

CURSO: Datos Masivos y Minería de Datos

PROFESOR: PhD David Diaz

SEMESTRE: PRIMAVERA 2022

ALUMNO: Francisca Barros Ávila, Daniel Estay Colistro

**Trabajo Final**

**Fecha de entrega: lunes 06 de marzo**

**Pregunta 1**

Explique con sus palabras cuáles son las principales diferencias y

similitudes entre:

• Una base de datos transaccional: es un tipo de base de datos optimizada para el registro de transacciones, abarcan desde sitios web hasta bases de datos de organizaciones bancarias, tiendas o retail, se destacan por leer y escribir filas de datos concretas de manera muy rápida sin debilitar la integridad de los datos. Las bases de datos transaccionales son almacenes de filas, lo que significa que los datos se almacenan en discos como filas, en vez de como columnas. Los almacenes de filas son la solución ideal para consultar toda la información de un cliente en la tabla de usuarios, ya que puedes buscar solo los datos que necesitas.

Ejemplo: base de datos de las ventas de un supermercado, por local, numero de transacción, productos comprados, monto de la compra, medio de pago etc.

• Un Data Warehouse: es un sistema de base de datos que se construye encima de todas las bases de datos OLTP (Online Transactión Processing) y que se usa típicamente para ser la fuente de datos de los sistemas de Business Intelligence. El Data Warehouse (DWH) obtiene la información de los sistemas internos y externos, limpia los datos, los unifica, les aplica reglas de negocio OLAP (Online Analytical Processing) creando así una capa de datos optimizada para ejecutar análisis de datos. Al contrario de una Base de Datos Transaccional un DWH suele tener el histórico de todos los datos (por ejemplo no contiene solo la dirección actual de un cliente, sino todas las dirección que ha ido obteniendo a lo largo de los años). En resumen un DWH es una base de datos con una estructura OLAP que se encuentra por encima de las bases de datos transaccionales y esta diseñada para el análisis de datos.

Entre las similitudes entre una base de datos relacional y un Data Warehouse, tenemos que ambas son bases de datos orientadas a datos estructurados, no obstante como diferencia es que la base de datos transaccional tiene un tipo de arquitectura que esta optimizada para realizar operaciones transaccionales (inserción, borrado, actualización) cuya arquitectura sigue un diseño OLTP (Online Transaction Processing) en cambio un Data Warehouse, como la operación fundamental es la consulta, la arquitectura OLPT no es óptima ya que la escritura de sentencias SQL para consultar la información puede ser compleja y eso hace que use otra arquitectura como es OLAP (Online Analytical Processing)

Ejemplo: registro de las ventas de un supermercado durante el año 2022, indicando tendencia, estacionalidad etc.

• Un Data Lake: A diferencia de una base de datos transaccional o un Data Warehouse, que tienen datos estructurados, debido a que enm los procesos de negocios se puede tener información que no sea estructurada y que a veces llegan en tiempo real, este tipo de datos antes del desarrollo de las técnicas modernas de Big Data, era muy complejo poder analizarlo, por ejemplo datos con:

* Fotografías, videos que actualmente pueden ser analizados por métodos de inteligencia artificial.
* Comentarios en redes sociales, que actualmente pueden ser procesados por algoritmos de procesamiento de lenguaje natural.
* Datos de sensores y dispositivos IoT, que típicamente transmiten datos a ficheros (a este tipo de datos se le considera como semiestructurado)

Em resumen, un Data Lake es un repositorio de datos centralizado que puede almacenar tanto datos estructurados como no estructurados, en tiempo real o no, donde se almacenan tal cual el sistema de origen, es decir, sin aplicar ningfun tipo de transformación entre ellos

Un DWH y un Data Lake tienen ciertas características en común y algunas diferencias importantes.

Similitudes:

Son repositorios centralizados de datos.

Pueden almacenar datos estructurados.

Pueden gestionar datos estructurados en tiempo real.

Diferencias:

Un DL almacena datos no estructurados.

Un DL no transforma los datos, sino que los almacena tal cual le llegan.

Un DL no almacena los datos con una estructura óptima para su consulta (OLAP).

Un DL tiene capacidad (de procesamiento y de almacenamiento) para gestionar datos no estructurados en tiempo real.

Ejemplo de Data Lake: Innoalimen es un proyecto en el ITAINNOVA participa desarrollando una herramienta de monitorización y análisis de Redes Sociales para determinar qué temas relacionados con el ámbito de la alimentación saludable están candentes en cada momento.

• Un Lake House: es un concepto nuevo y en evolución, que agrega capacidades de gestión de datos sobre un lago de datos tradicional. En esencia, se trata de la combinación de un lago de datos y un almacén de datos. Un data lake house es una arquitectura abierta de gestión de datos que combina las ventajas de flexibilidad y escalabilidad de un Data Lake con las estructuras y características de gestión de datos de un almacén de datos. La arquitectura de data lake house ofrece un buen número de ventajas:

1. Elimina los trabajos más sencillos de extracción, transformación y carga (ETL) porque los motores de consulta están conectados directamente con el Data Lake.

2. Permite conectar directamente diversas herramientas de análisis e inteligencia empresarial.

3. Simplifica la gobernanza de datos porque los datos confidenciales ya no tienen que moverse de un conjunto de datos a otro y se pueden gestionar desde un único punto.

Asimismo, ayuda a reducir los costes porque los datos se pueden almacenar en una ubicación mediante el almacenamiento de objetos.

Ejemplo: Soluciones data lake house de HPE

HPE Ezmeral Unified Analytics es la primera solución nativa de la nube que incorpora el análisis de Apache Spark basado en Kubernetes y la simplicidad de los data lake houses unificados usando Delta Lake a nivel local

**Pregunta 2.**

• Explique con sus palabras cuales son los drivers o causas que hacen necesario la utilización de “clústeres de computadoras”

En la década pasada debido al aumento en la necesidad de procesamiento, los procesadores de consumo general basaban en aumento en su capacidad de computo en el aumento en la frecuencia en un mismo núcleo. Cada núcleo podía ejecutar solo una tarea a la vez con un tiempo entre la tarea previa y la presente (tiempo de ejecución). Esta variable es el resultado de multiplicar el número de instrucciones por el tiempo de ejecución promedio y el único factor que podría influir era el aumento de la frecuencia. El problema es que el aumento de la frecuencia esta directamente relacionada con el aumento del consumo y por ende a la temperatura generada por la CPU. Si aumentamos la frecuencia se aumenta el consumo y el calor, entonces la solución sería obviamente disminuir el voltaje que además influye doblemente en la potencia. Si el voltaje se disminuye haciendo transistores más pequeños que puedan hacer lo mismo con menos energía de paso. Aquí finalmente entra en juego la ley de Moore, (la densidad de transistores se duplica cada 18 o 24 meses en los procesadores debido a la disminución del proceso de fabricación)n. Esto efectivamente ha sido así, al menos en AMD, ya que Intel lleva bastante tiempo atascado en los 14 nm.

Una forma de cómputo de alto rendimiento que puede evitar el problema del aumento de la frecuencia de las CPU y el consumo de energía (y generación de calor), puede ser por medio de la paralelización de tareas, de forma en que muchas instrucciones se ejecutan en forma simultánea operando sobre el principio de dividir un problema muy grande en mochos problemas más pequeños que luego son resueltos simultáneamente

Los procesos que son altamente paralelizables, en el cual la memora RAM puede ser compartir como recursos entre las distintas máquinas o Cores, la ventaja es que usando muchos Cores realizando tareas que se ejecutan simultáneamente pueden logar una capacidad de procesamiento mayores que trabajando en forma aislada, dicho de otra forma, hace que no sea necesario el uso de CPU demasiado poderosas para procesar una tarea, sino que múltiples Cores trabajando en forma paralela, para ello se debe lograr que múltiples CPU o Cores trabajen en conjunto para una tarea asignada, que puedan tener acceso a una memora RAM compartida y que el programa que se este ejecutando este construido para sacar provecho a esta arquitectura.

Sin embargo, no todas las tareas son adecuadas para ser descompuestas para el procesamiento paralelo. La ganancia global en velocidad de computación dependerá de la proporción de tareas del programa que puedan subdividirse

• Refiérase especialmente al uso de HPC (clústers intensivos en cómputo) vs al uso de clústers para Big Data (tipo Hadoop o Spark) • Proponga un ejemplo o caso de uso para cada uno de ellos.

La HPC permite a los usuarios procesar grandes cantidades de datos más rápido que una computadora estándar, lo que genera conocimientos más rápidos y ofrece a las organizaciones la capacidad de mantenerse por delante de la competencia. Las soluciones HPC pueden ser un millón de veces más potentes que la computadora portátil más rápida. Esta potencia permite a las empresas ejecutar grandes cálculos analíticos, como diferentes millones de escenarios que hacen uso de hasta terabytes (TB) de datos. Por ejemplo, la planificación de escenarios requiere grandes cálculos analíticos que proporciona una HPC, como la previsión meteorológica o las evaluaciones de gestión de riesgos. Las organizaciones también pueden ejecutar simulaciones de diseño antes de construir físicamente elementos como chips o automóviles. En resumen, la HPC potencia un rendimiento superior, lo que permite a las empresas hacer más gastando menos.

Una forma de obtener esto es por medio del diseño de un sistema computacional de alto rendimiento

En su mayor parte, la HPC ocurre en supercomputadoras. Estos poderosos sistemas ayudan a las organizaciones a resolver problemas que de otro modo serían insalvables. Y estos problemas, o tareas, requieren procesadores que puedan ejecutar instrucciones más rápido que las computadoras estándar, a veces ejecutando muchos procesadores en paralelo para obtener respuestas dentro de una duración práctica.

Además del procesamiento en paralelo, los trabajos de la HPC también requieren discos rápidos y memoria de alta velocidad. Por lo tanto, los sistemas HPC incluyen servidores de computación y de uso intensivo de datos con CPU potentes que se pueden escalar verticalmente y están disponibles para un grupo de usuarios. Los sistemas HPC también pueden tener muchas potentes unidades de procesamiento de gráficos (GPU) para tareas con uso intensivo de gráficos. Cabe destacar, sin embargo, que cada servidor solo aloja una única aplicación.

Otra aplicación importante de la HPC se encuentra en los campos de los avances médicos y de materiales.

Ejemplos:

Lucha contra el cáncer: los algoritmos de aprendizaje automático proporcionan a los investigadores médicos una visión integral de la población con cáncer en los EE. UU., con un gran nivel de detalle.

Identificar materiales de próxima generación: el aprendizaje profundo podría ayudar a los científicos a identificar materiales para obtener mejores baterías, materiales de construcción más resistentes y semiconductores más eficientes.

Otros Cluster para Big data tipo Hadoop o Spark, permiten el uso del hardware como un servicio, dependiendo de las necesidades de cómputo, de memoria RAM, de almacenaje de la información, hacen que no sea necesario para una organización el desarrollo y la adquisición de sistemas computacionales de altas capacidades, sino alquilar el procesamiento de acuerdo a la necesidad de negocio. En el caso del los Cluster Hadoop tienen la limitación que el proceso de extracción carga de datos y transformación (ELT) solo se puede hacer en ciclos de tiempo bien definidos (ciclos batch), Spark en cambio lo puede hacer en tiempo real lo que puede ser útil para el despliegue de información en dashboard para tomar decisiones de negocio en tiempo real, Cabe mencionar que si bien Spark está basado en Scala no es el único lenguaje que soporta: Spark posee APIs para Java y Python además de diferentes librerías para integrarlo en otros lenguajes como Spark R, para el lenguaje R, muy utilizado para el análisis estadístico de datos.

Un ejemplo en el uso de la plataforma Spark puede ser el uso de spark-submit en data Flow y sistemas de aprendizaje automático o machine learning.

**Pregunta 3.**

Ejercicio EDA con Spark SQL

* Utilizando Apache Spark (en Colab, o DataBricks, o local, …), las librerías de SparkSQL y el dataset [*SII\_roles\_avaluo\_table.zip*](https://www.dropbox.com/s/ianxkffu14dp0ru/SII_roles_avaluo_table.zip)…
* Escriba 3 nuevas consultas (queries) a la data:
  + Debe usar a lo menos un Left Join

Creamos la query para hacer un join con el nombre de las regiones

qry\_join\_nombre\_region="""

SELECT \*

FROM noagricola\_sql as A

LEFT JOIN region as B

ON A.n\_region = B.n\_region

"""

noagricola\_sql2=spark.sql(qry\_join\_nombre\_comuna)

noagricola\_sql2.show()

noagricola\_sql2.createOrReplaceTempView("noagricola\_sql2")

Texto

Descripción generada automáticamente

* + Debe usar a lo menos un GroupBy

Crearemos la query para obtener el promedio por comuna

qry\_promedio\_x\_comuna="""

select cod\_comuna, avg(avaluo\_fisc\_tot) as promedio\_avaluo, count(avaluo\_fisc\_tot) as conteo\_predios, avg(sup\_total\_terreno\_prod\_m2) as superficie\_promedio from noagricola\_sql

group by cod\_comuna

order by promedio\_avaluo desc

"""

df\_promedios\_comuna\_sp=spark.sql(qry\_promedio\_x\_comuna)

df\_promedios\_comuna\_sp.show()

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Debe usar un a lo menos Where con dos condiciones, realizamos una qry para obtener los valores promedios del avaluo, el número total de predios y la superficie promedio donde donde la superficie total del terreno de producción sea mayor o igual a 100 metros cuadrados y menor o igual a 200 metros cuadrados

qry\_promedio2="""

select avg(avaluo\_fisc\_tot) as promedio\_avaluo, count(avaluo\_fisc\_tot) as conteo\_predios, avg(sup\_total\_terreno\_prod\_m2) as superficie\_promedio from noagricola\_sql

where sup\_total\_terreno\_prod\_m2>=100 AND sup\_total\_terreno\_prod\_m2<=200

"""

from pyspark.sql.types import DecimalType

df\_promedios\_sp=spark.sql(qry\_promedio2)

df\_promedios\_sp.withColumn('promedio\_avaluo', df\_promedios\_sp.promedio\_avaluo.cast(DecimalType(18, 2)))

df\_promedios\_sp.show()

Texto

Descripción generada automáticamente

**Pregunta 4.**

Configuramos la sesión y lanzamos spark instalando py 3.7, jdk e instalamos la librería hadoop findspark.

Lanzamos las variables de entorno java home y spark home, utilizando findspark con la dependencia sparksession ejecutamos y configuramos spark.

Para el monitoreo y estado del cluster instalamos la librería pyngrok, configuramos ngrok entregándole el authtoken dado por el dashboard de la página. Abrimos el puerto 4040 e hicimos un túnel entre el sitio local y ngrok.

**A) qué tipo de preprocesamientos fue necesario realizarles a los datos, o si no fue necesario, el por qué.**

Modelamiento no supervisado

Desde la librería de Google colab con la dependencia drive montamos el drive y dentro de la variable ruta\_data\_set pegamos la ruta donde se encuentra el archivo “créditos\_bancarios\_csv”, leímos el archivo almacenado en variable ruta\_data\_set con spark y lo metimos dentro de la variable df, luego hicimos una consulta a la variable df sobre las columnas “status”, “saving”, “installment\_rate”, ”employment\_duration”, ”number\_credits”, concatenamos la 5 columnas en 1 utilizando la función vectorassembler y lo metimos dentro de la variable transformed\_data.

**B) Cómo se determinó el número óptimo de clusters a utilizar**

Utilizando la función for k in range (2,11) y utilizando k-means modelamos en matplotlib el grafico para definir con el método del codo el número de cluster ideal.

Esta función nos arrojó que el k=5, por ende, el número de cluster ideal es 5.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Modelamos la tabla con k=5 y luego contamos el número de predicciones lo que arrojó el siguiente resultado:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**C) La estadística descriptiva de los segmentos encontrados y qué nombre “comercial” le pondría al segmento dadas dichas características**

Los perfiles de clientes según estados crediticios son los siguientes.

**Perfil 2: Seguro. Dura en el trabajo, tiene ahorros y no pide muchos créditos**

Status: 1 = no presenta salario

Saving: 3 = tiene ahorros entre 100 y 500 DM

Installment\_rate: 2 = pide créditos con cuotas entre 25 y 35

Employment\_duration: 3 = duración en el empleo entre 1 y 4 años

Number\_credits: 1 = tiene solo un crédito

**Perfil 4: Precavido, está preocupado del futuro**

Status: 4 = ingresos mayores a 200 DM el último año

Saving: 5 = ahorros mayores a 1000 DM

Installment\_rate: 3 = pide créditos con cuota entre 20 y 25

Employment\_duration: 3 = duración en el empleo entre 1 y 4 años

Number\_credits: 1 = tiene solo un crédito

**Perfil 3: Cauteloso, ahorra, pero no al extremo**

Status: 2 = presenta ingresos menores a 0 DM

Saving: 2 = ahorros menores a 100 DM

Installment\_rate: 2 = pide créditos con cuota entre 25 y 35

Employment\_duration: 4 = duración en el empleo entre 4 y 7 años

Number\_credits: 1 tiene solo un crédito

**Perfil 1: Arriesgado.**

Status: 4 = presenta ingreso mayor a 200 DM

Saving: 1 = no tiene ahorros

Installment\_rate: 4 = pide créditos con cuotas menores a 20

Employment\_duration: 1 = no presenta trabajo

Number\_credits: 2 = tiene entre 2 y 3 créditos

**Perfil 0: Ultra arriesgado. No tiene estabilidad laboral e igual pide créditos**

Status: 1 = no presenta salario

Saving: 1 = no tiene ahorros

Installment\_rate: 4 = crédito menor a 20 cuotas

Employment\_duration: 2 = dura menos de 1 año en el trabajo

Number\_credits: 1 = tiene 2 crédito

**D) Acciones de negocios que podrían ser relevantes a sugerir para los segmentos encontrados**

Enfocar ofertas de créditos a personas que tienen estabilidad económica, en este caso serían los clústeres 2, 4 y 3 y son casi el 50% de la muestra.