Parte 1– Ingesta de datos

1. Propuesta del Stack de Tecnologías
Para resolver la necesidad y permitir escalar a futuro, se propone el siguiente stack de tecnologías:

Apache Kafka: Para la ingesta de datos en tiempo real y la transmisión de mensajes.
Apache Spark: Para el procesamiento de datos en tiempo real y en batch.
Apache Airflow: Para la orquestación y automatización de flujos de trabajo.
Amazon S3: Para el almacenamiento de datos en un Data Lake.
PostgreSQL: Para el almacenamiento de datos estructurados en una base de datos relacional.
Grafana: Para la visualización y monitoreo de métricas.
Prometheus: Para la recolección y almacenamiento de métricas.
Docker: Para la contenedorización y despliegue de aplicaciones.
Kubernetes: Para la orquestación de contenedores y escalabilidad.

**********

2. Planificación de Requerimientos Previos, Etapas y Plazos de Implementación

Requerimientos Previos
Infraestructura de servidores o servicios en la nube (AWS, GCP, Azure).
Configuración de redes y seguridad.
Instalación y configuración de Docker y Kubernetes.
Instalación y configuración de Apache Kafka, Apache Spark, Apache Airflow, PostgreSQL, Grafana y Prometheus.

Etapas y Plazos de Implementación

Preparación del Entorno (2 días)

Configuración de la infraestructura en la nube.
Configuración de redes y seguridad.
Instalación de Docker y Kubernetes.
Implementación de Apache Kafka (1 semana)

Instalación y configuración de Apache Kafka.
Creación de tópicos necesarios.
Implementación de Apache Spark (2 días)

Instalación y configuración de Apache Spark.
Integración con Apache Kafka.
Implementación de Apache Airflow (3 días)

Instalación y configuración de Apache Airflow.
Configuración de conexiones a Kafka, Spark y PostgreSQL.
Implementación de Almacenamiento en S3 y PostgreSQL (2 días)

Configuración de buckets en Amazon S3.
Instalación y configuración de PostgreSQL.
Desarrollo del Pipeline de Datos (1 semana)

Desarrollo de DAGs en Apache Airflow.
Implementación de scripts de extracción, transformación y carga (ETL).
Implementación de Monitoreo y Alertas (1 semana)

Instalación y configuración de Prometheus y Grafana.
Configuración de dashboards y alertas.
Pruebas y Validación (1 semana)

Pruebas de integración y funcionalidad.
Validación de resultados y ajustes necesarios.

**********

3. Almacenamiento de Tablas para Facilitar el Proceso Analítico

Para facilitar el proceso analítico posterior, se recomienda almacenar las tablas en un formato columnar como Parquet o ORC en Amazon S3. Estos formatos son eficientes en términos de almacenamiento y
permiten un acceso rápido a los datos para análisis. Además, se deben crear índices y particiones
en las tablas de PostgreSQL para optimizar las consultas.

**********

4. Estrategia de Monitoreo del Estado de las Operaciones del Pipeline

Prometheus: Para la recolección de métricas de rendimiento y estado de los componentes del pipeline.
Grafana: Para la visualización de métricas y creación de dashboards personalizados.
Alertas: Configuración de alertas en Grafana para notificar al equipo de soporte en caso de fallos o
anomalías en el pipeline.
Logs: Centralización de logs utilizando herramientas como ELK Stack (Elasticsearch, Logstash, Kibana)
para facilitar la búsqueda y análisis de eventos.

**********

5. Identificación de Puntos de Fallas y Resolución de Incidentes Nivel 1

Identificación de Puntos de Fallas y Resolución de Incidentes

Inicialización (inicializar_task)
Puntos de Falla:
Error en la configuración del entorno.
Recursos necesarios no disponibles.
Resolución:
Verificar los logs para identificar errores específicos.
Asegurarse de que todas las dependencias y recursos estén disponibles y correctamente configurados.
Reiniciar la tarea después de corregir los problemas.

Conexión a Kafka (conexion_kafka_task)
Puntos de Falla:
Fallo en la conexión al servidor Kafka.
Tópicos de Kafka no disponibles o incorrectos.
Resolución:
Verificar la configuración de conexión (host, puerto, credenciales).
Asegurarse de que los tópicos de Kafka estén disponibles y correctamente configurados.
Consultar los logs de Kafka para detalles adicionales.

Tareas de Kubernetes (kubernetes_task)
Puntos de Falla:
Fallo en la ejecución de scripts o comandos.
Recursos de Kubernetes no disponibles.
Resolución:
Verificar los logs de Kubernetes para identificar errores específicos.
Asegurarse de que los recursos de Kubernetes (pods, servicios) estén disponibles y funcionando.
Reiniciar los pods o servicios afectados.

Conexión a Spark (conexión spark task)
Puntos de Falla:
Fallo en la conexión al clúster de Spark.
Configuración incorrecta del clúster.
Resolución:
Verificar la configuración de conexión (host, puerto, credenciales).
Asegurarse de que el clúster de Spark esté operativo.
Consultar los logs de Spark para detalles adicionales.

Transformación de Datos (transformar_datos_task)
Puntos de Falla:
Errores en el código de transformación.
Datos de entrada no válidos.
Resolución:
Verificar los logs para identificar errores en el código de transformación.
Validar los datos de entrada para asegurarse de que cumplen con los requisitos esperados.
Corregir el código de transformación y volver a ejecutar la tarea.

Carga de Datos (cargar_datos_task)
Puntos de Falla:
Fallo en la conexión al destino de carga.
Datos no se cargan correctamente.
Resolución:
Verificar la configuración de conexión al destino de carga.
Asegurarse de que el destino de carga esté disponible y operativo.
Consultar los logs para identificar errores específicos en el proceso de carga.

---

Procedimiento General de Resolución de Incidentes Nivel 1

Monitoreo y Detección:

Utilizar herramientas de monitoreo para detectar fallas en tiempo real.
Configurar alertas para incidentes críticos.
Diagnóstico Inicial:

Revisar los logs y mensajes de error para identificar la causa raíz.
Verificar la disponibilidad de recursos y servicios relacionados.
Resolución:

Aplicar las soluciones mencionadas anteriormente según el punto de falla identificado.
Documentar el incidente y la resolución aplicada para futuras referencias.
Escalamiento:

Si el problema no puede ser resuelto a nivel 1, escalar a un equipo de soporte de nivel superior con toda la información recopilada.

---

Nota:

Se adjunta el esquema de la arquitectura propuesta en el archivo diagrama_pipeline.

se agrega el detalle de cada caja del diagrama detalle_cajas_diagrama_pipeline.txt

se adjunta el código ejemplo general python del Dag de Airflow para la orquestación de las tareas en dag_pipeline.txt

se adjunta un ejemplo más minucioso de los distintos códigos de las tareas que se ejecutan en

el pipeline detalle_dag.txt

Estos 2 ultimos archivos si se cambia a .py se pueden ejecutar en un ambiente de airflow para probar el pipeline

en el archivo dag.monitoreo.txt se detalla un ejemplo de cómo se podría monitorear el pipeline

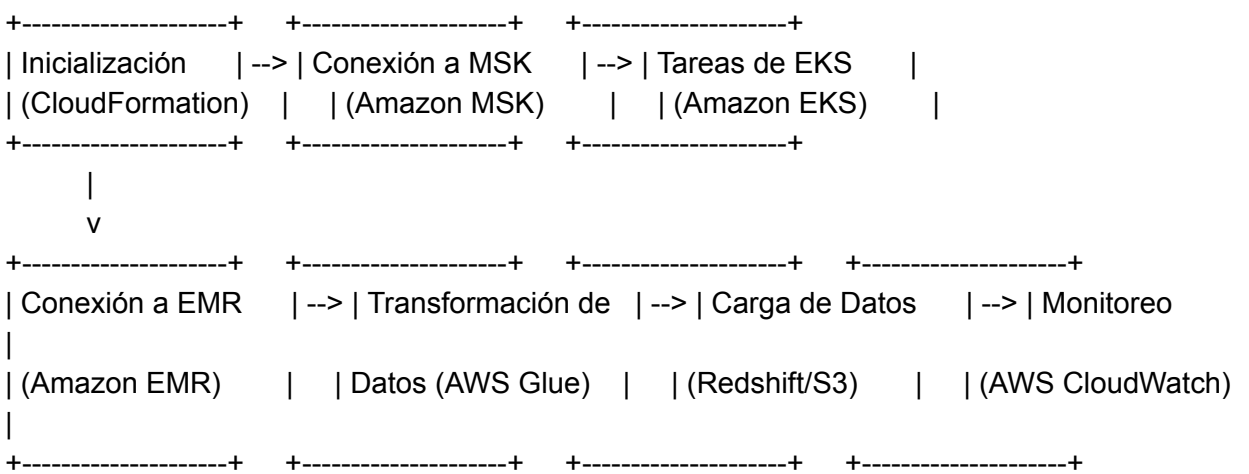
---

##########################################################################################################################

Observación : si uno dispone por ejemplo de los servicios pagos de una plataforma como AWS

_____muchas de las tareas de configuración de infraestructura y servicios se simplifican

por ejemplo :



Sustitución con Servicios de AWS
Inicialización (inicializar_task)

Herramienta Actual: Scripts personalizados o herramientas de configuración.
Sustitución en AWS: AWS CloudFormation
Descripción: AWS CloudFormation permite configurar y aprovisionar recursos de AWS utilizando plantillas en formato JSON o YAML.
Esto puede automatizar la configuración del entorno y preparar los recursos necesarios.
Conexión a Kafka (conexion_kafka_task)

Herramienta Actual: Apache Kafka.
Sustitución en AWS: Amazon Managed Streaming for Apache Kafka (Amazon MSK)
Descripción: Amazon MSK es un servicio completamente administrado que facilita la creación y ejecución de aplicaciones que utilizan
Apache Kafka para el procesamiento de datos en tiempo real.
Tareas de Kubernetes (kubernetes_task)

Herramienta Actual: Kubernetes.
Sustitución en AWS: Amazon Elastic Kubernetes Service (Amazon EKS)
Descripción: Amazon EKS es un servicio administrado que facilita la ejecución de Kubernetes en AWS sin necesidad de instalar y operar
su propio clúster de Kubernetes.
Conexión a Spark (conexion_spark_task)

Herramienta Actual: Apache Spark.
Sustitución en AWS: Amazon EMR (Elastic MapReduce)
Descripción: Amazon EMR proporciona un clúster administrado para ejecutar frameworks de big data como Apache Spark, Hadoop, Presto, y más.
Es ideal para el procesamiento de datos a gran escala.
Transformación de Datos (transformar_datos_task)

Herramienta Actual: Apache Spark o scripts personalizados.
Sustitución en AWS: AWS Glue
Descripción: AWS Glue es un servicio de ETL (Extract, Transform, Load) completamente administrado que facilita la preparación y carga
de datos para análisis. Puede descubrir, catalogar, limpiar, enriquecer y mover datos entre varios almacenes de datos.
Carga de Datos (cargar_datos_task)

Herramienta Actual: Scripts personalizados o herramientas de carga de datos.
Sustitución en AWS: Amazon Redshift o Amazon S3
Descripción:
Amazon Redshift: Es un almacén de datos en la nube completamente administrado que facilita la ejecución de consultas analíticas complejas
sobre grandes volúmenes de datos.
Amazon S3: Es un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, y rendimiento.

Herramientas de Monitoreo de AWS
Amazon CloudWatch

Descripción: Amazon CloudWatch es un servicio de monitoreo y observabilidad para los recursos y aplicaciones de AWS.
Permite recopilar y rastrear métricas, recopilar y monitorear archivos de registro, y configurar alarmas.
Características:
Monitoreo de métricas de rendimiento (CPU, memoria, I/O, etc.).
Configuración de alarmas para notificaciones automáticas.
Visualización de datos a través de dashboards personalizados.
Monitoreo de logs y generación de insights.