UNIVERSIDAD NACIONAL DE INGENIERÍA

CTIC UNI
CENTRO DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIONES
UNIVERSIDAD NACIONAL DE INGENIERÍA

CIENCIA DE DATOS II:

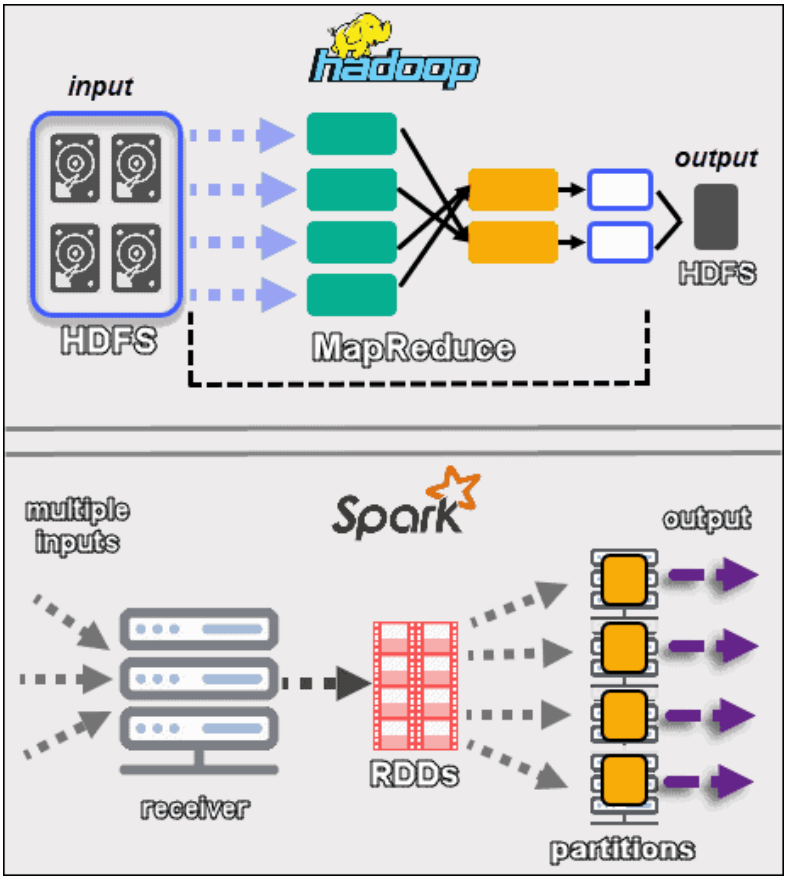
SESION 01

BIG DATA

*Las imágenes y graficos no me pertenecen y son usados solo con fines educativos

ÍNDICE

OBJETIVO



La disrupción tecnológica y los modelos derivados, han creado una especialidad que cada vez cobra más relevancia en el mundo moderno: La Ciencia de Datos. El Data Scientist es un profesional capaz de analizar realidades complejas y resolver problemas de negocio a través de la explotación de grandes volúmenes de datos, siendo capaz de ser el puente y traductor entre las áreas técnicas de la empresa y la alta gerencia.

La definición de Big Data es que son datos que contienen una mayor variedad, que llegan en volúmenes crecientes y con más velocidad. Esto también se conoce como las tres Vs. En pocas palabras, los grandes datos son conjuntos de datos más grandes y complejos, especialmente de nuevas fuentes de datos.

Los datos masivos tienen varias características además de ser muy abundantes, entre ellas, heterogeneidad, complejidad, veracidad, desestructuración, falta de completitud, y tener potencial para ser erróneos. Por esta razón, al diseñar procesos para gestionar datos, debemos tener en cuenta todos estos aspectos con el objetivo de garantizar y preservar su calidad, así

como la extracción de información útil y sin errores, lo cual garantizará la fiabilidad de los datos y, por ende del análisis resultante.

Actualmente son incontables los sistemas servicios o entornos que transmiten datos tanto a nivel individual como a nivel profesional cada vez queremos y necesitamos más y más datos para manipularlos, monitorizarlos o obtener un valor de ellos para asi poder tomar decisiones en tiempo real.

Big Data es la base en la que se fundamenta todo esto, el Big Data nos proporcionara todo tipo de conocimiento, por ejemplo si queremos predecir con un alto grado de probabilidad las tendencias del mercado, si un producto o servicio es el más adecuado o que productos son los más demandandos por tus clientes asi como para cuando y dónde lo demandan.

Como sabemos actualmente la competencia en el mercado es cada vez más fuerte y el primero que toma decisiones y lleva a cabo acciones es quien suele tener más probabilidades para sacar mejores beneficios resultando en un mejor posicionamiento de sus productos en el mercado.

El Big Data ha existido hace muchos años pero es ahora principalmente en este siglo cuando se ha comenzado a hablar mas del concepto en los diversos medios de comunicación. Las nuevas tecnologías Cloud han contribuido en gran medida a esta repercusion y esto se debe a que a medida que avanza la tecnología disponemos a nuestro alrededor de más y más sistemas que absorven y generan datos al mismo tiempo. Me explico, por ejemplo una interfaz puede capturar datos de temperatura de un laboratorio. Esa data servira para generar informacion y o disparar alarmas del estado en el que se encuentra algun observable, en este caso la temperatura.

A veces toda esta data estara debidamente estructurada y en otras no pero independientemente de eso todos los datos o casi todos deben ser guardados para ser analizados y procesados con lo cual nos proporcionaran valor y conocimiento para la toma de decisiones. En cuanto a entornos o sistemas que generan muchísimos datos tenemos por ejemplo los smartwatches o relojes inteligentes los smartphones, tenemos dispositivos como Alexa, Siri y actualmente sistemas IOT. Por ejemplo las lavadoras, frigoríficos y demás electrodomésticos modernos conectados a internet o al menos transmitiendo datos para diversos propósitos También tenemos la actividad generada por las redes sociales, streammings etc

En el mundo de las IT cada vez más sistemas proporcionan datos con un aumento exponencial de los mismos, los mismos pueden provenir de sistemas electrónicos instaladas en una línea de producción de algun producto o de sistemas robotizados. Los correos electrónicos también se consideran datos así como los vídeos, fotos, tablas Excel, las bitacoras de los sistemas operativos, los datos de geolocalización, sistemas tracking o seguimiento logístico información de cumplimiento, eventos de ciberseguridad etc.

Como podemos ver los origenes de datos pueden ser incontables por lo tanto el Big Data es el ecosistema que generamos y desplegamos en torno a todo este conjunto de datos así como el modo en que se capturan, gestionan y se manejan todos estos datos con objeto de impulsar un negocio o ciertas áreas del mismo. Esto es lo que da vida y de lo que trata el Big Data con esto impulsamos la inteligencia de los negocios para una mejor toma de decisiones y acciones a llevar a cabo es decir el Big Data es precursor de los sistemas y soluciones de BI pero también

impulsa la innovación empresarial y cientifica. El Big Data permite realizar análisis predictivos, prescriptivos, descriptivos de diagnóstico etc.

Las corporaciones modernas disponen de infinidad de fuentes de datos sistemas que se crean de manera continua conformando así el *ecosistema* Big Data, estos conjuntos de datos son tan grandes y tan complejos que ya no pueden tratarse ni gestionarse como se venía haciendo dessde el siglo pasado ie. mediante aplicaciones de gestión y procesamiento de datos tradicionales.

Para aprovechar al maximo al ecosistema de Big Data hay que realizarlo mediante soluciones mucho más avanzadas, muchas de las cuales tienen ya una buena madurez en el mercado es decir que no es algo nuevo, ahora bien hay cinco características que definen un ecosistema de Big Data y se le suelen denominar las cinco V. Aunque en otros artículos se mencionan algunas más. Estas cinco V son el volumen, la variedad, la veracidad, la velocidad y el valor.

El volumen, esta **característica** identifica el Big Data con un ecosistema compuesto por un gran volumen de datos procedentes de unas y otras fuentes, cuando hablamos de grandes volúmenes de datos nos referimos ya no a muchos gigabytes sino terabytes y petabytes de datos.

La variedad, esta característica se refiere a la diversidad y heterogeneidad de los datos procedentes de diversas fuentes las cuales generan datos de todo tipo y proporcionan datos tanto estructurados como no estructurados. Los datos estructurados son aquellos que ya están identificados, etiquetados y se pueden ordenadar y categorizar entre otras cosas, es decir son aquellos datos sobre los cuales se pueden aplicar filtros, ordenarlos, obtener reportes etc dado que previamente alguien o un programa de manera automática los ha estructurado, por ejemplo los datos en una hoja Excel. Por el contrario los datos no estructurados son aquellos que aún no han sido analizados y debidamente identificados, etiquetados, divididos, categorizados etc, están desorganizados por decirlo de alguna manera. Tengamos en cuenta que el volumen de datos no estructurados suele ser mayor que los datos estructurados, los correos electrónicos, los vídeos, los pdfs, documentos de texto etc son datos no estructurados. Se puede extraer informacion de ellos pero previamente hay que realizar una manipulacion para lograr que se conviertan en estructurados.

La velocidad esta se refiere a la velocidad a la que se generan y almacenan estos datos para que puedan ser procesados en algun momento o en tiempo real. A la fecha de hoy la infinidad de fuentes estan transmitiendo continuamente datos como por ejemplo los sensores, los dispositivos móviles, los logs de aplicaciones, de geolocalización etc y la regularidad con la que lo hacen con la que transmiten esos datos es impresionante cuanto menos abrumadora a esto se refiere la característica de velocidad.

La veracidad con esta característica se define que tan fiables son esos datos, es decir si podemos confiar en ellos, si son datos de calidad para utilizar estos datos primero nos debemos asegurar que sean precisos, que esten limpios de ruido que pudieran desvirtuar su propósito.

El valor, con el Big Data transformamos datos en información es decir en conocimiento para tomar decisiones y acciones es decir nos proporcionan valor. No todos los datos que obtenemos poseen la misma importancia y ponderación tenemos que darnos cuenta qué datos en sinergia y correlación con otros pueden aportar valor. Para que esto suceda hay un gran esfuerzo realizado por especialistas.

Qué aplicaciones o ejemplos del big Data hay en la vida real muchísimas pero por nombrar algunas el New York Stocks Exchange, mercado de valor en USA, entornos de Big Data que genera más de un terabyte de información diaria.

En este momento varios Facebook genera del orden de más de 500 terabytes de nuevos datos cada día en la modalidad de fotos, vídeos, mensajes, comentarios etc Los sistemas de Aviación pueden generar más de 10 terabytes de datos cada 30 minutos y sabiendo que hay miles de vuelos al día imaginemos de cuanta data estamos hablando.

En las compañías dedicadas a la logística por ejemplo se emplea el Big Data para mejorar los cálculos los tiempos de tránsito rutas u otras etc, las compañías farmacéuticas para mejorar el proceso de investigación, la cadena de producción de nuevos fármacos etc. En la investigación médica o científica mejorando el analisis de datos como por ejemplo el avance en el Laboratorio Lawrence Livermore para lograr la fusion nuclear.

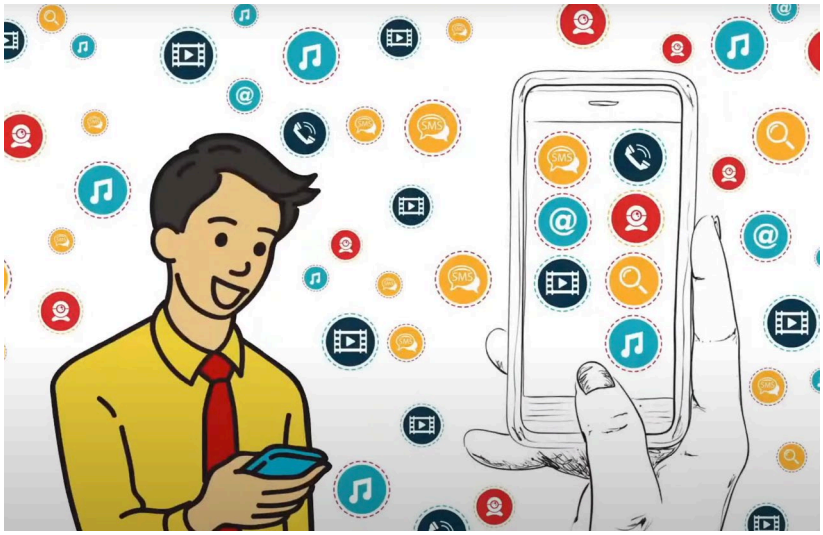
Ahora veamos algunos de los temas que tocaremos en esta y las proximas sesiones del curso:

Hadoop y Spark, ambos desarrollados por Apache Software Foundation, son frameworks de código abierto ampliamente utilizados para arquitecturas de big data. Cada framework contiene un extenso ecosistema de tecnologías de código abierto que preparan, procesan, administran y analizan grandes conjuntos de datos.

Apache Hadoop es una utilidad de software de código abierto que permite a los usuarios administrar grandes conjuntos de datos (desde gigabytes hasta petabytes) al habilitar una red de computadoras (o "nodos") para resolver problemas de datos extensos e intrincados. Es una solución altamente escalable y rentable que almacena y procesa datos estructurados, semiestructurados y no estructurados (por ejemplo, registros de flujo de clics de Internet, registros de servidores web, datos de sensores de IoT, etc.).

Apache Spark, que también es de código abierto, es un motor de procesamiento de datos para grandes conjuntos de datos. Al igual que Hadoop, Spark divide tareas grandes en diferentes nodos. Sin embargo, tiende a funcionar más rápido que Hadoop y utiliza memoria de acceso aleatorio (RAM) para almacenar en caché y procesar datos en lugar de un sistema de archivos. Esto permite que Spark maneje casos de uso que Hadoop no puede.

INTRODUCCION – BIG DATA



¿Qué son los datos?

Las cantidades, caracteres o símbolos sobre los que una computadora realiza operaciones, que pueden almacenarse y transmitirse en forma de señales eléctricas y grabarse en medios de grabación magnéticos, ópticos o mecánicos.

¿Qué es un ecosistema de datos?

Cuando pensamos en los datos de una empresa, es posible que nos venga a la mente imágenes de hojas de cálculo, bases de datos, gráficos y diagramas. Si bien estos son críticos para la estructura de datos de una organización, son solo una pequeña porción de un ecosistema de datos más grande. Tanto si es un científico de datos ambicioso o un analista que quiere trabajar directamente con datos o un administrador que depende de los datos para la toma de decisiones, debemos conocer los componentes que forman el ecosistema de datos de una organización.

Un ecosistema de datos comprende la infraestructura y las aplicaciones de una empresa que colaboran para recopilar y administrar datos. Estos programas recopilan y filtran datos para comprender mejor a sus usuarios, visitantes del sitio web y miembros de la audiencia. Comenzar con una herramienta de análisis de productos y expandirse es el mejor método para crear un ecosistema de datos viable.

Debido a que es un ecosistema en lugar de un entorno, sus aplicaciones e infraestructura deben construirse para adaptarse con el tiempo para satisfacer las demandas cambiantes de su organización. No existe una solución única para un ecosistema de datos, ya que cada organización tiene objetivos y metas diferentes. Las aplicaciones y los servicios en la nube que componen la infraestructura deben poder adaptarse a futuros cambios y actualizaciones, lo que permite que una organización se desarrolle.

¿Por qué crear un ecosistema de datos?

Los ecosistemas de datos recopilan datos para proporcionar información relevante. Los clientes dejan rastros de datos cuando utilizan cosas, especialmente las digitales. Las empresas pueden construir un ecosistema de datos para registrar y analizar rastros de datos para que los equipos de productos puedan descubrir qué disfrutan, disgustan y responden favorablemente sus consumidores. Los equipos de productos pueden utilizar conocimientos para mejorar el producto ajustando las funciones. Inicialmente, los ecosistemas se denominaban entornos de tecnología de la información. Estaban destinados a ser centralizados e inmóviles. Eso ha cambiado con la llegada de la web y los servicios en la nube. Los datos se registran y utilizan en todas las empresas, y los trabajadores de TI tienen un control menos centralizado.

Ahora deben adaptar y actualizar continuamente la infraestructura que utilizan para recopilar datos. Como resultado, la ecología de datos se refiere a los ecosistemas de datos creados para adaptarse. No existe tal cosa como una solución de "ecosistema de datos". Cada empresa crea su ecosistema, a veces conocido como “tech stack”, y lo llena con una mezcla de hardware y software para recopilar, almacenar, analizar y actuar sobre los datos. Los ecosistemas de datos más exitosos se basan en una plataforma de análisis de productos que conecta el ecosistema.

Los sistemas de análisis ayudan a los equipos a integrar numerosas fuentes de datos, proporcionando tecnologías de aprendizaje automático para automatizar el proceso analítico y rastreando a los usuarios para que los equipos puedan generar indicadores de rendimiento.

¿Cuáles son los componentes principales de un ecosistema de datos?

El ecosistema de datos se compone de tres componentes principales:

- Fuentes de datos

  Como su nombre lo dice, las fuentes de datos son de donde provienen sus datos. ¿Cómo recopila datos su infraestructura y de dónde provienen esos datos? Este componente también implica integrar diferentes datos para sacar conclusiones valiosas en la etapa de análisis.

- Gestión de datos

  La gestión de datos incluye la conexión de aplicaciones, el almacenamiento de datos y la determinación de cómo su infraestructura convierte los datos en informes utilizables. La captura de datos se vuelve casi inútil sin la capacidad de retenerlos y acceder a ellos correctamente en el futuro.

- Análisis de datos

  Sin pensamiento crítico y un análisis cuidadoso, los datos no tienen sentido. De lo contrario, se quedará con una gran cantidad de datos que no se convertirán en ideas significativas. El proceso de extraer información de los informes y darle sentido a la información se conoce como análisis de datos.

Las herramientas de análisis pueden ayudarlo a comprender por qué los consumidores específicos abandonan mientras que otros permanecen leales a su empresa.

Otros componentes del ecosistema de datos:

- Detección:

  El proceso de localizar fuentes de datos para su proyecto se denomina detección. Implica evaluar la calidad de los datos para determinar si vale la pena. Esta evaluación implica hacer preguntas como:

  ¿Es correcta la información?
  ¿La información está actualizada y actualizada?
  ¿Está completa la información?
  ¿Es precisa la información? ¿Se puede confiar?
  Las fuentes de datos internas incluyen bases de datos, hojas de cálculo, CRM y otras aplicaciones. También puede provenir de diferentes fuentes, como sitios web o agregadores de datos de terceros.

- Recopilación:

Los datos deben recopilarse una vez que se haya identificado una fuente de datos viable.

Se pueden utilizar técnicas manuales o automatizadas para recopilar datos. Por lo general, no es factible contener cantidades significativas de datos manualmente. Es por eso que los científicos de datos crean software en lenguajes de programación para automatizar el proceso de recopilación de datos.

Por ejemplo, es factible desarrollar programas que "recuperen" información útil de una página web (acertadamente llamado web scraper). También es posible crear y desarrollar una interfaz de programación de aplicaciones, o API, para recuperar información directamente de una base de datos o comunicarse con un servicio web.

- Data wrangling:

  El data wrangling es una colección de métodos para convertir datos sin procesar en un estado más utilizable. Según la calidad de los datos, puede implicar la combinación de conjuntos de datos dispares, la búsqueda y el llenado de lagunas en los datos, la eliminación de datos innecesarios o inexactos y la "limpieza" y organización de datos para estudios futuros.

  El data wrangling, como la recopilación de datos, se puede realizar de forma manual o automática. Los procedimientos manuales pueden ser efectivos si el conjunto de datos es lo suficientemente pequeño. El volumen de datos es demasiado grande para la mayoría de las principales iniciativas de datos, lo que requiere automatización.

- Análisis:

  Los datos sin procesar se pueden evaluar después de revisarlos y cambiarlos a un estado funcional. Dependiendo de la dificultad única que pretenda abordar su proyecto de datos, este análisis puede ser de diagnóstico, descriptivo, predictivo

o prescriptivo. Si bien cada tipo de investigación es distinto, todos se basan en los mismos métodos e instrumentos.

Por lo general, su estudio comenzará con cierta automatización, especialmente si su conjunto de datos es amplio. Tras la finalización de los procedimientos automatizados, los analistas de datos aplican su conocimiento para recopilar más información.

- Almacenamiento:

  Los datos deben mantenerse de manera segura y accesible en todas las etapas del ciclo de vida de los datos. Los procesos de gobierno de datos de su organización determinan los medios específicos utilizados para el almacenamiento.

Necesidad de un ecosistema de datos

Las empresas con un ecosistema de datos sólido pueden emitir juicios basados en hechos con respecto a la gestión de operaciones, la fijación de precios y las iniciativas de marketing. Todos los productos, en particular los digitales, rastrean cómo los usuarios interactúan con los productos y servicios de su empresa. Los datos pueden brindar información valiosa sobre por qué los consumidores se comportan de la manera en que lo hacen. Las siguientes secciones analizan las ventajas únicas de los ecosistemas de datos para las empresas.

- Aumenta la participación del usuario

  Los usuarios están mucho más inclinados a iniciar una discusión cuando creen que sus voces están siendo escuchadas. Los comentarios de los usuarios contienen parte de la información más valiosa sobre su organización. Los usuarios pueden discutir sus reacciones a los cambios en su sitio web, productos, servicios o publicidad una vez que exista un ecosistema de datos.

- Identifica relaciones de datos ocultos

  No siempre es sencillo detectar las conexiones entre varios componentes. ¿Qué características demográficas determinan si un visitante se convierte o no en usuario? ¿A quién debe promocionar su empresa? ¿Por qué los consumidores abandonan su sitio web sin realizar una compra?

- Notifica a los equipos de cambios

  Puede monitorear cómo reaccionan los usuarios a las actualizaciones o cambios en cualquier cosa, desde la estrategia publicitaria de su empresa hasta las funciones, si puede medir las reacciones de los usuarios y las interacciones con sus productos o servicios en tiempo real. De esta manera, sabrá si se está dirigiendo por el camino incorrecto y podrá intentar ajustarlo para que se adapte mejor a su clientela.

- Seguimiento de conversiones y embudos de marketing

  El núcleo de cada negocio es observar sus embudos de marketing para garantizar que los consumidores potenciales sean guiados a través del proceso de completar sus transacciones.

- Aumenta la retención de usuarios

  Al monitorear sus KPI (Key Performance Indicator), puede motivar a los consumidores a continuar usando sus productos y desarrollar la lealtad corporativa al comprender cuándo las personas responden negativamente a nuevas actualizaciones o adiciones.

  Saber lo que quiere su público objetivo es el método más efectivo para cumplir con sus requisitos y hacer que regresen por más.

- Realiza pruebas A/B

  Las pruebas A/B le permiten rastrear la diferencia en la interacción del usuario entre dos versiones distintas del mismo contenido. Esto se puede lograr contrastando dos campañas de correo electrónico con diferentes frases, la misma publicación en las redes sociales creada en múltiples plataformas o conceptos similares. El rendimiento de cada versión se basa en el objetivo de la tasa de conversión, como el porcentaje de personas que hacen clic en un enlace, completan un formulario o hacen una compra.

- Se conecta a otras aplicaciones

  Al automatizar los procedimientos manuales, puede ahorrarle a su personal tiempo y dinero que podría gastar en operar su negocio y mejorar sus productos. Todo el mundo ha oído eso, pero uno de los métodos más excelentes para ayudar a la automatización es vincular sus aplicaciones para que no reciba nada en partes.

  Al conectar sus aplicaciones a un sistema de organización central, garantiza que todo funcione correctamente y que los datos se le envíen para recopilarlos en informes limpios. Estos informes le permiten ver el panorama general y desarrollar planes.

  Además, la integración permite que varios equipos tengan acceso simultáneo a la información.

¿Cómo crear un ecosistema de datos?

Cada ecosistema de datos consta de tres componentes:

- Infraestructura

  La infraestructura es la base de un ecosistema de datos. Los servicios de hardware y software capturan, recopilan y organizan los datos. Los servidores de almacenamiento, los lenguajes de búsqueda como SQL y las plataformas de alojamiento son parte de la infraestructura. La infraestructura puede recopilar y almacenar tres categorías de datos: datos estructurados, no estructurados y multiestructurados. Los datos estructurados, como su nombre indica, están limpios, etiquetados y ordenados, como el número total de visitantes del sitio guardados en una hoja de cálculo de Excel desde un sitio web.

  Los datos no estructurados son información que no se ha organizado para su análisis, como el texto de las publicaciones. Los datos multiestructurados

provienen de varias fuentes en varios formatos; puede ser una combinación de datos estructurados y no estructurados. Si los ecosistemas incluyen una gran cantidad de datos, se requerirán más herramientas para que los equipos puedan acceder a ellos más fácilmente. Los equipos pueden utilizar la tecnología Hadoop o Not Only SQL (NoSQL) para particionar sus datos y permitir consultas más rápidas.

- Analítica

  Analytics es la entrada principal a través de la cual los equipos ingresan a su ecosistema de datos. Las plataformas de análisis buscan y resumen los datos contenidos dentro de la infraestructura y conectan elementos de la infraestructura para que todos los datos estén disponibles en una sola ubicación. Si bien los sistemas de infraestructura tienen análisis rudimentarios, estos rara vez son adecuados. Una plataforma de análisis especializada siempre podrá profundizar considerablemente en los datos, tener una interfaz mucho más fácil de usar y presentar una gama de herramientas diseñadas para ayudar a los equipos a realizar cálculos más rápidamente.

  Por ejemplo, aunque un servidor de aplicaciones puede decirle a un equipo cuántos datos procesa su aplicación, una plataforma de análisis puede ayudar a identificar a todos los usuarios individuales dentro de esos datos, rastrear lo que cada uno está haciendo ahora y predecir sus próximos pasos. Solo el análisis puede segmentar y rastrear a los usuarios a través de embudos de marketing, determinar las características de los clientes ideales y enviar automáticamente mensajes dentro de la aplicación a aquellos en riesgo de abandono.

- Aplicaciones

  Las aplicaciones son las paredes y el techo de la casa del ecosistema de datos: son servicios y sistemas que operan con datos y los hacen utilizables. Un equipo de producto, por ejemplo, puede optar por importar datos analíticos a sus sistemas de marketing, ventas y operaciones. Esto permitiría que el equipo de marketing califique clientes potenciales en función de la actividad, que el equipo

de ventas reciba notificaciones cuando interactúen los clientes potenciales ideales y que el equipo de operaciones cobre a los clientes automáticamente en función del uso del producto.

Aspectos a tener en cuenta al crear un ecosistema de datos

- Gobierno de Datos

  Las empresas deben desarrollar lineamientos explícitos de gobierno de datos en una era en la que TI ya no tiene una supervisión de datos centralizada aparente, a menudo publicando una política interna sobre cómo se pueden recopilar, utilizar, conservar, asegurar y eliminar los datos. Muchos equipos de productos se ven obligados a ser más abiertos debido a la legislación como el RGPD de la Unión Europea, pero aquellos que quieran generar confianza con sus consumidores deben estar a la vanguardia. Los principios de gobierno de datos de cada organización deben publicarse y seguirse.

- Democratizar la ciencia de datos

  La mayoría de los equipos pueden beneficiarse de la información del consumidor, pero si solo una persona tiene acceso al informe, esa persona se convierte en un cuello de botella. Muchas empresas invierten en soluciones de análisis con interfaces fáciles de usar que permiten que cualquier persona acceda a los datos.

  Comprender la ecología de datos de su empresa es el primer paso para segmentar e identificar a sus usuarios. Con esa información, podrá promocionar eficazmente su producto o servicio a un público más amplio con demandas comparables, y podrá personalizar su empresa para que sea exactamente lo que sus consumidores quieren ver.

¿Qué es Big Data?

Big Data es una colección de datos que es enorme en volumen, pero que crece exponencialmente con el tiempo. Se trata de datos con un tamaño y una complejidad tan grandes

que ninguna de las herramientas tradicionales de gestión de datos puede almacenarlos o procesarlos de manera eficiente. Big data también es un dato pero con un tamaño enorme.

¿Cuales son ejemplos de Big Data?

Los siguientes son algunos de los ejemplos de Big Data:

- La Bolsa de Valores de Nueva York es un ejemplo de Big Data que genera alrededor de un terabyte de nuevos datos comerciales por día.
- La estadística muestra que más de 500 terabytes de datos nuevos se introducen en las bases de datos del sitio de redes sociales Facebook todos los días. Estos datos se generan principalmente en términos de carga de fotos y videos, intercambio de mensajes, comentarios, etc.
- Un solo motor Jet puede generar más de 10 terabytes de datos en 30 minutos de tiempo de vuelo. Con muchos miles de vuelos por día, la generación de datos alcanza muchos Petabytes.

Las 10 v's del Big Data

1 - Volumen

El volumen es la cantidad masiva de datos que se generan cada segundo, minuto, hora o cualquier otra cifra de tiempo estimada. Han de ser grandes cifras de datos para considerarse Big Data. Por ejemplo, Youtube almacena 18.000 segundos de vídeo por minuto de sus usuarios.

2 - Velocidad

La velocidad a la que se generan o actualizan los datos. Un ejemplo de ello es Google, que procesa unas " 40,000 consultas de búsqueda por segundo ", lo que se traduce apróximadamente en más de 3,5 mil millones de búsquedas por día.

3 - Variedad

Cuando se trata de Big Data, no solo tenemos que manejar datos estructurados, sino también semiestructurados y principalmente no estructurados. Desde archivos de audio, imagen, video, actualizaciones de redes sociales y otros formatos de texto, hasta archivos de registro, datos de clics, de máquinas y sensores, etc.

4 - Variabilidad

La variabilidad en el contexto de big data tiene dos significados:
Uno es el número de inconsistencias en los datos. Estos deben ser encontrados por métodos de detección de anomalías y valores atípicos para que ocurra cualquier análisis significativo.
Otro es la multitud de dimensiones de datos que resultan de múltiples tipos y fuentes de datos dispares. La variabilidad también puede referirse a la velocidad inconsistente a la que se cargan grandes datos en bases de datos.

5 - Veracidad

La veracidad se refiere a la procedencia o confiabilidad de la fuente de datos, su contexto y cuán significativo es para el análisis basado en ella. A medida que aumentan algunas o todas las propiedades anteriores la veracidad disminuye.

Ejemplo: imagine un conjunto de datos estadísticos sobre lo que la gente compra en los restaurantes y los precios de estos artículos en los últimos cinco años. Puede preguntar: ¿Quién creó la fuente? ¿Qué metodología siguieron para recopilar los datos? ¿Solo se incluyeron ciertas cocinas o ciertos tipos de restaurantes? ¿Los creadores de datos resumieron la información? ¿Esta ha sido editada o modificada por alguien más?

6- Validez

Se refiere a la limpieza que tienen los datos, a cuán precisos y correctos son para su uso. El beneficio del análisis de Big Data es tan bueno como sus datos subyacentes, por lo que se deben adoptar buenas prácticas de *gobernanza de datos* para garantizar una calidad de datos coherente, definiciones comunes y metadatos.

7 - Vulnerabilidad

Toda preocupación de seguridad respecto a los datos. Se han dado muchos casos de hackeo y violación de macrodatos para posteriores actividades ilegales.

8 - Volatilidad

O el tiempo que deben conservarse los datos. Antes del Big Data, se tendía a almacenar datos indefinidamente debido a que a su pequeño volumen apenas suponía gastos. Incluso podía mantenerse en la base de datos en vivo sin causar problemas de rendimiento.

Sin embargo, debido a la velocidad y el volumen de los macrodatos, su volatilidad debe considerarse cuidadosamente. Ahora hay que establecer reglas para la disponibilidad y la vigencia de estos datos, así como para garantizar una recuperación rápida de la información cuando sea necesario.

9 - Visualización

Otra característica de los grandes datos es la complejidad para visualizarlos. No se puede confiar en los gráficos tradicionales para trazar un billón de puntos de datos, por ejemplo, por lo que son necesarias diferentes formas de representarlos, como la agrupación o el uso de mapas, las coordenadas, los diagramas, etc.

10 - Valor

Por último y posiblemente el más importante de todos. Las otras características no tienen sentido si no se obtiene un valor, como puede ser: comprender mejor a los clientes, optimizar procesos, mejorar el rendimiento, …

¿Quién es un ingeniero de Big Data?

Como se mencionó anteriormente, la generación de datos ha aumentado en todo el mundo. Pero, no sirve de nada hasta que se procese y analice de manera competente. Big Data se analiza para obtener información significativa de él, lo que a su vez mejora el rendimiento general. Al hacerlo, las organizaciones pueden mejorar sus decisiones comerciales, productos y efectividad de marketing. Y los profesionales en el campo del Big Data ayudan en esta tarea.

Uno de los mejores roles laborales en este campo es el de un ingeniero de Big Data. Los ingenieros de Big Data son profesionales que desarrollan, mantienen, prueban y evalúan la infraestructura de Big Data de una empresa. Juegan con Big Data y lo utilizan para el beneficio y crecimiento de la organización.

Los roles de un Ingeniero de Datos y el de un Ingeniero de Big Data son intercambiables. Con el auge de Big Data en el sistema de gestión de datos, también se requiere que los ingenieros de datos manejen Big Data. Imbuyen habilidades de ingeniería de Big Data para este propósito. Por lo tanto, un ingeniero de datos trabaja con varios marcos de Big Data y bases de datos NoSQL para administrar Big Data.

Avancemos y analicemos las diversas responsabilidades de un ingeniero de Big Data, una por una.

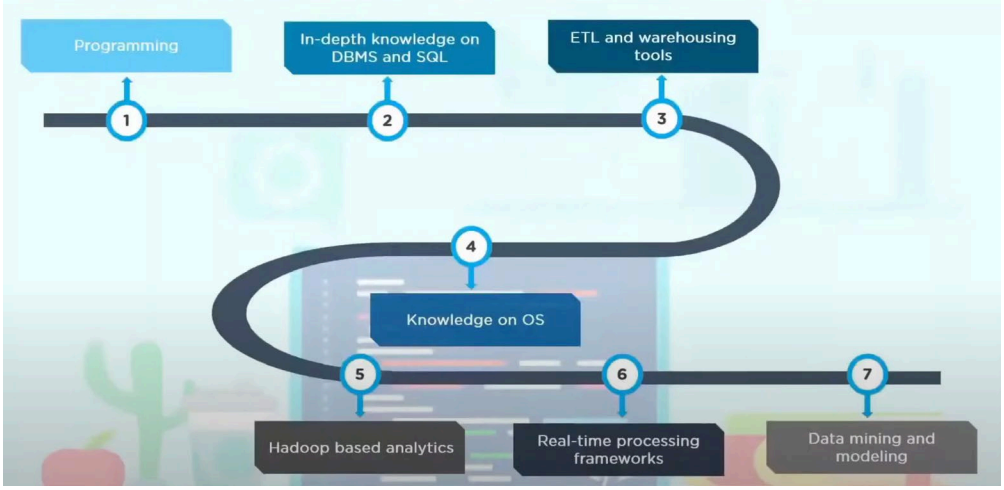
Responsabilidades de un ingeniero de Big Data

Los ingenieros de Big Data tienen un espectro de responsabilidades que van desde el diseño de sistemas de software hasta la colaboración y coordinación con científicos de datos. A continuación se detallan algunas de las funciones de un ingeniero de Big Data:

- En primer lugar, son responsables de diseñar e implementar sistemas de software. También verifican y mantienen estos sistemas.
- Los ingenieros de Big Data también construyen sistemas robustos para la ingesta y el procesamiento de datos.
- Las operaciones de extracción, transformación y carga, conocidas como proceso ETL, son realizadas por ingenieros de Big Data.
- También investigan varios métodos nuevos para obtener datos y mejorar su calidad.
- Los ingenieros de Big Data también son responsables de construir arquitecturas de datos que cumplan con los requisitos comerciales. Se encargan de generar una solución estructurada integrando varios lenguajes de programación y herramientas.
- Su principal responsabilidad es extraer datos de muchas fuentes diferentes para crear modelos comerciales eficientes.
- Finalmente, los ingenieros de Big Data trabajan con otros equipos, analistas de datos y científicos de datos.

Esas son solo algunas de las responsabilidades clave de un ingeniero de Big Data. Estas responsabilidades solo se pueden llevar a cabo si tiene un sólido conjunto de habilidades.

Habilidades del ingeniero de Big Data



Se requiere que un ingeniero de Big Data sea muy hábil en muchas áreas de especialización. A continuación se enumeran las 7 mejores habilidades de Big Data Engineer que necesitará:

- Programación: Comenzando, como la mayoría de los otros roles de trabajo orientados a la tecnología, de todas las habilidades de Big Data Engineer, la programación encabeza

la lista. Un ingeniero de Big Data debe tener experiencia práctica en cualquier lenguaje de programación predominante, como Java, C ++ o Python.

- Base de datos y SQL: Después de la programación viene el conocimiento profundo de DBMS y SQL. Esto ayudará a comprender cómo se gestionan y mantienen los datos en una base de datos. Debe saber cómo escribir consultas SQL para cualquier sistema de gestión de bases de datos relacionales. Algunos de los sistemas de administración de bases de datos comúnmente utilizados para la ingeniería de Big Data son MySQL, Oracle Database y Microsoft SQL Server.
- ETL y almacenamiento de datos: como se mencionó anteriormente, una de las principales responsabilidades de un ingeniero de Big Data es llevar a cabo operaciones de ETL. Para esto, necesitaría saber cómo construir y usar un almacén de datos.

Como ingeniero de Big Data, extraerá datos de varias fuentes, transformándolos en información significativa y cargándolos en otros almacenamientos de datos. Algunas de las herramientas utilizadas para este fin son Talend, IBM Datastage, Pentaho e Informatica.

- Sistema operativo: la cuarta habilidad que necesita es el conocimiento de los sistemas operativos. Las herramientas operativas son la base para ejecutar las herramientas de Big Data. Por lo tanto, es obligatorio tener una sólida comprensión de Unix, Linux, Windows y Solaris.
- Herramientas y framework de Hadoop: debe tener experiencia con análisis basados en Hadoop. Hadoop es una de las herramientas de ingeniería de Big Data más utilizadas, por lo que se entiende que debe tener experiencia con tecnologías basadas en Apache Hadoop como HDFS, MapReduce, Apache Pig, Hive y Apache HBase.
- Apache Spark: La sexta habilidad que requieres es haber trabajado con frameworks de procesamiento en tiempo real como Apache Spark. Como ingeniero de Big Data, se ocupará de enormes volúmenes de datos, por lo que necesita un motor de análisis como Spark, que se puede utilizar tanto para el procesamiento por lotes como en tiempo real. Spark puede procesar datos de transmisión en vivo de varias fuentes como Twitter, Instagram, Facebook, etc.
- Minería y modelado de datos: el requisito de habilidad final requiere que tenga experiencia en técnicas de minería de datos, data wrangling y modelado de datos. La extracción de datos y la data wrangling incluyen pasos para preprocesar y limpiar los

datos utilizando varios métodos, encontrar tendencias y patrones invisibles en los datos y prepararlos para el análisis.

Los ingenieros de Big Data examinan datos preexistentes masivos para descubrir nuevos conocimientos a través del modelado de datos. Algunas de las herramientas utilizadas para esto son Python, R, Rapid Miner, Weka y KNIME.

¿Qué problemas surgieron con Big Data?

- Existen programas que funcionan bien con una pequeña cantidad de datos.
- Al utilizarlos con datos reales el programa falla.
- No se puede cargar un archivo de 10GB en 4 GB de memoria

¿Qué se requiere para Big Data?

- Obtener un cluster de computadoras
- Configurar el cluster
- Aprende una nueva API
- Reescribir codigo

¿Cómo solucionamos usando hardware?

No usar RAM, solamente disco duro:

- Lectura desde SSD: aprox. 16 000 nanosegundos
- Lectura desde RAM: aprox.. 100 nanosegundos
- Si queremos procesamiento rápido la data debe caber en la RAM

Soluciones posibles:

Comprar mas RAM: 6 nucleos 128GB fácil 1000 dolares
Alquilar RAM: Cloud VM 128 nucleos 512 GB fácil 4 dolares/hora

¿Cómo solucionamos usando software?

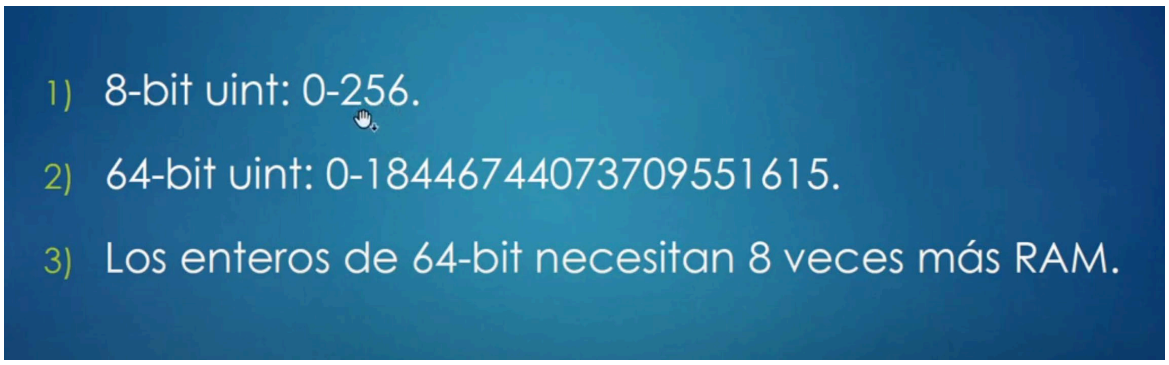
https://smallbigdata.github.io/manifesto.html

- Comprimir datos
- Procesamieto por lotes
- Indexacion de los datos para cargar solo lo necesario

Usando compresion:

- Representa la misma data con menos memoria
- Puede ser sin perdida o con poca perdida

- No se compirme en disco, se comprime la representacion de la data en RAM



Arreglos poco poblados:

- Cuando se tiene arreglos con muchos ceros (u otro valor)
- No se almancenan los ceros
- La librería sparse es compatible con numpy (https://sparse.pydata.org)
- Diferentes tipos de representaciones

Procesamiento por lotes

- Utiliza todos los datos, pero no todos a la misma vez
- Se cargan los datos a memoria por partes
- max(arr) vs max(max(arr[:50]), max(arr[50:]))

Indexado

- Un indice es un resumen de los datos
- Es mucho mas pequeño que los datos, por lo que cabe facilmente en memoria
- Permite encontrar de manera eficiente el subconjunto de los datos que nos interesa

Pandas & Sqlite

- Leer los datos por lotes utilizando pandas
- Ir agregando los datos leidos a una base de datos sqlite (se almancenan en disco duro)
- Ventaja del sqlite en python: permite interactuar con una base de datos sin la existencia de un servidor.

Sqlite vs csv

- Uso de memoria

  Csv: 100 000 filas en cada lote + filas seleccionadas
  Sqlite: indice + seleccionados

- Uso de cpu:

  Csv: aprox 9 segundos cada consulta

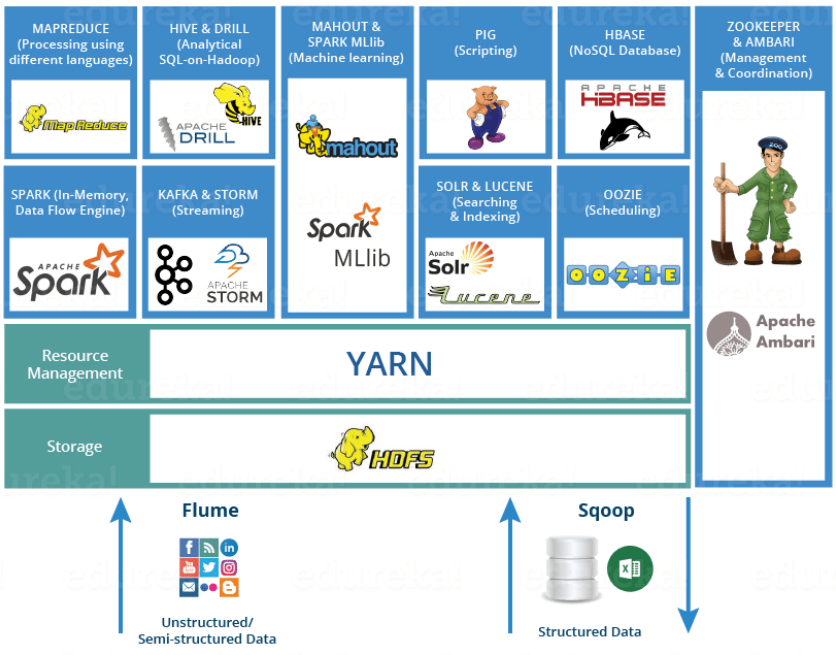
  Sqlite:
  Indexado 65 segundos una sola vez
  Por consulta: 0.2 segundos

Conclusion:

- El problema base siempre es el mismo, la memoria es rapida y cara, el disco duro es barato y lento.

- Otras librerias: Compresion, procesamiento por lotes, paralelismo en el procesamiento, indexado.

INTRODUCCIÓN – HADOOP



¿Qué es Hadoop?

Apache Hadoop es un framework de código abierto que se utiliza para desarrollar aplicaciones de procesamiento de datos que se ejecutan en un entorno informático distribuido.

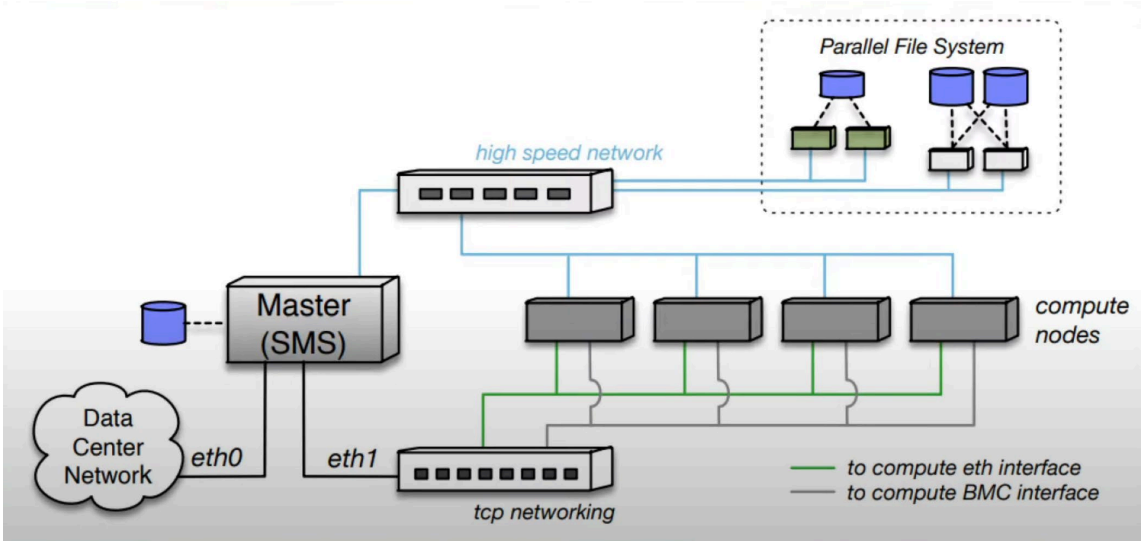
Las aplicaciones creadas con HADOOP se ejecutan en grandes conjuntos de datos distribuidos en grupos de computadoras básicas. Las computadoras comerciales son baratas y ampliamente disponibles. Estos son principalmente útiles para lograr una mayor potencia computacional a bajo costo.

Al igual que los datos que residen en un sistema de archivos local de un sistema informático personal, en Hadoop, los datos residen en un sistema de archivos distribuido que se denomina sistema de archivos distribuidos de Hadoop. El modelo de procesamiento se basa en el concepto de "localidad de datos" en el que la lógica computacional se envía a los nodos del clúster (servidor) que contienen datos. Esta lógica computacional no es más que una

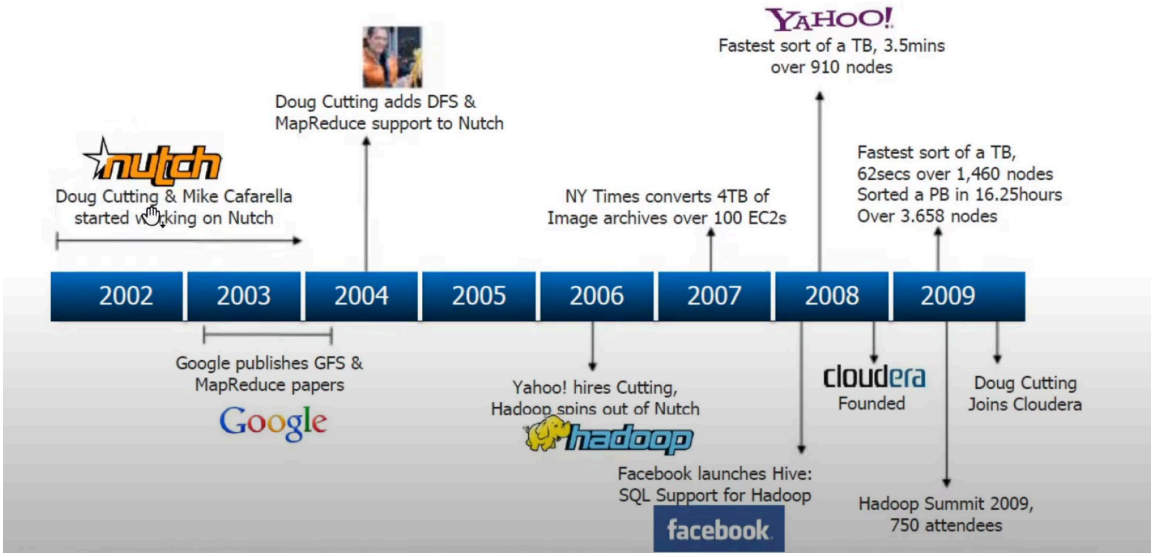
versión compilada de un programa escrito en un lenguaje de alto nivel como Java. Dicho programa procesa los datos almacenados en Hadoop HDFS.

- Plataforma escalable para almacenamiento y análisis.
- Utiliza hardware commodity → $ bajo costo
- Open Source.
  - https://github.com/apache/hadoop
  - https://github.com/apache/hadoop/graphs/contributors
- "Preguntas que toman mucho tiempo en ser respondidas, ahora pueden tener respuestas".
- Velocidad: http://sortbenchmark.org/

Topologia HPC



Nace Hadoop

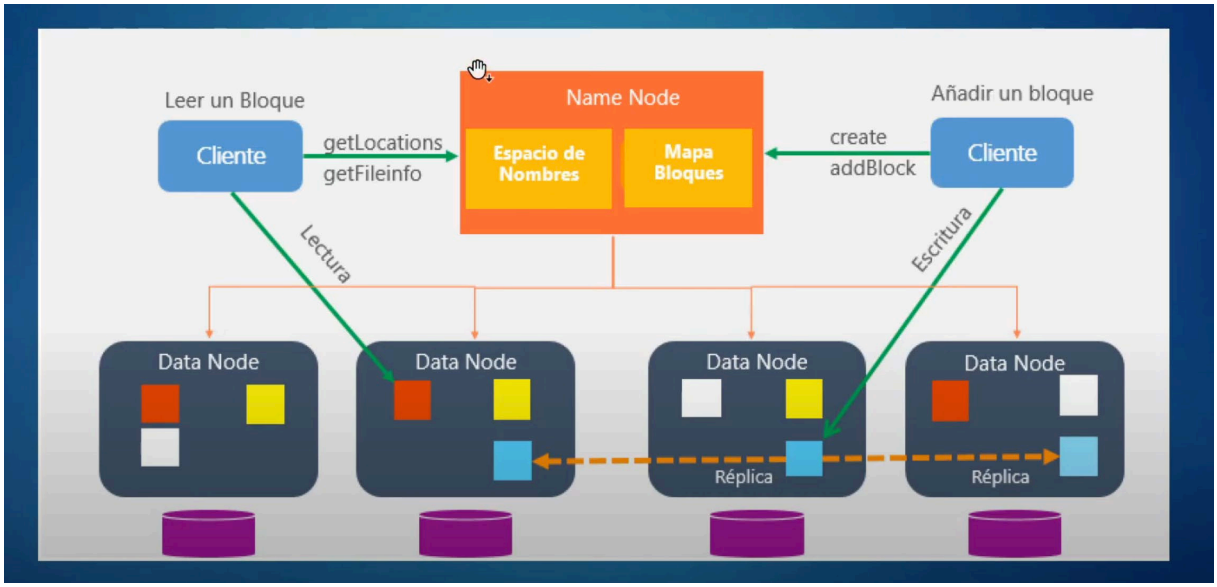


La importancia de Hadoop:

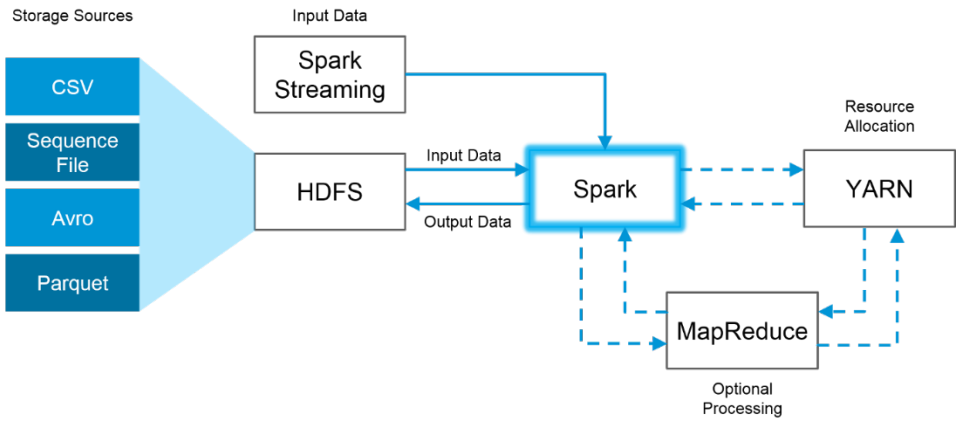
- Capacidad para almacenar y procesar grandes cantidades de cualquier tipo de datos rápidamente. Con volúmenes y variedades de datos en constante aumento, especialmente de las redes sociales y el Internet de las cosas (IoT), esa es una consideración clave.
- Poder computacional. El modelo de computación distribuida de Hadoop procesa big data rápidamente. Cuantos más nodos informáticos utilice, más potencia de procesamiento tendrá.
- Tolerancia a fallos. El procesamiento de datos y aplicaciones está protegido contra fallas de hardware. Si un nodo deja de funcionar, los trabajos se redirigen automáticamente a otros nodos para asegurarse de que la computación distribuida no falle. Múltiples copias de todos los datos se almacenan automáticamente.
- Flexibilidad. A diferencia de las bases de datos relacionales tradicionales, no es necesario preprocesar los datos antes de almacenarlos. Puede almacenar tantos datos como desee y decidir cómo usarlos más tarde. Eso incluye datos no estructurados como texto, imágenes y videos.

- Bajo costo. El marco de código abierto es gratuito y utiliza hardware básico para almacenar grandes cantidades de datos.
- Escalabilidad. Puede hacer crecer fácilmente su sistema para manejar más datos simplemente agregando nodos. Se requiere poca administración. Dentro de los límites de potencia y costo, se puede incorporar esencialmente cualquier número de nodos dentro de una sola supercomputadora.

HDFS: Name Node



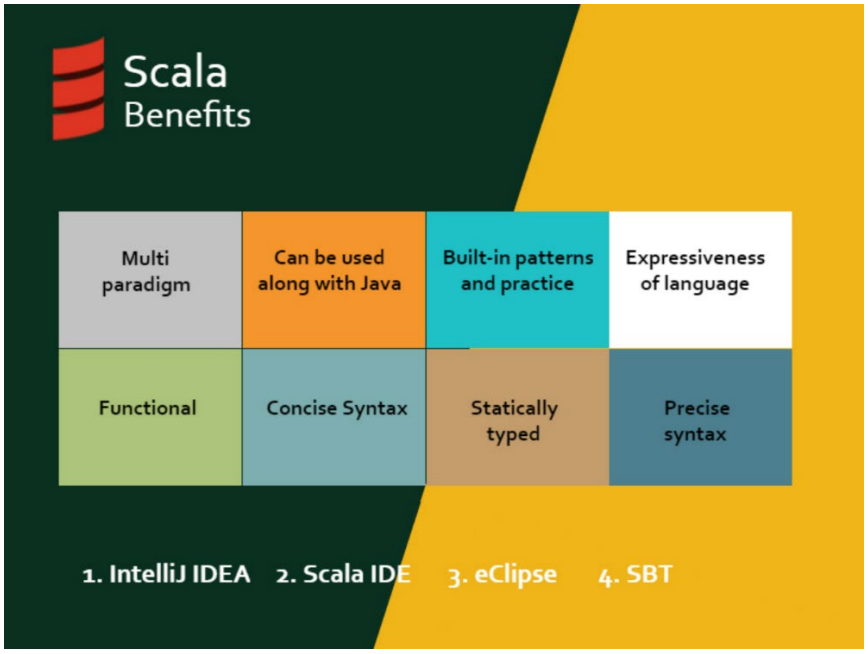
INTRODUCCIÓN – SPARK



Apache Spark es un motor de computación unificado y un conjunto de bibliotecas para el procesamiento de datos en paralelo en clústeres de computadoras. Al momento de escribir este manual, Spark es el motor de código abierto más activamente desarrollado para esta tarea; convirtiéndolo en la herramienta de facto para cualquier desarrollador o científico de datos interesado en Big Data. Spark admite múltiples lenguajes de programación ampliamente utilizados (Python, Java, Scala y R), incluye bibliotecas para diversas tareas que van desde SQL hasta transmisión y aprendizaje automático, y se ejecuta en cualquier lugar, desde una computadora portátil hasta un clúster de miles de servidores. Esto lo convierte en un sistema fácil para comenzar y escalar hasta el procesamiento de Big Data a una escala increíblemente grande.

Spark proporciona API sencillas para trabajar con una gran cantidad de datos, mientras que los usuarios finales apenas necesitan saber sobre la gestión de tareas y recursos en todas las máquinas, todo esto y mas hace a Spark una buena eleccion.

INTRODUCCIÓN – SCALA



Scala es un lenguaje de programación multiparadigma, de alto nivel y de propósito general. Es un lenguaje de programación puro orientado a objetos que también brinda soporte al enfoque de programación funcional.

No existe el concepto de datos primitivos ya que todo es un objeto en Scala. Está diseñado para expresar los patrones generales de programación de una manera refinada, sucinta y segura. Los programas de Scala se pueden convertir a bytecodes y se pueden ejecutar en la JVM (Java Virtual Machine).

Scala significa lenguaje escalable. También proporciona los tiempos de ejecución de Javascript. Scala está muy influenciado por Java y algunos otros lenguajes de programación como Lisp, Haskell, Pizza, etc.

¿Por qué Scala?

Scala tiene muchas razones para ser popular entre los programadores. Algunas de las razones son:

- *Fácil para comenzar:* Scala es un lenguaje de alto nivel, por lo que está más cerca de otros lenguajes de programación populares como Java, C, C++. Por lo tanto, se vuelve muy fácil aprender Scala para cualquier persona. Para los programadores de Java, Scala es más fácil de aprender.

- *Contiene las mejores funciones:* Scala contiene las funciones de diferentes lenguajes como C, C ++, Java, etc., lo que lo hace más útil, escalable y productivo.

- *Estrecha integración con Java:* El código fuente de Scala está diseñado de tal manera que su compilador puede interpretar las clases de Java. Además, su compilador puede utilizar los marcos, las bibliotecas Java y las herramientas, etc. Después de la compilación, los programas Scala pueden ejecutarse en JVM.

- *Desarrollo de aplicaciones de escritorio y basadas en la web:* para las aplicaciones web, proporciona soporte mediante la compilación en JavaScript. De manera similar, para las aplicaciones de escritorio, se puede compilar en el código de bytes JVM.

- *Utilizado por grandes empresas:* la mayoría de las empresas populares como Apple, Twitter, Walmart, Google, etc. mueven la mayoría de los códigos a Scala desde otros idiomas. La razón es que es altamente escalable y se puede usar en operaciones de back-end.

HADOOP

Apache Hadoop es el framework más importante para trabajar con Big Data. La mayor fortaleza de Hadoop es la escalabilidad. Se actualiza de trabajar en un solo nodo a miles de nodos sin ningún problema de manera transparente.

Los diferentes dominios de Big Data significan que podemos administrar los datos de videos, medios de texto, datos transaccionales, información de sensores, datos estadísticos, conversaciones de redes sociales, consultas de motores de búsqueda, datos de comercio electrónico, información financiera, datos meteorológicos, actualizaciones de noticias, debates en foros, informes ejecutivos, etc.

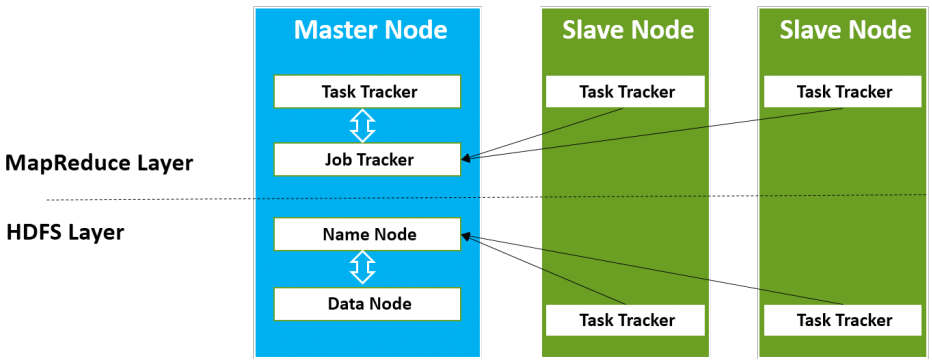
Doug Cutting de Google y los miembros de su equipo desarrollaron un proyecto de código abierto conocido como HADOOP que le permite manejar una gran cantidad de datos. Hadoop ejecuta las aplicaciones sobre la base de MapReduce, donde los datos se procesan en paralelo y realizan el análisis estadístico completo sobre una gran cantidad de datos.

Es un framework que se basa en la programación en Java. Está destinado a funcionar desde un solo servidor hasta miles de máquinas, cada una de las cuales ofrece computación y almacenamiento local. Admite la gran colección de conjuntos de datos en un entorno informático distribuido.

El framework basado en la biblioteca de software Apache Hadoop que otorga permisos para distribuir una gran cantidad de procesamiento de conjuntos de datos en grupos de computadoras utilizando modelos de programación sencillos.

La arquitectura de alto nivel de Hadoop

Arquitectura Hadoop basada en los dos componentes principales, a saber, MapReduce y HDFS



El módulo Apache Hadoop

- Hadoop Common: incluye las utilidades comunes que admiten los otros módulos de Hadoop

- HDFS: el sistema de archivos distribuidos de Hadoop proporciona acceso sin restricciones y de alta velocidad a la aplicación de datos.

- Hadoop YARN: esta tecnología se utiliza básicamente para la programación de trabajos y la gestión eficiente de los recursos del clúster.

- MapReduce: Esta es una metodología altamente eficiente para el procesamiento paralelo de grandes volúmenes de datos.

  Hay otros proyectos incluidos en el módulo Hadoop que son menos utilizados:

- Apache Ambari: Es una herramienta para la gestión, monitorización y aprovisionamiento de los clústeres de Hadoop. Apache Ambari es compatible con los programas HDFS y MapReduce. Los aspectos más destacados de Ambari son:

  La gestión del marco Hadoop es altamente eficiente, segura y consistente.

  Gestión de operaciones de clúster con una interfaz de usuario web intuitiva y una API robusta.

  La instalación y configuración del clúster de Hadoop se simplifica de manera efectiva.

  Se utiliza para admitir la automatización, la configuración inteligente y las recomendaciones.

  La configuración avanzada de seguridad del clúster viene adicional con este kit de herramientas.

  Todo el clúster se puede controlar utilizando métricas, mapas de calor, análisis y solución de problemas.

  Los mayores niveles de personalización y extensión hacen que esto sea más valioso.

- Cassandra: Es un sistema distribuido para manejar una cantidad extremadamente grande de datos que se almacenan en varios servidores básicos. El sistema de administración de bases de datos (DBMS) tiene una alta disponibilidad sin ningún punto único de falla.

- HBase: es un sistema de administración de bases de datos distribuidas no relacionales que funciona de manera eficiente en conjuntos de datos dispersos y es altamente escalable.
- Apache Spark: este es un motor de cómputo de Big Data altamente ágil, escalable y seguro, versátil y suficiente trabajo en una amplia variedad de aplicaciones como procesamiento en tiempo real, aprendizaje automático, ETL, etc.
- Hive: es una herramienta de almacenamiento de datos que se utiliza básicamente para analizar, consultar y resumir los conceptos de datos analizados sobre el marco Hadoop.
- Pig: Pig es un framework de alto nivel que nos garantiza trabajar en coordinación con Apache Spark o MapReduce para analizar los datos. El lenguaje utilizado para codificar los marcos se conoce como Pig Latin.
- Sqoop: este marco se utiliza para transferir los datos a Hadoop desde bases de datos relacionales. Esta aplicación se basa en una interfaz de línea de comandos.
- Oozie: este es un sistema de programación para la gestión del flujo de trabajo, ejecutando rutas de flujo de trabajo para completar con éxito la tarea en un Hadoop.
- Zookeeper: servicio centralizado de código abierto que se utiliza para proporcionar coordinación entre aplicaciones distribuidas de Hadoop. Ofrece el servicio de registro y sincronización a un alto nivel.
- Hadoop Mapreduce (capa de procesamiento/computación): MapReduce es un modelo de programación paralelo que se utiliza principalmente para escribir una gran cantidad de aplicaciones de distribución de datos diseñadas por Google para el procesamiento eficiente de grandes cantidades de conjuntos de datos, en un gran grupo de clústeres.
- Hadoop HDFS (capa de almacenamiento): el sistema de archivos distribuidos de Hadoop o HDFS se basa en el sistema de archivos de Google (GFS), que proporciona un sistema de archivos distribuido que está especialmente diseñado para ejecutarse en hardware básico. Reduce las fallas o errores y ayuda a incorporar hardware de bajo costo. Brinda acceso de rendimiento de procesamiento de alto nivel a los datos de la aplicación y es adecuado para aplicaciones con grandes conjuntos de datos.

- Hadoop YARN: Hadoop YARN es un marco utilizado para la programación de trabajos y la gestión de recursos de clúster.
- Hadoop Common: incluye bibliotecas y utilidades de Java que proporcionan los archivos Java que son esenciales para iniciar Hadoop.
- Rastreador de tareas: es un nodo que se utiliza para aceptar tareas como la reproducción aleatoria y el rastreador de trabajos de formulario Mapreduce.
- Rastreador de trabajos: es un proveedor de servicios que ejecuta trabajos de Mapreduce en un clúster.
- Nodo de nombre: es un nodo donde Hadoop almacena toda la información de ubicación de archivos (ubicación de datos almacenados) en el sistema de archivos distribuido de Hadoop.
- Nodo de datos: los datos se almacenan en el sistema de archivos distribuido de Hadoop.

¿Cómo funciona Hadoop?

Hadoop ayuda a ejecutar una gran cantidad de procesamiento en el que el usuario puede conectar varias computadoras a una sola CPU, como un único sistema funcional distribuido y asi tener un conjunto particular de máquinas agrupadas que leeran los datos en paralelo y asi obtener la salida deseada.

Hadoop ejecuta código en un grupo de computadoras y realiza las siguientes tareas:

- Los datos se dividen inicialmente en archivos y directorios. Los archivos se dividen en bloques de tamaño consistente que van desde 128M y 64M.
- Luego, los archivos se distribuyen en varios nodos de clúster para un mayor procesamiento de los datos.
- El Job Tracker inicia sus tareas de programación en nodos individuales.
- Una vez que todos los nodos han terminado con la programación, la salida es retornada.

Los desafíos que enfrentan los datos a escala y el alcance de Hadoop

Los Big Data se clasifican en:

- Estructurado: que almacena los datos en filas y columnas como conjuntos de datos relacionales

- No estructurado: aquí los datos no se pueden almacenar en filas y columnas como videos, imágenes, etc.
- Semiestructurado: los datos en formato XML son legibles por máquinas y humanos

Existe una metodología estandarizada que Big Data sigue, destacando la metodología de uso de ETL.

ETL: significa Extraer, Transformar y Cargar.

Extraer: obtener los datos de múltiples fuentes

Transformar: convertir los datos existentes para que se ajusten a las necesidades analíticas

Cargar: los sistemas correctos para obtener valor en ellos.

Comparación con tecnologías de bases de datos existentes

La mayoría de los sistemas de administración de bases de datos no están a la altura para operar a niveles tan elevados de exigencias de Big data debido a la pura ineficiencia técnica. Cuando los datos están totalmente desestructurados, el volumen de datos es enorme, donde los resultados son de alta velocidad, finalmente, la única plataforma que puede enfrentar el desafío de manera efectiva es Apache Hadoop.

Hadoop debe principalmente su éxito a un marco de procesamiento llamado MapReduce que es fundamental para su existencia. La tecnología MapReduce brinda la oportunidad a todos los programadores de contribuir con su parte donde se dividen grandes conjuntos de datos y se procesan de forma independiente en paralelo. Estos codificadores no necesitan conocer la computación de alto rendimiento y pueden trabajar de manera eficiente sin preocuparse por las complejidades dentro del clúster, el monitoreo de tareas, la administración de fallas de nodos, etc.

Hadoop también contribuye con su otra plataforma, conocida como Hadoop Distributed File System (HDFS). La principal fortaleza de HDFS es su capacidad para escalar rápidamente y funcionar sin problemas, independientemente de cualquier falla en los nodos. HDFS, en esencia, divide archivos grandes en bloques o unidades más pequeñas que van desde 64 a 128 MB y luego se copian en un par de nodos del clúster. A partir de esto, HDFS garantiza que no se detenga el trabajo, incluso cuando algunos nodos quedan fuera de servicio. HDFS posee API para garantizar que el programa MapReduce se utilice para leer y escribir datos (contenidos) simultáneamente a altas velocidades. Cuando existe la necesidad de acelerar el rendimiento, y luego agregar nodos adicionales en paralelo al clúster y la mayor demanda se puede satisfacer de inmediato.

Ventajas de Hadoop

Da acceso al usuario para escribir y probar rápidamente los sistemas distribuidos y luego distribuye automáticamente los datos y funciona entre las máquinas y, a su vez, utiliza el paralelismo primario de los núcleos de la CPU.

La biblioteca Hadoop está desarrollada para encontrar/buscar y manejar las fallas en la capa de aplicación.

Los servidores se pueden agregar o quitar del clúster dinámicamente en cualquier momento.
Es de código abierto basado en aplicaciones Java y, por lo tanto, compatible con todas las plataformas.

Funciones y características de Hadoop

Apache Hadoop es la herramienta de big data más popular y poderosa, que proporciona la capa de almacenamiento mas confiable del mundo: HDFS (Sistema de archivos distribuidos de Hadoop), un motor de procesamiento por lotes llamado MapReduce y una capa de administración de recursos como YARN.

Código abierto: Apache Hadoop es un proyecto de código abierto. Significa que su código se puede modificar de acuerdo con los requisitos comerciales.

Procesamiento distribuido: el almacenamiento de datos se mantiene de manera distribuida en HDFS en todo el clúster, los datos se procesan en paralelo en el clúster de nodos.

Tolerancia a fallas: de manera predeterminada, las tres réplicas de cada bloque se almacenan en el clúster en Hadoop y solo se modifican cuando es necesario. La tolerancia a fallas de Hadoop se puede examinar en tales casos, cuando cualquier nodo deja de funcionar, los datos en ese nodo se pueden recuperar fácilmente de otros nodos. El marco recupera automáticamente las fallas de un nodo o tarea en particular.

Confiabilidad: debido a la replicación de datos en el clúster, los datos pueden ser confiables y se almacenan en el clúster de la máquina a pesar de las fallas de la máquina. Incluso si su máquina falla, sus datos también se almacenarán de manera confiable.

Alta disponibilidad: los datos están disponibles y accesibles incluso si se produce una falla de hardware debido a múltiples copias de datos. Si ocurriera algún incidente, como si su máquina o algún hardware fallara, se accederá a los datos desde otra ruta.

Escalabilidad: Hadoop es altamente escalable y, de una manera única, se puede agregar fácilmente hardware a los nodos. También proporciona escalabilidad horizontal, lo que significa que se pueden agregar nuevos nodos en la parte superior sin ningún tiempo de inactividad.

Económico: Hadoop no es muy costoso, ya que se ejecuta en un grupo de hardware básico. No requerimos de ninguna máquina especializada para ello. Hadoop proporciona una gran reducción de costos ya que es muy fácil agregar más nodos en la parte superior aquí. Entonces, si el requisito aumenta, entonces hay un aumento de nodos, sin tiempo de inactividad y sin mucha planificación previa.

Fácil de usar: no es necesario que el cliente se ocupe de la informática distribuida, el marco se encarga de todo. Así que es fácil de usar.

Localidad de datos: Hadoop funciona según el principio de localidad de datos que establece que el movimiento de computación de datos en lugar de datos a computación. Cuando el cliente envía su algoritmo, el algoritmo se mueve a los datos en el clúster en lugar de llevar los datos a la ubicación donde se envía el algoritmo y luego procesarlo.

Suposiciones de Hadoop

Hadoop está escrito teniendo en cuenta una gran cantidad de clústeres de computadoras y se basa en las siguientes suposiciones:

- El hardware puede fallar debido a un mal funcionamiento externo o técnico donde, en su lugar, se puede usar hardware básico.
- El procesamiento se ejecutará en lotes y se hace hincapié en un alto rendimiento en lugar de una baja latencia.
- Las aplicaciones que se ejecutan en HDFS tienen grandes conjuntos de datos. Un archivo típico en HDFS puede tener un tamaño de gigabytes a terabytes.
- Las aplicaciones requieren un modelo de acceso de una sola escritura y varias lecturas.
- La computación en movimiento es más barata en comparación con los datos en movimiento.

Principios de diseño de Hadoop

Los siguientes son los principios de diseño sobre los que trabaja Hadoop:

- El sistema se administrará y reparará a sí mismo apenas ocurra un evento.
- Las tolerancias a fallas se enrutan de forma automática y transparente, se administran en torno a fallas, ejecutan especulativamente tareas redundantes si se detecta que ciertos nodos se ejecutan en una fase más lenta.
- El rendimiento se escala en función de la linealidad.
- Cambio proporcional en términos de capacidad con cambio de recursos (Escalabilidad)
- La localidad de datos se denomina latencia más baja, ancho de banda más bajo.
- Se basa en núcleo simple, modular y extensible (Económico).

APACHE SPARK

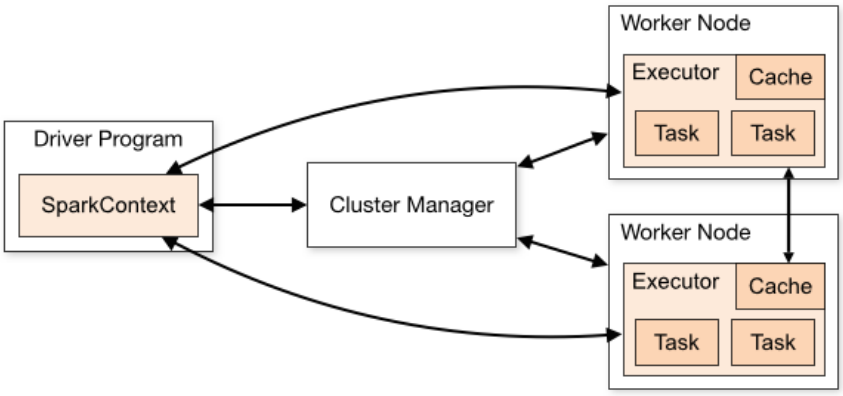
Apache Spark es un framework popular en el campo de Big Data. Viniendo de un background de codigo en Python y SQL, sin entender los mecanismos, confunde bastante al principio. El cambio de ejecutar código en una sola máquina hacia el uso de clústeres, junto con el cambio del tamaño de los datos procesados de MB a GB (incluso TB), nos lleva a comenzar a aprender Spark.

Computación distribuida

Para lograr la computación distribuida se requiere la gestión de recursos y tareas en un grupo de máquinas. La gestión de recursos implica adquirir las máquinas disponibles para la tarea actual, mientras que la gestión de tareas implica coordinar el código y los datos en todo el clúster.

Una aplicación Spark consta de un programa controlador y ejecuta computación paralela en un clúster. Para iniciar una aplicación Spark, el programa controlador que se ejecuta en una máquina host primero iniciará un objeto SparkContext. Este objeto SparkContext se comunicará con un administrador de clústeres, que puede ser el administrador de clústeres independiente de Spark, Mesos, YARN o Kubernetes, para adquirir recursos para esta aplicación. Luego, el objeto SparkContext enviará el código de la aplicación y las tareas a los nodos trabajadores.

Para una aplicación, un nodo worker puede tener varios executors, según la cantidad de CPU disponibles en este nodo worker. Durante el cómputo de una aplicación, cada executor mantiene los datos en la memoria o en el almacenamiento en disco y ejecuta tareas. De esta forma, los executors están aislados entre sí y las tareas de una misma aplicación se ejecutan en paralelo.



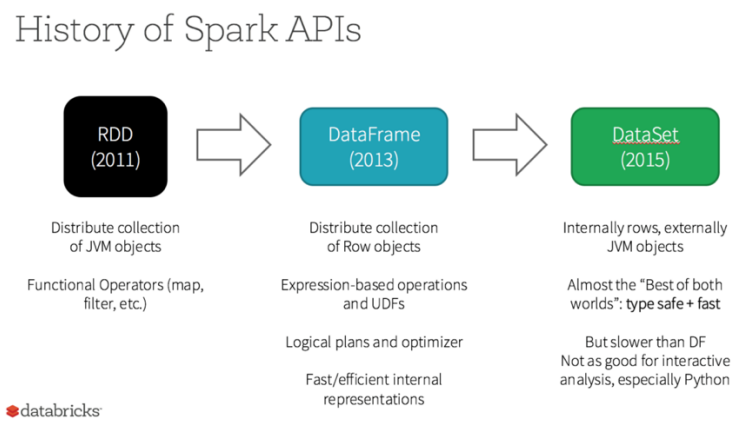
Conjunto de datos distribuido resistente (RDD)

RDD es una abstracción central en Spark, que significa Resilient Distributed Dataset. Permite la partición de grandes datos en datos más pequeños que se adaptan a cada máquina, de modo que el cálculo se puede realizar en paralelo en varias máquinas. Además, los RDD se recuperan automáticamente de las fallas de los nodos para garantizar la resiliencia del almacenamiento.

HDFS (Sistema de archivos distribuidos de Hadoop) es otro concepto importante con el que uno se encuentroa a menudo cuando usa Spark. Aunque tanto RDD como HDFS tienen que ver con el almacenamiento distribuido resiliente, están diseñados para manejar diferentes problemas. La perspectiva resiliente de RDD se refiere al manejo automático de fallas de cálculo. Si bien HDFS se trata de la administración del almacenamiento, está diseñado para manejar fallas de almacenamiento.

API de Spark

Spark proporciona tres API: RDD, DataFrames, Datasets. Estas tres API garantizan una computación de datos resiliente y distribuida, y son adecuadas para diferentes escenarios de aplicación.



RDD es la API fundamental de bajo nivel proporcionada por Spark y admite la manipulación de datos no estructurados o semiestructurados. Usar RDD es como decirle a Spark cómo realizar una tarea, en lugar de simplemente decirle a Spark qué tarea realizar. Como resultado, RDD ofrece la mejor flexibilidad de codificación y control de datos entre las tres API. Sin embargo, al mismo tiempo, sin aprovechar la optimización interna de Spark, un buen maestro de RDD impone requisitos más altos en la experiencia de los programadores.

Además de la API RDD de bajo nivel, Spark también proporciona la API DataFrames de alto nivel. Los DataFrames enfatizan la estructura de datos. Por lo tanto, DataFrames es amigable para los programadores que vienen con experiencia en bases de datos relacionales. Al usar DataFrames, se siente bastante similar a usar Pandas DataFrame o Excel Spreadsheet, pero con Spark manejando la computación del clúster bajo tras bambalinas.

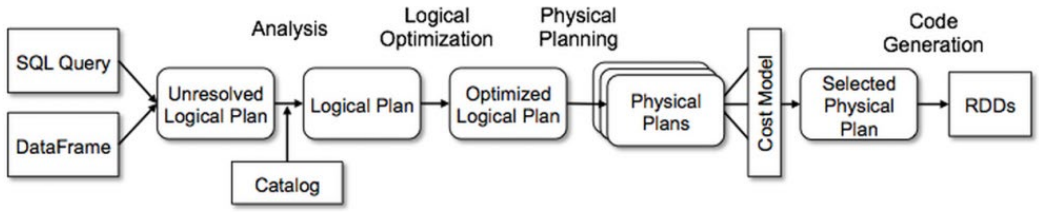
Si decimos que las API de RDD y DataFrames están en los dos lados de una inclinación, con RDD en el lado del control flexible de bajo nivel y DataFrames en el lado de la codificación fácil de alto nivel, entonces la API de conjuntos de datos se encuentra en el medio de los otros dos API. Además de la API de DataFrames, la API de conjuntos de datos impone seguridad de tipos para evitar errores en tiempo de ejecución.

Tanto las API de RDD como las de conjuntos de datos requieren seguridad de tipo y solo son compatibles con Java y Scala. Sin embargo, la API de DataFrame admite lenguajes de escritura dinámica, como Python y R.

Spark SQL

Spark SQL es el módulo de Spark para trabajar con datos estructurados.

Spark SQL funciona con las dos API estructuradas, a saber, conjuntos de datos y dataframes. Aprovechando la información del esquema que solo está disponible en Datasets o DataFrames, el código Spark SQL le dice a Spark qué hacer de manera declarativa, a diferencia de la forma en que le dice a Spark cómo hacerlo cuando se usa la API RDD de bajo nivel. De esta forma, el código escrito en Spark SQL se beneficia del catalizador de Spark, que optimiza el rendimiento. Por lo tanto, usar Spark SQL con las API estructuradas es más fácil de escribir código de alto rendimiento.



Usar Spark SQL con la API de DataFrames es equivalente a ejecutar una consulta SQL en la base de datos relacional. Las funciones SQL de uso común, como filtro, combinación, agregación y función de ventana, también están disponibles en Spark SQL. Spark SQL y la API de DataFrames admiten varios lenguajes de programación, incluidos Python, R, Scala y Java.

Spark SQL, Presto y Hive admiten consultas de datos a gran escala que residen en almacenamiento distribuido mediante la sintaxis SQL, pero se utilizan para diferentes escenarios.

Spark SQL es el módulo central de Spark, mientras que Presto está en el ecosistema de Hadoop. Spark SQL hace hincapié en el cálculo y, por lo general, se usa en ETA y canalización a gran escala. Sin embargo, Presto hace hincapié en la consulta y se utiliza más a menudo para análisis ad-hoc. Tanto Spark SQL como Presto calculan en memoria. Cuando se trata de escasez de memoria, Spark SQL permite que se haga spilling en el disco, mientras que Presto sufrirá problemas de OOM. La tolerancia a fallas también se considera en Spark SQL, pero no en Presto.

Hive es un software de almacenamiento de datos que gestiona datos estructurados a gran escala en el ecosistema Hadoop. Las consultas de Hive se pueden ejecutar a través de Spark o MapReduce. Hive tiene su propio motor SQL llamado HiveQL. Como hemos mencionado anteriormente, Spark SQL es el módulo de Spark para trabajar con datos estructurados. Del mismo modo, Hive es el módulo de Hadoop para trabajar con datos estructurados.

Hemos discutido algunos conceptos básicos de Spark. Aunque la API DataFrame de alto nivel junto con Spark SQL facilita la escritura de código de alto rendimiento, comprender cómo funciona Spark ayuda a mejorar aún más el rendimiento. En la próxima sesion, veremos YARN como ejemplo y analizaremos cómo comprender la administración de tareas y recursos de Spark mediante la interfaz de usuario web de YARN.

SCALA

Comenzando con la programación en Scala

Encontrar un compilador: hay varios IDE en línea, como GeeksforGeeks IDE, Scala Fiddle IDE, etc., que se pueden usar para ejecutar programas Scala sin instalar.

Programación en Scala: dado que Scala es muy similar sintácticamente a otros lenguajes ampliamente utilizados, es más fácil codificar y aprender en Scala. Los programas se pueden escribir en Scala en cualquiera de los editores de texto más utilizados como Notepad++, gedit, etc. o en cualquiera de los editores de texto. Después de escribir el programa, guarde el archivo con la extensión .sc o .scala.

Para Windows y Linux: Antes de instalar Scala en Windows o Linux, debe tener instalado el Kit de desarrollo de Java (JDK) 1.8 o superior en su sistema. Porque Scala siempre se ejecuta en Java 1.8 o superior.

La gente siempre piensa que Scala es una extensión de Java. Pero no es cierto. Es completamente interoperable con Java. Los programas de Scala se convierten en un archivo .class que contiene el código de bytes de Java después de la compilación exitosa y luego se pueden ejecutar en JVM (Java Virtual Machine).

En este sesion, discutiremos cómo ejecutar los programas Scala en IDE en línea.

Ejemplo: Un programa simple para imprimir Hello UNI! utilizando un enfoque orientado a objetos.

```
// Scala program to print Hello, UNI!
// by using object-oriented approach

// creating object
object Geeks {

        // Main method
        def main(args: Array[String])
        {

                // prints Hello, UNI!
                println("Hello, UNI!")
        }
}
```

Caracteristicas de scala

Hay muchas características que lo hacen diferente de otros idiomas.

- Orientado a objetos: cada valor en Scala es un objeto, por lo que es un lenguaje de programación puramente orientado a objetos. El comportamiento y el tipo de objetos están representados por las clases y rasgos en Scala.
- Funcional: también es un lenguaje de programación funcional ya que cada función es un valor y cada valor es un objeto. Proporciona soporte para funciones de alto orden, funciones anidadas, funciones anónimas, etc.
- Estáticamente tipificado: el proceso de verificar y hacer cumplir las restricciones de los tipos se realiza en tiempo de compilación en Scala. A diferencia de otros lenguajes de programación tipificados estáticamente como C++, C, etc., Scala no espera la información de tipo redundante del usuario. En la mayoría de los casos, el usuario no necesita especificar un tipo.
- Extensible: se pueden agregar nuevas construcciones de lenguaje a Scala en forma de bibliotecas. Scala está diseñado para interpolar con JRE (Java Runtime Environment).
- Procesamiento concurrente y sincronizado: Scala permite al usuario escribir los códigos de una manera inmutable que facilita la aplicación del paralelismo (sincronización) y la concurrencia.

- Ejecutar en JVM y puede ejecutar código Java: Java y Scala tienen un entorno de tiempo de ejecución común. Así, el usuario puede pasar fácilmente de Java a Scala. El compilador de Scala compila el programa en un archivo .class, que contiene el código de bytes que JVM puede ejecutar. Todas las clases de Java SDK pueden ser utilizadas por Scala. Con la ayuda de Scala, el usuario puede personalizar las clases de Java.

Ventajas

- Las características complejas de Scala proporcionaron una mejor codificación y eficiencia en el rendimiento.
- Tuplas, macros y funciones son los avances en Scala.
- Incorpora la programación funcional y orientada a objetos que a su vez lo convierten en un lenguaje poderoso.
- Es altamente escalable y, por lo tanto, brinda un mejor soporte para las operaciones de back-end.
- Reduce el riesgo asociado con la seguridad de subprocesos, que es mayor en Java.
- Debido al enfoque funcional, generalmente, un usuario termina con menos líneas de códigos y errores, lo que resulta en una mayor productividad y calidad.
- Debido a la lazy computation, Scala calcula las expresiones solo cuando son necesarias en el programa.
- No hay métodos estáticos ni variables en Scala. Utiliza el objeto singleton (clase con un objeto en el archivo fuente).
- También proporciona el concepto Traits. Los rasgos son la colección de métodos abstractos y no abstractos que se pueden compilar en las interfaces de Java.

Desventajas

- A veces, dos enfoques hacen que la Scala sea difícil de entender.
- Hay un número limitado de desarrolladores de Scala disponibles en comparación con los desarrolladores de Java.
- No tiene optimización recursiva de cola verdadera ya que se ejecuta en JVM.
- Siempre gira en torno al concepto orientado a objetos porque cada función es un valor y cada valor es un objeto en Scala.

Aplicaciones

- Se utiliza sobre todo en el análisis de datos con Spark.
- Se utiliza para desarrollar las aplicaciones web y la API.
- Proporciona la facilidad para desarrollar los frameworks y las bibliotecas.
- Se prefiere usar en operaciones de back-end para mejorar la productividad de los desarrolladores.
- El procesamiento por lotes en paralelo se puede realizar con Scala.