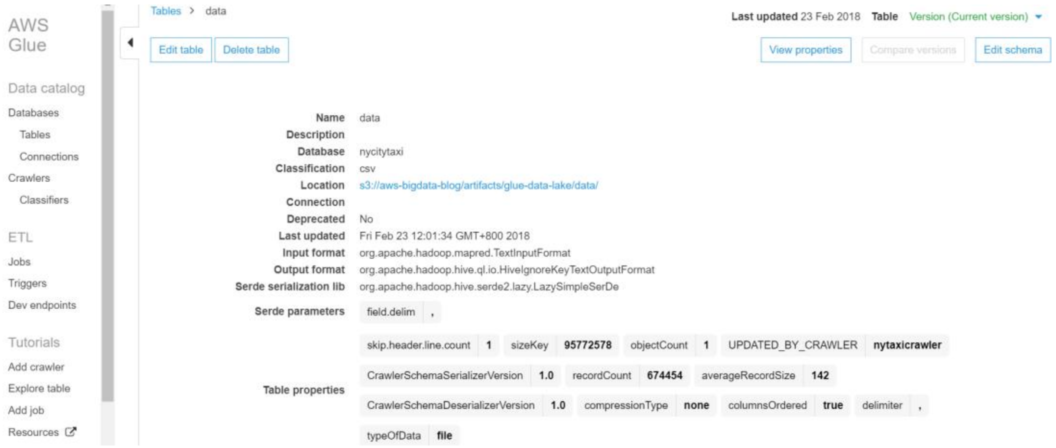


Tables

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.

Name	Database	Location	Classification	Last updated	Deprecated
data	nycitytaxi	s3://aws-bigdata-blog/artifacts/glue-...	csv	23 February 20...	

Figure2: AWS Glue table has been added



Last updated 23 Feb 2018 **Table** Version (Current version)

Name	data
Description	
Database	nycitytaxi
Classification	csv
Location	s3://aws-bigdata-blog/artifacts/glue-data-lake/data/
Connection	
Deprecated	No
Last updated	Fri Feb 23 12:01:34 GMT+800 2018
Input format	org.apache.hadoop.mapred.TextInputFormat
Output format	org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Serde serialization lib	org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Serde parameters	field.delim ,
Table properties	skip.header.line.count 1 sizeKey 95772578 objectCount 1 UPDATED_BY_CRAWLER nytaxicrawler CrawlerSchemaSerializerVersion 1.0 recordCount 674454 averageRecordSize 142 CrawlerSchemaDeserializerVersion 1.0 compressionType none columnsOrdered true delimiter , typeOfData file

Figure3: table information

1.3Transform the Data from CSV to Parquet Format

1.28. In the navigation pane, under **ETL**, choose **Jobs**, and then choose **Add job.**

1.29. On the Job properties, enter the following details:

Name: nytaxi-csv-parquet

IAM role: choose **AWSGlueServiceRoleDefault**

1.30. For **This job runs**, select **A proposed script generated by AWS Glue**.

1.31. Choose **Next.**

1.32. Choose **data** as the data source, and choose **Next.**

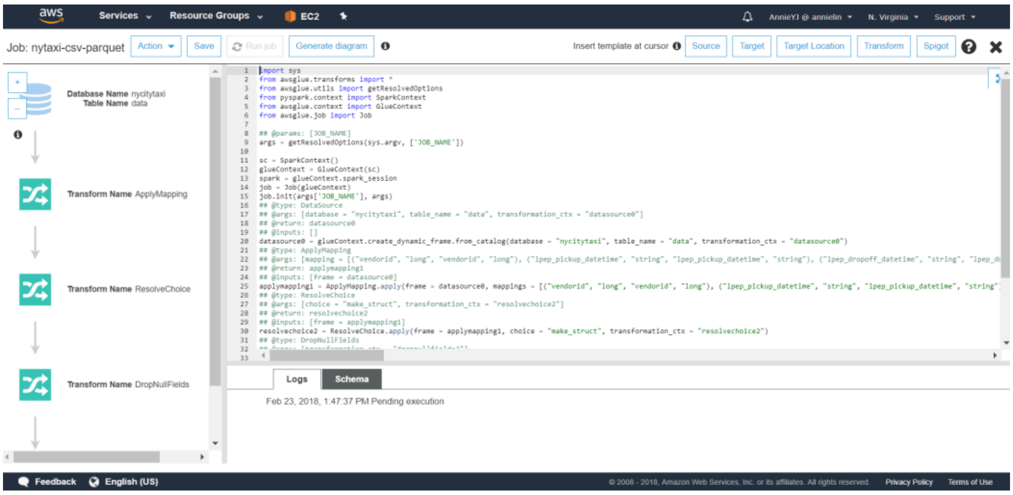
1.33. Choose **Create tables in your data target.**

1.34. For Data store, choose Amazon S3, and choose **Parque**t as the format.

1.35. For **Target path**, select S3 bucket with **s3://aws-glue-result-xxxx** to store the results.

1.36. Verify the schema mapping, and choose **Finish.**

1.37. View the job. This screen provides a complete view of the job and allows you to edit, click **Save**, and choose **Run job.** This steps may be waiting around 10 minutes.



```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [database = "nycitytaxi", table_name = "data", transformation_ctx = "datasource0"]
## @return: datasource0
## @inputs: []
datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "nycitytaxi", table_name = "data", transformation_ctx = "datasource0")
## @type: ApplyMapping
## @args: [mapping = [("vendorid", "long", "vendorid", "long"), ("lpep_pickup_datetime", "string", "lpep_pickup_datetime", "string"), ("lpep_dropoff_datetime", "string", "lpep_d
## @return: applymapping1
## @inputs: [frame = datasource0]
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("vendorid", "long", "vendorid", "long"), ("lpep_pickup_datetime", "string", "lpep_pickup_datetime", "string"
## @type: ResolveChoice
## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
## @return: resolvechoice2
## @inputs: [frame = applymapping1]
resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct", transformation_ctx = "resolvechoice2")
## @type: DropNullFields
## @args: [transformation_ctx = "dropnullfields3"]
```

```
Feb 23, 2018, 1:47:37 PM Pending execution
```

Figure4: job runing

1.4 Add the Parquet Table and Crawler

1.38. When the job has finished, add a new table for the Parquet data using a crawler.

1.39. In the navigation pane, choose **Add crawler**. Add type Crawler name **nytaxiparquet**, and choose **Next**.

1.40. Choose S3 as the **Data store**.

1.41. Include path choose **s3://aws-glue-result-xxxx** to store data.

1.42. Choose **Next.**

1.43. On **Add another data store** page, choose **No**, and choose **Next.**

1.44. Select **Choose an existing IAM role**, and choose the role **AWSGlueServiceRoleDefault** you just created in the drop-down list, and choose **Next.**

1.45. For **Frequency**, choose **Run on demand**, and choose **Next.**

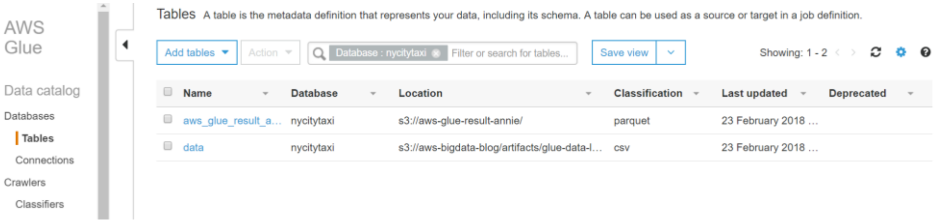
1.46. For **Database**, choose **nycitytaxi**, and choose **Next.**

1.47. Review the steps, and choose **Finish.**

1.48. The crawler is ready to run. Choose **Run it now**.

1.49. After the crawler has finished, there are two tables in the **nycitytaxi** database: a table for the raw CSV data and a table for the transformed Parquet data.

ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD. CONFIDENTIAL. THE INFORMATION IN THIS DOCUMENT MAY NOT BE DISCLOSED WITHOUT THE PRIOR WRITTEN CONSENT OF ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD.



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.

Database : nycitytaxi | Filter or search for tables...

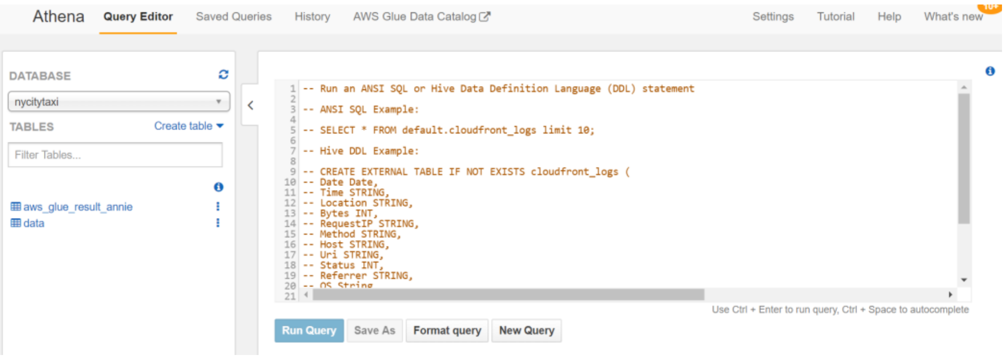
Name	Database	Location	Classification	Last updated	Deprecated
aws_glue_result_a...	nycitytaxi	s3://aws-glue-result-annie/	parquet	23 February 2018 ...	
data	nycitytaxi	s3://aws-bigdata-blog/artifacts/glue-data-l...	csv	23 February 2018 ...	

Figure 5: table for transformed parquet data

1.5 Analyze the Data with Amazon Athena

1.50. On the **Services** menu, click **Athena**.

1.51. On the **Query Editor** tab, choose the database **nycitytaxi**.



```
-- Run an ANSI SQL or Hive Data Definition Language (DDL) statement

-- ANSI SQL Example:

-- SELECT * FROM default.cloudfront_logs limit 10;

-- Hive DDL Example:

-- CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_logs (
-- Date Date,
-- Time STRING,
-- Location STRING,
-- Bytes INT,
-- RequestIP STRING,
-- Method STRING,
-- Host STRING,
-- Uri STRING,
-- Status INT,
-- Referrer STRING,
-- OS String
```

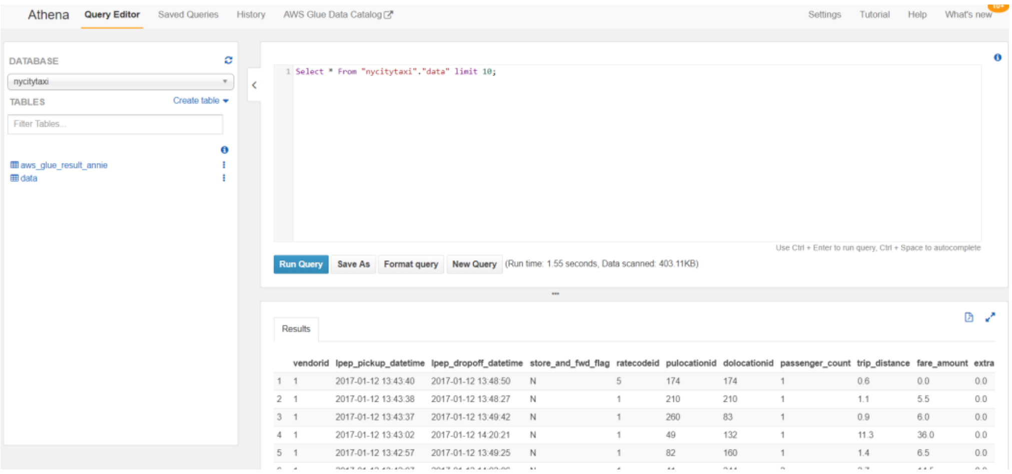
Figure 6: Athena query editor

1.52. Choose the **aws_glue_result_xxxx** table.

1.53. Query the data, type below standard SQL:

```
Select * From "nycitytaxi"."data" limit 10;
```

1.54. Choose **Run Query**.



```
Select * From "nycitytaxi"."data" limit 10;
```

(Run time: 1.55 seconds, Data scanned: 403.11KB)

	vendorid	lpep_pickup_datetime	lpep_dropoff_datetime	store_and_fwd_flag	ratecodeid	pulocationid	dolocationid	passenger_count	trip_distance	fare_amount	extra
1	1	2017-01-12 13:43:40	2017-01-12 13:48:50	N	5	174	174	1	0.6	0.0	0.0
2	1	2017-01-12 13:43:38	2017-01-12 13:48:27	N	1	210	210	1	1.1	5.5	0.0
3	1	2017-01-12 13:43:37	2017-01-12 13:49:42	N	1	260	83	1	0.9	6.0	0.0
4	1	2017-01-12 13:43:02	2017-01-12 14:20:21	N	1	49	132	1	11.3	36.0	0.0
5	1	2017-01-12 13:42:57	2017-01-12 13:49:25	N	1	82	160	1	1.4	6.5	0.0

Figure 7: Athena run query

ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD. CONFIDENTIAL. THE INFORMATION IN THIS DOCUMENT MAY NOT BE DISCLOSED WITHOUT THE PRIOR WRITTEN CONSENT OF ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD.

Conclusion

Congratulations! You now have learned how to:

- Build data lake using AWS Glue and Amazon S3.
- Crawler your data to Amazon S3 by AWS Glue.
- Analysis through Amazon Athena service.

ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD. CONFIDENTIAL. THE INFORMATION IN THIS DOCUMENT MAY NOT BE DISCLOSED WITHOUT THE PRIOR WRITTEN CONSENT OF ECLOUDVALLEY DIGITAL TECHNOLOGY CO., LTD.