



**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERIA**

**Escuela Profesional de Ingeniería de Sistemas**

**INFORME DE LABORATORIO N° 09  
“CONSTRUYENDO UN ETL SERVERLESS”**

Curso: Inteligencia de Negocios

Docente: Ing. Patrick Cuadros Quiroga

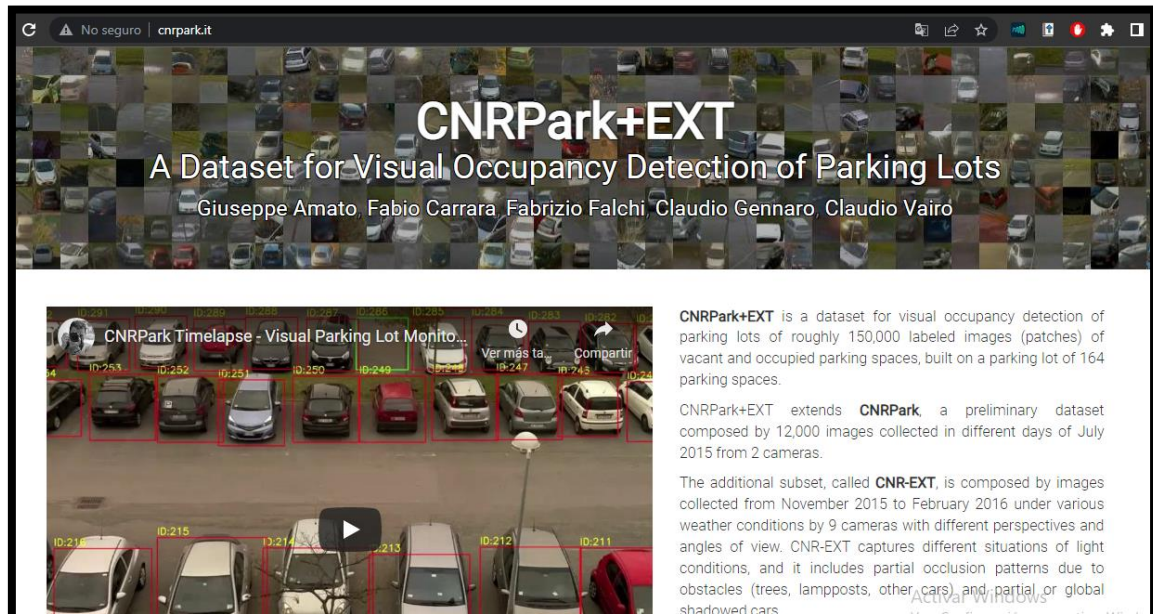
**Anahua Huayhua, Jenny Karen (2018062150)**

**Tacna – Perú  
2022**

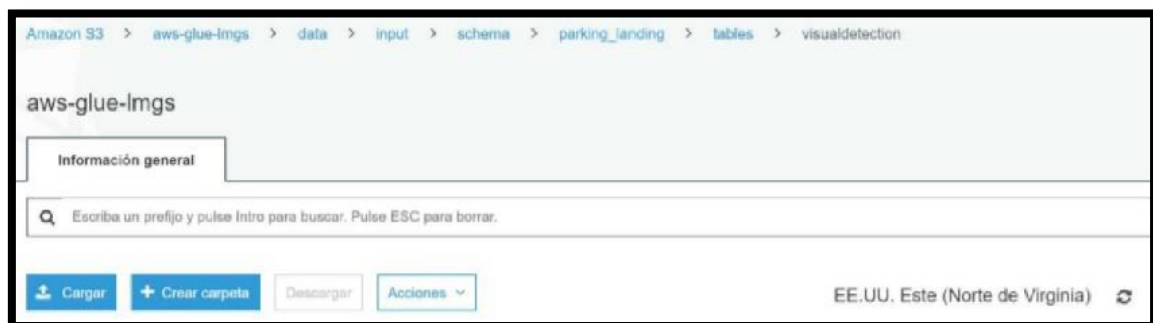
**PRACTICA DE LABORATORIO N° 09****TEMA: CONSTRUYENDO UN ETL SERVERLESS**

Realizar los siguientes pasos para el laboratorio

Url del dataset <http://cnrpark.it/>



1) Entramos a la consola de AWS y accedemos al servicio de S3

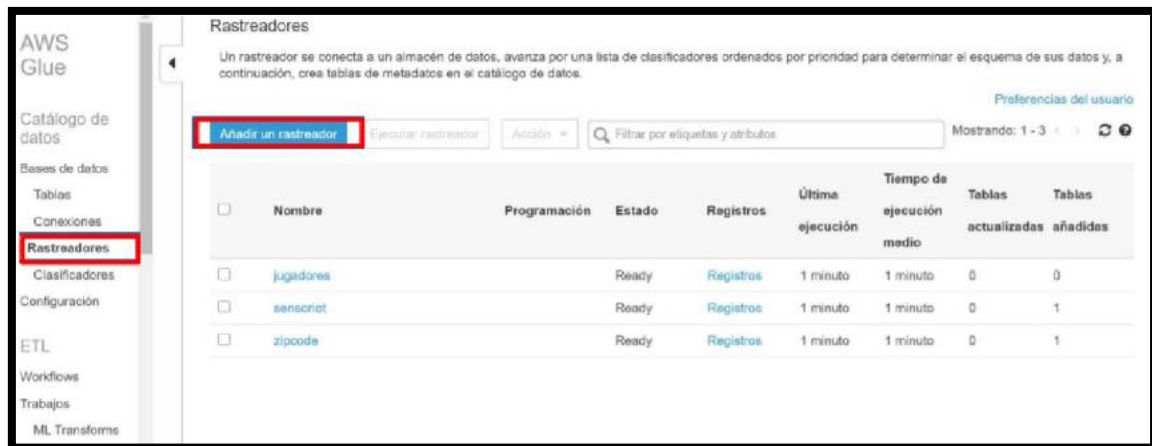


Bucket : aws-glue-lmgs (los últimos 4 caracteres cambiarlos por las iniciales de nuestros nombres, acordarse de que el nombre de los bucket es único a nivel global)

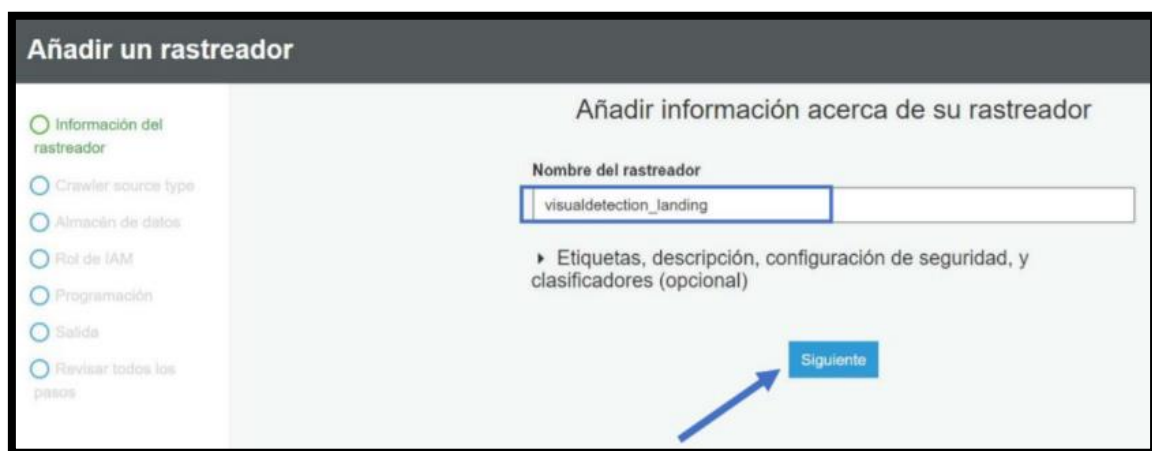
Creamos una carpeta /data/input/schema/parking\_landing/tables/visualdetectionSubir el archivo csv visualdetection.csv dentro de la carpeta /data/input/schema/parking\_landing/tables/visualdetection

**Entrar al servicio de Glue**

Opción Rastreadores o Crawler (en inglés)



Agregamos el nombre visualdetecion\_landing, clic en Siguiente.



Seleccionamos la opción : Data Stores (ya que a partir de un csv se creará una tabla)



Elegimos S3, y seleccionamos la carpeta donde se encuentra el archivo csv, clic en Siguiente.

Seleccionamos hasta la carpeta visualdetecion en S3.

En este caso, solo añadiremos un almacén de datos, elegimos No y clic en Siguiente.

Creamos un nuevo rol, que tendrá el permiso de leer el archivo csv de S3 y crear una tabla en Glue. Nombre del rol : AWSGlueServiceRole-Crawler (Si ya existe, seleccionar el indicado o

concatenarlo con sus iniciales de sus nombres) Este rol es importante, ya que permitirá al Crawler de Glue poder leer el dataset que se encuentra en S3.

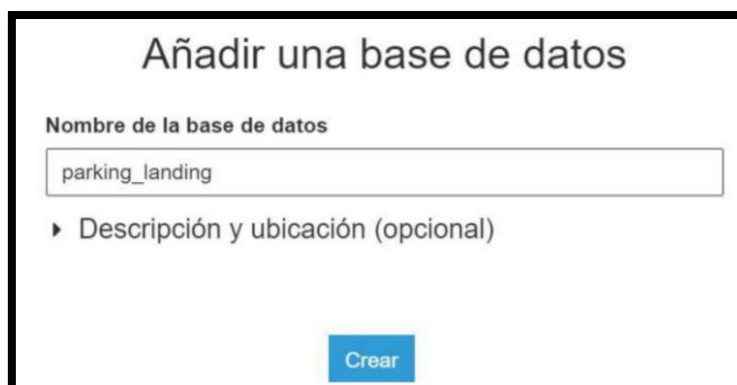


Seleccionamos : Ejecutar bajo demanda, clic en Siguiente.

Se tiene la opción de definir una periodicidad distinta, de acuerdo a la carga de los datos, si por ejemplo sé que todos los días a cierta hora, voy a tener un dataset, podría indicar la hora de ejecución automática de este crawler.

Añadimos una nueva base de datos, le ponemos como nombre parking\_landing, clic en Siguiente. En esta base de datos, se generará de manera automática a partir del dataset que se encuentra en S3, los datos se leerán tal cual están en la fuente de donde se descargó todavía no se ha realizado ninguna transformación.

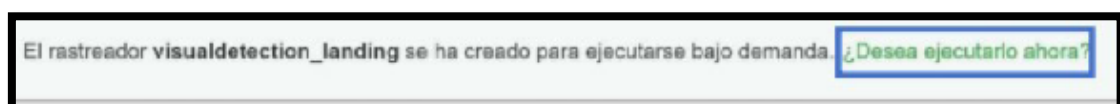
Asignamos el siguiente nombre a la base de datos:



Para no tener inconvenientes, cada uno le pondremos como nombre:

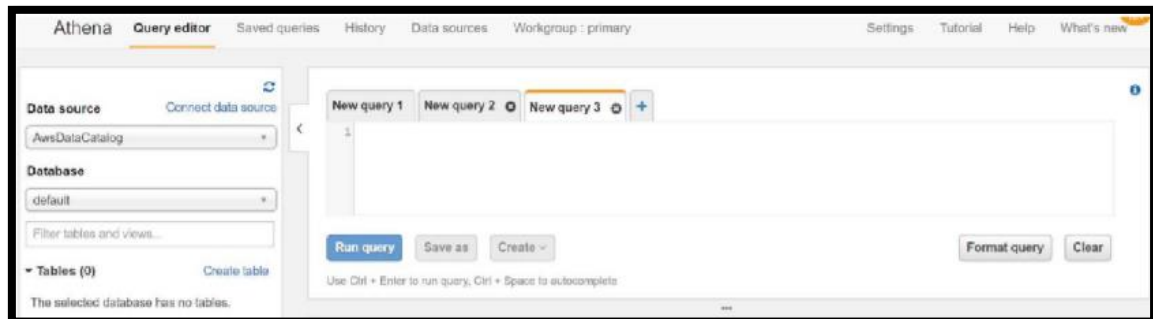
parking\_landing. Le damos clic en Siguiente. Clic en Finalizar.

Por último, para ejecutar este rastreador, le damos clic en ¿Desea ejecutarlo ahora?

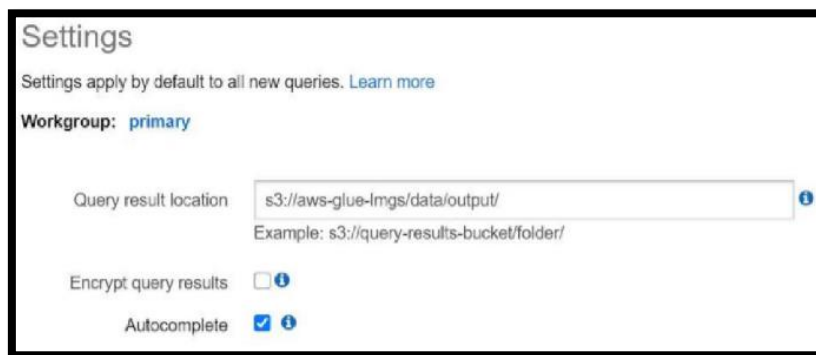




Después de algunos segundos, el rastreador se detiene y veremos en la última columna que se ha añadido una nueva tabla en el catálogo de Glue. Ahora para poder realizar una consulta SQL en el servicio de Athena, entramos al servicio. Antes de realizar una consulta SQL en Athena, debemos realizar la siguiente configuración. Clic en Settings.



Indicamos el bucket que hemos creado, y le agregamos /data/output/, en esta ruta se guardarán 2 archivos (1 el resultado de los queries y el segundo con la metadata).



Ahora podremos ejecutar consultas SQL en Athena.. Ejecutamos algunos queries para probar Athena:

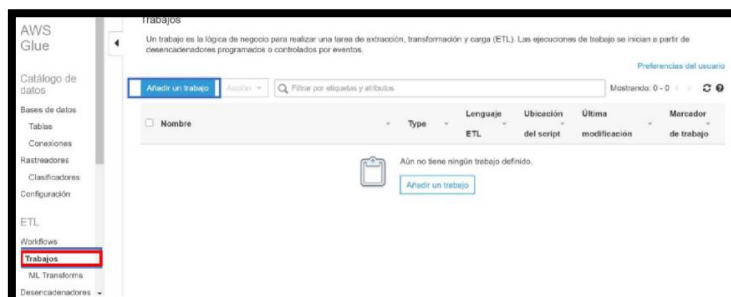
```
SELECT * FROM parking_landing.visualdetection limit 10;
```

```
SELECT * FROM parking_landing.visualdetection where camera = 'A';
```

Ahora crearemos un job de transformación de formato de tablas (de csv a parquet)

Creamos la siguiente estructura de carpetas en S3, dentro del bucket previamente creado.

Carpeta : data/input/schema/parking\_processed/tables/visualdetection. Luego, nos vamos a Glue para crear el job de transformación.





Definir como nombre del job : visualdeteccion\_processed, clic en siguiente.

Seleccionamos la tabla visualdeteccion que está dentro de la base de datos que hemos creado cada uno (es la que se encuentra en formato csv), clic en Siguiente.

Elegimos cambiar esquema y siguiente.

Seleccione, Crear tablas en el destino de datos.

Almacén de datos : S3

Formato : Parquet (óptimo para el consumo, reduce el peso al cambiar el formato)

Ruta de destino: s3://aws-glue-

Imgs/data/input/schema/parking\_processed/tables/visualdeteccionModificar el nombre de bucket por el nuestro.

Realizando el match de columnas, clic en Guardar el trabajo y editar el script.

Asigne las columnas de origen a las columnas de destino.

Verifique las correspondencias creadas por AWS Glue. Cambie las correspondencias eligiendo otras columnas en **Asignar a destino**. Puede **Borrar** todas las correspondencias y **Restablecer** las correspondencias de AWS Glue predeterminadas. AWS Glue genera el script con las correspondencias definidas.

Origen			Destino		
Nombre de la columna	Tipo	Asignar a destino	Nombre de la columna	Tipo	
camera	string	camera	camera	string	x ↓ ↑
datetime	string	datetime	datetime	string	x ↓ ↑
day	bigint	day	day	long	x ↓ ↑
hour	bigint	hour	hour	long	x ↓ ↑
image_url	string	image_url	image_url	string	x ↓ ↑
minute	bigint	minute	minute	long	x ↓ ↑
			month	long	x ↓ ↑

En la siguiente pantalla, podemos realizar modificaciones al código, o agregar otros componentes que ofrece Glue. Le damos clic en Ejecute el trabajo

Trabajo: visualdetection

Acción: Guardar Ejecutar el trabajo Insertar plantilla en cursor Origen Destino Ubicación de destino Transformación

Generar diagrama Spigot

Nombre de la base de datos: parking landing  
Nombre de la tabla: visualdetection

Nombre de la transformación: ApplyMapping

```
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
8 @parameters(['JOB_NAME'])
9 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
10
11 sc = SparkContext()
12 glueContext = GlueContext(sc)
13 spark = glueContext.spark_session
14 job = Job(glueContext)
15
```

Registros Esquema

Después de algunos minutos tendremos en la carpeta de S3, un archivo en formato parquet, que contiene toda la data de la tabla que se encuentra en format csv (17.2mb), realizando la ejecución de un Crawler podemos crear la metadata a partir de este archivo parquet.

aws-glue-lmgs

Información general

🔍 Escriba un prefijo y pulse Intro para buscar. Pulse ESC para borrar.

Cargar Crear carpeta Descargar Acciones

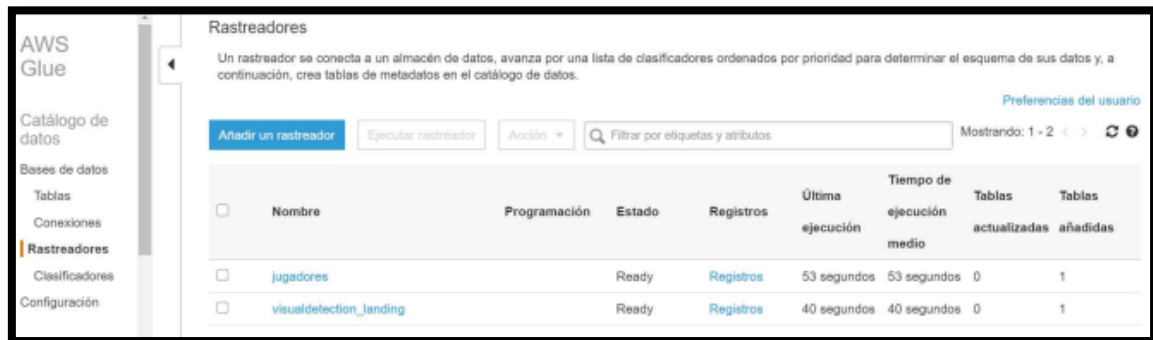
Nombre

part-00000-c1457efc-0a5a-430e-a27a-bf621ed2a059-c000.snappy.par.



Creación de Crawler para generar la metadata a partir del archivo en formato Parquet.

Clic en Rastreadores -> Añadir un rastreador

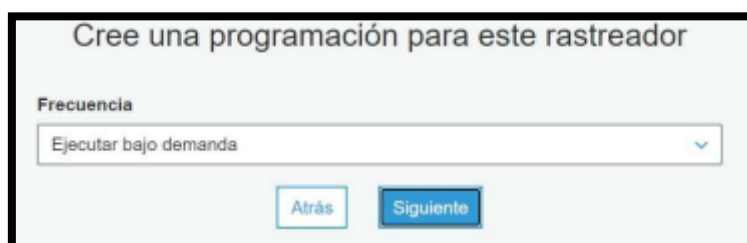


Le asignamos como nombre : visualdetection\_processed (concatenado con el nombre de las iniciales de nuestros nombres), clic en Siguiente.

Seleccionar Data Stores y Siguiente.



Seleccionamos la ruta donde se encuentra el archivo en formato parquet generado por el job ejecutado anteriormente y clic en Siguiente. Siguiente, no y siguiente. Seleccionar el rol que hemos creado, (acordarse de que este rol nos permite acceder a los archivos que se encuentran en S3, en específico realizará la inferencia del archivo en formato parquet), clic en Siguiente. En frecuencia lo dejamos por defecto, y clic en Siguiente.



Crearemos una nueva base de datos en el catálogo de Glue, clic en Añadir una base de datos, y le asignamos parking\_processed (y al último concatenado con las iniciales de nuestros nombres). Clic en Crear. Clic en Siguiente.



### Añadir un rastreador

☒ Información del rastreador

visualdetection\_procesed

☒ Crawler source type

Data stores

☒ Almacén de datos

S3: s3://aws-glue-l...

☒ Rol de IAM

arn:aws:iam::9714893:role/service-role/AWSGlueServiceRole-Crawler

☒ Programación

Ejecutar bajo demanda

☐ Salida

☐ Revisar los logs

#### Configure la salida del rastreador

**Base de datos** ⓘ

Elija una base de datos para incluir tablas

Añadir una base de datos

**Prefijo añadido a las tablas (opcional)** ⓘ

Escriba un prefijo añadido a los nombres de las tablas

► Agrupación de comportamiento para datos de S3 (opcional)

► Las opciones de configuración (opcional)

Atrás **Siguiente**

Veremos el detalle de la configuración del rastreador y clic en Finalizar.

Clic en ¿Desea ejecutarlo ahora?

El rastreador **visualdetection\_processed** se ha creado para ejecutarse bajo demanda. ¿Desea ejecutarlo ahora?

El rastreador ya se está ejecutando, y está realizando la inferencia de los datos en formato parquet que se encuentran en S3, veremos que en unos segundos tendremos una nueva tabla añadida. Efectivamente, se añadió una nueva tabla, ahora podemos consultarla mediante el servicio de Athena.

<input type="checkbox"/>	visualdetection_landing	Ready	Registros	40 segundos	40 segundos	0
<input type="checkbox"/>	visualdetection_processed	Starting		0 segundos	0 segundos	0

Entramos al servicio de Athena y veremos una nueva base de datos con la tabla visualdetection que consume los datos en formato parquet.

Comparaciones en formatos de archivo.

Consulta de la tabla que se encuentra en el esquema landing:

SELECT \* FROM parking\_landing.visualdetection;

### Athena Query editor

Saved queries History Data sources Workgroup: primary

Data source: **AwsDataCatalog** Connect data source

Database: **parking\_processed**

Filter tables and views...

Tables (4)

Create table

New query 1 New query 2 New query 3 New query 4

```

1 SELECT * FROM parking_landing.visualdetection;
2
3 SELECT * FROM parking_processed.visualdetection;

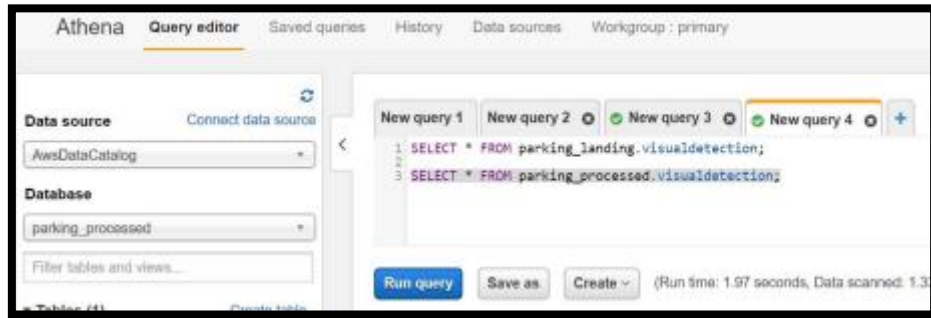
```

Run query Save as Create (Run time: 2:21 seconds, Data scanned)



Ahora realizamos un query a toda la tabla que se encuentran en el esquema processed.

```
SELECT * FROM parking_processed.visualdetection;
```



Vemos la diferencia en la data escaneada que consume cada consulta, siempre usemos tablas en formato columnar como parquet.

### Conclusión

Se realizo el laboratorio, usando los servicios S3 y Athena. Amazon Athena es un servicio de consultas interactivo que facilita el análisis de datos en Amazon S3 con SQL estándar. Athena no tiene servidor, de manera que no es necesario administrar infraestructura y solo paga por las consultas que ejecuta. Athena es sencillo de utilizar.