MODULO 08 - EJERCICIO 07-A

ALEXIS YURI M.

Diseña un flujo automaizado para orquestar el procesamiento de grandes volúmenes de datos médicos, identificando la plataforma más adecuada (AirFlow, NiFi o Talend) según el caso. Detalla las etapas del flujo, sus características técnicas, los beneficios esperados y las buenas prácticas que aplicarías.

1. Necesidad Principal del Flujo.

Tipos de datos: El flujo procesará grandes volúmenes de datos médicos que incluyen información de pacientes, estudios, recetas y turnos. La información proviene de múltiples sedes y sistemas, y es de naturaleza sensible y crítica.

Objetivo: El objetivo es automatizar y orquestar la consolidación diaria de estos datos para evitar los errores, duplicaciones y demoras que ocurren con la carga manual. La automatización permitirá tener una fuente de datos única y confiable para análisis, reportes y toma de decisiones.

2. Selección de la Plataforma.

Para la orquestación de un flujo diario(batch), complejo y masivo como este, la elección es Apache Airflow.

La justificación está dada por las siguientes consideraciones:

- Orquestación y complejidad: Airflow está diseñado específicamente para gestionar flujos de trabajo complejos. La consolidación de datos clínicos de múltiples fuentes con pasos de validación, transformación y carga es un caso de uso ideal para Airflow.

Procesamiento Batch: Aunque se menciona la carga diaria, es un proceso masivo que ocurre una vez al día, lo que encaja perfectamente con el modelo de procesamiento batch de Airflow.

Perfil técnico: Airflow utiliza Python, que es un lenguaje de programación muy común en los equipos de ingeniería y ciencia de datos. Esto facilita la creación, mantenimiento y extensión de los flujos por parte del equipo técnico.

Escalabilidad: Airflow permite la ejecución de tareas en paralelo y se integra con soluciones de nube y contenedores (Docker, Kubernetes) para manejar cargas de trabajo masivas y escalables.

Además, se descartan la otras 2 opciones por:

NiFi: Es excelente para la ingesta y el enrutamiento de datos en tiempo real (streaming), pero su modelo de flujos de trabajo visual podría volverse muy complejo y difícil de mantener para una orquestación masiva y secuencial con múltiples dependencias como la requerida.

Talend: Es una herramienta ETL muy potente, pero su enfoque es más en las transformaciones de datos en sí mismas. Aunque puede orquestar, Airflow es la solución estándar de la industria para la orquestación y el monitoreo de flujos de trabajo a gran escala.

3. Descripción del Flujo.

El flujo se diseñaría como un DAG en Airflow, donde cada nodo es una tarea y las flechas representan las dependencias.

Ingesta: Los datos se capturan diariamente de diferentes sistemas, que pueden ser bases de datos, APIs de sistemas de gestión hospitalaria o archivos de registros (CSV, JSON, etc.) generados por los sistemas de las sedes.

Las tareas de ingestión se ejecutarían en paralelo. Por ejemplo, una tarea para extraer los datos de pacientes de la sede A, otra para los estudios de la sede B, y así sucesivamente. Se usarían conectores de Airflow para bases de datos (PostgreSQLOperator, MySqlOperator) y APIs (SimpleHttpOperator).

Procesamiento: Una vez extraídos, los datos se someten a una serie de transformaciones y validaciones. Estas tareas solo se ejecutarían si la ingestión correspondiente se ha completado con éxito.

Las tareas de procesamiento serían las siguientes:

- Validación de esquema: Se verifica que el formato de los datos sea el esperado.

- Limpieza de datos: Se eliminan valores nulos y se estandarizan campos (ej. formatos de fecha, nombres de pacientes).

- Deduplicación y consolidación: Se aplican reglas de negocio para identificar y fusionar registros duplicados de un mismo paciente o estudio que provengan de diferentes sedes.

- Enriquecimiento: Se pueden agregar campos de metadatos como el origen del registro.

Entrega: Los datos limpios, validados y consolidados se cargan en un repositorio centralizado.

En este caso el destino sería un Data Lake (ej. S3 en AWS, ADLS en Azure) para almacenar los datos en su formato semi-estructurado o, idealmente, un Data Warehouse (ej. BigQuery, Snowflake) que permita un acceso analítico rápido y eficiente.

Las tareas de carga se ejecutarían después de que todas las transformaciones se hayan completado para cada tipo de dato.

4. Buenas Prácticas.

Control de Versiones: El código del DAG de Airflow debe estar bajo control de versiones (ej. Git). Esto facilita la colaboración, el seguimiento de cambios y el desplieglo en diferentes entornos (desarrollo, pruebas, producción).

Gestión de Errores y Notificaciones: Se deben configurar retries para tareas que fallen temporalmente. Se implementarán notificaciones automáticas (ej. por email o Slack) para alertar al equipo de datos sobre fallas críticas en el flujo, proporcionando el contexto necesario para su rápida resolución.

Seguridad: Dado que se trata de datos médicos, se deben aplicar estrictas políticas de seguridad. Esto incluye el control de acceso basado en roles en Airflow, el cifrado de datos en reposo y en tránsito, y el uso de variables y conexiones secretas para credenciales.

5. Solución en Esquema Textual.

Orígenes de datos:

Clínicas: A, B, C...

Formatos: Bases de datos, APIs, archivos de texto.

Plataforma de Orquestación:

Apache Airflow

Flujo (DAG en Airflow):

- Ingesta (Paralelo):

Tarea 1: Extraccion\_Pacientes\_Clinica\_A

Tarea 2: Extraccion\_Estudios\_Clinica\_B

Tarea 3: Extraccion\_Recetas\_Clinica\_C

Otras tareas de extracción.

- Procesamiento (Secuencial con dependencias):

Validacion\_y\_Limpieza\_Pacientes (depende de Tarea 1)

Validacion\_y\_Limpieza\_Estudios (depende de Tarea 2)

Validacion\_y\_Limpieza\_Recetas (depende de Tarea 3)

Consolidacion\_Global\_Registros (depende de todas las tareas de validación)

- Entrega:

Cargar\_Data\_Warehouse (depende de Consolidacion\_Global\_Registros)

Monitoreo y Reportes:

Envío de notificaciones de éxito o falla.

Visualización del DAG en la interfaz de Airflow.