# UNIDAD 1: ANTECEDENTES, DEFINICIONES Y BASES PARA UN CORRECTO ENTENDIMIENTO

## Módulo profesional:

## Análisis de grandes datos (Big Data)

# Índice

# RESUMEN INTRODUCTORIO

En esta unidad vamos a introducirnos de forma concisa y fácil de entender en los conceptos generales de big data, al tiempo que aprendemos una tecnología muy demanda, que es Spark.

Conoceremos cómo usarla para una variedad de tareas analíticas de big data. Cubriremos todos los aspectos que necesitamos comprender para utilizarla de modo productivo.

Con el contenido y la organización del material de esta unidad trataremos de responder a las preguntas y dudas que más frecuentemente se presentan cuando uno se inicia en una tecnología tan diferente a otras existentes.

# INTRODUCCIÓN

La aceleración del proceso de transformación digital, derivada de la implantación generalizada de tecnologías de la información y la comunicación, está modificando el panorama organizacional, especialmente en el sector empresarial privado, que es más dinámico y adaptativo.

Este proceso ha sido especialmente intenso desde principios del milenio por la difusión de internet, y en el último lustro por la democratización que se ha producido en tecnologías complejas, como la inteligencia artificial y, sobre todo, el machine learning, así como el análisis avanzado, con gran énfasis en el big data. En los próximos años se prevé que el ciclo tecnológico se mantendrá igualmente expansivo y más acelerado si cabe.

Las herramientas tecnológicas del pasado se han visto superadas por el impacto del big data y, por este motivo, en la última década se han lanzado al mercado productos que implementan nuevos paradigmas en cuanto a almacenamiento de datos, procesamiento de los mismos y capacidades analíticas.

Los avances en materia de análisis de datos, gracias a las nuevas visiones informáticas, provocan un cambio de modelo en la gestión de los negocios. Este fenómeno es lo que muchos analistas y académicos han calificado como la empresa data driven, o lo que es lo mismo, compañías en las cuales la toma de decisiones se efectúa sobre un conjunto de evidencias en forma de datos, superando una visión tradicional más intuitiva e informal.

Dos claros ejemplos de tecnologías propicias para este cambio de filosofía son Hadoop, con su sistema de almacenamiento HDFS y su enfoque de procesamiento MapReduce, y Spark, con su lenguaje de programación asociado Scala.

Al igual que estas novedosas plataformas tecnológicas persiguen responder a nuevas demandas de las tecnologías de la información, otras necesidades más convencionales siguen estando presentes. Por este motivo, herramientas como las ETL continúan presentes en cualquier ecosistema informático actual.

## CASO INTRODUCTORIO

El Gobierno de España está diseñando una convocatoria de oposiciones que será masiva para poder cubrir las miles de plazas de funcionario que quedarán vacantes en los próximos años por las jubilaciones de la generación denominada de los baby boomers. Dada la situación de crisis, se espera que sean cientos o incluso miles de candidatos los que se presentarán por cada plaza disponible.

Por este motivo, en previsión de que el proceso pueda saturarse, el Gobierno te ha contratado como consultor para que emitas una serie de recomendaciones donde la tecnología podría ayudar a alcanzar varios objetivos. Ellos están convencidos que tanto el big data, como la inteligencia artificial tienen mucho que aportar a este proceso.

El Gobierno te ha dicho está dispuesto a revolucionar el proceso de selección de servidores públicos y que tus propuestas no están limitadas por ningún criterio.

Al finalizar esta unidad serás capaz de entender de qué forma el big data te ayudará en este proceso y podrás identificar las tecnologías necesarias para esto.

# 1. ORIGEN Y CONTEXTUALIZACIÓN DEL ANÁLISIS DE GRANDES DATOS (BIG DATA)

*En primer lugar, has optado por proponer al Gobierno la implementación de un chatbot para automatizar la entrega de información y resolución de consultas más frecuentes a los candidatos opositores.*

*Estamos hablando de cuestiones como la titulación necesaria para optar a cada puesto, las fechas y ubicaciones para la realización de las pruebas, el contenido de los temarios de obligatorio estudio, el salario ofertado para cada puesto, las condiciones laborales a la incorporación, la evolución de la carrera profesional, etc. Este tipo de cuestiones se pueden responder mediante un auto servicio 24x7 sin necesidad de contar con un call center.*

*Piensas que es un buen hito de partida, porque desde el mismo momento en que se anuncie la convocatoria, decenas de miles de candidatos comenzarán a indagar y preguntar. Pero transcurrirán meses hasta que empiecen las primeras fases selectivas de cada oposición, por lo que tendrás tiempo para ir armando otras iniciativas que satisfagan a tu cliente.*

Aunque las definiciones exactas varían, la mayoría de los expertos estarían de acuerdo en que el big data se refiere a informaciones recolectadas que son tan grandes o complejas que las tecnologías tradicionales sufren cuando se intenta almacenar y procesar esa cantidad de datos.

Además, el big data implica una gran variedad de aplicaciones y análisis que se están construyendo alrededor esa información. Sin embargo, la historia de este es mucho más que simplemente empresas que acumulan grandes conjuntos de datos y luego encuentran nuevas formas de explotarlos. De hecho, lo que denominamos como big data es el resultado inevitable de tres tendencias globales:

- **Organizaciones que están capturando, gestionando y generando grandes cantidades de información:** numerosos estudios de mercado han reforzado la idea de que los volúmenes de datos continúan creciendo sin parar. Además, muchas organizaciones están almacenando tres o más años de datos históricos, para fines regulatorios u operativos.

- **Una gran cantidad de datos no son estructurados:** estos mismos estudios indican también que el 80% de los datos recolectados en la actualidad no llegan en un formato tradicional y estructurado, es decir, en el formato que se ajusta a filas y columnas y relacional, utilizado durante décadas en los sistemas gestores de bases de datos relacionales. Ahora muchos datos están compuestos por imágenes, audios, tweets y mensajes de texto, de los cuales, las empresas también quieren sacar provecho y analizarlos de forma completa, pero no se pueden analizar de la forma tradicional.

- **Una variedad de nuevas aplicaciones está siendo desarrollada para explotar esta gran cantidad de información y nuevos tipos de datos:** muchas de las herramientas y tecnologías que fueron diseñadas para trabajar con volúmenes de información relativamente grandes no evolucionaron mucho en los últimos veinte años. Esta rigidez significa que simplemente no pueden mantenerse al día con el big data. En respuesta, las empresas están construyendo o comprando nuevas clases de aplicaciones analíticas. Estas soluciones están transformando la forma en que las empresas están haciendo negocios, y construyendo un software, en gran medida, mediante las plataformas de big data de nueva generación, como MapReduce y Hadoop.

## 1.1 ¿Qué se puede hacer con el big data?

El big data tiene el potencial de revolucionar la forma de hacer negocios. Puede proporcionar nuevos conocimientos sobre todo tipo de aspectos acerca de cualquier empresa, lo que incluye lo siguiente:

- ¿Cómo los clientes localizan a una compañía? ¿Por qué se deciden por ella y cómo interactúan con nosotros?
- La forma en que se entregan productos y servicios al mercado.
- La posición de nuestra organización frente a los competidores.
- Las estrategias que podemos implementar para aumentar la rentabilidad.

Algunos casos específicos para industrias y empresas que se pueden destacar son los siguientes:

- **Servicios Financieros:**

    - Obtener un conocimiento más profundo sobre los clientes.
    - Atención a actividades fraudulentas.
    - Ofrecimiento de productos y servicios nuevos e innovadores.
    - Tomar mejores y más rápidas decisiones comerciales.

- **Telecomunicaciones:**

    - Proporcionar la más alta calidad de servicio.
    - Identificar y corregir rápidamente las anomalías de la red.
    - Tomar decisiones informadas sobre inversiones de capital.
    - Ofrecer paquetes altamente personalizados para retener más clientes.

- **Retail:**

    - Ofrecer recomendaciones más inteligentes de venta cruzada (cross selling) y up selling.
    - Obtener una mejor visión de las tendencias generales de compra.
    - Establecer precios y descuentos óptimos.
    - Monitorización de redes sociales para detectar satisfacción o descontento entre los clientes.

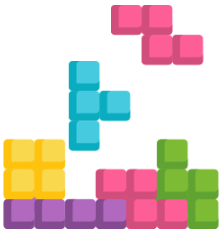
- **Energía:**

    - Detectar patrones significativos de flujos de los sensores sin procesar en fecha.
    - Identificar nuevas fuentes de energía prometedoras más rápidamente y con precisión.
    - Ajustar los niveles de inventario con mayor precisión.
    - Uso de análisis predictivos para detectar posibles fallos en los equipos y las redes antes de que causen algún daño.

- **Salud:**

    - Agilizar y acelerar el proceso para nuevos descubrimientos en cuestiones de investigación médica o farmacológica.
    - Garantizar el cumplimiento de regulaciones como HIPAA.
    - Monitorización de una gran cantidad de dispositivos médicos para detectar lo antes posible incipientes crisis o epidemias.
    - Detectar posibles interacciones de medicamentos antes que puedan provocar efectos adversos al paciente.

Lo más interesante de estos escenarios es que se benefician del potencial que tiene el poder obtener y manipular datos en tiempo real. En este tipo de circunstancias es donde Spark proporciona una infraestructura con el rendimiento, escalabilidad y seguridad necesarios para sacar el máximo beneficio del big data.



**EJEMPLO PRÁCTICO**

El director de recursos humanos de la empresa donde desempeñas el cargo de director de informática ha participado en un seminario de actualización en una escuela de negocios en la cual le han explicado el potencial que tiene el big data aplicado a los recursos humanos.

Ha regresado a la empresa con mucha ilusión, pero no tiene claro por dónde empezar. Por este motivo, aprovechando que habéis coincidido en una reunión, te pide que le ayudes a pensar 3 o 4 aplicaciones para su departamento.

¿Serías capaz de preparar un breve borrador con ideas atractivas para tu compañero?

**SOLUCIÓN:**

La realidad es que eres el responsable de informática de la empresa y estás más experimentado en cuestiones que tienen que ver con el software para la realización de las nóminas de la compañía, los sistemas de fichaje del personal o la compra de equipos informáticos para los trabajadores.

A pesar de que no tienes experiencia en big data, quieres quedar bien con tu compañero y echarle una mano, por lo que decides dedicar una tarde a indagar por internet sobre qué están haciendo algunas empresas de las más importantes del mundo y ver si hay alguna buena idea que se pueda copiar y adaptar a vuestra realidad.

Entre los casos reales que has descubierto, has decidido contarle al director de recursos humanos lo siguiente para que encuentre la inspiración que necesita:

- En materia de desgaste del empleado, has sabido que la empresa Experian consideraba que el desgaste o cansancio de los empleados era muy perjudicial para su actividad. Achacaban a ello que la rotación no deseada fuese de entre un 3 y un 4%. Decidieron construir un modelo predictivo basado en el big data con 200 variables, como el tamaño y otros parámetros organizacionales de los equipos y departamentos. El modelo también se alimentaba con datos sobre la valoración que recibían los supervisores y gerentes, así como el tiempo perdido por cada trabajador durante su traslado diario a la oficina, etc.

  Durante la construcción del modelo predictivo se detectaron correlaciones entre el tamaño del equipo y la insatisfacción laboral, siendo los equipos de tamaño superior a diez o doce personas los que generaban peores sensaciones.

El equipo de análisis también identificó los desencadenantes del riesgo de fuga en proporción a la distancia del domicilio personal a la sede corporativa. Una vez validado el algoritmo de predicción, se desplegó en varias áreas geográficas. Experian estimó un impacto financiero de unos 8.000.000 a 10.000.000 de dólares durante el primer año y medio de optimización de la estructura organizacional en base a los resultados de la analítica.

- En lo que tiene que ver con enfermedades laborales, has conocido que el gigante energético E.ON valoró que las bajas por enfermedad estaban por encima de la ratio que consideraban comparable en su sector, algo que intuyes que también está pasando en vuestra empresa. El equipo de analistas identificó 55 hipótesis que podrían explicar el alto absentismo y que permitirían construir un modelo de análisis de big data. De esas hipótesis, con los datos disponibles, pudo verificar 21 y decidió actuar sobre 11.

  Una de ellas sobre las cuales E.ON actuó fue lo que consideraron que era una mala gestión de los periodos vacaciones, tanto por la renuncia a las mismas de algunos empleados como por la existencia de largos periodos de tiempo sin descanso.

  El resultado de los análisis se compartió con los gerentes para que modificasen su comportamiento a la hora de aprobar las vacaciones de su equipo.

- Finalmente, la tercera idea que vas a compartir con tu colega de RRHH tiene que ver con la deserción de trabajadores, que es otro tema que preocupa y mucho a los directores de recursos humanos. The Wall Street publicó en 2015 un artículo titulado: "The Algorithm That Tells the Boss Who Might Quit", que se podría traducir como "El algoritmo que advierte al jefe quien podría renunciar", que exploraba cómo Credit Suisse fue capaz de predecir quienes podría abandonar la compañía. Fue uno de los primeros proyectos de análisis de la rotación no deseada de empleados.

  Los analistas de recursos humanos de Credit Suisse no sólo fueron capaces de predecir quién podría abandonar el banco, sino que también identificaban los motivos por los cuales estos trabajadores lo hacían. Esta información fue proporcionada anónimamente a los gerentes para que pudieran reducir los factores de riesgo de rotación y mantener a su equipo mejor cohesionado.

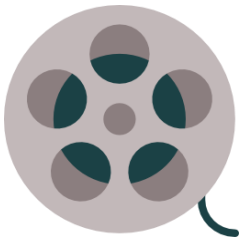
  Además, como derivada del análisis, se entrenó a los gerentes para retener a los empleados de alto rendimiento con riesgo de abandono. Credit Suisse estimó en setenta millones de dólares al año el ROI de este proyecto.

## 1.2 ¿Qué está impulsando el big data?

Así como no hay una definición universal para big data, tampoco hay una causa específica y única que provoque esta tasa de adopción tan alta que ha tenido. En cambio, muchas tendencias recientes vienen contribuyendo para conformar el momento actual del big data. Cinco de las más notables son:

- Surgimiento de nuevas fuentes de datos.
- Caminamos hacia una gran cantidad de datos.
- Categorías de datos más amplias.
- Comercialización de hardware y software innovador y muy competitivo en costes.
- Crecimiento de la inteligencia artificial.

Antes de empezar a detallarlas, es importante destacar que ninguno de estos avances sería posible sin las nuevas tecnologías para capturar, trabajar y analizar una gran cantidad de datos. Empresas como Google, Facebook, Yahoo y Amazon, así como entidades sin fines de lucro como Apache Software Foundation, tienen un papel importante potenciando estas tecnologías.



**VIDEO DE INTERÉS**

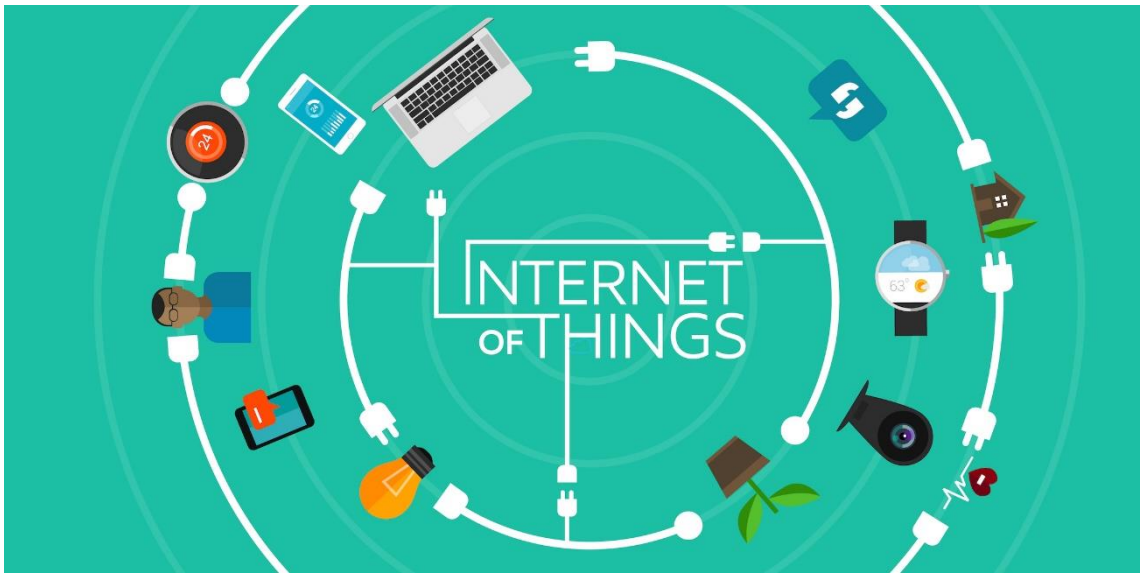
A continuación podrás ver un vídeo en el que se explica de forma breve una introducción a las claves del big data.

https://www.youtube.com/watch?v=w4vsFKMO7XA

### *1.2.1 Nuevas fuentes de datos*

Actualmente existen más generadores de información que nunca. Estas fuentes incluyen dispositivos, como teléfonos móviles, tabletas, sensores, equipos médicos y otras tecnologías, que reúnen gran cantidad de información. Muchas de estas nuevas fuentes son datos comúnmente etiquetados perteneciente al internet de las cosas (IoT) y su impacto está dejando pequeño todo lo que les precedió.

Por ejemplo, el número de "cosas" (dispositivos) conectadas ya está en las decenas de miles de millones y sigue creciendo. Algunas fuentes incluso afirman que habrá más de 20 mil millones de elementos conectados en este año de 2020.




IoT
Fuente: https://secmotic.com/wp-content/uploads/IoT.jpg

Mientras tanto, las aplicaciones empresariales convencionales están cambiando también: el comercio electrónico, la gestión de riesgos financieros y las cada vez más potentes soluciones científicas (como farmacéuticas, meteorológicas y de simulación por nombrar algunas) están contribuyendo al crecimiento general del big data.

**VIDEO DE INTERÉS**

En el siguiente enlace de vídeo podrás aprender más sobre el internet de las cosas con una explicación sobre ello.

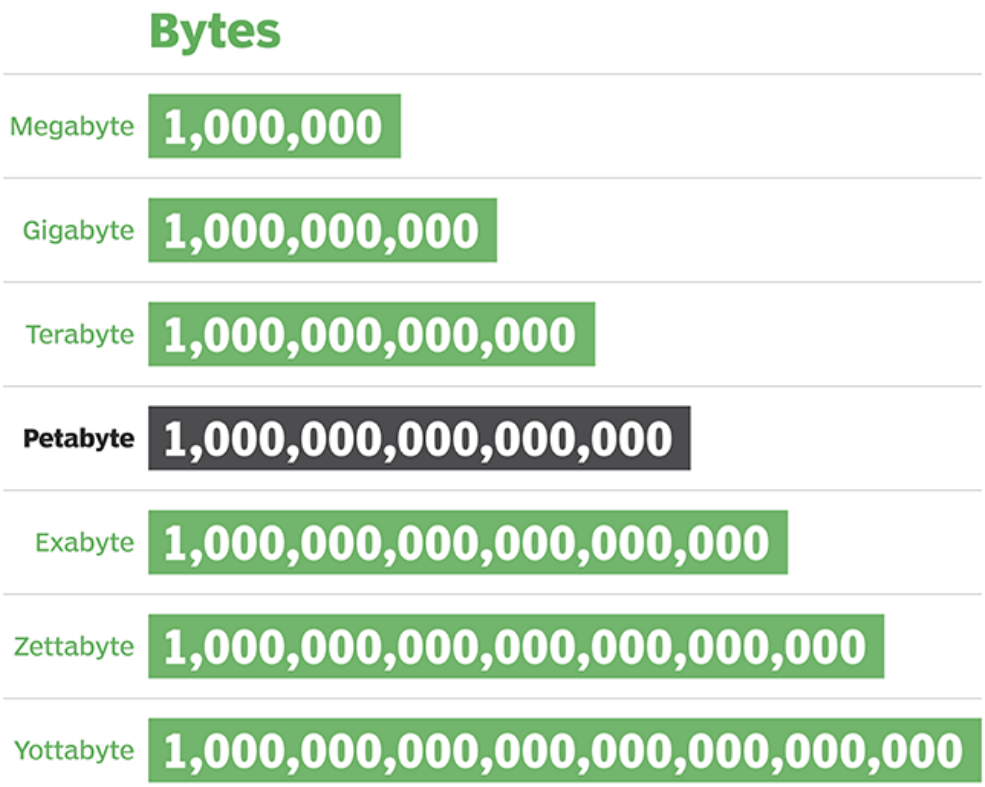
https://www.youtube.com/watch?v=uY-6PcO96Bw

### *1.2.2 Gran cantidad de datos*

Como se puede suponer por su nombre, big data también significa que ahora se están capturando, analizando y administrando volúmenes de datos mucho más grandes.

Para demostrar cuánto más voluminosos pueden ser los datos, consideremos esto: durante un histórico que abarca más de 40 años, los servidores de bases de datos SQL han tenido tradicionalmente tamaños medidos en gigabytes de información y alcanzar ese hito tomó mucho tiempo.

En los últimos 30 años, los almacenes de datos y análisis empresarial expandieron estos volúmenes a terabytes. Pero en los últimos diez años, los sistemas de archivos distribuidos que almacenan big data ahora albergan petabytes de información, y la Corporación Internacional de Datos (IDC) pronostica que generaremos 175 zettabytes (175 mil millones de terabytes) para 2025. Esto no es una sorpresa cuando somos conocedores de que ciertos automóviles híbridos pueden generar ya alrededor de 25 gigabytes de datos por hora.




Comparación entre las unidades de medida de datos
Fuente: techtarget.com

### *1.2.3 Categorías de datos más amplias*

¿Cómo los datos de una empresa de repente se disparan desde gigabytes, a cientos de terabytes y luego a petabytes? Una de las realidades que ha provocado esto es cuando las compañías comenzaron a trabajar con clases de información completamente nuevas. Además, mientras parte de esta nueva información es de naturaleza relacional, la mayoría ya no lo es.

En el pasado, la práctica totalidad de las bases de datos relacionales solo tenían registros completos, es decir, transacciones finalizadas. En el mundo del big data, la mayoría de los datos son sub-transaccionales. Es decir, datos que definimos como información que se recopila mientras la transacción aún no se ha completado, pero que también forman parte de la imagen de la realidad empresarial que se pretende modelizar en base a los datos. Estos son solo algunos ejemplos de datos sub-transaccionales:

- Clics en un sitio web.
- Uso y manipulación del carrito de compras (en comercio electrónico).
- Tweets.
- Mensajes de texto.

Hasta la fecha se han diseñado bases de datos relacionales y herramientas analíticas asociadas para interactuar con información estructurada. Mucha de la información que conforma el big data actual no está estructurada ni tan siquiera semiestructurada, por lo que se requiere de nuevas tecnologías para trabajar con ello. Entre este tipo de información podemos destacar:

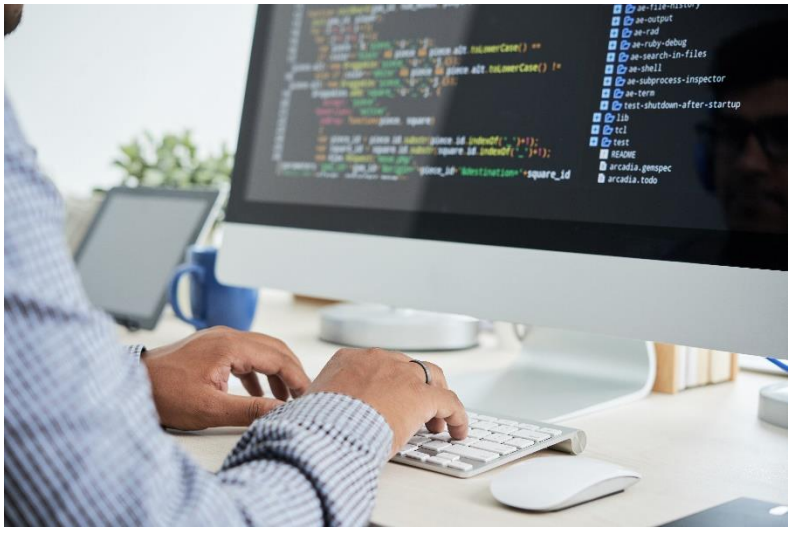
- Fotografías.
- Vídeos.
- Audios.
- Archivos y documentos XML y JSON.

Las aplicaciones empresariales de hoy en día suelen transmitir datos utilizando documentos codificados en lenguaje de marcado extensible (XML) o notación de objetos JavaScript (JSON). A pesar de que forman parte del núcleo del software actual, han demostrado ser demasiado exigentes para generaciones antiguas de herramientas analíticas, que tienen dificultades para capturar estas grandes cantidades de información. Esto es culpa de su tamaño masivo y su naturaleza semiestructurada.

Envío masivo de tweets
Fuente: https://www.freepik.es/vector-gratis/ilustracion-concepto-tormenta-tweets_6183011.htm#page=1&query=tweets&position=0

### *1.2.4 Comercialización de software y hardware económico*

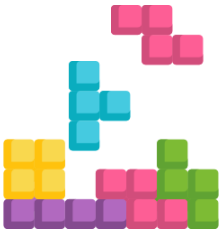
La pieza final del rompecabezas del big data es el hardware de bajo coste y los entornos de software que han transformado la tecnología, particularmente en los últimos diez años. Capturar y explotar una gran cantidad de datos sería mucho más difícil y costoso sin las contribuciones de estos avances que han demostrado ser muy rentables.



Datos
Fuente: https://www.freepik.es/foto-gratis/coding-man_5633683.htm#page=1&query=software&position=1

### *1.2.5 Inteligencia artificial*

Las aplicaciones de inteligencia artificial, que incluyen machine learning y deep learning, necesitan una enorme cantidad de datos para entrenar a los modelos de inteligencia artificial y proporcionar resultados precisos y efectivos. Estos resultados dependen también de la calidad, diversidad y dinamismo de los datos utilizados.



**EJEMPLO PRÁCTICO**

Una pequeña tienda, pero muy conocida, especializada en la venta de vinilos, ha pensado que el mejor modo para competir sería pasarse a la venta por internet. Tienen una especialización muy alta y son reconocidos como los mejores en su género, por lo que confían en ello.

El problema que ven los propietarios de esta tienda es que gran parte de su valor añadido está en ser capaces de recomendar a los clientes nuevos LPs. Los dueños, aparte de vendedores, son grandes melómanos y realizar ese tipo de recomendaciones les resulta muy natural, por lo que tienen miedo de que la tienda online sea demasiado fría y venda poco por esa falta de recomendación.

Cómo tú también eres aficionado a la música y cliente de esta tienda y ellos conocen que eres experto en datos, te piden consejo para lanzarse a internet. ¿Qué les dirías?

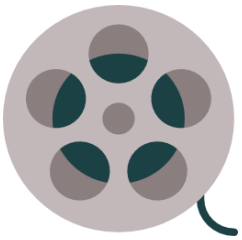
**SOLUCIÓN:**

Lo primero que les dices es que llevan razón. Que su gran conocimiento de los discos en vinilo clásicos es lo que les aporta su mayor valor añadido y que eso no se puede perder en internet.

Lo segundo, es que les haces ver que no será sencillo. Les informas que los sistemas de recomendación se suelen basar en técnicas de machine learning, que es una rama de la inteligencia artificial y que requiere de gran cantidad de datos para entrenar al sistema. Por tanto, como todavía no han creado su sistema informático, hay dos opciones:

- Una sería simplemente lanzar la tienda y esperar meses, tal vez años, a que se empiecen a generar volúmenes suficientes de datos sobre los cuales poder en el futuro aplicar inteligencia artificial.

- Buscar en otros lugares datos que permitan alimentar al sistema de machine learning, por ejemplo, utilizar bases de datos de catálogos musicales que están disponibles en internet.



**VIDEO DE INTERÉS**

La visión de Amazon integradora entre inteligencia artificial, machine learning y big data es algo que podrás ver en este vídeo.

https://www.youtube.com/watch?v=ijxySOpkGWk

## 1.3 Los desafíos del big data

Pensar en big data como solamente muchos más datos empresariales es tentador, pero es un grave error. Es tentador porque, si fuese así, los retos y dificultades del manejo de la información en la actualidad serían mucho más sencillos. Pero es un error porque, si de verdad creemos eso, nos estaremos auto engañando.

Primero, big data es notablemente más grande, a menudo en varios órdenes de magnitud. En segundo lugar, el big data se genera comúnmente fuera de las aplicaciones empresariales tradicionales. Finalmente, el big data a menudo se compone de elementos no estructurados o tipos de información semiestructurada que continuamente se producen y deben ser almacenadas en enormes cantidades.

Como es común en cualquier nuevo movimiento de cambio tecnológico, la primera generación del big data llegó también con algunas barreras y limitaciones:

- **Aumento de información:** los datos no estructurados constituyen la gran mayoría de lo que se está capturando hoy. Estos volúmenes masivos de datos asustaron hasta a las empresas de TI mejor preparadas.

- **Poder de procesamiento:** el enfoque habitual de un ordenador o servidor único, costoso y potente para procesar información simplemente no es escalable cuando se enfrenta a la gigantesca cantidad de datos actual. El enfoque más inteligente es crear una red interconectada de servidores basados en hardware barato y de uso comercial y después distribuir todo en pequeñas tareas de trabajo en cada servidor.

- **Almacenamiento físico:** capturar y gestionar toda esta información consume enormes recursos, superando las expectativas y capacidades presupuestarias de muchas organizaciones.

- **Problemas con los datos:** la falta de movilidad de datos, los formatos propietarios y algunos obstáculos de interoperabilidad hacen que el trabajo con big data sea todavía más complicado.

- **Costes:** los procesos de extracción, transformación y carga (ETL) para big data eran caros y llevaba mucho tiempo el ejecutarlos, particularmente, por la ausencia de software especializado y bien diseñado.

En los primeros tiempos del big data las complicaciones resultaron ser un obstáculo insuperable para muchas organizaciones. Construir aplicaciones usando las mismas herramientas y tecnologías que se habían utilizado durante las décadas previas fue difícil, costoso y arrojó resultados demasiado lentamente y con escaso valor.

Impulsadas por las deficiencias de usar enfoques heredados para trabajar con big data, las nuevas técnicas especializadas de computación y almacenamiento comenzaron a ganar tracción en el mercado.

Las nuevas técnicas favorecieron el desarrollo del big data
Fuente: https://fondation.univ-grenoble-alpes.fr/medias/photo/chaire-e-sante_1549453746921-jpg?ID_FICHE=210089

### *1.3.1 Metodologías de programación de big data*

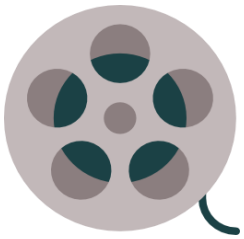
Las nuevas prácticas de programación permitieron aplicar computación distribuida en fuentes de datos masivos. De hecho, productos completos y nuevos ecosistemas han crecido en torno a estos avances. Tecnologías como MapReduce y Spark han contribuido y mucho a estos avances.

### *1.3.2 Arquitecturas de almacenamiento en big data*

Las metodologías de desarrollo de software tuvieron un efecto secundario de agitación en el tranquilo y hasta aburrido en el mundo del almacenamiento de datos. Nuevas tecnologías surgieron rápidamente para admitir nuevas arquitecturas de procesamiento distribuido, incluidos los sistemas de grandes archivos que se ejecutan en hardware económicos. Un ejemplo de una nueva tecnología de almacenamiento de datos es Hadoop Distributed File System (HDFS), que sirve como repositorio para enormes cantidades de datos estructurados y no estructurados.

La virtualización de las aplicaciones también ha traído considerables cambios en el mundo del almacenamiento de datos y sus diversas arquitecturas. Las arquitecturas tradicionales emplearon los sistemas de archivos de red (Network File System - NFS) y algunos otros protocolos en discos locales, mientras que las aplicaciones eran alojadas en servidores individuales dedicados.

El advenimiento de las máquinas virtuales cambió todo eso. Ahora, múltiples aplicaciones compatibles con su propia máquina virtual se ejecutan en un servidor físico y las arquitecturas de almacenamiento se trasladan a sus propios sistemas llamados redes de área de almacenamiento (Storage Area Networks - SAN). Estos almacenamientos compartidos y estas arquitecturas son cada vez más frecuentes, con sistemas de almacenamiento compartido ofreciendo análisis in situ, que permiten acceder al mismo dato utilizando muchos protocolos de archivo diferentes, incluidos NFS, SMB, POSIX, Object y HDFS. Los contenedores, a veces considerados una versión ligera de máquinas virtuales, aunque no convencionales, son el próximo paso de esta evolución.



**VIDEO DE INTERÉS**

A continuación podrás visualizar un vídeo introductorio sobre HDFS realizado por LUCA (Grupo Telefónica), y porque es importante para las empresas de big data.

https://www.youtube.com/watch?v=NQ8mjVPCDvk

# 2. LA IMPORTANCIA DEL DATO

*Tras la primera iniciativa del chatbot, llega el momento de centrarse en el filtrado y ofrecer un mecanismo para hacer buena inteligencia sobre todas las candidaturas recibidas. Es decir, validar qué candidatos son competentes y están cualificados para cada posición.*

*En este momento, tu idea es proponer la compra de un sistema de business intelligence, de forma que cada estamento lo pueda personalizar. Así, con una compra centralizada por parte de la Administración General del Estado, cada Ministerio, la Policía, la Guardia Civil, el ICEX, RTVE y todas y cada una de las agencias, empresas públicas e instituciones podrán modelizar su experiencia en selección de personal, para aplicar un cribado automático a todos los curriculum que se reciban, obteniéndose la máxima eficiencia en tiempos y costes y, lo más importante, se evitará la saturación del sistema.*

*Piensas que con un sistema de inteligencia de negocio se podrán analizar los datos de cientos de miles de candidatos. Incluso, se puede ir más allá y ofrecer una imagen de mayor transparencia, contando el sistema con un módulo de reporting que permitirá generar informes que expliquen el motivo que ha llevado a excluir o admitir a la fase posterior a cada perfil de candidato, etc.*

Es evidente que a lo largo de los últimos años la economía y los mercados han cambiado de manera muy importante, por varios motivos, pero especialmente gracias a Internet. De sobra sabemos que internet ha modificado nuestras formas de comunicarnos, relacionarnos e interactuar con los demás, pero también ha revolucionado las industrias y los mercados de todos los sectores.

Los competidores actuales ya no son los competidores que podríamos haber tenido hace diez años. Cuando se habla de mercados globales, los competidores también son globales: ya no solo importan las empresas próximas; en cualquier lugar puede haber un posible competidor.

Además, no solo es necesario conocer quiénes son nuestros competidores, debemos saber qué es lo que hacen, cómo lo hacen y prever, con el menor margen de error posible, qué es lo que van a hacer y cómo nos puede afectar eso.

Para que las empresas logren conocer a fondo el entorno y puedan decidir, de manera preventiva, las estrategias más adecuadas que han de realizar para mantener o mejorar nuestra posición en el mercado, el uso de los datos y su análisis es más importante que nunca.

El paso de los años ha demostrado que la utilización de sistemas de análisis es imprescindible para una correcta gestión de los activos informacionales, que determinarán nuestra posición en el mercado.

Las organizaciones de todo tipo buscan dotarse de un marco conceptual riguroso en el que fundamentar la toma de decisiones a partir de los activos informacionales. Para conseguir esto, se han realizado diversas propuestas, normalmente desde un plano tecnológico o informático.

Las organizaciones modernas, insertas en un entorno competitivo global y complejo, necesitan información confiable que pueda ser utilizada como un input más del proceso productivo o, mejor dicho, del proceso de adicción de valor. Es imperativo convertir datos e informaciones en conocimiento con un valor realmente estratégico.

El entorno globalizado en que se mueven las organizaciones requiere de sistemas y servicios de análisis de datos, que permitan consolidar informaciones dispersas, incluyendo fuentes humanas y técnicas, analizarlas y procesarlas adecuadamente para satisfacer las demandas cuantitativas y, sobre todo, cualitativas.

Red global
Fuente: pixabay.com

Es perfectamente comprobable que, desde su aparición, y en sus primeros estadios de racionalidad, el hombre ha requerido de informaciones para su subsistencia.

Los sistemas de información, antes de la aparición de la informática, se vienen utilizando para objetivos militares, políticos y comerciales desde hace siglos. Los comandantes y estrategas de las primeras civilizaciones humanas tenían muy claro lo crítico que era conocer al enemigo, como Sun Tzu, comprender los escenarios de conflicto, así como la situación meteorológica.

Históricamente, casi seguro que la primera cita escrita del uso de la información para prestar soporte a la toma de decisiones aparezca en el cuarto libro de Moisés, que describe un episodio bíblico donde se tomó la decisión de asentar las doce tribus de Israel en la tierra de Canaán.

Esta primera toma de decisión formal en la historia de la humanidad fue llevada a la pintura siglos más tarde de forma magistral por Giovanni Lanfranco, que en 1621 finaliza su obra titulada "Regreso de los espías de Canaán", donde nos muestra, de forma visual y apoteósica, como Moisés había enviado un informante a cada una de las doce tribus para conocer realmente el estado de las tierras donde, tentativamente, podría instalarse el pueblo de Israel.

El mundo de hoy y la estrategia contemporánea exigen nuevas demandas referidas al análisis de los datos, especialmente a quienes producen y usan la información procesada. Los cambios surgidos en las últimas décadas que han generado una nueva sociedad, donde la tecnología nos une y rompe fronteras, donde la globalización económica es imparable y, en general, los datos, la información y las tecnologías que sirven para gestionarlos son el sustento de cualquier organización, sea un ejército o una gran corporación mercantil.

Sin embargo, llegados a este punto, antes de profundizar en otras cuestiones, es necesario precisar una serie de conceptos imprescindibles:

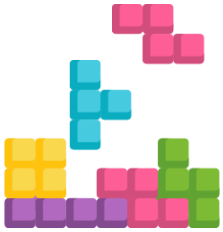
- **Datos:** los datos constituyen la materia prima para la producción de información y son todos aquellos materiales impresos, imágenes, etc. que dan lugar a un conjunto distinto de hechos objetivos.

- **Información:** es el resultado de la ordenación, integración y procesamiento de datos para producir un informe de interés genérico. Aporta un valor añadido a los datos a través de la contextualización, categorización, etc. Pueden ser fórmulas matemáticas, mapas, partituras…

- **Conocimiento:** la definición más habitual de conocimiento suele ser la de una creencia verdadera justificada. Sin embargo, hay dificultad en la definición de lo que es realmente el conocimiento, porque el conocimiento se basa en la paradoja de que reside sólo en las mentes de las personas y al mismo tiempo, se puede capturar, almacenar y compartir. Por lo tanto, el conocimiento no es un dato o un conjunto de datos, ni tampoco información. El conocimiento proporciona la capacidad de resolver problemas, innovar y aprender, con base a experiencias pasadas y mezcla diversos elementos tales como: la experiencia, el sentido de valor, la sabiduría…

- **Inteligencia:** son informaciones especialmente elaboradas para asesorar una decisión específica, tomada por una persona concreta. Se trata de un producto acabado, derivado de las informaciones y datos que previamente fueron seleccionados, validados, interpretados y, finalmente, expresados de tal forma que evidencian su importancia para un determinado problema.


Niveles conceptuales en torno a la información
Fuente: Elaboración propia



**EJEMPLO PRÁCTICO**

Se está desarrollando una entrevista que forma parte de la toma de requerimientos entre el equipo de usuarios del cliente y el equipo del proveedor de software que tu diriges.

En este caso, el cliente está interesado en disponer de un sistema de inteligencia de mercado que permita explotar del mejor modo posible la información de los clientes de la empresa contenida en el CRM.

En un momento de la conversación se llega a la conclusión por ambas partes de que se están utilizando los términos y conceptos básicos con distintos significados, lo que está provocando malentendidos entre vosotros.

Por este motivo, se te pide que pongas un ejemplo, en el contexto del cliente y del proyecto que se está planteando, que facilite el compartir una visión al respecto de los siguientes términos: dato, información, conocimiento e inteligencia

¿Qué ejemplos serían más visuales y claros para que el cliente tangibilizase el significado de las palabras?

**SOLUCIÓN:**

- La cifra con el importe en euros del dinero facturado a cada uno de los clientes de la empresa sería un dato que se almacena con total seguridad en el CRM que estén utilizando.

- Los gráficos que presentan la evolución histórica y la tendencia futura de las ventas a cada cliente por año, indicando si es un cliente en fase creciente o decreciente en importancia para la empresa, sería un ejemplo de información.

- La base de datos de preguntas frecuentes que se utiliza en el call center de la empresa para poder resolver las dudas comerciales de cada cliente al respecto de las características de los productos, el cómo se utilizan, etc., es un ejemplo de conocimiento.

- Finalmente, se puede decir que un ejemplo de inteligencia serían los informes periódicos que se elaboran para el director comercial, que le facilitan mensualmente tomar decisiones al respecto de las acciones comerciales para el mes siguiente, en lo que tiene que ver con la realización de eventos de ventas y campañas promocionales, contratación de nuevos comerciales, diseño de campañas de publicidad online, etc.

## 2.1 Contextualización práctica de la productividad del dato

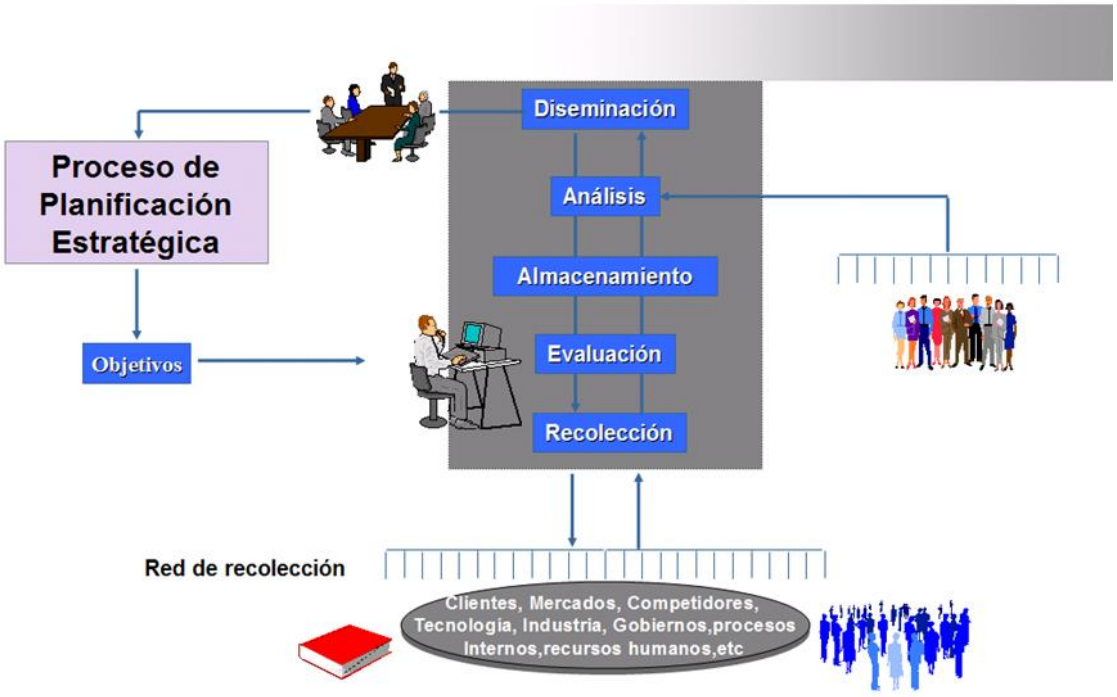
Veamos el siguiente modelo para obtener la mayor productividad posible de los datos disponibles.

Existe un modelo, generalmente aceptado, denominado ciclo de inteligencia, que presenta un núcleo común sobre los aspectos a contemplar cuando se realiza una explotación de los datos para obtener la máxima productividad posible sin perder rigor.

Este ciclo es especialmente reivindicado por algunos autores que lo recomiendan como fundamento del análisis de datos al considerar que genera interactividad, facilita la recolección de informaciones sobre los diversos elementos, sus intenciones y modos de desenvolverse en un escenario particular. Además, estos autores apuestan por el ciclo de inteligencia al considerar que, si ha sido probado y utilizado con éxito por organizaciones de todo tipo, se encuentra plenamente validado.

Se trata de un proceso que cuenta con distintas fases. En cada una de ellas se desarrollan diferentes actividades y tareas cuyo objetivo es convertir la información de que dispone una organización, y que ha sido captada por diversos medios, en conocimiento. La información debe ser, obviamente, verificada y analizada para que el directivo al que va dirigida la misma y que es el responsable de tomar una decisión, actúe en base a información confiable.

Como se observa en la imagen, el ciclo de inteligencia comienza en el propio proceso de planificación estratégica de cada organización, que es el que debe definir cuáles son las necesidades de análisis e inteligencia de una organización en base al alineamiento con los objetivos. Esto servirá para lanzar el ciclo de inteligencia donde en las sucesivas fases se procesa la información con el apoyo de analistas expertos y científicos de datos. La información proveniente de múltiples fuentes se adquiere mediante una red de recolección (tanto humana como informática).




Ciclo de inteligencia
Fuente: Elaboración propia

La creación de productos como resultado de este ciclo incluye tareas de selección, evaluación y creación de información, así como análisis de datos de numerosos orígenes. A diferencia de otros procesos secuenciales, este es un ciclo y, por tanto, todo lo que se realiza en cada una de las etapas condiciona los anteriores procedimientos.

**VIDEO DE INTERÉS**

En el siguiente enlace de vídeo podrás ver el ciclo de vida del dato en el marco de las tecnologías de la información y la comunicación.

https://www.youtube.com/watch?v=mU8dcfYdKyo

### *2.1.1 Planificación*

En la fase de planificación es fundamental establecer cuáles son los intereses que tiene una organización desde un punto de vista estratégico y, en base a ello, determinar qué necesidades de información tienen los tomadores de decisiones. Establecer objetivos erróneos en este momento provocará que el resultado del proceso de inteligencia no tenga ningún valor ni utilidad.

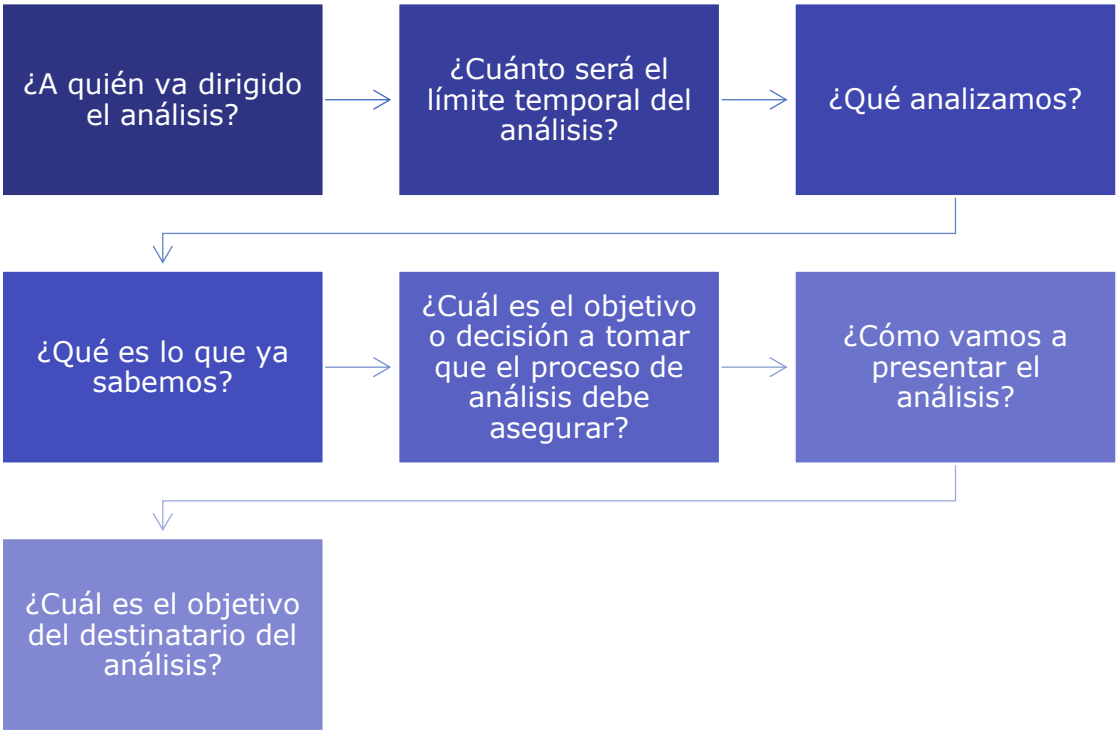
Es decir, si las necesidades de información de los usuarios, por así decirlo, los requerimientos, no han sido correctamente definidos, el resultado no tendrá valor para el consumidor del mismo, sino todo lo contrario, podría generar o influir en toma de decisiones incorrectas.

'Planificar' es la fase que implica definir objetivos fundamentales de la organización para la que trabaja el equipo de análisis de datos y los requerimientos de información concretos y demandados por los directivos. Es primordial, ya que la inteligencia es el producto de un procedimiento sistemático que se genera en las necesidades informativas de los usuarios.

Lamentablemente, como se observa de diversos estudios, son multitud los proyectos gestión de datos y análisis donde no existe una formalización de esta fase, de tal modo que la estrategia de recolección de información, la producción y entrega del resultado, etc., no está definido. Al no existir esta fase, el ciclo tampoco puede ser cerrado o retroalimentado, con lo que se genera una serie de incoherencias que impiden producir un análisis efectivo y evaluar la calidad de la pretendida analística que se ha generado.

Esta fase es el momento donde definir las que se han denominado las siete uves dobles del analista:

- Who: quien será el consumidor al que se destinará el producto de análisis resultante.
- When: que límite temporal tendrá el análisis realizado.
- What: la pregunta analítica.
- Know: lo que ya es conocido al respecto.
- Why: el objetivo o decisión a tomar que el proceso de análisis debe asegurar.
- How: el modo en presentar el análisis: dashboard, informe corto, informe largo, etc.
- What for: cuál es el objetivo del destinatario del análisis.




Siete preguntas para responder en el proceso de análisis
Fuente: Elaboración propia

La red de recolección es el conjunto de fuentes, internas y externas, humanas o informática, que permitirán generar respuestas a las preguntas de los usuarios a los que va dirigida la analítica, mediante la labor experta de los analistas.

Por lo tanto, tiene como objetivo captar información primaria, es decir, obtenida directamente en su origen, lo que permite cualificar la veracidad de la misma. Esto quiere decir que dentro del módulo de planificación y dirección de inteligencia es crítico para una organización saber con qué red de recolección cuenta y cuáles son las capacidades de la misma.

Asombraría conocer el número altísimo de oportunidades para acceder a fuentes de información primarias que tienen las organizaciones y que se desperdician porque no conocen las capacidades de sus colaboradores y empleados o, porque no saben guiarlos adecuadamente para ello.

Por ejemplo, **las plataformas en la nube de CRM son mecanismos excepcionales para recolectar multitud de datos e informaciones que están en la cabeza de todos los vendedores de la empresa**.

Igualmente, relevante es saber incorporar elementos de recolección no existentes en la organización.

### 2.1.2 Procesamiento

Esta etapa consiste en el **estudio técnico, recopilación, inspección y conservación de los datos que han sido reunidos** de maneras diversas para convertirlos e integrarlos en conjuntos estructurados de información que pueden adoptar la forma de mensajes documentales tras su recuperación.

**Las tareas relacionadas con la gestión de la documentación tales como la recopilación, identificación, descripción y análisis documental (incluyendo indexación y resumen) de los datos, representan una parte considerable de esta etapa del proceso de hacer productivo al dato**. La elaboración de bases de datos adecuadas en las que se incluyen los datos obtenidos de diversos orígenes, pero sobre un mismo asunto, es el principal objetivo.


Actividades fundamentales para hacer productivo al dato
Fuente: Elaboración propia

Es muy importante que en esta fase controlemos, de forma eficaz, el elevadísimo volumen de datos, información en bruto y documentos con los que se trabaja en cada departamento de informática o de análisis, para evitar la saturación o el colapso. La gestión y la consecutiva recuperación de la información implica **utilizar diferentes tecnologías auxiliares como sistemas electrónicos de gestión de datos, agentes inteligentes de recuperación de información y aplicaciones para la minería de datos**.

Profundizando un poco más en este tipo de tecnologías, diremos que:

- **Los sistemas electrónicos de gestión de datos más utilizados en análisis y gestión de datos son los gestores documentales**, que son aplicaciones informáticas creadas para la gestión de elevadas cantidades de información textual, normalmente no estructuradas o semiestructuradas. Estas herramientas utilizan técnicas de indexación y entendimiento de la información textual, desarrollando representaciones de la misma sobre las cuales se llevan a cabo los procesos de búsqueda y recuperación de la información. Incluso, actualmente, estas herramientas integran información y documentación multimedia e hipermedia y capacidades extendidas de acceso a recursos heterogéneos.

- **Los agentes inteligentes de recuperación de información son aplicaciones informáticas destinadas a la indexación automática y a la búsqueda de información** en un entorno de red, capaces de acceder a datos en diversos formatos y provenientes de diversas fuentes. Los objetivos a futuro de los agentes inteligentes es que sean capaces de aprender de forma eficiente del usuario -en base a los hábitos de recuperación que suela mostrar- así como alcanzar el ideal de estos agentes inteligentes que sería cuando estos pudiesen tomar decisiones propias de acuerdo con el contexto y con las reglas operativas que les ha definido el programador.

- Otro tipo de aplicaciones en liza son las de **minería de datos**, que se utilizan para identificar y **extraer información en grandes volúmenes de datos**, que de otro modo permanecería oculta si solo se utilizasen técnicas clásicas de recuperación de información. Existen dos tipos fundamentales de aplicaciones:

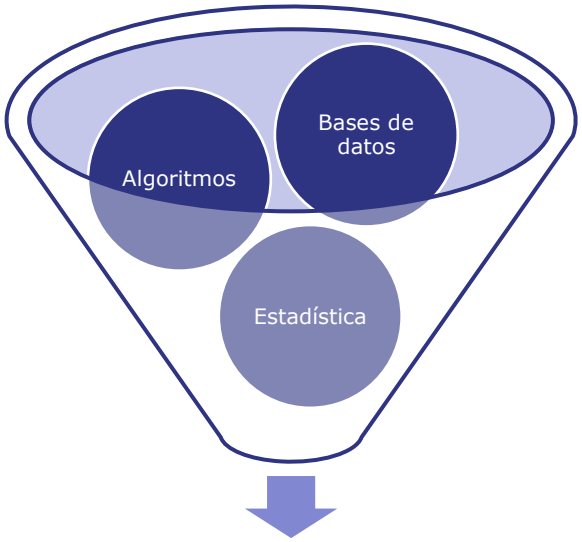
    - Unas están destinadas a la revelación de vínculos y patrones ocultos dentro de agrupaciones de datos estructurados en varias bases de datos relacionales, que sería la minería de datos.

    - Otras están destinadas a la instauración de asociaciones entre palabras y conceptos en datos no estructurados que es la minería de textos, también conocido como minería web si el entorno de información se encuentra disperso entre una red de servidores de información.

Para ambos objetivos se utilizan **algoritmos diversos de identificación y agrupación de información significativa, como la fijación de partes o cadenas de caracteres, el análisis clúster para la agrupación automática, la categorización automatizada**, etc.

Las aplicaciones y soportes que se encargan de estas funciones deben cumplir tres objetivos:

- Deben ser seguras.
- Tienen que ser los más invulnerables posibles al paso del tiempo.
- Deben garantizar, a lo largo del tiempo, el acceso y la legibilidad de la información contenida en los soportes.




Componentes de la minería de datos
Fuente: Elaboración propia

### *2.1.3 Análisis*

Esta fase del ciclo **es clave para la productividad y consiste en extraer con precisión y rapidez información a partir de los repositorios de datos que induzca a la construcción de conocimiento**. En ella se establece el límite entre datos / información e inteligencia.

Se realizan tres tareas en esta etapa:

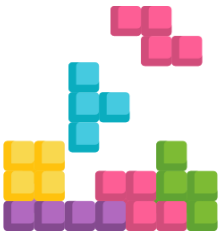
- **Evaluación o valoración de los datos.** Nos referimos a cuando estamos tratando de diferenciar qué informaciones de las que disponemos facilitan la satisfacción de las necesidades informativas fijadas (diferenciación que realizamos atendiendo a términos de fiabilidad, validez, oportunidad, pertenencia, relevancia y utilidad).

- **Integración de datos obtenidos a través de diversos orígenes.** Se sustenta en la regla principal del análisis que expone que no se debe tener en cuenta nunca una única fuente. Se trata de alcanzar una unión en la que la combinación de datos originarios de diferentes fuentes tenga mayor peso y relevancia que los obtenidos por fuentes individuales.

  Este proceso de integración puede llegar a ser realmente complejo en proporción a la cantidad de datos y de fuentes de información que empresa esté manejando. Este creciente exceso de información y fuentes nos lleva a la necesidad de contar con unidades de análisis de datos mucho menos jerárquicas y donde la información y los documentos menos críticos fluyan de forma horizontal entre todos los departamentos y analistas.

  En general, la tendencia actual es evitar una integración de los datos monolítica y tender más a una integración, por así decirlo, en tiempo real, donde la información fluya y sea cada analista el que pueda acceder a la que considere más relevante.

- **Interpretación.** La interpretación de los datos se realiza para alcanzar dos metas: fijar aquello que es exacto y lo que es pertinente para satisfacer las necesidades del directivo que está encargado de tomar las decisiones, es decir, del consumidor de los procesos analíticos.



**EJEMPLO PRÁCTICO**

En el proyecto dónde estás dirigiendo un equipo de desarrolladores software diseña el sistema de inteligencia de mercado partiendo del CRM de uno de vuestros clientes, teniendo claro que la etapa de análisis es especialmente importante para el éxito del proyecto.

Por este motivo, ¿qué aspectos debes cuidar más en cuanto a valoración de los datos e integración de datos obtenidos a través de diversos orígenes?
o
**SOLUCIÓN:**

En el primer punto, en lo que tiene que ver con evaluación o valoración de los datos disponibles se parte de una gran ventaja, que es que los datos disponibles son de tu propio cliente, por lo que la fiabilidad, validez, (y en menor medida la oportunidad, pertinencia, relevancia y utilidad) está garantizada, porque ellos mismos son quienes los han generado.

En el segundo punto, referido a lo interesante de integrar diversas fuentes de datos, tienes claro que muchas de esas integraciones, especialmente en lo referido con fuentes externas, deben ser en tiempo real. Por ejemplo, puede ser relevante para el sistema de inteligencia de mercado integrar las noticias que se publican de los clientes. Imagina que el sistema de inteligencia detecta que uno de los clientes ha abierto negocios en otro país, sería un momento perfecto para dirigirse a ellos e intentar venderles nuevos suministros.

Obviamente, esta integración debe permitir ese "tiempo real", porque de otro modo, es posible que un competidor se entere antes y se adelante. Queda claro que la integración de distintas fuentes combinadas ofrece un potencial elevadísimo.

Las necesidades del tomador de decisiones suelen ser el conocer y comprender un fenómeno determinado y tener una previsión de las consecuencias y evolución de dicho fenómeno. Por tanto, la interpretación es una labor para especialistas en cada área de estudio: tecnología, economía, seguridad, etc. que deben tener tantos conocimientos como suficientes aptitudes de ingenio y creatividad para ligar la información y predecir los sucesos.

En cuanto a las tecnologías que sirven de soporte a los analistas en esta fase de explotación del dato se encuentra el **groupware,** que facilita el trabajo en grupo, la coordinación de equipos, la comunicación, etc.


La herramienta Slack es un ejemplo de tecnología groupware
Fuente: slack.com

También puede ser interesante el uso de los "**decisión support systems**" o sistemas de apoyo a la toma de decisiones, que pueden contribuir al filtrado y la integración de gran cantidad de datos para facilitar la resolución de problemas con gran rapidez. Estas utilidades integran y gestionan muchos datos agrupados y en tiempo real, representan supuestos escenarios con situaciones del tipo "what if" y reemplazan la presentación ordenada de la información por vínculos de datos basados en parámetros mentales, destapando los nexos racionales, semánticos y de contenidos que pueden existir entre ellos.

### *2.1.4 Diseminación*

El resultado del análisis da lugar a un documento (report) cuya tipología (dossier, resumen, estadística) se fija según la mayor o menor elaboración de la información que transmite.

El documento que requiere una mayor elaboración es un informe de inteligencia completo, que es desarrollado por uno o varios especialistas en el dominio concreto de análisis, que reúne los sucesos que se conocen, indica los que se desconocen, anota los orígenes de los datos empleados, los integra, evalúa los sucesos que se estudian, aporta indicaciones para el análisis de la información, calcula las alternativas de interpretación y valoración, efectúa pronósticos y califica el grado de protección que se le debe aplicar.

Estos productos de análisis se reparten y ponen a disposición de la persona que solicitó esa información, de manera fiable y segura, y está encargada de las decisiones a seguir.

### *2.1.5 Evaluación*

La transmisión de los informes de análisis de los datos no implica el término del proceso. Si no que a continuación es necesario analizar las reacciones de los usuarios ante la información suministrada, identificar los objetivos que se han cumplido con las decisiones adoptadas sobre su base y valorar la importancia que la información ha tenido en el logro de los objetivos. Sus frutos posibilitan redefinir las conclusiones derivadas de las acciones seguidas durante la etapa de planificación e incluso determinan los venideros procedimientos de adquisición de información indicando los tipos de datos a recabar y la manera de conseguirlos.

Un recurso de información no tiene un sentido por sí mismo, sino que lo adquiere en un contexto concreto y será consecuencia del que le asigne el usuario, que dependerá de su habilidad para generar conocimiento que sea utilizable y de aplicación en el área de acción de la organización. Su valor se halla en función de los objetivos y requerimientos de la organización y, como estos son variables, resulta que los recursos de información solo tienen un valor potencial, cuya transformación en el tiempo resulta inesperada, porque está determinada por quién, cómo, cuándo y para qué lo use.

La evaluación debe mostrar también cuales han sido los obstáculos más importantes que han surgido a lo largo del análisis para subsanarlas. Hay que considerar que existen unas problemáticas como la urgencia impuesta por los usuarios para contar con un análisis incluso antes de que se tengan las fuentes aceptables y necesarias, la incapacitación del analista para contrastar todas las variables analizables, el desconocimiento de la calidad de la información a causa de errores en el proceso de selección, el hincapié que se debe dar a la predicción y la perspectiva utilitaria de los resultados, etc.

Evaluando el error

Fuente: https://www.freepik.es/foto-gratis/hombre-negocios-que-senala-dedo-analisis-estadisticas-financieras-grafico_1025846.htm#page=1&query=evaluacion&position=4

## 2.2 Tipología de los datos

En todo proyecto de análisis e investigación con datos, el acceso y gestión de los mismos resulta fundamental. Sin embargo, ¿qué es lo que podemos considerar un dato en este contexto?

La Organización para la Cooperación y el Desarrollo Económicos considera datos de investigación y análisis a todo aquel material que ha sido registrado durante una investigación y que sirve para certificar los resultados de dicha investigación.

La Office of Management and Budget de EE. UU. define los datos de una investigación de la siguiente manera: "Los datos de investigación se definen como aquel material registrado comúnmente aceptado por la comunidad científica como necesario para validar resultados de investigación. No serían datos de investigación ni los análisis preliminares, borradores de la elaboración de artículos, planes de investigaciones futuras, revisiones por pares ni comunicaciones entre colegas."


Logo de la Office of Management and Budget de EE. UU.
Fuente: whitehouse.gov

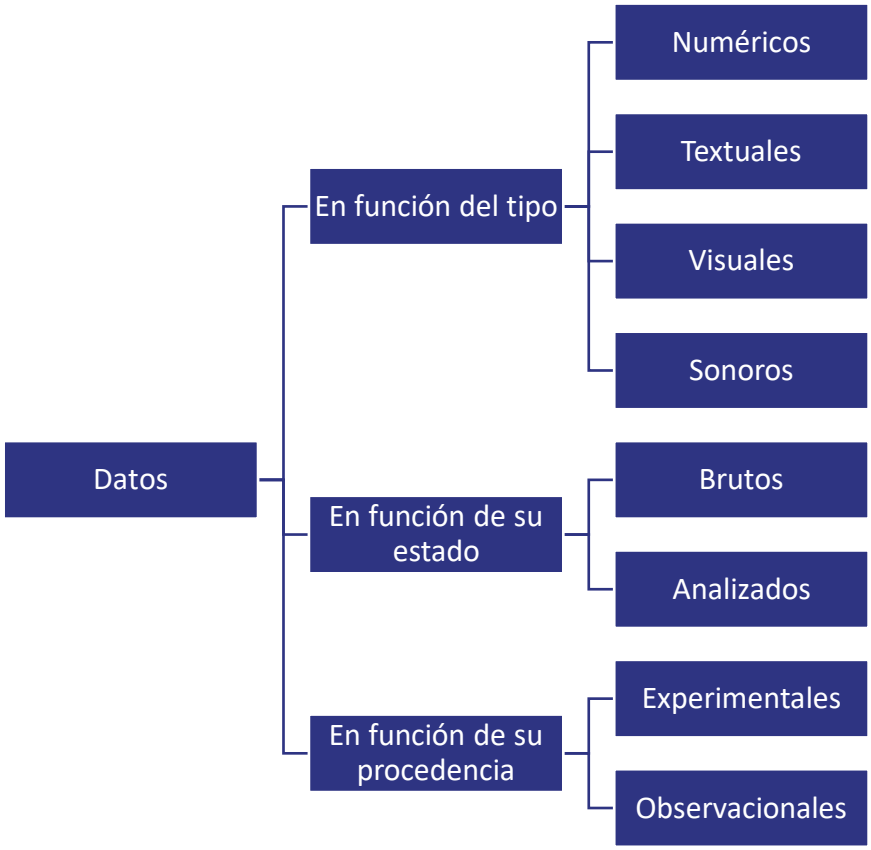
Otra definición de datos de una investigación es la que da Borgman (2008): "Los datos son una representación reinterpretable de la información de manera formalizada y adecuada para su comunicación, interpretación o procesamiento. Algunos ejemplos de datos incluyen **una secuencia de bits, una tabla de números, los caracteres de una página o la grabación de sonidos hechos por una persona que habla**."

Los datos se pueden clasificar de acuerdo a varios criterios:

- En función del tipo de dato: numéricos, textuales, visuales y sonoros.
- En función de su estado: brutos y analizados.
- En función de su procedencia: experimentales y observacionales.

El repositorio científico Recolecta, desarrollado por el Ministerio de Ciencia e Innovación, considera datos científicos:

"Cuadernos de laboratorio, cuadernos de campo, datos de investigación primaria (incluidos los datos en papel o en soporte informático), cuestionarios, cintas de audio, vídeos, desarrollo de modelos, fotografías, películas, y las comprobaciones y las respuestas de la prueba. Las colecciones datos para la investigación pueden incluir diapositivas; diseños y muestras. En la información sobre la procedencia de los datos también se podría incluir: el cómo, cuándo, donde se recogió y con qué (por ejemplo, instrumentos). **El código de software utilizado para generar, comentar o analizar los datos también pueden ser considerados datos.**"




Clasificación de los datos
Fuente: Elaboración propia

Una vez que hemos identificado los tipos de datos que necesitamos manejar tendremos que seleccionar las **fuentes adecuadas**, así como los **métodos de captura** de dichos datos.

Se consideran fuentes de información todos aquellos mecanismos mediante los cuales se pueden obtener datos útiles para satisfacer una necesidad de conocimiento.

La clasificación de las fuentes de información nos permite identificar rápidamente la audiencia, el perfil del creador (humano o máquina), el contenido, el lenguaje y el propósito de la información. Esto nos ayuda a elegir el tipo de información correcta para satisfacer nuestras necesidades de información profesional y empresarial.

El público al que se dirige, el tipo de material y el formato son las tres formas de categorizar las fuentes de información más habituales.

En función del público al que están dirigidas, nos encontramos que las fuentes pueden ser:

- **Fuentes populares:** son aquellas que informan y entretienen. Revistas generalistas, periódicos o libros. Una publicación popular contendrá un lenguaje fácilmente comprensible para una audiencia general. Generalmente, son escritas por periodistas o escritores independientes y no se someten a una revisión formal por parte de expertos antes de su publicación. Las fuentes de información populares generalmente no tienen citas o referencias bibliográficas completas de la información utilizada para escribir el artículo o libro. Las revistas y los periódicos incluyen lenguaje no técnico y suelen estar libres de jerga para facilitar la lectura. Tienden a cubrir temas con resúmenes relativamente breves. Las publicaciones populares pueden ser un buen punto de partida para aprender los componentes básicos de un tema, para comprender los diversos puntos de vista en torno a un asunto o para descubrir posibles ángulos a explorar con un análisis más profundo.

- **Fuentes académicas:** Las publicaciones académicas son una fuente de información adecuada para conocer en profundidad una materia. Estas publicaciones suelen consistir en investigaciones y estudios originales. Las publicaciones académicas también contienen análisis de expertos sobre temas o asuntos, como puede ser una obra literaria o un problema social. Muchas de estas fuentes de información se someten a un examen por otros expertos del área antes de su publicación, lo que se conoce como revisión por pares o peer-review. Estas revisiones permiten garantizar la calidad, credibilidad y exactitud de la información. Los expertos en un campo escriben información académica para que otros expertos o académicos lean y avancen en el conocimiento de dicho campo. Esto provoca que la información puede ser a veces difícil de entender para los que están comenzando en esa área o para personas de otras áreas que quieren saber un poco más sobre esa materia. Las publicaciones académicas, a menudo, utilizan terminología técnica y científica, lo que supone que sus autores tienen un conocimiento previo del tema bastante profundo. Además, tienden a ser más extensas y profundas que las fuentes de información populares o comerciales y se centran en aspectos muy concretos más que en una visión general. Todas las publicaciones académicas incluyen una bibliografía y una lista de referencia de las fuentes consultadas o citadas por los autores.

- **Fuentes comerciales:** son diferentes de las fuentes de información académica y popular, aunque pueden contener elementos de ambas. Se trata de publicaciones que comparten información entre personas de una industria específica para mejorar su negocio y para mantenerse al día sobre las tendencias del mercado. Las publicaciones comerciales pueden ser muy útiles para la investigación dentro de un campo específico. Los profesionales de una industria escriben información dirigida a otras personas que trabajan en su ámbito. La información que se comparte en una publicación comercial tiende a ser actual y a incluir las tendencias. Estas publicaciones están muy especializadas y abordan gran variedad de cuestiones, puntos de vista y perspectivas relacionadas con el tema. La mayoría de las publicaciones comerciales gozan de buena reputación entre los expertos de la industria y proporcionan información verificable y fiable. Algunas publicaciones comerciales se someten a un proceso de revisión por pares para garantizar la exactitud y la pertinencia antes de su publicación. Un ejemplo de este tipo de publicación podrían ser los informes que realiza Gartner que son utilizados por empresas de diversos sectores informáticos.


Clasificación de las fuentes de información por el público al que se dirigen
Fuente: Elaboración propia

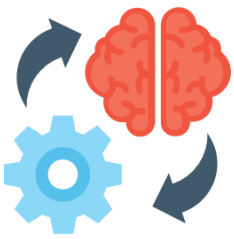
Las fuentes de información, por el tipo de material, pueden ser de un tipo primario, secundario o terciario, dependiendo de cuándo fue creada y de su propósito y alcance. Es importante comprender el valor y las diferencias de utilizar estas fuentes de información, ya que cada una tiene un propósito diferente.

- Por lo general, se considera que las fuentes primarias son aquellas que no han sido analizadas o alteradas. A veces, los documentos de fuentes primarias son de la época en la que se produjo el suceso. Los tipos de documentos que se consideran fuentes primarias varían según la disciplina.

Las fuentes primarias incluyen estudios de investigación y conjuntos de datos originales, como la **información del censo**, en estado bruto y sin analizar.

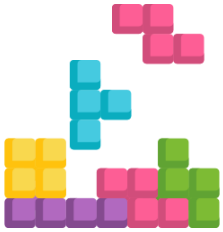
- Para algunos autores, las fuentes primarias son todas aquellas que contienen información original no abreviada ni traducida, y las fuentes secundarias serían aquellas que contienen datos sintetizados a partir de fuentes primarias. Es decir, las fuentes primarias se diferenciarían de las secundarias en que contienen información original e inédita, mientras que **las secundarias son fruto del análisis y organización de la información que se encuentra en las fuentes primarias**.

- Las fuentes de información terciaria recopilan, indexan u organizan información de fuentes primarias y secundarias, a menudo, para proporcionar una visión general de un tema. Este tipo de documentos rara vez contiene material original. Suelen ser una buena fuente de datos y hechos presentados con contexto para ayudar a interpretar un tema, porque proporcionan una perspectiva amplia sin ninguna crítica o punto de vista relacionado con el tema. También pueden servir de directorio de otras fuentes primarias o secundarias importantes, porque hay analistas que a menudo escriben el contenido de muchos materiales terciarios. Entre los ejemplos estarían los boletines de resúmenes, bibliografías, enciclopedias, diccionarios o directorios.



**RECUERDA**

El hecho de que una fuente de información no contenga información original o inédita no quiere decir que no sea una fuente de información válida.



**EJEMPLO PRÁCTICO**

Estás haciendo un análisis de cómo afecta la crisis sanitaria del COVID-19 a la economía en España, concretamente, en el sector naval. ¿Qué fuentes de información tomarías?

**SOLUCIÓN**:

Las fuentes de información populares, como los periódicos, mostrarán información muy superficial sobre cómo afecta a la economía, porque están dirigidos a una audiencia muy amplia, por lo que no son la fuente adecuada para una investigación profunda.

Las fuentes de información gubernamentales nos proporcionarán información estadística valiosa pero los organismos públicos tardan muchos meses en poner a disposición del público información muy específica como la que podríamos necesitar para esta investigación.

Desde que se produce la crisis del COVID-19 hasta que se llegan a generar documentos académicos de un ámbito tan específico pasarán meses o incluso años, por lo que probablemente no encontremos ningún documento que pueda ser de utilidad en este momento.

Los informes realizados por las propias empresas del sector naval serán los que nos proporcionen la información más precisa y actualizada. Esta sería la fuente de información ideal en este caso.

Las fuentes de información se distinguen también por el tipo de formato. La organización, el público destinatario, la duración y las normas de publicación definen el tipo de formato de la información. Cada formato presenta la información de una manera diferente y con un propósito distinto. Podemos distinguir los siguientes tipos de fuentes de información en función de su formato:

- **Libros.** Los libros existen en formato impreso, electrónico y de audio. Se estructuran, por lo general, siempre de la misma manera.

- **Revistas científicas.** Contienen artículos escritos por expertos en la materia, científicos o investigadores, y son una excelente fuente de información.

- **Revistas generalistas.** Estas revistas entretienen e informan a una audiencia general y, frecuentemente, discuten sucesos recientes. Los periodistas u otros escritores profesionales suelen ser los autores de los artículos en lugar de expertos en la materia.

- **Periódicos.** Al igual que las revistas generalistas, los periódicos proporcionan información a un amplio público, cubren temas de actualidad y son una buena fuente de información popular.

- **Vídeo y audio.** Muchas grabaciones de vídeo y audio proporcionan buena información para su uso en un proyecto de análisis.

- **Documentos gubernamentales.** Incluyen una amplia gama de información actual e histórica, como documentos de los tribunales, informes, estadísticas, resoluciones, tratados... Los documentos del gobierno son fuentes de información autorizada.

- **Literatura gris.** La literatura gris es material publicado informalmente por expertos o investigadores en un campo. Entre los ejemplos de literatura gris se incluyen actas de congresos, informes técnicos, ensayos clínicos, conjuntos de datos de investigaciones, tesis y otros trabajos universitarios, presentaciones de conferencias, los boletines de novedades departamentos o las publicaciones en blogs de expertos acreditados.

- **Sitios web.** Los sitios web son la principal forma de información electrónica disponible. La calidad, cobertura y propósito de estas fuentes variará significativamente. Hay mucha información excelente en línea, pero debido a la amplia gama de fuentes, es cada vez más importante examinar la información que se encuentra en los sitios web para determinar su fiabilidad.

**¿SABÍAS QUE...?**

En 1990, el centro de investigaciones nucleares CERN, en Ginebra, puso en línea la primera página web de la World Wide Web.

**Cada vez es más frecuente la utilización de datos abiertos** (open data) disponibles en línea, ya sean procedentes de organismos públicos o privados.

Los datos abiertos tienen las siguientes características:

- Son accesibles. Esto quiere decir que se pueden encontrar fácilmente para que puedan utilizarse.
- Han sido certificados, de forma que son creíbles.
- Su código es accesible y manipulable, lo que puede dar lugar a crear nuevos datos.
- Son comprensibles.
- Son libres y se pueden reutilizar sin restricciones.
- Son gratuitos, es decir, que acceder a ellos no implica ningún coste.

Los beneficios que aporta la publicación de los datos en abierto son:

- Aumento de la visibilidad. El acceso abierto significa los datos están abiertos a cualquiera, en cualquier lugar, para leer, compartir y reutilizar. Así se facilita la conexión con otros analistas.
- Rapidez. Con el Open Access, los trabajos están disponibles inmediatamente. Cuando los analistas pueden leer y construir sobre los hallazgos de otros sin restricciones, se avanza más rápido.
- Acceso en igualdad de condiciones. El acceso abierto permite que todos se beneficien de los avances al proporciona un acceso igualitario a los últimos hallazgos.

**Algunos repositorios informáticos de datos abiertos que pueden resultar interesantes** para realizar análisis son los siguientes. Existen multitud de fuentes abiertas que son accesibles y permiten enriquecer los procesos de análisis si son fusionadas con las fuentes de datos internas a la empresa:

- **Google Dataset Search.** Es un servicio que permite buscar en cerca de 25 millones de diferentes conjuntos de datos disponibles públicamente. Los investigadores pueden acceder a conjuntos de datos de todo tipo que contienen incluso datos de audio e imágenes anotadas, para comprobar sus hipótesis analíticas.

**ENLACE DE INTERÉS**

En el siguiente enlace puedes acceder a la página web de Google Dataset Search.

https://datasetsearch.research.google.com/

Entre los datos que podemos encontrar se destacan:

- Kaggle human resources dataset: datos de test y académicos para aprendizaje en cuestiones analíticas como: ¿Hay alguna relación entre el jefe de un trabajador y su rendimiento? ¿Cuál es el grado de diversidad de la organización? ¿Cuáles son las mejores fuentes de reclutamiento si queremos asegurar una organización diversa? ¿Es posible predecir quién va a ser despedido y quién no? ¿Qué nivel de precisión podemos lograr en esas predicciones? ¿Existen áreas de la empresa en las que la remuneración no es equitativa?...

- Varios sets de encuestas sobre recursos humanos en países, ministerios, alcaldías, etc. De diversos países del mundo.

- **Aporta.** Iniciativa del gobierno de España que busca promover la apertura de la información pública y desarrollo de servicios avanzados basados en datos abiertos. Actualmente reúne datos de organismos de diferentes administraciones:

  - 43 de la Administración General del Estado.
  - 18 de administraciones regionales.
  - 230 de entidades locales.
  - 10 de universidades.

- **Data.gov.** El Gobierno de los Estados Unidos se comprometió en 2015 a poner todos los datos del gobierno a disposición del público de forma abierta. Este sitio actúa como un portal para todo tipo de información que va desde el clima hasta el crimen.

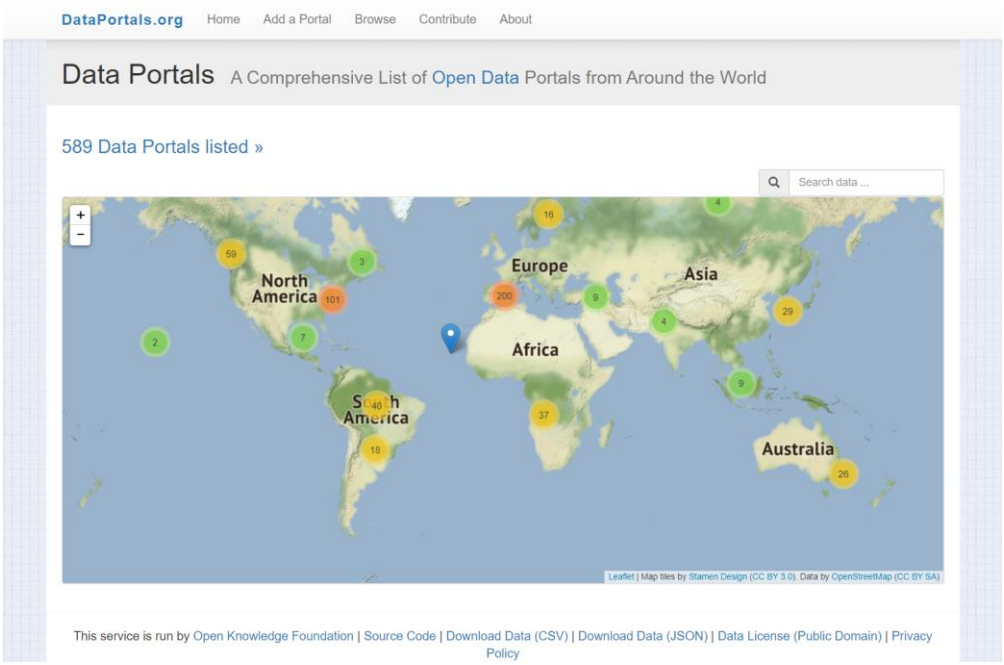
- **Oficina del Censo de EE. UU.** Contiene una gran cantidad de información sobre los ciudadanos de los EE. UU., que abarca datos de población, datos geográficos y educación.

- **Portal de Datos Abiertos de la Unión Europea.** Contiene datos de todo tipo de las instituciones de la Unión Europea.

- **Data.gov.uk.** Datos del Gobierno del Reino Unido, incluida la Bibliografía Nacional Británica, con metadatos de todos los libros y publicaciones del Reino Unido desde 1950.

- **Canada Open Data.** Es un proyecto piloto con muchos conjuntos de datos gubernamentales y geoespaciales.

- **Datacatalogs.org.** Ofrece datos gubernamentales abiertos de los Estados Unidos, la Unión Europea, Canadá...




Página principal de Datacatalogs
Fuente: datacatalogs.org

- **The CIA World Factbook.** Información sobre la historia, la población, la economía, el gobierno, la infraestructura y el ejército de 267 países.

- **Healthdata.gov.** Datos desde hace 125 años sobre la atención de la salud en los Estados Unidos, incluidos datos de Medicare a nivel de reclamaciones, epidemiología y estadísticas demográficas.

- **Centro de Información sobre Salud y Atención Social del NHS.** Conjuntos de datos de salud del Servicio Nacional de Salud del Reino Unido.

- **UNICEF.** Ofrece estadísticas sobre la situación de las mujeres y los niños en todo el mundo.

- **Organización Mundial de la Salud.** Ofrece estadísticas sobre el hambre, la salud y las enfermedades en el mundo.

- **Conjuntos de datos abiertos disponibles en Amazon Web Services.** Incluye el 1000 Genomes Project, un intento de construir la base de datos más completa de información genética humana y la base de datos de la NASA de imágenes satelitales de la Tierra.

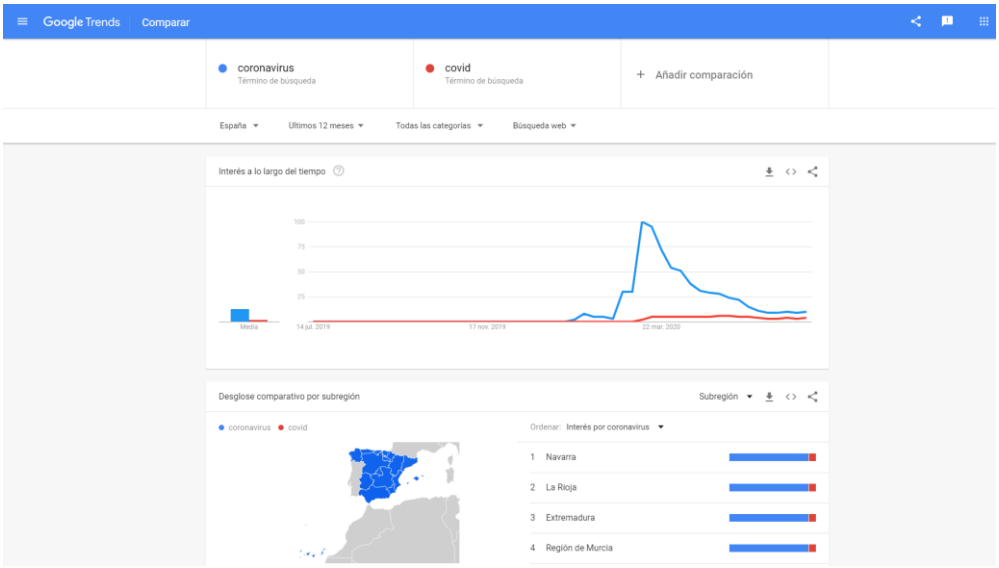
- **Facebook API Graph.** Aunque mucha de la información del perfil de los usuarios de Facebook es privada, otra no lo es. Facebook proporciona esta herramienta para consultar la gran cantidad de información que sus usuarios están dispuestos a compartir con el mundo.

- **Data Market.** Es un repositorio para revisar datos relacionados con la economía, la salud, la alimentación y la agricultura y la industria automotriz.

- **Buzzdata.** Es un servicio social de intercambio de datos que le permite a los usuarios cargar sus propios datos y conectarse con otros que están cargando sus datos.

- **Gapminder.** Recopilación de fuentes de datos como la Organización Mundial de la Salud y el Banco Mundial que abarcan estadísticas económicas, médicas y sociales de todo el mundo.

- **Google Trends.** Estadísticas sobre el volumen de búsqueda (como proporción del total de la búsqueda) para cualquier término dado, desde 2004.




Ejemplo de búsqueda en Google Trends
Fuente: Elaboración propia

- **Google Finance**. Datos bursátiles desde hace 40 años, actualizados en tiempo real.

- **Google Books Ngrams.** Busca y analiza el texto completo de cualquiera de los millones de libros digitalizados.

- **Centro Nacional de Datos Climáticos.** Gran colección de conjuntos de datos ambientales, meteorológicos y climáticos del Centro Nacional de Datos Climáticos de los Estados Unidos. Se trata del archivo más grande del mundo de datos meteorológicos.

- **DBPedia.** Wikipedia está compuesta por millones de datos, estructurados y no estructurados sobre cualquier tema. DBPedia es un proyecto para catalogar y crear una base de datos pública y de libre distribución que permita a cualquiera analizar estos datos.

- **New York Times.** Cuenta con un archivo indexado y recuperable de artículos de noticias que se remontan a 1851.

- **Open Data Network.** Permite a los usuarios buscar datos utilizando un robusto motor de búsqueda. Aplica filtros avanzados a las búsquedas, y saca datos de todo tipo: seguridad pública, finanzas, infraestructura...

- **Global Financial Data.** Con una suscripción gratuita, los usuarios pueden acceder a los conjuntos completos de datos e investigaciones que realizan para analizar los principales mercados y economías mundiales.

- **Base de datos Comtrade de las Naciones Unidas.** Esta base de datos de libre acceso contiene conjuntos de datos sobre el comercio mundial y es accesible a través de su API. También hay disponibles herramientas de visualización y extracción de datos.

- **Fondo Monetario Internacional**. Contiene datos abiertos para conocer las perspectivas económicas mundiales, la estabilidad financiera, la supervisión fiscal...

- **Oficina de Análisis Económico del Departamento de Comercio de los Estados Unidos.** Esta fuente de datos abierta se actualiza frecuentemente con conjuntos de datos sobre el PIB, el comercio internacional de bienes y servicios o las transacciones internacionales entre otras materias.

### 2.2.1 Business intelligence

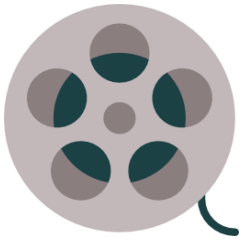
Para lidiar con las distintas tipologías de datos, nace el concepto de business intelligence. Es un término que, según la empresa de análisis de mercado y tecnologías, Gartner Group, fue concebido por ellos en los años ochenta para mostrar las aptitudes de una empresa para alcanzar y aprovechar las informaciones que se encuentran en una base de datos, para que pueda ser estudiada por los usuarios y ser utilizada para generar teorías y conocimientos que apoyen la toma de decisiones.

Más concretamente, según el investigador del Grupo Gartner, fue Howard Dresner quien popularizó la expresión business intelligence, que consistiría en la habilidad de los usuarios finales para acceder y analizar variedades cuantitativas de información y tomar las decisiones adecuadas.

Puede que ellos sean los que hayan introducido comercialmente tal denominación, pero mucho antes ya se escribía y se practicaba business intelligence y, muchísimo antes, inteligencia en general.

**¿SABÍAS QUE...?**

El término business intelligence, o inteligencia empresarial, surgía en un artículo de 1958 del investigador de IBM Hans Peter Luhn.

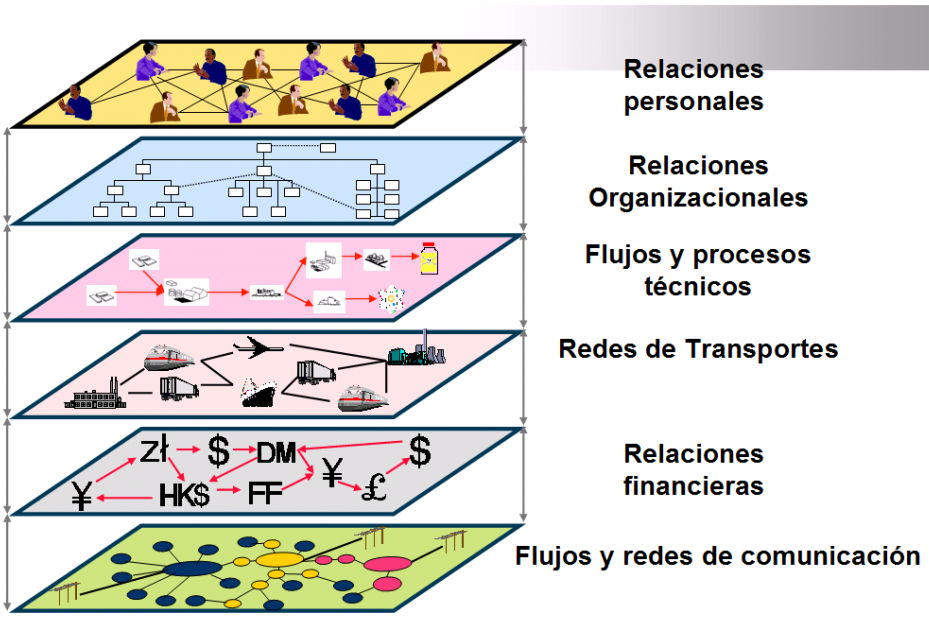


**VIDEO DE INTERÉS**

Este vídeo de animación te ofrece un recorrido histórico por la evolución del business intelligence.

https://www.youtube.com/watch?v=BOWCZxSSOMM

**La transformación de datos en informaciones y conocimientos es un camino arduo de recorrer**. Los datos son elementos que mantienen una estructura en bruto (texto, imágenes, etc.) y que, por tanto, no ayudan a comprender las situaciones, siendo necesaria la información, en forma y fondo adecuados para un uso específico, y eso es lo que pretende el business intelligence. Pero, con frecuencia, ocurre un problema grave, que es el **carácter multinivel y relacional de la información** necesaria para generar inteligencia.




Carácter multinivel de los datos y la información
Fuente: Elaboración propia

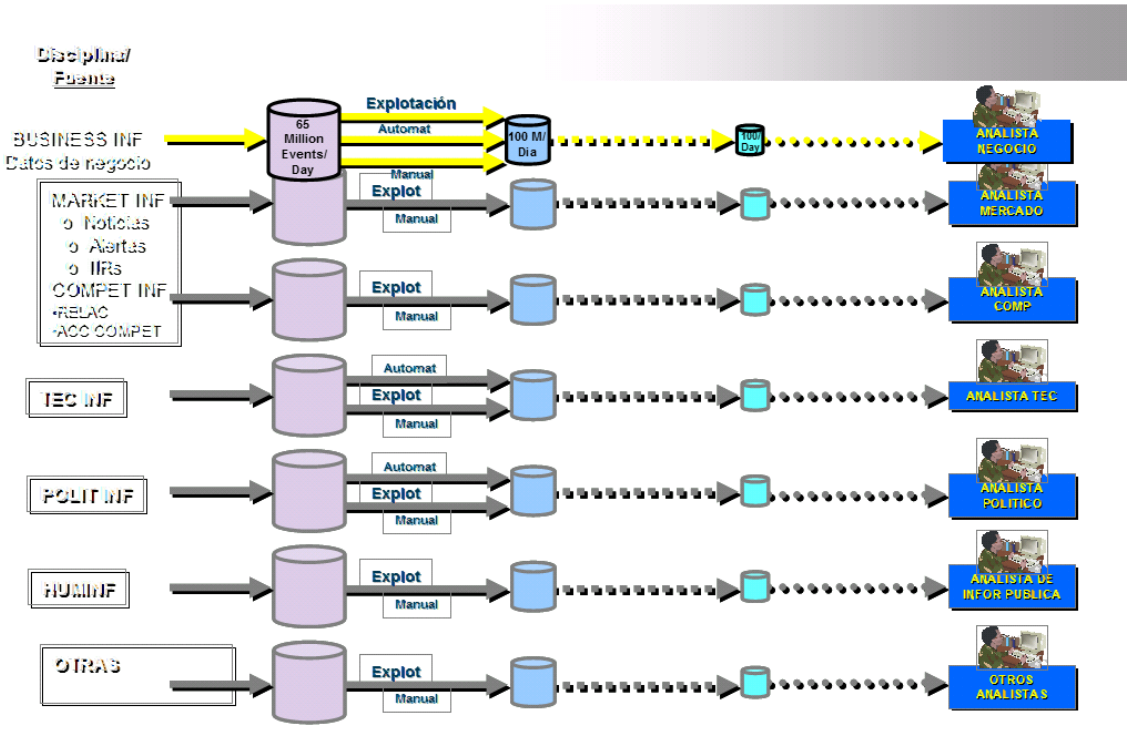
Esto quiere decir que, actualmente en las organizaciones, existe un volumen de datos de tal envergadura que recopilar todos los datos necesarios para poder prestar un soporte eficiente a la toma de decisiones se torna imposible físicamente, ya que excede las capacidades humanas o, cuando la relación coste beneficio es excesiva, aunque de modo teórico el proceso fuese factible.

Además de los datos aislados, **las relaciones entre ellos y las interrelaciones entre planos informacionales distintos**, lleva a que los directivos a la hora de tomar una decisión se encuentren incapacitados para estimar todos los datos e informaciones generadas por otros individuos y, en muchos casos, el propio proceso de investigación se realiza en base a datos insuficientes, ya que los profesionales individuales no están capacitados para asumir esa labor.

Parece lógico que, en base a un planteamiento convencional de business intelligence, sean **muchos los elementos informativos que se escapan del control y del análisis**, por ejemplo, las relaciones entre individuos, las relaciones entre elementos organizacionales complejos, etc.

Para posibilitar el análisis convencional de la información y datos en una organización moderna sería necesario contar con analistas expertos en inteligencia de negocio pero, de igual modo, analistas políticos, de información tecnológica, etc., que equilibren sus tareas entre en análisis manual de información y la destilación automática o semiautomática de otras informaciones.

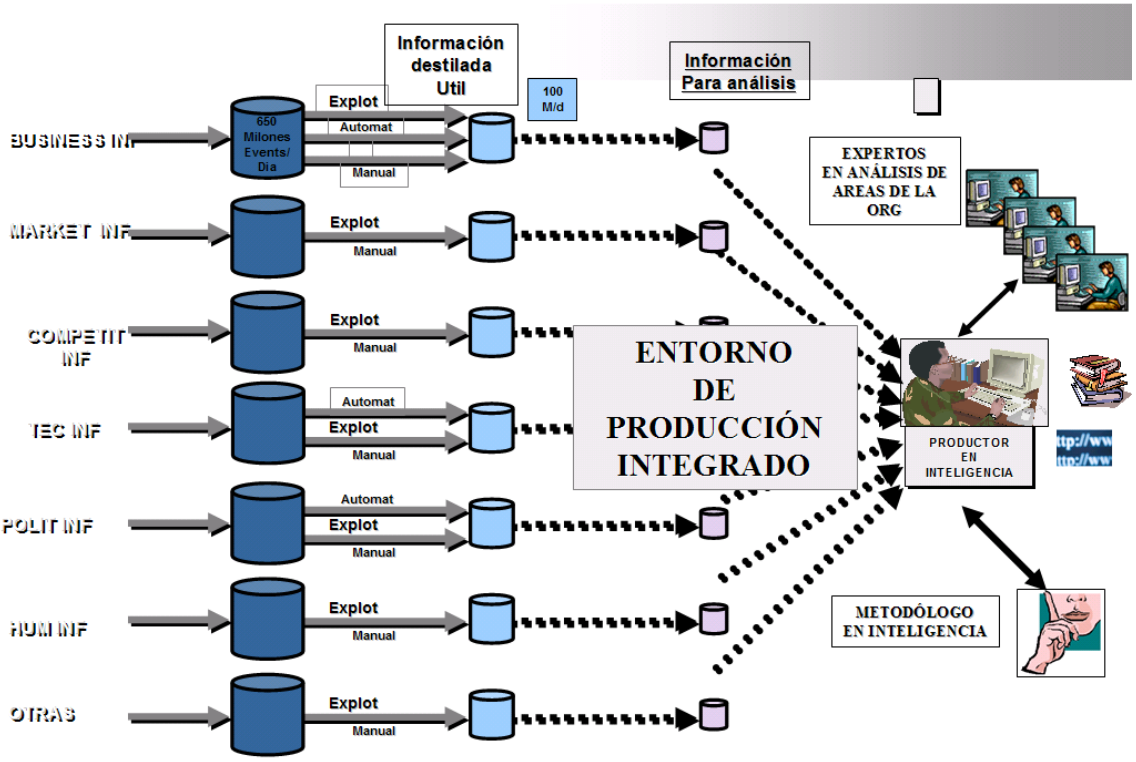
Estudios empíricos han demostrado que **una organización de tipo medio puede generar más de 65 millones de eventos al día** que deben ser registrados para hacer un análisis de la cesta de la compra, por ejemplo, o multitud de proyectos de análisis de tráfico web generan 100 millones de eventos al día según un documento del departamento de ciencias de la computación de la danesa Universidad de Aalborg titulado como, "Supporting imprecision in multidimensional databases using granularities".




Ámbitos de análisis
Fuente: Elaboración propia

El enfoque multidisciplinar es la clave para afrontar una situación de conocimiento determinada. Una propuesta adecuada para las empresas actuales consiste en aprovechar el propio valor de los analistas y científicos de datos, cuyas capacidades intelectuales, su conocimiento tácito acumulado y la ayuda de tecnologías de la información, continuamente actualizadas tras un proceso de vigilancia tecnológica, pongan el acento sobre el análisis y la explotación más que sobre la obtención de los datos.

Por consiguiente, el objeto de la inteligencia es determinar con rigurosidad el análisis de los temas que proyectan el futuro, aproximándose con objetividad e imparcialidad a la certeza. Por este motivo, la inteligencia se nutre "de" y "con" informaciones, pero no es una simple sumatoria de las mismas, como se propone desde ciertos ámbitos.




Entorno de análisis integrado
Fuente: Elaboración propia

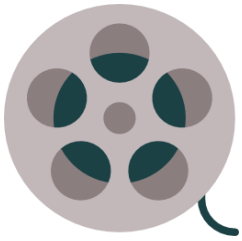
### 2.2.1.1 Críticas al BI

Ciertos autores consideran que el business intelligence o inteligencia de negocio es insuficiente respuesta a la búsqueda de la productividad de los datos.

Estos autores critican a los fabricantes de software, porque consideran que llaman eufemísticamente business intelligence a procesos simples pero costosos de almacenamiento, recuperación y sumarización de datos -que no información- y a otros procesos de descubrimiento de patrones de difícil interpretación y más dudosa utilidad. También opinan que estas informaciones (datos puestos en contexto e importancia) deben ser sometidas a un riguroso y complejo proceso, el cual, comprende la combinación de distintas técnicas y metodologías de análisis, para obtener un conocimiento diferente a sus componentes originales.

Para conocer un poco más de la crítica que recibe el BI, sus detractores indican que cabría aclarar que una base de datos, por copiosa y compleja que sea, no es inteligencia.

Para que ella se transforme en inteligencia tiene que estar organizada para una tarea, dirigida a una actuación específica, enmarcada en una planificación estratégica y orientada a una decisión o a un proceso específico. Las bases de datos pueden ser o tener infinitas respuestas, pero no hacen preguntas, y en inteligencia lo más importante es saber preguntar. Esto último, preguntar, es la guía del marco metodológico de investigación y descubrimiento, del proceso de producción de inteligencia.



**VIDEO DE INTERÉS**

En este enlace de vídeo podrás ver cuáles son las ventajas y los beneficios del business intelligence que obtienen las empresas.

https://www.youtube.com/watch?v=S5Icnhe6qYc

### 2.2.1.2 Clasificación de la inteligencia

En cuanto a los niveles de inteligencia podemos hablar de:

- **Inteligencia estratégica:** se trata de la inteligencia aplicable para la toma de decisiones que tienen que ver directamente con los objetivos de la compañía.
- **Inteligencia táctica:** en este caso, nos estamos refiriendo a la inteligencia relacionada con los medios que se aplican para alcanzar los objetivos.
- **Inteligencia operacional:** este tipo de inteligencia se utiliza para el desarrollo o ejecución de distintas acciones.

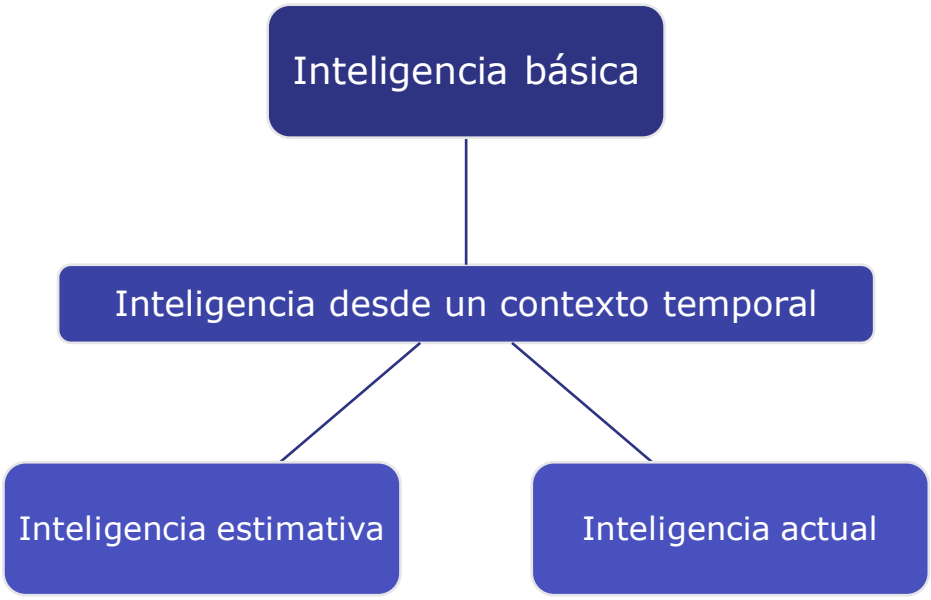

Niveles de inteligencia
Fuente: Elaboración propia

Desde un contexto temporal, la inteligencia se divide en básica, actual y estimativa. Esta categorización de la inteligencia se basa en el instante de uso de la misma:

- **Inteligencia básica:** tiene un carácter general y es de permanencia en el tiempo. Satisface unas necesidades generales y estratégicas de la organización y es, por ello, que intenta conocer todos los aspectos de cada uno de los objetivos fijados para proporcionar respuestas para las consultas de información concretas. En inteligencia empresarial, podemos estar hablando de los históricos de performance de la banca que, hasta la llegada de internet y el impacto del multicanal, podían llegar a ser válidos durante décadas.

- **Inteligencia actual:** es la que se utiliza para actualizar con nueva información a toda aquella inteligencia que queda englobada dentro de la inteligencia básica. La inteligencia actual se suele mostrar como boletines periódicos de actualización. Asimismo, la inteligencia actual cumple la función de dar respuesta rápida a necesidades informativas puntuales en el presente. Sobre una petición de información al respecto de una cuestión puntual se inicia un proceso de investigación bien con los datos en poder o mediante el lanzamiento de un proceso de búsqueda, que permita prestar soporte a la toma de decisiones.

- **Inteligencia estimativa:** es la encargada de fijar, ante una circunstancia específica de una pregunta de un directivo o persona que está encargada de tomar las decisiones, la probable transformación de la situación y las opciones de intervención de los elementos involucrados en la misma para facilitar la predicción y las posteriores elecciones.




Categorización de la inteligencia en función del instante en el que se usa
Fuente: Elaboración propia

### 2.2.1.3 Características de la inteligencia

Es también importante destacar las características que distinguen a un buen enfoque de inteligencia:

- **Instrumental:** la inteligencia no aporta nada por sí sola. Únicamente se puede considerar valiosa cuando está destinada a que alguien la utilice, de lo contrario, si nadie llega a consumirla, podría decirse que no ha servido para nada su elaboración.

- **Objetiva:** la función de un sistema de inteligencia es poder hacer llegar a los directivos responsables de tomar las decisiones sobre un aspecto determinado la información adquirida gracias a dicho sistema. La inteligencia no pretende transmitir a los encargados de la toma de decisiones lo que la información que les hubiese gustado recibir, tan solo pretende decir la verdad para que puedan actuar de la manera que consideren más adecuada.

- **Oportuna:** debe llegar cuando es necesaria, si lo hace demasiado tarde, no sirve para nada y puede provocar graves consecuencias para la organización.

- **Calidad:** el resultado final de un sistema de inteligencia debe guiar, de manera precisa, los procesos de toma de decisiones en una organización.

- **Humilde:** hay múltiples poseedores de conocimientos de alto nivel sobre una materia más allá de los sistemas de inteligencia. Es importante acatar y preguntar incesantemente a directivos, personal de la organización en general y académicos y técnicos independientes en los diferentes temas que deben atender.

#### 2.2.1.4 Evolución del uso de datos y la inteligencia de negocio

Sin duda, han sido muchos y variados los modelos que se han propuesto a lo largo del tiempo para el uso y gestión de los datos. Algunos de ellos han sido pequeños o grandes fracasos. Seguramente, el gran error ha sido partir de visiones muy simplistas, basadas en un tratamiento automatizado o meramente informática de los datos, sin tener en cuenta una visión estratégica, donde la información y la inteligencia es mucho más que eso.

Dichos modelos anteriores, además de fracasar por errores conceptuales de base y por su escasa fundamentación teórica, no han dado más de sí, seguramente por el aislamiento que han establecido con otras disciplinas, como la gestión del conocimiento, la estrategia o el management.

Uno de los autores más críticos con estos fracasos es un profesor argentino de nombre Piscitelli. El autor reconoce la necesidad de la información en las organizaciones y el equilibrio que se debe dar entre novedad y seguridad, entre información nueva y vieja y, sobre todo, da en el clavo cuando se refiere a los conocimientos cambiantes que son necesarios para la acción, es decir, la **información para la toma de decisiones**.

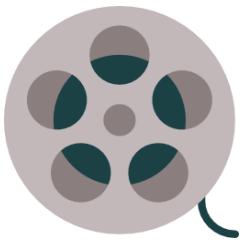
**¿SABÍAS QUE…?**

Alejandro Piscitelli es un consultor y profesor de la Universidad de Buenos Aires. Es una de las figuras más relevantes en todo lo relativo a los nuevos medios en todo el mundo.

De igual modo, el profesor Piscitelli descubre que las necesidades informativas de las organizaciones han cambiado al hilo del proceso de cambio social y empresarial de los últimos años y que lo que era considerado información tiempo atrás, hoy se encuentra devaluada sobremanera. Es decir, esos datos e informaciones consolidadas, internas a las organizaciones, substantivadas en la contabilidad y el proceso contable tienen ya escaso valor.

Según Piscitelli, ciertas innovaciones, como en balanced score card o cuadro de mando integral, mejoran esa entrada de datos, pero siempre desde un punto de vista interno. Es lo que el mismo autor ha denominado neurosis informacional. Esto provoca que muchos gestores consideren que la gestión empresarial actual es más precisa y eficiente, ya que se puede controlar al milímetro lo que ocurre en una organización y en sus integrantes, algo que finalmente no es así, ya que solo se posterga el conocimiento de la realidad.

Los sistemas de información que dan respuesta a estos modelos de gestión, por tanto, ofrecen de forma diversa información que, en realidad, ya conocemos. Para Piscitelli, a las organizaciones les debería preocupar más conocer todo lo posible sobre sus no-clientes en vez de saber lo que ya conocen sobre sus clientes.



**VIDEO DE INTERÉS**

A continuación podrás ver una charla TED del profesor Piscitelli en la que habla sobre las múltiples cuestiones interesantes de la cultura digital.

https://www.youtube.com/watch?v=MCAsC6OIJGg

Por una parte, cierto tipo de información ha perdido valor y se ha commoditizado y, otro tipo de información, sin embargo, cada día es más demandada en las organizaciones.

Esto se explica en gran medida porque el modo en que las empresas deben ser gestionadas ha cambiado radicalmente. El management puede ser considerado ya una ciencia social, aunque en su aplicación siga siendo un arte. El corpus doctrinal de la gestión tiene una solidez ya comparable al de otras ciencias, con aportaciones muy destacables en los campos de las relaciones humanas, la producción, la calidad o los sistemas de información.

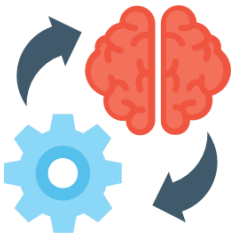
La situación de los mercados se ha trastocado y nos encontramos en un mercado donde el exceso de oferta es más claro que nunca y los ciclos de vida de los productos son cada vez más cortos. Las comunicaciones y las tecnologías de la información han facilitado el acceso a contenidos que, hasta hace escasas fechas, eran considerados de alto valor e inaccesibles para organizaciones con escaso poder.

Dentro de este último aspecto, hay que destacar el altísimo impacto que ha tenido internet en cuanto a las modificaciones en el modo en que se realizan las transacciones, los intercambios de valor, el proceso decisional de compra, la prestación de servicios, etc. Y, dentro de los cambios generados por internet, el impacto del multicanal ha sido especialmente relevante y es ahí donde la inteligencia tiene una función más importante y útil.

Hasta la fecha, casi cualquier organización comercial era capaz de predecir su futuro a corto y medio plazo en base a los indicadores históricos de rendimiento. Con el impacto del multicanal y la aparición de productos con una vida de duración extremadamente corta, incluso de horas, es imposible para estas organizaciones conocer cuál es la rentabilidad de dichos productos más allá de la intuición.

Otros muchos supuestos de gestión han sido desplazados por estos cambios: las estructuras arcaicas de organización, las segmentaciones estrictas de mercados en base a categorías o zonas geográficas de ventas y, en general, cualquier otra dinámica que se quiera incluir bajo los grandes epígrafes de globalización, liberalismo económico, sociedad de la información y el conocimiento, etc.

Por lo tanto, la inteligencia moderna debe ser entendida como una actividad multidisciplinaria, compleja, dinámica. Sin duda, es una actividad imprescindible en un mundo donde, conocer con precisión la realidad y aprovechar las oportunidades de futuro, son factores críticos de éxito.



**RECUERDA**

La inteligencia es la capacidad de enlazar, de encontrar nexos entre las cosas y destapar atributos subyacentes a la variedad. Se trata del conocimiento hallado a través del procesamiento preciso de la información, que se proporciona a los directivos de una organización sobre un asunto determinado.

Desde un punto de vista más operacional, la inteligencia se podría entender como el producto del compendio, valoración e interpretación de la información disponible, de inminente o latente envergadura para alcanzar un fin determinado. Y es, asimismo, el procedimiento lógico racional que se le aplica a los datos para transformarlos en instrumento útil en la toma de decisiones.

En relación con el proceso decisional, la principal dificultad que se puede encontrar en el marco de la incertidumbre es la imposibilidad de que la inteligencia alcance la convicción total. La disminución de la incertidumbre estará determinada por la precisión de la apreciación que haga el analista acerca de la situación que enfrenta, de los intereses del directivo que va a tomar la decisión y de los intereses de otros actores que interactúan.

Inteligencia moderna
Fuente: https://www.freepik.es/vector-gratis/sistema-nervioso-humano_4239572.htm#page=1&query=inteligencia%20moderna&position=17

## 2.3 Tratamiento del dato

Para hablar de los posibles tratamientos que podemos aplicar a los datos, vamos a utilizar como herramienta de trabajo Scala. Los motivos por los que Scala es una buena opción de aprendizaje los presentamos a continuación.

### 2.3.1 Introducción a Scala

Scala es uno de los lenguajes de programación modernos más populares. Es, por su alto rendimiento, considerado por algunos de sus usuarios como el Ferrari de los lenguajes de programación. No solo es muy potente, sino también es un lenguaje "bonito" en palabras de algunos de sus "fans". Aprender Scala en el momento actual proporciona un impulso a la carrera profesional, porque es un sector de la informática incipiente, donde todavía no es mucha la competencia existente entre profesionales.

Las aplicaciones de big data suelen estar escritas en varios lenguajes de programación, incluidos Java, Python, C++, Scala y otros. Hadoop, por ejemplo, está escrito en Java. La mayoría de las aplicaciones de Hadoop están también programadas en Java, pero igualmente es compatible con otros lenguajes de programación. Del mismo modo, Spark está escrito en Scala, pero admite múltiples lenguajes de programación, incluidos Scala, Java Python y R.

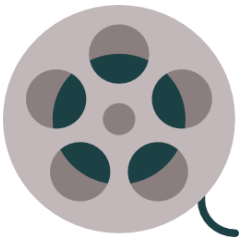
Scala es un gran lenguaje para desarrollar aplicaciones de big data. Proporciona una serie de beneficios. Primero, un desarrollador puede lograr un salto significativo en la productividad utilizando Scala. Segundo, ayuda a los desarrolladores a escribir códigos robustos con pocos errores. Tercero, Spark está escrito en Scala, por lo que Scala es una opción natural para desarrollar aplicaciones sobre el stack de Spark.

Ahora vamos a presentar Scala desde la perspectiva de un lenguaje de programación de propósito general. El objetivo no es convertirnos de la noche a la mañana en un experto en Scala, pero si saber ponerlo en marcha y escribir algunos códigos básicos. Este apartado también tiene como objetivo presentar los conceptos teóricos de Scala para comprender mejor como ha sido desarrollado Spark.

**ARTÍCULO DE INTERÉS**

En el siguiente artículo podrás leer una explicación acerca del origen de Scala.

https://recluit.com/un-vistazo-a-la-historia-de-scala/#.XwnhLsfVJPY



**VIDEO DE INTERÉS**

En este vídeo verás la experiencia en primera persona de cómo un desarrollador migró desde su experiencia en Java a Scala y como se adaptó al cambio de paradigma y plataforma.

https://www.youtube.com/watch?v=r0iStjRB5SM

### *2.3.2 Programación funcional (functional programming-FP)*

La programación funcional es un estilo o paradigma de la programación que usa funciones como bloques de construcción y evita variables mutables, bucles y otras estructuras de control imperativas. Trata la computación como una evaluación de funciones matemáticas, donde la salida de una función depende solo de los argumentos de esta, de modo que un programa cualquiera estará compuesto de tales funciones.

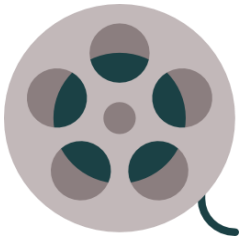
La programación funcional ha atraído mucha atención en los últimos años. Incluso los principales lenguajes y más difundidos, como C++, Java o Python, han añadido soporte para programación funcional. Este paradigma se ha vuelto muy popular por algunas buenas razones:

- Primero, la programación funcional proporciona un tremendo impulso a la productividad del desarrollador. Permite resolver un problema con menos líneas de código en comparación con los lenguajes imperativos. Por ejemplo, una tarea que requiere 100 líneas de código en Java puede requerir solamente 10 o 20 líneas de código en Scala. Por lo tanto, la tarea de la programación puede aumentar su productividad de cinco a diez veces.

- En segundo lugar, la programación funcional facilita la escritura de aplicaciones concurrentes o multiproceso. La capacidad de escribir aplicaciones multiproceso se ha vuelto muy importante con el advenimiento de servidores de múltiples CPU o de ordenadores de tipo multinúcleo. Mantenerse al día con la ley de Moore se vuelve cada vez más difícil para los fabricantes de hardware, así que, en lugar de acelerar los procesadores, comenzaron a agregar más CPU y núcleos. Los equipos multinúcleo se han vuelto comunes hoy en día y las aplicaciones deben aprovechar todos los núcleos, porque lo contrario sería un desperdicio de los recursos disponibles.

- Tercero, la programación funcional nos ayuda a escribir código robusto y de calidad. Nos hace sencillo evitar los errores más comunes de programación. Además, el número de errores en una aplicación es proporcional a las líneas de código. Dado que la programación funcional requiere muchas menos líneas de código en comparación con la programación tradicional, se introducen menos errores en el código.

- Finalmente, los lenguajes de programación funcionales hacen que sea más fácil escribir código elegante, que es fácil de leer, entender y razonar sobre él. Un código funcional correctamente escrito podemos decir incluso que es más bonito, no es complejo ni introduce distorsiones o ruido en su interpretación. La comunidad de desarrolladores en lenguajes funcionales que obtienen inmensas satisfacciones con su código es creciente.



**VIDEO DE INTERÉS**

A través de este vídeo podrás entender mejor la programación funcional explicada por el desarrollador Rodrigo de Frutos.

https://www.youtube.com/watch?v=hzy4_K25h3U

### *2.3.3 Funciones*

Una función es un bloque de código ejecutable. Permiten a un programador dividir un programa grande en pequeñas piezas manejables. En la programación funcional, una aplicación se construye completamente ensamblando funciones.

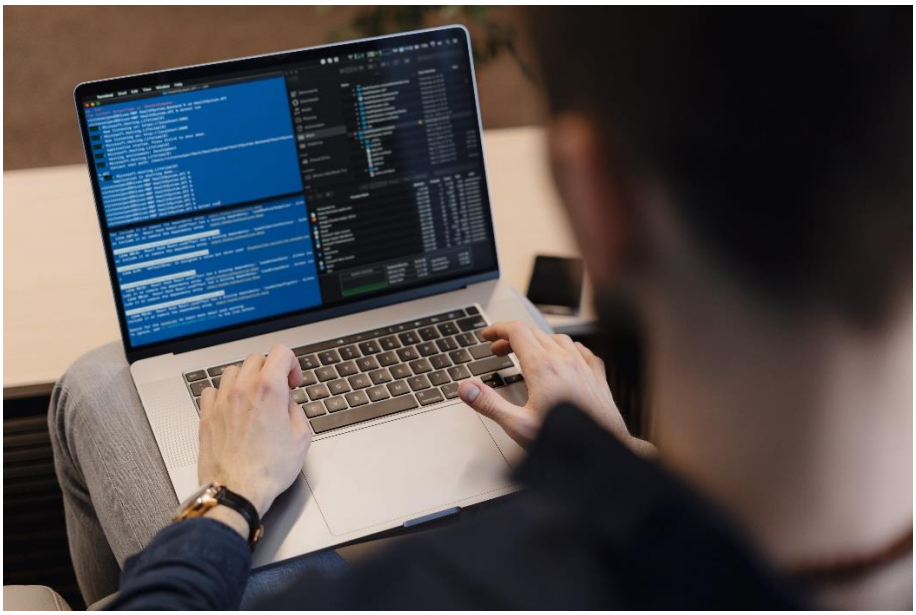
Aunque muchos lenguajes de programación soportan el concepto de funciones, bien de forma nativa o adaptándose a ello, en la programación funcional, los lenguajes tratan las funciones como un "ciudadano de primera clase" desde sus orígenes.

### *2.3.4 Primera clase*

Como decíamos, la programación funcional trata a las funciones como ciudadanos de primera clase. Una función tiene el mismo estado que una variable o valor. Es más fácil entender este concepto si se comparan las funciones en programación funcional con las funciones en lenguajes imperativos como C.

Los lenguajes imperativos tratan las variables y las funciones de manera diferente. Por ejemplo, C no permite que una función pueda ser definida dentro de otra función. No permite tampoco que se pase una función como parámetro de entrada a otra función.

La programación funcional sí que permite que una función se pase como entrada a otra función. Habilita igualmente que una función se devuelva como valor de retorno de otra función. Una función se puede definir en cualquier lugar, incluso dentro de otra función. Se puede definir como un literal de función sin nombre, al igual que un literal de cadena y pasarla como entrada a una función.

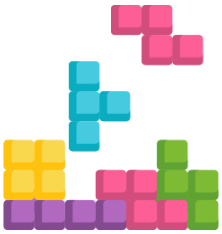


Programación
Fuente: https://www.freepik.es/foto-gratis/cerrar-hombre-escribiendo-codigo-computadora-portatil_7621115.htm#query=codigo&position=21

### *2.3.5 Simples*

Las funciones en la programación funcional son simples. Una función consta de unas pocas líneas de código y solo hace una cosa. Una función simple es fácil de razonar y de entender su contexto en el negocio. Por lo tanto, permite una mayor robustez, resilencia y alta calidad en el software codificado.

La composición de funciones simples facilita la implementación de algoritmos complejos con confianza.



**EJEMPLO PRÁCTICO**

Debes realizar un programa que resuelva un problema complejo, sin embargo, no eres capaz de implementar una función que funcione correctamente. ¿Qué podrías hacer?

**SOLUCIÓN:**

La programación funcional alienta la división recursiva de un problema en subproblemas, hasta que se pueda resolver ese subproblema con una simple función.

Las funciones simples se pueden ensamblar hacia atrás para formar una función que resuelva un problema complejo.

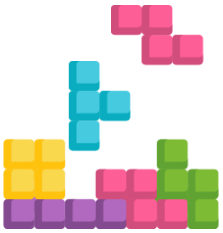
### *2.3.6 Estructura de datos inmutable*

La programación funcional enfatiza el uso de estructuras de datos inmutables. Un programa puramente funcional no utiliza ninguna estructura de datos mutable o variable. En otras palabras, los datos nunca se modifican en su ubicación, a diferencia de los lenguajes de programación imperativos como C o C++, Java y Python. A los desarrolladores o ingenieros de software sin experiencia previa en este tipo de programación les resulta difícil imaginar un programa sin variables mutables. En la práctica, no es difícil escribir código con estructuras de datos inmutables una vez que nos acostumbramos a ello.

Las estructuras de datos inmutables ofrecen varios beneficios:

- Primero, reducen los errores. Es fácil razonar sobre un código escrito con estructuras de datos inmutables. Además, los lenguajes funcionales proporcionan construcciones que permiten que un compilador imponga la inmutabilidad. Por lo tanto, muchos errores se detectan en tiempo de compilación.

- En segundo lugar, las estructuras de datos inmutables facilitan la escritura de aplicaciones de subprocesos múltiples. Escribir una aplicación que utiliza todos los núcleos no es una tarea fácil. Las condiciones de carrera y la corrupción de datos son problemas frecuentes que aparecen con aplicaciones multiproceso. El uso de estructuras de datos inmutables ayuda a evitar estos problemas.



**EJEMPLO PRÁCTICO**

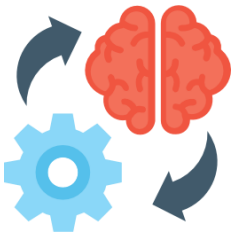
Si estamos trabajando en Java y queremos hacer inmutable la clase Alumno que indicamos a continuación, ¿cómo lo haríamos?

```
public class Alumno {
    public String nombre;
    public String apellido;

    public Alumno (String nombre, String apellido) {
        this.nombre = nombre;
        this.apellido = apellido;
    }
}
```

**SOLUCIÓN:**

Para convertir en inmutable una clase debemos añadir la palabra final, de forma que el código quedaría de la siguiente manera:

```
Public final class Alumno {
    Public final String nombre;
    Public final String apellido;

    public Alumno (String nombre, String apellido) {
        this.nombre = nombre;
        this.apellido = apellido;
    }
}
```



**RECUERDA**

En la programación funcional, cada declaración es una expresión que devuelve un valor.

Por ejemplo, la estructura de control if-else, en Scala es una expresión que devuelve un valor. Este comportamiento es diferente de otros lenguajes tradicionales, donde puedes agrupar una gran cantidad de declaraciones dentro de un if-else.

Esta característica es útil para escribir aplicaciones sin variables mutables.

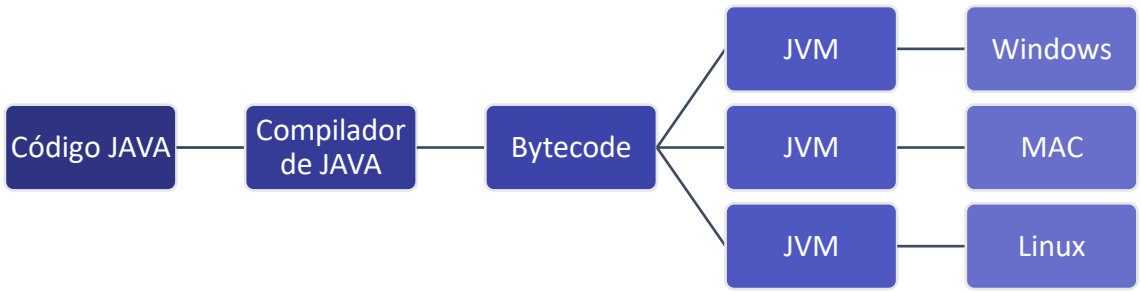
### 2.3.7 Fundamentos de Scala

Scala es un lenguaje de programación híbrido, que admite programación funcional y orientada a objetos. En ella están presentes conceptos de programación funcional, como estructuras de datos inmutables y funciones de primera clase. Para la programación orientada a objetos, también admite conceptos como clase, objeto y rasgo. También están disponibles las propiedades como encapsulación, herencia, polimorfismo y otras características importantes en la orientación a objetos.

Es un **lenguaje estáticamente escrito**. El compilador Scala reúne una aplicación Scala siendo un tipo de lenguaje seguro, por lo que impone la seguridad de tipos en tiempo de compilación. Esto ayuda a reducir la cantidad de errores en una aplicación.

Finalmente, diremos que Scala es un lenguaje basado en la máquina virtual de Java (JVM), por esto es muy sencillo a la hora de programar el mezclar los stacks de Scala y de Java.

El compilador Scala compila dicha aplicación en código de bytes (bytecode) Java, que se ejecutará en cualquier JVM, ya que está basado en él. En el nivel de bytecode, una aplicación Scala es indistinguible de una aplicación Java.




Funcionamiento de la Java Virtual Machine y bytecode
Fuente: Elaboración propia

Una biblioteca Scala se puede usar fácilmente desde una aplicación Java. Más importante aún, una aplicación Scala puede usar cualquier biblioteca Java sin ningún contenedor o código de pegamento. Por lo tanto, las aplicaciones Scala se benefician de la gran variedad de bibliotecas de código Java existentes y que las empresas y programadores han desarrollado en las últimas dos décadas de presencia masiva de Java en el mercado.

Podemos decir que es un programa que enfatiza la parte de programación funcional, eso es lo que lo convierte en un lenguaje tan poderoso y, por lo cual, obtendremos mayores beneficios al usarlo. Si por lo contrario, queremos aprovechar Scala como un lenguaje de programación orientado a objetos, tampoco diferirá mucho de otros lenguajes de programación de dicho tipo.

Profundizar en él requeriría de un curso específico para cubrirlo en detalle, pero las construcciones fundamentales necesarias para escribir una aplicación Spark tampoco son muy complejas de controlar. Además, contando con algo de experiencia en programación, y conociendo los conceptos básicos de ingeniería del software y algoritmia, no tiene por qué hacerse cuesta arriba.

Scala, como otros lenguajes potentes, implica también cierta complejidad. Algunos novatos en su uso se sienten intimidados con este lenguaje, porque intentan aprender todas las características a la vez. Sin embargo, no es imprescindible para usarlo de manera efectiva. Podemos comenzar a desarrollar aplicaciones Scala de forma productiva una vez que aprendemos los fundamentos.

### *2.3.8 Empezando con Scala*

La mejor manera de aprender un lenguaje de programación siempre ha sido programando. Así que en cuanto seas capaz de entender el contenido presentado a continuación, te animamos a que pruebes los ejemplos de código que vayas viendo.

Podemos escribir código Scala en cualquier editor de texto, para después compilarlo y ejecutarlo. Alternativamente, para mayor comodidad y productividad, podemos usar el IDE basado en navegador proporcionado por Typesafe, actualmente parte de Lightbend.

También podríamos usar el IDE de Scala basado en Eclipse, IntelliJ IDEA o NetBeans IDE.

**ENLACE DE INTERÉS**

En este enlace podrás descargar los ejecutables de Scala y Typesafe Activator, y también encontrarás enlaces para descargar Scala basado en el IDE Eclipse, IntelliJ IDEA o NetBean.

www.scala-lang.org/download

**BIBLIOGRAFÍA RECOMENDADA**

Algunos ejemplos de códigos muy interesantes que aparecen es este apartado se encuentran en el libro de Mohammerd Guller, titulado "Big Data Analytics With Spark" (2016).

La manera más fácil de comenzar a usar Scala es usar el intérprete de Scala, que proporciona una shell para escribir el código Scala. Es una herramienta REPL (lectura, evaluación, impresión, bucle). Cuando escribes una expresión en el shell de esta, evalúa la expresión, imprime el resultado en la consola y espera por la siguiente expresión. Instalar el shell interactivo de Scala es tan simple como descargar los binarios y desempaquetarlo. El shell se llama Scala y se encuentra en el directorio bin. Se inicia escribiendo scala en la terminal.

```
$ cd /path/to/scala-binaries
$ bin/scala
```

En este punto, deberías ver el indicador del Shell de Scala, como se muestra en la imagen.

```
Welcome to Scala version 2.11.7 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_67).
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Indicador del Shell de Scala
Fuente: Elaboración propia

Ahora ya puedes escribir cualquier expresión Scala. A continuación, se muestra un ejemplo:

```
scala> println("hello world")
```

Después de presionar la tecla enter, el intérprete de Scala evalúa el código e imprime el resultado en la consola. Podemos usar esta shell de forma sencilla para jugar con los ejemplos de código que se muestran en este apartado.

Al igual que otros lenguajes de programación, Scala viene pre empaquetado con una lista de tipos básicos y operaciones permitidas sobre esos tipos.

Hay que tener en cuenta que Scala no tiene tipos primitivos. Cada tipo en Scala se implementa como una clase. Cuando una aplicación Scala se compila a bytecode de Java, el compilador convierte automáticamente los tipos de Scala a tipos primitivos de Java, siempre que sea posible, para optimizar el rendimiento de la aplicación.

| | |
|---|---|
| **Byte** | 8 bits con signo complemento entero. Rango de valores entre -128 a la 127 |
| **Short** | 16 bits con signo complemento entero. Rango de valores -32.768-32767 |
| **Int** | De 32 bits con signo complemento entero. Rango de valores desde -2147483648 hasta 2147483647 |
| **Long** | De 64 bits con signo complemento entero. intervalo numérico -92233720368547758O8-9223372036854775807 |
| **Float** | 32-bit de punto flotante de precisión simple IEEE754 |
| **Double** | 64-bit de punto flotante de precisión simple IEEE754 |
| **Char** | 16 sin signo de caracteres Unicode U + 0000 a un rango de valores U + FFFF |
| **String** | Secuencia de caracteres |
| **Boolean** | Verdadero o falso |

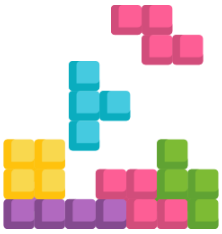
Tipos básicos en Scala
Fuente: Elaboración propia

### 2.3.9 Variables

Scala tiene dos tipos de variables: mutables e inmutables. Insistimos en que se desaconseja el uso de variables mutables. Un programa funcional puro nunca usaría una variable mutable. Sin embargo, a veces el uso de variables mutables puede ser un atajo, es decir, puede dar como resultado un código menos complejo, por lo que Scala también admite variables mutables. Aunque deben ser utilizadas con precaución y siendo conscientes del impacto.

Una variable mutable se declara utilizando la palabra clave var, es similar a una variable en lenguajes imperativos, como C o C++ y Java. Se puede reasignar después de haber sido creada.

Mientras que una variable inmutable se declara usando la palabra clave val no se puede reasignar después de haber sido inicializada.



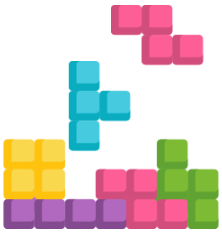
**EJEMPLO PRÁCTICO**

En un programa que estás desarrollando necesitas crear una variable mutable, asignarle un valor y, posteriormente, modificar dicho valor. ¿Cómo lo harías?

**SOLUCIÓN:**

La sintaxis para crear y modificar una variable se muestra a continuación donde inicialmente asignamos 10 como valor de x, para después reasignarlo a 20.

```
var x = 10
x = 20
```



**EJEMPLO PRÁCTICO**

En un programa que estás desarrollando necesitas crear una variable inmutable y asignarle un valor.

La sintaxis para crear un val se muestra a continuación.
val y = 10

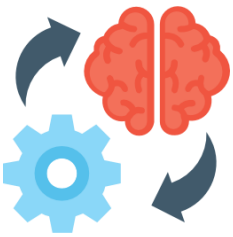
¿Qué sucede si, más adelante, en el programa, se añade la siguiente declaración?
y = 20

**SOLUCIÓN:**

Pues que el compilador generará un error.

Es importante señalar algunas facilidades que nos ofrece el compilador Scala. Primero, si usamos punto y coma en el final de una declaración, significa que es opcional.

En segundo lugar, el compilador infiere el tipo siempre que sea posible. Scala es un lenguaje de tipificado estático, por lo que todo tiene un tipo. Sin embargo, el compilador Scala no obliga al desarrollador a declarar el tipo de algo si puede inferirlo. Por lo tanto, la codificación en Scala requiere menos declaración de tipos y el código resulta más sencillo.



**RECUERDA**

Las siguientes dos declaraciones son equivalentes.

val y: Int = 10;
val y = 10

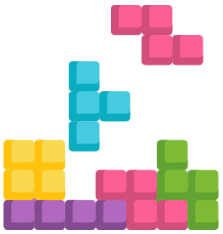
### *2.3.10 Funciones*

Como se mencionó anteriormente, una función es un bloque de código ejecutable que devuelve un valor. Es conceptualmente similar a una función en matemáticas, que toma entradas y devuelve una salida.

Una función se puede usar como una variable. Por lo que se puede pasar como una entrada a otra función. Se puede definir como una función literal sin nombre, como ocurre con un literal de cadena. Puede ser asignado a una variable. Se puede definir dentro de otra función y se puede devolver como salida de otra función.

Una función en Scala se define con la palabra clave def. Una definición de función comienza con el nombre de la función, seguido de los parámetros de entrada separados por comas y entre paréntesis junto con sus tipos. El paréntesis de cierre va seguido de dos puntos, tipo de salida de la función, signo igual y el cuerpo de la función entre llaves opcionales.

En cuanto a la función add, esta toma dos parámetros de entrada, los cuales son de tipo Int. Devuelve un valor, también del tipo Int. Esta función simplemente añade sus dos parámetros de entrada y devuelve la suma como salida.



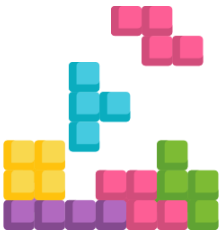
**EJEMPLO PRÁCTICO**

En un programa informático que estás creando tienes la necesidad de disponer de una función que realice la suma de dos sumandos. ¿Cómo lo harías?

**SOLUCIÓN:**

Para obtener dicha funcionalidad puedes codificar una función análoga a la que sigue sin mucho esfuerzo, porque aplicarás los mismos criterios de la programación declarativa:

```
def add(firstInput: Int, secondInput: Int): Int = {
val sum = firstInput + secondInput
return sum
}
```



**EJEMPLO PRÁCTICO**

En un programa informático que estás creando tienes la necesidad de disponer de una función que realice la suma de dos sumandos, p ero quieres aprovechar al máximo las capacidades de la programación funcional. ¿Cómo lo harías?

**SOLUCIÓN:**

Para obtener dicha funcionalidad se puede codificar una función en Scala pero en una versión mucho más concisa de la misma función en formato declarativo como se muestra a continuación.

```
def add(firstInput: Int, secondInput: Int) = firstInput + secondInput
```

La segunda versión hace exactamente lo mismo que la primera versión. El tipo de los datos devueltos es omitido, ya que el compilador puede inferirlo del código. Sin embargo, se recomienda no omitir la devolución de tipo de una función para mayor claridad. Las llaves también se omiten en esta versión. Solo son imprescindibles si el cuerpo de una función consiste de más de una declaración.

Además, la palabra clave return se omite, ya que es opcional. Todo en Scala es una expresión y devuelve un valor. El resultado de la última expresión, representada por la última declaración, en un cuerpo de función, se convierte en el valor de retorno de esa función.

El fragmento de código anterior es solo un ejemplo de cómo Scala permite escribir código conciso. Elimina código repetitivo y, por lo tanto, mejora la legibilidad y la facilidad de mantenimiento del código. Scala admite diferentes tipos de funciones que trataremos a continuación.

### *2.3.11 Métodos*

Un método es una función que es miembro de un objeto. Se define y funciona igual que una función. La única diferencia es que un método tiene acceso a todos los campos del objeto al que pertenece.

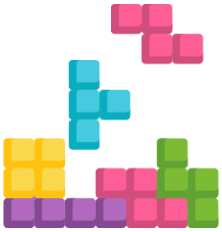
### *2.3.12 Funciones locales*

Una función definida dentro de otra función o método se denomina función local. Tiene acceso a las variables y parámetros de entrada de la función de cierre.

Una función local es visible solo dentro de la función en la que está definida. Esta es una característica útil que permite agrupar declaraciones dentro de una función sin contaminar el espacio de nombres de la aplicación.

### 2.3.13 Métodos de orden superior

Un método que toma una función como parámetro de entrada se llama método de orden superior. Del mismo modo, la función de orden superior es una función que toma otra función como entrada y, además, ayudan a reducir las duplicidades de código e incluso a escribir código sencillo.



**EJEMPLO PRÁCTICO**

Si es necesario crear una función de codificación que, por ejemplo, tome dos parámetros de entrada y devuelva un valor largo. ¿Cómo lo harías?

**SOLUCIÓN:**

El primer tipo de entrada es un Int. La segunda entrada es una función f que toma un Int como entrada y devuelve un Long.

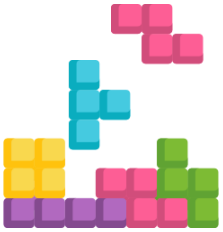
El cuerpo de la de la función multiplica la primera entrada por 10 y luego llama a la función que recibió como entrada, el cual, lo puedes realizar tal y como se observa en el siguiente ejemplo, que muestra una función simple de orden superior.

```
def encode(n: Int, f: (Int) => Long): Long = {
val x = n * 10
f(x)
}
```

### 2.3.14 Funciones literales

Una función literal es una función anónima o sin nombre en el código fuente. Se puede usar en una aplicación solo como una cadena literal. Se puede pasar como entrada a un método o función de orden superior. También se puede asignar a una variable.

Una función literal se define con parámetros de entrada entre paréntesis, seguido de una flecha hacia la derecha y el cuerpo de la función. El cuerpo de una función literal está encerrado entre llaves opcionales.



**EJEMPLO PRÁCTICO**

Nos encontramos que el cuerpo de la función consta de una sola declaración, pudiendo omitirse las llaves.

```
(x: Int) => {
x + 100
    }
```

Una versión concisa de la misma función literal se muestra a continuación.

```
(x: Int) => x + 100
```

La función de orden superior definida anteriormente (llamada encode) se puede usar con una función literal, ¿Cómo lo harías?

**SOLUCIÓN:**

```
val code = encode(10, (x: Int) => x + 100)
```

### *2.3.15 Clases*

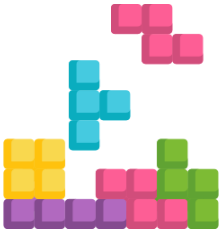
Una clase es un concepto de programación orientada a objetos. Proporciona una abstracción de programación de nivel superior. A un nivel muy básico, es una técnica de organización del código que permite agrupar datos y todas sus operaciones conjuntamente. Conceptualmente, representa una entidad con propiedades y comportamiento.

Una clase en Scala es similar a las de otros lenguajes orientados a objetos. Consiste en campos y métodos. Un campo es una variable que se utiliza para almacenar datos. Un método contiene código ejecutable. Es una función definida dentro de una clase. Un método tiene acceso a todos los campos de una clase.

Una clase es una plantilla o modelo para crear objetos en tiempo de ejecución. Se define en el código fuente usando la palabra clave class. Una definición de clase comienza con el nombre de la clase, seguido de los parámetros de clase separados por comas en paréntesis, y luego campos y métodos encerrados entre llaves. Se usa generalmente como una estructura de datos mutable. Puede tener campos que se definen usando `var`.

Un objeto es una instancia de una clase. Esta solo existe solamente en tiempo de ejecución. Tiene un estado que cambia con el tiempo.

Dado que Scala se ejecuta en la JVM, no es necesario eliminar explícitamente un objeto. El recolector de basura de Java elimina automáticamente los objetos que ya no están en uso.



**EJEMPLO PRÁCTICO**

Estamos programando y descubrimos que es necesario utilizar las clases siempre que las necesitemos en nuestros desarrollos, ¿cómo podríamos hacerlo?

**SOLUCIÓN:**

A continuación, se muestra un ejemplo de cómo utilizar clases siempre que las necesites en nuestros desarrollos:

```
class Coche(mk: String, ml: String, cr: String) {
val fabricante = mk
val modelo = ml
var color = cr
def repintado(newColor: String) = {
color = newColor
}
}
```

Se crea una instancia de una clase utilizando la palabra clave new.

```
val mustang = new Coche("Ford", "Mustang", "Rojo")
val corvette = new Coche("GM", "Corvette", "Negro")
```

### *2.3.16 Operadores*

Scala proporciona un amplio conjunto de operadores para los tipos básicos. Sin embargo, no tiene operadores integrados. En Scala, cada tipo es una clase y cada operador es un método. Usar un operador es equivalente a llamar a un método.

Veamos el siguiente ejemplo:

```
val x = 10
val y = 20
val z = x + y
```

+ no es un operador integrado en Scala. Es un método definido en la clase Int.

La última declaración en el código anterior es la misma que en el siguiente.

```
val z = x.+(y)
```

Scala permite llamar a cualquier método utilizando la notación de operador.

### *2.3.17 Tuplas*

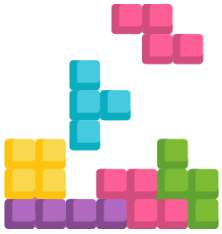
Una tupla es un contenedor para almacenar dos o más elementos de diferentes tipos. Es inmutable, por lo que no puede ser modificada después de que se haya creado. Tiene una sintaxis directa, como se muestra a continuación.

```
val dosElementos = ("10", true)
val tresElementos = ("10", "harry", true)
```

Es útil en situaciones en las que se desea agrupar elementos no relacionados. Si los elementos son del mismo tipo, puede usarse una colección, como una matriz o una lista. Si los elementos son de diferentes tipos pero relacionados, pueden almacenarse como campos en una clase. Sin embargo, usar una clase puede ser exagerado en algunas situaciones.

Por ejemplo, podemos tener una función que devuelve más de un valor. Una tupla puede ser más apropiada en estos casos.

Un elemento en una tupla tiene un índice.



**EJEMPLO PRÁCTICO**

Es necesario acceder a los elementos de una tupla, ¿cómo podríamos hacerlo?

**SOLUCIÓN:**

```
val primero = tresElementos._1
val segundo = tresElementos._2
val tercero = tresElementos._3
```

**¿SABÍAS QUE…?**

Los beneficios clave de usar Scala incluyen un salto significativo en la productividad del desarrollador y la calidad del código. Spark está escrito en Scala y es solo un ejemplo de los muchos sistemas distribuidos populares creados con Scala.

# 3. ALGUNOS CONCEPTOS TÉCNICOS DE LA ANALÍTICA TRADICIONAL

*El volumen tan grande de información que tienes que manejar te está orientando a utilizar una base de datos NoSQL. Además, la intención del proyecto es explotar otras fuentes de información, como por ejemplo las redes sociales, motivo por el cual todavía se hace más imprescindible apostar por alguna de las nuevas bases de datos que hay en el mercado.*

*Como la variedad es muy amplia, crees que tal vez sea conveniente, si el presupuesto y el tiempo disponible te lo permite, llevar a cabo algún ejemplo piloto que permita comparar cómo se comporta cada una de ellas respecto de las necesidades que se prevén.*

*En caso de no disponer de ese tiempo o presupuesto, sabes que la opción menos arriesgada será optar por el ecosistema de Hadoop, por ser sencillo para encontrar variedad de productos que hablan adecuadamente entre ellos y comparten una filosofía común prestando cobertura a las distintas necesidades que se vayan detectando.*

*Piensas además que, en caso de que en etapas posteriores se considere más adecuado optar por otras plataformas como Spark, el trabajo realizado hasta el momento no se perdería.*

*En este momento del proyecto tienes claro que MapReduce será necesario para facilitar el procesamiento de las grandes cantidades de datos que potencialmente se manejaran en el proyecto que estás ayudando a diseñar.*

## 3.1 El teorema de Brewer

El teorema de Brewer, que también se suele denominar como el teorema CAP y, en ocasiones, Conjetura de Brewer, dice que no hay ningún sistema de computación distribuida que sea capaz de garantizar al mismo tiempo las siguientes tres condiciones:

- **Consistencia** (Consistency), o lo que es lo mismo, que cualquier lectura recibe como respuesta la escritura más reciente o un error.

- **Disponibilidad** (Availability), o lo que es lo mismo, que cualquier petición recibe una respuesta no errónea, pero sin la garantía de que contenga la escritura más reciente.

- **Tolerancia al particionado** (Partition Tolerance), o lo que es lo mismo, que el sistema sigue funcionando si un número arbitrario de mensajes son descartados (o retrasados) entre nodos de la red.

Concretamente, Eric Brewer, cuando formuló su teorema, lo que quería expresar es que un sistema, como mucho, puede garantizar solo dos de las tres características simultáneamente y deberá sacrificar la restante.

**¿SABÍAS QUE...?**

Además de ser conocido por su teorema, Eric Brewer inventó el esquema para redes inalámbricas denominado WiLDNet, cuyo objetivo es mejorar la conexión a Internet en el Tercer Mundo.

## 3.2 Las nuevas bases de datos

El término NoSQL se usa para catalogar e incluir una amplia categoría de bases de datos modernas de tipo no relacional. Inicialmente, NoSQL significaba "sin compatibilidad con SQL", ya que estas bases de datos no admitían SQL. Sin embargo, ahora se tiende a interpretar que significa "no solo SQL", ya que algunas de estas bases de datos admiten un subconjunto de comandos SQL. Las bases de datos NoSQL tienen un diseño diferente de las bases de datos típicas de los sistemas RDBMS (sistemas gestores de bases de datos relacionales).

Una base de datos relacional garantiza el compromiso ACID (atomicidad, consistencia, aislamiento y durabilidad). Una base de datos NoSQL, por el contrario, sacrifica e intercambia el cumplimiento de ACID a cambio de una escalabilidad lineal, mayor rendimiento, alta disponibilidad, disponer de un esquema flexible y otras características que las hacen ventajosas en determinados escenarios.

### *3.2.1 Cassandra*

Cassandra es una base de datos NoSQL distribuida, escalable y tolerante a fallos diseñada para almacenar grandes conjuntos de datos. Es un almacén de filas particionado con consistencia ajustable. Una de sus características clave es el esquema dinámico. Cada fila puede almacenar diferentes columnas, a diferencia de las bases de datos relacionales, donde cada fila tiene exactamente las mismas columnas. Además, Cassandra está optimizada para tareas de escritura, por lo que las inserciones tienen un alto rendimiento.


Logo de Cassandra
Fuente: https://cassandra.apache.org

Tiene una arquitectura distribuida sin maestría, es decir, sin existencia de elementos maestro. Por lo tanto, no presenta un único punto de fallo, lo que la hace más resilente. Además, proporciona una distribución automática de filas en un clúster. Una aplicación de cliente que lee o escribe los datos puede conectarse a cualquier nodo en un clúster de Cassandra.

Ofrece alta disponibilidad a través del soporte integrado para la replicación de datos. El número de réplicas a guardar son configurables. Cada réplica se almacena en un nodo diferente en un clúster. Entonces, si el factor de replicación es tres, incluso si uno o dos nodos, fallan, el clúster todavía estará disponible y funcionará correctamente.

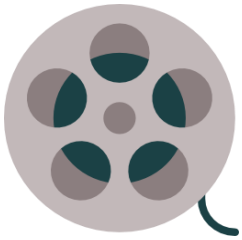
Los datos se modelan en ella utilizando una jerarquía de espacio de claves (keyspace), tablas, filas y columnas. Un keyspace es conceptualmente similar a una base de datos o esquema en un RDBMS. Es una colección lógica de tablas que representa un espacio de nombres. Está diseñado para controlar la replicación de datos para un conjunto de tablas. Una tabla, también conocida como familia de columnas, es parecida a una tabla en un RDBMS. Una familia de columnas consta de una colección de filas particionadas. Cada fila consta de una clave de partición y un conjunto de columnas. Por tanto, un espacio de claves, tablas, filas y columnas en Cassandra son similares a un esquema, tabla, fila y columna, respectivamente en una base de datos relacional, aunque su implementación y almacenamiento físico es diferente.

Los patrones de consulta guían los modelos de datos en Cassandra. Una familia de columnas o una tabla en Cassandra es básicamente una vista. A diferencia de las bases de datos relacionales, Cassandra no admite joins. Esto significa que los mismos datos pueden duplicarse en varias familias de columnas.

**ARTÍCULO DE INTERÉS**

En el siguiente artículo puedes aprender más sobre el funcionamiento e historia de Cassandra.

https://www.paradigmadigital.com/dev/cassandra-la-dama-de-las-bases-de-datos-nosql/



**VIDEO DE INTERÉS**

En este enlace de vídeo podrás conocer más en profundidad la historia e introducción de Cassandra a través del siguiente tutorial.

https://www.youtube.com/watch?v=7bM5e2xa6Ic

### 3.2.2 HBase

HBase también es un almacén de datos NoSQL distribuido, escalable y tolerante a fallos diseñado para almacenar grandes conjuntos de datos y se ejecuta sobre HDFS. Tiene características similares a Cassandra, ya que ambas están inspiradas en BigTable, un tipo de gestor de bases de datos creado por Google.


Logo de Hbase
Fuente: https://hbase.apache.org

BigTable es un sistema de almacenamiento distribuido que Google creó para administrar petabytes de datos estructurados a través de miles de servidores de bajo coste. No es compatible con el modelo de datos relacionales, pero proporciona un modelo de datos simple, que proporciona a las aplicaciones del cliente un control dinámico sobre el almacenamiento de datos.

HBase almacena datos en tablas. Una tabla consta de filas. Una fila está formada de familias de columnas. Una familia de columnas se compone de columnas versionadas. Sin embargo, una tabla y una columna en HBase son muy diferentes de una tabla y columna en una base de datos relacional. Una tabla HBase es esencialmente una tabla dispersa, distribuida, persistente y multidimensional. Google la define como un mapa ordenado con las características indicadas.

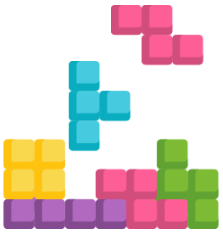
El mapa es una estructura de datos compatible con la mayoría de los lenguajes de programación. Es un contenedor para almacenar pares clave-valor. Es una estructura de datos muy eficiente para buscar valores por claves. Generalmente, el orden de las claves no es definido y una aplicación no se preocupa por el orden, ya que le da una clave al mapa y recupera un valor por dicha clave. Hay que tener en cuenta que la estructura de datos del mapa no debe confundirse con la función del mapa en Hadoop MapReduce, ya que la función del mapa es un concepto de lenguaje funcional para transformar datos.

**¿SABÍAS QUE...?**

La estructura de datos del mapa se denomina con diferentes nombres en los diversos lenguajes de programación. Por ejemplo, en PHP se llama matriz asociativa. En Python se conoce como diccionario. Ruby lo llama hash. Java y Scala lo llama mapa.

Una tabla HBase es un mapa multidimensional o multinivel ordenado. La clave de primer nivel es la clave de fila, lo que permite que una aplicación lea rápidamente una fila de miles de millones de filas. La clave del segundo nivel es la familia de columnas. La clave de tercer nivel es el nombre de la columna, también conocido como calificador de columna. La clave del cuarto nivel es la marca de tiempo. Una combinación de clave de fila, familia de columnas, columna y marca de tiempo, identifica de forma exclusiva una celda que contiene un valor. Un valor es una matriz de bytes no interpretada.

Una fila en una tabla HBase es escasa. A diferencia de las filas en una base de datos relacional, no todas las filas en HBase necesitan tener las mismas columnas. Cada fila tiene el mismo conjunto de familias de columnas, pero una fila puede no almacenar nada en algunas familias de columnas. Una celda vacía no ocupa espacio de almacenamiento.



**EJEMPLO PRÁCTICO**

Necesitas crear una base de datos en la que hay una tabla dispersa y con elementos maestros. ¿Qué arquitectura deberías utilizar?

**SOLUCIÓN:**

Cassandra tiene una arquitectura distribuida sin maestría, es decir, sin existencia de elementos maestros. Sin embargo, HBase sí permite la existencia de elementos maestros, por lo que sería la opción adecuada.

### *3.2.3 Spark*

Spark, como ya sabemos, es una tecnología muy relevante en los análisis de macro datos. Por ello, vamos a comenzar con una pequeña introducción de algunos conceptos de esta tecnología.


Logo de Spark
Fuente: https://spark.apache.org

Es un motor de procesamiento de datos en memoria optimizado que permite ejecutar operaciones de datos en Hadoop (HDFS) con mejor rendimiento que MapReduce. También puede procesar datos en otros entornos, como Cassandra o incluso en agrupaciones o archivos. Al minimizar las lecturas y escrituras realizadas en disco, Spark ofrece acceso de muy alto rendimiento a datos dinámicos y colecciones de datos complejas, lo que proporciona una funcionalidad interactiva como consultas ad doc de forma escalable.

**¿SABÍAS QUE...?**

Apache Spark nació en 2009, en la Universidad de Berkeley, y fue liberado como código abierto en 2010.

Extiende el paradigma de procesamiento por lotes, que es tan popular en MapReduce hacia las aplicaciones en tiempo real que los usuarios demandan. Los desarrolladores aprecian las ganancias de productividad que proporciona Spark a través de su amplia base de lenguajes, incluido Java, Scala, Python y R. Soportando tantos lenguajes diferentes, Spark tiene muchas menos barreras técnicas que las demás tecnologías antes comentadas.

Para el uso de datos estructurados en Spark hay varias opciones basadas en SQL, como Spark SQL. Es decir, Spark funciona con una amplia variedad de fuentes de datos, incluidas las bases de datos SQL.

En resumen, Spark ha nacido como resultado de un mundo donde recolectar datos se ha vuelto muy barato y, en el cual, muchas organizaciones consideran negligente no guardar sus datos de cara al futuro donde podrían ser aprovechados para posibles negocios relevantes. Es verdad que esta acumulación masiva de datos hace que su procesamiento requiera cálculos largos y complejos, muchas veces en clúster. Pero también es verdad que todo apunta a que el coste del procesamiento se seguirá abaratando.

En este nuevo mundo de datos, los softwares desarrollados en los últimos cincuenta años, basados en las bases de datos relacionales nacidas en los años setenta, no son escalables. Tampoco es posible que los modelos de programación tradicional procesen las aplicaciones de datos, lo que crea la necesidad de nuevos modelos de programación.

**ENLACE DE INTERÉS**

En el siguiente enlace se pueden apreciar las ventajas e inconvenientes de Spark y Hadoop

https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/spark-vs-hadoop-quien-saldra-vencedor

### *3.2.4 Relacionando big data, mapreduce, hadoop y spark*

Ahora que ya empezamos a conocer aspectos sobre big data, MapReduce, Hadoop y Spark, para comprender mejor como encajan con claridad los conceptos, aquí veremos un resumen rápido de cómo se relacionan entre ellos:

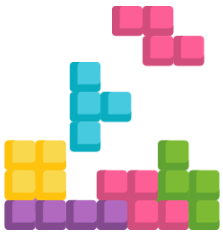
- **Big data:** actualmente, la mayoría de las empresas se enfrentan a volúmenes de datos nuevos, que llegan en muchas formas diferentes. El big data tiene el potencial de proporcionar ideas que pueden transformar todos los sectores empresariales y negocios. Además, el uso de big data ha generado una industria completamente nueva de arquitecturas de apoyo como MapReduce.

- **MapReduce:** es un nuevo marco de programación, creado y desplegado con éxito por Google, que utiliza el método de dividir en muchos servidores básicos para descomponer problemas complejos de big data en unidades pequeñas de trabajo. Posteriormente los procesa en paralelo.

Estos problemas ahora se pueden resolver más rápido que nunca, pero implementar MapReduce de modo independiente es demasiado complejo para la mayoría de las empresas. Esta realidad llevó a la creación de Hadoop.

- **Hadoop:** es una pila (stack) tecnológica completa que implementa los conceptos de MapReduce para explotar el big data. Hadoop también generó un mercado potente basado en un producto de código abierto de gran valor añadido. En los últimos años, los científicos de datos y los desarrolladores de aplicaciones se han orientado hacia la combinación de Spark con Hadoop.

- **Spark:** se trata de un nuevo enfoque, compuesto por una infraestructura de software junto con un modelo de desarrollo de aplicaciones simple, pero con técnicas potentes para desarrollar y comercializar aplicaciones de big data. En lugar de ser visto como un reemplazo del MapReduce, Spark se puede apreciar mucho mejor como el siguiente paso en la evolución del big data. De hecho, la mayoría de las distribuciones de Hadoop también incluyen una distribución de Spark, que se instalará en el mismo ambiente. Trabaja en memoria.



**EJEMPLO PRÁCTICO**

Necesitas crear una base de datos en la que los procesos sean lo más rápidos posibles, teniendo en cuenta que no tienes problemas ni de memoria ni de almacenamiento. ¿Qué resultaría más adecuado: Spark o Hadoop?

**SOLUCIÓN:**

Como Spark trabaja en memoria, todos los procesos son más rápidos que en Hadoop, así que la opción adecuada es Spark.

## 3.3 Procesamientos distribuidos. MapReduce

Las técnicas antiguas para trabajar con datos simplemente no son escalables para el big data, porque son demasiado costosas, lentas y complicadas. Por lo tanto, se creó una nueva forma de interactuar con estos datos y es ahí donde surgió el MapReduce.

En pocas palabras, el MapReduce se basa en el concepto de división mediante el uso de computación distribuida y procesamiento paralelo, dividiendo las tareas masivas en trozos más pequeños, asignándolas a múltiples servidores y procesándolas en paralelo.

Frente a su propio conjunto de desafíos particulares, en 2004 Google decidió introducir el poder de la computación distribuida paralela para que le ayudase a digerir las enormes cantidades de datos de sus operaciones diarias. El resultado fue un grupo de tecnologías, arquitecturas y filosofías de diseño que se conocieron como MapReduce.

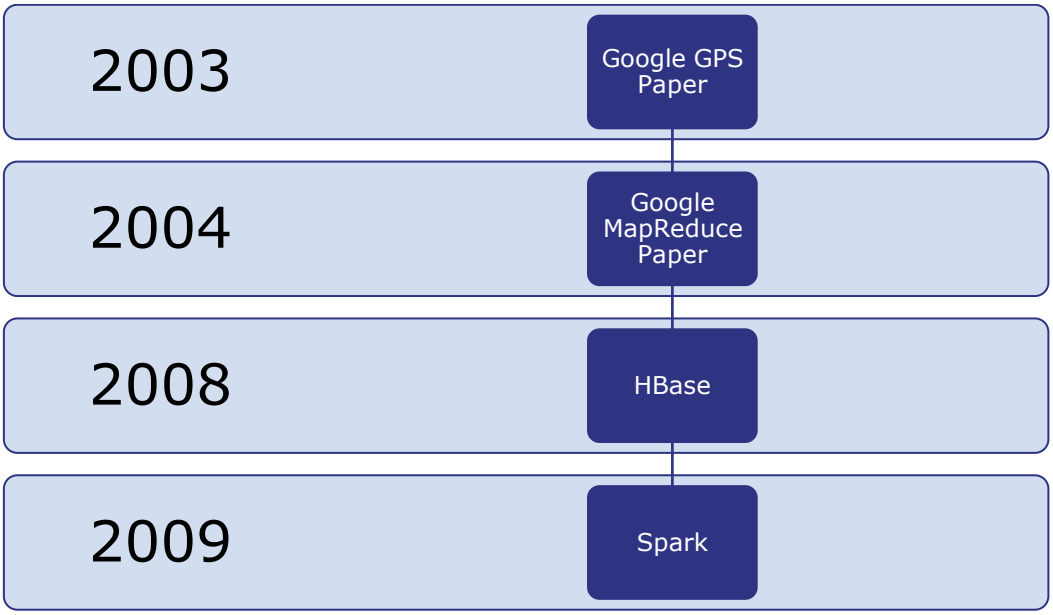

MapReduce
Fuente: https://errequeerre.es/wp-content/uploads/2017/08/Hadoop-Map-Reduce.png

En MapReduce, la lógica de programación basada en tareas se coloca tan cerca a los datos como sea posible. Esta técnica funciona bien, tanto con datos estructurados como no estructurados. En esencia, MapReduce está compuesto de dos pasos principales de procesamiento: mapear y reducir. Si los ponemos juntos, ya tenemos MapReduce.

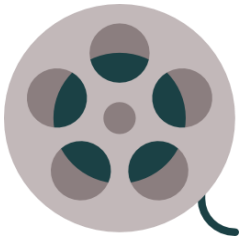
En la fase de mapeo de MapReduce, los registros de una gran cantidad de datos de las fuentes se dividen y procesan en tantos servidores como sea necesario - en paralelo - para producir valores intermedios. Después de ello, se realiza el procesamiento del mapeo, se recopilan los resultados intermedios y se combinan (reducen) en los valores finales.

Este es un enfoque mucho más simple para los cálculos a gran escala y está destinado a abstraer gran parte de la complejidad del procesamiento de datos. Sin embargo, a pesar de su simplicidad, MapReduce permite procesar cantidades masivas de información mucho más rápido que nunca antes. No es de extrañar que Google haya elegido una metodología basada en dividir los datos, dada su filosofía organizacional de usar muchos servidores para el procesamiento y almacenamiento de datos en vez de concentrarse en menos servidores, más potentes y caros.

Junto con la arquitectura MapReduce, Google también creó el sistema de archivos de Google (GFS o Google File System). Esta tecnología innovadora es un potente sistema de archivos distribuido destinado a contener enormes cantidades de datos. Google optimizó este sistema de archivos para cumplir con su voraz necesidad de procesamiento de información. Sin embargo, esto fue solo el comienzo. MapReduce de Google sirvió como base para posteriores tecnologías como Hadoop, mientras que GFS era la base para HDFS.




Evolución de la arquitectura MapReduce
Fuente: Elaboración propia



**VIDEO DE INTERÉS**

En este vídeo podrás ver una explicación muy sencilla y visual para entender mejor el concepto tan abstracto de MapReduce por parte del equipo de LUCA (expertos en datos de Grupo Telefónica).

https://www.youtube.com/watch?v=NQ8mjVPCDvk

### 3.3.1 La influencia del MapReduce

Si Google fuera la única organización que implementara el MapReduce, la historia probablemente no hubiese ido más lejos. Pero como señalamos anteriormente, cuando hablamos de qué es el big data, el crecimiento explosivo del big data colocó a los departamentos de TI de todas las industrias bajo un gran estrés.

Los viejos procedimientos para manejar toda esta información ya no se podían escalar y las organizaciones necesitaban un nuevo enfoque. El procesamiento distribuido había demostrado ser una excelente manera de hacer frente a grandes cantidades de datos de entrada. El hardware y el software de las categorías más bajas hicieron rentable emplear cientos o miles de servidores trabajando en paralelo para responder una pregunta.

MapReduce fue solo el comienzo y proporcionó una versión correctamente validada de una arquitectura tecnológica, que ayudó a resolver los desafíos del big data, frente a las propuestas de diversos productos comerciales. Además, MapReduce sentó las bases para el siguiente paso en la evolución de big data que fue Hadoop.

### 3.3.2 ¿Qué es Hadoop?

MapReduce fue un gran comienzo, pero requería que se invirtiese una cantidad significativa de recursos tecnológicos y de desarrollo para que pudiese funcionar en una empresa. Esto no era factible para la mayoría de las compañías y esta relativa complejidad condujo a la aparición de Hadoop.

Por definición, al ser Hadoop un software de código abierto está ampliamente difundido. Está basado en patrones y ha sido creado sobre la base de datos MapReduce de Google. Está destinado a aprovechar el poder del procesamiento masivo de datos del big data mediante el uso de una gran cantidad de servidores de bajo coste.

Hadoop fue diseñado para abstraer gran parte de la complejidad del procesamiento distribuido. Esto permitió que los desarrolladores se centrasen en su tarea, en lugar de perderse en los detalles técnicos de la implementación de un entorno funcional. Pero la estrategia de procesamiento de MapReduce que se ejecuta en Hadoop ha traído algunos desafíos que, a su vez, han sido abordados en Apache Spark.

Después de que comenzó a hacerse popular este software, la Fundación Apache, desde su perspectiva sin ánimo de lucro, se hizo cargo del mantenimiento de Hadoop, con Yahoo! haciendo contribuciones significativas. Hadoop ha ganado una tremenda adopción en una amplia variedad de organizaciones, incluyendo redes sociales, como Facebook, LinkedIn o Twitter. Empresas científicas y tecnológicas, servicios financieros, redes minoristas y gobiernos lo han adoptado.

### *3.3.3 Arquitectura Hadoop*

Apache Hadoop es la plataforma más popular para el procesamiento de big data, y se puede combinar con una gran variedad de otras grandes herramientas de datos para construir potentes soluciones analíticas.

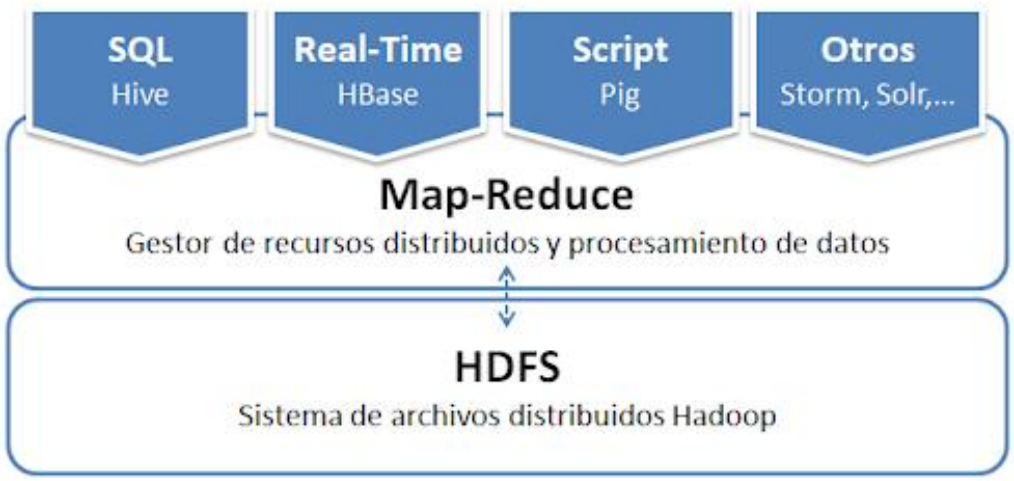
Podemos decir que Hadoop es un paraguas bajo el cual caben muchos subproyectos que construyen los componentes de su arquitectura.

El núcleo de Hadoop proporciona dos servicios que son la capacidad de almacenamiento y la de computación. De esta forma, el flujo de trabajo habitual de Hadoop consiste en cargar datos en el sistema de archivos distribuidos de Hadoop (HDFS) y procesarlos usando la API de MapReduce u otras herramientas que se basan en MapReduce como marco de ejecución.

Tanto HDFS como MapReduce comparten varios aspectos:

- Están diseñados para funcionar en clusters de servidores de baja a media capacidad.
- Escalan su capacidad agregando más servidores (scale-out) a diferencia de los modelos anteriores de usar hardware más grande (scale-up).
- Tienen mecanismos para identificar y trabajar en torno a los fallos.
- Proporcionan la mayoría de sus servicios de forma transparente, permitiendo al usuario concentrarse en el problema de negocio a resolver.

Imagina un entorno Hadoop como algo que consta de tres capas básicas. Estas capas representan una jerarquía lógica con una separación completa de actividades, en otras palabras, cada capa tiene sus propias especializaciones, que se llevan a cabo independiente de las otras capas. Juntos, estos niveles ofrecen una implementación completa de MapReduce, conforme se explica en la siguiente imagen.




Arquitectura Hadoop
Fuente: https://www.diegocalvo.es/en/apache-hadoop/hadoop-basic-architecture

## 3.4 Herramientas para fines operacionales vs analíticos

Profundizando en Hadoop, como referencia de las tecnologías clave para fines analíticos, diremos que este presenta varias capas, las cuales, veremos a continuación una a una.

### *3.4.1 Capa de aplicación o capa de acceso del usuario final*

Proporciona un marco de programación para desarrollar software aplicando técnicas informáticas distribuidas para extraer y explotar cantidades significativas de datos. Por lo tanto, sirve como punto principal para que otros niveles o capas interactúen con Hadoop.

Estas capas, en forma de aplicaciones, pueden ser soluciones desarrolladas internamente a medida o a través de herramientas externas de business intelligence (BI) y algunos softwares empresariales.

Para construir una de estas aplicaciones, normalmente se emplea una interfaz convencional de programación como Java o Pig (lenguaje especializado en MapReduce). Mientras tanto, para gestionar los trabajos de Hadoop, se puede usar un software desarrollado también por Apache que es Oozie.

**ENLACE DE INTERÉS**

Para acceder a más información acerca de Apache Pig, Apache Hive o Apache Oozie, se pueden visitar los siguientes enlaces.

- http://pig.apache.org/
- http://hive.apache.org/
- http://oozie.apache.org/

### *3.4.2 Capa de gestión de la carga de trabajo de MapReduce*

Comúnmente conocido como JobTracker, este componente de Hadoop proporciona un motor de ejecución de trabajos en tiempo de ejecución de código abierto.

Este motor coordina todos los aspectos del entorno Hadoop, como programar y lanzar trabajos, equilibrando la carga entre diferentes recursos, lidiar con los fallos y otros problemas. Es decir, se ocupa de todas las responsabilidades esenciales en un entorno de datos altamente distribuido.

### *3.4.3 Sistemas de archivos paralelos distribuidos/capa de datos*

La capa de datos es responsable del almacenamiento real de los datos. Para obtener eficiencia, normalmente utiliza un sistema distribuido especializado, generalmente el HDFS.

Junto con el HDFS, esta capa también puede basarse en implementaciones de almacenamiento de terceras partes, como Azure, Amazon u otras bases de datos en la nube.

Los inventores de HDFS tomaron una serie de decisiones al respeto de su diseño:

- Los archivos se almacenan en bloques más grandes: muchos sistemas de archivos almacenan su información en bloques de 4 kilobytes (Kb).

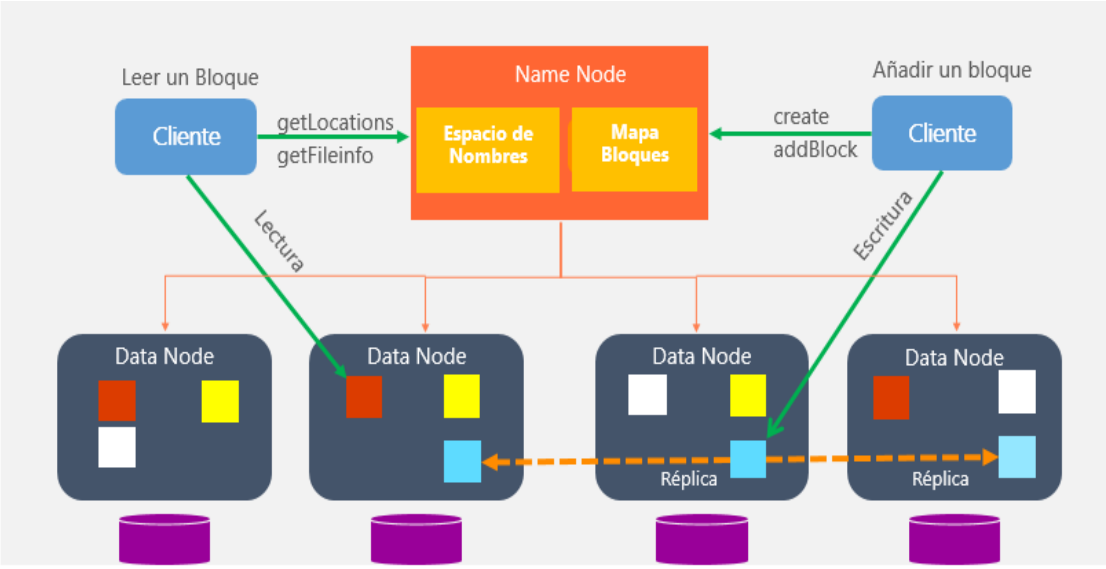
Algunos incluso usan bloques más pequeños. En contraste, los bloques en HDFS son cientos de veces más grandes, con un valor predeterminado de 64 o 128 megabytes. Almacenar la información de Hadoop en estos bloques grandes ayuda a reducir el número de bloques que deben ser mantenidos y también reducen el consumo de recursos generales.

- La confiabilidad se logra a través de la replicación: cada bloque es replicado en dos o más servidores que tienen asignado el trabajo de almacenar la información. Formalmente conocido como DataNodes, la mayoría de los entornos de Hadoop se replican a un número predeterminado de tres.

- Un único NameNode maestro coordina el acceso y la gestión de los metadatos, lo que simplifica y centraliza la gestión.

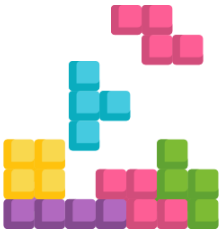
- No hay captura de datos: no vale la pena capturar los datos, debido a los grandes conjuntos y consultas secuenciales.

- Hay una interfaz con una Application Programming Interface (API) personalizable: esto permite simplificar el problema y centrase en las aplicaciones distribuidas, en lugar de manipular datos a bajo nivel.




Lectura y Escritura en HDFS
Fuente: blogthinkbig.com

HDFS era una opción razonable para un sistema de archivos, pero en los últimos años, muchas otras opciones de almacenamiento de datos se volvieron populares. Esta diversidad no solo trajo algunos nuevos desafíos para Hadoop y MapReduce, sino que también fue un gran catalizador para nuevas técnicas como Spark.



**EJEMPLO PRÁCTICO**

Necesitamos conocer dónde y cómo se encuentran almacenados los datos en disco. Teniendo en cuenta que usamos la arquitectura Hadoop, ¿cómo podríamos averiguar esta información?

**SOLUCIÓN:**

Hadoop cuenta con NameNode, que es un maestro que coordina todos los metadatos. Con NameNode podemos saber los identificadores de los bloques en disco y los nodos que están en los bloques.

### *3.4.4 El ecosistema Hadoop*

La estructura modular de Hadoop ha generado un dinámico ecosistema de tecnologías y proveedores altamente especializados. Los elementos principales en el ecosistema de Hadoop incluyen:

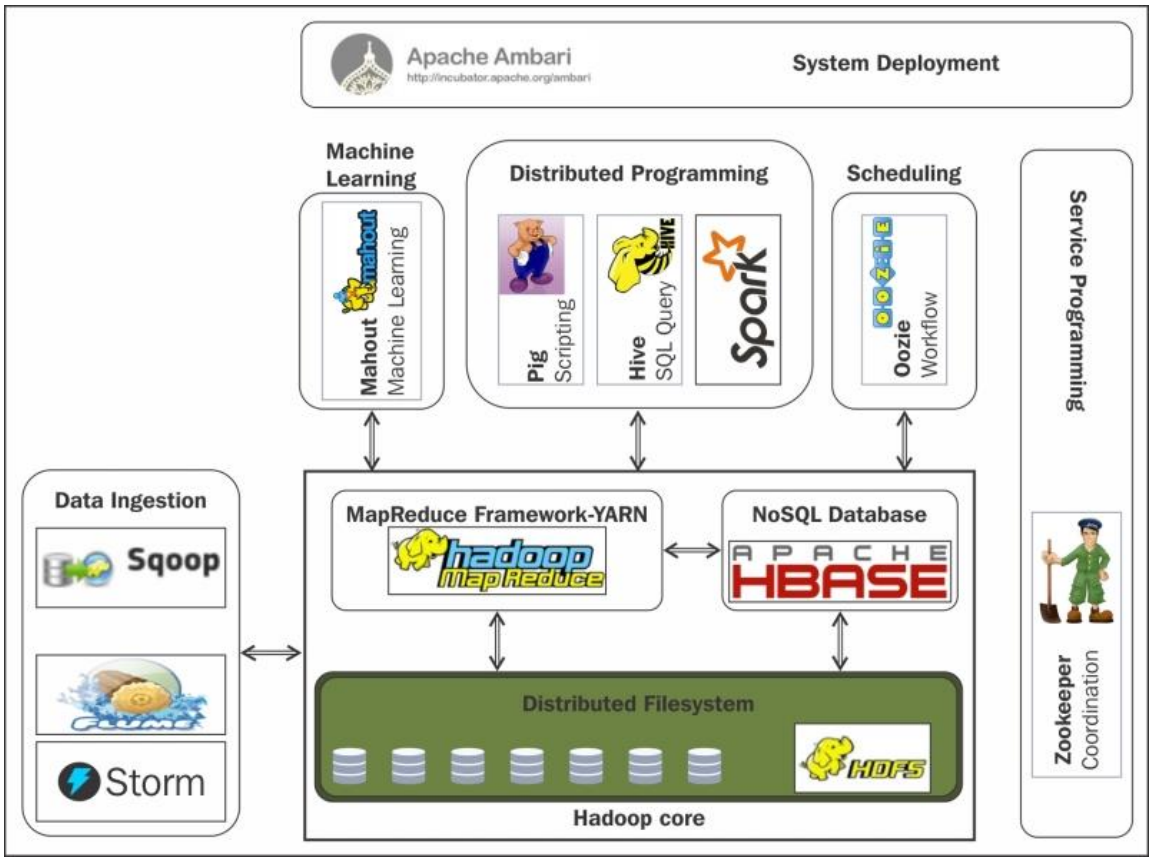
- **Almacenamiento de datos distribuido:** estas tecnologías se utilizan para mantener grandes conjuntos de datos en clústeres de almacenamiento.

- **Distribución del tiempo de ejecución de MapReduce:** este software tiene asignada la importante responsabilidad de programar y distribuir trabajos que procesan la información mantenida en el almacenamiento de datos distribuido.

- **Herramientas y aplicaciones de desarrollo de software:** una amplia gama de soluciones que permite a programadores y no programadores generar valor y encontrar significado en la información almacenada en Hadoop.

- **Distribuciones Hadoop:** estos paquetes comerciales proporcionan un único conjunto integrado de componentes que se prueban previamente y se certifican para garantir que operarán correctamente de forma conjunta.

- **Herramientas de business intelligence:** estas populares tecnologías, que se llevan utilizando durante años operando con datos relacionales tradicionales, ahora se han ampliado para trabajar con datos accesibles a través de Hadoop. Algunas herramientas cuentan incluso con conexión directa a Hadoop.




Proyectos del Ecosistema Hadoop
Fuente: blogthinkbig.com

### 3.4.4.1 Apache Hive

El mundo de la ciencia de la computación es abstracto. Todos sabemos que los datos son información en forma de bits y que los lenguajes de programación como C nos proporcionan una capa de abstracción sobre la máquina y nos evitan la necesidad de conocer el lenguaje ensamblador. También sabemos que una mayor abstracción es proporcionada por otros lenguajes de más alto nivel que C.

El lenguaje de consulta estructurado (SQL) es una de esas abstracciones. SQL es utilizado en todo el mundo por muchos expertos en modelado de datos. Hadoop es bueno para el análisis de big data, así que una pregunta latente es, entonces, ¿cómo pueden aprovecharse todo gran grupo de profesionales que conoce SQL para utilizar el poder de computación de Hadoop y empezar a hacer big data?

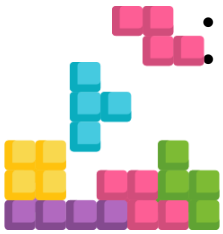
Para escribir el programa MapReduce de Hadoop, los usuarios deben conocer un lenguaje de programación que pueda usarse para ello.

Los problemas cotidianos del mundo empresarial siguen ciertos patrones. Algunos de estos problemas son comunes y recurrentes, como la manipulación de datos, el manejo de valores faltantes, la transformación de datos y el resumen de estos.

Escribir código de MapReduce para estas situaciones es un trabajo que obliga a pensar diferente, sobre todo a aquellos que no cuentan con experiencia como programadores. Podemos decir que escribir código para resolver este tipo de problemas no es algo muy inteligente ni valioso, porque será difícil hacer un código óptimo. Pero escribir código eficiente que tenga rendimiento, escalabilidad y puede extenderse sí que es valioso.

Por eso, con ese problema en mente, Apache Hive se desarrolló en Facebook, para que los problemas del día a día en materia de data warehouse pudiesen resolverse sin tener que escribir el código MapReduce.

Según se describe en la propia wiki de Hive: es una infraestructura de almacén de datos basada en Apache Hadoop y tiene su propio dialecto SQL, conocido como el lenguaje de consulta Hive o también denominado como HiveQL o, a veces, HQL.



**EJEMPLO PRÁCTICO**

Necesitamos crear una tabla en una base de datos con Hive, en la que registrar el nombre, apellido e ID de los alumnos del centro en el que trabajamos. La base de datos se llama Curso y la tabla alumnos, ¿cómo podríamos hacerlo?

**SOLUCIÓN:**

En primer lugar, debemos arrancar el servicio de Hadoop y Hive de la siguiente forma:

```
$ cd hive
$ bin/hive
```

Antes de crear la base de datos consultaremos que bases de datos existen:

```
hive> SHOW DATABASES;
```

Si la base de datos no existe, la crearemos de la siguiente forma:

```
hive> CREATE DATABASE Curso
```

Ahora tendremos que seleccionar dicha base de datos:

```
hive> use Curso;
```

Una vez que nos situamos en la base de datos, para ver las tablas que contiene ejecutaremos:

```
hive> show tables;
```

Ahora solo nos falta crear la tabla con los campos y lo haremos de la siguiente forma:

```
CREATE EXTERNAL TABLE alumnos (AlumnoID INT, Nombre STRING, Apellido STRING)
COMMENT 'lista de alumnos'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/hadoop/alumnos';
```

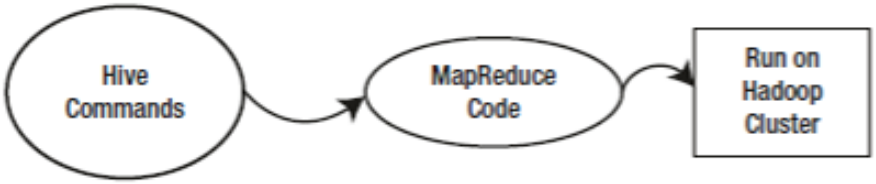
Mediante HiveQL, Hive consulta datos en HDFS. Hive no solo se puede ejecutar en HDFS, sino que también se ejecuta en Spark y otros frameworks de big data.

Hive proporciona un sistema de gestión de bases de datos relacionales, así como aporta abstracción a los usuarios para el manejo de datos estructurados en HDFS. Podemos crear tablas y ejecutar consultas similares a SQL en ellos.

Hive guarda el esquema de la tabla en algunos RDBMS (sistemas gestores de bases de datos relacionales). Apache Derby es el RDBMS predeterminado que se incluye con la distribución de Apache Hive. Apache Derby ha sido escrito completamente en Java y es un RDBMS de código abierto que viene con la licencia de Apache versión 2.0.

Los comandos de HiveQL se transforman en el código MapReduce de Hadoop, y luego se ejecutan en el clúster de Hadoop. Podemos ver el flujo de ejecución de comandos en Hive en la siguiente figura.

Una persona que conozca SQL puede aprender fácilmente Apache Hive y HiveQL y, por tanto, podrá usar el poder de almacenamiento y cálculo de Hadoop en sus trabajos de análisis de datos diarios en big data. HiveQL también es compatible con PySpark SQL. Podemos ejecutar comandos HiveQL en PySpark SQL. Aparte de ejecutar consultas HiveQL, también podemos leer datos de Hive directamente con PySpark SQL y escribir los resultados en Hive.




**ENLACE DE INTERÉS**

Para más información sobre Apache Hive y Apache Derby, podemos consultar las siguientes páginas web.

- https://cwiki.apache.org/confluence/display/Hive/Tutorial
- https://db.apache.org/derby/

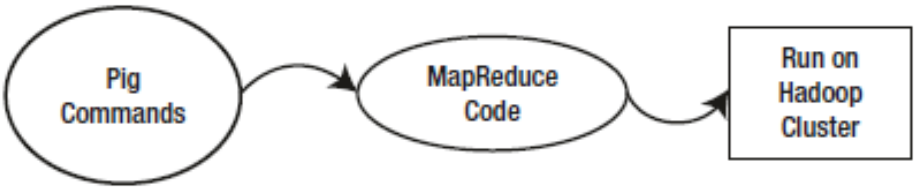
### 3.4.4.2 Apache Pig

Apache Pig es una plataforma de flujo de datos para realizar análisis de macro datos. Fue desarrollado por Yahoo y en la actualidad, como tantos otros, es un proyecto open source de la Fundación Apache. Concretamente, en la actualidad, está disponible bajo licencia Apache versión 2.0. El lenguaje de programación Pig es un script del lenguaje denominado Pig Latin. Pig está conectado libremente a Hadoop, lo que significa que podemos usarlo vinculado a Hadoop y realizar infinidad de análisis, pero Pig también se puede usar de forma independiente con otras herramientas, como Apache Tez y Apache Spark.

Apache Hive se suele utilizar como la herramienta de informes, mientras Apache Pig se usa para extraer, transformar y cargar, es decir, para los procesos de ETL. Podemos ampliar la funcionalidad de Pig utilizando funciones definidas por el usuario (UDF o user defined functions). Las funciones definidas por el usuario pueden escribirse en muchos lenguajes, incluidos Java, Python, Ruby, JavaScript, Groovy y Jython.

Apache Pig usa HDFS para leer y almacenar los datos y Hadoop MapReduce para ejecutar los algoritmos. Apache Pig es similar a Apache Hive en términos de uso del clúster Hadoop. Como se muestra en la siguiente imagen, en Hadoop, los comandos de Pig Latin se transforman primero en códigos de Hadoop MapReduce. Luego se transforman en el código MapReduce, que se ejecuta en el clúster de Hadoop.

La mejor parte de Pig es que el código está optimizado y probado para que funcione sin problemas. Simplemente necesitamos instalar Pig y comenzar a usarlo. Esta herramienta proporciona el shell Grunt para ejecutar los comandos interactivos de Pig. De este modo, cualquier usuario que aprenda Pig Latin puede disfrutar de los beneficios de HDFS y MapReduce sin necesidad de conocer lenguajes de programación avanzados y más complejos de estudiar como Java o Python.




**ENLACE DE INTERÉS**

Para conocer más sobre Apache Pig, puedes acceder a la siguiente página, donde encontrarás el index.

http://pig.apache.org/docs/

### 3.4.4.3 Desafíos de Hadoop y MapReduce

MapReduce y Hadoop fueron un gran comienzo y continúan siendo muy utilizados, pero no son el final de la historia para el procesamiento de macro datos.

Por el contrario, las técnicas y herramientas han continuado evolucionando, en parte, impulsadas por un notable desarrollo de soluciones que resuelven los principales quebraderos de cabeza, como:

- **Procesamiento por lotes:** desde el principio, Hadoop junto con MapReduce ha sido principalmente implementado en ambientes de procesamiento por lotes. Pero una vez que los usuarios perciben lo que es posible hacer con el big data, ellos quieren más y quieren obtener las respuestas en el momento exacto.

- **Rendimiento:** Hadoop y MapReduce son muy dependientes de las interacciones de lectura y escritura, lo que les toma mucho tiempo operando con las unidades de disco. A medida que crecen, tanto los volúmenes de datos almacenados como las expectativas de los usuarios, las ralentizaciones se vuelven cada día menos aceptables.

- **Complejidad en el desarrollo de la solución:** mientras que Hadoop y las metodologías de construcción de aplicaciones de MapReduce son mucho más simples que las técnicas de desarrollo de aplicaciones anteriores hechas de un modo más manual, la realidad es que aún están fuera del alcance de la gran mayoría de los desarrolladores de software, porque la curva de aprendizaje sigue siendo demasiado elevada.

- **Proliferación de silos:** con los usuarios atraídos por la posibilidad de un acceso rápido a una gran cantidad de datos valiosos y la ausencia de una supervisión adecuada, la realidad organizacional puede caminar rápidamente hacia la formación de colecciones de datos aislados y costosos de mantener. Esto es debido a que, por defecto, cada dato aislado tendrá su propio hardware dedicado y será inaccesible desde el resto de la organización. A pesar de ser una pesadilla para la seguridad y para las tareas de los administradores de sistemas, no siempre se limita a tiempo esta proliferación, lo que redunda en mayores gastos y en una disminución del retorno de la inversión (ROI).

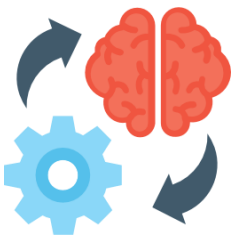
- **Soportar múltiples versiones:** los software de código abierto, y más en el caso de infraestructuras especiales como Hadoop, están caracterizados por ser aplicaciones con actualizaciones sin fin. Muchas organizaciones se ven obligadas a ejecutar múltiples versiones de sus softwares de código abierto al mismo tiempo, provocando también la aparición de colecciones cada vez mayores de datos aislados.

- **Apoyo a múltiples usuarios:** cuando una empresa planea cuantos costes va a tener derivados de gastos no sustanciales para hardware de big data, software y profesionales capacitados, siempre espera el máximo retorno financiero. Esto significa proporcionar datos y soluciones para el mayor número de usuarios en la organización que sea posible. El problema es que esta estrategia de compartición, sin embargo, abre un potencial muy grande de riesgos en seguridad de datos y de la información, especialmente cuando se anima cada vez más gente a participar de las plataformas de datos.

- **Dimensionamiento de la infraestructura:** a medida que una empresa despliega su infraestructura destinada a soportar el big data, es fácil caer en dos errores muy comunes, que son las dos caras de la misma moneda: contratar más capacidad de la necesaria o, por el contrario, no contratar la suficiente y que el servidor no funcione correctamente.

- **Restricciones de almacenamiento:** en parte por el diseño, en parte por el uso común que se hace de ellas, Hadoop y MapReduce se han concentrado en un número relativamente pequeño de repositorios de información, siendo HDFS el más popular de ellos.

  Sin embargo, casi una docena de repositorios están ampliamente difundidos y otras bases de datos menos frecuentes también se encuentran en uso en el mundo corporativo. Aprovechar al máximo todos esos datos, significa que el entorno de big data debe ser capaz de trabajar con una gama completa de orígenes.



**RECUERDA**

Si no hay suficiente capacidad de servidor para satisfacer las necesidades de los usuarios, resultará una experiencia desagradable para la comunidad de usuarios.

Por otro lado, si las organizaciones adquieren un exceso de capacidad, resultará en dinero y personal malgastado, pudiendo haber invertido ese dinero en otras cuestiones.

**¿SABÍAS QUE…?**

Aparte de los inconvenientes inherentes que presentan Hadoop y MapReduce, nuevas tecnologías, metodologías y tendencias de mercado están afectando al big data y cambiando la forma en que se llevan a cabo estas iniciativas en las empresas. Algunos de los movimientos más significativos incluyen:

- **Ecosistemas de software en rápida evolución:** el ritmo de cambio en los ecosistemas de software es acelerado. Por ejemplo, el propio Spark apareció en el mercado rápidamente después de Hadoop y MapReduce e igual de rápido comenzó a ganar una gran adopción por parte de los usuarios.

- **Tecnologías en la nube:** estamos en un punto donde la mayoría de las versiones de los software y soluciones empresariales son desarrolladas e implementados en la nube.

- **Nuevos almacenes de datos y modelos de procesamiento:** los visionarios de la tecnología y las empresas más innovadoras continúan aprovechando la potencia de procesamiento, la capacidad de almacenamiento y el ancho de banda de internet al máximo, para lanzar nuevos enfoques de negocio. La dirección general va por el camino de democratizar el acceso a la información y crear soluciones innovadoras en este proceso.

### *3.4.5 Sistemas de mensajería*

Los datos generalmente fluyen de una aplicación a otra. Los datos pueden ser producidos por una aplicación y usados por otra u otras varias. En general, bajo este modelo, la aplicación que genera o envía datos se denomina productor y la que recibe los datos se llama consumidor.

A veces hay una asimetría entre el número de aplicaciones que producen datos y el número de aplicaciones que consumen esos datos. Por ejemplo, una aplicación puede producir datos que son utilizados por múltiples consumidores. Del mismo modo, una aplicación puede consumir datos de múltiples productores.

A veces también hay asimetría entre la velocidad a la que una aplicación produce datos y la velocidad a la que otra aplicación puede consumirlos. Una aplicación puede producir datos más rápido que la velocidad a la cual los consumidores pueden consumir esos datos.

Una forma sencilla de enviar datos de una aplicación a otra es conectarlas directamente entre sí. Sin embargo, esto no funcionará si hay asimetría en el número de productores y consumidores de datos o la proporción en la que se producen y consumen los datos. Por tanto, un desafío más en el mundo del big data es ajustar ese acoplamiento estrecho entre los productores y consumidores. Estas circunstancias requieren que se ejecuten al mismo tiempo las aplicaciones y al mismo ritmo de producción e ingesta de los datos. O, por lo contrario, que se implementen mecanismos complejos de buffering, que actúen como colchón entre las aplicaciones. Por lo tanto, las conexiones directas entre productores y consumidores parece que no escalan adecuadamente.

Una solución flexible y escalable es utilizar un **agente de mensajes** o un **sistema de mensajería**. En lugar de disponer de aplicaciones conectándose directamente entre sí, estas se conectan a un agente de mensajes o un sistema de mensajería. Esta arquitectura facilita la adición de nuevos productores o consumidores a una **tubería de datos** (pipeline). También permite que las aplicaciones produzcan y consuman datos a diferentes velocidades.

### *3.4.6 Apache Kafka*

Apache Kafka es una plataforma de mensajería distribuida de las denominadas de tipo publicación-suscripción. Fue desarrollado en LinkedIn y después ha sido lanzado como un proyecto abierto desde la Fundación Apache.


Logo de Apache Kafka
Fuente: https://kafka.apache.org

Es tolerante a fallos, escalable y rápida. Un mensaje, en términos de Kafka, es la unidad de datos más pequeña, que fluye desde el productor al consumidor a través de un servidor de Kafka y puede ser persistente y utilizado en un momento posterior.

Además de los términos productor y consumidor, otro término clave que vamos a usar en el contexto de Kafka es el tema. Un tema es un flujo de mensajes de una categoría similar. Kafka viene con una API incorporada que los desarrolladores pueden usar para construir sus aplicaciones. Por lo tanto, nosotros seremos los que definimos el tema. A continuación, discutimos los tres componentes principales de Apache Kafka.

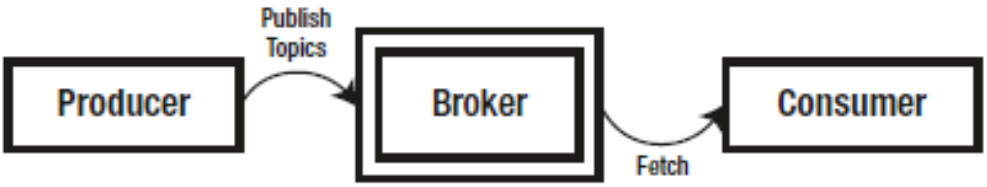
- **Productor:** el productor en Kafka produce el mensaje a un tema de Kafka. También puede publicar datos a más de un tema.

- **Agente** (Kafka broker): este es el servidor principal de Kafka, que se ejecuta en una máquina dedicada. Los mensajes son empujados al agente por el productor. El agente se encarga de los temas en diferentes particiones y estas particiones se replican a diferentes agentes para lidiar con los posibles fallos. Es de naturaleza "apátrida", por lo tanto, el consumidor tiene que rastrear el mensaje que consume.

- **Consumidor:** el consumidor obtiene el mensaje del agente de Kafka. Recuerda que como ya se ha mencionado, es él quien busca los mensajes. El broker de Kafka no está enviando mensajes al consumidor. Más bien es el consumidor quien está extrayendo los datos del Kafka broker (o agente Kafka). Los consumidores están suscritos a uno o más temas en el Kafka broker y ellos leen los mensajes.

El agente también realiza un seguimiento de todos los mensajes. Los datos se conservan en el intermediario (bróker) durante un tiempo especificado. Si el consumidor tuviese algún fallo, podría recuperar los datos después de reiniciarse.

La siguiente imagen pertenece al libro Spark for Dummies de IBM, y muestra muy bien el flujo de mensajes de Apache Kafka. El productor publica mensajes a los temas. Luego, el consumidor extrae datos del agente. Entre la publicación y la extracción, el mensaje es persistido por el agente de Kafka.




**ENLACE DE INTERÉS**

Para más información acerca de Apache Kafka, puedes visitar las siguientes webs:

- https://kafka.apache.org/documentation/
- https://kafka.apache.org/quickstart

# 4. REPRESENTACIÓN DE LOS DATOS

*Eres consciente de que son muchas las personas diferentes las que van a ser usuarias de tu labor. Además, también eres conocedor que entre los funcionarios hay profesionales muy diferentes y que entre ellos presentan distintos perfiles, cualificaciones, experiencias, edades, etc.*

*Por todos estos motivos no dudas que los aspectos de representación y visualización de los datos serán clave para el éxito del proyecto, porque de otro modo, muchos usuarios abandonarían el uso de los datos sin haberlos utilizado, bien por falta de tiempo, pereza o dificultad.*

*En esta etapa del proyecto debes resolver dos cuestiones. La primera será definir el modo o modos de visualización y representación que necesitarás aplicar. Es probable que por la gran heterogeneidad que ya se ha mencionado de los perfiles usuarios sea necesario diseñar varias representaciones diferentes, por ejemplo, una para los funcionarios de tipología más técnica y otra para los funcionarios de carácter más estratégico o político.*

*La segunda cuestión es determinar qué tecnología te resultará más conveniente para cubrir todas tus necesidades de visualización. Probablemente alguna de las librerías más populares sean una opción a tener en cuenta.*

Stephen Thomas es uno de los principales expertos a nivel mundial en visualización de datos. En uno de sus libros más populares, titulado Visualización de Datos con Javascript, expone que se está haciendo difícil ignorar la importancia de los datos en nuestras vidas.

Los datos son críticos para las organizaciones más grandes y con más usuarios de la historia como, por ejemplo, Facebook y Google. Pero al mismo tiempo, también es verdad que es fácil ignorar los datos por la gran abundancia de los mismos, lo que provoca que todos estemos saturados de ellos en mayor o menor medida.

Una estimación de Stephen Thomas dice que el 99,5% de los datos que los sistemas informáticos capturan y almacenan, se desperdician. Por este motivo, la adecuada visualización de datos es una herramienta que aborda este problema porque con diferentes herramientas visuales los datos se pueden hacer más atractivos, más llamativos e, incluso, más fáciles de entender.

**BIBLIOGRAFÍA RECOMENDADA**

Thomas, S. A. (2015). *Data visualization with javascript.* New York: No Starch Press.

## 4.1 La visualización de los datos

Las visualizaciones efectivas sirven para aclarar las ideas en general, porque transforman colecciones abstractas de números en presentaciones alternativas que los espectadores captan y entienden rápidamente. Las mejores visualizaciones son las que consiguen hacerse entender de forma intuitiva. Es decir, que quien está viendo los datos, los comprenda sin que nadie se los explique.

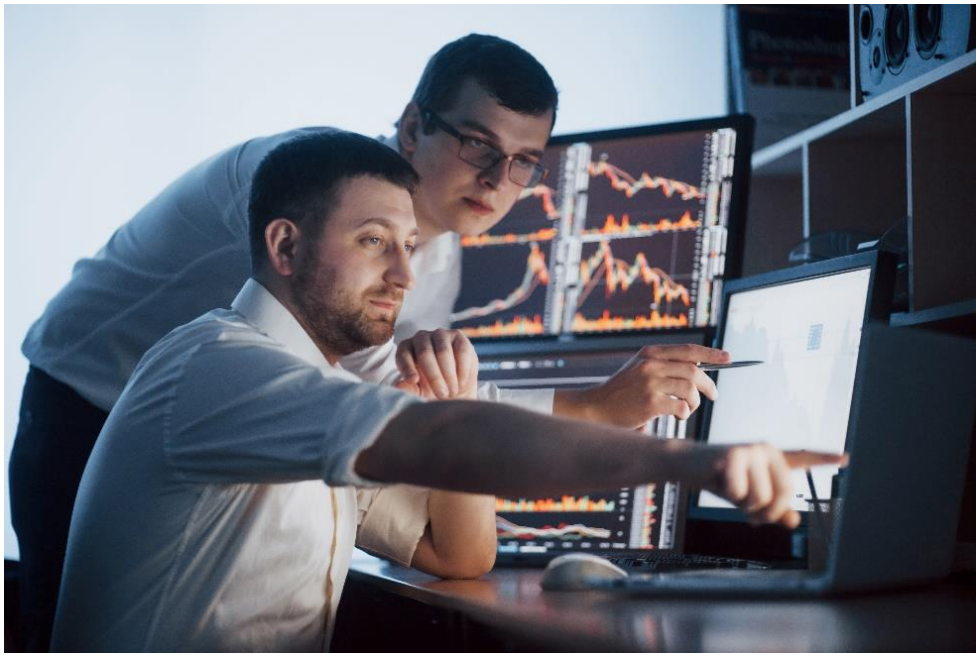
Según Thomas: "Los espectadores comprenden los datos inmediatamente, sin pensar. Esto libera a los espectadores para considerar más plenamente las implicaciones de los datos: las historias que cuentan, las percepciones que revelan, o incluso las advertencias que ofrecen".

En la actualidad, son muchos los roles informáticos que tienen que lidiar con la presentación y visualización de datos. Por ejemplo, los desarrolladores de sitios o aplicaciones web es probable que tengan que presentar a sus usuarios algún tipo de datos.

Lo mismo ocurre con quien diseñan o realizan informes empresariales, quienes programan software de gestión empresarial, los analistas de negocio o los científicos de datos. Para prácticamente todas las profesiones informáticas, la representación y visualización de datos forma parte de sus tareas y responsabilidades.

Para hacer una buena labor hay que saber qué tipo de visualización es la apropiada en cada caso. Eso es lo más importante. Y, posteriormente, habrá que ser capaz de crearla o buscar ayuda en quien sepa hacerlo. Aunque la realidad es que una vez se ha tenido la idea, la ejecución de esta es cada vez más sencilla porque, tanto las aplicaciones de análisis de big data como los lenguajes de programación, tienden a incorporar módulos y librerías respectivamente, que los dotan de capacidades potentes, a la vez que sencillas de aplicar en materia de visualización.

Existen docenas de diferentes visualizaciones, técnicas y herramientas, cada una idónea para una necesidad.



Visualizando datos

Fuente: https://www.freepik.es/foto-gratis/equipo-corredores-bolsa-teniendo-conversacion-oficina-oscura-pantallas-visualizacion-analisis-datos-graficos-e-informes-fines-inversion-comerciantes-creativos-trabajo-equipo_9277155.htm#page=1&query=visualizacion%20de%20datos&position=2

## 4.2 Principios en la representación de datos

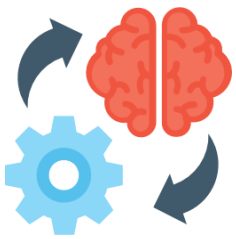
Stephen Thomas recomienda seguir cuatro principios fundamentales cuando se habla de representación de datos:

- **Implementación vs. Diseño:** hay que distinguir entre diseñar visualizaciones de datos e implementación de las visualizaciones. No todo el mundo necesita saber diseñar y puede ser suficiente con la capacidad de implementar ideas de otros.

- **Código vs. Estilo:** hay que distinguir también entre cómo utilizar correctamente el lenguaje de programación elegido para crear visualizaciones y la parte de estilo, que normalmente depende de aspectos como HTML, CSS, etc. Obviamente, contar con experiencia, aunque sea básica y conocimientos de HTML y CSS será útil cuando se agreguen visualizaciones a las páginas web.

- **Simple vs. Complejo:** las visualizaciones complejas pueden ser atractivas y convincentes, pero requieren de aprender mucha programación avanzada que a lo mejor no estamos en disposición de alcanzar. En muchas ocasiones, una buena idea sencilla es suficientemente atractiva. Simple no significa aburrido, e incluso las visualizaciones más simples pueden ser esclarecedoras e inspiradoras.

- **La realidad vs el mundo ideal:** cuando se empieza a construir visualizaciones, se descubre que el mundo real rara vez es tan amable como se desea. Las bibliotecas de código abierto tienen errores, los servidores de terceros tienen problemas de seguridad y no todos los usuarios se han actualizado al último y mejor navegador web. Por eso hay que aspirar a que nuestras visualizaciones sean resilentes e, incluso en las peores circunstancias, sean visualizadas tal y como las hemos diseñado.



**RECUERDA**

Existen multitud de herramientas para visualización de datos y debemos elegir la más adecuada para el objetivo de alcanzar una mejor comunicación.

**ARTÍCULO DE INTERÉS**

En el siguiente artículo se muestran la gran cantidad de herramientas que existen para la visualización de datos.

https://blogthinkbig.com/herramientas-para-la-visualizacion-de-datos

# 5. PROCESO DE ETL. DEL DATO A LA INFORMACIÓN

*Entre todos los CV descartados en primera instancia para la plaza a la cual estaban concurriendo has pensado en someterlos a un proceso de clusterización, de forma que se encuentren grupos homogéneos de individuos sobre los cuales poder tomar decisiones. De este modo, estarías aplicando uno de los tipos de algoritmos más populares, como es el clustering o agrupamiento a un ámbito especialmente innovador, el de los recursos humanos.*

*Esto serviría para que el Gobierno tuviese un mejor conocimiento de la estructura del capital humano que está buscando una mejora en su empleo. Con ese conocimiento, se podría modificar el plan nacional de formación.*

*En este momento también tendrás que proponer el diseño de un dashboard, que es la herramienta de visualización de datos más crítica de todas, porque está orientada a tomadores de decisiones de alto nivel. Si finalmente en el proyecto se ha optado por apoyarse en una plataforma como Spark, podría construirse el dashboard sobre ella.*

*De igual modo, si hemos optado por una plataforma como Stark, también será posible beneficiarse de las ventajas de contar con las capacidades propias de ETL.*

En este apartado nos centraremos en ver cómo se integran los datos para la inteligencia de negocios y en comprender el diseño de procesos ETL, que significa extracción, transformación y carga.

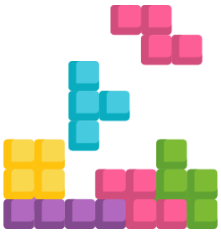
Todos somos conscientes de que en un mundo cada vez más globalizado e hiperconectado, las técnicas para integración de datos son variadas. Vamos a conocer tres tipos:

- **Propagación de datos:** este método consiste básicamente en realizar copias de los datos desde un lugar de origen determinado a otro entorno remoto. Estos datos son extraídos mediante programas que crean los ficheros en blanco y van suministrando la copia de los datos al destino final. Esta propagación puede, a su vez, dividirse en: distribución e intercambio bidireccional.

- **Consolidación de datos:** tal y como indican Josep Curto y Jordi Conesa, "en este caso la integración se realiza mediante una captura de los cambios en los orígenes de datos definidos y que serán consolidados y llevados a un solo lugar de destino que conserva la copia de todos estos datos".

- **CDC (Change Data Capture):** esta técnica es útil cuando se requiere realizar la captura de los cambios producidos en los datos en un origen determinado, los cuales, serán propagados y/o almacenados en diversos entornos de destino. Así mismo, este método tiene 4 técnicas de CDC comúnmente utilizadas:

    - CDC por aplicación: la aplicación por si misma actualiza los cambios en el origen de los datos.
    - CDC por timestamp: los datos de origen se van a incorporar por un periodo de tiempo definido, leyendo la última vez que se realizó la actualización, por ejemplo, en una tabla relacional.
    - CDC por triggers: son acciones que se ejecutan como disparadores cuando, por ejemplo, se realiza un update, insert o delete en sentencias o consultas de SQL.
    - CDC por captura de logs: se trata de establecer un programa de inspección constante al fichero de log de la base de datos para localizar los cambios más recientemente insertados.



**EJEMPLO PRÁCTICO**

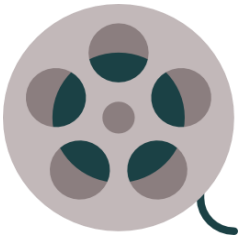
Tenemos una base de datos en la que hay una tabla con las facturas de una empresa. Esta tiene una columna Id_Cliente, que es la clave primaria, y un campo llamado Id_Factura. Si se realizara una modificación en el cliente con Id_Cliente=22 en el campo Id_Factura, ¿cómo podemos ver los cambios de manera que se pudiera comparar el valor antiguo y el nuevo del campo Id_Factura?

**SOLUCIÓN:**

CDC nos permite recoger y almacenar los datos que cambian para una tabla de forma que conservaremos un histórico con los datos y podremos comparar los cambios.

Hay diferentes tipos de aplicaciones de las tecnologías y métodos de ETL a proyectos de BI. Todo debe ser evaluado desde una perspectiva particular y personalizada de acuerdo con la tipología del negocio. Algunos de los usos de ETL son:

- **Proceso de calidad y refinamiento de datos:** técnica para depuración y mejoramiento de datos para análisis.
- **Master data management:** método usado para definir y administrar datos sensibles.
- **Customer data integration:** integración y manipulación de datos del cliente proveniente de diversas fuentes.
- **Migración de datos:** traslado de datos entre fuentes o ambientes de bases de datos, por ejemplo, durante una implementación de un software de BI o ERP.



**VIDEO DE INTERÉS**

A continuación podrás visualizar un vídeo en el que se compara cinco de las herramientas de ETL más populares en el mercado actual.

https://www.youtube.com/watch?v=ly2O_I3stls

La integración de datos es un elemento clave cuando se trata de implementar un modelo de inteligencia de negocio. En la integración de los datos, siempre debemos tener presentes cuatro dimensiones:

- Integración de datos que muestre una visión global de los datos de una organización.
- Integración de procesos de negocio que proporcione una visión unificada e integrada de todos los procesos de negocio.
- Integración de aplicaciones que proporciona una visión global unificada de todas las aplicaciones internas y externas de la empresa.
- Integración de la interacción de usuarios. Esta integración es determinante pues aporta una interfaz segura y adaptada a los usuarios.


Dimensiones de la integración de los datos
Fuente: Elaboración propia

Para explicar las funcionalidades básicas de ETL encontramos cinco elementos centrales:

- Gestión y administración de servicios.
- Extracción de datos.
- Transformación de datos.
- Carga de datos.
- Gestión de datos.

Debido a las diferencias que existen entre los tipos de datos y su gestión, un primer punto que debemos tener claro es cómo definimos la integración de los datos. Según diversos autores se entiende por integración de datos "al conjunto de aplicaciones, productos, técnicas y tecnologías que permiten una visión única y consistente de nuestros datos de negocio."

Partiendo de esta definición, veamos sus componentes, las técnicas, herramientas y aplicaciones principales. Existen varios tipos de tecnologías orientadas a la integración de datos de tipo ETL:

- **Herramientas ETL de generación de código.** Disponen de un entorno en el que se crean y modifican las interacciones, así como las transformaciones necesarias antes de enviar los datos a los entornos de destino.

- **Herramientas ETL basadas en motor.** Hacen un mapeo y crean flujos de trabajo usando herramientas gráficas. Se dividen en:

    - Motor de extracción: realizan un pull de los datos mediante adaptadores ODBC, SQL, JDBC, entre otros, con técnicas de propagación en línea y luego hacer un push para la consolidación en destino.
    - Motor de transformación: funciona con una librería de objetos para transformar los datos de origen y adaptarlos a los destinos determinados.
    - Motor de carga: los datos se cargan de forma masiva para actualizar el entorno de destino.

- **Herramientas ETL incluidas en la base de datos.** Algunos fabricantes incluyen, dentro del motor de la base de datos, funcionalidades ETL. Pueden ser:

    - ETL cooperativos.
    - ETL complementarios.
    - ETL competitivos.

- **EII (Enterprise Information Integration).** Busca que las aplicaciones puedan acceder a los datos que están dispersos, desde data marts a un fichero de texto.

- **EDR:** su objetivo es detectar, de forma sistemática, los cambios que se han realizado en el origen de los datos. Opera con la técnica de propagación de datos y el change data capture (CDC).

    - Programa de captura: recupera los cambios que se han hecho en la base de datos de origen.
    - Sistema de transporte: normalmente, estos sistemas se hacen a través de stagging areas o tablas de datos de paso.
    - Programa de aplicación de cambios: este programa opera básicamente en dos formas. Lee mediante consultas SQL las tablas de stagging o lee el sistema de colas de las replicaciones y con otras tablas almacenadas controla y realiza un mapeo de los datos de origen y destino.

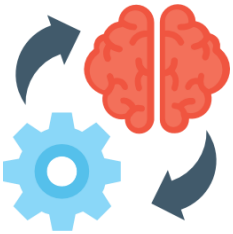
## 5.1 Análisis y creación de algoritmos

Primero, es importante tener una definición aproximada del concepto de algoritmo y algunas de sus características para que sea fácilmente entendible y posteriormente se comprenda con mayor facilidad sus aplicaciones.

En base a lo anterior diremos que los algoritmos tienen diferentes usos y aplicaciones, pero en la inteligencia de negocios y la explotación de datos se emplean básicamente para identificar tendencias o patrones de comportamiento que permitan tomar decisiones entorno a un producto o servicio determinado, y detectar mejoras o anticipar situaciones que puedan ser críticas para la organización.

Cuando un algoritmo se aplica en un contexto con un objetivo predictivo, habrá de elegirse bien el tipo de algoritmo a utilizar, pues todos los existentes no tiene propiamente la misma función y, en base a ello, se clasificarán en tres grandes grupos:

- Algoritmos para resolver problemas de negocio.
- Algoritmos para optimizar otros algoritmos.
- Algoritmos para obtener mejoras en las capacidades.



**RECUERDA**

En el contexto de la gestión de datos, un algoritmo es un conjunto de instrucciones insertadas en un programa de forma lógica que permitirá analizar un conjunto de datos y establecer un punto de salida o destino.

Los algoritmos más populares que se usan en un contexto predictivo, con el fin de encontrar o predecir situaciones que permitan tomar las decisiones apropiadas y en el momento oportuno, son:

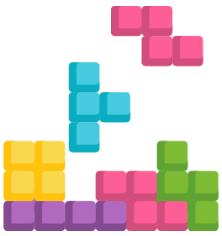
- **Algoritmos de agrupamiento:** los algoritmos de agrupamiento, o en ingles clustering, son un procedimiento con el cual se busca agrupar distintos tipos de datos con base a un criterio o variable propiamente definida. Con frecuencia se ejecutan basados en un criterio de distancia o similitud. Por ejemplo, la distancia entre grupos de datos o por similitud, basándose en un aspecto concreto del dato, por ejemplo, todos los pacientes con sobrepeso.

- **Algoritmos de clasificación:** un algoritmo de clasificación básicamente consiste en un método de análisis basado en muestras previamente conocidas, que buscan clasificar o indicar a que grupo debe pertenecer el nuevo dato basado en la característica de la muestra previamente definida.

- **Algoritmos de regresión:** la regresión lineal es uno de los algoritmos más frecuentes para análisis avanzado. Los usuarios pueden visualizar fácilmente cómo funciona y cómo se relacionan los datos de entrada con los datos de salida.

  La regresión lineal utiliza la relación entre dos conjuntos de medidas cuantitativas continuas. El primer conjunto se llama predictor o variable independiente. La otra es la respuesta o variable dependiente. El objetivo de la regresión lineal es identificar la relación en forma de una fórmula que describe la variable dependiente en términos de la variable independiente. Una vez que esta relación se cuantifica, la variable dependiente se puede predecir para cualquier instancia de una variable independiente.

Una de las variables independientes más comunes utilizadas es el tiempo. Si la variable independiente que nos interesa explotar son ingresos, costes, clientes, uso o productividad y podemos definir la relación que tiene con el tiempo, entonces podemos pronosticar un valor con regresión lineal.



**EJEMPLO PRÁCTICO**

Dentro del departamento de laboratorio farmacéutico o de una compañía de alta tecnología, hay gran cantidad de personal científico y técnico, con perfiles profesionales muy complejos, definidos por múltiples parámetros como son:

- Universidades donde han cursado su grado y postgrado.
- Centro y laboratorios de investigación de los cuales han sido miembros.
- Académicos y científicos con quienes han colaborado a lo largo de su trayectoria.
- Empresas en las cuales han trabajado.
- Becas y programas de financiación que han recibido.
- Revistas y diarios científicos en los cuales han publicado.
- Premios y reconocimientos obtenidos.

Estos perfiles, por su alta cualificación, son muy difíciles de gestionar y cuesta retener ese tipo de talento. ¿Qué tipo de agrupamiento establecerías?

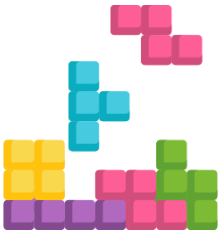
**SOLUCIÓN:**

Si realizamos un **proceso de clusterización** de todos los profesionales de la empresa, será posible encontrar grupos con características homogéneas que permitiría una gestión del talento más personalizada.

Por ejemplo, se encontrarían agrupaciones formadas por profesionales individualistas, cuya trayectoria ha estado siempre alejada de las grandes empresas. Por el contrario, se encontrarían otros clusteres de trabajadores con una componente social acentuada, que han formado parte de eventos masivos y han participado de estructuras directivas en asociaciones sectoriales, centros académicos, etc.

Descubriendo cosas como esas, los primeros podrían recibir más autonomía para que se encuentren más cómodos en la compañía y que puedan desarrollar su individualismo en beneficio de la empresa, mientras que los segundos podrían recibir una formación complementaria en comunicación y marketing, para que pudiesen dedicarse a tareas comerciales.

De este modo, todos se encontrarían más cómodos y mejor aprovechados dentro de la empresa.



**EJEMPLO PRÁCTICO**

Una empresa está valorando el pagar a un grupo de empleados la matrícula en un prestigioso máster. Obviamente, la empresa quiere que los trabajadores se esfuercen, aprovechen el máster y lo aprueben, porque de otro modo sería tirar el dinero.

¿Cómo crees que los algoritmos pueden ayudar a la empresa a superar esa incertidumbre?

**SOLUCIÓN:**

Aplicar la regresión lineal para estimar la probabilidad de que un grupo de empleados aprueben un máster es posible. Para ello, hay que conocer qué relación se da entre distintas variables. Pensemos, por ejemplo, que el examen del máster se realiza en forma de test. Si son pruebas de tipo test con cuatro opciones a elegir, podemos decir que hay una entre cuatro posibilidades de acertar la respuesta.

Pero, por otro lado, también podríamos decir que según las horas de estudio que cada trabajador ha dedicado a preparar el máster se produce un incremento de probabilidad de aprobar por cada hora extra de estudio.

De esta forma se establece una correlación que es el nivel de dependencia que existe entre ambas variables. Cuando su valor es más próximo a 1, se trata de una dependencia más profunda. Mientras que, cuando tiende a -1, su relación es inversa.

Si disponemos de datos de los trabajadores que han hecho ese mismo máster en años anteriores podremos estimar el número de horas libres que debemos dar a cada empleado para que estudien con altas probabilidades de aprobar el máster.

### *5.1.1 Recomendaciones a la hora de usar algoritmos*

El concepto de algoritmo puede, en principio, parecer algo complejo de comprender. Debemos ser conscientes de que para operar sobre un conjunto de datos de forma predictiva, tanto un profundo conocimiento del negocio como de los distintos tipos de algoritmos, pueden usarse en cada sector (financiero, comercial, marketing, logística de distribución entre otros).

A continuación, se presentan diez consejos que pueden ser un vector de dirección para usar los algoritmos en un entorno empresarial:

- Definir un responsable de alto nivel que determine las expectativas y tenga en mente al cliente, así como los objetivos estratégicos. Que sea responsable de la inversión y el retorno de esta.

- Listar e identificar los objetivos de negocio a atacar. Un modelo predictivo exitoso, que aplique de modo efectivo los algoritmos, debe basarse en dos elementos. El primero, es el nivel de precisión en las predicciones. El segundo, es la relevancia para el negocio. Por lo anterior, es clave elegir los procesos a involucrar y los resultados esperados.

- Evaluar el impacto en el negocio con el fin de aprovechar las verdaderas oportunidades de negocio o de mejora potenciales.

- Detallar y documentar las métricas de rendimiento y/o desempeño. Los modelos definidos deben tener un impacto cuantificable en los objetivos de la organización.

- Seleccionar un equipo de trabajo competente que básicamente debe constituirse como un equipo multidisciplinario con amplios conocimientos técnicos en materia estadística, análisis de datos e ingeniería de datos, entre otros.

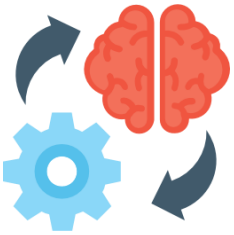
- Establecer una metodología de proyecto y desarrollo de modelos. De una u otra forma, el éxito de una estrategia algorítmica no solo depende de las herramientas y técnicas dispuestas, sino que también debe adoptarse una metodología estructurada que permita desarrollar los modelos predictivos en un tiempo competitivo antes de que se pierda la oportunidad de negocio.

- Asegurar la disponibilidad y accesibilidad de datos requeridos. El big data es clave, por lo cual, la capacidad de recopilar y almacenar datos y, así mismo, procesarlos, es fundamental a la hora de trabajar con algoritmos.

- Instituir un sistema de gobierno de datos. Dada la variedad, volumen y velocidad de crecimiento de los datos es necesario establecer un sistema efectivo de gobierno de datos que garantice la calidad de estos.

- Planificar acciones de mejora. Adaptarse al cambio, de forma ágil, es fundamental, con el fin de mejorar la habilidad de uso de algoritmos.

- Elegir las herramientas apropiadas. En el mercado hay un sin fin de herramientas, tanto de código abierto como de pago. En cualquier caso, deben elegirse las herramientas idóneas, desde la perspectiva de satisfacción de necesidades, capacidad de aplicación y escalado potencial.



**RECUERDA**

Existen retos importantes a la hora de trabajar con datos. De cara al crecimiento acelerado de su uso, se deben considerar estos retos.

**CITA**

"*Estamos escribiendo un código que no sabemos leer, con unas consecuencias que no podemos controlar*." Kevin Slavin, profesor en el MIT Media Lab.

## 5.2 Dashboards como herramienta de visualización

Los dashboards son herramientas muy útiles para desarrollar la actividad de control de gestión. Con frecuencia, en nuestra vida cotidiana, hablamos de situaciones que podemos tener o no controladas, que están condicionadas por la sensación de dominio que tenemos de las mismas. En el ámbito empresarial también podemos sentir que controlamos o no la empresa.

Sentimos que la controlamos cuando la dominamos y ejercemos la acción de control y, por el contrario, no lo sentimos cuando no sabemos con precisión lo que está pasando en la empresa y su entorno.

Observando un dashboard
Fuente: https://www.freepik.es/foto-gratis/hombre-negocios-mostrando-grafico-tablero_1128217.htm#page=1&query=dashboards&position=0

**VIDEO DE INTERÉS**

En ocasiones no es necesario disponer de una herramienta compleja y costosa para crear un dashboard, y será posible crearlo con herramientas gratuitas como Google Data Studio.

https://www.youtube.com/watch?v=RtbVBeu3gtk

### *5.2.1 Los indicadores en un dashboard*

El diseño de un dashboard, para que se convierta en una estructura de control eficaz de los distintos departamentos de la empresa, requiere definir un conjunto de indicadores de control o unidades de medida.

Esto se lleva a cabo en función de la identificación de las variables o factores clave de la empresa en su conjunto y de cada departamento en particular. Un factor clave es aquella variable o característica que se considera crítica para el éxito de la empresa a largo plazo, permitiéndole aumentar y mantener su ventaja competitiva. De manera que, si orientamos nuestros esfuerzos hacia esas variables o factores clave, estaremos contribuyendo a alcanzar los objetivos estratégicos.

Un sistema de indicadores nos debe proporcionar información acerca del grado de consecución de los objetivos, pero también sobre cómo se están logrando dichos objetivos, por lo tanto, cuanto más clara tenemos nuestra estrategia, más sencillo resulta su diseño e implantación. Enlazar los indicadores con nuestra estrategia, que será uno de los principales objetivos de nuestro sistema de indicadores, se puede conseguir gracias a la descomposición de los factores clave de negocio en los objetivos estratégicos y desglosando estos en las diferentes actividades y procesos que realizamos para alcanzarlos.


Podemos identificar como principales características de los factores clave, las siguientes:

- Pueden explicar nuestro éxito o fracaso.
- Tienen un impacto significativo en la cuenta de resultados.
- Son representativos de los cambios del entorno.
- Cuando hay cambios en el factor se derivan acciones inmediatas.
- Son cuantificables de forma directa o indirecta.

Además, los factores clave pueden tener un carácter interno o externo en función de si los encargados de la toma de decisiones pueden influir en ellos. Una característica importante de estos factores es que, por lo general, son factores internos y su cumplimiento depende de las decisiones de los directivos de las empresas. Los factores externos son los que están originados en el entorno.

Atendiendo a su ámbito, podemos identificar los siguientes tipos de factores clave:

- **Factores clave derivados del entorno:** tienen un carácter genérico y su delimitación está relacionada con la extensión de los mercados en los que opera la empresa. En este sentido, aspectos como la coyuntura económica, las legislaciones laborales, las políticas de empleo, etc., pueden convertirse en factores críticos para nuestra empresa, porque no podemos controlarlos y debemos adaptarnos a ellos en la medida que podamos.

- **Factores clave derivados de la industria y del sector:** cada sector cuenta con un conjunto de factores que definen su estructura y características. En algunos sectores es necesario controlar especialmente el riesgo, en otros los costes…, de manera que para cada sector habrá que identificar cuáles son aquellos aspectos especialmente relevantes.

- **Factores clave derivados de la posición competitiva y de la estrategia elegida:** hay que tener presente que nuestra empresa ocupa una determinada posición competitiva dentro del sector en el que actúa. Nuestra posición, así como nuestra forma de competir, determinarán nuestros factores clave, del que su grado de prioridad o su importancia de cara a la competitividad y la rentabilidad pueden ser diferentes de los de nuestros competidores, porque dependen de la estrategia elegida por cada empresa.

- **Factores clave temporales:** son aquellos factores importantes para el éxito de la organización durante un periodo de tiempo concreto, y pueden ser de carácter estratégico u operativo. Algunos ejemplos de factores clave temporales pueden ser la sustitución masiva de directivos, la implantación de nuevas políticas, la implantación de nuevos métodos de trabajo, el control de ciertas partidas del balance, la cuenta de resultados o la introducción de nuevas tecnologías.

Otras formas de clasificar los factores clave es relacionarlos con las diferentes funciones o áreas críticas en que la empresa divide sus actividades. A veces estos factores tienen que ver con las habilidades necesarias para determinadas estrategias o simplemente con la capacidad de organización de la empresa.

Los factores clave afectan a todos los niveles de la organización, de manera que su análisis puede proporcionar muchas ventajas cuando se está gestionando una empresa y al utilizarlos como punto de partida del sistema de indicadores. Estas son algunas de las ventajas que podemos destacar:

- Contribuye a configurar nuestra posición competitiva desde un punto de vista externo e interno.
- Permite relacionar la estrategia con factores concretos que es necesario medir.
- Obliga a buscar indicadores de rendimiento y resultados por cada factor clave.
- Orienta y centra el trabajo directivo.
- Define la cantidad y calidad de la información necesaria para la toma de decisiones.

- Los factores clave permiten alcanzar los objetivos estratégicos de manera que ayuden en el proceso de planificación.

Pero no todo son ventajas. El uso de los factores clave como base del sistema de indicadores también plantea problemas y limitaciones como los siguientes:

- Los sistemas contables con frecuencia no ofrecen información que pueda resultar relevante para el seguimiento y control de los factores críticos.
- Parte de la información que es necesario controlar no se encuentra en la empresa: entorno, estructura de los mercados, competidores…
- Reunir la información necesaria para controlar un factor clave puede implicar coordinar diversas fuentes de información distribuidas por la empresa.
- La medición de algunos factores en ocasiones puede necesitar juicios subjetivos que completen su marco de información.

Nuestro dashboard debe aportar información acerca de la consecución de los objetivos y el grado en el que se están alcanzando, de manera que es necesario que los objetivos estratégicos se identifiquen y definan de la forma adecuada. Por este motivo, los objetivos estratégicos deben cumplir los siguientes requisitos:

- Que se enuncien en forma de acciones y/o cambios.
- Que se enuncien de manera que puedan ser medibles.

Al definir los objetivos como acciones, eso nos lleva al análisis de la cadena de valor y, por lo tanto, al análisis de los procesos y actividades de la empresa. La cadena de valor como herramienta para el control de la gestión implica considerar una serie de factores clave internos que van a definir la eficacia y eficiencia de la empresa, y la elección de un sistema de indicadores basado en el análisis de las operaciones y de los procesos y actividades.


Ejemplo de dashboard de factores clave en marketing online
Fuente: renaud.es

Por lo tanto, podemos decir que los procesos y actividades muestran lo que está haciendo nuestra empresa para generar valor añadido para los consumidores, mientras que los indicadores nos aportan información acerca de cómo lo estamos haciendo. Lo ideal es utilizar indicadores de rendimiento –y no indicadores que analicen tan solo los resultados- que nos expliquen lo que está ocurriendo en cada momento, porque así conseguimos cierto carácter preventivo que nos permite aplicar las medidas correctoras oportunas.

Existen algunos factores clave que son comunes a todas las empresas, como la calidad, la flexibilidad o la productividad, aunque éstos pueden tomar formas diferentes en cada compañía.

Es importante destacar que es necesario que en todos los niveles de la empresa haya un entendimiento común de los factores clave, de manera que todos puedan aplicar las mejores prácticas que ayuden a su cumplimiento. Además, debemos tener presente que la elección de los indicadores estará condicionada por la manera en que quedan definidos los factores clave.

Cuando ya hemos analizado los factores clave y los objetivos estratégicos, el siguiente paso es fijar y elegir los indicadores que utilizaremos.

| FACTOR CLAVE | DEFINICIÓN DEL FACTOR CLAVE | EJEMPLO DE INDICADOR |
|---|---|---|
| Flexibilidad | Capacidad de respuesta a las variables y exigencias de los clientes | - Plazos de entrega.<br>- Nº de referencias no estándar. |
| Productividad | Eficacia con que se gestionan los recursos, incluido el tiempo | - Tiempo de proceso/tiempo total.<br>- Productividad de la M. O. |
| Calidad | Cumplimiento de las expectativas de los clientes internos y externos | - Calidad a la primera.<br>- Nº de reclamaciones internas |

Ejemplos de factores clave internos y su definición como punto de partida para establecer indicadores
Fuente: Elaboración propia

### *5.2.2 ¿Cómo elegir los indicadores de un dashboard?*

Un indicador es un instrumento de medida que permite representar la dimensión teórica de un factor clave, siendo el factor clave lo que necesitamos medir o evaluar y el indicador la forma en que vamos a hacerlo. Es decir, el indicador nos permite evaluar la evolución o, incluso, consecución de un factor clave, teniendo en cuenta que esta medición será más fácil o difícil dependiendo de qué factor se trate.

Aunque en algunas ocasiones un factor clave y un indicador pueden coincidir, siempre serán considerados como conceptos diferentes. Por ejemplo, la cuota de mercado de una empresa puede ser al mismo tiempo un factor clave y un indicador.

También es conveniente señalar la diferencia entre ratio e indicador. Toda ratio será siempre un indicador, pero **no todos los indicadores tienen que representarse en forma de ratio**. Las ratios son relaciones entre dos magnitudes que suelen tener un significado más amplio de forma conjunta que al analizarlas por separado. Un ejemplo de esto lo tenemos en la rentabilidad sobre ventas: las ventas y los beneficios son indicadores por sí solos y su cociente es la ratio de rentabilidad sobre ventas.

Los indicadores de gestión pueden agruparse de diversas maneras, dependiendo de los criterios que utilicemos para clasificarlos y se pueden atribuir a varias clasificaciones al mismo tiempo. A continuación, exponemos una posible manera de agruparlos:

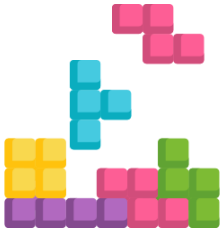
- **Internos y externos.** Los primeros miden la evolución de variables que se producen en el interior de la empresa, por lo que responden a factores clave internos y permiten medir la competitividad interna de la compañía. Los indicadores externos hacen referencia a variables que se producen fuera de la empresa, como la competencia o la evolución de los tipos de interés.

- **Cuantitativos y cualitativos.** Un indicador cuantitativo es el que mide los factores clave de manera numérica, siendo el caso de la mayoría de los indicadores financieros. Los indicadores cualitativos se basan en la descripción cualitativa de la situación o en los atributos que distinguen a una situación o variable determinada. Un ejemplo típico de indicador cualitativo es la medida de satisfacción de los clientes (muy satisfecho, satisfecho, poco satisfecho, nada satisfecho).

- **Monetarios y no monetarios.** Los indicadores monetarios miden el resultado del comportamiento de un factor clave en términos monetarios como magnitud de medida (ventas, costes, gastos, inversión…). Un indicador no monetario mide la actuación en otros términos (por ejemplo, el número de días que la empresa tarda en entregar los pedidos a sus clientes).

- **Financieros y no financieros.** Los indicadores financieros relacionan la evolución del factor clave con la situación económica o la financiera (rentabilidad, liquidez…). Se obtienen a través de la contabilidad. Los indicadores no financieros miden otros aspectos como la calidad de los productos.

- **De resultado e indicadores de proceso.** Esta función se establece en función del momento en el que se mide el factor clave. Así, un indicador de proceso informa sobre lo que pasa cuando el proceso aún está en marcha y los de resultado cuando ya ha finalizado. La mayoría de los indicadores de proceso utilizan la variable tiempo como elemento de medida. Los indicadores de proceso generalmente se relacionan con la eficiencia y los de resultado con la eficacia en la consecución del objetivo.



**EJEMPLO PRÁCTICO**

Te encuentras desarrollando el software para la gestión de una cadena hotelera. Por este motivo, te han solicitado que realices una propuesta de indicadores que dicho software permitirá controlar para así verificar por parte del cliente si están conformes con ellos o consideran que falta alguno. ¿Qué indicadores propondrías a esta empresa enfocada en el sector turístico?

**SOLUCIÓN:**

El indicador más utilizado en el sector hotelero a nivel mundial es el que mide la cantidad de ingresos por cada habitación disponible. Este indicador se suele denominar RevPAR, que en inglés significa Revenue Per Available Room, o lo que es lo mismo, ingresos por habitación disponible.

Este indicador claramente es de tipo monetario, puesto que se medirá en la misma moneda que la contabilidad del hotel, sean euros, dólares o la moneda del respectivo país.

También se trata de un indicador de tipo financiero, porque está referido a los ingresos de la empresa medidos por habitación durante un periodo de tiempo determinado, por ejemplo, los euros que se facturan por cada habitación de media al día o al mes o al año.

Otro indicador que recomendaría a la cadena hotelera es el de satisfacción del cliente, de forma que el director de cada hotel y el director de toda la cadena pudiese conocer, de forma muy sencilla y mediante este indicador, cuál es el grado de satisfacción de los clientes que pasan por el hotel. Este indicador se podría generar en base a las encuestas o cuestionarios que suelen rellenar los clientes cuando finalizan la estancia en cada hotel.

Este indicador es de tipo no monetario y no financiero. Igualmente, diremos que es un indicador de resultado y no de proceso, porque lo que mide es el resultado percibido por el cliente, pero no a que se debe esa satisfacción alta o baja.

Un ejemplo de indicador externo, es decir, ajeno al control de la propia cadena de hoteles, podría ser el que indicase el número de turistas que llegan cada mes a cada una de las ciudades donde tenemos un hotel.

### 5.2.3 Un caso especial de dashboard: el cuadro de mando

Tanto a nivel individual como a nivel de sistema, los indicadores deben reunir una serie de características en su diseño para que sean útiles como herramienta de gestión y de información respecto al desarrollo y evolución de la empresa.

Desde un punto de vista estratégico, los indicadores deben:

- Provenir y derivarse de la estrategia de la empresa para responder e informar de la evolución y consecución de los objetivos estratégicos. Para lograr esto es necesario enlazar la estrategia de la empresa con las operaciones y procesos a partir de los factores clave.
- Deben referirse a objetivos específicos, alcanzables y precisos y deben tener un propósito explícito.
- Mostrar lo que ocurre en los procesos de negocio. Los indicadores más eficaces y de mayor calidad se basan en un indicador de rendimiento que mide las causas de la eficiencia de los procesos críticos de la empresa.
- Los indicadores de proceso deben combinarse con los de resultado cuando el proceso ha terminado.
- Medir siempre aspectos relevantes relacionados con la actuación de la empresa.
- Reflejar la evolución de los procesos críticos.
- Ser fáciles de entender y comprender por toda la empresa. Todo el mundo debe conocer y saber interpretar los indicadores mediante los cuales es gestionada y controlada la organización.

El diseño de los indicadores debe presentar las siguientes características:

- Presentar un formato que resulte sencillo y que pueda mantener su forma con el paso del tiempo para hacerlos comparables. **El impacto visual del formato es importante, porque la información gráfica puede proporcionar más información que la numérica.**
- Basarse en magnitudes y cantidades que pueda controlar el usuario.
- Analizarse como una tendencia y no como una acción concreta y aislada. Los procesos de mejora continua se basan en asegurar una tendencia favorable en cada indicador.
- Ser consistentes en el tiempo.
- Evitar la subjetividad y no fundarse en opiniones.
- Estar diseñados, definidos y enfocados a las necesidades de los clientes internos y externos.
- Aportar información precisa y oportuna, porque son una parte fundamental del sistema de información para la dirección.
- Estar enfocados a la mejora.

- Los indicadores deben diseñarse como un sistema, de forma global. En el conjunto de indicadores debe haber coherencia jerárquica (los objetivos e indicadores establecidos sirven y se prolongan a los niveles inferiores) y transversal (los indicadores se alinean de acuerdo con los procesos).
- El sistema de indicadores debe actuar sobre el comportamiento de las personas de la organización, orientándolas a la consecución de los objetivos.

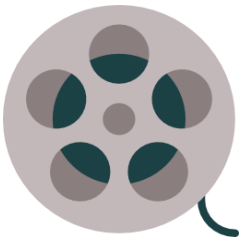
Como los indicadores deben establecerse con la máxima claridad, es necesario definir los elementos que intervienen en el diseño de un sistema de indicadores. Los elementos que, de manera genérica, debe reunir un sistema de indicadores pueden ser los siguientes:

- La denominación del indicador, que debe explicar por sí misma qué medida es y por qué es importante.
- El propósito al que responde, que normalmente está relacionado con algún factor clave de la empresa.
- El objetivo estratégico que queremos evaluar y controlar con el que está relacionado el indicador.
- El objetivo por conseguir debe explicar, de forma clara, el rendimiento que se quiere conseguir y una escala de tiempo para su consecución.
- La fórmula de medición, la cual, no debe crear disfuncionalidad, ni con otros indicadores ni entre los diferentes objetivos de la empresa.
- La frecuencia con la que el indicador debe ser contabilizado, medido y comunicado es en función de la importancia de la medida, del objetivo que cubre y del volumen de datos de los que disponemos. Si el indicador es de proceso, lógicamente, la frecuencia será mayor que si es de resultado.
- El responsable de recoger y comunicar los datos.
- La fuente de obtención de la información debe ser consistente y única para que el indicador sea fiable y comparable.
- El responsable del resultado que está midiendo el indicador.
- Las acciones a tomar cuando el resultado obtenido no es el esperado que puede incluir: el establecimiento de un grupo de mejora que identifique los motivos por los que el resultado no ha sido el esperado o, la fijación de un nuevo compromiso y la forma de llevarlo a cabo para reconducir los resultados.
- Las notas y comentarios que permitan explicar o justificar hechos que completen la información de los demás elementos.

| | |
|---|---|
| **Indicador** | Tiempo de respuesta de las reclamaciones |
| **Propósito** | Estimular la reducción continua de las reclamaciones para apoyar al plan de fidelización puesto en marcha |
| **Objetivo estratégico relacionado** | Conseguir un índice de fidelización del 90% para el ejercicio en curso |
| **Objetivo a alcanzar** | Atender un 95% de las reclamaciones de los clientes dentro de los dos días siguientes a que esta se produzca |
| **Forma de medición** | Fecha y hora de la comunicación del cliente – Fecha y hora del primer contacto con el cliente |
| **Frecuencia** | Semanal |
| **Responsable de recoger los datos** | Departamento de Atención al Cliente |
| **Fuente de obtención de la información** | Documentos de entrada que recojan el día y la hora de entrada de la reclamación – Documentos de salida que recojan el día y hora del primer contacto con el cliente |
| **Responsable de resultado** | Director de Dpto. de Atención al Cliente |
| **Acciones a emprender** | Reunión de grupo para determinar las causas de los retrasos producidos durante la última quincena. Rediseño del documento de salida |
| **Notas y comentarios** | Objetivo prioritario de la Dirección dentro de los objetivos del mes |

Ejemplo de hoja de diseño de indicadores
Fuente: Elaboración propia

**VIDEO DE INTERÉS**



En este enlace de vídeo podrás ver una explicación sobre qué es un cuadro de mando, al ser una herramienta muy utilizada para la gestión empresarial.

https://www.youtube.com/watch?v=l54CTD5qE74

A través de los indicadores de control y otro tipo de informaciones que nos permiten conocer el grado en el que se están cumpliendo nuestros objetivos, el cuadro de mando es el soporte que aporta información a los directivos de la empresa para la toma de decisiones. Hay que destacar que los cuadros de mando contienen más información además de la proporcionada por los indicadores, como pueden ser determinados tipos de información cualitativa sobre los competidores o el mercado.

Como ya se ha mencionado, el objetivo fundamental de los cuadros de mando es apoyar la toma de decisiones, pero, además, podemos identificar otros objetivos más específicos:

- Constituir el soporte operativo del control de gestión.
- Contribuir a la implantación de la Dirección por Objetivos (D.P.O.).
- Crear los mecanismos que permitan revisar los presupuestos en ejecución.
- Definir el grado de avance en la ejecución de los objetivos establecidos para un periodo de tiempo concreto.
- Desarrollar la formación directiva.

La utilización de cuadros de mando puede suponer las siguientes ventajas:

- Permiten llevar a cabo un seguimiento periódico de los objetivos, planes y presupuestos.
- Mantienen un estado de alerta permanente en la empresa sobre los factores clave y sus componentes.
- Ayudan a establecer pautas para corregir las desviaciones.
- Identifican las causas de las desviaciones y su incidencia en la evaluación del desempeño directivo.
- Actualizan las bases de información de los centros de responsabilidad de la empresa.
- Permiten realizar un control previo a la ejecución.

Las principales características de los cuadros de mando pueden resumirse en:

- Tratar de actuar en los puntos críticos de la gestión –derivados de los factores clave- para alcanzar los objetivos estratégicos.
- Profundizan en el análisis de los datos destacando las desviaciones producidas frente a sus previsiones, causas y repercusiones.
- Destacan las causas y responsables de las desviaciones críticas.
- Es una herramienta que fomenta el cambio a través del autocontrol y el reenfoque de los problemas.
- Trata solo información relevante, actual y fiable evitando detalles excesivos.
- Señala las acciones oportunas a tomar en cada momento.

Podríamos decir que el cuadro de mando es un sistema de información altamente formalizado, porque recoge de manera formal los hechos que se van produciendo en la empresa. En líneas generales, un cuadro de mando debe aportar información sobre los siguientes aspectos:

- El entorno y los competidores.
- Los costes y su comportamiento.
- Los presupuestos y su evolución.
- La rentabilidad de determinados objetivos (productos, clientes, mercados…).
- La evolución de los procesos.
- El control operativo de la empresa.

Además, podemos destacar una serie de requisitos que deben reunir las informaciones contenidas en los cuadros de mando:

- Es necesaria para la toma de decisiones.
- Es reciente y actualizada de acuerdo con las características operativas y del entorno.
- Es oportuna.
- Es fiable.
- Es suficiente y relevante.
- Es adecuada para medir la gestión de cada responsable.
- Es flexible cuando se producen nuevas necesidades o cambios.

La estructura de un cuadro de mando debe seguir un **criterio piramidal de la información**, teniendo en cuenta a los usuarios de este. Cuando el cuadro de mando está orientado a la alta dirección de nuestra empresa es conveniente que cumpla con las siguientes características:

- Contiene información de todas las áreas de la empresa.

- Destaca con claridad los hechos relevantes.
- Aporta datos y comentarios por áreas críticas de la empresa comparando datos, periodos e indicadores.
- Mide el nivel de cumplimiento de previsiones y objetivos fijados para cada periodo.

Si el cuadro de mando está destinado a la segunda línea de dirección debe ser más detallado, amplio y operativo, de manera que actúe como una verdadera herramienta de gestión a nivel de información y para la toma de decisiones. En este caso debería contener:

- Información del nivel de cumplimiento de los objetivos marcados para cada dirección o área de la empresa.
- Información del nivel de cumplimiento de los compromisos presupuestarios de cada área con sus desviaciones y causas.
- Información de los indicadores clave con el análisis de sus desviaciones y sus causas.

Los cuadros de mando están estrechamente relacionados con otros instrumentos de planificación y control de gestión como pueden ser la planificación estratégica (de donde provienen los objetivos) y la contabilidad que aporta información financiera y analítica.

Respecto al diseño y construcción de los cuadros de mando, éste es similar al diseño y construcción de los propios indicadores, de manera que podemos identificar los siguientes pasos:

- Identificar las necesidades de información en base a los factores clave derivados del entorno y de la estrategia.
- Establecimiento de los objetivos cuantificando los factores clave.
- Determinar las áreas y actividades clave que es necesario estudiar y medir.
- Identificación de las necesidades de información en base a los objetivos y las actividades clave.
- Identificación concreta de los emisores y usuarios de la información. Hay que destacar que la información necesaria para completar el cuadro de mando con frecuencia requiere de fuentes más amplias que la de los propios indicadores. Además, la participación de los usuarios es un factor esencial para el buen funcionamiento y utilización del cuadro de mando como herramienta de información y gestión.
- Identificación y elección de la tecnología en función de las necesidades informativas.

- Especificación de las características de la información en aspectos como:

    - La orientación de la información.
    - El diseño y establecimiento de los indicadores clave.
    - El soporte, en el cual, va a darse a la información (informático, papel…).
    - La forma en que la información se va a presentar (numérica, gráfica…).
    - La frecuencia (cuadro de mando diario, semanal…).
    - El horizonte de la información (corto, medio y largo plazo).

Podemos hablar de diversas formas de presentar los sistemas de indicadores y la información que se deriva de los mismos, siempre teniendo presente que el diseño de un cuadro de mando de indicadores debe adaptarse a cada empresa. En cualquier caso, las diferentes presentaciones de diseño suelen ser las siguientes:

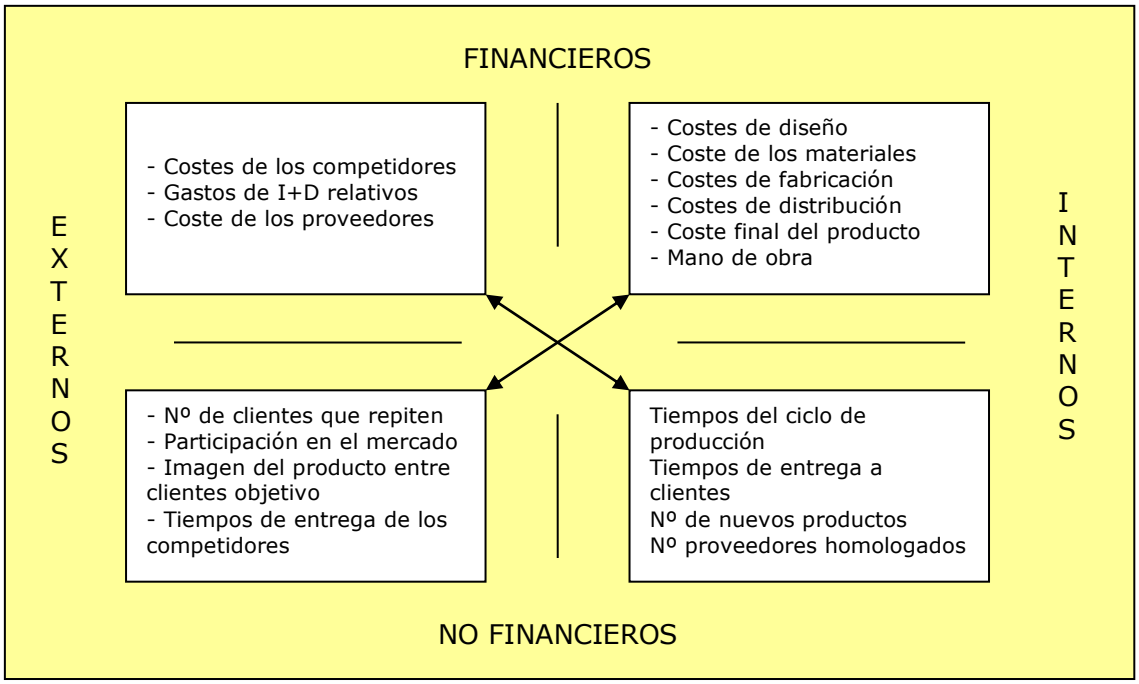
- **Por áreas o factores clave:** diseñar el cuadro de indicadores, de acuerdo a los factores clave de la empresa, proporciona un mayor enfoque estratégico y menos departamental. Definir los indicadores dependerá de la identificación y definición de los factores clave. Muchas veces lo óptimo es utilizar un cuadro de indicadores que adopte una forma mixta entre el cuadro por áreas funcionales y el cuadro por factores clave.

- **Financieros, no financieros, internos y externos:** este criterio de clasificación implica diversas posibilidades de combinar los indicadores, de manera que tenemos:

    - Indicadores financieros internos.
    - Indicadores financieros externos.
    - Indicadores no financieros internos.
    - Indicadores no financieros externos.

    Esto nos permite comparar lo que ocurre en el exterior de nuestra empresa con lo que ocurre en el interior, tanto desde una perspectiva financiera como desde una perspectiva no financiera. Este tipo de cuadro de indicadores cuenta con dos ventajas:

    - Nos permite ver lo que pasa en la empresa desde varias perspectivas.
    - Reduce la cantidad de información, pero no la visión de conjunto.




Cuadro de mando con indicadores financieros, no financieros, internos y externos
Fuente: Elaboración propia

- **En forma piramidal:** la estructura piramidal es otra forma muy habitual de presentar la información obtenida a través de los indicadores. Así, es muy habitual presentar la información financiera de una empresa de manera ordenada y lógica a través de una pirámide de ratios financieros.

  Este tipo de estructuras tiene sentido cuando la información que se deriva de un indicador da lugar a otros indicadores produciendo información en cascada. Para realizar una pirámide de indicadores, es necesario realizar antes una pirámide de factores clave derivados.

- **Por áreas funcionales:** esta es la forma más habitual de presentar la información en la mayoría de las empresas. En este caso, se divide la información del cuadro de mando de la misma forma que se divide en las empresas, que es de manera funcional o departamental.

  Este tipo de cuadros de mando suele ser el reflejo del organigrama funcional de la empresa y representa, de forma clara, la organización. Sin embargo, presenta algunas limitaciones, como que al dividir en exceso la información se pierde la visión de proceso. En un cuadro de mando de este estilo podríamos identificar los siguientes apartados:

- Indicadores del área financiera. Tratan de informar sobre la salud económica y financiera de la empresa y pueden estructurarse en tres apartados:

  - Ratios e indicadores financieros.
  - Ratios e indicadores económicos.
  - Ratios e indicadores bursátiles en el caso de empresas cotizadas.

- Indicadores del área comercial y marketing. Los indicadores en esta área pueden enfocarse de diferentes formas, pero aportarán información, entre otros aspectos, sobre ventas, costes comerciales, costes de almacenaje, costes de logística, costes de publicidad y promoción…, por lo que es necesario acompañar todo esto con información sobre el mercado y los competidores, para tener una idea acerca del posicionamiento competitivo de la empresa en el mercado.

- Indicadores del área de RR. HH. Informan sobre el desarrollo de aspectos relacionados con el cumplimiento de las políticas y objetivos referentes a las políticas de personal, productividad de la plantilla, estructura de la misma, rotaciones…

- Indicadores del área de producción y operaciones. La mayoría de los factores clave de competitividad los encontramos en las operaciones de la empresa. Tradicionalmente, se han utilizado indicadores de resultados basados en la eficacia y la eficiencia productiva, sin embargo, ahora hay una tendencia encaminada al control de los procesos y no de las operaciones.

- Indicadores del área de compras. El diseño de los indicadores, en este caso, estará condicionado por el nivel de integración de la empresa con sus proveedores. Estos indicadores están relacionados con tres aspectos: la calidad del producto o servicio recibido, los plazos de entrega de los proveedores y los costes de aprovisionamiento.

- **Cuadro de mando integral (Balanced Scorecard):** El objetivo principal es encontrar un conjunto de indicadores que integren todas las áreas de la empresa, controlen los factores críticos derivados de la estrategia y, todo ello, de una manera equilibrada, atendiendo a las diferentes perspectivas que pueden darse en la misma. Además, este modelo pretende poner de manifiesto las relaciones causa-efecto que se dan entre los distintos indicadores.

Trata de integrar los aspectos de la gerencia estratégica y la evaluación del desempeño del negocio. Este sistema fue propuesto en 1992 por Robert Kaplan y David Norton. Aporta una visión rápida (a través del uso de pocos indicadores de gestión relevantes) y certera de la situación y evolución de la compañía y está orientado a la imagen externa (accionistas y clientes) y a la mejora interna (procesos y personas).

Establece un balance entre las diferentes dimensiones del desempeño, lo que garantiza que los resultados positivos en un área no queden ocultos tras los resultados negativos de otra. Es un sistema orientado al futuro y al establecimiento de estrategias de mejora continua. Además, permite la posibilidad de extenderlo hacia los niveles más bajos de la organización.

Este sistema se apoya en cuatro perspectivas de negocio:

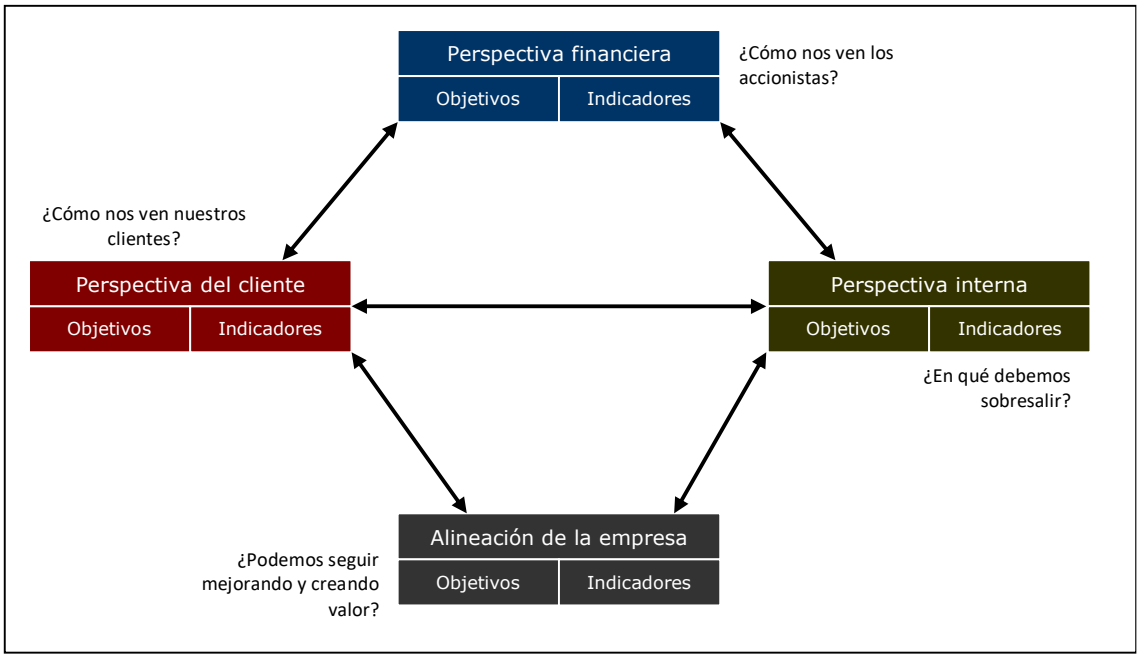
- La perspectiva financiera. Los indicadores están orientados a evaluar información de un carácter claramente financiero que interesa a los accionistas.

- La visión de los clientes. Se basa en conocer y comprender lo que de verdad valoran los clientes de la empresa. Esto lo podemos conseguir preguntándoles de manera directa o contratando a una empresa externa para que lo haga. Las variables a las que suelen dar más importancia los clientes se agrupan en torno a cinco dimensiones: plazo, calidad, eficacia, servicio y coste.

- La excelencia en procesos internos y operaciones: perspectiva interna. Se evalúan los procesos internos clave que generan valor para accionistas y clientes. Estos procesos se pueden agrupar de la siguiente manera:

    - Procesos relacionados con la innovación de productos que satisfagan las necesidades de los clientes.
    - Procesos relacionados con las operaciones.
    - Procesos de fabricación y procesos relacionados con el servicio post-venta.

- La alineación de la empresa. Esta perspectiva trata de mantener la posición competitiva actual y futura de la empresa. En este caso, se intenta medir y controlar la capacidad de la organización para innovar, mejorar y aprender, que son las bases para asegurar una posición competitiva sostenible.




Las cuatro perspectivas del cuadro de mando integral
Fuente: Elaboración propia

Esta forma de presentar el cuadro de indicadores se basa en los distintos grupos de intereses que podemos encontrar en la empresa. Por ejemplo, la información que necesitan los accionistas es diferente de la que interesa a los clientes, y ambas han de ser complementadas con la que necesita la dirección de la empresa para garantizar la mejora continua en todas las actividades.

El aspecto más importante de la utilización del cuadro de mando integral radica en encontrar un equilibrio en los indicadores de manera que se pueda controlar un factor clave desde cualquiera de las perspectivas.

### *5.2.4 Implementación práctica de un dashboard*

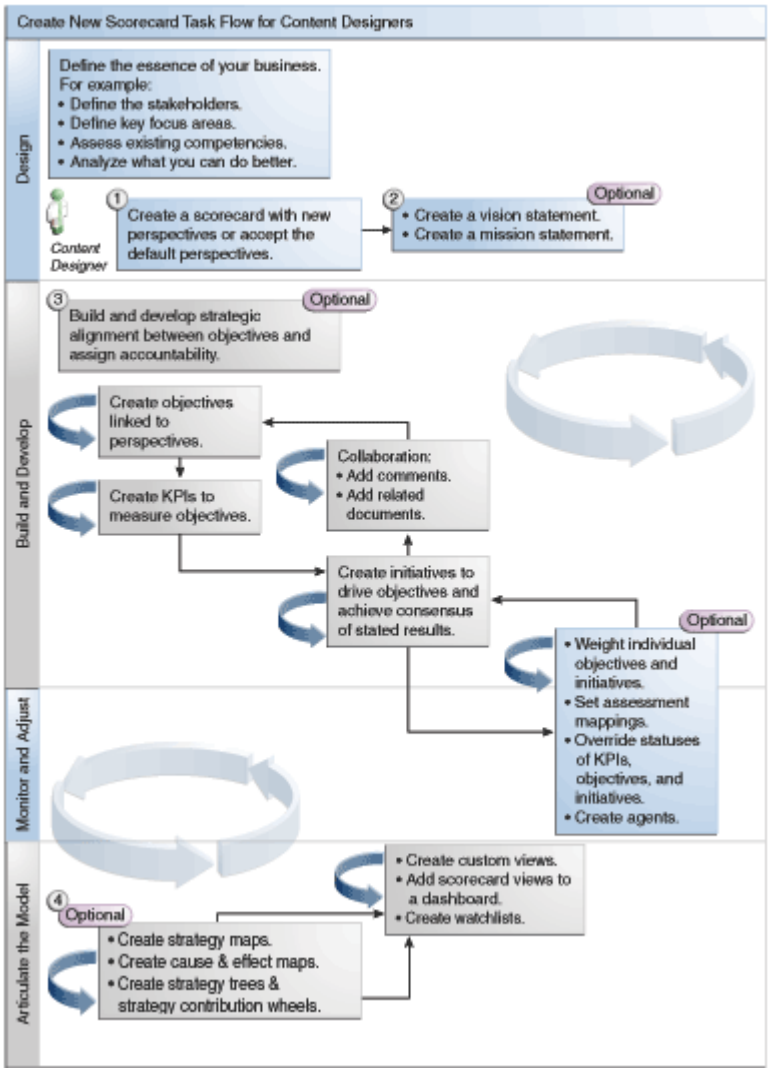
De entrada, es importante tener claro que el proceso para la creación de un dashboard tablero de control está basado en iteraciones que combinan diversos hasta la implementación final.

Es importante también considerar en todo proyecto de implementación de un dashboard se debe tener en cuenta la triple restricción: costes, alcance y tiempo.

Para implementar un dashboard, el equipo de trabajo debe cumplir al menos estas condiciones:

- Conocimiento profundo del negocio o proceso de negocio elegido.
- Equipo interdisciplinar con los diversos actores de la organización (IT, comercial, marketing…).
- Objetivo estratégico elegido y alineado con la visión de empresa, claro y entendido por todos.
- Definir un responsable con la capacidad suficiente para toma de decisiones.

En la imagen se muestra un diagrama que representa el proceso para la creación de un dashboard o tablero de control. Este método está tomado de del Oracle Help Center pero sería aplicable a cualquier otro software equivalente.




Proceso de creación de dashboards
Fuente: https://docs.oracle.com/middleware/12211help/biee/es/bi.12211/e73374/img/GUID-F6F63C36-8DD0-481F-AD14-18AF2C187182-default.gif

Aunque este esquema es propuesto por Oracle para la creación de tableros con su herramienta de BI, este proceso está bien ilustrado para implementarlo en la creación de cualquier cuadro de mando y para cualquier herramienta elegida.

En principio, se recomienda que el proceso se realice en el orden de flujo de tareas indicado. Este método consiste en cuatro etapas:

- **Diseño.** El diseño es el paso inicial que marcará hacia dónde se dirige la estrategia del dashboard. En el diseño es determinante identificar y tener claramente definido:

    - Necesidades que cubrirá.
    - Identificar a los stakeholders o interesados.
    - Analizar qué es lo que se puede hacer.

- **Construcción y desarrollo.** Una vez identificados y documentadas las perspectivas y objetivos de medición, es hora de alinear y organizar las metas y resultados deseados en función del proceso de negocio.

  Los indicadores claves de desempeño o KPI medirán el desempeño del proceso. Basado en las necesidades identificadas en el diseño, es hora de elegir y construir los gráficos idóneos. Estos objetos gráficos deben estar ligados a las perspectivas de negocio del proceso, por ejemplo, cantidad acumulada de ventas, comparación de series de tiempo de un flujo financiero, etc.

  Los indicadores elegidos deben ser medibles, realizables y no ambiguos. Se recomienda evaluar la creación de los indicadores clave basados en el método SMART:

  - Specific/Especifico: debe estar enfocado a un objetivo específico del negocio.
  - Mesurable/Medible: cuantificable o al menos ser un indicador de progreso.
  - Achievable/Realizable: debe ser realizable y asignable a un responsable.
  - Relevant/Relevante: sus resultados deben tener mucho valor.
  - Time-Bound/Acotado en Tiempo: el objetivo debe ser trazado en el tiempo.

  Una vez establecidos los objetivos, se deben identificar y crear los métodos de captura de datos y la política de gobierno de datos que marque de donde se obtendrá la información susceptible de análisis. Igualmente, deben ser creadas y definidas las listas de comprobación del indicador.

- **Monitorización y ajuste.** La monitorización y ajuste será el proceso que depende de la evaluación de los KPI definidos para el tablero. Esto llevará a realizar una valoración de los resultados obtenidos y su correspondencia con los objetivos del negocio. Esto definirá la siguiente iteración, donde se eliminarán, sobrescribirán o crearán nuevos objetivos de medición y métricas personalizadas con base en las relaciones especificadas entre los KPI.

- **Articular el modelo y despliegue.** Una vez que hemos ejecutado los pasos anteriores, es el momento de crear las vistas personalizadas que permitan acotar más los resultados acordes a la estrategia de la organización. Con el fin de que el tablero represente la esencia del negocio, puede apoyarse en el uso de árboles de decisión o diagramas de causa y efecto para establecer las relaciones pertinentes.

  Establecido y aprobado el tablero, es hora de distribuirlo a los clientes o usuarios finales, quienes serán la voz que se deba escuchar para realizar un diseño a la medida de los reportes. Una vez puesto en marcha el dashboard no acaba el trabajo, puesto que se inicia otro ciclo que permitirá personalizar el tablero.

  En resumen, para la creación efectiva de tableros de control coherentes con la estrategia de BI se debe aplicar un proceso estructurado que gira en torno a cuatro ejes fundamentales:

  - Determinar los datos a utilizar y mostrar.
  - Determinar el formato de presentación (interfaz gráfica) de acuerdo con los datos identificados (determinar los elementos gráficos que mejor representan la visión de los datos).
  - Realizar combinaciones de los datos, objetos visuales e informes para presentarlos conjuntamente en un tablero de control.
  - Planificar la interactividad y navegación del usuario.

### *5.2.5 Elementos visuales*

Un dashboard se compone de diversos elementos, los cuales, se van combinando a lo largo del diseño de este, empleando gráficos y métodos de estadística descriptiva. Entre los elementos que un software para dashboards permite utilizar, se encuentran los siguientes:

- **Tablas:** son matrices de x columnas por n filas. Es un elemento útil cuando se busca realizar resúmenes de grandes cantidades de datos. Una tabla puede ser dinámica o estática, lo que principalmente depende del tipo de visualización buscada y como se quiere presentar la información al cliente o usuario final.

- **Métricas:** las métricas son parte esencial del proceso para la definición de la estrategia de BI. Estas deben recoger los valores resultantes de un proceso de negocio determinado y suelen ser definidas dentro de un KPI.

- **Listas:** las listas típicamente provienen de los KPI definidos y no son más que la representación de los valores, nombres y métricas definidas para los KPI.

- **Gráficos:** los gráficos y objetos visuales varían según la necesidad y lo que se quiere mostrar. No debemos incluir gráficos que no estén justificados, porque distraerían la atención del usuario final y podrían afectar de forma negativa a la toma de decisiones en el proceso de negocio.

- **Mapas:** tienen mucho impacto visual en un dashboard. Suelen emplearse para mostrar una determinada información con base a su geolocalización. A veces, no resulta sencillo integrar este tipo de información en un dashboard y es necesario combinar los mapas con otros elementos para que se comprendan más fácilmente. Dada la complejidad que puede suponer su inclusión, debemos evaluar de forma precisa si son necesarios.

- **Alertas visuales:** las alertas visuales funcionan con la definición de un evento, en el cual, se debe poner en evidencia un cambio en los datos. Comúnmente, se muestran como códigos de colores (semáforos) o indicadores en formato de velocímetro o tacómetro.

- **Menú de navegación o filtros:** por último, y no menos importante, tenemos este elemento, determinante cuando se busca una comprensión efectiva y fácil navegación por el dashboard. Estos elementos dan al usuario la facilidad de filtrar la información presentada a su gusto, siempre en base al proceso de negocio elegido.

En conclusión, los distintos elementos deben combinarse y emplearse de acuerdo con la necesidad de información. Así mismo, dependerán también del proceso de negocio, puesto que no todos los gráficos o elementos visuales son apropiados en todos los casos. Es clave revisar detenidamente el proceso de negocio, identificar las vistas y usuarios finales, las dimensiones requeridas y las métricas idóneas, para que la información a condensar en el dashboard sea oportuna y esté ligada estrechamente a los objetivos de la organización.

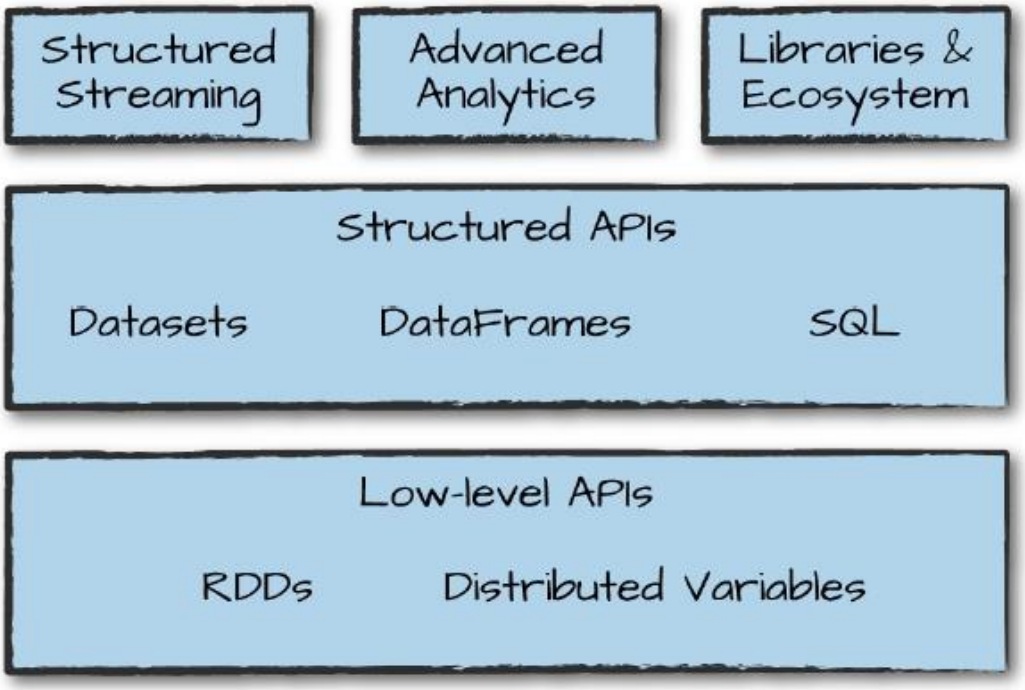
El uso de software no significa que los cuadros de mando sean algo nuevo. De hecho, son prácticas comunes en distintas organizaciones puesto que son formas de realizar la medición del desempeño de procesos estratégicos.

Lo que el software aporta nuevo es el dinamismo de los dashboards. Hasta hace un tiempo, estos cuadros eran fotos fijas en un momento dado. Ahora, se abre la posibilidad de disponer de la información en tiempo real y desde cualquier lugar.

## 5.3 Spark en profundidad

Apache Spark es un motor de computación unificado y un conjunto de bibliotecas para el procesamiento paralelo de datos en clústeres de servidores. Actualmente, Spark es el motor de código abierto más utilizado para esta tarea, convirtiéndolo en una herramienta estándar para cualquier desarrollador o científico de datos interesado en big data.

Admite múltiples lenguajes de programación, entre los que se encuentran algunos de los más difundidos, como Python, Java, Scala y R. También incluye bibliotecas para diversas tareas que van desde SQL hasta streaming y aprendizaje automático. Se ejecuta en cualquier lugar, desde un pequeño equipo portátil hasta un clúster de miles de servidores. Esto hace que sea un sistema con el que es fácil comenzar un proyecto y ampliarlo hasta el procesamiento de big data a una escala increíblemente grande.




Componentes y bibliotecas de Spark
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

### 5.3.1 La historia de Spark

Spark empezó a fraguarse en 2009, en la Universidad de Berkley, como un proyecto de investigación en el AMPLab. En 2010 se publicó un artículo llamado Spark: Computación en clúster con conjuntos de trabajo de Matei Zaharia, Mosharaf Chowdhury, Michael Franklin, Scott Shenker y Ion Stoica. En ese momento, Hadoop y MapReduce eran el estándar para procesamiento paralelo en clústeres de cientos de nodos. Sin embargo, Hadoop era ineficiente para problemas iterativos como por ejemplo machine learning o grafos. Por este motivo, se tomaron la molestia de desarrollar otras soluciones innovadoras en big data.

Spark basó su API en la programación funcional e implementó el motor de procesamiento para que prestase soporte al almacenamiento de datos en memoria para poder hacer iteraciones sobre los datos más eficientemente. Como inicialmente solo se soportaban aplicaciones batch, se comprobó que era posible resolver consultas ad hoc y realizar tareas interactivas de machine learning, principalmente, en Scala.

En 2011 se creó el **Apache Shark**, que fue el primer motor de SQL para Spark. En ese momento se empiezan a añadir diferentes librerías construidas sobre Spark y fue cuando llegaron las primeras versiones de MLib, SparkStreaming y GraphX.

En 2013 obtuvo una amplia difusión ya que más de treinta organizaciones contribuyeron a la evolución de Spark. AMPLab cede Spark a la Apache Software Foundation, considerando que en su seno podría tener una mejor evolución. También se lanza la empresa DataBricks. Un año después, en 2014, llega al mercado el Spark 1.0, con SparkSQL, orientado a la captación de datos estructurados. En 2016 llegó el Spark 2.0, con la opción de DataFrames, StructuredStreaming, Pipelines en machine learning, etc. Lo que lo aproximó a un enfoque más parecido al Spark que hay hoy en día.

Gracias a sus primeros éxitos y ventajas frente a los métodos anteriores, se va asentando y creando una comunidad de usuarios. Recordemos que durante años MapReduce ha sido el paradigma de software más robusto para procesamiento por lotes distribuidos en big data. Sin embargo, el procesamiento por lotes, aunque es muy útil, no es la mejor respuesta a determinadas necesidades de cómputo. Impulsado por la adopción progresiva y generalizada del big data, junto con las ganas de los usuarios de trabajar con más casos de usos (impulsados por la primera generación de soluciones novedosas en big data) los investigadores comenzaron a buscar formas de mejorar las capacidades de MapReduce. Un resultado interesante de estas investigaciones fue **Apache Drill**.

Drill es un lenguaje de consulta estructurado de alto rendimiento que no requiere que las empresas realicen peticiones frecuentes al departamento de TI.

Spark fue diseñado para ser un motor general con propósitos interactivos, procesamiento por lotes y tareas de streaming, capaz de aprovechar los mismos tipos de recursos de procesamiento distribuido, que hasta aquel momento eran iniciativas exclusivas de MapReduce.

Cuando se ve el contexto histórico del open source en big data, la búsqueda continua de mejores soluciones no es sorprendente. Por ejemplo, el Apache Pig (el lenguaje MapReduce especializado de alto nivel) fue el resultado de investigaciones de Yahoo para acceder a los datos almacenados en Hadoop.

Apache Hive (que trajo la funcionalidad de SQL a MapReduce) fue el resultado de una investigación de Facebook para conseguir un mecanismo de interacción con los datos optimizado y basado en SQL.

La tendencia continua de otras iniciativas respaldadas por Apache, como Impala (orientado a consultas comerciales de tipo analítico para Hadoop) y Presto (un motor de procesamiento distribuido de alta velocidad para ejecutar consultas SQL contra bases de datos masivas) motivaron todavía más, si cabe, a los desarrolladores de Spark, que lo crearon con la expectativa de que funcionaría con petabytes de datos, que se distribuirían a través de clústeres de miles de servidores, tanto físicos como virtuales.

Desde el principio estaba destinado a explotar el potencial –particularmente en velocidad y escalabilidad- ofrecido por el procesamiento de datos en memoria. Estos ahorros de tiempo realmente suman si comparamos con los trabajos realizados en MapReduce. La norma general es que Spark ofrece resultados con los mismos datos hasta cien veces más rápido que MapReduce.

En comparación con los enfoques anteriores que se basan en MapReduce, Spark ofrece cuatro ventajas principales para desarrollar soluciones de big data:

- Desempeño.
- Simplicidad.
- Facilidad de administración.
- Desarrollo rápido de aplicaciones.

Gracias a estos argumentos, Spark ha conseguido una rápida adopción y ha sido promovido a un proyecto de alto nivel por la Apache Software Foundation en 2014. Dado que más de 300 aplicaciones de la Fundación estaban en marcha, esta promoción tuvo realmente mucha importancia para poner a Spark en el lugar que merecía.

### *5.3.2 Filosofía Apache Spark*

Como ya se ha mencionado, Apache Spark es un motor informático unificado y un conjunto de bibliotecas para big data, que puede ser desgranado en varios componentes clave:

- **Unificado.** El objetivo clave de Spark es ofrecer una plataforma unificada para escribir aplicaciones de big data. ¿Qué queremos decir con unificado? Spark está diseñado para admitir una amplia gama de tareas de análisis de datos, que van desde la simple carga de datos y consultas SQL, hasta aprendizaje automático y streaming, todo ello sobre el mismo motor informático y en un conjunto consistente con una API.

  La idea principal que subyace a este objetivo es poder analizar los datos del mundo real y que estos análisis sean interactivos y puedan ser realizados mediante herramientas sencillas como Jupyter Notebook u otros softwares de desarrollo (la idea es combinar diferentes tipos de procesos y librerías)

  La naturaleza unificada de Spark hace que estas tareas sean más fáciles y eficientes de escribir. Primero, Spark proporciona una API consistente e integrable, que puede utilizarse para crear aplicaciones a partir de pequeñas piezas ya existentes o al margen de las bibliotecas estándar.

  Esto nos facilita escribir nuestros propios análisis y bibliotecas. Además, la API es totalmente completa y está diseñada para permitir un alto rendimiento al optimizar diferentes bibliotecas y funciones en un único programa. Por ejemplo, si cargamos datos utilizando una consulta SQL y luego evaluamos un modelo de aprendizaje automático utilizando la biblioteca ML de Spark, el motor puede combinar estos pasos. La combinación de APIs generales con un alto rendimiento, hace de Spark una plataforma potente para aplicaciones interactivas y de producción.

El enfoque de Spark en definir una plataforma unificada es la misma idea detrás de las plataformas unificadas en otras áreas de software. Por ejemplo, los científicos de datos se benefician de un conjunto unificado de bibliotecas (por ejemplo, en Python o R) al modelar, y los desarrolladores web se benefician de frameworks unificados como Node.js o Django.

Antes de Spark, ningún sistema de código abierto intentaba proporcionar este tipo de motor unificado para el procesamiento de datos en paralelo, lo que significaba que los desarrolladores tenían que unir en una aplicación múltiples APIs y sistemas. Por lo tanto, Spark se convirtió rápidamente el estándar para este tipo de desarrollo. Con el tiempo, Spark ha continuado expandiéndose, basándose en APIs para cubrir más cargas de trabajo.

- **Motor informático.** Al mismo tiempo que Spark se esfuerza por la unificación, limita cuidadosamente su alcance a un motor informático. Con esto, queremos decir que Spark gestiona los datos de los sistemas de almacenamiento y realiza cálculos sobre ellos, pero no realiza almacenamiento permanente.

  Se puede usar Spark con una amplia variedad de sistemas de almacenamiento persistente, incluidos sistemas de almacenamiento en la nube, como Azure Storage, Amazon S4, sistemas de archivos distribuidos como Apache Hadoop, Apache Cassandra e incluso Apache Kafka. Sin embargo, Spark no almacena a largo plazo.

  Los datos son difíciles de mover de un sitio a otro, por eso Spark se enfoca en realizar cálculos sobre ellos, sin importar donde estos residan. Para cada usuario de la API, Spark trabaja para hacer que estos sistemas de almacenamiento se vean homogéneamente, para que las aplicaciones no necesiten preocuparse por dónde están los datos.

  El enfoque de Spark lo hace diferente de las plataformas de software de big data anteriores, como Apache Hadoop. Hadoop incluía un sistema de almacenamiento (el sistema de archivos Hadoop, diseñado para almacenamiento de bajo coste en clústeres de servidores básicos) y un sistema informático (MapReduce), que se integraba estrechamente.

Sin embargo, esta elección hace difícil ejecutar uno de los sistemas sin el otro. Además, esta elección trae el desafío de escribir aplicaciones que accedan a los datos almacenados en cualquier otro lugar. Aunque Spark funcione bien en el almacenamiento de Hadoop, actualmente también se usa mucho en entornos en los cuales la arquitectura de Hadoop no tiene sentido, como en la nube pública (donde el almacenamiento puede ser comprado por separado de la capacidad de cómputo) o aplicaciones de streaming.

- **Librerías.** El componente final de Spark son sus bibliotecas, que se basan en su diseño integrado con el motor unificado para proporcionar una API para tareas comunes de análisis de datos.

  Spark admite tanto bibliotecas estándar que se incluyen con el motor, así como con una amplia gama de bibliotecas externas publicadas por terceros que forma parte de las comunidades de código abierto. Hoy, las bibliotecas estándar de Spark son en realidad una gran parte del proyecto de código abierto: el motor central de Spark en sí mismo ha cambiado poco desde que fue lanzado, pero las bibliotecas han crecido para proporcionar más y más tipos de funcionalidad.

  Spark incluye bibliotecas para SQL y datos estructurados (Spark SQL), aprendizaje automático (MLlib), procesamiento de flujo (Spark Streaming y el más reciente Structured Streaming) y análisis gráfico (GraphX).

  Más allá de estas bibliotecas, hay cientos de bibliotecas open source que van desde conectores para varios sistemas de almacenamiento hasta algoritmos de aprendizaje automático.

**ENLACE DE INTERÉS**

Si quieres acceder al índice de bibliotecas externas lo podrá encontrar disponible en la siguiente página.

https://spark-packages.org/

### 5.3.3 Usos comunes de Spark

Las soluciones basadas en Spark son innumerables y algunas de ellas son especialmente valiosas en la situación actual del mercado. Su velocidad, simplicidad y productividad ha dejado atrás un enorme número de tipos de proyectos más largos y costosos. Dicho esto, Spark es particularmente eficaz en varias aplicaciones.

Para presentar ejemplos más tangibles, veremos a continuación situaciones actuales en las operaciones diarias de una empresa de servicios financieros. Estos casos son aplicables a cualquier tipo de industria:

- **Transmisión de datos (streaming).** Aunque las fuentes de datos y su uso varían según la industria, muchas empresas tratan constantemente con una variedad de datos transmitidos (en streaming) y generados por:

    - Sensores.
    - Transacciones financieras.
    - Dispositivos móviles.
    - Monitores médicos.
    - Registros (logs) de accesos a sitios web.
    - Actualizaciones de redes sociales como Facebook, Twitter y LinkedIn.

  Trabajar directamente con la transmisión de datos es diferente hoy en día en comparación con el pasado, cuando las transacciones realizadas por los softwares empresariales se hacían a un ritmo más lento y en volúmenes pequeños.

  Usando el ejemplo de servicios financieros, una aplicación Spark podría analizar las autorizaciones de tarjeta de crédito entrantes en tiempo real, en lugar de analizar esta información mucho después de realizada. Este tiempo ahorrado ha sido un gran avance para las soluciones de seguridad, prevención de fraudes (en vez de simplemente detección) y otras aplicaciones análogas.

  Por ejemplo, un banco podría tomar una decisión basada en evidencias y rechazar una transacción con una tarjeta de crédito falsa en el punto de venta, en tiempo real, en lugar de permitir que el estafador compre el producto y, por tanto, se genere una pérdida para en banco y para el minorista.

- **Inteligencia artificial (IA), aprendizaje automático (machine learning) y aprendizaje profundo (deep learning).** Las organizaciones están cada día utilizando más los algoritmos de inteligencia artificial, machine learning y deep learning para desarrollar una capacidad muy potente de analítica para aplicaciones que van desde la detección de objetos hasta la mejora de interacción entre humano y máquina a través del procesamiento de lenguaje natural, así como sofisticadas aplicaciones para detección de anomalías.

  Muchas aplicaciones de IA, ML y DL, como las de reconocimiento y clasificación, han servido como punto de partida para aplicaciones revolucionarias, con muchas perspectivas para el futuro, como el coche autónomo.

  Los proyectos de IA comenzaron como problemas de big data, porque la IA se ve directamente afectada por la cantidad y calidad de los datos utilizados. Los expertos estiman que las soluciones emergentes de IA necesitan entre ocho y diez veces el volumen de datos utilizado en el big data actual. Estos datos provienen de muchas fuentes distintas como sistemas ERP, CRM, bases de datos, data lakes, sensores, datos públicos, aplicaciones móviles, redes sociales y datos heredados de sistemas anteriores. Estos datos llegan de muchas formas, pudiendo ser de forma estructurada, pero también de forma no estructurada.

  El procesamiento en memoria y los pipelines de datos, transformación y manipulación para proyectos de IA tiene muchos pasos involucrados:

  - Identificar y recolectar fuentes de datos.
  - Extraer los datos.
  - Etiquetar los datos.
  - Utilizar herramientas para manipular los datos (eliminar elementos anormales, valores nulos y otros procesos de limpieza).

- **Inteligencia de negocios (business Intelligence – BI):** Para ser más efectivos en sus tareas y poder visualizar sus datos con más facilidad y precisión, los usuarios finales y las empresas detectaron la necesidad de hacer algo más allá de los informes prediseñados que los gerentes y directores están acostumbrados a utilizar.

Estos profesionales se orientaron hacia plataformas más flexibles e interactivas, donde pivotar los datos para ser analizados fácilmente no generase mayores problemas.

Spark proporciona el rendimiento necesario para permitir análisis interactivos, no solamente por algoritmos automatizados, sino también por las personas. En el sector de servicios financieros, la combinación entre análisis humano e informático de los factores de riesgo proporciona como resultado decisiones comerciales más rápidas y precisas. Por ejemplo, los bancos otorgan préstamos a aquellos clientes que de verdad tienen capacidad para devolver el dinero.

- **Integración de datos (data integration):** La mayoría de las empresas implementan una gran variedad de sistemas comerciales. Algunos son locales, mientras que otros están basados en la nube. Unos son desarrollados internamente, mientras que otros son comprados a proveedores.

  Integrar todos esos datos puede ser complejo y computacionalmente intensivo en consumo de recursos. Este proceso se conoce como extraer, transformar y cargar (ETL o extract, transform and load), y muchas empresas han gastado innumerables horas de personal y millones de euros del presupuesto de TI para ejecutar y ajustar este proceso continuamente.

  **Incorporar Spark a la tarea de ETL significa que se puede hacer este proceso rápidamente** y combinando fácilmente datos de múltiples silos para obtener una mejor visión de la situación. Spark hace una integración impresionante con numerosos proveedores y plataformas de almacenamiento de datos y sobresale procesando estos tipos de cargas de trabajo.

  Partiendo de la base del streaming en servicios financieros y los ejemplos de ML ya conocidos, pensemos que interesante sería analizar en tiempo real los datos datos obtenidos desde los puntos de venta. Tomar la información de los TPVs y combinarla con la información de transacciones históricas registradas en las bases de datos relacionales, para refinar aún más la detección de fraude en el proceso comercial.

### 5.3.4 Entendiendo como Spark procesa las informaciones

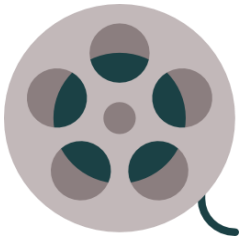
Los diseñadores de Spark eran muy conscientes del hecho básico de que en memoria el procesamiento siempre es más rápido que interactuando con discos (una ganancia promedio del rendimiento de diez veces).

Por este motivo, la funcionalidad ha sido muy bien recibida por los arquitectos de datos. Si a los beneficios de trabajar en memoria le añadimos otros de Spark, en conjunto, las soluciones de Spark pueden ser hasta 100 veces más rápidas que las aplicaciones anteriores de MapReduce basadas en disco.

Los conjuntos de datos distribuidos resilientes (RDD) son estructuras en memoria destinadas a contener la información que se desea cargar y después procesar dentro de Spark. Son llamados resilientes porque realmente son eso, dado que realizan un seguimiento del historial de los datos (por ejemplo, ediciones y eliminaciones) para que puedan reconstruirse en caso de que surja algún fallo en el camino. También son distribuidos, porque están dispersos en nodos de un clúster para ayudar a proteger los datos y al mismo tiempo aumentar su eficiencia a través del procesamiento paralelo.

Cuando se le presenta una solicitud a una aplicación, Spark usa las funciones de la plataforma para crear y cargar datos en un RDD. Esta información puede provenir de casi cualquier fuente de datos, incluido el sistema de archivos Hadoop, Apache Cassandra, Amazon S3, HBase, bases datos relacionales, etc.

Después de que los datos en cuestión se hayan colocado en su RDD, Spark permite dos tipos principales de operaciones: transformaciones y acciones.



**VIDEO DE INTERÉS**

En el siguiente enlace podrás ver un vídeo de introducción sobre Apache Spark, sus características y sus tipos de instalación.

https://www.youtube.com/watch?v=WR9HnAdYOfs

### 5.3.5 Arquitectura Spark

Generalmente, cuando pensamos en un ordenador, nos referimos a un equipo de sobremesa como el de casa o el del trabajo. Esta máquina funciona perfectamente bien para ver películas o trabajar con hojas de cálculo. Sin embargo, como muchos usuarios probablemente experimentan en algún momento, para algunas tareas el ordenador no es lo suficientemente potente como para realizarlas.

Un área particularmente desafiante es el procesamiento de datos. Las máquinas individuales no tienen suficiente potencia y recursos para realizar cálculos sobre grandes cantidades de información (o el usuario probablemente no tiene tiempo y no puede esperar a que termine el cálculo). Un clúster o grupo de ordenadores agrupa los recursos de muchas máquinas juntas, dándonos la capacidad de usar todos los recursos acumulativamente como si fueran un solo sistema.

Pero un grupo de máquinas por sí solo no es suficiente, si no que necesita un marco para coordinar el trabajo entre ellas. Spark hace exactamente eso, gestionando y coordinando la ejecución de tareas sobre datos a través de un grupo de ordenadores.

El grupo de máquinas que Spark usará para ejecutar tareas es administrado por un administrador de clúster, como el administrador de clúster independiente de Spark, YARN o Mesos. Posteriormente, enviamos las aplicaciones de Spark a estos administradores de clúster, que otorgarán recursos a nuestra aplicación para que podamos completar el trabajo.

### 5.3.6 Aplicaciones Spark

Las aplicaciones Spark consisten en un proceso controlador y un conjunto de procesos de ejecución. El proceso del controlador ejecuta su función main (), se instala en un nodo en el clúster y es responsable de tres cosas:

- Mantener información sobre la aplicación Spark.
- Responder al programa o entrada de un usuario.
- Analizar, distribuir y programar el trabajo entre los ejecutores.

El proceso del controlador es absolutamente esencial: es el corazón de una aplicación Spark y mantiene toda la información relevante durante la vida útil de la aplicación.

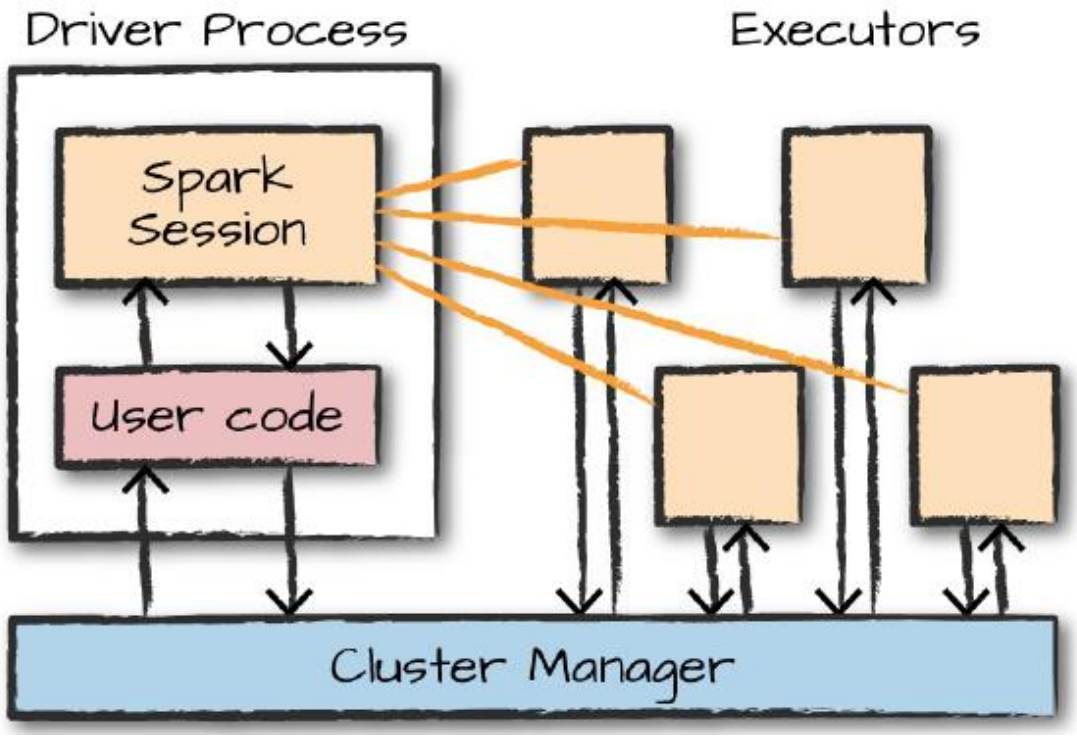
Los ejecutores son responsables de llevar a cabo el trabajo que el driver les asigna. Esto significa que cada ejecutor es responsable de solo dos cosas:

- Ejecutar el código asignado por el controlador.
- Informar del estado del cálculo en ese ejecutor nuevamente al nodo del controlador.

En la imagen se muestra cómo el administrador de clúster controla las máquinas físicas y asigna recursos para las aplicaciones Spark. Este puede ser uno de los tres administradores principales de clúster:

- Administrador Spark de clúster independiente.
- YARN.
- Mesos.

Esto significa que puede haber múltiples aplicaciones Spark que se ejecutan en un clúster al mismo tiempo.




Aplicaciones de Spark
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

También podemos ver el proceso controlador o driver a la izquierda y cuatro ejecutores a la derecha. En este diagrama, eliminamos el concepto de nodos del clúster. El usuario puede especificar cuántos ejecutores debe haber en cada nodo a través de las configuraciones.

Spark, además de su modo de clúster, también tiene un modo local. El driver y los ejecutores son simplemente procesos, lo que significa que pueden vivir en la misma máquina o en máquinas diferentes. En modo local, el controlador y los ejecutores se ejecutan (como hilos) en un ordenador individual en lugar de un clúster.

Los puntos clave que debemos comprender sobre las aplicaciones de Spark son:

- Spark emplea un administrador de clúster que realiza un seguimiento de los recursos disponibles.
- El proceso del controlador es responsable de ejecutar los comandos del programa del controlador en los ejecutores para completar una tarea determinada.

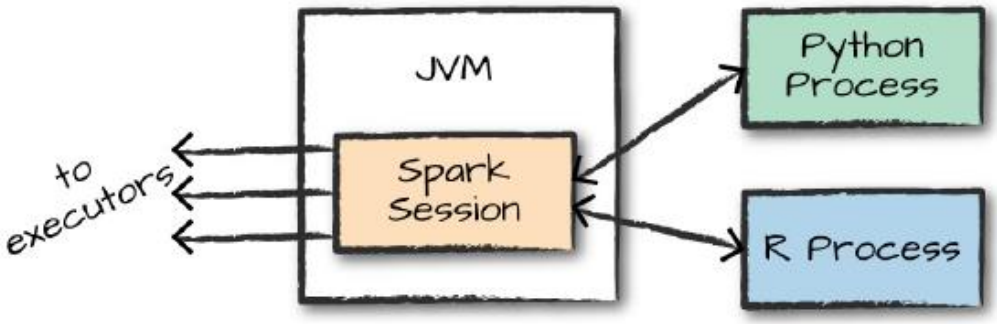
Los ejecutores, en su mayor parte, siempre ejecutarán el código Spark. Sin embargo, el driver puede ser "impulsado" desde varios lenguajes diferentes a través de las API de Spark.

### *5.3.7 APIs de lenguajes Spark*

Las APIs de lenguajes de Spark permiten ejecutar el código de Spark usando varios lenguajes de programación. Spark presenta algunos conceptos básicos en cada lenguaje. Estos conceptos después se traducen al código Spark, que se ejecuta en el clúster de máquinas. Con datos estructurados, es previsible que todos los lenguajes tengan características de rendimiento similares. A continuación, un breve resumen:

- **Scala:** Spark está escrito principalmente en Scala, por lo que es el lenguaje predeterminado de Spark.
- **Java:** aunque Spark está escrito en Scala, los autores de Spark han sido cuidadosos para asegurarse de que se puede escribir código Spark en Java.
- **Python:** admite casi todas las construcciones que admite Scala. Frecuentemente, incluiremos código Python para los ejemplos.
- **SQL:** Spark admite un subconjunto del estándar de SQL. Esto facilita a los analistas y no programadores aprovechar las capacidades de big data de Spark sin necesidad de adquirir nuevos conocimientos.
- **R:** Spark tiene dos bibliotecas R de uso común. Una que forma parte del núcleo de Spark (SparkR) y otra que un paquete desarrollado por la comunidad R (sparklyr).

Cada lenguaje mantiene los mismos conceptos centrales. Hay un objeto SparkSession disponible para el usuario, que es el punto de entrada para ejecutar el código Spark. Al usar Spark desde Python o R, no se escriben instrucciones explícitas de JVM. Por el contrario, escribimos el código en Python o R y Spark lo traduce a código que después se puede ejecutar en el ejecutor JVM como se observa en la siguiente figura.




APIs de lenguajes Spark
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

### *5.3.8 Arrancando Spark*

Hasta ahora, hemos visto los conceptos básicos de las aplicaciones Spark. Todo ello de forma conceptual. Pero cuando realmente escribamos una aplicación Spark, necesitaremos una forma de enviar comandos de usuario y datos hacia él. Eso lo conseguimos creando primero una SparkSession.

Para hacer esto, iniciaremos el modo local de Spark. Esto significa ejecutar ./bin/spark-shell para acceder a la consola de Scala e iniciar una sesión interactiva. También podemos iniciar la consola Python usando ./bin/pyspark. Esto inicia una sesión interactiva de Spark.

Cuando se inicia Spark en este modo interactivo, se crea implícitamente una SparkSession que administra la aplicación Spark. Cuando se inicia a través de una aplicación independiente, debemos crear el código SparkSession en el código de aplicación.

**VIDEOTUTORIAL**

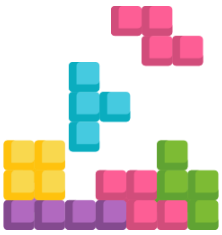
En este videotutorial podrás aprender cómo arrancar Spark.

### 5.3.9 SparkSession

Como ya sabemos, nosotros controlamos la aplicación Spark a través de un proceso del controlador llamado SparkSession. La instancia de SparkSession es la forma en que Spark ejecuta las manipulaciones definidas por el usuario en todo el clúster. Hay una correspondencia uno a uno entre un SparkSession y una aplicación Spark.

En Scala y Python, la variable está disponible como Spark cuando iniciamos la consola.

Hemos creado un DataFrame con una columna que contiene 1,000 filas con valores de 0 a 999. Este rango de números representa una colección distribuida. Cuando es ejecutado en un clúster, cada parte de este rango de números existe en un ejecutor diferente. Esto es un DataFrame de Spark.



**EJEMPLO PRÁCTICO**

Necesitamos controlar la aplicación Spark a través del proceso controlador SparkSession. ¿Cómo podríamos hacerlo con Scala y Python?

**SOLUCIÓN:**

En Scala, sería algo como lo siguiente:
res0: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession @ ...

En Python, veríamos algo como esto:
<pyspark.sql.session.SparkSession at 0x1efda1c1ccd1>

**EJEMPLO PRÁCTICO**

Necesitamos realizar una tarea sencilla de crear un rango de números. Este rango de números es como una columna con nombre en una hoja de cálculo. ¿Cómo lo haríamos en Scala y Python?

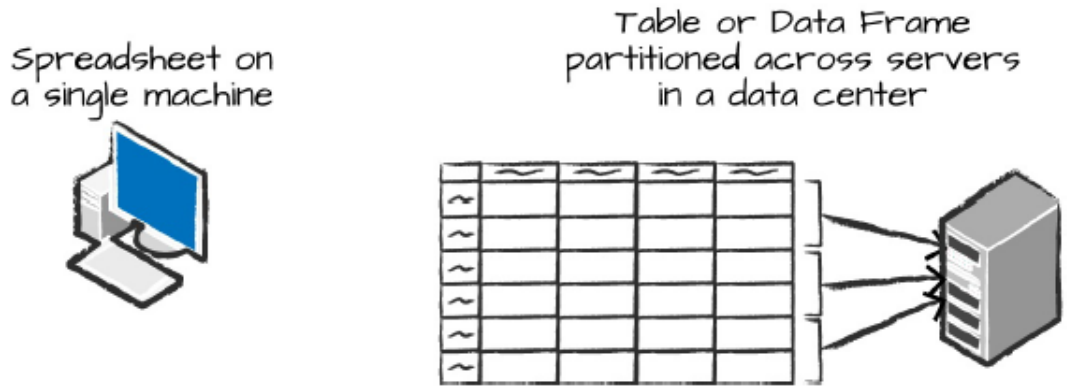
**SOLUCIÓN:**

```
// en Scala
val myRange = spark.range (1000) .toDF ("número")

# en Python
myRange = spark.range (1000) .toDF ("número")
```

### *5.3.10 DataFrames*

Los DataFrames son la estructura más común y simplemente representan una tabla de datos con filas y columnas. La lista que define las columnas y los tipos dentro de esas columnas se llama esquema. Podemos pensar en un DataFrame como una hoja de cálculo con columnas con nombre.

En la imagen se ilustra la diferencia fundamental: una hoja de cálculo se encuentra en una ubicación específica del ordenador, mientras que un DataFrame de Spark puede abarcar miles de equipos. La razón para poner los datos en más de un equipo debe ser ya intuitiva para nosotros y es porque algunos datos son demasiado grandes para entrar en una máquina o, simplemente, porque llevaría demasiado tiempo realizar ese cálculo en una única máquina.




Representación de DataFrames
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

El concepto DataFrame no es exclusivo de Spark. R y Python tienen conceptos similares. Sin embargo, los DataFrames, o marcos de datos en Python y en R (con algunas excepciones) existen solo en una máquina, en lugar de múltiples máquinas. Esto limita lo que se puede hacer con un DataFrame a los recursos disponibles en esa máquina específica. Sin embargo, dado que Spark tiene interfaces de lenguaje para Python y R, es bastante fácil convertir Pandas (Python) DataFrames a Spark DataFrames y R DataFrames a Spark DataFrames.

Spark tiene varias abstracciones principales: conjuntos de datos, marcos de datos, tablas SQL y RDDs. Todas ellas representan colecciones distribuidas de datos. Los más fáciles y los más eficientes son DataFrames, que están disponibles en todos los lenguajes de programación tradicionales.

### *5.3.11 Particiones*

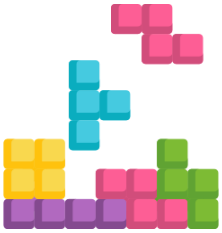
Para permitir que cada ejecutor realice un trabajo en paralelo, Spark divide los datos en fragmentos llamados particiones. Una partición es una colección de filas que se encuentran en una máquina física del clúster. Una de las particiones del DataFrame representa cómo se distribuyen físicamente los datos en el clúster de máquinas durante la ejecución. Si tenemos una partición, Spark tendrá un paralelismo de solo una, incluso si tenemos miles de ejecutores. En caso de tener muchas particiones, pero solo un ejecutor, Spark seguirá teniendo un paralelismo de solo uno, porque solo hay un recurso de cálculo.

### *5.3.12 Transformaciones*

Las transformaciones incluyen filtrar, mapear o manipular los datos, que activan automáticamente la creación de un nuevo RDD. El RDD original permanece sin cambios, pase lo que pase a los RDDs superiores. Esta inmutabilidad permite a Spark mantener el seguimiento de las alteraciones que se llevaron a cabo durante la transformación y, por lo tanto, hace que sea posible restaurar los datos de nuevo a la normalidad en caso de que ocurra algún fallo u otro problema a lo largo del proceso. De ahí, como ya hemos mencionado, que viene la denominación de resiliente.

Las transformaciones no se ejecutan hasta el momento que es realmente necesario, como una acción. Este comportamiento se conoce como evaluación perezosa y ayuda a mejorar el rendimiento al retrasar los cálculos hasta el instante en que se requieren, realizando solamente los cálculos que son necesarios.

En Spark, las estructuras de datos principales son inmutables, lo que significa que no se pueden cambiar una vez que están creadas. Esto puede parecer un concepto extraño al principio. Si no puedes cambiarlo, ¿cómo funciona? Para "cambiar" un DataFrame, debemos indicarle a Spark cómo nos gustaría modificarlo. Estas instrucciones se llaman transformaciones.



**EJEMPLO PRÁCTICO**

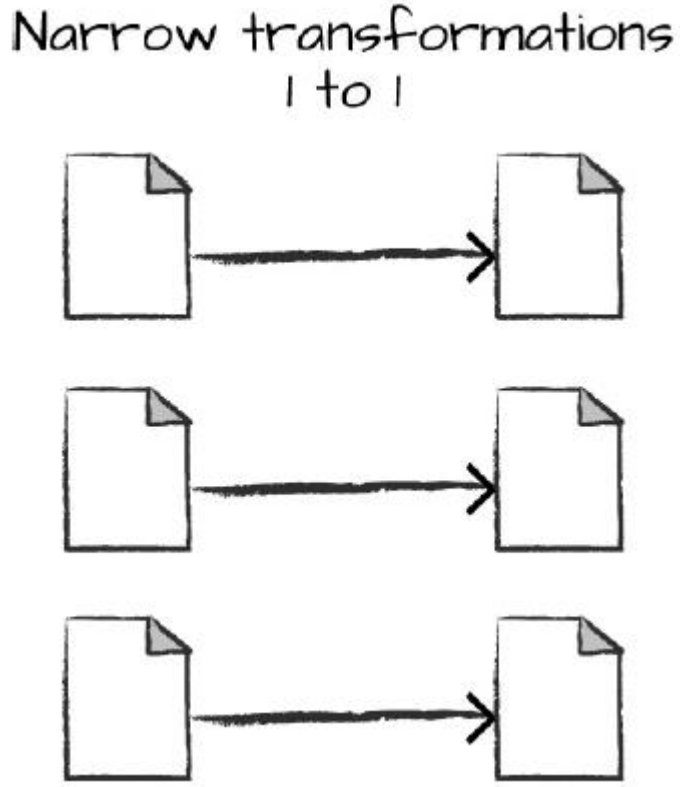
Necesitamos realizar una transformación simple para encontrar todos los números pares en nuestro DataFrame actual. ¿Cómo lo haríamos en Scala y Python?

**SOLUCIÓN:**

```
// en Scala
val divisBy2 = myRange.where ("número% 2 = 0")

# en Python
divisBy2 = myRange.where ("número% 2 = 0")
```

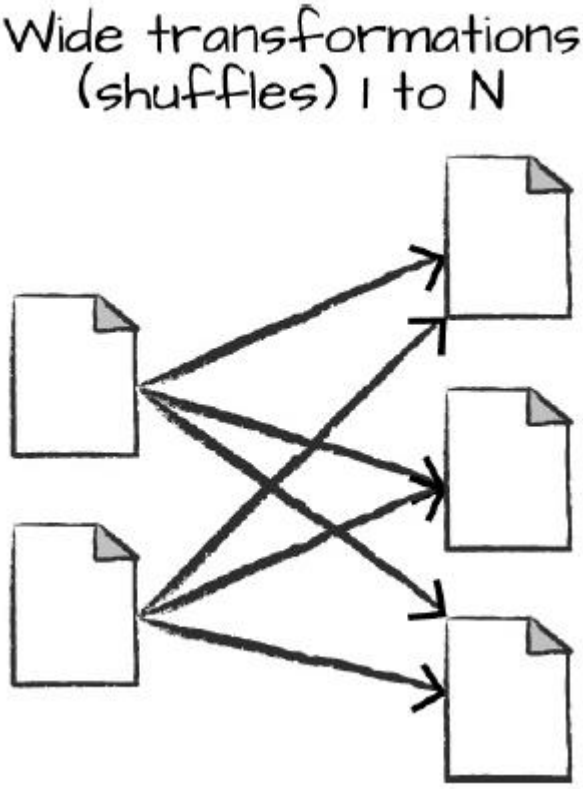
Tengamos en cuenta que estos no devuelven una salida. Esto se debe a que solo especificamos una transformación abstracta y Spark no actuará en las transformaciones hasta que llamemos a una acción. Las transformaciones son el núcleo de cómo se expresa la lógica de negocio usando Spark. Hay dos tipos de transformaciones:

- **Las que especifican dependencias estrechas.** Las transformaciones que consisten en dependencias estrechas (las llamaremos transformaciones estrechas) son aquellas para las cuales cada partición de entrada contribuirá a una sola partición de salida. En el anterior fragmento de código, la instrucción where especifica una dependencia estrecha, donde solo una partición contribuye como máximo a una partición de salida, como puede verse en la figura de abajo.




Transformaciones en Spark
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

- **Las que especifican dependencias amplias.** Una transformación de tipo dependencia amplia (o transformación amplia) tendrá particiones de entrada contribuyendo a muchas particiones de salida. A menudo esto es referido como un shuffle, por el cual, Spark intercambiará particiones en todo el clúster. Con transformaciones estrechas, Spark puede realizar automáticamente una operación llamada pipeline, lo que significa que, si especificamos varios filtros en DataFrames, todos se realizarán en memoria. No se puede decir lo mismo de los shuffles. Cuando realizamos un shuffle, Spark escribe los resultados en el disco.




Transformaciones amplias
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

Hay mucho debate sobre la optimización de esto, porque es un tema importante, pero por ahora, todo lo que necesitamos entender es que hay dos tipos de transformaciones. En este momento entendamos las transformaciones como simplemente formas de especificar diferentes series de datos manipulados y que esto nos ha llevado a la cuestión de la evaluación perezosa que analizaremos con un poco más de detalle.

### *5.3.13 Evaluación diferida o perezosa*

La evaluación diferida significa que Spark esperará hasta el último momento para ejecutar el cálculo. En Spark, en lugar de modificar los datos inmediatamente con cada operación, construimos un plan de transformaciones que nos gustaría aplicar a los datos de origen. Al esperar hasta el último minuto para ejecutar el código, Spark compila el plan de transformaciones del DataFrame en un plan que se ejecutará de la manera más eficiente posible a través del clúster.

Esto proporciona enormes beneficios, porque Spark puede optimizar el flujo de datos completo de extremo a extremo (end to end). Un ejemplo de esto es algo llamado predictive pushdown en DataFrames. Si creamos un trabajo Spark grande, pero especificamos un filtro al final que solo requiere buscar una fila de nuestros datos de origen, la forma más eficiente de ejecutar esto es acceder a la única fila y grabar qué necesitamos. Spark realmente optimizará esto por nosotros empujando el filtro hacia abajo automáticamente.

### *5.3.14 Acciones*

Las acciones son comportamientos que interactúan con los datos, pero no los cambian, como: contar, devolver un elemento específico y agregar.

Después de que una aplicación solicita una acción, Spark evalúa el RDD inicial y crea instancias de RDDs posteriores basadas en lo que se pide. Al generar este RDD adicional, Spark examina el clúster para encontrar el lugar más eficiente para colocar el nuevo RDD.

Las transformaciones nos permiten construir nuestro plan de transformación. Para desencadenar el cálculo, ejecutamos una acción. Una acción indica a Spark que calcule un resultado de una serie de transformaciones. La acción más simple es contar, que nos da el número total de registros en el DataFrame: divisBy2.count ().

La salida del código anterior debe ser 500. Por supuesto, contar no es la única acción. Hay tres tipos de acciones:

- Acciones para ver datos en la consola.
- Acciones para recopilar datos a objetos nativos en el lenguaje respectivo.
- Acciones para escribir en fuentes de datos.

Al especificar esta acción, comenzamos un trabajo de Spark que ejecuta nuestra transformación de filtro (una transformación estrecha), después una agregación (una transformación amplia) que realiza los recuentos por partición, y luego una recopilación, que lleva nuestro resultado a un objeto nativo en el respectivo lenguaje.

Podemos ver todo esto inspeccionando la interfaz de usuario de Spark, una herramienta incluida en Spark con la cual podemos supervisar los trabajos de Spark que se ejecutan en un clúster.

### 5.3.15 Un ejemplo de principio a fin

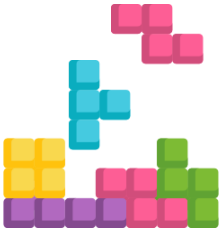
En el ejemplo anterior, creamos un DataFrame de un rango de números. Algo no especialmente complejo. En esta sección, reforzaremos todo lo que aprendimos anteriormente con un ejemplo más realista y que explique paso a paso lo que está sucediendo. Utilizaremos Spark para analizar algunos datos de vuelo de la oficina americana de estadísticas de transporte.

Disponemos de una carpeta CSV con varios archivos. También hay una serie de carpetas con diferentes formatos de archivo. Por ahora, centrémonos en los archivos CSV.

Cada archivo tiene varias filas. Estos archivos son CSV, lo que significa que son formatos semiestructurados y cada fila en el archivo representa una fila en el futuro DataFrame como se comprueba en el siguiente formato:

```
$ head /data/flight-data/csv/2015-summary.csv
DEST_COUNTRY_NAME,ORIGIN_COUNTRY_NAME,count
United States,Romania,15
United States,Croatia,1
United States,Ireland,344
```

Spark tiene la capacidad de leer y escribir desde una gran cantidad de fuentes de datos. Para leer estos datos, utilizaremos un DataFrameReader, que está asociado con nuestra SparkSession. Al hacerlo, especificaremos el formato del archivo y las opciones que queramos. En nuestro caso, queremos hacer algo llamado inferencia de esquema, lo que significa que queremos que Spark adivine qué esquema debe tener nuestro DataFrame. También queremos especificar que la primera fila es la cabecera del archivo, así que lo indicaremos también como una opción.



**EJEMPLO PRÁCTICO**

Para realizar nuestro trabajo necesitamos obtener la información del esquema, ¿cómo lo haríamos en Scala y Python?

**SOLUCIÓN:**

Para obtener la información del esquema, Spark lee parte de los datos y luego intenta determinar los tipos en las filas de acuerdo con los tipos disponibles en Spark.

También está la opción de especificar estrictamente un esquema cuando lee datos (que es recomendable en situaciones de producción):

```
// en Scala
val flightData2015 = spark
.read
.option("inferSchema", "true")
.option("header", "true")
.csv("/data/flight-data/csv/2015-summary.csv")

# en Python
flightData2015 = spark\
.read\
.option("inferSchema", "true")\
.option("header", "true")\
.csv("/data/flight-data/csv/2015-summary.csv")
```

Cada uno de estos DataFrames (en Scala y Python) tiene un conjunto de columnas con un número de filas no especificado. La razón por la que no se especifica el número de filas es porque la lectura de datos es una transformación, y es una operación perezosa. Spark analizó solo un par de filas de datos para tratar de adivinar qué tipos debería haber en cada columna. La figura siguiente proporciona una ilustración del archivo CSV que se lee en un DataFrame y luego se convierte en una matriz local o una lista de filas:

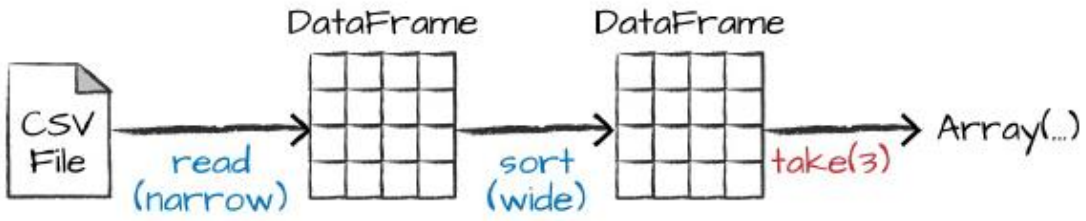

Conversión de CSV en matriz local

Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

Si realizamos la acción en el DataFrame, podremos ver los mismos resultados que vimos antes cuando usamos la línea de comandos:

```
flightData2015.take(3)
Array([United States,Romania,15], [United States,Croatia...
```

Ahora, clasifiquemos nuestros datos según la columna count, que es un tipo entero. La imagen ilustra este proceso:




Clasificación de datos
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

Ordenar no modifica el DataFrame. Usamos sort como una transformación que devuelve un nuevo DataFrame transformando el DataFrame anterior.

No sucede nada con los datos cuando llamamos a sort, porque es solo una transformación. Sin embargo, podemos ver que Spark está construyendo un plan sobre cómo ejecutará esto en el clúster. Podemos llamar a explain en cualquier objeto DataFrame para ver el linaje del DataFrame (o cómo Spark ejecutará esta consulta):

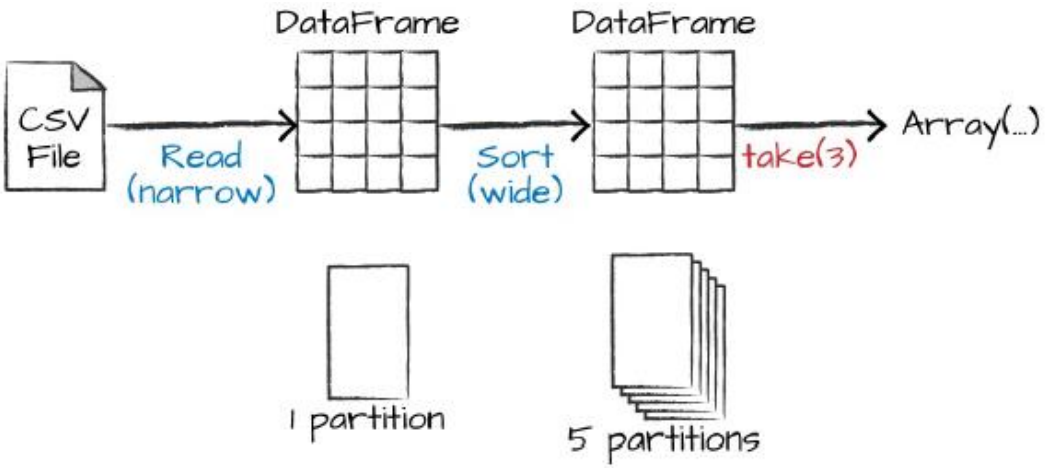
```
flightData2015.sort("count").explain()

== Physical Plan ==
*Sort [count#195 ASC NULLS FIRST], true, 0
+- Exchange rangepartitioning(count#195 ASC NULLS FIRST, 200)
+- *FileScan csv
[DEST_COUNTRY_NAME#193,ORIGIN_COUNTRY_NAME#194,count#195]
...
```

Explicar qué son los planes es un poco complejo, pero con un poco de práctica se hacen entendibles. No nos preocupemos demasiado por entender todo sobre los planes ahora, pensemos simplemente que son herramientas útiles para mejorar nuestro conocimiento a medida que avanzamos con Spark.

Ahora, tal como hicimos antes, podemos especificar una acción para poner en marcha este plan. Sin embargo, antes de hacer esto, vamos a establecer una configuración. Por defecto, cuando realizamos un shuffle, Spark produce 200 particiones. Establezcamos este valor en 5 para reducir el número de particiones:

```
spark.conf.set("spark.sql.shuffle.partitions", "5")
flightData2015.sort("count").take(2)
... Array([United States,Singapore,1], [Moldova,United States,1])
```

En la imagen se muestra dicha operación. Tengamos en cuenta que, además de las transformaciones lógicas, incluimos también el recuento de particiones físicas.




Reducción del número de particiones
Fuente: Spark's Toolkit, de Bill Chambers y Matei Zaharia

**ENLACE DE INTERÉS**

Puedes descargar los datos que hemos utilizado para que puedas practicar desde el siguiente enlace en casa.

https://www.bts.gov/browse-statistical-products-and-data

El plan lógico de transformaciones que construimos define un linaje para el DataFrame para que, en cualquier momento, Spark sepa cómo volver a calcular cualquier partición realizando todas las operaciones que tenía antes en los mismos datos de entrada. Esto es clave en el modelo de programación de Spark (programación funcional donde las mismas entradas siempre resultan en las mismas salidas, cuando las transformaciones en esos datos se mantienen constantes).

No manipulamos los datos físicos. En su lugar, configuramos características de ejecución física a través de aspectos como el parámetro de particiones shuffle. Terminamos con cinco particiones de salida porque ese es el valor que especificamos en la partición shuffle. Se puede cambiar esto para ayudar a controlar las características de ejecución física de los trabajos de Spark.

Podemos probar diferentes valores y ver el número de particiones. En la experimentación con valores diferentes, deberíamos comprobar como los tiempos de ejecución cambian drásticamente. Debemos supervisar el progreso del trabajo navegando por la interfaz de usuario de Spark en el puerto 4040 para ver las características lógicas y físicas de los trabajos.

**VIDEOTUTORIAL**

A continuación podrás ver un vídeo tutorial de uso de Spark que te servirá como información complementaria a lo estudiado en la unidad.

## 5.4 Profundización en Spark

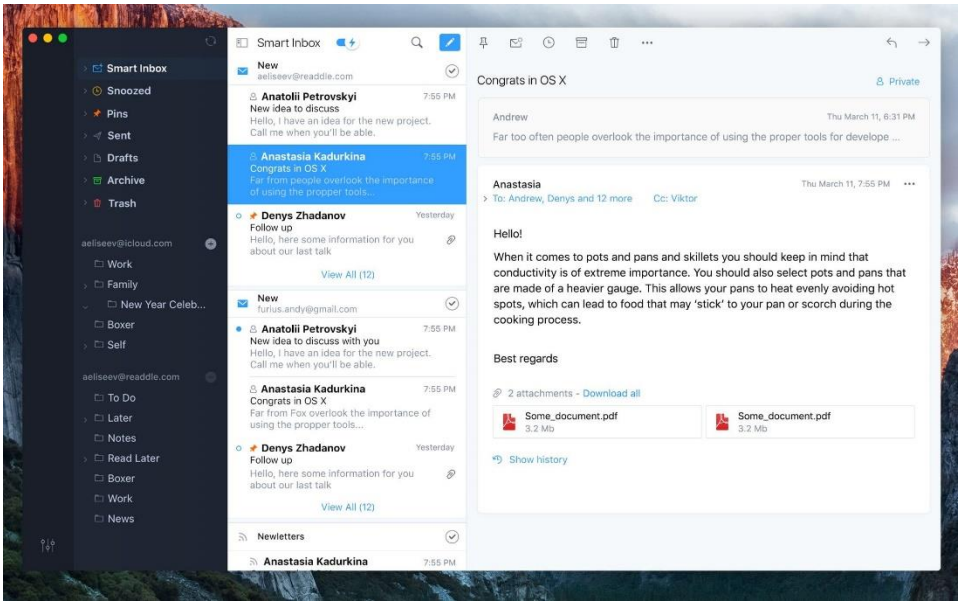
Además de la visión general de Spark, para profesionalizar más nuestros conocimientos de la plataforma, también hay que conocer diferentes librerías de Spark que, aunque tal vez no tengas necesidad inmediata de aplicarlas, sí que será interesante que las conozcas para poder responder a necesidades futuras que te surjan en tu actividad profesional.

Aprenderás mucho sobre Spark y las tecnologías relacionadas no solo leyendo esta unidad, sino que para obtener un mayor aprendizaje, es necesario que practiques escribiendo los ejemplos que se muestran, modificándolos para comprender mejor como funciona Spark y experimentando con tus propios casos prácticos. Una vez asentados los conocimientos básicos, no es difícil continuar una buena progresión de forma autodidacta.

Toda experiencia que tengas desarrollando aplicaciones fuera de Spark, seguro que ha dejado claro que experimentar con ejemplos de código es la mejor forma de que todo se vuelva más claro. Y Spark en eso, no es diferente a otras plataformas o entornos de desarrollo. Practicar y experimentar con ejemplos es el camino más directo para convertirse en un desarrollador sólido de Spark.

Para aquellos que después de haber estudiado esta unidad tengan claro que quieren profundizar más y dedicarse profesionalmente a Spark, sería muy recomendable que aprovechen otro recurso muy útil, que es la **lista de correo de Spark** y sus **foros**. **La comunidad de Spark es muy activa** y valiosa. No solo los desarrolladores de Spark responden a las preguntas, sino que también los usuarios experimentados de Spark ofrecen voluntaria y desinteresadamente su tiempo para ayudar usuarios novatos.

No importa con qué problema te encuentres. Es probable que alguien en la lista de correo de Spark haya resuelto ya ese problema en el pasado.




Interfaz del correo de Spark en Mac
Fuente: https://www.soydemac.com/wp-content/uploads/2016/07/spark-mac.jpeg

El acercamiento al big data y a Spark no tiene por qué ser igual para todas las personas. Es posible que si estás estudiando Spark es porque quieres convertirte en un científico de datos o, incluso ya lo eres y buscar actualizar y mejorar tus habilidades. Aunque es posible que tu perfil esté más orientado a ser un ingeniero de datos, que un científico. Pero también podría ser que tu rol sea el de usuario experto, que quiere conocer los máximos detalles posibles de Spark para extraer todo el rendimiento posible de la plataforma analítica a la hora de tomar las decisiones inherentes a su trabajo.

Según el rol que quieras jugar es verdad que tendrás necesidades ligeramente diferentes, pero, en realidad, la mayor parte del desarrollo de aplicaciones cubre un poco de ambos mundos. En principio, aunque hay opiniones divergentes según quien sea el consultado, una idea generalmente aceptada es que la carga de trabajo del científico de datos se centra más en la consulta interactiva de datos para responder preguntas y construir modelos estadísticos.

Mientras que el trabajo de ingeniero de datos se centra más en la escritura aplicaciones de producción que faciliten que muchos de los procesos sean repetibles y mantenibles. Estas aplicaciones podrían ser usadas por el propio científico de datos para contrastar sus modelos en la práctica o simplemente para los preparar datos para un análisis posterior.

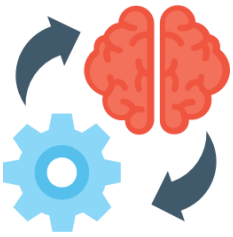
Sin embargo, no queremos ser tan específicos, ni considerar que los roles alrededor del big data son compartimentos estancos. Por ejemplo, los científicos de datos pueden usar las aplicaciones de producción de Spark a través de paquetes sin demasiada dificultad. Y los ingenieros de datos pueden utilizar análisis interactivos para comprender e inspeccionar sus datos para así construir y mantener pipelines, que son secuencias de procesamiento y pasos de análisis aplicados a datos para un propósito específico.

Si te habías aproximado al big data y a Spark de forma preliminar, sin realmente tener claro un motivo, más allá de aprender e incrementar conocimientos, tal vez ahora que ya dispones de una base inicial sería útil que decidas cuál de los caminos anteriores es el que más te interesa de modo que tu aprendizaje futuro esté más focalizado.

Sea cual sea el itinerario profesional elegido, no es suficiente con dominar la herramienta informática que es Spark, sino que hay que comprender las técnicas analíticas que se pueden usar en Apache Spark, como es el caso del aprendizaje automático. Tendremos ocasión de estudiar esos aspectos en otras de las unidades de este curso.

Una vez conocida la teoría y los aspectos conceptuales no será difícil invocar esas técnicas usando bibliotecas en Spark, de forma que cuando se tiene una formación básica en aprendizaje automático, bases de datos SQL y NoSQL y lenguaje R, todo toma mucho más sentido.

Otra alternativa para continuar el aprendizaje en el futuro sería enfocarse en operaciones y administración (por ejemplo, cómo administrar un clúster Apache Spark con docenas de usuarios) más que en las actividades de desarrollo y programación alrededor de Spark. Estamos hablando de aspectos como monitorización, depuración y configuración de Spark para ayudar en los retos técnicos que requieren de una ejecución de las aplicaciones especialmente eficiente y también trucos y recomendaciones que nos ayuden en el mantenimiento diario.



**RECUERDA**

De igual modo al estar Spark inmenso en un conjunto de tecnologías en un mercado tan dinámico, es importante estar al tanto de los cambios y novedades que se producen constantemente. Consolas como Pyspark y DataBricks, las plataformas de Spark más utilizadas hoy en día pueden cambiar por otras en el futuro.

Scala, lenguaje de programación en el cual Spark está basado, seguirá evolucionando y lanzando nuevas versiones al mercado. Lo mismo ocurrirá con Python o, probablemente en menor medida, con SQL, lo que impactará a la hora de diseñar tus aplicaciones en Spark.

# RESUMEN FINAL

A lo largo de esta unidad hemos comenzado un largo recorrido por el big data, donde hemos conocido los cambios que han provocado su aparición y la creciente importancia que ha tomado en los últimos años. De igual modo, ahora empezamos a vislumbrar los campos de aplicación que tiene el análisis de estos macro datos.

En particular, se ha presentado la metodología del ciclo de inteligencia, como herramienta práctica para proceder a realizar una explotación eficiente de los datos e informaciones en cualquier organización.

También se ha introducido el business intelligence. No solo se ha presentado su origen y puntos a favor, sino que también hemos realizado un análisis crítico en base a las opiniones de diversos autores que lo consideran una herramienta interesante, pero insuficiente para hacer una gestión completa de la información empresarial.

Hemos presentado Scala, el lenguaje de Spark, conociendo los aspectos comunes a cualquier lenguaje de programación con el uso de tipos de datos, las asignaciones, las funciones, tan relevantes en programación funcional, etc.

Dado que Spark no gestiona su propio repositorio, sino que está diseñado para operar sobre otros muchos, también hemos conocido alguna de las bases de datos de big data más populares y, especialmente, lo importante que tiene el enfoque de MapReduce para conseguir una capacidad de procesamiento distribuido realmente notable.

De igual modo, también ha sido imprescindible conocer los procesos ETL, que permiten tomar datos de diferentes fuentes e integrarlos para comenzar la explotación de estos.

Los dashboards han sido unos de los puntos finales que hemos tocado en este recorrido y hemos aprendido desde las diferencias existentes entre diversos tipos de dashboards hasta las recomendaciones más prácticas para diseñar e implementar uno de ellos.