Proyecto de la UT01

PARTE I: PROCESO ETL CON PENTAHO Y AWS S3

Apartado 1: Instalación y configuración de Pentaho Data Integration

En este primer apartado deberás instalar **Pentaho Data Integration** en tu equipo y verificar que el entorno funciona correctamente. Descarga la última versión disponible de Pentaho Data Integration desde la página oficial y asegúrate de tener instalado Java JDK 8 o superior con las variables de entorno correctamente configuradas. Una vez completada la instalación, ejecuta la aplicación **Spoon** para comprobar que se abre sin problemas y está lista para trabajar.

Como evidencia de este apartado deberás entregar una captura de pantalla de Spoon abierto en tu entorno de trabajo. Acompaña la captura con una breve explicación indicando la versión de Java que utilizas, la versión de Pentaho instalada, el sistema operativo de tu equipo y una confirmación de que la aplicación queda operativa para comenzar el proyecto ETL.

Apartado 2: Extracción de datos desde Kaggle

El segundo apartado consiste en realizar la fase de **extracción** del proceso ETL obteniendo un dataset real desde un repositorio público. Para ello deberás registrarte en la plataforma Kaggle si aún no tienes cuenta. Una vez dentro, explora los conjuntos de datos disponibles y selecciona libremente cualquier dataset en formato CSV que te parezca interesante para trabajar. Puede ser sobre cualquier temática: deportes, economía, salud, redes sociales, comercio electrónico o cualquier otro tema que despierte tu curiosidad. Descarga el dataset seleccionado y guárdalo en una carpeta de trabajo organizada en tu equipo local.

Deberás entregar como evidencias una captura de pantalla del dataset tal como aparece en la página de Kaggle, donde se vea claramente el nombre, la descripción y las características principales del conjunto de datos. También incluye una captura del archivo CSV ya descargado en tu sistema de archivos local. Finalmente, escribe un texto breve justificando por qué has elegido ese dataset en particular y qué características lo hacen interesante o adecuado para este proyecto.

Apartado 3: Transformación y limpieza de datos en Spoon

Ahora crearás tu **primera transformación** en Pentaho orientada específicamente a la limpieza del dataset que has seleccionado. Abre Spoon y crea una nueva transformación. Utiliza el componente "CSV file input" para cargar tu archivo y analiza detenidamente los datos para identificar problemas de calidad. Busca valores nulos o vacíos, datos inconsistentes, campos mal formateados, tipos de datos incorrectos o registros duplicados. Una vez identificados los problemas, aplica las técnicas de limpieza apropiadas utilizando los componentes que ofrece PDI como "Filter rows" para eliminar registros que no cumplan ciertos criterios, "Select values" para convertir tipos de datos, "Replace in string" para corregir formatos, "If field value is null" para tratar valores nulos o cualquier otro componente que consideres necesario para dejar tus datos limpios y preparados.

La transformación debe quedar correctamente construida y ejecutarse sin errores. Como evidencias entregarás el archivo .ktr de la transformación, una captura del flujo completo tal como se ve en Spoon y un documento explicativo donde describas qué problemas concretos tenía tu dataset original, qué técnicas de limpieza has aplicado y por qué, qué componentes específicos de PDI has utilizado y algunas estadísticas básicas como el número de registros antes y después de la limpieza o cuántos campos has modificado.

Apartado 4: Creación de un Job de orquestación

En Spoon, crea un nuevo Job e incorpora la transformación de limpieza que creaste en el apartado anterior. Ahora deberás añadir al menos un paso adicional que realice cálculos sobre porcentajes, agregaciones o métricas relevantes de tus datos transformados. Por ejemplo, podrías calcular el porcentaje de registros por cada categoría, obtener medias o totales de campos numéricos, o cualquier otro cálculo que tenga sentido para tu dataset. Configura el flujo de ejecución del Job estableciendo las condiciones apropiadas para manejar tanto los casos de éxito como los de fallo. Ejecuta el Job completo y verifica que todo el proceso se completa correctamente sin errores.

Entregarás el archivo .kjb del Job creado, una captura de pantalla mostrando la secuencia completa del Job en Spoon y el log de ejecución que demuestre que se ha ejecutado con éxito. Acompaña estos archivos con una explicación describiendo la lógica que has seguido para construir el flujo del Job, qué pasos has incluido y cuál es la función de cada uno, qué transformaciones adicionales has realizado para calcular porcentajes o métricas, y cómo has implementado el manejo de posibles errores.

Apartado 5: Carga de resultados en Amazon S3

En este apartado completarás la fase de **carga** del proceso ETL almacenando los datos ya transformados y limpios en un bucket de Amazon S3. Modifica el Job que creaste anteriormente para que genere un archivo de salida final con los datos procesados, puede ser en formato CSV u otro formato que consideres apropiado. Una vez dentro de AWS, accede al servicio S3 y crea un bucket con un nombre único y la configuración que consideres adecuada. Sube el archivo generado por tu Job a este bucket de S3, puedes hacerlo a través de la consola web de AWS o si dispones de plugins configurados en PDI, directamente desde la herramienta. Una vez subido el archivo, verifica su integridad comparando el tamaño del archivo local con el que aparece en S3 para asegurarte de que la carga se ha realizado correctamente.

Como evidencias deberás entregar una captura de pantalla del bucket de S3 donde se vea claramente el archivo cargado junto con su ruta completa. Incluye también una captura que muestre los detalles del archivo como su tamaño, fecha de carga y otras propiedades. Proporciona además alguna verificación de integridad, por ejemplo, comparando los tamaños de archivo entre el local y el de S3. Finalmente, escribe una breve explicación del método que has utilizado para realizar la carga y comenta cualquier dificultad que hayas encontrado durante el proceso y cómo la has resuelto.

PARTE II: BASES DE DATOS NoSQL - MONGODB

Apartado 6: Consultas en MongoDB Atlas con la colección Movies

Regístrate en **MongoDB Atlas** accediendo a su página web y crea un **clusternombre** gratuito, una vez creado el cluster, carga la base de datos de ejemplo que MongoDB proporciona, específicamente la que incluye la colección "**movies**" dentro de la base de datos **sample_mflix**. Conecta a tu base de datos utilizando la shell de mongo. Ahora deberás realizar 5 consultas sobre esta colección de películas:

1. **Primero**, obtén todas las películas que se estrenaron en el año 2010, mostrando únicamente los campos título, año y género.
2. **Segundo**, cuenta cuántas películas tienen una puntuación en IMDb superior a 8.0.
3. **Tercera** consulta consistirá en encontrar todas las películas que contengan tanto "Drama" como Romance" simultáneamente en su array de géneros, y deberás ordenar los resultados por puntuación IMDb de forma descendente.
4. **Cuarta** consulta, busca las películas donde el array de directores contenga exactamente dos elementos y que además tengan al menos un actor ganador de un Oscar en su reparto, utilizando el operador $size para contar elementos del array y trabajando también con el array de actores.
5. **Quinta** consulta agregada utilizando el pipeline de agregación de MongoDB. Deberás calcular el número promedio de comentarios por película para cada año, mostrando únicamente aquellos años que tengan más de 50 películas, y ordenar los resultados por el promedio de comentarios de mayor a menor.

Para cada una de las cinco consultas entregarás el código MongoDB completo que has ejecutado, una captura de pantalla mostrando el resultado obtenido y una breve explicación de qué hace la consulta y qué operadores o técnicas has utilizado. Además, incluye capturas de pantalla del cluster creado en MongoDB Atlas para demostrar que has completado el registro y configuración inicial. Finalmente, redacta un documento resumen donde reflexiones sobre las principales diferencias que has encontrado entre trabajar con bases de datos relacionales tradicionales y MongoDB, comentando aspectos como la flexibilidad del esquema, el trabajo con documentos anidados y arrays, y la sintaxis de las consultas.

PARTE III: BASES DE DATOS DE GRAFOS CON NEO4J

Apartado 7: Grafo de municipios de Gran Canaria

En este apartado crearás y analizarás un grafo completo que represente los **21 municipios** de Gran Canaria y las relaciones entre ellos basadas en su proximidad geográfica. Comienza instalando Neo4j Desktop en tu equipo o utiliza Neo4j Sandbox si prefieres trabajar en la nube. Crea una nueva base de datos de grafos y genera 21 nodos, uno por cada municipio de la isla.

Una vez creados los nodos, deberás establecer relaciones entre aquellos municipios que comparten frontera física. Cada relación debe incluir como propiedad la distancia aproximada en kilómetros entre los núcleos urbanos de ambos municipios. Investiga qué municipios son limítrofes entre sí y estima las distancias utilizando herramientas como Google Maps o similar. Con el grafo completamente construido:

1. Aplica el algoritmo de camino mínimo para calcular cuál sería la ruta más corta que te permitiría visitar todos los municipios de la isla, simulando un problema similar al del viajante.
2. Calcula la centralidad de grado (Degree Centrality) para identificar qué municipios tienen más conexiones directas con otros.
3. Calcula la centralidad de intermediación (Betweenness Centrality) para descubrir qué municipios actúan como puentes importantes en la red.
4. Aplica la centralidad de cercanía (Closeness Centrality) para encontrar qué municipios están más cerca del resto en términos de la red.
5. Ejecuta el algoritmo PageRank para determinar la importancia relativa de cada municipio en la red.
6. Calcula el coeficiente de agrupamiento (Clustering Coefficient) para analizar cuán interconectados están los vecinos de cada municipio.

Entregarás el script completo de Cypher que has utilizado para crear tanto los nodos como todas las relaciones del grafo. Incluye una captura de la visualización completa del grafo tal como se ve en Neo4j Browser. Para cada una de las medidas calculadas, entrega el código Cypher ejecutado, una tabla mostrando los cinco municipios principales según esa medida específica y una explicación interpretando qué significa ese resultado en el contexto geográfico real de Gran Canaria. Cierra este

apartado con un documento de conclusiones donde analices qué municipios resultan ser más centrales según los diferentes criterios aplicados y reflexiona sobre las implicaciones prácticas que estos resultados podrían tener, por ejemplo, para planificación de servicios públicos, infraestructuras o desarrollo territorial.

Apartado 8: Grafo de empresas y cálculo de rutas óptimas

Para este apartado crearás un grafo que represente empresas ubicadas en diferentes puntos de Gran Canaria y calcularás caminos mínimos para planificar rutas óptimas de visitas de seguimiento. Listado de empresas:

1. SERVI BYTE CANARIAS S.L. / SAN BARTOLOMÉ DE TIRAJANA
2. FOTÓN SISTEMAS INTELIGENTES S.L. / LAS PALMAS DE GRAN CANARIA
3. GRUPO CAPISA GESTIÓN Y SERVICIOS, S.L. / LAS PALMAS DE GRAN CANARIA
4. AYUNTAMIENTO DE AGÜIMES / Ayto. Agüimes
5. CENTRUM DISTRIBUCIÓN Y SERVICIOS INSULARES S.L. / AGÜIMES
6. Laberit Canarias S.L. / LAS PALMAS DE GRAN CANARIA
7. GLOBAL (SALCAI-UTINSA S.A.) / LAS PALMAS DE GRAN CANARIA
8. AEROLASER SYSTEMS S.L. / AeTechCenterAgüimes
9. RED IMPULSA FORMACIÓN S.L. / LAS PALMAS DE GRAN CANARIA
10. INSTITUTO TECNOLÓGICO DE CANARIAS / AGÜIMES
11. HOSTELCO CONSULTORES S.L. / TELDE
12. INSTITUTO TECNOLÓGICO DE CANARIAS / LAS PALMAS DE GRAN CANARIA
13. CAINSER, S.A. / LAS PALMAS DE GRAN CANARIA
14. GERENCIA MUNICIPAL DE CULTURAL Y DEPORTES DE SANTA LUCÍA S.A. / Ateneo Municipal
15. GRUPO PÉREZ MORENO, S.L. / LAS PALMAS DE GRAN CANARIA

Crea un nuevo grafo en Neo4j donde cada nodo represente una empresa. Cada nodo debe tener propiedades que incluyan al menos el nombre de la empresa, el sector al que pertenece y el municipio donde está ubicada. Establece relaciones entre las empresas basándote en su proximidad geográfica, de forma que cada relación incluya como propiedad la distancia en kilómetros entre ambas empresas. No es necesario que todas las empresas estén conectadas con todas las demás, simplemente crea conexiones lógicas basándote en cuáles están relativamente cerca unas de otras.

Una vez construido el grafo, tu objetivo será calcular el camino mínimo para visitar todas las empresas partiendo desde el CIFP Villa de Agüimes como punto de partida, simulando la planificación de una

ruta comercial. Aplica el algoritmo de Dijkstra para encontrar los caminos más cortos desde tu empresa de partida hacia todas las demás. Utiliza también la función `Shortest` Path para calcular la ruta más corta entre pares específicos de empresas. Si encuentras que existen rutas alternativas entre algunos puntos, utiliza `All Shortest Paths` para identificarlas y compararlas.

Como evidencias proporciona el script Cypher completo que has utilizado para crear el grafo con todos sus nodos y relaciones. Incluye una captura de la visualización del grafo de empresas tal como se muestra en Neo4j Browser. Para cada algoritmo que hayas aplicado entrega el código Cypher correspondiente y el resultado obtenido, comparando las diferentes rutas y distancias calculadas. Finalmente, redacta una propuesta de ruta óptima donde justifiques cuál sería la mejor secuencia para visitar todas las empresas, explicando tu razonamiento y calculando cuántos kilómetros de ahorro supondría seguir esta ruta optimizada en comparación con una ruta no planificada.

Apartado 9: Análisis avanzado de grafos - Parte I

El apartado se divide en dos tareas que deberás completar de forma independiente.

1. La primera tarea consiste en trabajar con caminos mínimos en una red de ciudades europeas. Importa en Neo4j el grafo de ciudades que está en el **repositorio de BIU**, que representa ciudades europeas conectadas por carreteras.
   a. Ejecuta el algoritmo de **Dijkstra** tomando como ciudad origen Doncaster y calcula las distancias mínimas desde Doncaster hacia todas las demás ciudades de la red.
   b. Ejecuta el algoritmo de **Dijkstra** específicamente para el par de ciudades Felixstowe y Utrecht, obteniendo el camino mínimo entre ellas junto con la distancia total.
   c. Ejecuta el algoritmo A* para el mismo par de ciudades Felixstowe y Utrecht. El algoritmo A* es más avanzado que Dijkstra porque utiliza heurísticas para mejorar la eficiencia.
   d. Redacta un análisis comparativo donde discutas las diferencias entre los resultados que has obtenido con Dijkstra y con A*, explica las ventajas y desventajas que has observado.
2. La segunda tarea de este apartado consiste en calcular medidas de centralidad sobre un grafo de dependencias entre librerías **Python**. Accede al repositorio oficial disponible en GitHub en la ruta https://github.com/neo4j-graph-analytics/book/tree/master/data y descarga los archivos **sw-nodes.csv** que contiene los nodos representando librerías Python, y **sw-relationships.csv** que contiene las relaciones de dependencia entre estas librerías.

a. Una vez importado el grafo, calcula las siguientes **medidas de centralidad**. (Grado de dependencia, Grado de cercanía y Grado de intermediación)

Entregarás los scripts Cypher que has utilizado para importar los archivos CSV en Neo4j y una visualización de una muestra representativa del grafo para que se vea la estructura de dependencias. Para cada medida de centralidad calculada proporciona el código, una tabla con las 10 librerías principales según esa medida específica y si es posible algún gráfico o visualización de los resultados. Redacta un documento de análisis donde discutas qué librerías resultan ser más centrales según cada métrica, explica qué significa en este contexto que una librería tenga alta centralidad, compara las diferencias entre las distintas medidas de centralidad y qué información diferente aporta cada una, reflexiona sobre las implicaciones prácticas preguntándote qué librerías serían críticas si estuvieras gestionando un proyecto de software.

Apartado 10: Análisis avanzado de grafos - Parte II

Al igual que el apartado anterior, se divide en dos tareas independientes que exploran diferentes aspectos del análisis de grafos.

1. **Detección de comunidades** sobre el grafo de dependencias entre librerías Python, calular las medidas de conteo de triángulos, coeficiente de Clustering y componentes fuertemente conexas. Una vez ejecutados todos los algoritmos, analiza las comunidades detectadas buscando patrones y características comunes dentro de cada grupo.

2. **Predicción de enlaces.** Utiliza el grafo de ciudades europeas, tu objetivo será aplicar diferentes métricas de predicción de enlaces para determinar cuán probable sería que se construyera una nueva carretera entre las ciudades de Den Haag y Utrecht. Calcula la métrica de vecinos comunes, el coeficiente de vinculación preferencial, la métrica de vecinos totales y la asignación de recursos. Para poder interpretar correctamente estos valores, no te limites únicamente al par Den Haag-Utrecht, sino que deberás calcular las mismas métricas para al menos tres pares de ciudades que ya están conectadas en el grafo y tres pares de ciudades que no lo están, de forma que puedas comparar los scores obtenidos y tener un punto de referencia.

FORMATO DE ENTREGA

Deberás entregar toda la práctica en un único documento PDF que incluya una portada con tus datos personales, el nombre del curso y la fecha de entrega. Incorpora un índice detallado al principio del documento. Desarrolla los 10 apartados en orden, incluyendo todas las evidencias solicitadas para cada uno de ellos. Finaliza el documento con unas conclusiones generales del proyecto donde reflexiones sobre los conocimientos que has adquirido, las dificultades que has encontrado durante el desarrollo y cómo las has superado, las aplicaciones prácticas que ves para las tecnologías que has trabajado y tu valoración personal sobre el proyecto completo.

Además del PDF, entregarás una carpeta comprimida conteniendo todos los archivos generados durante la práctica. Esta carpeta debe incluir los archivos .ktr y .kjb de Pentaho, los datasets que has utilizado, todos los scripts de MongoDB y Neo4j que has ejecutado, y los archivos CSV u otros formatos que hayas generado durante el proceso.