1) Título:

FileProp Extractor: Sistema de Extracción y Gestión de Propiedades de Archivos para Optimización de Consultas en Base de Datos

2) Objetivo General:

Desarrollar un sistema automatizado para la extracción, transformación y almacenamiento de propiedades de archivos desde carpetas y archivos estructurados, con el fin de construir una base de datos optimizada que permita la gestión eficiente y la mejora en las consultas de la información almacenada.

2.1 Objetivos Específicos:

A. Extraer y transformar datos automáticamente desde archivos y carpetas almacenados en un servidor, capturando propiedades clave como el nombre, tamaño, tipo de archivo, fechas de creación y modificación, y organizar esta información en una estructura coherente.
B. Verificar la integridad de los datos mediante la validación de identificadores únicos (UWI) de pozos, registrando errores y excepciones de manera eficiente en una base de datos separada para mantener la calidad de los datos.
C. Optimizar la consulta de los datos almacenados en la base de datos PostgreSQL, mediante la creación de índices y estructuras de datos adecuadas, permitiendo consultas rápidas y eficaces de la información relevante.

3) Justificación:

En proyectos que manejan grandes volúmenes de datos y múltiples tipos de productos almacenados en archivos y carpetas, identificar y gestionar esta información de manera eficiente se convierte en un desafío. La variedad de tipos de productos y la complejidad de las estructuras de carpetas dificultan el acceso rápido y organizado a los datos relevantes. Además, es crucial garantizar la integridad de los datos y validar que la estructura de carpetas sigue los lineamientos del proyecto.

Este sistema de extracción y gestión de propiedades de archivos aborda estos problemas al automatizar el proceso de extracción de datos, facilitando la identificación precisa de los archivos almacenados, su tipo de producto y otras propiedades clave, como el nombre del pozo y el UWI. La incorporación de mecanismos de validación garantiza que los datos almacenados mantengan su integridad, mientras que el registro de errores asegura el seguimiento de cualquier discrepancia en la estructura.

Finalmente, la optimización de las consultas mediante una base de datos bien estructurada e indexada permite acceder rápidamente a la información, agilizando las búsquedas y mejorando la toma de decisiones. De esta manera, el proyecto no solo centraliza la información, sino que también mejora la eficiencia operativa y el

control de calidad en la gestión de archivos y datos relacionados con pozos petroleros.

4) Metodología:

La metodología para el desarrollo del sistema de extracción, transformación y carga (ETL) de propiedades de archivos y optimización de la base de datos está fundamentada en principios teóricos del procesamiento de datos automatizado, integridad de datos, y mejora en la consulta de grandes volúmenes de información. Para garantizar el éxito de este proceso, se aplican las siguientes fases:

1. Fase de Extracción de Datos:

Basada en el modelo de procesos ETL (Extract, Transform, Load), esta fase se sustenta en teorías de sistemas de archivos distribuidos y la necesidad de minería de datos para recuperar información desde fuentes heterogéneas (archivos en diversas carpetas, con distintos formatos). La automatización de la extracción se fundamenta en la metodología de exploración de estructuras de datos (file system traversal) que permite identificar y recuperar propiedades como nombre, tamaño, extensión y fechas asociadas de cada archivo. La extracción sistemática de los datos permite un análisis y procesamiento más eficiente en fases posteriores.

2. Fase Gestión de Calidad de datos y Normalización de datos:

La metodología incluye la verificación de la integridad y consistencia de los datos extraídos mediante la validación de los UWI (Unique Well Identifier) y otras claves principales, asegurando que los archivos y carpetas cumplen con las especificaciones del proyecto. Este proceso se apoya en técnicas de validación cruzada contra la base de datos de pozos existente, utilizando reglas predefinidas que aseguran que la estructura de carpetas y la información asociada son coherentes y válidas. Además, los datos erróneos o ausentes se registran en una tabla de errores, utilizando principios de gestión de excepciones.

3. Fase de Carga y Optimización:

Esta fase se basa en la teoría de bases de datos relacionales y el diseño eficiente de índices. La información validada y transformada se carga en la base de datos PostgreSQL, siguiendo un modelo estructurado para optimizar la consulta posterior. Se aplican técnicas de normalización y indexación para mejorar la eficiencia en las búsquedas de grandes volúmenes de datos. La teoría de optimización de consultas respalda el uso de índices y estructuras que permiten la búsqueda rápida de información por atributos clave, como el identificador único, tipo de producto, y nombre del archivo.

4. Fase de Control de Calidad y Mejora Continua:

Finalmente, la metodología se sustenta en los principios de auditoría de datos y gestión del ciclo de vida de la información, donde se monitorizan los errores

registrados, se realizan validaciones periódicas de la integridad de los datos almacenados, y se implementan mejoras continuas en el proceso de extracción y carga. Las métricas de rendimiento, como el tiempo de búsqueda y la exactitud de los datos, son elementos clave que permiten ajustar los algoritmos de extracción y optimización.

5) Resultados:

5.1. Base de Datos del Sistema:

El sistema utiliza una base de datos PostgreSQL complementada con la extensión PostGIS, lo que permite almacenar y gestionar información espacial, como la geolocalización de los pozos. Esta característica es fundamental para el manejo eficiente de datos geoespaciales, facilitando la referencia precisa de ubicaciones en el contexto de la información almacenada. La base de datos se organiza en las siguientes tablas clave:

- **Tabla pozos_epis_sgc**: Esta tabla almacena la información espacial y geoespacial relacionada con los pozos, utilizando las capacidades de PostGIS para gestionar las coordenadas y ubicaciones de los pozos petroleros, asegurando que la información esté geográficamente referenciada.
- **Tabla recepción**: Contiene los datos referentes a la recepción de archivos, incluyendo metadatos importantes como fechas de recepción, responsables, y detalles generales de los documentos o archivos recibidos.
- **Tabla directorios**: Esta tabla captura y almacena los resultados del recorrido por los directorios del sistema de archivos, incluyendo información sobre la estructura de carpetas, los archivos encontrados, sus características (nombre, tamaño, fecha de creación, modificación, etc.), así como el tipo de producto y el identificador único.
- **Tabla errores**: Se utiliza para registrar carpetas o archivos que no cumplen con la estructura esperada o que presentan inconsistencias. Esto permite identificar problemas durante el proceso de análisis y facilita el seguimiento de directorios que requieren corrección o revisión.

5.2. Script generado:

Este sistema se implementa mediante un script que automatiza la extracción de datos desde directorios de red, gestionando los archivos en una base de datos PostgreSQL. Las áreas clave son:

- **Instalación de Librerías**:

    - pip install pillow psycopg2 exifread: Instala las librerías necesarias para manipular imágenes, interactuar con PostgreSQL y extraer metadatos EXIF.
    - pip install pywin32: Facilita la interacción con las API de Windows para operaciones como el mapeo de red.
    - pip install psycopg2: Instala el adaptador para conectarse a bases de datos PostgreSQL.
- **Mapeo de Red y Acceso a Carpetas**: Verifica si una carpeta de red es accesible y cuenta las carpetas en la primera posición para identificar proyectos.
- **Descripción de Funciones del Código de Extracción**:
    - **conectar_base_datos()**: Se conecta a PostgreSQL.
    - **insertar_en_base_de_datos_batch(conn, batch)**: Inserta filas en la tabla directorios en lotes.
    - **registrar_error(conn, uwi_upi, nombre_pozo, ruta_archivo)**: Registra errores en la tabla errores.
    - **verificar_uwi(conn, uwi_upi)**: Verifica si el UWI está en la tabla pozos_epis_sgc.
    - **procesar_carpetas(carpeta_base)**: Recorrerá los directorios de la red, procesando archivos y subcarpetas.