Proyecto Técnico – Ingeniero de Datos

Nombre del proyecto:  
Sistema de Detección de Fraude en Transacciones Bancarias con Apache NiFi y Python

Nombre completo: Diego Chinchilla

Correo electrónico: diego.chinchilla23@gmail.com

Ciudad/País: Colombia

Rol: Ingeniero de Datos

Especialidad: Big Data, ETL, Arquitecturas Cloud, Apache NiFi, Spark, Python, AWS

Telefono:3133153488

# **Alcance del proyecto y captura de datos**

## Objetivo general.

* Construir una arquitectura escalable en la nube aws que permita capturar, procesar y almacenar datos con el fin de:
* Detectar comportamientos financiero anuales o sospechoso de los usuarios.
* Identificar patrones de cliente por cuenta producto o actividad.

Creación de una base de datos solida robusta para alimentar dashboards y modeles predictivos

# **Casos de uso.**

## Análisis exploratorio y reportes.

Comprender los patrones de comportamientos financieros por categoría ( Domain), ubicación(location) y volumnes de transacciones.

Uso. Cracion de reportes interactivos o dashboards con herramientas de power bi table o amazon quiksigt

## Detección de fraude

Identificar patrones en dominios, ubicaciones, y frecuencias de transacciones.

Uso. Generar alertas tempranas para posibles fraudes bancarios.

## Análisis de datos.

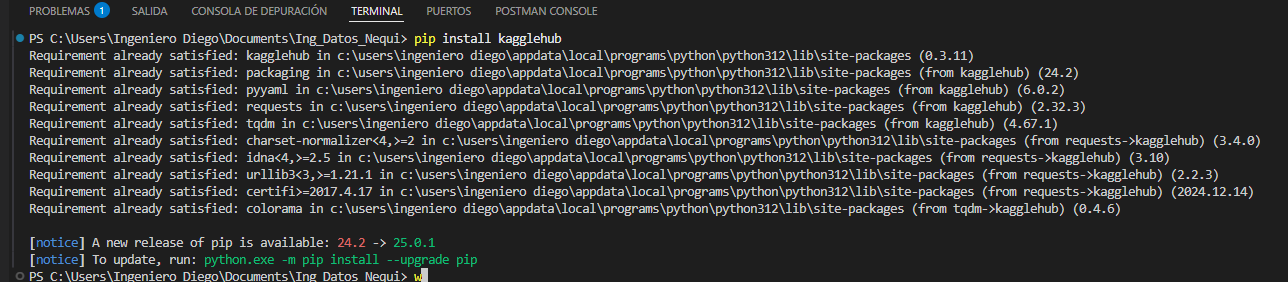
Construcción de una bd como fuente de verdad analítica para la creación de dashboards donde muestre en comportamiento de sectores y regiones la cual servirá como referencia para futuros análisis o auditorias.

## Automatización en nube.

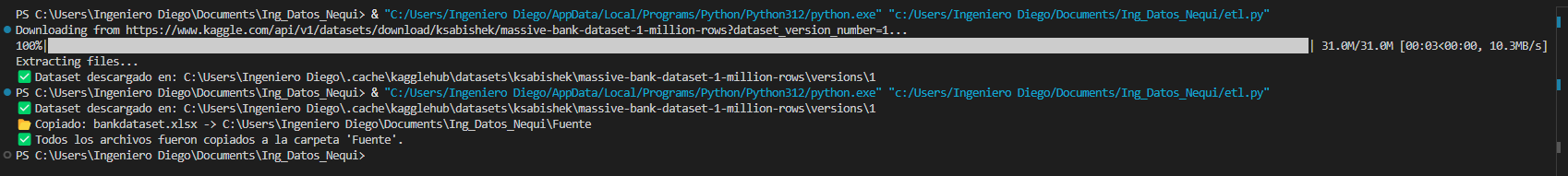
Pensado para poder ejecutarse en pipelines automáticos en aws(glue, s3, athena,Redshift)

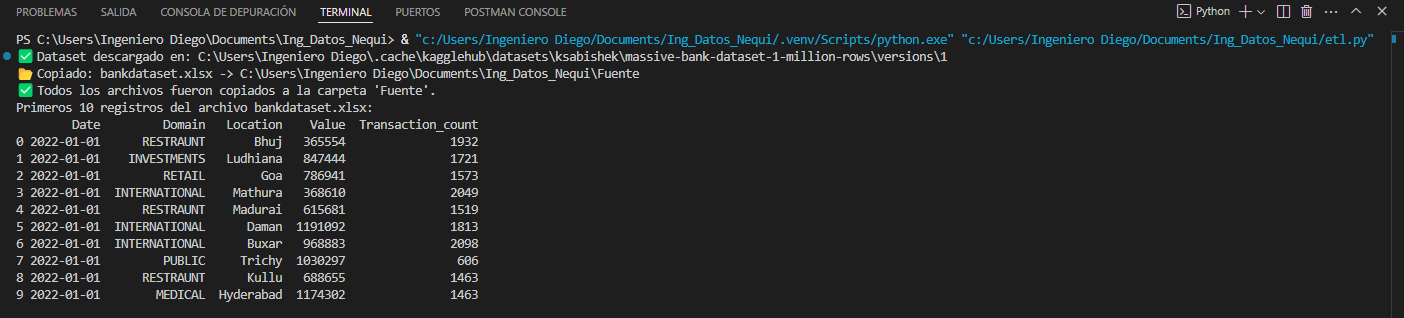
# **Explorar y evaluar los datos, el EDA.**

Instalación



Extracción de datos, el archivo a usar queda alojado en la carpeta de usuario se crea un. chache de donde luego se copia y se para a la carpeta Fuente si no esta creada se crea la carpeta para dejar el dataset en dicha ruta.



Una vez realizada la extracción se procede a generar el proceso de evaluación y muestro del data set , lectura del archivo, donde muestra 10 registros del dataset, los cuales me brindan el esquema de datos que tiene 

Esquema del dataset : Date, Domain, Location, Value, Transaction\_count

Date = Fecha de la trx

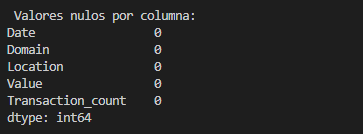
Domain = Categoria o tipo comercio

Location = Ubicación geográfica

Value = Valor de la transacción

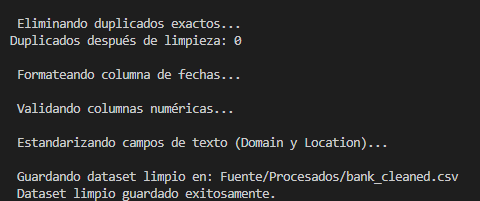
Transaccition\_count = cuenta de transacciones

**Revisión de valores nulos**



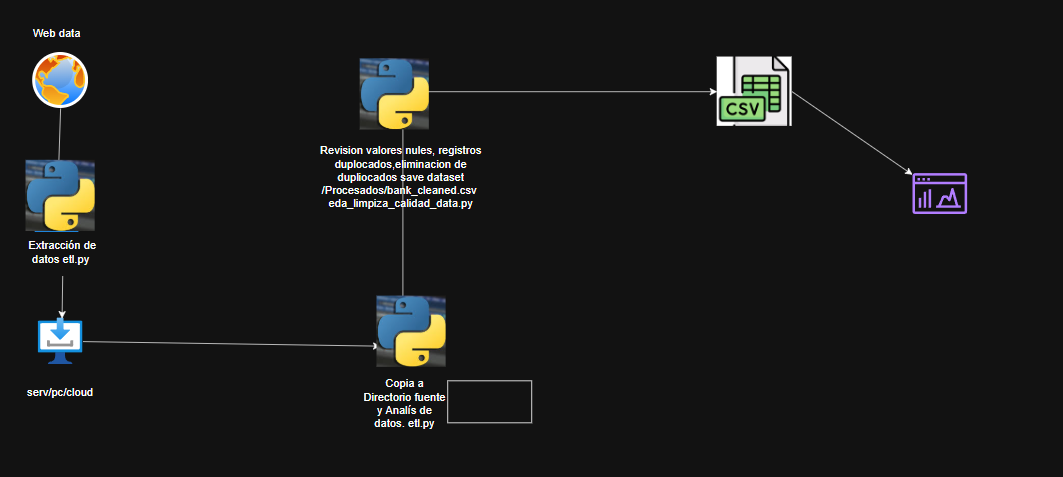
Registros duplicados.





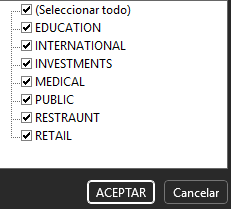
El proceso genera la revisión de y transformación de los datos, validando el esquema que tiene el archivo como se menciono en ocasiones anteriores, luego genera el proceso de validación de registros duplicados, y de presentarse procesara la eliminación de los datos.

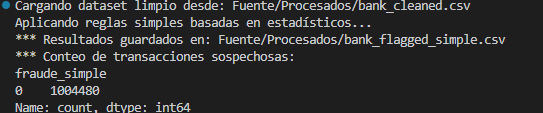
## Diagrama de proceso



## Aplicación del modelo de detección de fraude.

Recordemos que domain hace énfasis a:



Al aplicar el modelo el resultado en esta ocasión es ; 

Donde no se genera novedad y se imprime en pantalla las diferentes salidas donde nos mostrara la ejecicion de cada parte de nuestro codigo , generando un nuevo archivo en formato csv llamado bak\_flgged\_simple.csv el cual contendra las marcaciones o alertas generadas si las presenta.

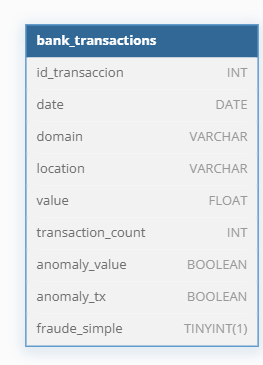
# **Trazar el modelo de datos conceptual y explicar por qué se eligió ese modelo.**

## Modelo de datos conceptual.

Representa cada registro de dataset, enriquecido con información procesada y clasificaciones

## Atributos del modelo

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Atributo** | **Tipo de dato** | **Descripción** |
| id\_transaccion | INT | ID único generado |
| date | DATE | Fecha de la transacción |
| domain | STRING | Categoría del comercio o sector (ej. RETAIL, MEDICAL) |
| location | STRING | Ciudad o ubicación geográfica |
| value | FLOAT | Valor monetario de la transacción |
| transaction\_count | INT | Cantidad de transacciones asociadas a ese registro |
| anomaly\_value | BOOLEAN | Si el valor fue considerado atípico por las reglas |
| anomaly\_tx | BOOLEAN | Si la frecuencia fue atípica |
| fraude\_simple | INT (0/1) | Si fue marcada como sospechosa (1) o no (0) |



**Por que se elige este modelo:**

**Simplicidad:** Este modelo se elige conforme al data set, y al ser muy reducida la muestra que se toma es perecto para el proceso lo que permite integrarse fácilmente con otras herramientas.

**Desnormalizado y eficiencia:** Diseñado para lectura rápida, consultas por dash o bi o micro servicios.

**Escalable:** Si se desea pasar a a modelo relacional pude ampliarse con relaciones fácilmente.

**Trazabilidad y reglas de fraude:** almacenamiento de flags de anomilias para emitir análisis posteriores.

**Ejemplo básico de consulta simple**

SELECT date, domain, location, value

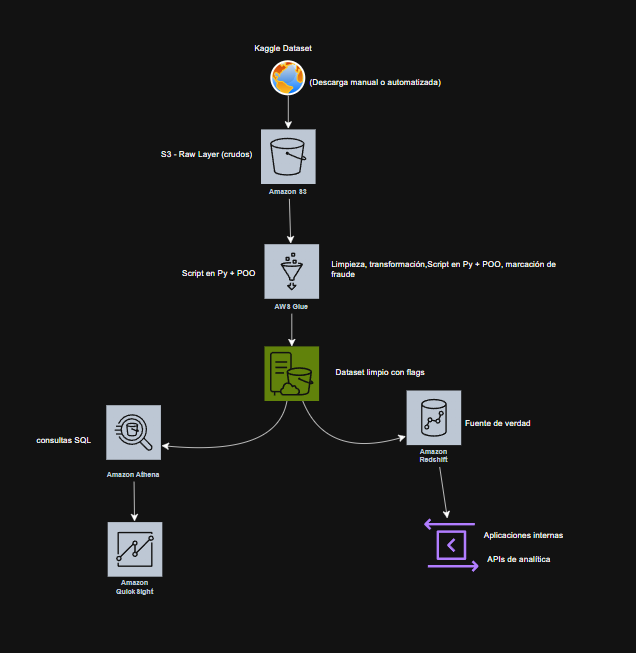
FROM transacciones\_bancarias

WHERE fraude\_simple = 1

ORDER BY value DESC

LIMIT 10;

## Diseñar la arquitectura y los recursos utilizados.



## Indique claramente los motivos de la elección de las herramientas y tecnologías para el proyecto

|  |  |  |
| --- | --- | --- |
| **Recurso** | **Rol en la arquitectura** | **¿Por qué se elige?** |
| Amazon S3 | Almacenamiento de datos crudos y limpios | Escalable, económico y nativo para Glue/Athena |
| AWS Glue | ETL para limpieza, estandarización y marcado | Totalmente serverless, corre Py con POO fácilmente |
| Athena | SQL sobre datos en S3 (raw o procesados) | Ideal para análisis sin mover los datos |
| Redshift | Base de datos analítica (fuente de verdad) | Potente, integrable con BI y dashboards |
| QuickSight | Visualización de KPIs y anomalías detectadas | Dashboard nativo y seguro, directo desde Redshift |
| IAM Roles | Control de acceso a recursos sensibles | Seguridad y trazabilidad |
| (opcional) **Lambda** | Automatización del flujo ETL | Para futuros triggers o procesamiento por evento |

## Proponga con qué frecuencia deben actualizarse los datos y por qué.

Se propone una estrategia escalonada y flexible de actualización, basada en proyección de crecimiento del volumen de datos y la necesidad de detección oportuna de anomalías.

La fase actual, tiene una actualización de cada 24 horas, lo que lo hace ideal para el volumen de transacciones ya que es manejable. (1.000.000), lo que permite generar controles de calidad, validación y reglas de generación de alertas al ciclo de revisión diaria.

También permite minimizar el costo y complejidad de infraestructura.

**Ahora un escenario a largo plazo.**

Ahora bien, ante una posible evolución a volúmenes de datos hasta mil millones de registros diarios se proyecta:

Incrementar la frecuenta de actualización a cada 3 horas en 8 ciclos que es = 8x3=24 cubriendo las 24 horas del día, que nos generara esto mayor granularidad en la detección de novedades, respuesta más rápida a comportamientos sospechosos y reducción del impacto de fraude a operaciones irregulares.

**Resumen de la estrategia planteada**

|  |  |  |
| --- | --- | --- |
| **Frecuencia** | **Escenario** | **Justificacion** |
| Diario (cada 24h) | Actual | Balance entre carga, control y costos |
| Cada 3 horas | Proyección futura | Volumen alto + fluctuación del comportamiento transaccional |

Nota: No solo es pensar en el hoy si no también adelantarnos al mañana, esto nos permite tener mayor planeación y control de eventos que se pudiesen salir del rango del ciclo de vida de las ETL y sus resultados.

# Ejecutar la ETL

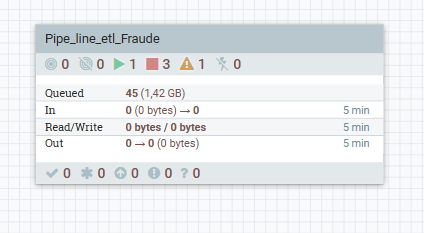
Objetivo.

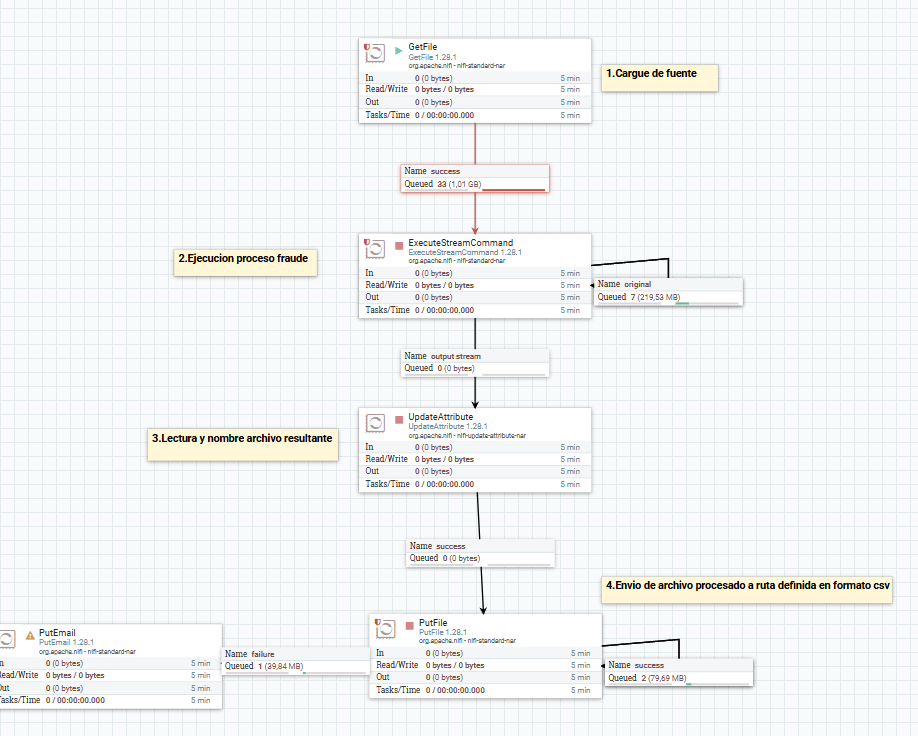
Garantizar que los datos procesados y cargados sean confiables y completos que cumplan reglas de integridad necesaria para entrar en análisis.

## Control de calidad estructural.

¿Que se usó? Apache nifi permite validar los tipos de datos con validación simples dentro del flujo. Al cargar una base de datos relacional como postgresql, Redshif se puede garantizar claves primarias, tipos correctos, restricciones, relaciones entre fk(llaves foráneas) para expansión del modelo.

## Ejecución de pipeline desde apache niffi.





Como vemos se genera la lectura del archivo origen se genera proceso de ejecución de proceso fraude, generación de archivo

