**Tarea de Vacaciones**

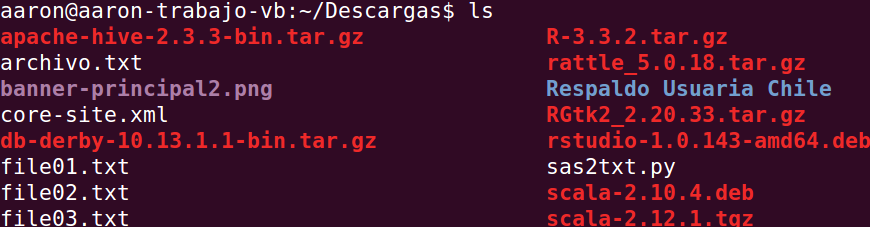
Para esta tarea se deberá responder una serie de preguntas de temas que se han abordado en el curso, el objetivo consiste entonces en reforzar los conocimientos e indagar en otros nuevos que, aunque no forman parte explícita del temario, le servirán al estudiante para incursionar en temas Big Data a plenitud.

El estudiante debe crear primero que nada un directorio dentro de su directorio local de Git llamado **TareaVacaciones** y dentro de éste crear una copia de esta tarea que lleve por nombre TareaX **y colocarla juntos con los resultados en formato PDF**, donde X es su nombre de usuario empleado en Github. Por ejemplo:

**TareaYoNoFui**

Con respecto de los ejercicios, a menos que se indique lo contrario, todas las respuestas constarán del código o instrucción resultante acompañada de una captura de pantalla. Ejemplo:

*-1.- Indique el comando que se emplea para listar archivos en GNU/Linux de manera simple:  
Respuesta:* ***ls***



En este tipo de preguntas de faltar alguno de los elementos señalados se considerará como errónea la respuesta y no se obtendrá el acierto.

Es menester mencionar que hay casos donde las preguntas son de tipo abierto, entonces en esos casos lo único que se pide adjuntar es tanto la respuesta como la(s) fuente(s). Ejemplo:

*0.- ¿Cuál es el significado de la vida?  
Respuesta:* ***42*** *Fuente:*[***https://www.independent.co.uk/life-style/history/42-the-answer-to-life-the-universe-and-everything-2205734.html***](https://www.independent.co.uk/life-style/history/42-the-answer-to-life-the-universe-and-everything-2205734.html)

De nueva cuenta, si no existe al menos uno de estos dos elementos, la respuesta se considerará como inválida.

La fecha límite de entrega es el **Lunes 25 de Junio a las 15:00:00**, como se había mencionado con anterioridad el flujo de archivos se mantiene única y exclusivamente por Github, para ello se dejan los comandos a emplearse:

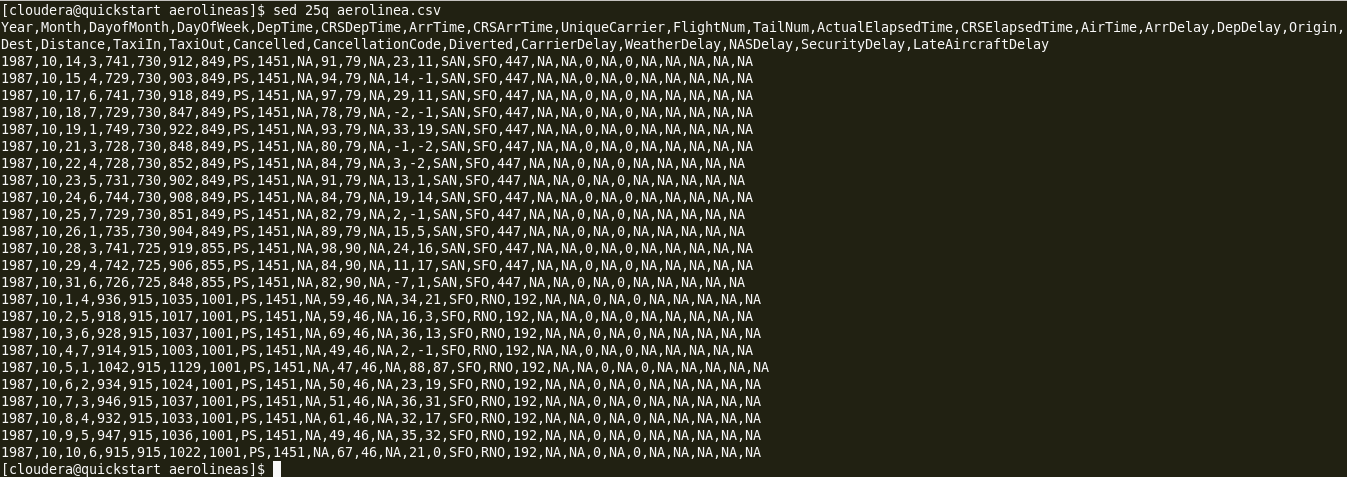
* **git pull (para actualizar el repositorio)**
* **git add . (para indicar todos los elementos que se desean agregar al repositorio)**
* **git commit –m “TareaVacaciones nombre\_usuario” (para colocar un mensaje que distinga a esta subida de las de los demás usuarios)**
* **git push origin master (para efectuar los cambios)**

Por cierto que en lo que se repara Git en Cloudera puede ocupar Git de Windows y de esta manera, ya que se sugirió la instalación de Guest Additions en VirtualBox, copiar los resultados al primer sistema.

Nuevamente, de no cumplirse al menos uno de los señalamientos anteriores la tarea se considerará no entregada.  
  
Dicho lo anterior se les desea mucho éxito en la travesía, cualquier cosa no duden en preguntar…

**SECCION 1. GNU/LINUX**

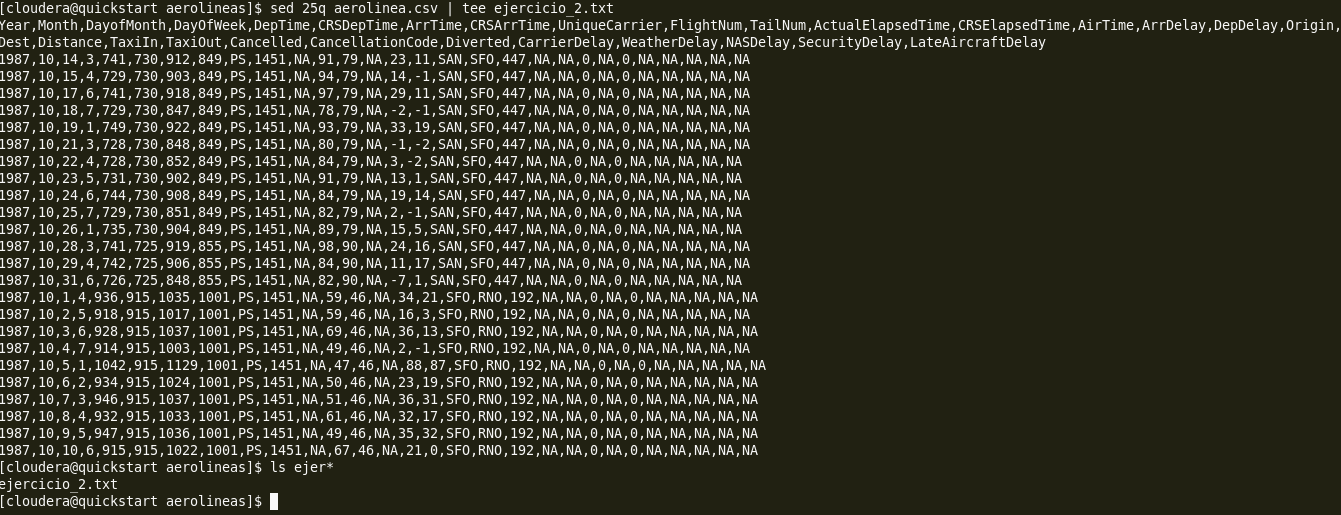
1.- Del archivo **aerolineas.csv** (el archivo descomprimido que todavía debería estar en su local y no el del HDFS) use comandos de GNU/Linux para obtener las 25 primeras líneas (incluyendo encabezado) **SIN** usar el comando head.

sed 25q aerolínea.csv  


2.-Como ya se ha visto, utilizar el redireccionamiento destructivo (>) implica almacenar típicamente algún contenido en un archivo **(ej. echo "contenido" > archivo)**.   
Pero lo cierto es que con este comando no se apreciará en pantalla lo que se desea almacenar en dicho archivo, por ello es que se necesita que, con base en el comando resultado del ejercicio 1 y con la investigación del comando **tee**, por un lado el contenido se introduzca en el archivo **ejercicio\_2.txt** y por el otro se muestre en pantalla la operación.  
Caber mencionar que todo se debe registrar como una sola instrucción, es decir, no se puede ejecutar el resultado por partes, para ello tal vez quiera leer esta liga:

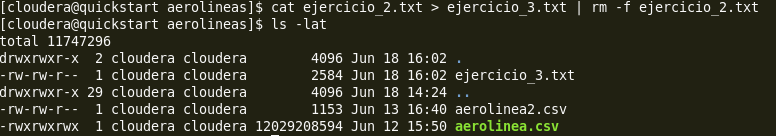
**http://www.linfo.org/pipes.html**

sed 25q aerolínea.csv | tee ejercicio\_2.txt



3.- Cambie el nombre del archivo **ejercicio\_2.txt** a **ejercicio\_3.txt** **SIN** usar el comando rename

cat ejercicio\_2.txt > ejercicio\_3.txt | rm –f ejercicio\_2.txt



4.- Con algún comando en GNU/Linux tome las 25 últimas líneas del archivo aerolínea.csv **SIN** emplear el comando tail y guárdelo como **ejercicio\_4.txt**

**sed -e :a -e '$q;N;26,$D;ba' aerolinea.csv > ejercicio\_4.txt**

:a # define label a

${ # match the last line

P # print the first line of the pattern space

q # quit

}

N # match all lines: append the next line to the pattern

26,${ # match the range of lines 26 to the end of the file

D # delete the first line of the pattern space

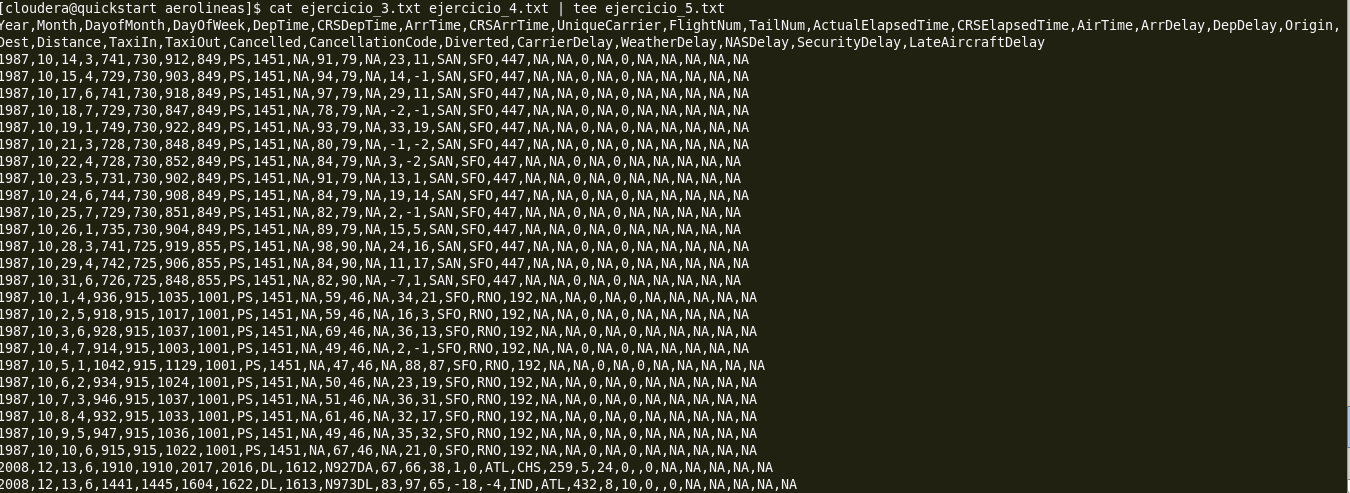
}

ba # match all lines: jump back to label a



5.- Concatene los archivos **ejercicio\_3.txt** y **ejercicio\_4.txt** en un archivo **ejercicio\_5.txt** y en esa misma pantalla resultado muestre el contenido de **ejercicio\_5.txt**

**cat ejercicio\_3.txt ejercicio\_4.txt | tee ejercicio\_5.txt**



6.- Usando el comando **ls** y sus opciones, verifique el peso de **ejercicio\_5.txt,** señalando en la captura de pantalla dónde se encuentra éste.

ls -lah ejercicio\_5.txt

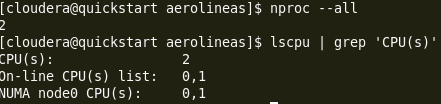


7.- Modifique la fecha de acceso de **ejercicio\_5.txt** al 25 de Agosto del 2018 y muestre en pantalla dónde se puede apreciar ese resultado.

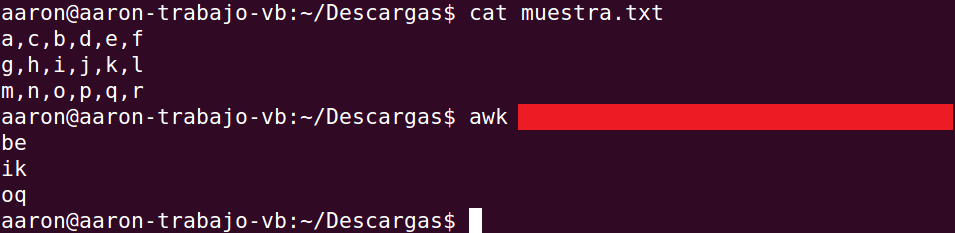
touch -t 1808250000 ejercicio\_5.txt

  
  
8.- ¿Con cuál comando se puede averiguar el número de núcleos en un sistema GNU/Linux? Investigue y coloque el resultado, haciendo énfasis en el lugar donde se puede apreciar esa información.

nproc –all ; lscpu | grep 'CPU(s)'



9.- Investigue en qué consiste awk y por medio de esa herramienta imprima en pantalla sólo la tercera y quinta columnas (de izquierda a derecha) del archivo **ejercicio\_5.txt**. He aquí un ejemplo de cómo se ve el resultado con otro archivo que no tiene que ver con el curso:



AWK.- Lenguaje de programación diseñado para trabajar con archivos estructurados y patrones de texto.

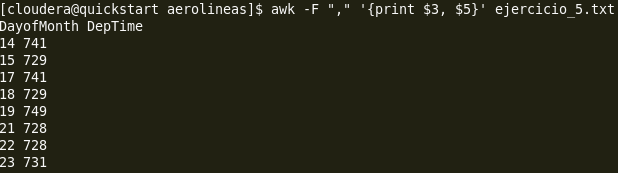
**AWK** utiliza un archivo o emisión de ordenes y un archivo o emisión de entrada. El primero indica como procesar al segundo. El archivo de entrada es por lo general texto con algún formato que puede ser un archivo o bien la salida de otro programa.

La sintaxis general para **awk** utiliza el siguiente patrón:

awk 'expresión-regular { orden }'

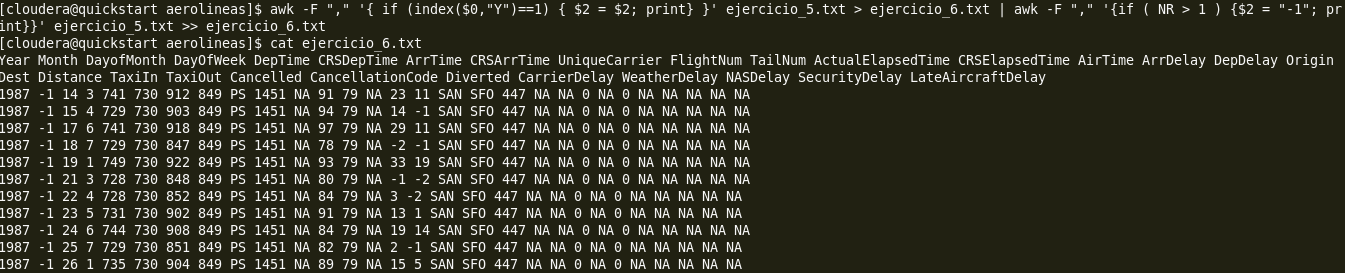
Cuando se utiliza **awk**, éste examina el archivo de entrada y ejecuta la orden cuando encuentra la expresión regular especificada.

**awk -F "," '{print $3, $5}' ejercicio\_5.txt**



10.- Sin usar vim, nano o editor de texto alguno use comandos de Linux para reemplazar TODOS los elementos de la segunda columna por -1, guárdelo como **archivo\_6.txt** y hágale un cat a ese mismo archivo.

**awk -F "," '{ if (index($0,"Y")==1) { $2 = $2; print} }' ejercicio\_5.txt > ejercicio\_6.txt | awk -F "," '{if ( NR > 1 ) {$2 = "-1"; print}}' ejercicio\_5.txt >> ejercicio\_6.txt**



**SECCION 2. HDFS Y HIVE**

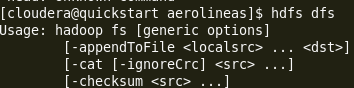
11.- Se está tratando de hacer la siguiente operación:

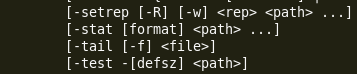
**hdfs dfs -head /raw/aerolínea.csv**

Con una captura muestre qué es lo que pasa y por medio de argumentos sólidos (una captura de pantalla con la evidencia, una fuente de consulta) por qué sucede esto



Mediante el comando hdfs dfs, podemos verificar que comando podemos utilizar en el FyleSystem de hdfs:





Como podemos observar solo existe en comando tail, el comando head no funciona en FS.

12.- Cuente cuántas líneas tiene el archivo **aerolínea.csv** que está **en el HDFS**. Recuerde el carácter pipe (|) empleado en ejercicios anteriores.



13.- Indague en la instrucción de HDFS para averiguar el factor de réplica del archivo aerolínea.csv y colóquelo aquí junto con captura del resultado.

hdfs getconf -confKey dfs.replication



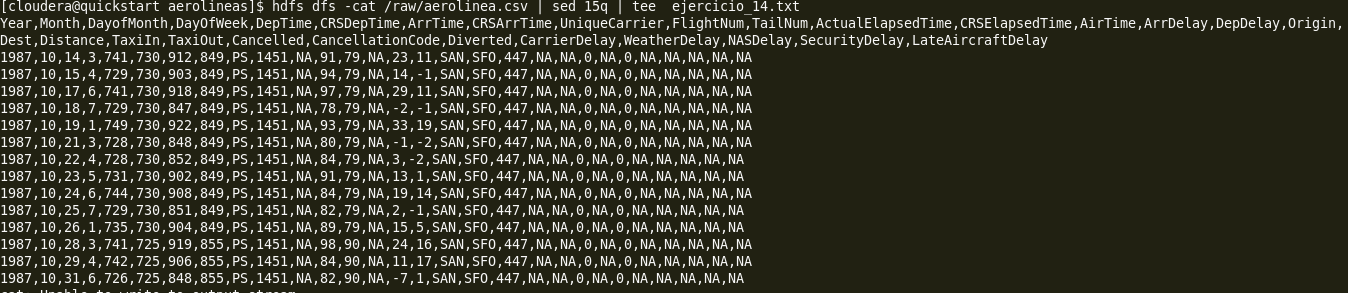
Gracias al comando ***getconf*** puedo obtener información de configuración de la instalación HDFS. Dónde está almacenada dicha configuración? La respuesta está en el archivo **hdfs-site.xml**. En el caso de nuestra máquina virtual Cloudera podemos encontrarlo en **/etc/hadoop/conf**. Si editamos el archivo hdfs-site.xml y buscamos allí la configuración de replicación, encontraremos las siguientes líneas:

<configuration>  
  <property>  
    <name>dfs.replication</name>  
    <value>1</value>  
  </property>

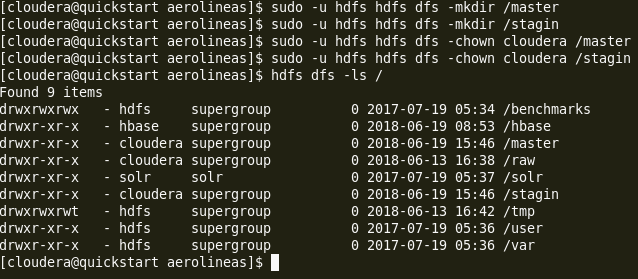
https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HDFSCommands.html

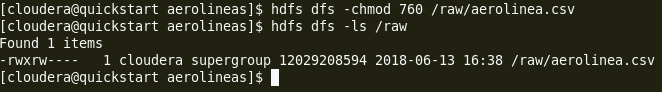
14.- Tome como base el archivo **aerolínea.csv** del HDFS y almacene en el sistema local un archivo **ejercicio\_14.txt** que contenga las primeras 15 líneas sin usar el comando -tail del HDFS. Muestre ese contenido también.

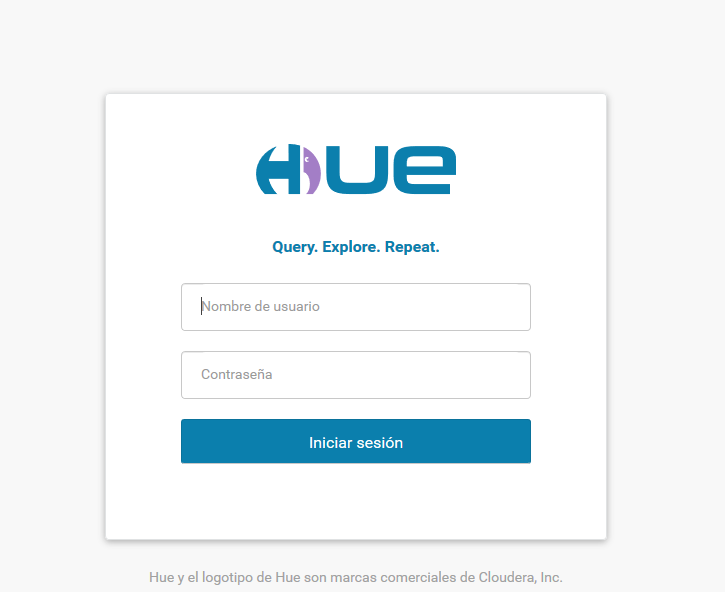
**hdfs dfs -cat /raw/aerolinea.csv | sed 15q | tee ejercicio\_14.txt**



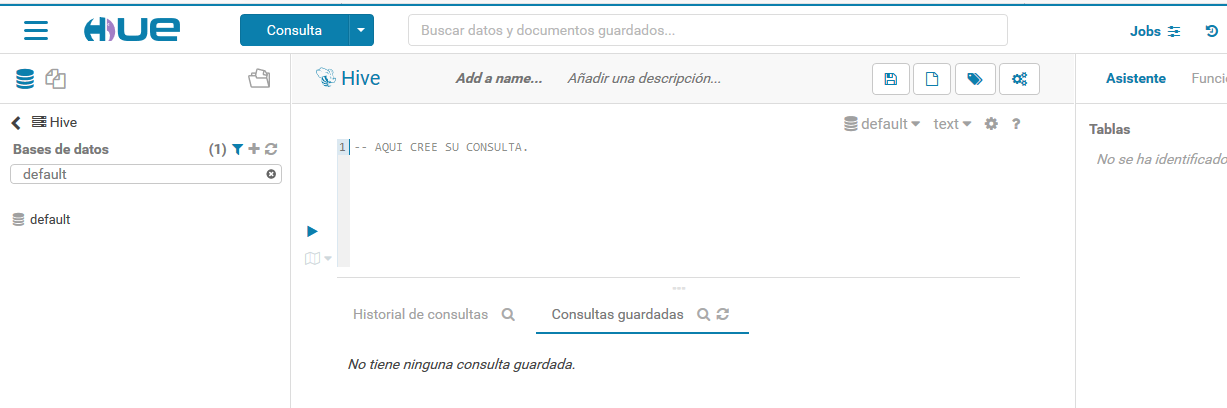
15.- Cree los directorios **master**  y **stagin** en el directorio raíz del HDFS y además al archivo aerolínea.csv que está en raw cámbiele los permisos de tal manera que el propietario tenga todas las facilidades sobre él, el grupo sólo pueda leer y escribir y cualquier otro no tenga ningún permiso. Coloque las capturas de ambos ejercicios por separado.



  
  
16.- Para los siguientes ejercicios puede hacer uso del servicio Hue (si no ha activado los servicios en Cloudera Manager tiene que hacerlo antes, para entrar a Hue en el mismo navegador se encuentra esta opción).   
Aparecerá una ventana como ésta:



Recuerde que tanto el usuario como la contraseña es **cloudera**:

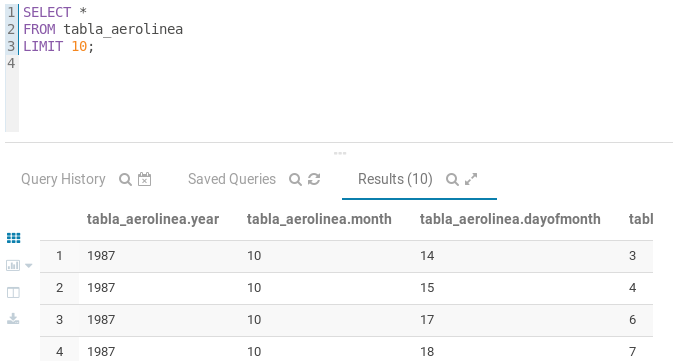


Entonces tome el siguiente código y cree una tabla en Hive:  
  
CREATE EXTERNAL TABLE tabla\_aerolinea(

Year STRING,  
Month STRING,  
DayofMonth STRING,  
DayOfWeek STRING,  
DepTime STRING,  
CRSDepTime STRING,  
ArrTime STRING,  
CRSArrTime STRING,  
UniqueCarrier STRING,  
FlightNum STRING,  
TailNum STRING,  
ActualElapsedTime STRING,  
CRSElapsedTime STRING,  
AirTime STRING,  
ArrDelay STRING,  
DepDelay STRING,  
Origin STRING,  
Dest STRING,  
Distance STRING,  
TaxiIn STRING,  
TaxiOut STRING,  
Cancelled STRING,  
CancellationCode STRING,  
Diverted STRING,  
CarrierDelay STRING,  
WeatherDelay STRING,  
NASDelay STRING,   
SecurityDelay STRING,  
LateAircraftDelay STRING)

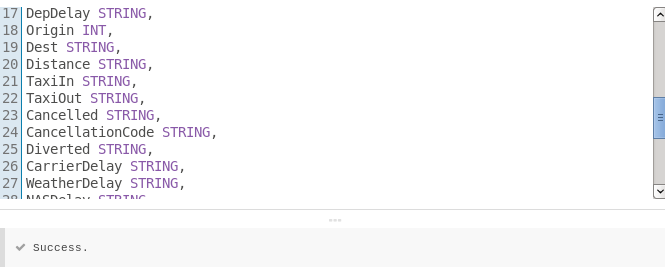
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
STORED AS TEXTFILE  
location '/raw'

tblproperties ("skip.header.line.count"="1");  
  
En el código anterior **NO** existe una forma de omitir los encabezados por lo que es su deber encontrar esa manera, incluirla en el código y crear la tabla.  
Para acreditar el ejercicio debe mostrar la sentencia que requirió para la parte de los encabezados y hacer un SELECT de los 10 primeros elementos de la tabla.

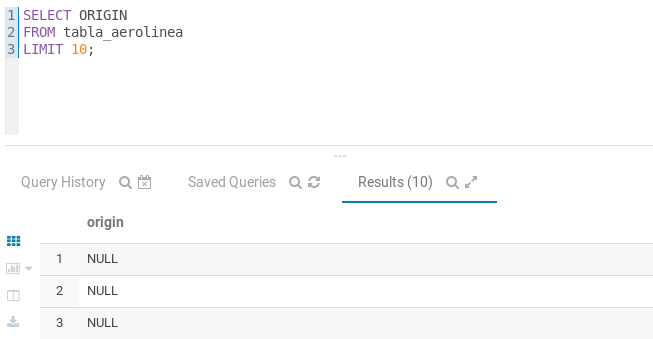


17.- Borre la tabla anterior y vuélvala a crear pero ahora el tipo de dato Origin debe ser INT, entonces vuelva a ejecutar la consulta y especifique qué ha pasado y con una captura muéstrelo.

La tabla se creo:



El dato de Origin es de tipo alfanumérico, por ello todos los datos de la columna Origin se reportan como NULL:

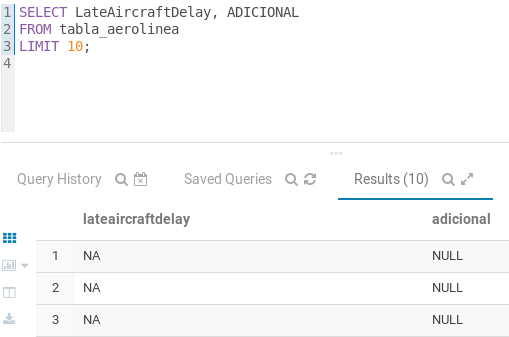


18.- Borre la tabla anterior, vuélvala a crear (con Origin STRING) pero ahora añada una columna después de LateAircraftDelay llamada **Adicional** con tipo de dato **STRING**, ejecute la creación, indique qué ha sucedido y coloque captura del resultado.

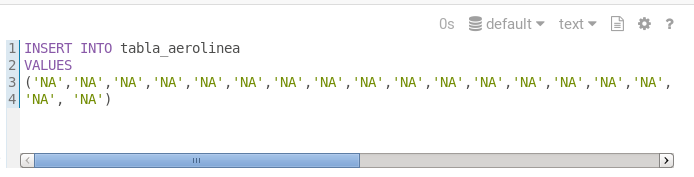
La tabla se creo sin errores:



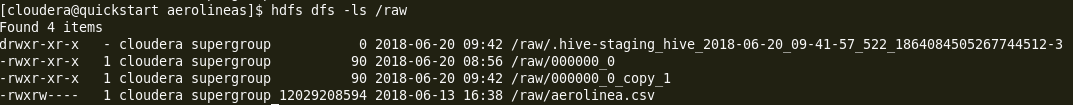
Sin embargo, el campo adicional tiene valores nulos:



19.- En esta tabla anterior inserte un renglón a la tabla con todos los valores iguales a “NA” (tiene que investigar cómo añadir elementos a la tabla), y luego después de la inserción del elemento indague en qué parte del HDFS se ha guardado ese nuevo elemento.



El elemento se guardo en los archivos:



**SECCIÓN 3. PREGUNTAS ABIERTAS**

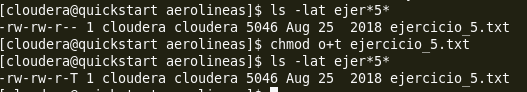
20.- ¿Qué es el Sticky Bit? Ejemplifíquelo con el archivo **ejercicio\_5.txt** adjuntando una captura de pantalla.

https://hvivani.com.ar/2013/09/06/permisos-especiales-sticky-bit-suid-sgid/

Hay una serie de permisos especiales sobre el sistema de archivos de Linux que pueden resultarnos útiles para determinadas tareas o para organizar directorios colaborativos entre diferentes usuarios.

Sticky bit

El Sticky bit se utiliza para permitir que cualquiera pueda escribir y modificar sobre un archivo o directorio, pero que solo su propietario o root pueda eliminarlo. Un ejemplo de uso es el directorio /tmp, que debe tener permisos para ser utilizado por cualquier proceso, pero solo el dueño o root puede eliminar los archivos que crea.



21.- ¿A qué se le conoce como NoSQL?, ¿considera que Hive e Impala son representantes? Justifique la respuesta.

https://searchdatacenter.techtarget.com/es/definicion/NoSQL-No-Solo-SQL

Las bases de datos NoSQL, también llamadas No Solo SQL, son un enfoque hacia la gestión de datos y el diseño de base de datos que es útil para grandes conjuntos de datos distribuidos.

NoSQL es especialmente útil cuando una empresa necesita acceder y analizar grandes cantidades de datos no estructurados o datos que se almacenan de forma remota en varios servidores virtuales en la nube.

Contrariamente a las ideas falsas causadas por su nombre, [NoSQL no prohíbe el lenguaje estructurado de consultas (SQL)](https://searchdatacenter.techtarget.com/es/definicion/SQL-o-lenguaje-de-consultas-estructuradas). Si bien es cierto que algunos sistemas NoSQL son totalmente no-relacionales, otros simplemente evitan funcionalidades relacionales seleccionadas como esquemas de tablas fijas y operaciones conjuntas. Por ejemplo, en lugar de utilizar tablas, una base de datos NoSQL podría organizar los datos en objetos, pares clave/valor o tuplas.

https://mauricioanderson.com/hive-introduccion-instalacion/

**HIVE**

[Apache Hive](https://hive.apache.org/) es un software que forma parte del ecosistema de [Hadoop](http://hadoop.apache.org/" \t "_blank). Es utilizado para gestionar enormes datasets almacenados bajo el HDFS de Hadoop y realizar consultas (*queries*) sobre los mismos.

Para esto, Hive implementa una variante al SQL, llamada HQL (Hive QL). Esta variante no soporta la especificación SQL-92 completa, pero sí una gran parte. Realmente, cuando se ejecutan consultas HQL, Hive convierte la consulta HQL a un trabajo [MapReduce](https://es.wikipedia.org/wiki/MapReduce" \t "_blank)que es ejecutado para obtener los datos. La finalidad de esto es permitir a usuarios que no cuenten con experiencia en programación de algoritmos MapReduce (que suelen ser bastante laboriosos), pero que sí cuenten con conocimientos de SQL, poder consultar datos.

Un punto a favor de Hive es que trabajar con este es similar a trabajar con bases de datos tradicionales. Desde Hive debemos estructurar los datos agrupándolos en tablas, con sus columnas y tipos de datos asociados. Pintar en estas tablas los datos, luego procesarlos y finalmente analizarlos. No obstante, no hay que olvidar que Hive no es un motor de bases de datos.

El inconveniente que presenta Hive es que, al necesitar procesar la consulta y traducirla a lenguaje Java para crear el MapReduce, la latencia de respuesta es alta

IMPALA

Cloudera Impala es un motor de consulta que corre en Apache Hadoop

Impala está dirigido a los analistas y científicos de datos para realizar análisis en los datos almacenados en Hadoop a través de herramientas de SQL o [business intelligence](https://es.wikipedia.org/wiki/Business_intelligence" \o "Business intelligence). El resultado es que el procesamiento de datos a gran escala (a través de MapReduce) y las consultas interactivas se pueden hacer en el mismo sistema utilizando los mismos datos y metadatos - eliminando la necesidad de migrar los conjuntos de datos a sistemas especializados y/o formatos propietarios solo para realizar el análisis.

Las características incluyen:

Soporte de [HDFS](https://es.wikipedia.org/wiki/HDFS) y almacenamiento [Apache HBase](https://es.wikipedia.org/wiki/Apache_HBase)

Lee formatos de archivos de Hadoop, incluyendo texto, LZO, SequenceFile, Avro, RCFile, y Parquet

Soporta seguridad Hadoop (autenticación Kerberos)

Autorización fine-grained, basada en roles con Sentry[5](https://es.wikipedia.org/wiki/Cloudera_Impala#cite_note-5)​

Utiliza metadata, controlador ODBC, y sintaxis SQL de [Apache Hive](https://es.wikipedia.org/wiki/Apache_Hive)

22.- Investigue el uso del comando nohup en GNU/Linux y con base en esto responda: ¿cómo puede ser aplicado dicho comando en un sistema distribuido?

http://maslinux.es/que-es-el-comando-nohup-y-como-usarlo/

Nohup es la abreviatura de “No Hangups”. Es un comando complementario que le dice al sistema Linux que no detenga otro comando una vez que haya comenzado. Eso significa que seguirá funcionando hasta que se complete, incluso si el usuario que lo inició cierra la sesión. La sintaxis para Nohup es simple y se ve así:

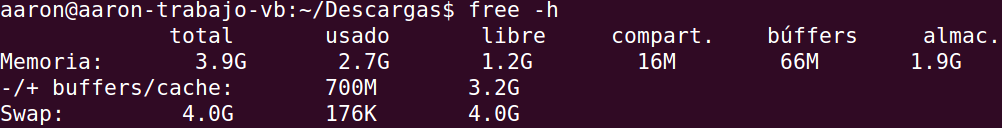
nohup sh your-script.sh &

Observa el “&” al final del comando. Eso mueve el comando al fondo, liberando el terminal en el que estás trabajando.

Debido a que *Nohup* puede seguir ejecutándose independientemente del usuario que lo inició, el comando necesita algún lugar para emitir cualquier mensaje o error. Como no hay un terminal para asociarlo, *Nohup* lo registra todo en un archivo de salida, *nohup.out.*

**Lo podemos aplicar a sistemas distribuidos para tareas que tarden mucho tiempo en procesarse, entonces las podemos dejas ejecutando en segundo plano, ejemplo: las cargas de información de millones de registros.**

23.- Se quiere averiguar la memoria RAM disponible con base en la siguiente imagen:

  
Indique el o los valores adecuados y por qué.

El espacio libre es de 1.2 GB

El espacio libre + que no esta siendo utilizado (Total) es de 3.2 GB

24.- Se tiene el siguiente escenario: personal ajeno a su área de sistemas desea tener acceso al sistema, en particular para ver algunos datos del archivo **objetivo.txt**Por otra parte se sabe de manera extraoficial que la meta de ellos consiste en “ensuciar” el archivo para que el área no tenga tanto repunte como la nuestra.  
Por cuestiones burocráticas la creación de algún usuario nuevo no es plausible no obstante debido a asuntos políticos es prácticamente un hecho que se le tiene que dar permiso, por ello es que se optó por prestarles un usuario **(usuario\_nuestro)** cuyo grupo es **grupo\_nuestro**.

Con base en estas características y limitando el escenario únicamente a comandos **chmod (y si lo desea chown y chgrp)**, ¿cuál sería la configuración que usted propondría para garantizar el acceso al archivo pero al mismo tiempo protegerlo de las circunstancias mencionadas y sin afectar al mismo tiempo a los demás miembros de **grupo\_nuestro**?

25.- ¿Cuál es la diferencia entre Hadoop y Cloudera?

https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/que-es-big-data-hadoop-y-para-que-sirve

[Hadoop](https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/387604/m-s-all-de-cloudera-soluciones-de-hadoop-y-big-data)**es un framework opensource para almacenar datos y ejecutar aplicaciones en clusters de hardware básicos**. Proporciona un almacenamiento masivo para cualquier tipo de datos, un enorme poder de procesamiento y la capacidad de manejar tareas o trabajos prácticamente ilimitados.

* **C**apacidad para almacenar y procesar grandes cantidades de cualquier tipo de datos rápidamente. Con volúmenes y variedad de datos en constante aumento, especialmente de lo que se refiere a medios de comunicación social y del Internet de las Cosas, esto es una consideración clave.
* Poder de procesamiento. El modelo de computación distribuida de Hadoop procesa rápidamente [Big Data](https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/402575/Big-Data-Hadoop-es-s-lo-para-las-grandes-empresas). Cuantos más nodos de computación se utilizan, más poder de procesamiento tendrás.
* Tolerancia a fallos. Los datos y el procesamiento de aplicaciones están protegidos contra fallos de hardware. Si un nodo disminuye, los trabajos se redirigen automáticamente a otros nodos para asegurarse de que la computación distribuida no falla.  Se almacenan automáticamente varias copias de todos los datos.
* Flexibilidad. A diferencia de las bases de datos relacionales tradicionales, no es necesario preprocesar los datos antes de almacenarlos. Puedes almacenar tantos datos como desees ydecidir cómo utilizarlos más tarde. Esto incluye datos no estructurados como texto imágenes y vídeo.
* Bajo coste. El framework de código abierto es gratuito y utiliza hardware básico para almacenar grandes cantidades de datos.
* Escalabilidad. Puedes hacer crecer fácilmente el sistema para manejar más datos simplemente añadiendo nodos. Se requiere poca administración.

Cloudera es la plataforma Apache Hadoop más rápida, fácil de usar y segura con la que solucionar los problemas de datos más complejos. Y es que Hadoop se ha convertido en el nuevo paradigma de arquitectura de computación en paralelo.

¿Qué es Cloudera?

Es una firma especializada en Big Data, que permite añadir funciones a la arquitectura Hadoop de seguridad, control y gestión necesarios para establecer una solución empresarial robusta y fiable.

Su software está basado en Apache Hadoop y ofrecen soporte, servicios y formación para grandes clientes.

¿Ques es Hadoop?

Hadoop es un ecosistema de componentes de código abierto que cambia fundamentalmente la manera en que las empresas almacenan, procesan y analizan datos.

A diferencia de los sistemas tradicionales, Hadoop permite que múltiples tipos de cargas de trabajo y analíticas se ejecuten en los mismos datos al mismo tiempo a gran escala.

26.- ¿Cuáles son los tipos de archivos existentes en GNU/Linux y Windows?

**https://geekytheory.com/permisos-de-archivos-en-linux**

**Linux:**

Tipos de archivo (primer caracter):

| Tipo de archivo | Significado |
| --- | --- |
| - | Archivo común |
| d | Directorio |
| l | Enlace simbólico |
| s | Socket |
| p | Pipe |

WINDOWS

27.- ¿Qué es el SerDe y cuál es su relación con Hive e Impala?

https://www.quora.com/What-is-SerDe-in-Hive

El interfaz SerDe permite indicarle a Hive como debe procesar un registro. SerDe es una combinación de Serializer y Deserializer.

· Deserializer toma una representación string o binaria y lo convierte a un objeto Java que Hive puede manipular.

· Serializer: toma un objeto Java y lo convierte en algo que Hive puede escribir a HDFS.

Hive viene con varios SerDe como JSon SerDe para archivos JSon, CSV SerDe para archivos CSV, etc.

28.- ¿A qué se le conoce como Big Table y Big Query?

https://www.codifica.me/big-table-no-sql/

**Big Table** es un SGBD que creó Google para gestionar sus bases de datos en el año 2004, entre las características más importantes del desarrollo es que querían conseguir un sistema que fuese escalable fácilmente, el sistema iba a ser distribuido en multitud de máquinas y necesitaban que fuese altamente eficiente.

Google puso en marcha este proyecto porque los sistemas de bases de datos tradicionales no le convencían lo suficiente para almacenar grandes cantidades de datos, además la rapidez que ofrecían estas bases de datos no eran lo suficiente cuando hablamos de grandes cantidades de datos. La mayoría de los SGBD fueron diseñados para trabajar en un servidor o en arquitecturas poco escalables en comparación con lo que Google necesitaba para tratar sus enormes cantidades de datos, del orden de Petabytes.  
  
Es un sistema que divide la información en columnas, para almacenar la información utiliza tablas multidimensionales compuestas de celdas, cada uno de estas celdas dispone de varias versiones para poder realizar un seguimiento de los valores que ha ido tomando esa celda en el tiempo. Como concepto podría entenderse como un mapa multidimensional dónde podemos ver las filas, las columnas y por último el tiempo.

El sistema de archivos que utiliza Big Table es el denominado GFS (Google File System), que no es más que un sistema de archivos distribuido del mismo propietario (Google).

Una característica muy importante de este sistema es la compresión de los datos, Big Table utiliza dos algoritmos de compresión muy rápidos para comprimir los datos. No obstante, con este algoritmo Big Table nos permite obtener parte de los datos sin tener que descomprimir el dato en su totalidad. Esto les da cierta flexibilidad y sobretodo velocidad frente a otros gestores de datos. Las estimaciones de velocidad en compresión y descompresión de datos apuntan a los valores de 100 – 200 MB/s y 400 – 1000 MB/s respectivamente.

Como viene siendo habitual en Google, el API de Big Table es de acceso público, por lo que cualquier persona puede acceder al API y realizar cualquier prueba o desarrollo. No obstante Big Table no se distribuye fuera de Google aunque se puede acceder a ella a través de Google App Engine. El lenguaje de implantación que utiliza es el lenguaje C y/o lenguaje C++.

GFS está desarrollado para poder almacenar información sobre un sistema de ficheros distribuidos, de una manera rápida y eficaz, además GFS le da mucha importancia a la seguridad de los datos. Este sistema no guarda los datos en un mismo ordenador o servidor sino que cada vez que almacena un dato realiza varios trozos (chunk) y almacena hasta tres copias de cada trozo.

La arquitectura tiene dos tipos de ordenadores diferentes. Por un lado están los de tipo Master que almacenan la dirección dónde están guardados los trozos que componen los ficheros. Por otro lado, están los Chunk Server, que son los ordenadores o servidores que tienen almacenados los datos. Con esta jerarquía de trabajo logran no saturar los servidores Maestros de peticiones y respuestas de datos ya que ellos redirigen el trabajo a los Chunk Server, que serán los encargados de enviar los datos al usuario final.

Estos Chunk Server son los encargados de almacenar y enviar los datos, cuando almacenan no distinguen entre operaciones ya que solo utilizan la operación de escribir datos, es decir, solo es posible añadir datos, no se sobrescriben como otras muchos sistemas de almacenamiento de datos.

Este sistema tiene la capacidad de asignar más recursos a los ficheros que más se utilizan logrando así que la asignación de recursos no sea homogénea y se pueden liberar recursos de ficheros que se utilizan menos a otros ficheros que se utilizan más.

Por último, comentar que GFS es un sistema de archivos que no está dentro del sistema operativo, sino que es accesible por medio de librerías, esto nos permite utilizar GFS con varios sistemas operativos y no depender del sistema operativo.

https://bbvaopen4u.com/es/actualidad/analisis-de-big-data-como-servicio-con-google-bigquery

**BigQuery** es la solución Big Data de Google. Se trata de un servicio web que permite almacenar y consultar grandes volúmenes de datos en pocos segundos. Un servicio de análisis de datos a gran escala. Tú le envías a BigQuery los datos desde tus sistemas, y BigQuery los almacena para cuando necesites consultarlos. Cuando quieras hacer un análisis, BigQuery te proporciona un mecanismo que te permite hacer cualquier consulta y obtener los resultados en segundos, independientemente de tu volumen de datos.

La potencia de BigQuery se basa en tres pilares fundamentales:

● Una estructura de datos interna basada en columnas que le permite almacenar y consultar la información de una forma muy eficiente.

● Un uso exhaustivo de la nube, realizando todas las consultas en paralelo para conseguir la mayor velocidad. No es raro que una única consulta se esté ejecutando en paralelo en cientos de servidores, cada uno procesando una cantidad pequeña de información antes de proporcionarnos la respuesta final. De esta manera, Google utiliza su enorme infraestructura para conseguir tiempos de respuesta que nos hubieran parecido increíbles hace apenas unos años.

● El lenguaje de consulta SQL. Aunque la infraestructura de Google BigQuery no es la de una base de datos más convencional, el lenguaje de consultas que utiliza es exactamente el mismo, por lo que si ya has utilizado SQL en el pasado, es muy fácil utilizar BigQuery.

El tipo de datos que puedes enviar a Google BigQuery es prácticamente cualquier cosa. Puedes enviar datos sobre tu *CRM*, tu analítica, tu sistema de facturación, el seguimiento de envío de pedidos, y después cruzar datos de todos ellos para extraer información y patrones. Si eres más técnico, BigQuery puede ingerir datos en formato CSV (datos organizados en filas y columnas, como un Excel) o datos en formato JSON (datos estructurados y anidados, para representar jerarquías complejas de información). Estos datos los podemos enviar de vez en cuando (cada hora, una vez al día…) o en modo *streaming*, para poder ser analizados en tiempo real (donde real significa unos pocos segundos).

29.- ¿A qué se le denomina Data Lake y Data Warehouse?

https://www.powerdata.es/data-lake

Un [**data lake**](https://www.powerdata.es/el-valor-de-la-gestion-de-datos/elementos-que-debemos-considerar-en-la-gestion-de-data-lake) es un repositorio de almacenamiento que contienen una gran cantidad de datos en bruto y que se mantienen allí hasta que sea necesario. A diferencia de un data warehouse jerárquico que almacena datos en ficheros o carpetas, un data lake utiliza una arquitectura plana para almacenar los datos.

A cada elemento de un data lake se le asigna un identificador único y se etiqueta con un conjunto de etiquetas de metadatos extendidas. Cuando se presenta una cuestión de negocios que debe ser resuelta, podemos solicitarle al data lake los datos que estén relacionados con esa cuestión. Una vez obtenidos podemos analizar ese conjunto de datos más pequeño para ayudar a obtener una respuesta.

El data lake se asocia a menudo con el almacenamiento de objetos orientado a Hadoop. En este escenario, los datos de una organización se cargan primero en la plataforma Hadoop y, a continuación, se aplican las herramientas de análisis y de minería de datos a los datos que residen en los nodos clúster de Hadoop.

El principal beneficio de un data lake es la centralización de fuentes de contenido dispares. Una vez reunidas (de sus "[silos de información](https://www.powerdata.es/el-valor-de-la-gestion-de-datos/estudio-sobre-el-problema-de-los-silos-para-un-enfoque-data-driven)"), estas fuentes pueden ser combinadas y procesadas utilizando big data, búsquedas y análisis que de otro modo hubieran sido imposibles. Las fuentes de contenido dispares a menudo contienen información confidencial que requerirá la implementación de las medidas de seguridad apropiadas en el data lake.

Una vez que el contenido está en el data lake, puede normalizarse y enriquecerse. Esto puede incluir extracción de metadatos, conversión de formatos, aumento, extracción de entidades, reticulación, agregación, des-normalización o indexación.

## ****2****. Principales diferencias entre Data Lakes y Data Warehouses

#### 1. Una Data Lake conserva todos los datos

Durante el desarrollo de un data warehouse, se gasta una cantidad considerable de tiempo analizando las fuentes de datos, entendiendo los procesos de negocio y perfilando los datos. El resultado es un modelo de datos altamente estructurado diseñado para la generación de informes. Una gran parte de este proceso incluye tomar decisiones sobre qué datos incluir y no incluir en el almacén. Generalmente, si los datos no se utilizan para responder a preguntas específicas o en un informe definido, pueden excluirse del almacén. Esto se hace generalmente para simplificar el modelo de datos y también para conservar el costoso espacio en el almacenamiento de disco que se utiliza para hacer el data warehouse.

En contraste, **el data lake conserva todos los datos. No sólo los datos que se utilizan actualmente, sino los datos que se pueden utilizar e incluso los datos que nunca se van a ser utilizados sólo porque quizás podrían ser utilizados algún día**. Los datos también se mantienen todo el tiempo para que podamos volver en el tiempo a cualquier punto para hacer el análisis.

Este enfoque se hace posible porque el hardware para un data lake suele ser muy diferente del utilizado para un data warehouse. La ampliación de un data lake a terabytes y petabytes puede hacerse de manera bastante económica.

#### 2. Un Data Lake soporta todos los tipos de datos

Los data warehouses generalmente se componen de datos extraídos de sistemas transaccionales junto con métricas cuantitativas y los atributos que las describen. Las fuentes de datos no tradicionales, como los registros del servidor web, los datos de sensores, la actividad de las redes sociales, el texto y las imágenes, se ignoran en gran medida. Se siguen encontrando nuevos usos para estos tipos de datos, pero consumirlos y almacenarlos puede ser costoso y difícil.

**El enfoque del data lake abarca estos tipos de datos no tradicionales. En el data lake, guardamos todos los datos independientemente de la fuente y la estructura. Los mantenemos en su forma bruta y sólo los transformamos cuando estamos listos para usarlos**. Este enfoque se conoce como "Schema on Read" en comparación con el "Schema on Write" que es el enfoque utilizado en el data warehouse.

#### 3. Un Data Lakes soporta a todos los usuarios

En la mayoría de las organizaciones, el 80% o más de los usuarios son "operacionales". Quieren obtener sus informes, ver sus KPIs o seleccionar el mismo conjunto de datos en una hoja de cálculo todos los días. El data warehouse suele ser ideal para estos usuarios porque está bien estructurado, fácil de usar y comprender y está diseñado para responder a sus preguntas.  
  
El siguiente 10% más o menos, hace más análisis en esos datos. Utilizan el data warehouse como una fuente, pero a menudo vuelven a los sistemas de origen para obtener datos que no están incluidos en el almacén y a veces traen datos de fuera de la organización. Su herramienta favorita es la hoja de cálculo y crean nuevos informes que a menudo se distribuyen en toda la organización. El data warehouse es su fuente de acceso a los datos, pero a menudo van más allá de sus límites  
  
Por último, el restante tanto por ciento de los usuarios hace un análisis profundo. Pueden crear fuentes de datos totalmente nuevas basadas en la investigación. Ellos mezclan muchos tipos diferentes de datos y llegan a nuevas preguntas que deben responderse. Estos usuarios pueden utilizar el data warehouse, pero a menudo lo ignoran, ya que normalmente se les solicita que vayan más allá de sus capacidades. Estos usuarios incluyen a los científicos de datos y pueden utilizar avanzadas herramientas analíticas y capacidades como el análisis estadístico y el modelado predictivo.  
  
El enfoque del data lake soporta igualmente a todos estos usuarios. Los científicos de datos pueden ir al data lake y trabajar con el gran y variado conjunto de datos que necesitan, mientras que otros usuarios hacen uso de vistas más estructuradas de los datos proporcionadas para su uso.

#### 4. Los Data Lakes se adaptan fácilmente a los cambios

Una de las principales quejas sobre los data warehouses es cuánto tiempo se tarda en cambiarlos. Un tiempo considerable se gasta por adelantado durante el desarrollo de la estructura del almacén. Un buen diseño de almacén puede adaptarse al cambio, pero debido a la complejidad del [proceso de carga de datos](https://www.powerdata.es/el-valor-de-la-gestion-de-datos/bid/352729/mapeo-y-carga-de-datos-culminando-el-proceso-de-migraci-n) y al trabajo realizado para facilitar el análisis y la elaboración de informes, estos cambios necesariamente consumirán algunos recursos de desarrolladores y tomarán algún tiempo.

Muchas preguntas comerciales no pueden esperar a que el equipo del data warehouse adapte su sistema para responderlas. La necesidad cada vez mayor de respuestas más rápidas es lo que ha dado lugar al concepto de auto-servicio de inteligencia empresarial.

En el data lake, por otro lado, como todos los datos se almacenan en bruto y siempre con accesibles a alguien que necesite utilizarlos, los usuarios tienen el poder de ir más allá de la estructura del almacén para explorar datos de nuevas maneras y responder a sus preguntas a su ritmo.

Si se demuestra que el resultado de una exploración es útil y existe el deseo de repetirlo, entonces se puede aplicar un esquema más formal y se puede desarrollar la automatización y la reutilización para ayudar a extender los resultados a un público más amplio. Si se determina que el resultado no es útil, puede descartarse y no se han realizado cambios en las estructuras de datos ni se han consumido recursos de desarrollo.

#### 5. Los Data Lakes proporcionan una visión más rápida

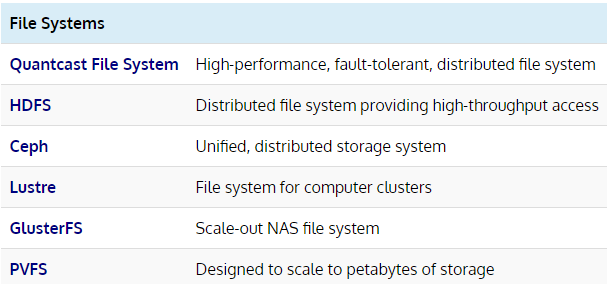
Esta última diferencia es realmente el resultado de las otras cuatro. Debido a que los data lakes contienen todos los datos y tipos de datos, y a que permite a los usuarios acceder a los datos antes de que se hayan transformado, limpiado y estructurado, permite a los usuarios llegar a sus resultados más rápido que el método tradicional de data warehouse.

Sin embargo, este acceso temprano a los datos tiene un precio. El trabajo típicamente realizado por el equipo de desarrollo de data warehouse no se puede hacer para algunas o todas las fuentes de datos requeridas para realizar un análisis. Esto permite a los usuarios explorar y usar los datos como mejor les parezca, pero el primer nivel de usuarios de negocios que he descrito anteriormente tal vez no quiera hacer ese trabajo. Todavía quieren sus informes y KPI's.

En los data lakes, estos consumidores de informes operativos harán uso de vistas más estructuradas de los datos en el data lake que se parecen a lo que siempre han tenido antes en el data warehouse. La diferencia es que estas vistas existen principalmente como metadatos que se sitúan sobre los datos en el lago en lugar de tablas físicamente rígidas que requieren un desarrollador para cambiarlas.

30.- ¿Existe algún otro tipo de sistemas de archivos distribuidos que NO sea HDFS? si es así, ¿de cuáles se trata?

https://www.linuxlinks.com/filesystems/



**Quantcast File System (QFS)** es un sistema de archivos distribuidos de alto rendimiento, de fuente abierta, tolerante a fallas, desarrollado para admitir el procesamiento de MapReduce u otras aplicaciones que leen y escriben grandes archivos secuencialmente. Es una alternativa a HDFS para el procesamiento de datos por lotes a gran escala.

QFS consta de 3 componentes:

Metaserver: un servidor de metadatos de proceso central que administra la estructura del directorio y los mapas de los archivos para el almacenamiento físico

Chunk Server: el componente distribuido del sistema de archivos distribuidos. Un servidor de fragmentos se ejecuta en cada equipo que alojará datos, gestionará E / S en sus discos duros y supervisará su actividad y capacidad.

Client Library: una biblioteca que proporciona la API del sistema de archivos para permitir que las aplicaciones interactúen con QFS. Realiza solicitudes al metaservidor para identificar qué servidores de fragmentos mantienen (o mantendrán) sus datos, luego interactúa directamente con los servidores de fragmentos para leer y escribir.

**Ceph** es un sistema de almacenamiento distribuido, masivo, de código abierto, unificado y escalable diseñado para un excelente rendimiento, confiabilidad y escalabilidad.

Los principales objetivos de Ceph deben distribuirse por completo sin un solo punto de falla, escalable al nivel exabyte y libremente disponible. Los datos se replican sin problemas, por lo que es tolerante a las fallas.

Está compuesto por un almacén de objetos, una tienda de bloques y un sistema de archivos distribuidos. Ceph funciona con hardware básico, es autocurable y autogestionable, y no tiene un solo punto de falla.

**Lustre** es un sistema de archivos masivamente distribuido, paralelo y global, generalmente utilizado para la computación en clúster a gran escala.

Los sistemas de archivos Luster están disponibles bajo GNU GPL (solo v2) y proporcionan un sistema de archivos de alto rendimiento para clústeres de computadoras que varían en tamaño desde pequeños grupos de trabajo hasta clusters de múltiples sitios a gran escala.

Hay dos componentes principales del servidor Lustre de un sistema de archivos Lustre: Nodos de servidores de almacenamiento de objetos (OSS) y nodos de servidores de metadatos (MDS). Los metadatos del sistema de archivos se almacenan en los nodos Lustre MDS y los datos del archivo se almacenan en los Servidores de almacenamiento de objetos. Los datos para el servidor MDS se almacenan en un Meta Data Target (MDT), que corresponde esencialmente a cualquier LUN que se utilice para almacenar los metadatos reales. Los datos para los servidores OSS se almacenan en LUN de hardware denominados Object Storage Targets (OST).

**GlusterFS** es un sistema de archivos distribuido de código abierto capaz de escalar hasta varios petabytes y manejar miles de clientes.

GlusterFS agrupa bloques de almacenamiento en interconexión Infiniband RDMA o TCP / IP, agregando recursos de disco y memoria y administrando datos en un solo espacio de nombres global. GlusterFS se basa en un diseño de espacio de usuario apilable y puede ofrecer un rendimiento excepcional para diversas cargas de trabajo.

**PVFS** es un sistema de archivos paralelos de código abierto de alto rendimiento dirigido a entornos de computación paralelos de producción. Está diseñado específicamente para escalar a un gran número de clientes y servidores. La arquitectura es muy modular, lo que permite una fácil inclusión de nuevos soportes de hardware y nuevos algoritmos. Esto hace que PVFS sea un banco de pruebas de investigación excelente también.

PVFS se centra en el acceso de alto rendimiento a grandes conjuntos de datos. Consiste en un proceso de servidor y una biblioteca de cliente, ambos escritos enteramente de código de nivel de usuario.

PVFS está diseñado para proporcionar un alto rendimiento para aplicaciones paralelas, donde concurren, gran IO y muchos accesos de archivos son comunes.

**SECCIÓN 4. ESPECIAL**

31.- Instale Jupyter en Cloudera, para ello puede basarse en la siguiente liga:

<https://medium.com/@vando/install-jupyter-notebook-on-centos-7-1d596abf08da>

**Es importante señalar que para continuar el curso es imprescindible esta herramienta y no existirán pausas para su instalación durante las sesiones, motivo por el cual es menester llevar a cabo esta operación aunque solamente valga 1 crédito. Para validar este ejercicio se requiere una captura de pantalla del menú principal, algo así:**

