

Tecnologías para el análisis de datos basadas en software libre

Guía de desarrollo y aplicación

Javier Díaz, María Alejandra Osorio
y Ana Paola Amadeo (coordinadores)

FACULTAD DE
INFORMÁTICA

e
exactas



UNIVERSIDAD
NACIONAL
DE LA PLATA

Tecnologías para el análisis de datos basadas en software libre

Guía de desarrollo y aplicación

Javier Díaz
María Alejandra Osorio
Ana Paola Amadeo
(coordinadores)

Facultad de Informática



Agradecimientos

A los alumnos que han pasado por la cátedra de Tecnologías Aplicadas a Business Intelligence desde el año 2011, por sus ideas, iniciativas y desarrollos, varios de los cuales ilustran los ejemplos de este libro.

A la Facultad de Informática de la Universidad Nacional de La Plata junto a la editorial EDULP, por respaldar y promover libros de cátedra que se generan a partir del conocimiento específico de la disciplina por parte de los docentes y se multiplica a partir del generado por los propios estudiantes.

A nuestras familias, por su apoyo permanente e incondicional.

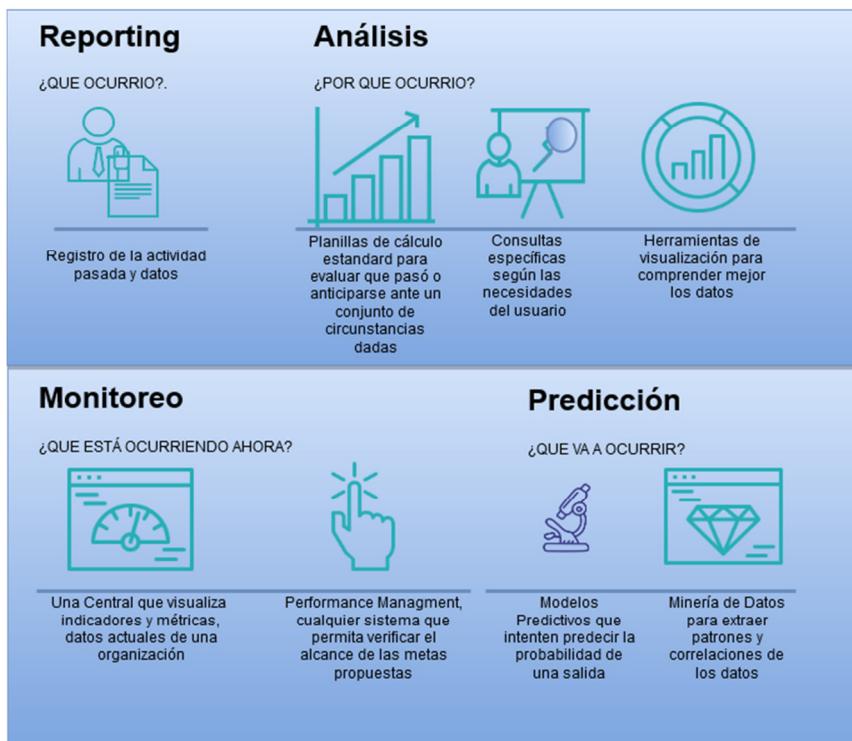
Índice

Introducción	5
1.- Comenzamos	7
Presentación de la problemática. Definiciones y conceptos principales que hacen a la Inteligencia de Negocio. Evolución. Presentación de desarrollos de estudiantes relacionados a esta unidad.	
2.- Modelo Dimensional	21
Técnicas para el modelado dimensional. Esquemas de Modelamiento Multidimensional. Definición y navegación de cubos de análisis de información. MDX. Presentación de casos de modelado dimensional.	
3.- Arquitectura del data warehouse	45
Arquitectura del Data Warehouse. Enfoques de Inmon, Kimball y Data Vault. Metodologías para el desarrollo de DW, comparativa con metodologías ágiles. ETL. Ejemplo concreto trabajado en la materia, utilizando herramientas basadas en Software Libre.	
4.- Análisis OLAP y Reporting	73
Definiciones. Tipo de navegación de los cubos. Un ejemplo con Pentaho Workbench. Herramientas de Reporting.	
5.- Introducción a Big Data y Ciencia de los Datos	91
Introducción. Motivación y Metodología. Modelos. Visualización.	
Anexo capítulo 2	115
Subsistema ETL de Kimball	
Anexo capítulo 5	118
Ejemplos de Análisis de Datos desarrollados en R y Python	
Los autores	158

Introducción

La manera en que los datos cambiaron la forma de aprender, de enseñar, las diferentes formas en que se generan los negocios, se cura, y muchísimas actividades de la vida cotidiana es innegable e inimaginable años atrás. Sin embargo, esto ocurrió sólo en aquellas organizaciones que encontraron la manera de interpretarlos y usarlos. Business Intelligence (BI) habilita a los mandos gerenciales a tomar ventaja de sus datos a partir del análisis de hechos concretos, en lugar de sólo opiniones e ideas. Es una disciplina que lleva muchos años en el mercado y en tiempo reciente, dado el caudal de datos disponibles de todo tipo, que es posible analizarlos en forma conjunta, y el desarrollo tecnológico, tanto en capacidad de procesamiento como en las soluciones móviles, facilitaron un interés mucho mayor con resultados tangibles. Si no estás familiarizado con BI este libro te va a ayudar a conocer desde lo más básico a lo último que se está trabajando en el área. Como verás, BI abarca técnicas y herramientas de la más diversa índole y ha generado también la necesidad de profesionales especializados en datos, para su gestión adecuada y también para su análisis a fin de sacar el mejor provecho de ellos.

Todos los conceptos principales que abarca Business Intelligence pueden resumirse en la siguiente infografía:



Son varios, cada uno de ellos con su conjunto de tecnologías, herramientas y técnicas, haciendo de Business Intelligence un mundo increíble para descubrir y trabajar, ¡hay mucho por hacer!

En el capítulo 1 presentamos los principales conceptos y casos de éxito. Además, incluimos trabajos de estudiantes que realizaron a partir del estudio de la problemática. En el capítulo 2 abordamos el modelado dimensional, una técnica que lleva más de 50 años aplicándose con mucho éxito, en general es el primer paso en el abordaje. En el capítulo 3 trabajamos con la arquitectura de un Data Warehouse, que los distintos autores lo toman como el corazón de una solución de Business Intelligence. También abordamos la problemática de la calidad de los datos, vista a partir de distintas estrategias de integración y limpieza de los datos, paso fundamental en cualquier solución de Business Intelligence. El capítulo 4 nos presenta OLAP y reporting, dos aplicaciones elementales para los usuarios finales de BI, y su implementación con Pentaho Designer y Workbench. Pentaho es una suite de productos open source que a lo largo del libro se estudian sus distintas componentes. Además, como el caudal de datos crece y crece día a día y la computación en la nube hace posible contar con una potencia de cálculo exponencial, en los últimos años se consolidó un nuevo campo de investigación y desarrollo: la ciencia de los datos, que toma de distintas disciplinas tradicionales como estadísticas, aprendizaje automático, minería de datos y analítica predictiva. Facilita el análisis de los datos a medida que se van generando, de forma incompleta y desordenada y es precisamente allí donde está el desafío, que se introduce en el capítulo 5. Big Data, el ecosistema Hadoop y una introducción a la ciencia de los datos se presentan en el último capítulo junto con un anexo con el desarrollo de ejemplos en R, el lenguaje de la ciencia de los datos y librerías del lenguaje de programación Python para este fin. ¡La visualización de grandes volúmenes es el gran desafío! Estudiar las mejores estrategias y técnicas es indispensable. Se incluyen trabajos de estudiantes de distintos años que desarrollaron sobre la temática utilizando mapas y librerías específicas.

Los autores somos docentes de la Facultad de Informática de la Universidad de La Plata y varios de nosotros trabajamos en áreas específicas de análisis de datos. Mucho de lo que vas a encontrar en este libro es fruto de nuestra experiencia y lo que compartimos y aprendimos con nuestros alumnos desde el año 2011 que comenzó el dictado de la materia. Esta 1º edición es nuestro punto de partida, desarrollamos muchos temas pero quedaron todavía muchos por incluir. ¡Y ese es nuestro próximo desafío!

Los invitamos también a recorrer nuestro blog <http://catedrasbi.blogspot.com> donde vas a encontrar trabajos de estudiantes de años anteriores, y también nos sigas en Twitter @catedrabi y participes de nuestro grupo de Facebook <https://www.facebook.com/groups/294053034035388/?ref=ts> donde publicamos artículos, novedades de la cátedra y demás información de interés. Por ejemplo este año nos visitaron de empresas del sector y también de la Dirección de Datos Abiertos de la Gobernación de la Provincia de Buenos Aires.

¡Todo comentario o sugerencia que nos ayude a mejorar serán bienvenidas!

CAPÍTULO 1

Comenzamos

En las organizaciones podemos encontrar procesos de negocio y procesos de toma de decisiones que actúan sobre su activo más importante: sus datos. Los procesos de negocio son aquellos realizados para soportar la actividad de la organización, por ejemplo en una empresa comprar, vender, gestionar los RRHH, la logística o los recursos financieros de una empresa, etc.; mientras que los procesos de toma de decisiones permiten decidir Cómo se van a gestionar cada una de esas áreas y procesos de negocio a partir del uso de la evidencia y de un análisis sistemático de sus datos. Durante la carrera nos enseñaron a crear soluciones eficientes, seguras, centradas en el usuario, utilizando distintos modelos y tecnologías, para los primeros procesos. Ahora te propongo que trabajemos sobre la segunda, los procesos relacionados con la toma de decisiones o la inteligencia de negocio “Business Intelligence”.

La primera pregunta que nos hacemos al comenzar la materia es “¿A qué se denomina Business Intelligence? Y como en la mayoría de los puntos donde es necesario introducir una nueva definición, comenzaremos con un caso de la realidad. En este caso, pensemos en una PYME de cerveza artesanal cuyo director ha convocado a una reunión a los gerentes para analizar las ventas. Cuando comienza la reunión descubren que ¡cada uno tiene su versión! Cada uno presenta su información, sus fuentes pero los resultados no son tan homogéneos y/o comparables como para hacer un análisis cuantitativo que permita obtener información confiable para la toma de decisiones. Por ejemplo, el gerente A dice que en la ciudad de La Plata se vendieron 10.000 botellas de cerveza negra y CABA se vendieron 100.000 de la misma variedad. Mientras que en la ciudad de Mar del Plata se informa una venta de ¡1.000 botellas! Mientras que el gerente B dice que en la ciudad de La Plata se vendieron 5.000 botellas de cerveza negra y CABA se vendieron 3.500 de la misma variedad. Mientras que en la ciudad de Mar del Plata se informa una venta de 7.000 botellas. Entonces comienzan las dudas, ¿qué fechas se está considerando? ¿es realmente fidedigna la información de Mar del Plata? ¿El término *venta* es la venta que se vendió al cliente o puede ser que quedó en stock en un supermercado o en una góndola? Es decir, las razones de las discrepancias pueden ser distintas:

- ✓ Distintos significados para los mismos términos.
- ✓ Distintos sistemas no integrados para obtener información consistente.
- ✓ Distintos períodos de tiempo para el análisis.

Cada uno de los participantes comienza a plantear su análisis y precondiciones, quedando al final de la reunión en que uno de ellos informará al resto, sin completar el orden del día. Esto es más habitual de lo que se supone. Para el caso planteado, una solución de BI permite analizar además otras aristas como el nivel de cumplimiento del proveedor, las ventas de otros productos asociadas a este, la estacionalidad de las ventas, etc., y, por tanto, establecer cuál es riesgo de un problema con el stock de ese producto y las repercusiones del mismo.

Consideremos este otro ejemplo: en 2007 Anne Milgram procuradora de Nueva Jersey cuando asumió se dio cuenta de que no tenía modo de saber a quién estaban enviando a la cárcel y tampoco si sus decisiones estaban llegando a más gente. En el siguiente video [TED](#) cuenta su experiencia para comprender la importancia de la toma de decisiones basadas en datos. No sólo calidad de datos y que sean fácilmente accesibles, sino también se necesita capacidades para comprender y utilizar los datos existentes. Mediante la utilización de datos y herramientas de análisis adecuadas el equipo de Milgram pudo reducir los delitos en un 26%.



He buscado por todo el país y he encontrado que entre un 5% y un 10% de todas las jurisdicciones en EE.UU. utilizan en la práctica algún tipo de herramienta de evaluación de riesgos; y, cuando observé estas herramientas, me di cuenta rápidamente del porqué: su administración era increíblemente cara, requerían mucho tiempo de dedicación y se limitaban a la jurisdicción local en la que se habían creado. Así que, básicamente, no podían adaptarse en tamaño o trasladarse a otros lugares.

Fig. 1-1. Video TED. IDB10x Datos para la efectividad de las políticas públicas (2017)

 El verdadero valor de la información se obtiene cuando podemos obtener conocimiento de ella. Este es el objetivo de BI

En toda organización pequeña, mediana, grande, con o sin fines de lucro, que cuenta con información digitalizada o no, surge la “necesidad” de contar con datos para la toma de decisiones sobre hechos concretos. Necesidad es aquello que no tenemos, o no tenemos como nos gustaría, y que nos incita a buscar una solución para....., y éste es el punto central, el “para qué”, que suele ser:

- Mejorar la calidad de datos: estamos hablando de proyectos destinados a mejorar la calidad de los datos, garantizar la integridad de los mismos, etc. Esto puede que no tenga que ver con los sistemas de BI porque se trata de procesos informáticos complejos y que requieren un alto nivel tecnológico o muchas horas de programación y verificación. No tienen que ver con la gestión, pero las herramientas de BI actuales brindan alguna solución para llevar adelante esta tarea básica.
- Excel caos: tantas planillas, como están, no es posible gestionarlas en forma integrada para los niveles gerenciales. No la puedo manejar o no me sirve para gestionar tal y como es mi organización.

- Necesidad de cruzar datos con otras aplicaciones propias o externas, como datos abiertos.
- Demasiada información para ser analizada de forma habitual.
- Es necesario automatizar los procesos de extracción y automatización de la información

Probablemente sea necesario crear un Data Warehouse con toda la información relevante para tomar decisiones...., y éste es punto fundamental para considerar que es BI: la toma de decisiones. La tecnología de BI es la que permite ejecutar mejor los procesos de toma de decisiones. En estos casos, lo importante es determinar qué información vamos a necesitar y para qué la vamos a necesitar, es decir, qué voy a hacer luego con ella para organizar la información de la mejor manera posible. La información para la toma de decisiones refleja un modelo de gestión y debe responder a él.

En los últimos años, el mercado de Business Intelligence ha tenido una clara evolución y se está consolidado como un mercado maduro. Para esto han contribuido la compra y venta de empresas por parte de SAP, Oracle, Microsoft, entre otras, y se generaron nuevas soluciones como PowerBI de MS, ClickView o Tableau. Se han desarrollado soluciones Open Source diversas y potentes como Pentaho Community, Talend e IBM integra herramientas open source y las potencian en soluciones propias y también compartidas como el caso de BlueMix con Hadoop y Hive. Algunas herramientas acumulan años de experiencia y poseen un claro modelo de negocios, y generan sinergias entre ellas. Es posible encontrar herramientas para bases de datos, como PostgreSQL o MySQL en su versión Community, como para minería de datos como es el caso de Weka de la Universidad de Waikato que se integra con Pentaho y RapidMiner. También para pequeñas, medianas y grandes empresas. Se incluyen nuevos focos de innovación como usabilidad de las soluciones de BI, visualización de la información, real time BI para el seguimiento de tópicos a través de las redes sociales por ejemplo, científicos de datos para bucear en los datos, que ha generado un tremendo interés en el mercado laboral, como se observa en la siguiente imagen de Google Trends sobre Business Intelligence y Data science y Data Analysis. Los números representan el interés de las búsquedas relacionadas con el valor máximo del gráfico para el tiempo especificado, y la región. El 100 indica la máxima popularidad para el término, 50 implica la mitad de popularidad y 0 equivale a una popularidad menor del 1% en comparación con el valor 100.

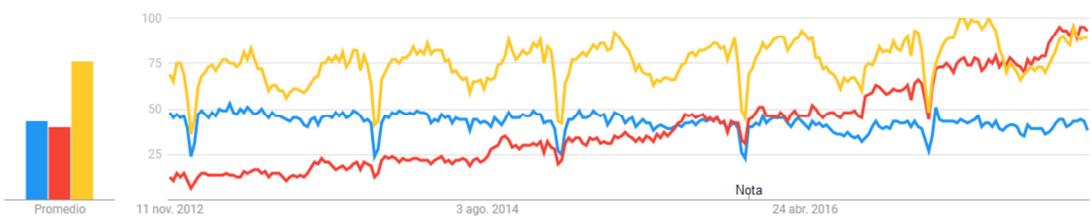


Fig 1.2 – Google Trends. Business Intelligence (azul), Data Science (rojo) y Data Analysis (amarillo)

Actualmente se identifica como una necesidad crítica de toda organización.

Seguimos avanzando con el tema y sin embargo aún no hemos dado una definición formal de BI. BI no es un término nuevo, en 1989 Howard Dresden, analista de Gartner, nos presenta una descripción más formal del término:

Conceptos y métodos para mejorar las decisiones del negocio, mediante el uso de sistemas de soporte basados en hechos.-

Otras definiciones importantes:

- ✓ Business Intelligence es un término genérico que incluye las aplicaciones, la infraestructura, las herramientas y las mejores prácticas que permitan el acceso y análisis de la información para mejorar y optimizar las decisiones y el rendimiento, también del glosario de Gartner IT.
- ✓ Es un conjunto de tecnologías y procedimientos que permite a las personas de los diferentes niveles de una organización acceder a los datos, analizarlos e interpretarlos. Howard (2007)
- ✓ La inteligencia de negocios es el conjunto de estrategias y metodologías que nos van a ayudar a convertir los datos en información de calidad, y dicha información en conocimiento que nos permita una toma de decisiones más acertada y nos ayude así a mejorar la competitividad. Salvador Ramos, SolidQ MS BI, (2017)
- ✓ Conjunto de metodologías, aplicaciones prácticas y capacidades enfocadas a la creación y administración de la información que permite tomar mejores decisiones. Introducción a BI. Curto Díaz. UOC. (2013)
- ✓ Es lograr que los gerentes y directivos de las organizaciones, y por extensión todos los usuarios de la información, tomen las mejores decisiones cada día accediendo de forma directa a la información “clave” de su negocio de manera ágil y sencilla. Zorrilla. UNICAM (2011)
- ✓ Se puede describir BI, como un concepto que integra por un lado el almacenamiento y por el otro el procesamiento de grandes cantidades de datos, con el principal objetivo de transformarlos en conocimiento y en decisiones en tiempo real, a través de un sencillo análisis y exploración. Bernabeu (2012).

Entonces, podemos decir que el desafío principal de BI es reunir y presentar de manera organizada la información referente a todos los factores relevantes que conducen el negocio y habilitar el acceso al conocimiento a los usuarios finales de manera fácil y eficiente, de acuerdo a su función, con el efecto de maximizar el éxito de una organización. Esta organización puede ser pequeña, mediana o grande.

Contar con una solución de BI en la organización brinda distintas ventajas, como las mencionadas por Curto Díaz, que mencionamos a continuación:

- ✓ Crear un círculo virtuoso de la información:

- ✓ Contar con información única, relevante, centralizada y consolidada.
- ✓ Automatización de la extracción y unificación de la información.
- ✓ Brinda herramientas para el análisis en línea de información.
- ✓ Facilita al usuario el acceso a información en forma dinámica e independiente para la toma ágil de decisiones.
- ✓ Facilita la identificación de indicadores claves para la organización.
- ✓ Permite predecir comportamiento futuro.
- ✓ Da respuesta a los siguientes interrogantes:
 - ¿Qué ocurrió?
 - ¿Dónde ocurrió?
 - ¿Por qué ocurrió?
 - ¿Qué ocurrirá?
 - ¿Qué está ocurriendo?
 - ¿Qué queremos que ocurra?

La situación actual se constituye como la evolución natural de los sistemas de análisis de información o DSS, ya desde la década de 1990. Un DSS es un sistema que colabora en las toma de decisiones gerenciales. Involucra el análisis de muchas unidades de datos de una manera heurística. Es el soporte de los tomadores de decisiones a nivel gerencial, permitiendo combinar el juicio humano e información objetiva. En los distintos años de la cursada de TABI, se construyeron mapas mentales entre todos los estudiantes para trabajar este concepto.

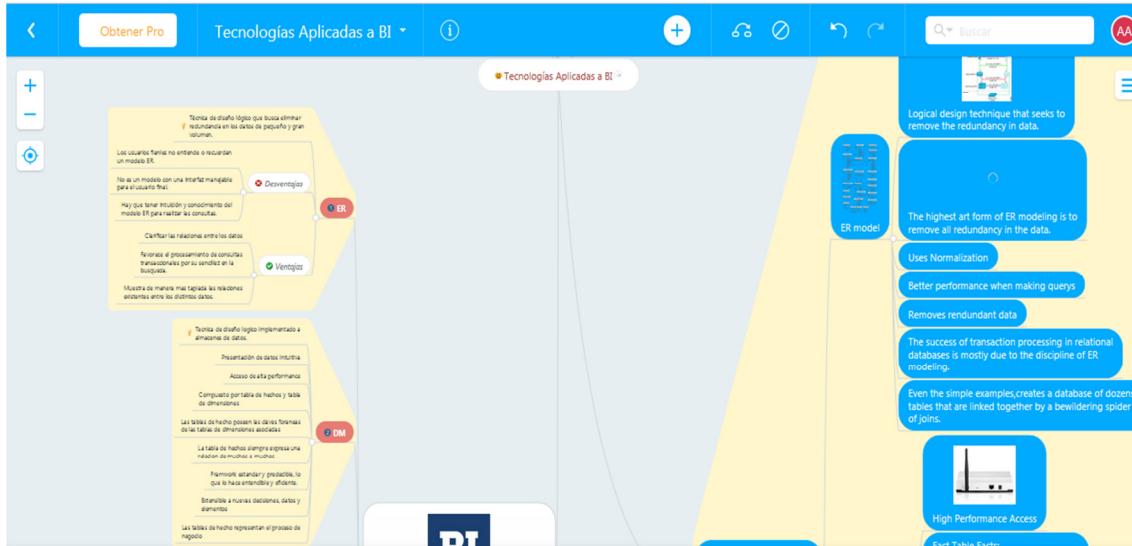


Fig. 1.3 – Mapa mental según el manifiesto ágil. <https://www.mindmeister.com/447414179#>

Datos, Información, Conocimiento

En mucha bibliografía solemos encontrar el uso indistinto de datos, información y conocimiento. Sin embargo, son términos diferentes que es importante que tengamos presente sus diferencias:

- Datos: Insumos básicos sin procesar con orientación valorativa. Por ejemplo un 8 en un examen de la materia A del legajo L1.
- Información: Resultado del procesamiento de los datos, su clasificación y ordenamiento en un contexto. Promedio de la materia A o promedio del legajo L1.
- Conocimiento: Resultados de la re significación de la información, a partir de la orientación a la acción y la aplicabilidad transformadora. Análisis de los últimos 5 años del promedio de la materia A, análisis de los alumnos que tienen promedio inferior al promedio.

Sistemas para capturar datos y sistemas para la toma de decisiones basadas en datos

En la actualidad, las tecnologías de la información han permitido automatizar los procesos de carácter administrativos o repetitivos, a través de los sistemas de información, caracterizado por altos tiempos de respuesta conocidos como OLTP *Online Transaction Processing*. Estos entornos son complejos para realizar tareas relacionadas con el análisis de datos, donde se requiere gestionar en forma conjunta información histórica, sumarizada y con flexibilidad suficiente para realizar cruces entre distintas variables o aplicar técnicas de análisis que requieren de operaciones de consultas y poca inserción o actualización, conocidos como OLAP *Online Analyst Processing*.

Los usuarios de los sistemas transaccionales son la rueda que hace girar todo el engranaje de la información de la organización. Ellos procesan los pedidos, dan de alta a nuevos clientes o usuarios, monitorean el estado de las actividades operativas y quedando todo registrado en forma adecuada y consistente. Los procesos están optimizados para que se ejecuten rápidamente, la misma tarea una y otra vez, trabajando con un registro por vez. El objetivo es la ejecución, en general no mantienen la información histórica, sino que agrega y actualiza información que refleja el estado actual del sistema.

Por su parte, el sistema para la toma de decisiones, mira la rueda de la organización y evalúa el rendimiento. El usuario de este sistema cuenta la cantidad de nuevos clientes, de pedidos, las compara con las de la semana anterior, analiza las quejas de los usuarios. Necesita información detallada para tener flexibilidad para responder preguntas complejas y cambiantes. Nunca trabaja con un dato a la vez. Este sistema está optimizado para realizar consultas sobre

grandes volúmenes de datos rápidamente, de niveles de cientos de miles de registros. En general demandan información histórica para evaluar la evolución de la organización a través de diferentes variables a lo largo del tiempo. Los usuarios, las necesidades, las estructuras y el ritmo de los sistemas operacionales son muy distintos a los sistemas para la toma de decisiones basadas en datos.

Es muy importante no incurrir en errores habituales, como hacer una copia del sistema operacional en un momento del tiempo, que se actualiza cada cierto período de tiempo, optimizar la infraestructura para que soporte operaciones de consultas sobre grandes volúmenes y nada más. Es necesario entender las necesidades del usuario final, cuestiones de usabilidad y visualizaciones adecuadas en el momento oportuno, que hacen al éxito de un sistema para la toma de decisiones basadas en hechos: que los usuarios finales realmente lo usen.

La siguiente tabla comparativa intenta resumir lo mencionado en el párrafo anterior:

	OLTP Online Transaction Processing	OLAP Online Analyst Processing
Objetivos	Asistir aplicaciones específicas Mantener la integridad de los datos	Comparaciones. Identificar tendencias. Almacenamiento de datos históricos.
Perfil del Usuario	Operativo. Entrada de Datos. Consultas Puntuales. Tareas Repetitivas	Gerencial. Toma de decisiones. Consultas globales.
Datos	Alineados por aplicación No integrados y no históricos	Alineados por dimensión. Integrados e históricos.
Acceso y Manipulación	Muchas y pequeñas transacciones, performantes, ABMs, Consistencia.	Pocas transacciones con muchos registros. Picos eventuales y carga masiva.

Tabla 1.1 – Comparativa de OLAP y OLTP

Analytic Maturity Model

En una organización, al comenzar el proceso de toma de decisiones basada en datos, es imprescindible saber en qué situación se encuentra. [TDWI Transforming Data With Intelligence](#) es una organización que desde hace más de 20 años asesora y brinda información y capacitaciones sobre inteligencia de negocio, analítica de datos avanzada, data warehousing, entre otros temas relacionados con análisis de datos. Un modelo de maduración sobre el análisis de datos es una herramienta muy útil para las organizaciones que ya tengan alguna iniciativa o no cuenten con alguna. Ayuda a crear una estructura alrededor de un programa de analítica y determina donde comenzar. Permite identificar las metas de la organización, y crea un canal de

comunicación a través del cual se pueda compartir está visión con toda la organización. También brinda una metodología para medir el grado y monitorear el avance de un programa, el esfuerzo necesario para completar la etapa y continuar a la siguiente.

La analítica de los datos involucra más que tecnología. Involucra a la organización, la infraestructura, la gestión y el análisis de datos, el gobierno o gobernanza y componentes organizacionales. Propone 5 dimensiones a partir de las cuales partir para implementar un proceso de análisis de datos en una organización.

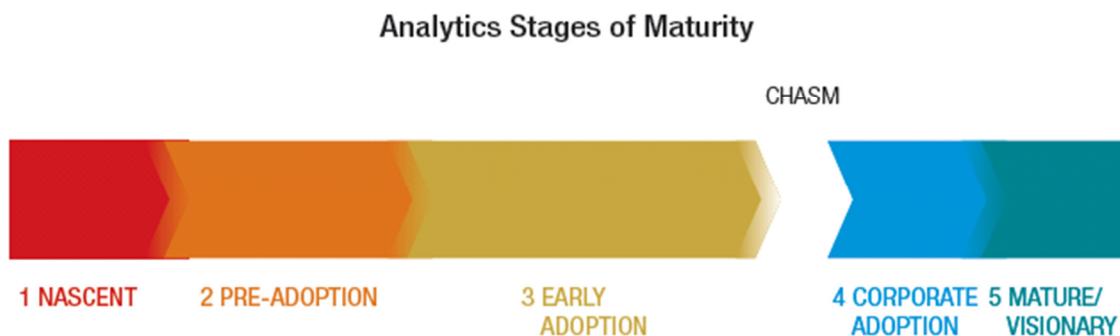


Fig. 1.3 – Modelo de maduración de Analítica de Datos en una organización

1. Nascent
2. Pre-Adoption: Las personas comienzan a entender las ventajas de BI para la toma de decisiones y la generación de salidas. Utilizan herramientas de BI de bajo costo o para descubrir conocimiento. Se comienzan a delinejar los Data marts o DW para consolidar datos, gestionar distintas fuentes y mejorar la calidad. Involucramiento del área de IT. El análisis de los datos es rudimentario, se comienzan a definir los usuarios y las cuestiones a responder.
3. Early-Adoption: La organización está comenzando a utilizar reportes y dashboard. Los usuarios suelen estar bastante tiempo en esta etapa. Las organizaciones pueden utilizar analítica para alguno de sus negocios, pero no todos. Es interesante para generar buenas prácticas y ver como los frutos pueden extenderse a otros departamentos. Pueden evaluar una infraestructura acorde, que permita acceder a múltiples fuentes de datos en forma integrada. En ésta etapa se comienza a delinejar aspectos de seguridad y acceso a los datos, metadatos a nivel departamental. Identificación de datos sensibles.

Para pasar del paso 3 al 4 se suelen encontrar muchos obstáculos. Se necesita de una política que apoye la analítica para la toma de decisiones y emponerar las capacidades de los usuarios. Además, los usuarios comienzan la lucha sobre quién es el dueño de los datos, o visiones particulares. Por esto es interesante incluir un ejecutivo exitoso, que motive al resto; IT y la organización trabajan en conjunto.

Se requiere de flexibilidad para acceder a los datos, a través de toda la organización, se requiere de una arquitectura de información unificada.

El personal de BI también requiere de nuevas habilidades para Hadoop, bases de datos NOSQL, self-service BI, entre otras. Es necesario establecer un ciclo de vida, una arquitectura de los datos.

4. Está incluido el usuario final y cambia la forma en que hace BI. Utiliza distintos tipos de datos, estructurados y no estructurados. Prevalece la cultura de los datos. Se cuenta con BD NoSQL, computación en la nube, datawarehouse, una arquitectura unificada accesible a través de distintos dispositivos. Se logra la democratización de los datos, todos pueden acceder, con el control de la gobernanza de los datos. Las organizaciones utilizan datos internos y externos. Ciclos de vida de los datos, auditoría y framework de desarrollo o proceso.
5. Los usuarios pueden definir sus propias visualizaciones, con un esquema de gobernanza e infraestructura adecuados. Como manejar la complejidad es la clave. Se desarrolla como un centro de excelencia, donde IT y el personal son un grupo que trabaja identificando, integrando datos internos y externos. Innovación en infraestructura y tecnología.

Llegar al punto 5 es difícil, pero es imprescindible el cambio de metodología en la toma de decisiones, que deje de ser un “dedómetro” para pasar a ser basada en datos concretos, medibles, a partir de diferentes variables, con análisis de riesgos reales que permitan minimizarlos.

Tecnologías Aplicadas a Business Intelligence

La inteligencia de negocio involucra un conjunto de tecnologías, que cada vez se va diversificando más para brindar soluciones en el menor tiempo posible, que permitan responder a las necesidades de todos los actores, sobre datos en distintos formatos, que están en permanente expansión y cambio. A continuación se presenta un conjunto de tecnologías, que no intenta ser exhaustivo, que se van a ir trabajando en los distintos capítulos del libro y durante la cursada de la materia:



- Integración de Datos: generalmente, los datos de los sistemas operacionales, necesitan transformaciones para que las herramientas de explotación se apliquen correctamente. Esta disciplina, se encarga del diseño, optimización y mantenimiento de estructuras de datos, tanto operacionales como analíticos.
- Data Warehouse: Es un entorno orientado a la consulta de datos, separado del operacional, que se nutre de la información de éste y otras fuentes externas, con el fin de utilizar dicha información como fuente única para la obtención de conocimiento.
- OLAP: Es una disciplina de generación de conocimiento deductivo o para obtención de información como verificación de hipótesis. Son herramientas que presentan los datos al usuario con una visión multidimensional, de manera rápida e interactiva.
- Big Data: datos de gran volumen, diversidad y complejidad que escapan a las soluciones convencionales para manejarlos de manera eficiente y económica.
- Minería de Datos
- Análisis predictivos: técnicas estadísticas de aprendizaje automático, modelización y minería de datos para analizar datos actuales e históricos. En los próximos capítulos se analizarán con más detalle
- Análisis visual: visualización de datos, fundamental para comprender y tomar decisiones. Más adelante en el libro se abordará esta temática.
- Previsiones: interpretar lo que va a suceder a través de indicios y señales, ver con anticipación.
- Tablero de control: herramienta para diagnosticar adecuadamente una situación. Integra varios indicadores relacionados de gestión.
- Reporting: entendido como elaboración de informes de gestión, con frecuencia diaria o semanal. Debe ser ágil, inmediato y flexible.

Todo esto con el soporte y la evolución de las bases de datos. A modo de introducción en este punto del libro es interesante la tabla que presenta Marta Zorrilla de la Universidad de Cantabria <https://www.scribd.com/document/247735327/Data-Warehouse> .

Hito Histórico	Pregunta de Negocio	Tecnología que lo posibilita	Sumunistrador	Característica principal
Recolección de datos (1960)	¿Cuáles fueron mis ingresos en los últimos 5 años?	Computadoras, cintas, discos, DBMS jerárquicos y en red	IBM / CDC	Datos históricos
Acceso a los datos (1980)	¿Cuántas unidades vendí el mes pasado en España?	Bases de Datos relacionales	Oracle, Sybase, Informix, IBM, Microsoft	Datos dinámicos a nivel de registro.

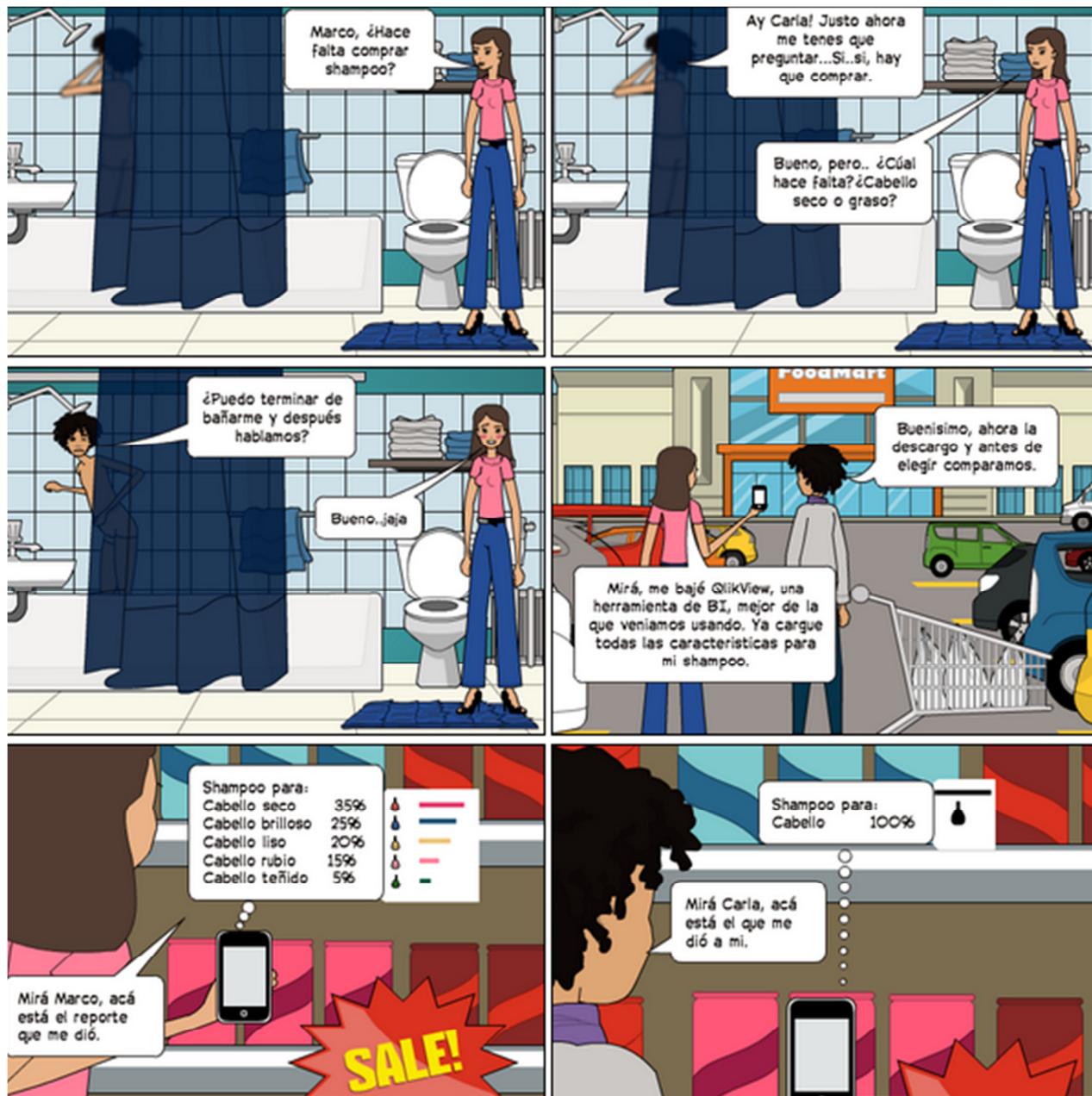
Data warehousing y Soporte para la toma de decisiones (1990)	¿Cuántas unidades vendí el mes pasado en España en relación con Europa?	Online Analytic Processing, gestores multidimensionales,	Cognos, Business Object, Microstrategy, Oracle, etc.	Datos dinámicos en múltiples niveles o jerarquías (histórico)
Minería de Datos (2000s)	¿Cuántas serán las ventas del último mes en Europa?	Algoritmos avanzados, data streaming, bio data, RDBMS	SPSS/Clementine, IBM, SAS, Oracle, etc.	Datos de prospección (análisis de mercado, de riesgos)

Y podemos completar la tabla con la siguiente fila, el año es en el que Peter Drucker se publicó el artículo “Data Scientist: The Sexiest Job of the 21st Century” La temática está en su “primavera” actualmente, dado que ha evolucionado la infraestructura, los servicios y todo lo necesario para soportar potencia de cálculo, almacenamiento y generación de conocimiento a partir de esto.

Big Data y Ciencia de los Datos (2012)	¿Cuál es la venta ahora en Europa según la campaña que lanzamos ayer?	Computación en la nube, aprendizaje automático, IAS, grandes volúmenes de datos provenientes de las redes sociales, sensores, internet de las cosas por ejemplo	Bases de Datos no estructuradas como MongoDB, BlueMix de IBM, Azzure MS, Amazon Web Services, Cloudera, sólo por citar algunos	Datos estructurados, no estructurados y semi estructurados.
--	---	---	--	---

Casos Prácticos

A continuación se presenta una selección de historietas que desarrollaron estudiantes de la cátedra sobre los conceptos iniciales de BI.



Luciano Frazia, Julieta Corvi. Cursada 2016.



Da Costa, Faro y Titarelli. Cursada 2016.



Bibliografía

- Kimball, R (1997) A Dimensional Modeling Manifesto <http://www.kimballgroup.com/1997/08/a-dimensional-modeling-manifesto/>
- Inmon, B. (2005) Building the data warehouse. Fourth edition, John Wiley and Sons, NY, USA.
- Díaz, C. (2013) Introducción al Business Intelligence. Ed. UOC.
- Transforming Data With Intelligence <https://tdwi.org/pages/about-tdwi/tdwi-business-intelligence-and-data-warehousing-education-and-research.aspx>
- Zorrilla M. (2011). Data warehouse y OLAP. Universidad de Cantabria .
<https://www.scribd.com/document/247735327/Data-Warehouse>
- Drucker, P. (2012). *Data Scientist: The Sexiest Job of the 21st Century*. Harvard Review
<https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century>
- Datos para la efectividad de las políticas públicas. Banco Interamericano de Desarrollo
<https://www.youtube.com/watch?v=xNsZIipFPEM>
- Bernabeu, D. (2012) <http://tgc-hefesto.blogspot.com.ar/2010/07/hefesto-v20.html>
- Ramos, S (2017) ¿Qué es Business Intelligence? Introducción a los sistemas de BI
<http://blogs.solidq.com/es/business-analytics/que-es-business-intelligence-introduccion-01>
<http://blogs.solidq.com/es/business-analytics/herramientas-sistemas-bi-situacion-actual-tendencias-07>
- Gartner IT Glossary; Defining the IT Industry. <http://www.gartner.com/it-glossary/>
- Howson Cindi (2007) "Successful Business Intelligence: Secrets to Making BI a Killer App". Ed. McGraw-Hill Osborne Media. 2007. ISBN-P: 978-0-07-149851-7

CAPÍTULO 2

Modelo Dimensional

Uno de los activos más importantes en una organización está basado en sus datos, esta es la base fundamental para construir conocimiento.

A medida que avanza el desarrollo tecnológico, cada vez son más económicas y más rápidas las memorias de almacenamiento y esto promueve el registro de cada vez más eventos.

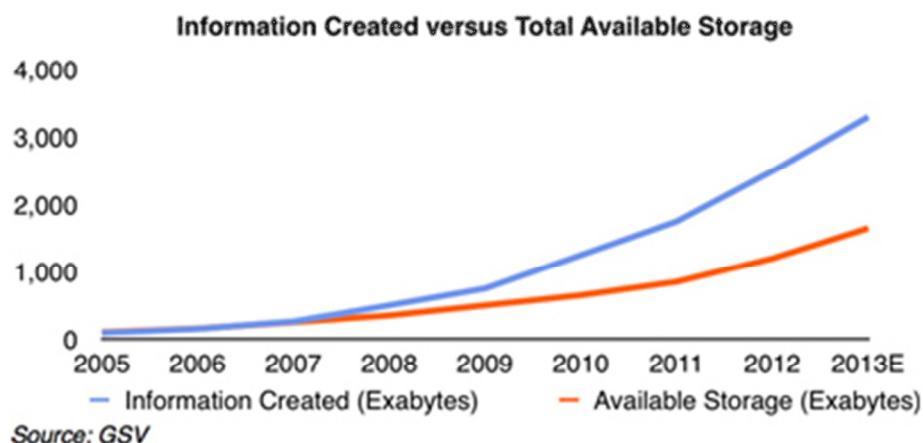


Fig. 2.1 - GSV Capital. <http://gsvcap.com/wp/market-commentary/san-francisco-37/>

Mucha información tampoco sirve de nada si no nos es posible asimilarla y comprender su significado. Y sirve poco si para logra asimilarla demoramos demasiado tiempo. Tener una habitación repleta de libros de contabilidad sirve de poco si necesitamos hacer una evaluación financiera para el día siguiente.

Así nos vemos en la necesidad de contar con alguna herramienta que nos facilite la tarea de compilar y procesar la información que tenemos para que su compresión sea más rápida y fácil de asimilar. A este tipo de herramientas, como vimos en el capítulo anterior, se los denomina en general Inteligencia Empresarial o BI por sus siglas en inglés “Business Intelligence”.

Al corazón de estos sistemas se los denomina Almacén de Datos (Data warhouse), dado que pueden almacenar la información de una manera eficiente para su análisis. Y de hecho en la bibliografía se los suele nombrar juntos, bajo el término “Sistemas DW/BI”.

Objetivos de un DW / BI

- Debe facilitar el acceso a la información. La estructura debe ser simple para los usuarios finales, las etiquetas deben tener nombres de fácil y rápida interpretación.
- Debe presentar la información de manera consistente.
Los datos deben ser confiables, si dos datos tienen el mismo nombre, deben significar lo mismo.
- Adaptable a los cambios. Debe ser diseñado de manera que admita modificaciones.
Los cambios deben ser transparentes al usuario final.
- Debe dar respuesta en tiempos razonables.
- Debe proteger el acceso a información confidencial.
- Debe servir como fuente autorizada para el DDS. La consecuencia más importante del DWH son las decisiones que se toma a partir de la información que brinda.
- Debe ser aceptado por los usuarios del negocio.
No importa cuán elegante sea, si los usuarios no lo adoptan y utilizan activamente.
Es diferente a los sistemas operacionales, donde su utilización es obligatoria.
Simple y Rápido, son características fundamentales para este objetivo.

Sistemas operacionales vs Sistemas BI / DW

Esta información es casi siempre utilizada con dos propósitos: registrar eventos y analizarlos. **Los sistemas operacionales** son mayormente quienes registran los eventos y por lo tanto ESCRIBEN los datos, y **el DW/BI** fundamentalmente los LEE para analizar y fundamentar decisiones. Estos tipos de sistemas tienen necesidades totalmente diferentes, que se mencionaron también en el capítulo anterior y aquí profundizamos.

Los sistemas operacionales:

- Con el registro de cada evento, **guardan unidades pequeñas de información**, con una frecuencia muy alta. (Cientos o miles de operaciones por hora)
- Cada registro debe garantizar integridad. (Ej. Factura y renglones)
- La unidad de datos es horizontal (Pocos registros, muchas columnas)

Estas necesidades hacen que se estructuren los datos de tal manera que eviten redundancia y faciliten la escritura constante de unidades de datos.

En cambio un sistema de DW/BI:

- Leen grandes volúmenes de datos
- Debe ser simple y entendible
- La unidad de datos es vertical (muchos registros, pocas columnas)

Por lo cual, necesitan una estructura de datos que privilegie la simplicidad y la velocidad de lectura de datos en favor de la eficiencia en la escritura de los mismos.

La estructura de datos para un DW/BI, es la que motiva la técnica de **modelado dimensional**.

Modelado Dimensional

El modelo dimensional es una técnica de diseño lógico que provee una forma sencilla de navegar la información. Contiene la misma información que el modelo operacional y pueden soportar las mismas consultas y el mismo análisis final. En el modelo dimensional los datos se muestran de manera diferente, en un formato simétrico, posee más eficiencia en las consultas y es resistente a los cambios.

El modelado dimensional es una técnica ampliamente aceptada para estructurar datos analíticos porque cumple con 2 requerimientos primordiales:

- Entrega información de forma entendible a los usuarios del negocio
- Lo hace en un tiempo aceptable

Simplicidad, es la clave fundamental para que los usuarios entiendan fácilmente los datos y los programas los naveguen con eficiencia. Es una técnica que lleva más de 50 años de vigencia, provee un nivel de simplicidad y detalle que establece un puente entre los especialistas de TI y los ejecutivos expertos en el negocio. Por ejemplo, un gerente de una empresa que vende productos a distintos supermercados realiza la siguiente afirmación sobre su labor “Nosotros vendemos productos en varios países y medimos nuestro rendimiento todo el tiempo”

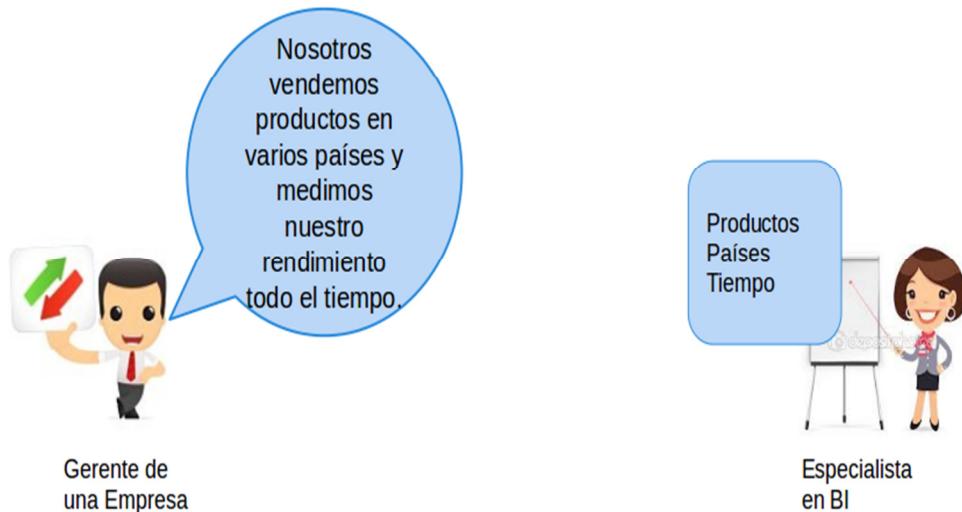


Fig. 2.2 - Modelo Dimensional

El especialista en BI debería prestar atención al especial énfasis que realiza sobre producto, países, tiempo. Una forma intuitiva de pensar el negocio es como un cubo de datos, donde cada extremo son los productos, los países y el tiempo, como se presenta en la siguiente imagen.

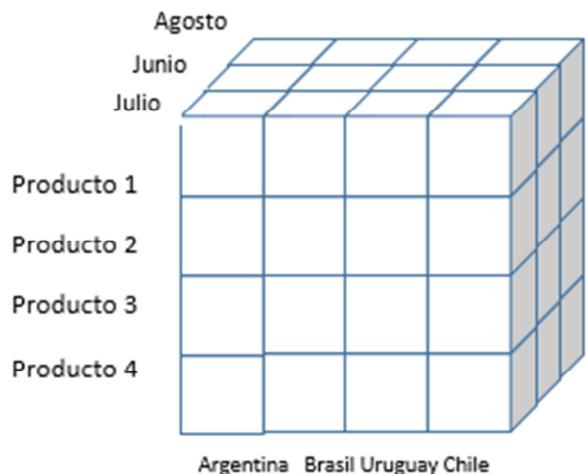


Fig. 2.3 - Cubo de Información

Es posible navegar ese cubo a través de las distintas dimensiones, como analizar la evolución de las ventas del Producto 1 en los distintos países, o analizar las ventas en Argentina de los distintos productos. Además, cada punto dentro del cubo representa una medida como un volumen de ventas o beneficio a través de la combinación de producto, país y momento en que fue registrado. La clave es iniciar un modelo de datos dimensional con un análisis sencillo para todas las partes involucradas, que permita obtener un modelo dimensional claro y conciso a lo largo de todas las etapas del proyecto. Si es complejo es altamente probable que el usuario final lo rechace y que las consultas demanden demasiado tiempo.

Es común que en la jerga se nombre a los modelos en 3º forma normal como “modelos relacionales”. Denominaremos a estos modelos como “normalizados”, dado que tanto estos como los modelos dimensionales pueden representados en Bases de datos Relacionales y graficarse con Diagramas de Entidad-Relación (DER).

Una diferencia clave entre los modelos de datos normalizados y los modelos dimensionales, es obviamente su grado de normalización. Los modelos normalizados, son muy útiles en sistemas operacionales, porque las miles de instrucciones INSERT y UPDATE, modifican el dato en un solo lugar.

Sin embargo, estos modelos normalizados, son demasiado complicados para realizar consultas analíticas. Un usuario de DW/BI no debería tener que reunir varias tablas y utilizar consultas complejas.

 Un modelo dimensional contiene la misma información que un modelo normalizado, pero organizado de tal manera que es más fácil de comprender, más rápido de consultar.

Dimensiones y Hechos

Los modelos dimensionales están compuestos por tablas de HECHOS (facts) y tablas de DIMENSIONES (dimensions).

Las tablas de HECHOS contienen los valores medibles de un proceso, como por ejemplo montos, cantidades, etc. Cada registro en la tabla de hechos, está relacionado a un evento del proceso. Una venta puede identificarse como un proceso de negocio. Es importante almacenar la medida al menor nivel o granularidad, para que pueda ser aprovechada en todas sus perspectivas, por toda la organización. Esto permitirá constituir un repositorio único de datos consistentes, que pueda ser accedido por todos los usuarios tomadores de decisión.

El término hecho representa una medida del negocio. Volvamos al ejemplo de las ventas y la caja registradora en un supermercado. Cada vez que se registra una transacción, se registra un producto, la cantidad vendida de ese producto y el costo total. Se transforma en Tabla de Hechos Ventas

Tabla de Hechos Ventas



Se transforma en

Fecha (FK)
Producto (FK)
Sucursal (FK)
Promoción (FK)
Cliente (FK)
Vendedor (FK)
Transacción #
Monto en pesos
Unidades vendidas

Cada fila en la tabla de hechos se transforma en un evento medible. Los datos de cada fila poseen un nivel de detalle específico, referido como grano, como una fila por producto vendido en una transacción.

Cada fila en la tabla de hechos debe tener la misma granularidad. El nivel de granularidad afecta a las preguntas que se pueden hacer y el espacio que ocupa la información. Esto implica también la gestión de índices adecuada.

Es importante crear tablas de hechos con un nivel de detalle único, para asegurar que los hechos medidos no se cuenten más de una vez.

La idea que un evento medible en el mundo real tenga una relación uno a uno en la tabla de hechos es un principio base para el modelo dimensional. Todo el resto se construye a partir de este principio.

Características de las tablas de hechos

Los eventos más útiles son los numéricos y aditivos, como la cantidad de dinero involucrada en la venta. La aditividad es fundamental porque raramente se referencia como una única fila en la tabla de hechos, en general se recuperan miles o millones de filas a la vez.

En el caso de las ventas, cada pasada de productos por la caja registradora aportan al total. Las medidas pueden ser semi aditivas, como un balance, que no puede ser llevado a través de la dimensión tiempo. Los hechos no aditivos, como precio unitario, nunca pueden ser sumados y se utilizan en cuentas o promedios.

Los hechos se suelen describir como algo valuado en forma constante, como el costo total de una transacción en una venta.

Teóricamente es posible tener medidas textuales, aunque es raro verlas implementadas dado que los valores de un texto a analizar, como los campos de entrada de un formulario, es imposible de analizar como un hecho. En general es una descripción de algo, que puede ser implementada como una lista de valores discretos. Es importante tratar de colocar los datos textuales en dimensiones, donde pueden ser correlacionadas más efectivamente con las otros atributos de las dimensiones textuales y consumir menos espacio. A menos que el texto se distinto en cada fila, tiene que manejarse a través de dimensiones.

Se recomienda no llenar las tablas con 0. Las no ventas de un producto no implica insertar una fila en la tabla de hechos con valores nulos. Las tablas de hechos suelen ser profundas en cuanto a cantidad de filas pero angosta en cantidad de columnas. Es importante tener presente el uso del espacio.

Todas las tablas de hechos tienen FKs que las conectan a las dimensiones. Por ejemplo, el id de producto con la tabla Productos.

La tabla de hechos suele tener su propia PK compuesta por un subconjunto de FKs, en general llamadas claves compuestas. La tabla de hechos expresan relaciones muchos a muchos.

En general se cuenta con un puñado de dimensiones que representan la unicidad en cada fila de la tabla de hechos. Luego de identificar este subconjunto de dimensiones, el resto de las dimensiones toman un valor sencillo dentro del contexto de la tabla de hechos.

Granularidad de los hechos

La granularidad es el nivel de detalle del proceso que se registra en la tabla de hechos del modelo dimensional.

Por ejemplo, para un modelo de un proceso de ventas, se puede definir la granularidad como un registro por cada ítem de factura.

Siempre se recomienda establecer la granularidad al máximo nivel de detalle.

De esta manera es más fácil dar respuesta a nuevas consultas (antes no previstas), extendiendo el modelo con nuevas dimensiones.

Dimensiones

Las tablas de DIMENSIONES contiene la información textual que describe los hechos. Las dimensiones constituyen las compañeras indispensables de la tabla de hechos dado que contienen el contexto asociado a los eventos medibles del proceso de negocio. Suelen contener muchos atributos o columnas, entre 50 y 100 es habitual. Sin embargo, suelen tener pocas filas. Cada dimensión contiene una única clave primaria, que sirve como integridad referencial única para poder referenciarla únicamente en la tabla de hechos.

También se la puede ver como una vista del proceso de negocio. Por ejemplo, para una venta tenemos el cliente que ha comprado, la fecha en que ha comprado, etc. Puede ser interesante recuperar las ventas que hizo un determinado cliente.

Siguiendo el ejemplo de la tabla de hechos Ventas, podemos tener las siguientes dimensiones:

Dimensión Producto

Id producto (PK)
Número SKU (**Clave natural**)
Descripción
Nombre de Marca
Nombre de Categoría
Nombre de Departamento
Tipo de paquete
Tamaño del paquete
Peso
Unidad de medida del peso
Tipo de almacenamiento
Tiempo de caducidad
Alto
Ancho
Profundidad

Los atributos tienen como objetivo ser la fuente primaria de las consultas, agrupamientos y reportes. En un reporte, los atributos identifican los requerimientos o las palabras de la consulta, por ejemplo en una venta, un analista en marketing puede requerir consultar por marca entonces la marca debe ser un atributo de la dimensión.

Además, los atributos hacen usable y comprensible el sistema de DW/BI. Es muy importante que los atributos cuenten con palabras reales y no abreviaciones crípticas, minimizar el uso de códigos por texto manteniendo la consistencia entre los atributos de todas las dimensiones para mantener la consistencia de todo el sistema.

Pueden existir situaciones en las cuales los códigos poseen legitimidad por sí mismos. En estos casos los códigos deberían aparecer como atributos de las dimensiones. Y si los códigos tienen su

propio significado, por ejemplo los dos primeros dígitos indican el producto y los siguientes la región, deberían ir en dimensiones separadas que permitan el filtrado y agrupamiento.

La calidad de un sistema de DW/BI depende de la calidad y profundidad de los atributos. Cuanto mayor palabras propias del negocio tengan y sean de calidad mucho mejor. Además es importante que tengan la capacidad de poder navegar los cubos a través de diferentes cortes que sean significativos.

Los atributos se constituyen como los puntos de entrada de los datos, definen los agrupamientos y la capacidad de análisis del sistema de DW/BI.

La siguiente tabla presenta un ejemplo de dimensión, que representa relaciones de jerarquías. Por ejemplo, productos pueden “navegarse” a agruparse según la marca, y ellas en categorías. Para cada fila se tiene almacenada la marca asociada y la categoría. La información jerárquica se almacena redundante en forma intencional, para mejorar la performance de las consultas. Es posible normalizar este esquema a través de FKs y tablas asociadas, en un esquema denominado **copo de nieve**. En lugar de la normalización en 3° forma normal, se incluyen en la misma tabla muchas relaciones uno a uno. Dado que las tablas de dimensiones posee pocas filas, no afecta la performance o armar un esquema de copo de nieve no aporta una mejora significativa. Lo importante es que sea simple y accesible.

Id producto	Descripción	Marca	Categoría
1	Tita	Terrabusi	Golosinas
2	Rodhesia	Terrabusi	Golosinas
3	Huevo Kinder	Kinder	Golosinas
4	Gomitas	Mogul	Golosinas
5	Pico Dulce	Lheritier	Golosinas
6	Cerealitas	Arcor	Galletitas
7	Pepitos	Terrabusi	Galletitas
8	Sonrisas	Terrabusi	Galletitas
9	Panal	Okebon	Galletitas
10	Melitas	Bagley	Galletitas

Tabla 2.1 - Ejemplo de tabla de hechos

Los HECHOS responden el ¿CUANTO ?

Las DIMENSIONES, el ¿QUIEN? ¿QUE? ¿CUANDO? ¿DONDE? y ¿POR QUE?

Jerarquías de las dimensiones

Las tablas de dimensiones contienen la información relacionada a los eventos registrados en la tabla de hechos, como la fecha, el lugar, el responsable, etc.

Esta información, normalmente se puede visualizar en forma jerárquica, de manera de poder descender por dicha jerarquía obteniendo mayor nivel de detalle, o ascender para visualizar información resumida.

Por ejemplo, una de las dimensiones más comunes de ver, es la dimensión “tiempo”.

Esta dimensión se puede visualizar con las siguientes jerarquías:

Año > Mes > Día

o bien

Año > Trimestre > Mes > Día

o también

Año > Semana del Año > Día

Una misma tabla de dimensiones puede visualizarse con distintas jerarquías según el objetivo que se persiga.

Otra manera de pensar la naturaleza complementaria de la tabla de hechos y dimensiones es organizarlas a un reporte. Los atributos de las dimensiones proveen brindan los filtros y etiquetas de los reportes, mientras que la tabla de hechos provee los valores numéricos.

Fácilmente se deduce el código SQL de este reporte:

SELECT con INNER JOIN

```
d_almacen.ciudad_nombre,  
d_producto.marca_nombre,  
sum(h_ventas.importe) AS Importe  
FROM d_almacen, d_producto, d_tiempo, h_ventas  
WHERE d_tiempo.mes_nombre = 'Junio'  
AND d_tiempo.año = 2016  
AND d_almacen.id_almacen = h_ventas.id_almacen  
AND d_producto.id_producto = h_ventas.id_producto  
AND d_tiempo.id_fecha = h_ventas.id_fecha  
GROUP BY d_almacen.ciudad_nombre, d_producto.marca_nombre
```

Las dos primeras líneas de la sentencia SELECT identifican los atributos de las dimensiones, seguido por las métricas agregadas de la tabla de hechos.

La sentencia FROM identifica todas las dimensiones involucradas. La primera línea del WHERE identifican el filtro y luego los joins entre la tabla de hecho y las dimensiones. Finalmente, el GROUP BY establece la agregación del reporte.

Esquemas para estructurar los datos. Esquema Estrella y Copo de Nieve

Los modelos dimensionales implementados en bases de datos relacionales, se denominan como “**esquemas estrella**” (star schemes), mientras que los que se implementan en bases de datos multidimensionales son referenciados como “**OLAP**” (OnLine Analytical Processing)

El nombre “estrella” se debe a la forma en que se ve reflejado el modelo en un diagrama de entidad-relación.

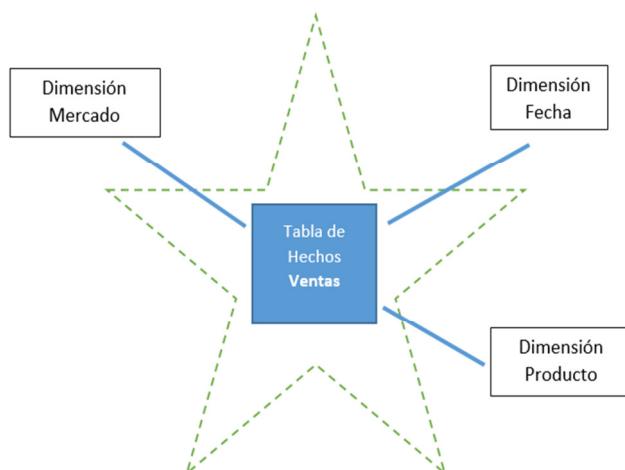


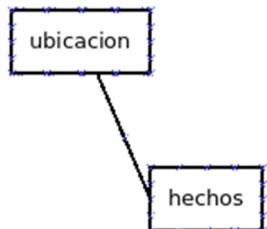
Fig. 2.4 - Esquema estrella

La tabla de hechos es la central, que contiene todos los datos medibles, y también una clave foránea a cada tabla de dimensión que describen los hechos.

Desnormalizar para simplificar

Un aspecto clave en los modelos dimensionales y que generalmente cuesta asimilarlo por quienes están familiarizados con los modelos relacionales, es que las tablas de dimensiones están desnormalizadas.

Supongamos la siguiente tabla de dimension “ubicacion”



PAIS	PROVINCIA	CUIDAD
ARGENTINA	BUENOS AIRES	SALTO
ARGENTINA	BUENOS AIRES	CARMEN DE ARECO
ARGENTINA	BUENOS AIRES	ROJAS
ARGENTINA	BUENOS AIRES	PERGAMINO

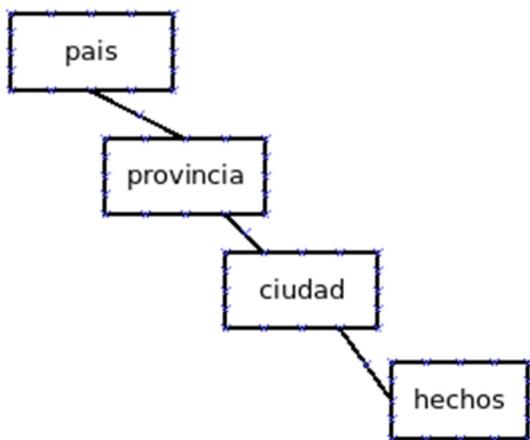
Vemos que el dato de PAIS y PROVINCIA se repiten.

Es común, verse tentado a normalizarlo, separando la información en 3 tablas

ID_PAIS	PAIS
1	ARGENTINA

ID_PROVINCIA	PROVINCIA	ID_PAIS
11	BUENOS AIRES	1

CUIDAD	ID_PROVINCIA
SALTO	11
CARMEN DE ARECO	11
ROJAS	11
PERGAMINO	11



A este forma de organizar las dimensiones se lo denomina “copo de nieve” (snowflake) por la forma en que se ve el diagrama.

Salvo en casos excepcionales es totalmente desaconsejable normalizar las dimensiones.

Dado que en general el tamaño de las tablas de dimensiones es muy inferior al tamaño de la tabla de hechos, el ahorro en bytes que implica la normalización, no es en general lo suficientemente significativo como para atentar contra un aspecto clave del corazón del modelado dimensional, como es la simplicidad.

En pocas palabras, el ahorro de normalizar, no justifica quitarle simplicidad al modelo.

Técnica de modelado dimensional en 4 pasos

En el libro “The datawarehouse toolkit” de Ralph Kimball, ofrece una técnica de modelado que consiste en 4 pasos:

1. Seleccionar el proceso
2. Establecer el nivel de granularidad
3. Identificar las dimensiones
4. Identificar los hechos

1 - Seleccionar el proceso

El primer paso es seleccionar el proceso a modelar. ¿En qué actividades del negocio nos interesa focalizar? Los modelos de negocio son simplificaciones de la realidad que nos sirven para comprender qué está sucediendo. Si está bien definido nos permitirá responder preguntas claves de la organización. Es interesante la propuesta que realiza Xavier Mendoza utilizando

indicadores claves del negocio (KPI) para evaluar si se han alcanzado los objetivos. Están relacionados con los objetivos y con las actividades de la organización. Deben ser medibles, comparables y cuantificables. Por ejemplo, en un sistema de ventas podemos definir KPI sobre las unidades, el período de tiempo, el objetivo de ventas a conseguir.

Según el negocio, empresa, institución u organización, existen varios procesos relacionados.

Por ejemplo, en un comercio seguramente encontraremos como los procesos principales:

Ventas – compras - almacenamiento

Mientras que en un hospital podrían ser:

Internaciones – atenciones – guardias - prácticas

El o los procesos a modelar se definen de acuerdo al objetivo. Seguramente queramos modelar más de un proceso, cada uno con su tabla de hechos y dimensiones, y cada uno de ellos con sus propios objetivos e indicadores de gestión.

A veces suele confundirse los procesos con los sectores de una empresa, porque generalmente las organizaciones se dividen en áreas con propósitos afines. No hay que perder de vista que es mejor modelar el proceso y no el sector.

2 - Establecer el nivel de granularidad

Una vez identificado el proceso, tenemos que conocer qué información disponemos para analizarlo, definir el nivel de detalle que vamos a registrar en la tabla de hechos.

Por ejemplo podemos definir las ventas en por mes, por producto, o bien las ventas por día por vendedor.

Hay que tener en cuenta que si se define un grano más grueso (por ejemplo, ventas por mes), no podrá luego consultarse las ventas por día. Por eso se recomienda utilizar una granularidad más fina, de manera de poder responder a consultas con mayor nivel de detalle.

Suele ser en general beneficioso, registrar cada evento o transacción de manera individual. A estas tablas de hechos se las denomina “transaccionales” (transactional fact tables) y garantizan el nivel máximo de detalle.

Sin embargo, es posible que nos veamos limitados en recursos y nos interese solo conocer resúmenes mensuales. A estas tablas de hechos se las denomina “resumen” (snapshots fact tables).

3 - Identificar las dimensiones

Definida la granularidad, se enumeran las dimensiones asociadas. Las más comunes son, tiempos, lugar, responsable, sector, etc.

Es muy importante nombrar las dimensiones de manera que sean simples de identificar, como también así sus atributos. Cada dimensión deberá contener su propia clave primaria diferente de la clave natural de la dimensión. Normalmente una clave entera autogenerada es suficiente. Esto es altamente recomendado, ya que en general no tenemos control sobre los sistemas operacionales que proveen los datos.

También es necesario contar en cada dimensión con un registro que identifica en el “no valor” o “valor no especificado”. Cuando un registro no pueda identificarse con una dimensión, deberá utilizar como clave foránea el valor clave de este que normalmente se identifica con el entero “0”. No se deben colocar valores nulos como claves foráneas en las tablas de hechos. Los valores nulos si están permitidos en las medidas.

Una misma tabla física de dimensión, puede ser utilizada para representar varias dimensiones con distintos roles. Un ejemplo común de esto es la dimensión “tiempo”, que con una única tabla física puede utilizarse como dimensiones “fecha de venta”, “fecha de vencimiento”, etc.

4 - Identificar los hechos

Los hechos, como ya precisamos, son los valores numéricos asociados al proceso que queremos medir. Por ejemplo en un proceso de compras, serán la cantidad vendida y el valor unitario. En un proceso de atención de reclamos podría ser el tiempo en segundos que demoró en ser atendido, el tiempo en segundos que tomó la atención, etc.

Los hechos pueden ser totalmente aditivos, semi-aditivos o no aditivos, según tenga sentido sumarlos por todas las dimensiones, algunas dimensiones o ninguna dimensión.

Por ejemplo la cantidad vendida es totalmente aditiva, ya que se puede resumir por producto, por tiempo, por vendor, etc. En cambio, un saldo de cuenta corriente es semi-aditivo porque puede resumirse agrupando por distintas dimensiones por no por la dimensión tiempo ya que no tendría sentido.

Los valores que representan porcentajes son en general no aditivos.

Existen también hechos derivados que pueden surgir de un cálculo realizado sobre otros hechos, como valor de venta multiplicando el valor unitario por la cantidad vendida, o el porcentaje del monto vendido respecto al total ventas del dia.

Estos tipos de hechos, no se registran en la tabla física de hechos, sino que se definen en la tabla virtual.

Al igual que con las dimensiones, es muy importante nombrar los hechos de manera que se interpreten de manera correcta.

Hechos y Dimensiones Consolidados

Insistente reiteramos que uno de los aspectos más importantes de un modelo dimensional es su simplicidad y consistencia. Por eso es muy importante que los nombres de las medidas y dimensiones sean coherentes en todo el modelo.

En este sentido, si una medida se repite en más de una tabla de hechos debe llamarse por el mismo nombre si significa lo mismo, y debe llamarse por un nombre distinto si su significado es otro.

Este criterio también aplica a las dimensiones, es muy frecuente que una dimensión sea común a varios modelos. Dicha dimensión debe llamarse igual para todos ellos.

Consideraciones para tablas de hechos

Las tablas de hechos consisten en tablas físicas que contienen un campo para cada medida, claves foráneas a dimensiones y columnas que son en sí misma una dimensión (dimensiones degeneradas, no tienen una tabla física de dimensión propia).

Las tablas de hechos más comunes son las denominadas **transaccionales** (transactional fact table), dado que almacenan cada transacción y esto permite que ser bastante potentes, dado que pueden llegar a detallar la información al nivel de mayor precisión. Al ser tan detalladas son también muy densas y requieren mucho espacio de almacenamiento.

Las **tablas de hechos periódicas** (snapshot fact tables), son tablas que almacenan datos con un corte periódico, como por ejemplo mensualmente. Necesitan menos espacio que una tabla de hecho transaccional, pero no es posible luego llegar a mayor nivel de detalle de información.

Tanto las tablas de hechos transaccionales como las periódicas, nunca se actualizan, sino que siempre se insertan nuevos registros.

Para casos donde los hechos se deben actualizar, como por ejemplo en los procesos que contienen workflows, se utilizan **tablas de hechos periódicas acumulativas** (Accumulating Snapshot Fact tables). Tales tablas de hechos, tienen medidas que pueden estar en null al comienzo y pueden actualizarse posteriormente. Generalmente se necesita una dimensión de estado para guardar el valor actual de cada ítem. En estas tablas cada registro se actualizará frecuentemente cuando se tenga nueva información disponible, para reflejar el estado y las métricas acumuladas. Esta es la diferencia fundamental con los otros tipos de tablas. A veces las fotos acumulativas se usan en conjunto con las fotos periódicas. Se puede pensar en un DW/BI que guarde fotos periódicas por mes y que lleve una fotografía acumulativa para la información del mes en curso. En estos casos cuando finaliza el último día del mes la fotografía acumulada pasa a ser una más de las fotos periódicas y se crea una nueva foto acumulativa para registrar la actividad del nuevo mes a partir del día siguiente. La fotografía mensual se construye de forma incremental, agregando el efecto de las transacciones diarias a una fotogra-

fía acumulativa. Un ejemplo para las fotografías acumulativas podría ser la línea de producción del vino:

- * Cuando se cosecha
- * Cuando entra en la triturada para separar el mosto
- * Cuando entra en los barriles de aluminio
- * Cuando pasa a los toneles de madera
- * Cuando pasa a las botellas
- * Cuando llega al usuario final.

Aunque la mayoría de las tablas de hechos contienen medidas, es posible que en ciertos procesos sólo requieran almacenar la ocurrencia de un evento, sin medidas adicionales. En estos casos cada registro tiene solo la combinación de claves foráneas de las dimensiones y/o campos de dimensiones degeneradas. Estos tipos de tablas se denominan **tablas de hechos sin hechos** (Factless Fact tables), y utilizan el count(*) para medir únicamente la cantidad de ocurrencias. Se registra un evento por línea. Por ejemplo el registro de asistencias y las inscripciones a cursos de alumnos universitarios. Se registran las inscripciones a materias entonces la granularidad de la tabla será de una fila por cada inscripción de un alumno a una materia en un período académico. El período será el nivel más bajo disponible para la registración de eventos, debe estar de acuerdo con la dimensión de fechas. Este tipo de tablas, compuestas sólo por claves de dimensiones es también una tabla de hechos. Permitirá responder a preguntas del tipo ¿qué cursos tienen mayor cantidad de inscripciones?, o cantidad de inscripciones promedio por alumno y período, entre otras. Este tipo de tabla de hechos permite registrar los eventos de cobertura, por ejemplo para analizar la venta de productos en promoción. Sería deseable saber qué productos estaban en promoción y no se vendieron. Incluir información necesaria para este análisis en la tabla de ventas significaría la incorporación de muchísimos registros con ceros representando los eventos que no sucedieron (todos los productos que no se vendieron), lo que haría crecer enormemente el tamaño de la tabla en forma rala (poco densa considerando las combinaciones de los valores de las dimensiones que contiene respecto a todas las posibles). Se crea entonces una tabla donde se registran todas las promociones. La respuesta a los productos promocionados y no vendidos se resuelve en dos pasos, primero se consulta cuales estaban promocionados, y con ese resultado se busca en la tabla de ventas cuales no están durante la vigencia de la promoción. También se puede consultar sobre el uso de las aulas o instalaciones de una institución educativa.

En ciertas circunstancias, cuando las tablas de hechos son muy grandes, y la necesidad de tiempo de respuesta es muy importante, se suelen utilizar tablas de hechos de apoyo, que contienen sumas precalculadas. Esta tipo de tablas se denominan **tablas de hechos agregadas** (Aggregated Fact Tables), y pueden existir más de una asociada a una tabla de hecho, donde cada una agrupa por distintas claves.

Generalmente se escogen las combinaciones de claves más frecuentemente utilizadas para generar estas tablas. La ventaja en velocidad de respuesta que se gana al generar una tabla de hechos agregada, debe ameritar el costo de mantenimiento que ello implica.

Consideraciones para tablas de dimensiones

Las dimensiones son tablas que en general tienen una estructura muy similar. Tienen una clave primaria que se utiliza también en las tablas de hechos como clave foránea para asociar la descripción al hecho, y muchos atributos textuales, desnormalizados. Estos atributos son los que se utilizan para filtrar y agrupar los registros en las consultas que realizan los usuarios finales.

Naturalmente, como diseñadores de bases de datos, tendemos a normalizar los datos de manera que no sean redundantes. Cuando una dimensión se jerarquiza en más de una tabla, (como por ejemplo las tablas “tipo de producto”, “subtipo de producto”, “producto”) se denomina dimensión “copo de nieve” (snowflaked).

Hay que evitar esta forma de representar las dimensiones salvo que sea una elección justificada.

Clave primaria de dimensión

Las dimensiones deben tener una clave primaria sustituta (surrogated key) diferente de la clave primaria que tienen en los sistemas operacionales.

Esto es necesario porque el modelo dimensional debe tener control sobre estas claves, y los sistemas operacionales pueden no administrar adecuadamente estas claves, o incluso, ciertas dimensiones pueden ser referidas con distintas claves en más de un sistema operacional.

Además, como veremos más adelante, las dimensiones pueden tener más de una versión de cada registro.

La clave primaria debería ser un número entero secuencial, comenzando en 1, el 0 se reserva para el registro que identifica el “valor no especificado”.

Dimensiones con valores variables

Existen casos donde las dimensiones se definen en el proceso inicial del modelo conceptual y ya en el DW/BI no cambiar su valor. Sin embargo, puede ocurrir que el identificador de la dimensión no cambia pero sí su descripción. Esto puede ocurrir por cambios en el sistema fuente o corrección de errores, a estos casos se los conoce como “slowly changing dimension” y hay tres opciones para manejarlos:

- sobrescribir el registro de la dimensión con el valor nuevo. Esta opción se usa para correcciones de errores, y pierde la historia del elemento.
- crear un nuevo registro en la dimensión para el nuevo valor. Es la solución recomendada para manejar los cambios en los atributos, sin embargo requiere del uso

de claves sustitutas para el DW dado que no puede tomarse la misma clave que existe en el sistema operacional.

- crear un campo “descripcion_anterior” en el registro de dimensión para guardar el valor previo inmediato del atributo. Esto es útil para cambios no muy importantes, que no determinan una partición en la serie histórica. Aunque el valor cambió se puede hacer de cuenta que no se modificó.

También existe la posibilidad de no actualizar nunca el valor, el DW no reflejará los cambios que se realicen en los sistemas fuentes.

Si los cambios son bastante frecuentes las técnicas a utilizar son las mismas, sólo que es probable que en estos casos sea más importante guardar la historia de los valores.

Dimensiones Degeneradas

Existen casos especiales en los cuales se necesita guardar junto con el hecho, la clave del sistema operacional que lo generó, como el número de orden. Este tipo de datos se representa con una columna más en la tabla de hechos, sin ninguna tabla con información extra asociada. La dimensión consiste sólo del código y por eso se dice que está degenerada (no tendrá tabla asociada). Suele utilizarse en dimensiones “grandes” o que incluyen gran cantidad de elementos, por ejemplo alumnos o personas. En muchos casos pueden soportarse usando estructuras ROLAP y HOLAP. Es necesario aplicar técnicas para mantenerlas controladas, sobre todo si cambian con frecuencia.

Métricas

Una vez diseñado el data mart, definida la granularidad de las tablas de hechos y diseñado las dimensiones, el último paso en la definición de cada modelo dimensional consiste en la definición de las medidas. Las medidas típicas son numéricas o aditivas. Las aditivas son aquellas que tiene sentido sumar sus valores a través de sus posibles dimensiones, como unidades o importes. Los datos en el nivel de granularidad más bajo suele ser aditivo. Las no aditivas son los promedios, por ejemplo. A continuación se presentan los diferentes tipos de métricas que existen y algunas técnicas para manejarlas.

Medidas de intensidad

Las medidas de intensidad, también numéricas, como balances de cuentas y niveles de inventario, no son perfectamente aditivas. En estos casos generalmente es posible sumar los valores a través de todas las dimensiones menos en el tiempo. Medidas de este tipo se pueden combinar a través del tiempo calculando lo que denomina “time-average” o promedio en el tiempo. Esta fórmula de cálculo está basada en el “promedio de los hijos”. En caso que existan tres niveles: año, mes y día, el valor en el mes será la suma de los valores en cada día del mes dividido la cantidad de días del mes, y el valor en el año será la suma de los valores en los meses dividido doce (o la cantidad de meses correspondiente al año en curso).

Cuando las medidas son no aditivas, como un valor promedio, o una temperatura, habrá que ver si puede definirse algún método de agregación que sea válido para todas las dimensiones. Otra posibilidad es limitar el alcance de las consultas al nivel de granularidad en que está definido el dato. La mayoría de estos casos pueden surgir de no haber elegido bien la granularidad de la tabla de hechos. Los datos en el nivel de granularidad más bajo suelen ser aditivos.

Medidas con múltiples unidades

Diferentes medidas de un negocio pueden ser la misma expresada en diferentes unidades de medición. Por ejemplo expresar distancia en metros, kilómetros, millas, leguas, o importe de una venta en pesos, en miles de pesos, en dólares, etc.

En los negocios que se requiere analizar una cadena de producción, que incluyen varios procesos en etapas y en los cuales se monitorea el flujo de los productos a través del sistema suelen presentarse casos un poco más complejos de conflictos con las medidas de cantidades. Esto es porque en los diferentes puntos de la cadena existen múltiples medidas, debido a que los números son expresados en diferentes unidades de medición. Por ejemplo, en los distintos sectores pueden contarse: unidades de envío (paquetes y cajas que se trasladan del depósito a la sucursal), unidades vendidas, unidades escaneadas (paquetes de venta), unidades consumibles (contenido individual de los paquetes), etc.

De forma similar la cantidad de un producto puede tener varias valuaciones económicas posibles, en términos de valor de inventario, precio de lista, precio de venta original, precio de venta final.

Esta situación puede ser complejizarse aún más si además se tienen muchas métricas diferentes. En estos casos no es correcto incluir en la tabla de hechos sólo a las medidas fundamentales, porque se perdería parte del análisis. Tampoco es correcto incluir a todas las posibles combinaciones de medidas fundamentales expresadas en las diferentes unidades de medida y por cada factor de valuación posible (porque la tabla sería gigante). La solución consiste en incorporar a la tabla de hechos las medidas fundamentales, los factores de conversión y los

factores de valuación identificados previamente. De esta forma es posible obtener cualquier medida final que se desee a partir de la combinación correspondiente.

A veces se cae en la tentación de incluir los factores de conversión en alguna tabla de dimensión (por ejemplo: contenido individual, cantidad de unidades por paquete de venta, y cantidad de paquetes por caja de envíos, podrían incluirse en la tabla de productos). Esto no es recomendable porque los factores de conversión suelen variar, lo que obligaría a generar nuevos registros en la tabla de dimensión para mantener los cambios. Esto, especialmente los asociados a costos y precios, suelen evolucionar en el tiempo y se asemejan mucho más a medidas que a atributos de dimensión.

Se recomienda incluir todas las medidas y todos los factores de conversión juntos en el mismo registro de la tabla de hechos para garantizar que esos factores serán usados correctamente.

Medidas básicas y derivadas

Las medidas básicas, que existen naturalmente a partir de un evento, corresponde a una columna de una fuente de datos, por ejemplo cantidades e importes. Las que se calculan en función de una o más de ellas, se denominan medidas derivadas, por ejemplo el importe de ventas acumulado a la fecha, la rentabilidad, la diferencia entre la cantidad de ingresantes del año actual con respecto al mismo valor el año anterior, la tasa de egresados sobre ingresantes, etc.

Hay dos tipos de medidas derivadas:

- Las aditivas: pueden ser calculadas a partir de otras medidas existentes en el mismo registro de la tabla de hechos. Por ejemplo el caso de múltiples unidades de una medida presentado anteriormente, el cálculo de ganancia (como diferencia entre ventas y costos), etc. Estos casos suelen resolverse mediante vistas que se presentarán a los usuarios.
- Los cálculos no aditivos: como porcentajes (ratios) o medidas acumulativas expresadas en tablas agregadas. Estos casos muchas veces requieren ser calculados al momento que se consultan por la herramienta de consultas o por quien escribe el reporte. A veces es preferible incluir algunas de estas medidas en las tablas de hechos agregadas para simplificar los cálculos y los tiempos de respuestas.

Las herramientas del mercado suelen proveer mecanismos para la definición de medidas derivadas en función de las básicas. Incluyen fórmulas de cálculo complejas y muchas veces evitan tener que realizar cálculos en el DW. Estas fórmulas pueden utilizarse en la etapa de diseño, quedando así las medidas derivadas incorporadas al Data Mart o cubo. También pueden usarse en la interfaz de consulta, permitiendo a los usuarios avanzados definir sus propias medidas y compartirlas.

Un error frecuente es finalizar el diseño físico de la base de datos ignorando las medidas derivadas y recién tenerlas en cuenta cuando comienza el desarrollo de las aplicaciones de usuarios. Generalmente serán necesarios muchos cálculos del negocio que habrá que concen-

suar y es preferible anticiparse a estas tareas, sobre todo para asegurarse que todas las medidas básicas necesarias son tenidas en cuenta. La medida derivada es aditiva también por las dimensiones del modelo. Esto no significa que sea calculada como una suma de medidas básicas, sino que la medida resultante puede sumarse.

Normalización de la tabla de hechos

Hay otro tipo de normalización que no se explicó en la sección anterior porque no refiere a las dimensiones sino a las tablas de hechos y en particular a las medidas. Cuando se tienen diferentes medidas asociadas a un evento, por ejemplo monto bruto, descuento, etc., lo más común es modelar estas medidas como columnas separadas dentro de la tabla de hechos. También es válido usar una sola columna denominada “monto” junto con una dimensión adicional que represente el “tipo de monto” (en este caso los valores: bruto, descuento, etc.). Esta solución es útil en algunos casos, particularmente cuando se tienen muchas medidas sin valor en muchos de los registros. Caso contrario implica multiplicar la cantidad de registros de la tabla de hechos, aumentando el tamaño de esta. Además, tener las medidas como columnas de un mismo registro es en general mucho más útil para operar con ellas y realizar cálculos (por ejemplo porcentaje del descuento sobre el monto bruto). Estas operaciones se dificultan si los valores están en registros separados.

Caso de Estudio

Caso del Municipio

Un municipio es una entidad administrativa que puede agrupar una o varias localidades de una misma región.

El municipio está compuesto por un territorio claramente definido por un término municipal de límites fijados y la población que lo habita está regulada jurídicamente.

Los objetivos más importantes que tiene el municipio para el año corriente son:

- Detección de focos de evasión, para luego implementar políticas que permitan mejorar la recaudación.
- Optimizar el servicio de cobranza a partir de la información con que se cuenta sobre las formas y lugares de pago.
- Diseñar planeamiento urbano, teniendo en cuenta las siguientes necesidades y realidades:
 - de edificaciones futuras, a partir de un análisis de la realidad actual de los barrios y sus tipos de construcciones.
 - Construcción de hospitales o centros de salud.
 - Construcción de escuelas.

Podemos definir dos procesos a evaluar con el problema planteado:

Proceso FOCO DE EVASIÓN:

- ¿Qué tipo de tasa inmobiliaria (1 Alumbrado, 2 Publicidad y 3 Seguridad) es la que más adeuda en el año 2013?

El tipo de tasa es un tabla dimensión y “la que más adeuda” es una medida, que va a ser el atributo **cantidad** en la tabla de hechos.

- ¿Cuál es el barrio en donde más se adeuda en el año 2013?

El barrio es una tabla dimensión y “el que más adeuda” es una medida, que va a ser el atributo **cantidad** en la tabla de hechos.

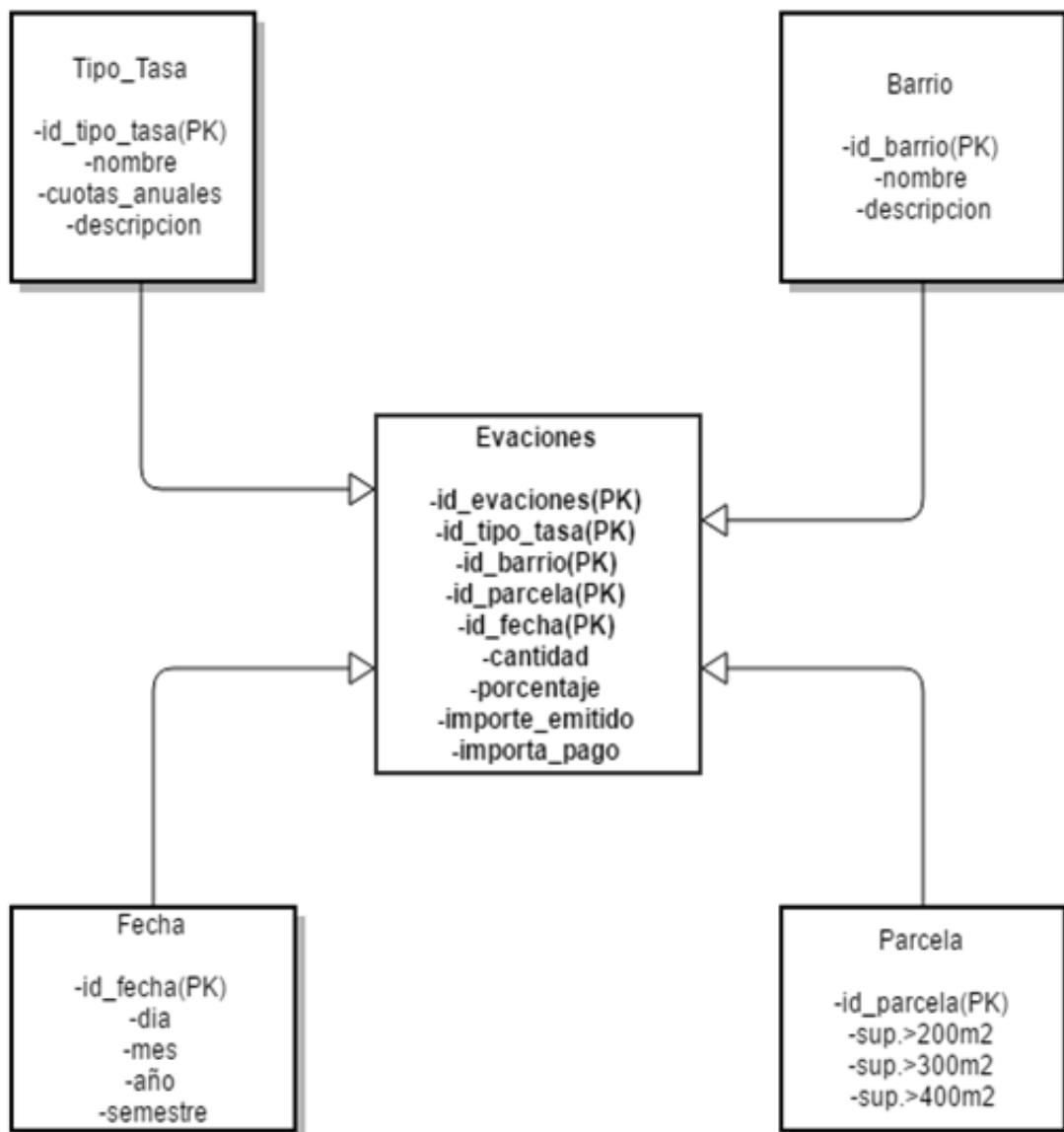
- ¿Cuál es el **porcentaje** de vecinos con pileta de natación que adeudan **tasa inmobiliaria** en el año 2013?

El **porcentaje** es una medida, va a ser el atributo porcentaje en la tabla de hechos.

- ¿Cuál es la **cantidad** de inmuebles que tienen una **parcela superior a 200m²** y adeudan **tasa inmobiliaria** en el año 2013?

Vamos a tener un tabla de dimensión tipo parcela que tendrá parcelas mayores a 200, mayores a 300, mayores a 400. Por otro lado la medida de esta pregunta es la **cantidad**, atributo de la tabla de hechos.

También tendremos la tabla dimensión fecha y en la tabla de hecho tendremos dos medidas extra que son **importe emitido** e **importe pago**.



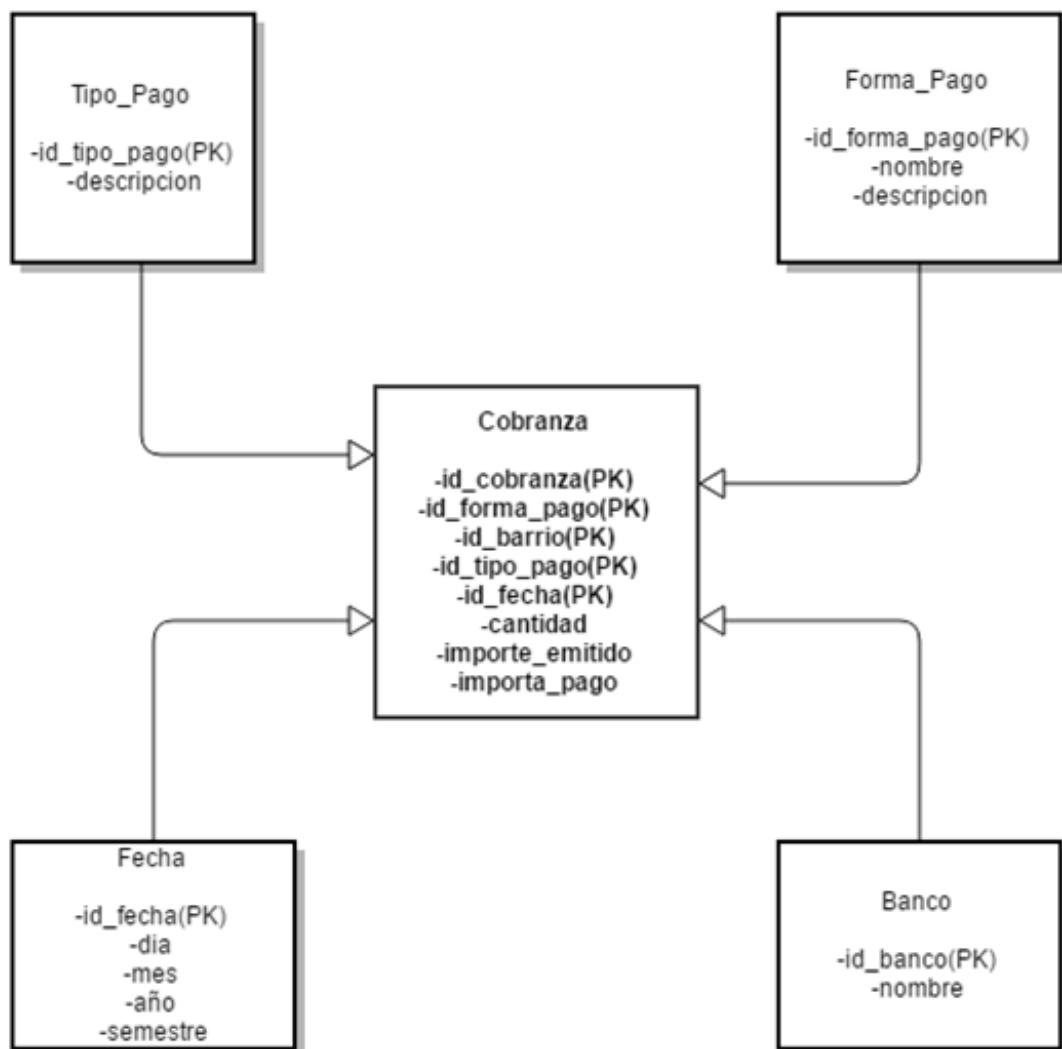
Proceso SERVICIOS DE COBRANZA

¿Cuál es la forma de pago con la que más pagos se efectuaron en el 2013?

Forma de pago es una tabla dimensión y en donde más pagos se efectuaron es una cantidad, que va a ser atributo de la tabla de hecho.

¿Qué tipo de pago se utiliza mayormente en cada barrio?

¿Cuál es el banco que tiene más pagos recibidos?



CAPÍTULO 3

Arquitectura del Data Warehouse

Ralph Kimball propone el Data Warehouse como un conjunto de data marts (subconjunto de la información del DW), que implementan el diseño multidimensional de cada departamento de una organización, que puede incluir cierta redundancia de información con distintos fines. Además, pueden construirse en forma modular e independiente siguiendo ciertas consideraciones, constituyendo finalmente el DW. Por su parte Bill Inmon, considerado el padre del Data Warehouse presenta al DW como una componente más de todo el proceso de inteligencia de negocio, y los data marts tienen como fuente de datos a la estructura del DW. El DW se encuentra estructurado en tablas en 3º forma normal. A partir de esta discusión, Ralph Kimball que toma ideas del anterior y propone un esquema mixto. También se puede dar una arquitectura de análisis independientes, que no son recomendadas dado que proveen una visión parcial y sectorizada de cada parte de la organización. En los últimos año surgió una nueva propuesta denominada data vault, que se incluye también a continuación.

El enfoque de Inmon, considerado el padre del concepto, propone el data warehouse como un conjunto de datos organizados por tema, variantes en el tiempo, integrados y no volátiles. El objetivo es dar soporte a la toma de decisiones. Kimball, considerado el promotor del enfoque dimensional para la construcción de DW, el DW es una copia de los datos transaccionales estructurados para la consulta y el análisis de los mismos. Es importante destacar que las propuestas en algún momento confluyen y más allá de cuales es mejor o cual es la que mejor se ajusta, lo importante es esforzarse para que la solución sea ampliamente aceptada por los usuarios finales a fin de tomar decisiones basadas en datos.

Ralph Kimball propone una cuatro componentes para considerar en un entorno de DW/BI: sistemas fuentes operacionales, sistema ETL (Extract, Transform & Load), presentación de datos y aplicaciones de Business Intelligence, como se puede observar en la figura 3.1.

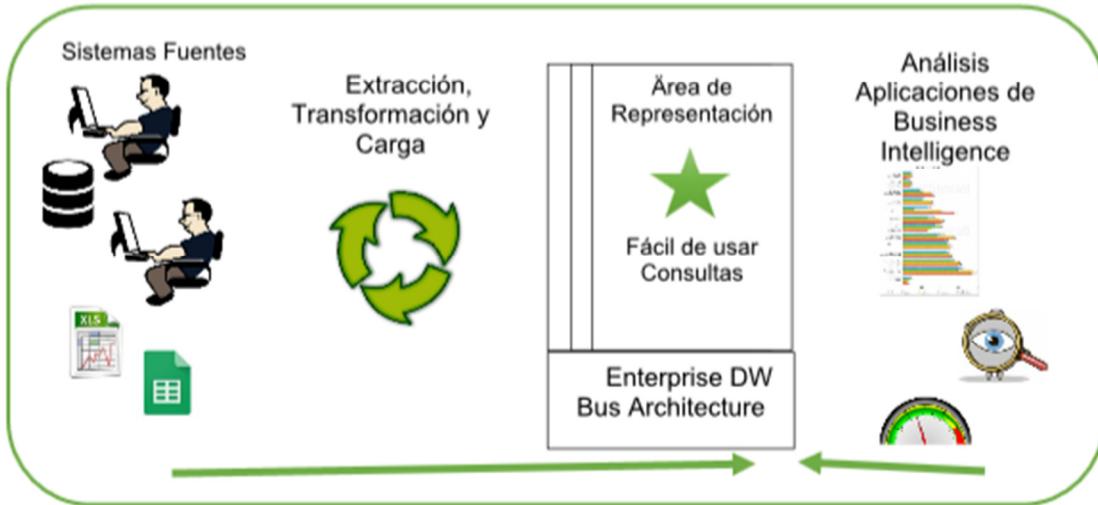


Fig. 3.1 – Arquitectura DW/BI propuesta por Kimball

- **Sistemas Fuentes Operativos:** diseñados para llevar el registro de las transacciones y la operatoria diaria de la organización. Estas fuentes de datos son externas y posiblemente el diseñador de una solución de BI no tenga control sobre el contenido y el formato de los datos que gestionan. La prioridad es alto rendimiento y disponibilidad, con consultas puntuales, de a un registro por vez y con mayoría de operaciones de inserciones, bajas y actualizaciones. Mantienen pocos datos históricos, esto se lleva a través del DW. También son específicos para cada área o negocio de la organización, y pueden ser integrados a través de aplicaciones como ERP.
- **ETL (Extract, Transformation and Load):** transforma los datos fuentes a un objetivo. En este espacio de completan las dimensiones y la normalización es opcional. Los usuarios no generan consultas. El objetivo es asegurar el rendimiento, integridad y consistencia. La etapa de ETL incluye un área de trabajo, las estructuras de datos instanciadas y un conjunto de procesos. Es el intermediario entre las fuentes de datos y el área de presentación de BI. La extracción es el primer paso del proceso de cargar los datos en el entorno del DW para luego manipularlos a través de diferentes técnicas y metodologías. Por ejemplo la limpieza de los datos (corregir datos erróneos, resolver conflictos de distintos dominios, completar elementos faltantes), combinar datos de distintas fuentes y datos duplicados. Se agrega valor a los datos identificando y construyendo metadatos, un diccionario de datos e identificando las entradas de datos con problemas para plantear una reingeniería a fin de garantizar la calidad en forma constante. La etapa final del proceso de ETL es la estructuración física y la carga de los datos según el modelo dimensional definido en el área de representación. Esta etapa es fundamental porque es donde se cargan las tablas de dimensiones y hechos. El tema de la carga de los datos en 3º forma normal o no es un tema de debate, como se trabajó en el capítulo 2. En general, el sistema de ETL está formado por consultas sencillas, de ordenamiento y consultas secuenciales, puede no estar basado en una estructura re-

lacional, pueden ser incluso archivos planos. Hay casos en los que los desarrolladores encuentran útil la normalización para llevar adelante las tareas de limpieza e integración de datos. Estos datos luego son desnormalizados para ingresarlos al sistema dimensional, con lo cual la carga se realiza dos veces con la consecuente sobrecarga de tiempo, puede ser que de espacios también en cada actualización y carga periódica. Lo importante es no perder el foco al construir una solución de BI. En palabras de Kimball, *es aceptable crear una base de datos normalizada que brinde soporte al proceso de ETL, sin embargo, esta no es la meta final. Las estructuras normalizadas deben ser prohibidas a las consultas de los usuarios porque ellas hacen fracasar las metas de comprensión y rendimiento.*

- **Área de Almacenamiento y Representación para soporte de la Inteligencia de Negocio:** este espacio es donde los datos son organizados, almacenados y se ponen a disposición para las consultas de los usuarios, reportes y demás aplicaciones de BI. Los datos almacenados deben ser atómicos para resistir las consultas impredecibles del usuario final. Pueden existir tablas con datos sumarizados para brindar mayor performance, pero los datos atómicos no pueden estar lockeados para realizar consultas lineales de los usuarios finales, cuyas preguntas cambian y en la mayoría de los casos son impredecibles. Puede existir una ODS (Operational Data Store) en caso de ser necesario. El área de representación se estructura en función de los eventos medibles del proceso de negocio, más allá de las visiones de los departamentos o los estamentos de la organización y todas las dimensiones deben crearse como *dimensiones conformadas*. Este es el principio de la arquitectura de bus o *Enterprise Data Warehouse Bus Architecture*.

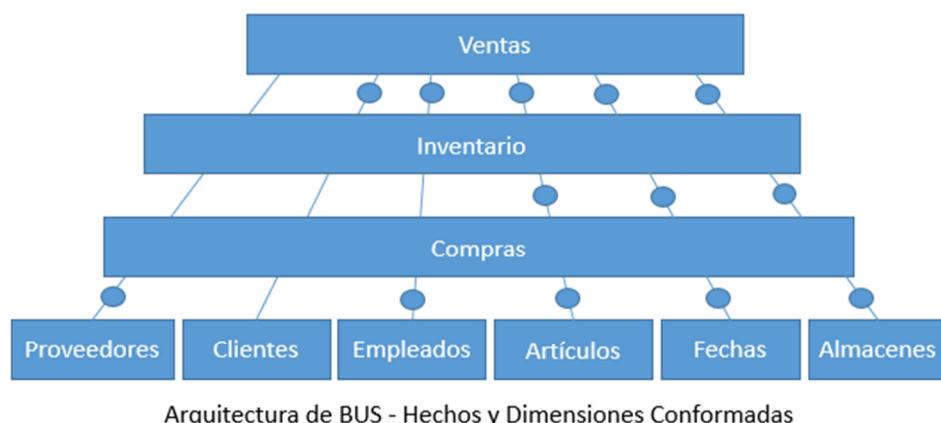


Fig. 3.2 – Enterprise Data Warehouse Bus Architecture.

No tener en cuenta este principio puede llevar a distintas visiones de la misma entidad dentro de la organización. Por ejemplo los clientes, las sedes u sucursales de la organización, los empleados, los departamentos, constituyen entidades comunes para realizar un análisis focalizando desde cualquier perspectiva. *El área de representación es una*

gran solución de DW/BI para toda la organización, formada por docenas de modelos dimensionales con muchas dimensiones asociadas a través de tablas de hechos.

- **Aplicaciones de Business Intelligence:** en esta componente los usuarios utilizan los datos de la etapa anterior para trabajar en la toma de decisiones basadas en hechos. Las consultas obviamente, es la actividad más importante de esta etapa. La aplicación puede ser una consulta ad-hoc como minería de datos o análisis de datos, de todas maneras recordar que existen distintos tipos de usuarios y es necesario que nuestro sistema de BI de respuesta a todos: a aquellos que quieren realizar las consultas a través de interfaces específicas como aquellos que requieren de un tablero de control con los indicadores relevantes acordados en la etapa de requisitos.

Arquitecturas alternativas de DW / BI

Actualmente existen tres alternativas dominantes a la arquitectura propuesta por Kimball para la construcción de data warehouse, cada una de ellas con sus similitudes y diferencias:

1. Arquitectura de Data Marts independientes: no es lo más recomendable. Los data marts se desarrollan en cada departamento sin contar con una visión integral que permita el análisis en su conjunto a futuro. Los requerimientos se ajustan a la visión de un departamento particular, de acuerdo a sus reglas de negocio y nombres específicos, generando islas independientes. Es habitual que otros departamentos estén interesados en analizar los mismos datos y métricas, resultados de los procesos principales de la organización. Si estos departamentos, cada uno genera sus propios reportes de acuerdo a sus sistemas de gestión, sus métricas y nombres es altamente probable que manejen distinta información. Por ejemplo, en una organización educativa como la universidad, se cuenta con un departamento de Personal que lleva adelante la gestión del personal docente, administrativo y mantenimiento y autoridades de la organización. Asimismo, el departamento de Enseñanza se ocupa de gestionar las materias, los estudiantes que la cursan y los docentes a cargo de las mismas. Si ambas desean analizar a los docentes, por ejemplo dedicación, rama de conocimiento, antigüedad, género, cargo es necesario que ambos departamentos trabajen con los mismos parámetros para identificar a los docentes, la escala de dedicación y cargos, el género a fin de obtener por ejemplo la relación docente alumno desde la perspectiva de antigüedad, género, área, etc.

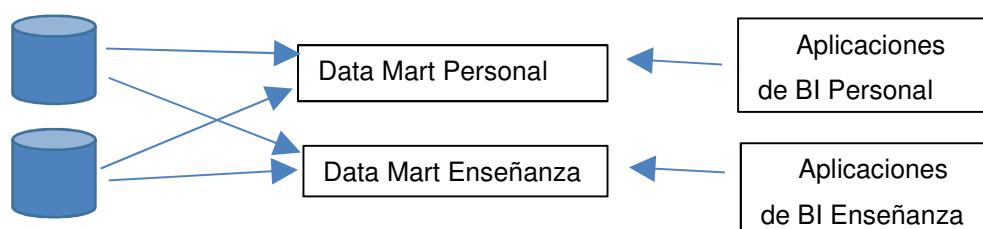


Fig. 3.3 – Data marts independientes por departamentos. Silos de información

2. Corporate Information Factory de Inmon

The Corporate Information Factory (CIF) Bill Inmon, integra extracciones coordinadas de los sistemas fuente con las que alimenta una base de datos relacional de información atómica en 3° forma normal, a través de procesos de ETLs. Este Data Warehouse normalizado es usado para conglomerar repositorios adicionales para presentación de la información, lo que incluye almacenes de propósitos especiales para la exploración y minería de datos, y también Data Marts. Este modelo se lo denomina también hub-and-spoke por la estructura de los rayos de una rueda de bicicleta con un centro. En forma similar a Kimball, esta arquitectura promueve la integración y coordinación de todos los datos de la organización. La diferencia es que Kimball propone una estructura de bus con dimensiones conformadas en lugar de una estructura normalizada en 3° forma normal. El proceso de normalizar no implica una integración coherente de los datos de distintas fuentes, la normalización crea tablas físicas que implementa la relación de muchos a uno mientras que en una integración entre distintas fuentes surgen inconsistencias que es necesario analizar y resolver. Suele suceder que los usuarios acceden al EDW debido a que requieren mayor nivel de detalle o algún dato específico en un momento dado.

El acceso a los datos se realiza a través de los data marts o la ODS (Operational Data Store) en caso de ser necesario.

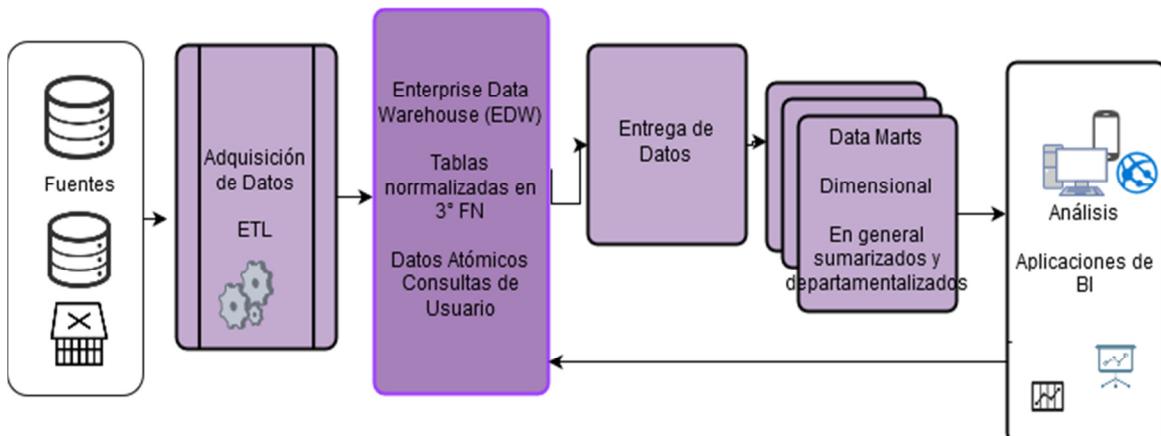


Fig. 3.4 – Corporate Information Factory de Bill Inmon

Inmon liberó la arquitectura Enterprise Data Warehouse 2.0 en el año 2008, a partir de la experiencia de los últimos 20 años. Lo que hace es catalogar la información por edad de la misma y se clasifica por su uso. Incluye información estructurada y no estructurada y pone el objetivo en responder a los objetivos del negocio. Se la suele combinar con Data Vault para lograr mayor flexibilidad a la hora de realizar cambios en el entorno del DW. A continuación se detallan las características principales.

3. Data Vault:

Su objetivo es otorgar adaptabilidad, flexibilidad y consistencia en el Data warehouse, que serán indispensables partir del crecimiento del negocio. Es un híbrido que combina lo mejor de la 3° forma normal y el esquema estrella.

Para convertir un esquema OLTP en 3° forma normal a una arquitectura Data Vault Enterprise Data Warehouse, es necesario dar respuesta a los siguientes interrogantes: ¿se respetó el estándar de nombres? Esto facilitará luego la toma de decisiones en el esquema DV; ¿cuántas tablas son independientes del modelo de datos? estas tablas regularmente provocan serios problemas al momento convertir al modelo DV; ¿existen claves primarias y claves foráneas definidas? en DV se eliminarán estas relaciones; ¿el modelo utiliza claves naturales o claves subrrogadas? en DV se utilizarán únicamente claves naturales; ¿el modelo permite responder a los requerimientos funcionales del DW? si no lo hiciera, tendremos algunas dificultades adicionales al momento de la creación del modelo en DV; ¿puede la información separarse por clase o tipo de dato? ¿con qué frecuencia se generan cambios en los atributos?

A partir de las respuestas a estos interrogantes, es posible comenzar a construir el modelo del negocio, que está formado por Links, Hubs y Satellites.

Los **Hubs** identifican las claves de negocio y las claves subrrogadas. Los **Links** identifican las relaciones entre las tablas creadas y la información descriptiva del modelo se denomina **satélite**. En el siguiente ejemplo de registro de productos de una organización, la siguiente tabla es un ejemplo de Hubs. Como se puede observar, no contiene FK para mantener la integridad del negocio.

ID	Product #	Load DTS	RCRD SRC
1	MFG-PRD123456	6-1-2015	MANUFACT
2	P1235	6-2-2015	CONVENIOS
3	*P1235	15-2-2016	CONVENIOS
4	MFG-1235	17-6-2016	MANUFACT
5	1235-MFG	14-7-2016	FINANZAS
6	1235	23-10-2016	FINANZAS
7	PRD128582	12-4-2017	MANUFACT
8	PRD128586	12-4-2017	MANUFACT
9	PRD128256	23-4-2017	MANUFACT
10	PRD929929-*	23-4-2017	MANUFACT

Tabla 3.1 – Data Vault – Hubs

Los links por su parte, son quienes relacionan uno o más hubs o business keys. Su objetivo es registrar, capturar el pasado, presente y futuro de la organización. En general representan transacciones, asociaciones, jerarquías y redefiniciones de términos del negocio. Brindan la flexibilidad a la metodología dado que pueden cambiar en el tiempo, incluso puede ser unidireccional en un momento, bidireccional en otro. Esta posibilidad de cambiar las relaciones nos permite cambiar la estructura.

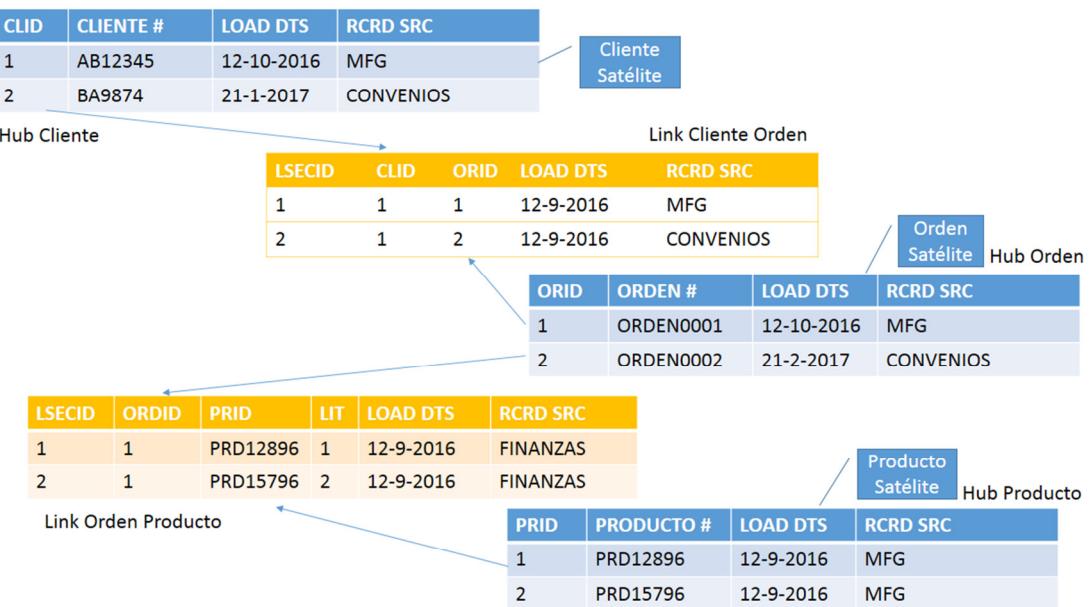


Fig. 3.5. Ejemplo de diseño aplicando data vault

Los satélites mantienen el registro de los cambios de los campos de las tablas que pueden cambiar, de los hubs y los links. Son los responsables de recolectar datos para brindar contexto e historia. Sólo se actualiza si hay un cambio en algún registro. Su clave primaria está formada por la clave primaria del hub con la fecha de carga incorporada. Continuando con el ejemplo anterior, el hub Cliente contiene la siguiente información:

CLID	CLIENTE #	LOAD DTS	RCRD SRC
1	AB12345	12-10-2016	MFG
2	BA9874	21-1-2017	CONVENIOS

Fig. 3.6. Ejemplo de hub

El satélite *nombre del cliente* llevará el siguiente registro:

CSID	LOAD DTS	NAME	RCRD SRC
1	12-10-2016	AB SOLUCIONES	MFG
1	15-9-2016	AB SOLUCIONES SA	MFG
2	12-1-2017	BA CREACIONES	CONVENIOS
2	8-6-2016	BA CREACIONES Y ASOC	MFG

Kimball, en su libro propone finalmente un esquema compartido, donde toma lo mejor de la metodología de Inmon y concluyen en la siguiente arquitectura:

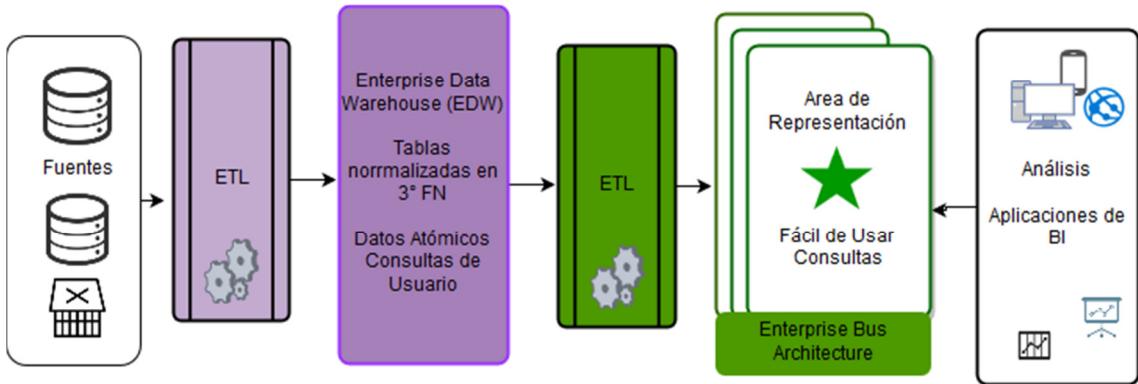


Fig. 3.7 - Arquitectura híbrida CIF - Kimball

Esta arquitectura puebla un EDW centrado en CIF, que está completamente fuera del alcance de los usuarios de BI para análisis y reportes. El dato es dimensional, atómico (complementado con agregaciones), centrado en procesos y conforme a la arquitectura de bus. En este modelo preexiste un repositorio integrado en 3º forma normal, con sus atributos de alto rendimiento y usabilidad, permitiendo las consultas de los usuarios sólo a través del área de representación, la cual está diseñada de acuerdo a los principios de Kimball.

Para las organizaciones donde existe un repositorio integrado 3ºFN EDW que aún no cumple con los requisitos de flexibilidad para las consultas y análisis que puede dar el enfoque de Kimball esta arquitectura puede resultar óptima. Sin embargo, cuando se comienza desde cero, esta aproximación puede ser más costosa en las fases de desarrollo y producción dado que conlleva muchos movimientos de datos y almacenamiento redundante de detalles atómicos. Sin embargo, si se cuenta con presupuesto y paciencia por parte de la organización como para normalizar los datos antes de poblar el modelo dimensional diseñado de acuerdo a los principios de Kimball, es una opción viable también.

Modelo Físico

El modelo físico es la siguiente etapa del proyecto del modelo dimensional. Una data warehouse está formado por un conjunto de tablas. Según el ciclo de vida del desarrollo de un proyecto de BI, en el modelado multidimensional, el diseño lógico y físico se asemeja mucho.

El objetivo es definir entonces, para cada tabla, su clave y su atributo. Se establecen convenciones de nombres y seteos específicos del ambiente de la BD. Es importante tener en cuenta las siguientes consideraciones:

- Se recomienda que las tablas sean identificadores independientes de la clave primaria de la fuente de datos.
- Si bien las métricas pueden ser aditivas (números), semiaditivas (con particularidades al momento de sumar las cantidades) o no aditivas (descriptivas, atributos cualitativos) en la tabla de hechos implementarlas como aditivas.

- Incluir metadatos como fecha y hora de la carga del dato, fecha de modificación, autor, y fuente de origen.

También se incluye en esta etapa la definición de agregaciones que permiten mejorar el tiempo de respuesta. Se estimación de volúmenes, planes y estrategias de indexación, configuración de la base de datos, como tamaño de bloque, particionamiento, monitoreo, entre otros.

Diseño del ETL

Como pudimos observar en la sección de Arquitectura, todas ellas incluyen una sesión denominada ETL, por sus siglas en inglés de Extract, Transform & Load. Esta etapa es fundamental y suele insumir el 80% del tiempo para el desarrollo de un data warehouse exitoso. Además, DW es sólo una de las tipologías de proyectos que requieren de procesos de ETL, tales como BI, procesos de calidad de datos, integración de datos de clientes, de proveedores, migración de datos. Es decir, esta etapa es necesaria en cualquier proyecto de gestión de datos donde es requisito indispensable contar con datos comprensibles, consistentes, limpios y actualizados, incluyendo minería, la ciencia de los datos y cualquier actividad que requiera una organización para la toma de decisiones consolidada de sus datos. En esta etapa de consolidación de los datos, la calidad de datos de las fuentes es fundamental. Garantizarla es imprescindible para propiciar que el usuario se apropie de la herramienta y realmente sea un insumo para la toma de decisiones. Tampoco podemos invertir demasiado tiempo en esta etapa sin tener resultados visibles porque es necesario mostrar resultados de nuestro trabajo en el corto plazo. Por esto, algunas preguntas que pueden dirigir el análisis sobre por donde comenzar seguramente han surgido ya en la etapa de análisis de requisitos y de diseño, pero que son importantes de retomar. Dado nuestro objetivo final, al momento de determinar los hitos del proyecto, ¿cuáles son las fuentes de datos más confiables? ¿están relacionadas con usuarios proactivos y comprometidos con los cuales se puede contar en caso de tener que realizar consultas específicas? ¿tienen impacto sólo en ese departamento o es una de las dimensiones conformadas más significativas? ¿qué podemos decir de la documentación asociada a las fuentes? ¿es plausible tomar un subconjunto de ellas para comenzar una integración mínima que nos permita tener resultados visibles rápidamente? Por ejemplo un tablero de control con indicadores relacionados, un esquema de sugerencias para los clientes, entre otras posibilidades. La idea es “como empezamos a comer este elefante”. A continuación se presenta un resumen de las áreas de integración de datos más habituales. Kimball, en su libro que utilizamos como material en la cátedra y lo incluimos en las referencias del capítulo, presenta 34 subsistemas de ETL, cada uno de ellos con sus objetivos y particularidades que se incluyen en el anexo 1 de este capítulo.

La integración dentro de una organización puede plantearse en cuatro áreas significativas:

- Integración de Datos: visión única de todos los datos del negocio y es el contexto de la inteligencia del negocio, principal abordaje de la materia.

- Integración de Aplicaciones: brinda una visión unificada de todas las aplicaciones internas como externas de la organización. Se coordinan los flujos de eventos (transacciones, mensajes, datos, etc.) entre aplicaciones por ejemplo a través de APIs (Application Programming Interfaces). Para lograr que esta integración funcione es imprescindible realizar previamente un trabajo de integración de datos compartidos, por ejemplo de tipos de documento, género, información de geolocalización, sólo por nombrar las más habituales. En el contexto de la Universidad Nacional de La Plata podemos mencionar integraciones exitosas entre las aplicaciones internas de gestión académica *SIU Guaraní* con el sistema de gestión de Títulos *Quimey*, donde el trámite completo de cada diploma de lleva a través de Quimey y el estudiante ve cada paso del workflow a través de un reporte en SIU Guaraní.
- Integración de Procesos de Negocio: visión unificada de todos los procesos de negocio. Esto es muy importante, sobre todo porque la integración de aplicaciones suele darse en forma aislada.
- Integración de la Interacción de Usuarios: proporciona una visión unificada al usuario final, a partir de una interfaz segura y personalizada.

A continuación se hará hincapié en la integración de datos en general y los procesos de ETL existentes para llevarla a cabo con éxito, imprescindible en un proyecto de BI.

Los procesos de ETL no son algo nuevo, e incluso muchos técnicos lo realizan implícitamente al brindar un informe específico o realizar una migración de datos entre distintas aplicaciones. A medida que las necesidades del negocio se diversifican y los tipos de los datos a analizar también, las herramientas para completar esta tarea también. Por ejemplo, los datos a analizar provienen de distintas fuentes, no siempre son archivos de texto con un formato conocido y fácilmente interpretable. Se clasifican entonces de la siguiente manera:

- Estructurados: contenidos en una base de datos
- Semi estructurados: formatos legibles por máquinas, si bien no están del todo tabulados como planillas Excel, archivos CSV, HTMLs que pueden obtenerse mediante técnicas estándar de extracción de datos.
- No estructurados: formato legible para personas y no tanto para máquinas, como un documento en formato de texto enriquecido, PDFs, un post en un blog o en una red social, que pueden extraerse mediante técnicas más avanzadas de text mining.

Por este motivo, la funcionalidad brindada por las aplicaciones de integración de datos también se ven diversificadas. El siguiente diagrama, tomado de Nicolás Gerard BI Consultant es muy ilustrativo en forma general, respecto a las etapas involucradas, e indica, con mucho acierto, “ETL es mucho más que un proceso batch”:

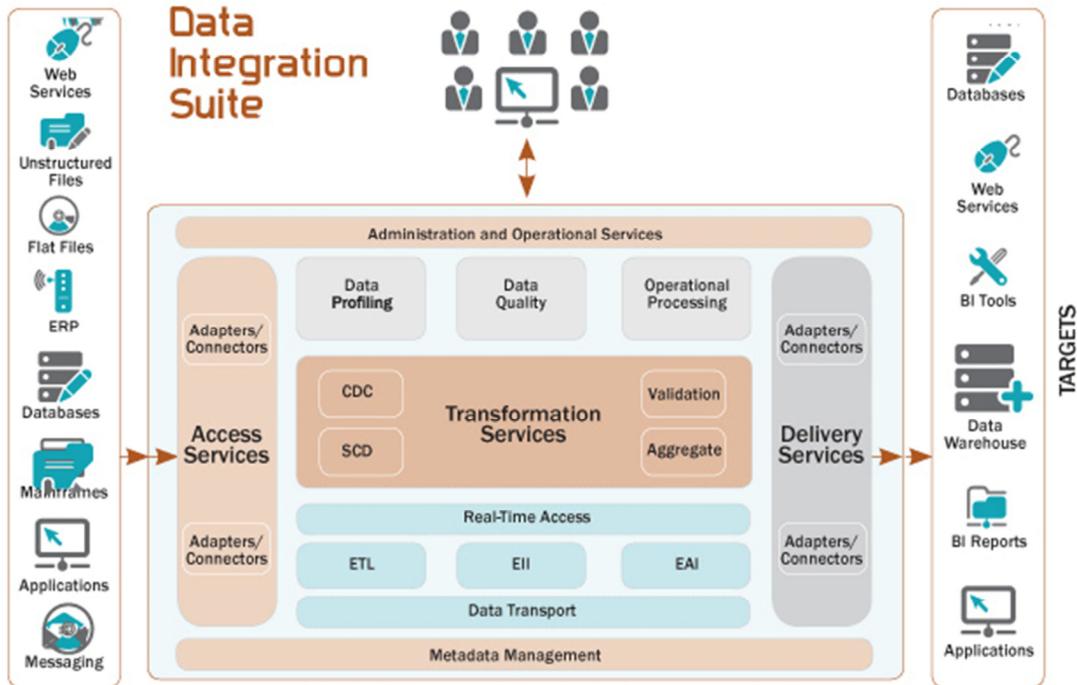


Fig. 3.8 - Suite de integración de datos. ETL

Como se puede observar en la figura 3.8, se requieren herramientas para:

- Data Quality: es el elemento central y se debe garantizar para datos no transaccionales y transaccionales. En esta etapa se suelen realizar tareas relacionadas con el análisis de datos faltantes (pueden ser registros completos o campos en vacío que fueron depurados por incompatibilidad de tipos), el análisis de datos mal cargados o erróneos, detección de anomalías y metadatos. Incluye actividades data profiling, data matching, definición del ciclo de vida de los datos, corrección de los datos, modelado y monitoreo adecuado.
- Data Profiling: actividades relacionadas con el relevamiento del universo de datos a trabajar. Se analiza la calidad de los mismos identificando el dueño de los datos, origen y producción de los datos, puntos de alteración, relevancia, frecuencia de actualización y consumidores, etc. Suele incluir tareas relacionadas con la detección de valores nulos o vacíos, detección de valores inválidos por ejemplo valores textuales en campos numéricos, detección de valores semánticamente incorrectos por ejemplo fechas de nacimiento incoherentes, valores absurdos, etc. Detección de valores atípicos (outliers) mediante técnicas estadísticas, detección de datos nominales dispersos por ejemplo Femenino, F ,Fem, Mujer; detección de duplicados mediante técnicas de huella, n-gram ; entre otras.
- Data Matching: actividades relacionadas con la detección de datos duplicados y la relación entre dos fuentes de datos que no tienen campos de unión. Se suelen utilizar técnicas más avanzadas como la detección de unidades familiares (householding) dado que pueden existir numerosos fuentes de datos y hay unicidades no establecidas en la base de datos que dependen del comportamiento y el alcance de

las aplicaciones. Entre de las técnicas de identificación de duplicados podemos nombrar Fingerprint, N-Gram Fingerprint, Phonetic Fingerprint, Nearest Neighbor, Levenshtein Distance y PPM.

- Servicios de transformación: Change Data Capture o CDC, Slowly Changed Dimension o SCD, validación y agregación. Se describen a continuación.
- Acceso en tiempo real: los usuarios acceden directamente a los resultados procesados, en algunos casos aceptando cierto grado de incertidumbre por ejemplo en el análisis de streaming de big data.
- Extract, Transform & Load: se extraen los datos en el origen, se transforman según las necesidades del negocio para la integración y se cargan en el destino, que suele ser una base de datos y/o archivos, aunque pueden ser colas de mensajes en un middleware por ejemplo u otras fuentes no estructuradas. Las herramientas de ETL suelen incluir una interfaz gráfica que permita gestionar todo el proceso. Permiten documentar cómo estos datos se transforman y se almacena esta información en un catálogo propio de metadatos, se registran las ejecuciones y los resultados en un log de errores o de cambios.
- Enterprise Information Integration (EII): permiten a las aplicaciones el acceso a datos en un archivo de texto, una base de datos o un web services, que se encuentran dispersos como si estuvieran residiendo en una base datos común. Se realiza la consulta en forma federada, que se verá en detalle a continuación, se consolidan adecuadamente y se devuelven a la aplicación que realizó la consulta. Estas herramientas se caracterizan por brindar *transparencia* (los datos parecen estar en un único origen), *heterogeneidad* al brindar una única vista de análisis a partir de datos en distintas fuentes e incluso distintos tipos, tales como estructurados y no estructurados. *Extensibilidad* para federar cualquier fuente de datos, alta funcionalidad, autonomía y rendimiento a fin de acceder en forma no disruptiva y eficiente de acuerdo a la naturaleza de la misma.
- Enterprise Application Integration (EAI): según Wikipedia, el objetivo de un sistema EAI es intercambiar información operativa y financiera de las organizaciones, intra y extra organización, sin necesidad de compartir sus estructuras internas o re implementar las reglas de negocio. Es para que las aplicaciones individuales interactúen pero también para generar conocimiento y valor a partir de esta interacción. Se suele utilizar para integración de datos, integración de procesos de negocio dentro de la misma organización, independencia del proveedor, o FACADE común entendido como un front-end de un conjunto de aplicaciones, proporcionando un acceso único a las distintas aplicaciones, a partir de una interfaz común.
- Capa de transporte de datos: los sistemas de transporte habituales son a través de SQL, tablas intermedias (staging) que dan lugar a una replicación del tipo SQL, o a través de un middleware de gestión de colas por ejemplo.

- Gestión de metadatos: fundamental contar con un esquema de metadatos versionado y consistente en todo el proceso de integración. Para cada dato dentro de los esquemas, identificar rápidamente su dueño, consumidores, mecanismos y frecuencia de actualización, validaciones, etc.
- Entrega/Acceso a datos: a través de servicios y conectores implementados, por ejemplo, a través de webservices.

Técnicas de Integración de Datos

Entre las técnicas de integración de datos podemos mencionar:

- Propagación de Datos: copia los datos de un lugar a otro, local o remoto. Los datos pueden extraerse mediante programas que generen un archivo que traslada a un destino donde se importa a una base de datos, o en forma más eficiente sólo se trasladan las modificaciones y se realiza una carga incremental. Esto suele realizarse en línea y puede ser como un intercambio bidireccional o distribuido.
- Consolidación de datos: se capturan los datos de distintos orígenes y se consolidan en un destino, donde se almacena una copia de todos los datos. El ejemplo típico es el ODS de un data warehouse alimentado por varios entornos de producción. Esta técnica suele tener tiempos de latencia altos, actualizando los datos con tiempos prefijados. Cuando se requiere tiempo de latencia bajos se utiliza la técnica push, donde se integran sólo los cambios (CDC).
- Federación de datos: proporciona a las aplicaciones una visión lógica virtual a una o más fuente de datos, que pueden estar en los mismos o distintos gestores de datos o máquinas. Cuando una aplicación manda una consulta a la federación el motor se ocupa de descomponer la consulta en consultas individuales y las lanza contra los motores físicos individuales. Cuando recibe todas las respuestas o a medida que las va recibiendo, las consolida realizando summarizaciones, ordenaciones o agregaciones según el caso, para resolver la consulta original y devolver los datos a la aplicación original. Un elemento fundamental es el catálogo de metadatos común con la estructura de los datos, localización, cardinalidad, etc. con fines de optimización y para el acceso más eficiente a los datos.
- CDC (Change Data Capture): permiten tomar los datos en el origen y propagarlos manteniendo la consistencia de los mismos en todos los ambientes. Entre las principales técnicas podemos mencionar:
 - CDC por aplicación: la misma aplicación actualiza los destinos o mantiene actualizada una tabla de paso (staging) mediante una operación de INSERT dentro de la misma unidad lógica de trabajo.
 - CDC por timestamp: cuando los datos en el origen almacenan el timestamp de la última actualización. El proceso toma todas las modificaciones desde la última vez que se ejecutó. Esta última captura.

- CDC por triggers: se definen a nivel de tabla y se disparan cuando se realiza una operación sobre una tabla de la base de datos relacional, para generar modificaciones en una tabla destino, local o remota.
 - CDC por captura de logs: monitorean el buffer de log para buscar las modificaciones en las tablas origen y aplicarlas en el destino.
- Técnicas híbridas: la técnica elegida para la integración dependerá de la naturaleza del negocio y de los requisitos tecnológicos y las restricciones presupuestarias. En la práctica estas técnicas se suelen combinar.

Actualmente existen en el mercado numerosas aplicaciones que expanden las posibilidades de ETL para incluir EII e EAI como mencionamos previamente y además una arquitectura orientada a servicios que ofrecen integración a través de procesos batch, interoperabilidad de aplicaciones o en tiempo real para aplicaciones de BI. Entre algunas comerciales y otras open source como:

- Pentaho Data Integration <http://www.pentaho.com/product/data-integration>
- Pentaho Data Integration DataClean plugin <http://wiki.pentaho.com/display/EAI/Kettle+Data+Profiling+with+DataCleaner>
- R-Studio: muy útil para realizar análisis sobre los datos y detectar inconsistencias utilizando el lenguaje R. Numerosas librerías que permiten realizar cleasing, detección de duplicados, matching, etc. Suite open source. <https://www.rstudio.com/>
- Open Refine: inicialmente un proyecto de Google y actualmente.
- DataCleaner https://datacleaner.org/get_datacleaner_ce
- Talend Data Integration <https://www.talend.com/products/data-integration/>
- Talend Data Quality <https://www.talend.com/products/data-quality/>
- Talend Data Preparation <https://www.talend.com/products/data-preparation/>
- IBM InfoSphere Information Analyzer <http://www-03.ibm.com/software/products/es/ibminfoinfoanal>
- IBM InfoSphere QualityStage <http://www-03.ibm.com/software/products/es/ibminfoqual>
- IBM BigQuality <https://www.ibm.com/analytics/us/en/unified-governance-integration/data-quality/>

Casos de estudio

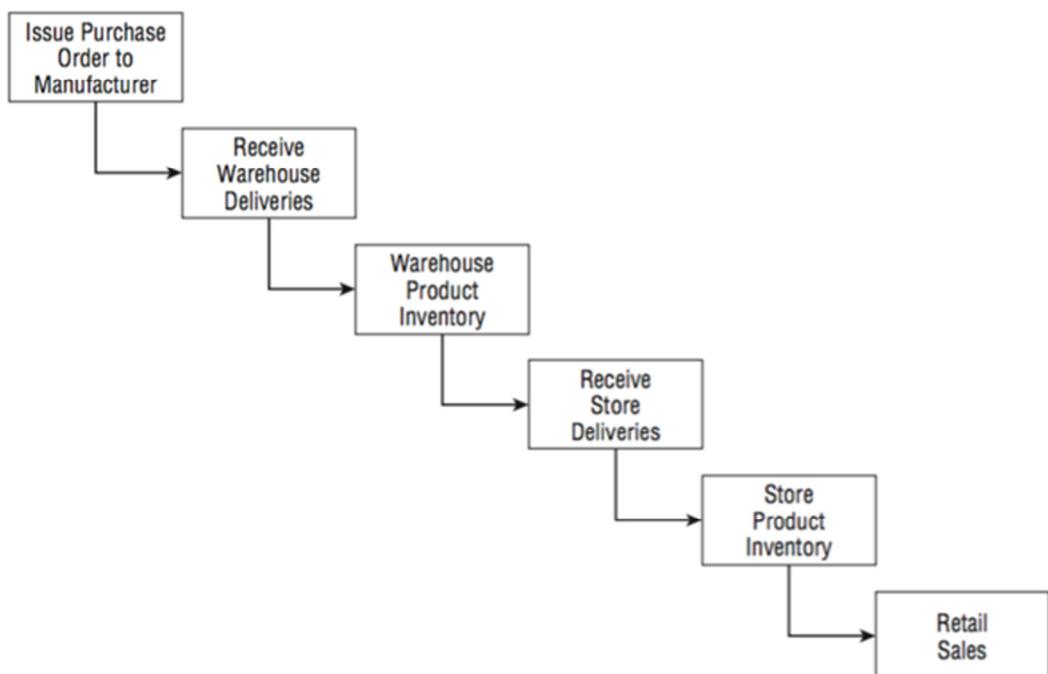
Durante la cursada, los estudiantes analizan los casos presentados por Kimball en su libro y lo presentan en una clase especial. Cada capítulo cuenta con definiciones y situaciones que ha experimentado el autor en su extensa trayectoria de consultor. Luego, todos los trabajos se comparten en un repositorio común y los puntos más importantes son evaluados en un cuestionario. A continuación se presentan tres los casos de todos los que se brindan en el libro, de los estudiantes más destacados.

Inventario

La mayoría de las organizaciones tienen una “cadena de valores” de sus procesos de negocio más importantes.

La cadena de valores describe el flujo de actividades primarias de la organización, y el principal objetivo de BI/DW es analizar la performance de estas.

ejemplo: Un vendedor le compra a sus proveedores y guarda los productos en un galpón de mercadería, y luego lleva algunos al local de venta.

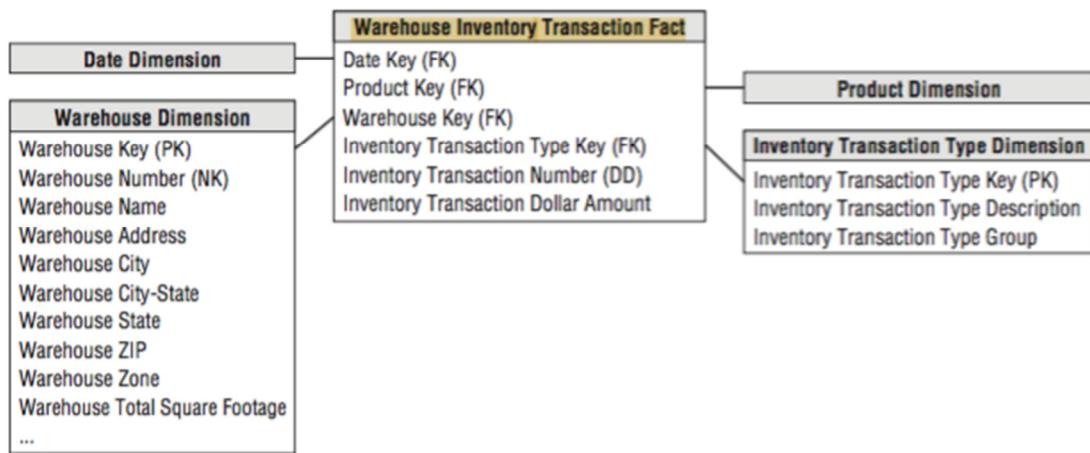


Diariamente se genera una fila por cada producto en cada local.

¿Qué pasa con 60.000 productos en 100 locales ? ¿Cuál es una posible solución?

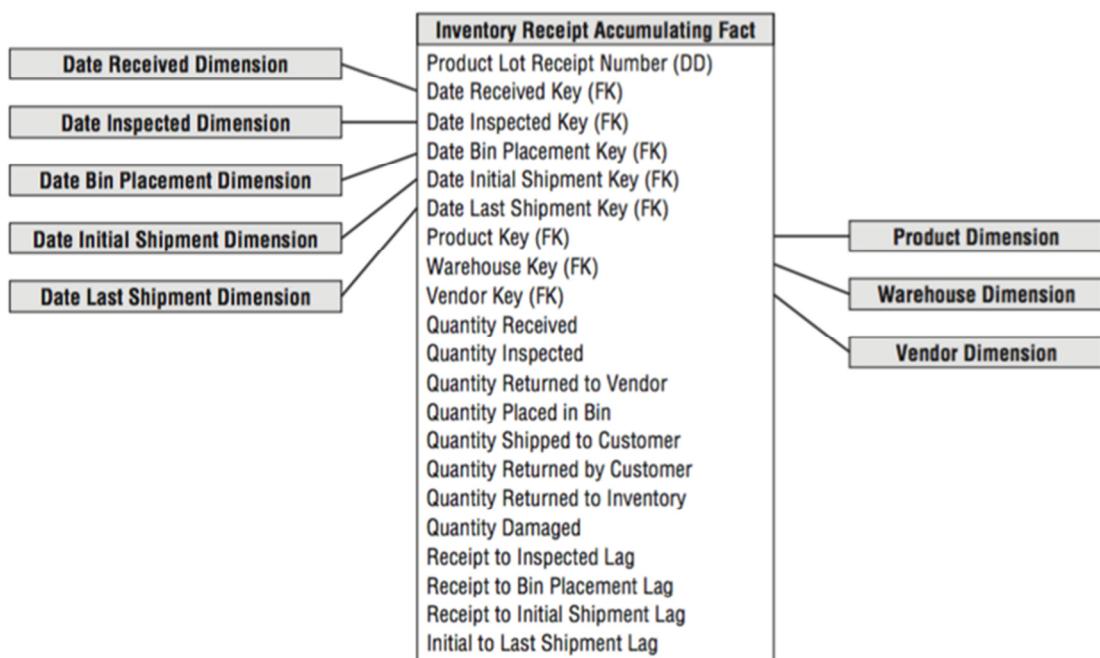
Hechos semi-agregados

Hay datos que no son agregables en algunas dimensiones porque representan un momento particular. Ejemplo: Balances económicos diarios de ingreso, una solución: Promedios (OLAP, no sql), otra posible solución son las tablas de hechos mejoradas: deltas; las mismas generan una fila por cada transacción que sucede en el negocio.



¿Qué pasa si hay distintos tipos de transacciones?

Se usan para procesos que tienen comienzo, fin e hitos en el medio, es decir una misma fila que se actualiza por producto.



Adquisiciones

La adquisición es una actividad de negocios muy importante dentro de las distintas empresas y organizaciones, ya que de ella dependen los precios de reventa, también tiene que ver con la reducción de costos significativos al reducir el número de proveedores y al realizar mejores acuerdos con los mismos.

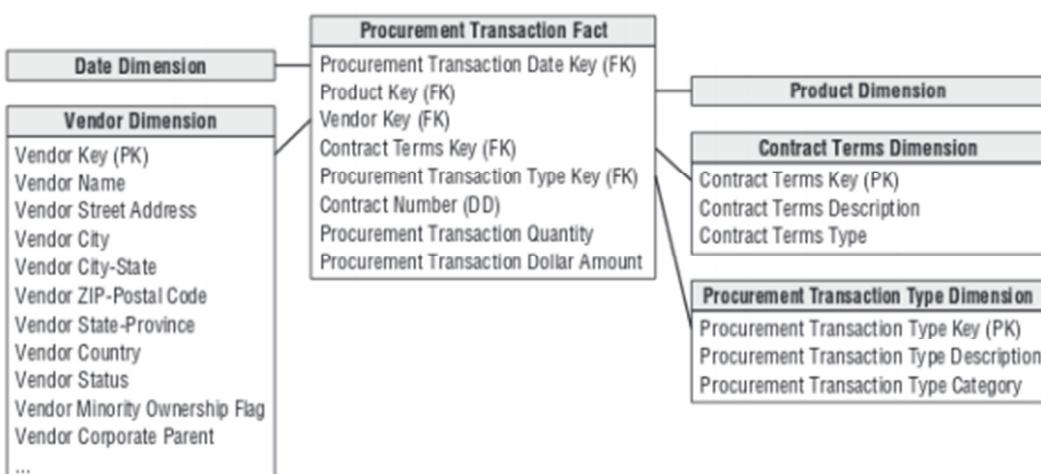
Este proceso es esencial tanto para minoristas como también mayoristas, y además tanto para aquellas empresas de compra de productos como así también de materias primas para la fabricación.

- ¿Cuáles materiales o productos que se compran más frecuentemente?
- ¿Cuántos vendedores suministran estos productos? ¿En qué precios?

En cuanto a la demanda,

- ¿Hay oportunidades de negociar precios favorables para consolidar proveedores, asegurando abastecimiento único, o haciendo compras garantizadas?
- ¿Sus empleados compran de los vendedores preferidos o bordean los acuerdos de venta negociables con gastos disidentes?
- ¿Está recibiendo el precio negociado con sus vendedores o hay allí discrepancia de precio de compra con el contrato del vendedor?
- ¿Cómo funcionan sus vendedores? ¿Cuál es el vendedor que llena la tarifa?
- ¿A qué tiempo funciona la entrega? ¿Hay entregas pendientes? ¿Tasa de rechazo basada en el recibo de inspección?

Para el proceso de diseño dimensional de cuatro pasos, se determina qué adquisición del proceso de negocios va a ser modelada. En el estudio de proceso, se observa una serie de transacciones de compras, tales como solicitudes de compra, órdenes de compra, avisos de entrega, recibos y pagos. Se podría inicialmente diseñar una tabla de hechos con el grano de una fila por cada operación de adquisición con la fecha de la transacción, el producto, el proveedor, los términos del contrato y tipo de transacción de adquisición como clave de dimensiones. La cantidad de transacciones de compras y cantidad en dólares son los hechos. El diseño resultante se muestra en la siguiente figura.



Si se trabaja con las adquisiciones de fabricación, los productos de las materias primas probablemente se encuentran en una tabla de dimensiones de materiales crudos separados, en lugar de incluirse en la dimensión del producto para los productos vendibles. Las dimensiones de los vendedores, términos del contrato, y el tipo de operación de adquisición son nuevas en este esquema. La dimensión de vendedor contiene una fila para cada uno de ellos, con atributos descriptivos para soportar una variedad de análisis. La dimensión de los términos del contrato contiene una fila para cada conjunto generalizado de términos negociados. La dimensión del tipo de transacción de adquisiciones permite agrupar o filtrar los tipos de transacciones, tales como órdenes de compra. El número de contrato es una dimensión degenerada; que podría ser utilizado para determinar el volumen de negocios que se realizan en cada contrato negociado.

Tablas de hechos de transacciones simples vs tablas de hechos de transacciones múltiples.

No existe una fórmula sencilla para hacer la determinación definitiva de si se debe utilizar una sola tabla de hechos o varias tablas de hechos. Para esta decisión de diseño, tener en cuenta las siguientes consideraciones:

- El objetivo es reducir la complejidad mediante la presentación de los datos de la forma más eficaz posible para los usuarios de negocios.
- La existencia de números de control independientes para cada paso en el proceso es un indicio de que se trata de procesos separados. Ante esta situación, debería inclinarse hacia tablas de hechos separadas.
- ¿Son múltiples sistemas de origen capturando métricas con granulaciones únicas? Si esto es así, esto sugeriría tablas de hechos separados.
- ¿Cuál es la dimensionalidad de los hechos? En este ejemplo de adquisiciones, varios de las dimensiones son aplicables a algunos tipos de transacción, pero no a otros. Esto conduciría a separar las tablas de hechos.

Una forma sencilla de tener en cuenta estas consideraciones será la elaboración de una matriz de bus. Como se ilustra en la siguiente figura:

Business Processes	Atomic Granularity	Metrics	Date	Product	Vendor	Contract Terms	Employee	Warehouse	Carrier
Purchase Requisitions	1 row per requisition line	Requisition Quantity & Dollars	X	X	X	X	X		
Purchase Orders	1 row per PO line	PO Quantity & Dollars	X	X	X	X	X	X	X
Shipping Notifications	1 row per shipping notice line	Shipped Quantity	X	X	X		X	X	X
Warehouse Receipts	1 row per receipt line	Received Quantity	X	X	X		X	X	X
Vendor Invoices	1 row per invoice line	Invoice Quantity & Dollars	X	X	X	X	X	X	
Vendor Payments	1 row per payment	Invoice, Discount & Net Payment Dollars	X	X	X	X		X	

Dimensiones lentamente cambiantes básicas

Los atributos de las dimensiones a pesar de que son relativamente estáticos, cambian, con bastante lentitud, con el tiempo, por lo que es necesario tener una estrategia para poder manejar esos cambios dependiendo de objetivos o enfoques, quienes gestionan esos datos. Para ello se desarrollan distintas técnicas (denominadas SCD, Slowly Changing Dimension).

- Tipo 0 : Conservar el original. Con el tipo 0, el valor del atributo de la dimensión nunca cambia, por lo que siempre se agrupan por este valor original.
- Tipo 1: Sobreescribir. Se sobrescribe el valor antiguo del atributo en la fila de dimensión, reemplazandolo con el valor actual; el atributo siempre refleja la asignación más reciente (Usualmente este tipo es utilizado en casos en donde la información histórica no sea importante de mantener).
- Tipo 2: Agregar una nueva fila. Esta estrategia requiere que se agreguen algunas columnas adicionales a la tabla de dimensión, para almacenar el historial de cambios. Es común agregar columnas con la fecha de vigencia y de vencimiento para referirse a los momentos en que los valores de los atributos de la fila son validos o inválidos.
- Tipo 3: Agregar nuevo atributo (columna). Esta estrategia requiere que se agregue a la tabla de dimensión una columna adicional por cada columna cuyos valores se desea mantener un historial de cambios (Esta técnica permite guardar una limitada información de cambios).
- Tipo 4: Agregar mini-dimensión. Su función básica es almacenar en una tabla adicional los detalles de cambios históricos realizados en una tabla de dimensión. Esta tabla histórica indicará por ejemplo qué tipo de operación se ha realizado (Insert, Update, Delete), sobre qué campo y en qué fecha. Dimensiones lentamente cambiantes híbridas:
- Tipo 5: Mini-Dimensión y estabilizadores de tipo 1. Esta técnica consiste en añadir una PK de la mini-dimensión actual como un atributo FK en la dimensión principal. Esta FK es un atributo de tipo 1.
- Tipo 6: Agregar atributos de tipo 1 a una dimensión de tipo 2. Con este enfoque híbrido, se emite una nueva fila para capturar el cambio (tipo 2) y añadir una nueva columna para realizar un seguimiento histórico del atributo (tipo 3), donde los cambios posteriores se manejan como una respuesta de tipo 1.
- Tipo 7: Doble dimensiones tipo 1 y tipo 2. Este enfoque ofrece la misma funcionalidad que el tipo 6, aunque la respuesta de tipo 6 genera más columnas de atributos en una sola tabla de dimensiones, este enfoque se basa en dos claves externas de la tabla de hechos.

Análisis Financiero

Gerentes de todos los niveles de una organización utilizan métricas resultantes de procesos contables para toma de decisiones o para realizar diferentes análisis. Un sistema de DW/BI puede proveer una sola fuente de información, la cual es utilizable y entendible, asegurando

que todos trabajen con la misma información, con las mismas definiciones y herramientas. Como los destinatarios de esta información son gerentes de todos los niveles de la organización, cada uno de ellos necesitará un subconjunto distinto de información, diferentes formatos y posiblemente diferente frecuencia de análisis. Los analistas utilizan la información de más alto nivel y luego profundizan en entradas diarias para mayor detalle. Los ejecutivos utilizarán la información que brinda el sistema para nutrir sus tableros (dashboards) y sus indicadores de desempeño (performance). A pesar de tener un impacto a lo largo de toda la organización, en un comienzo muchos negocios enfocan sus implementaciones de DW/BI en estrategias para la generación de ingresos.

	<i>Date</i>	<i>Ledger</i>	<i>Account</i>	<i>Organization</i>	<i>Budget Line</i>	<i>Commitment Profile</i>	<i>Payment Profile</i>
General Ledger Transactions	X	X	X	X			
General Ledger Snapshot	X	X	X	X			
Budget	X	X	X	X	X		
Commitment	X	X	X	X	X	X	
Payments	X	X	X	X	X	X	X
Actual-Budget Variance	X	X	X	X			

La figura muestra un extracto de una matriz contable de una organización. Las dimensiones asociadas con los procesos contables suelen ser usados exclusivamente por procesos relacionados a la contabilidad de libros contables, a diferencia de las dimensiones de producto, cliente principal y empleado, que son usadas repetidamente a lo largo de diversos procesos de negocio.

Capturas periódicas y transacciones diarias de libros contables

El libro de contabilidad es un sistema financiero central que junta la información detallada que recolectan los libros contables o sistemas separados, para la compra, pagos (lo que se le debe a otros) y recibos (lo que los otros te deben). Se hace evidente la necesidad de dos esquemas complementarios: tablas de hechos de capturas periódicas y transacciones. Comenzaremos ahondando en las capturas para las cuentas de un libro contable al final de cada período fiscal. La granularidad de esta captura es una fila por cada período contable para el nivel más granular en los gráficos para cuentas del libro contable.

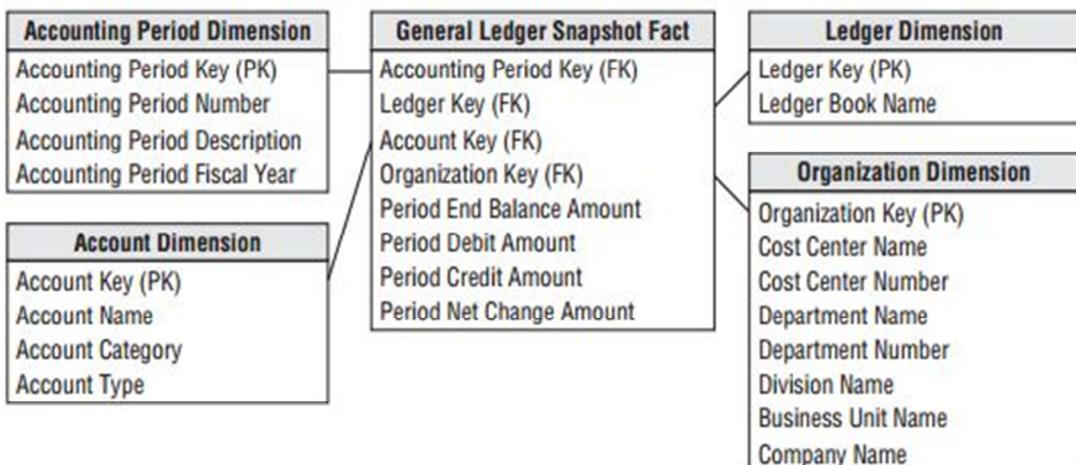
Plan de cuentas

La parte más importante de un libro contable es el plan de cuentas. Éste es el resumen de un identificador inteligente, ya que por lo general consiste en una serie de identificadores. Por ejemplo, el primer set de números puede identificar una cuenta, su tipo (por ejemplo, bienes,

entrada, gasto), y otras características de la cuenta también. A veces dicha inteligencia está embebida en el esquema numérico de la cuenta. Obviamente, en la DW se incluirá el tipo de la cuenta como un atributo de una dimensión en vez de forzar a los usuarios a hacer un análisis del esquema numérico de la cuenta. El plan de cuentas asocia el centro de costo de la organización con la cuenta. Si el libro contable de la empresa cruza información con múltiples unidades de negocio, el plan de cuentas va a variar en cada unidad. En este caso de estudio, el plan de cuentas se descompone en dos dimensiones. Una representa las cuentas del libro contable, mientras que la otra representa las unidades que componen la organización. Si la tarea asignada consiste en construir un libro contable que abarque múltiples organizaciones en el sistema de DW/BI, se debería intentar hacer un plan de cuentas donde el tipo de cuenta signifique lo mismo a lo largo de cada organización. A nivel de datos, significa que la dimensión de cuenta central que definimos contiene nombres de cuentas cuidadosamente definidos. Este tipo de dimensiones tienen un nombre ya conocido en el ámbito financiero: plan de cuentas uniformes. El libro contable a veces rastrea resultados financieros para múltiples conjuntos de libros o sublibros para sustentar diferentes requerimientos, como impuestos o informes para las agencias reguladoras (impuestos). Se puede tratar como una dimensión aparte porque es un filtro muy importante.

Cierre de período

Al final de cada período contable, la organización financiera es responsable de determinar los resultados financieros, y con ello realizar reportes tanto internos como externos. Reconciliar y hacer un balance de los libros antes de que se cierre el período con la aprobación oficial suele ser un proceso que toma varios días. A partir de este punto, el foco de las finanzas se posa en el reporte e interpretación de los resultados. Por lo general, cada mes se ejecutan muchos reportes y respuestas variadas para las mismas preguntas. Los analistas financieros están constantemente intentando coordinar el proceso para el cierre del período, reconciliación y reporte de resultados del libro contable. A pesar de que los sistemas operacionales de libros contables suelen soportar estos requisitos, puede que se vean sobrepasados de información, especialmente si se está trabajando con libros contables modernos. Este capítulo se centra en el fácil análisis de los resultados del cierre financiero, en vez de facilitar el cierre en sí. De todos modos, en muchas organizaciones los balances son cargados en el sistema de DW/BI para aprovechar la capacidad de presentación del sistema, logrando con esto hacer los ajustes operacionales adecuados antes del cierre del período.



El esquema muestra el balance de un libro contable al final de cada periodo contable. Esto puede ser útil para muchos tipos de análisis financieros, tales como patrones en las tendencias, comparaciones por período, etc. Las dos dimensiones más importantes del esquema planteado son Cuenta y Organización. La dimensión Cuenta deriva de un plan de cuentas uniforme de la empresa. La dimensión Organización describe las entidades de reporte financieros en la empresa. Desafortunadamente, estas dos dimensiones casi nunca se corresponden con la dimensión operativa como es el caso de cliente, producto o servicio.

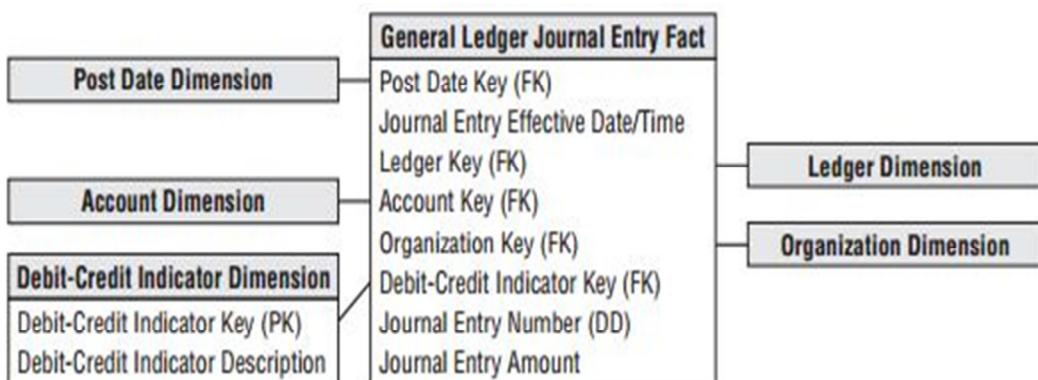
Hechos fecha Existe una tendencia a almacenar columnas de “to-date” en las tablas de hechos. Por lo general se piensa que puede ser útil almacenar totales de un cuarto de año (para empresas que hacen cortes periódicos cada 4 meses) o de un año en cada fila de la tabla de hechos así no se necesita calcular dichos valores. Debemos recordar que los hechos numéricos deben ser consistentes respecto a la granularidad. Hechos correspondiente a fechas no respetan la granularidad, lo que lleva a diversos riesgos. Cuando las tablas de hechos son consultadas y sumarizadas de maneras arbitrarias, estos hechos que no respetan la granularidad producen resultados sin sentido. Es por esto que se los debe dejar fuera del esquema y calcular estos totales en los reportes de BI. Es importante destacar que los totales a una fecha deberían ser calculados, pero no almacenados en las tablas de hechos. -

Múltiples monedas

Con la información financiera por lo general se quieren representar los hechos en términos de la moneda local y de la moneda corporativa estandarizada. En este caso, cada fila de la tabla de hechos representaría un conjunto de hechos expresados en términos de la moneda local y otro conjunto separado de hechos expresados en el equivalente a la moneda corporativa. Haciendo esto se puede sumarizar fácilmente los hechos en la moneda corporativa sin tener que pasar por las aplicaciones de BI. Se asume que también se agregaría la dimensión Moneda como una clave foránea en la tabla de hechos para identificar el tipo de moneda local.

Capturas periódicas y transacciones diarias de libros contables

Ya analizamos las capturas al final de un periodo fiscal, las cuales nos proveen de una gran variedad de análisis financieros posibles. Ahora bien, puede que muchos usuarios necesiten entrar en los detalles subyacentes. Si se detecta una anomalía en el resumen, los analistas querrán observar las transacciones de forma detallada para hallar el problema. Se podría completar una captura periódica con una entrada detallada de las transacciones. Obviamente, las cuentas de pagos y recibos pueden contener transacciones a un nivel de mayor detalle, que pueden ser capturados en una tabla de hechos separada con dimensiones adicionales. La granularidad de la tabla de hechos ahora es de una fila por cada transacción diaria en nuestro libro contable. Esta transacción identifica la cuenta del libro contable y el importe de débito o crédito.



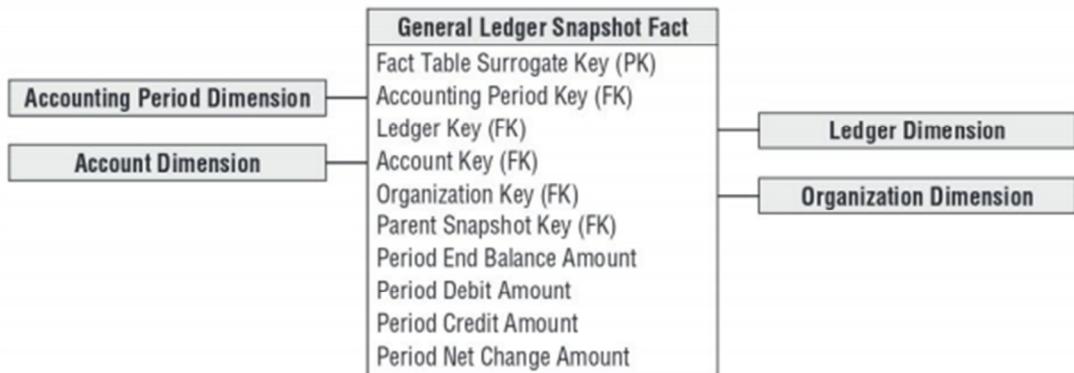
Como vemos, se incluyen las dimensiones de Cuenta y Organización, y se reutilizan otras. Se deberá agregar una dimensión de libro contable si es que se saca información de varios otros libros. Normalmente se capta la entrada diaria de transacciones a través de la fecha de posteo, por lo que se recomienda usar una tabla de fecha (enfocada en un nivel diario) en este esquema. Dependiendo de las reglas de negocio asociadas con la fuente de información, puede que se necesite una segunda dimensión de fecha para distinguir la fecha de posteo de la fecha efectiva de contabilidad. El número de asiento en el libro diario suele ser una dimensión degenerada sin vínculo de asociación a otra dimensión de la tabla. Si los números de asiento de una fuente están ordenados, entonces esta dimensión degenerada puede ser usada para ordenar las entradas diarias ya que la dimensión de fecha en esta tabla de hechos es muy gruesa para proveer dicho ordenamiento. Si los números no se encuentran ordenados, entonces puede que se tenga entrada diaria de la transacción e incluso una descripción. En estos casos, se crea una dimensión aparte para cada entrada del libro diario. Asumiendo que las descripciones no son sólo texto, esta dimensión tendrá una considerable menor cantidad de filas comparada con la tabla de hechos (la cual tendrá una fila por cada entrada diaria). Cada fila de la tabla de hechos es identificada como débito o crédito. Este indicador toma dos y sólo dos valores.

Múltiples calendarios de contabilidad fiscal

En la última imagen que presentamos, la información se toma a partir de la fecha en que se postea, pero puede que los usuarios también quieran sumarizar la información por un periodo fiscal contable. Desafortunadamente, dichos periodos no están alineados con los meses gregorianos estándar. Por ejemplo, puede que una empresa tenga 13 periodos contables de 4 semanas cada uno en un año fiscal que comienza el 1º de septiembre, en vez de un periodo de 12 meses que comienza el 1º de enero. Si se trabaja con un solo calendario fiscal, cada día en el año se corresponde con mes calendario particular, al igual con un período contable particular. A partir de estas relaciones, el calendario y los períodos contables son atributos jerárquicos en la dimensión fecha. Esta tabla de dimensión conforma simultáneamente una tabla de dimensión de mes calendario y la tabla de dimensión de periodo fiscal contable. En otras situaciones, puede que se tengan múltiples calendarios fiscales contables que varían por subsidarios o línea de negocios. Si se tiene un número fijo de calendarios fiscales únicos, entonces se puede incluir cada conjunto de atributos del calendario fiscal en una sola dimensión Fecha. Una fila en la dimensión de fecha se identificará como perteneciente al periodo contable 1 para subsidiario A, y como periodo contable 7 para subsidiario B. En una situación más compleja con un número mucho mayor de calendarios fiscales diferentes, se podría identificar el calendario fiscal oficial de la corporación en la tabla de dimensión Fecha. Así se tendrían varias opciones para localizar los calendarios fiscales que se correspondan con cada subsidiario. La aproximación más común es crear una dimensión Fecha que haga de puente con una clave formada por la fecha y las claves subsidiarias. En esta tabla tendríamos una fila por cada día por cada subsidiario. Los atributos en este “puente” serían los agrupamientos fiscales (como fecha semana fiscal y fecha final de periodo fiscal). A su vez se necesitaría de un mecanismo para filtrar un subsidiario específico en este puente. Una segunda aproximación sería crear una dimensión fecha separada para cada calendario subsidiario, usando un conjunto común de identificadores de fecha. Esta opción conviene usarla si la tabla de hecho está descentralizada por subsidiario. Dependiendo de las herramientas de BI, puede ser más fácil la primera como la segunda aproximación. Por último, se podría alojar otra clave foránea en la tabla de hechos para una tabla de dimensión de periodo fiscal subsidiario. Esta aproximación facilita el acceso del usuario por este criterio y agrega un esfuerzo adicional al sistema ETL porque debe insertar la clave correcta del periodo fiscal durante el proceso de transformación.

Desglose de jerarquías de libros contables

Las agencias gubernamentales o compañías de gran tamaño pueden tener variados libros contables ordenados en jerarquía ascendente por empresa, división o departamento. En los niveles más bajos, las entradas en los libros contables de una división pueden ser agrupadas para formar una entrada (una tupla) más simple en el libro contable de la división en cuestión. Esto es usual para la granularidad del corte (snapshot) periódico en estos libros. Una manera de modelar esta jerarquía es introduciendo la clave primaria del corte de la tabla de hecho del padre en la tabla de hecho, como se muestra en el diagrama



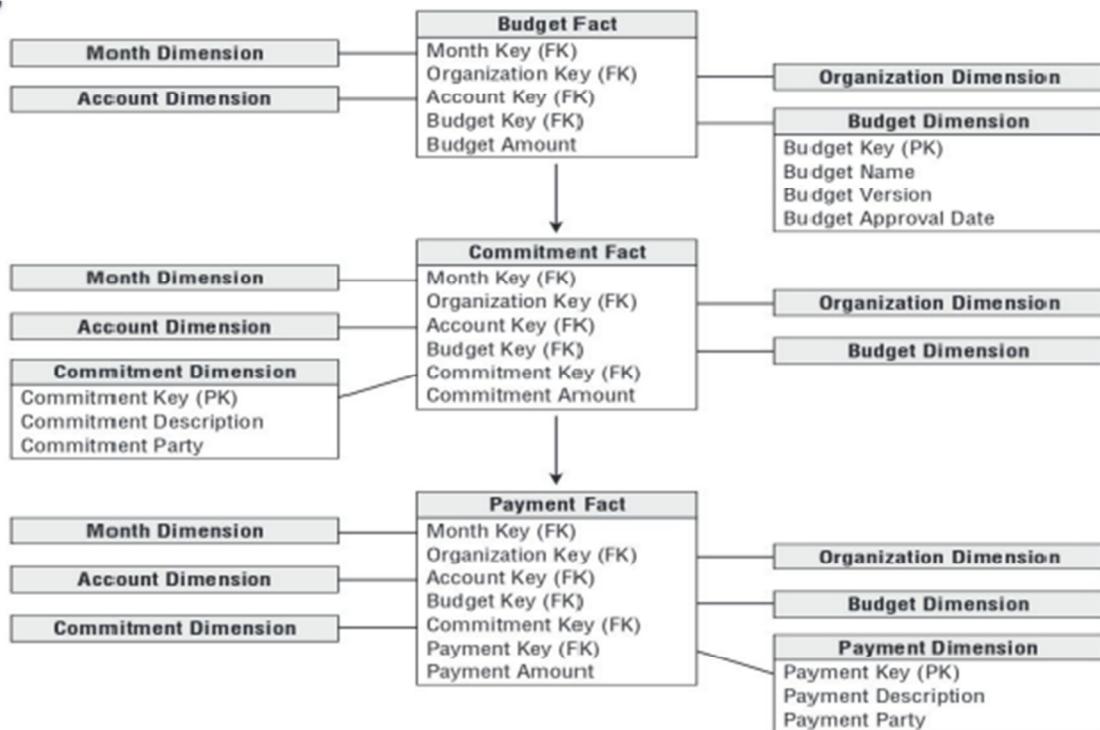
En este caso, definimos una relación padre/hijo entre las tuplas, agregamos una clave primaria explícita en la tabla de hecho, una columna con un identificador numérico que se incrementa a medida que se agregan tuplas en la tabla de hecho. Se puede utilizar la clave primaria del corte del padre para adentrarse en el libro contable correspondiente. Supongamos que detectamos un movimiento grande por un viaje en el nivel más alto del libro. Podemos tomar la clave primaria para esa tupla de alto nivel y luego recuperar todas las tuplas cuya clave del corte padre es igual a esta clave. Esto expone las tuplas en el nivel inmediatamente inferior que hacen que la tupla del registro de alto nivel tome estos valores.

Declaraciones financieras

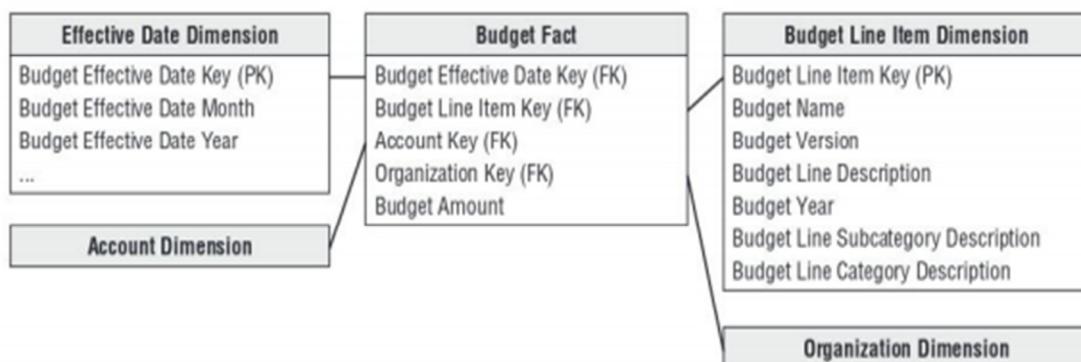
Una de las principales funciones de un sistema para el libro contable general es producir los reportes financieros oficiales, como el balance y la declaración de ingresos. El sistema operacional maneja la generación de estos reportes. El sistema de DW/BI (Data warehouse/ Business Intelligence) no tendría por objetivo reemplazar los reportes publicados por los sistemas operacionales financieros. Pero puede crear datos complementarios calculados que proveen un acceso simplificado a información que puede estar diseminada por toda la empresa. Las dimensiones de las declaraciones financieras incluyen período fiscal y centro de costo. Pero en vez de mirarse en detalle, la tabla de hecho va a tener información agregada (lograda mediante cálculos) y etiquetada con la declaración financiera correspondiente. De esta forma los gerentes pueden ver fácilmente la tendencia en performance para su organización a través del tiempo.

Planeamiento presupuestario y procesos asociados.

Los gerentes de áreas hacen un pedido de presupuesto, pero luego quieren hacer un seguimiento de lo gastado contra lo presupuestado, o ver si necesitan hacer ajustes en sus proyecciones por cambios en el ambiente de negocios. También si restringieron o hubo cambios en el presupuesto desde su aprobación, o si el ritmo de gastos no viene de acuerdo a lo planeado. De esta manera, el proceso de planeamiento financiero es cada vez más dinámico. Para el modelo dimensional, se puede ver la cadena de presupuesto como una serie de tablas de hecho.



Tenemos una tabla de presupuesto, una de compromiso (de pago) y una de pagos. El movimiento empieza con la tabla de presupuesto para una organización y para una cuenta. Mientras el período avanza, compromisos de pago se realizan contra esos presupuestos. Finalmente, se realizan pagos contra esos compromisos y por lo tanto contra los correspondientes presupuestos. Cada tupla de la tabla de presupuesto identifica lo que cada organización de la compañía puede gastar, en qué propósito y en qué período de tiempo. De manera similar, si la tupla identifica un ingreso, este representa lo que la organización espera ganar, de qué origen de ingresos y en qué período de tiempo.



Para la tabla de hecho Presupuesto dimensiones:

- Fecha efectiva -> el 'mes efectivo' es el mes donde hay cambios en el presupuesto. Las primeras entradas para un presupuesto anual en particular mostrarán el mes efectivo de aprobación del presupuesto. Si hay una modificación o actualización del ítem de presupuesto du-

rante el año corriente habrá nuevas entradas en la dimensión fecha. Por eso la granularidad es por ítem de presupuesto y por movimiento de saldo. Si durante el año no hay cambios, entonces las únicas entradas van a ser las de la aprobación del presupuesto.

- Cuenta -> relacionada con cuentas de la contabilidad general. Una línea de presupuestos puede afectar varias cuentas, por ejemplo: 'suministros de oficina' puede incluir la cuenta 'Muebles y útiles' (lapiceras, sillas de oficina) y la cuenta 'gastos generales' (agua envasada, café).
- Ítem de Presupuesto -> por ejemplo suministros de oficina. Tiene categoría y subcategoría, aunque los presupuestos no siempre las tengan. En ese caso se recomienda repetir la categoría en la subcategoría para evitar líneas de 'No Aplica'. También identifica el año de presupuesto o la versión (para contemplar cambios posteriores)
- Organización -> Ventas, Cobranzas, Compras, dependiendo de la organización de la compañía. Cuando el año del presupuesto comienza, los gerentes empiezan a hacer compromisos (commitments) de gastos a través de órdenes de compra o contratos. Los gerentes se interesan en comparar los compromisos de gastos con el presupuesto asignado. En cambio al área de finanzas le interesan los pagos efectivos de esos compromisos asumidos.

Rol de OLAP y paquetes de soluciones analíticas financieras

Los productos OLAP han sido utilizados mayormente en reportes financieros. Los modelos dimensionales relacionales a menudo se alimentan de cubos OLAP financieros. Los cubos OLAP pueden ofrecer un alto rendimiento de consulta para uso ejecutivo. Los volúmenes de información, tanto para balances diarios o agregados de los estados financieros, en general no sobrepasan las limitaciones de tamaño de un producto multidimensional. La elección de una solución OLAP es muy adecuada para manejar las complicaciones de la organización, tanto para cálculo complejo como para manipulaciones de filas. La mayoría de los productos ofrecen funciones financieras (por ejemplo, el valor actual neto o el crecimiento compuesto), un manejo adecuado de información de los estados financieros (en el orden secuencial esperado como resultado previendo los gastos), y el tratamiento adecuado de los débitos y créditos en función del tipo de cuenta, así como las avanzadas funcionalidades para una consolidación financiera. Los cubos OLAP soportan fácilmente los modelos de seguridad complejos, tales como la limitación del acceso a los datos al tiempo que proporcionan un acceso abierto a las métricas. Dado el carácter estándar de procesamiento, en lo que refiere a contabilidad, la compra de un producto genérico, en vez de llevar a cabo un desarrollo desde el comienzo, ha sido la solución más popular en los últimos años. Casi todos los paquetes que se ofrecen incluyen una solución analítica. En muchos casos, las soluciones enlatadas, en base a la experiencia acumulada por el vendedor del producto, es una manera idónea de poner en marcha un proyecto de Data Warehousing/BI con la disminución de costes y riesgos. Las soluciones analíticas a menudo poseen herramientas que asisten al usuario en la extracción y puesta a punto de los datos, así también en el análisis y la interpretación de los mismos. Es necesario conformar dimensiones

en todo el entorno al Data Warehouse/BI, independientemente de si se construye una solución o se implementan paquetes pre enlatados. Los paquetes de soluciones analíticas pueden impulsar la implementación de una solución DW/BI, sin embargo, no alivian la necesidad de conformidad. La mayoría de las organizaciones se basan en una combinación de la construcción, compra, e integración de una solución completa.

En el siguiente capítulo se aborda el concepto de OLAP.

Bibliografía

Tederman I (2001) Data Warehousing for E-Business. John Wiley & Sons, Inc. New York, NY, USA.

Kimball,R., Ross, M – Nachdr (2002) The data warehouse toolkit: the complete guide to dimensional modelling. Ed. Wiley.

Kimball, R. Ross, M (2013) The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3° Edition. Ed. Wiley.

Christiansen, T. (2012) Thoughts on Data Vault vs. Star Schema

<https://thebibbackend.wordpress.com/2012/06/05/thoughts-on-data-vault-vs-star-schemas/>

Data Vault. Más discusiones: <http://forum.kimballgroup.com/t362-data-vault-v-s-dimensional-model>

Data Vault. Más material y links de interés en <http://danlinstedt.com/about/data-vault-basics/> y <http://es.slideshare.net/kgraziano/introduction-to-data-vault-modeling>

<http://tdan.com/data-vault-series-2-data-vault-components/5155#thumb>

Libro de Dan Linstedt (Building a Scalable Data Warehouse with Data Vault 2.0: Implementation Guide ...) <http://tdan.com/data-vault-series-2-data-vault-components/5155#thumb>

Data Vault Basics, 2015. Linstedt, Dan. <http://danlinstedt.com/>

Data Integration - ETL evolves into Data Integration

https://gerardnico.com/wiki/data/processing/etl_become_di

CAPÍTULO 4

Análisis OLAP y Reporting

El análisis OLAP (On-line Analytical Processing) es una de las herramientas incluidas dentro de las tecnologías de Business Intelligence para el análisis de los datos por parte del usuario final. Frecuentemente involucra análisis de tendencias, comparaciones de períodos, y navegación de datos.

Los sistemas OLTP (On-line Transactional Processing) son diseñados para soportar el procesamiento de información diaria de las empresas, y el énfasis recae en maximizar la capacidad transaccional de sus datos. Su estructura es altamente normalizada, para brindar mayor eficiencia a sistemas con muchas transacciones que acceden a un pequeño número de registros y están fuertemente condicionadas por los procesos operacionales que deben soportar, para la óptima actualización de sus datos. Esta estructura es ideal para llevar a cabo el proceso transaccional diario, brindar consultas sobre los datos cargados y tomar decisiones diarias. En cambio los esquemas de DW están diseñados para poder llevar a cabo procesos de consulta y análisis para luego tomar decisiones estratégicas y tácticas de alto nivel. En los capítulos anteriores se han mencionado las diferencias, que resumiremos en la siguiente tabla:

	OLTP Online Transaction Processing	OLAP Online Analyst Processing
Objetivos	Asistir aplicaciones específicas Mantener la integridad de los datos	Comparaciones. Identificar tendencias. Almacenamiento de datos históricos.
Perfil del Usuario	Operativo Entrada de Datos Consultas Puntuales Tareas repetitivas	Gerencial. Toma de decisiones. Consultas globales.
Datos	Alineados por aplicación No integrados y no históricos	Alineados por dimensión. Integrados e históricos.
Acceso y Manipulación	Muchas y pequeñas transacciones, performantes, ABMs, Consistencia.	Pocas transacciones con muchos registros. Picos eventuales y carga masiva.

Fig 4.1 - OLTP vs OLAP

La siguiente es una definición formal de OLAP presentada por Curto Díaz :

“método ágil y flexible para organizar datos, especialmente metadatos sobre un objeto o jerarquías de objetos como en un sistema u organización multidimensional, cuyo objetivo es recuperar y manipular datos y combinaciones a través de consultas o incluso informes.“

Los sistemas OLAP facilitan el análisis multidimensional a partir de matrices o tablas pivotantes, que se implementan a través de un motor y una herramienta de visualización.

El concepto data de 1993 y en las últimas décadas es ampliamente utilizado y prácticamente todos los motores incluyen la tecnología OLAP.

El modelo lógico de los datos para OLAP debe seguir el diseño multidimensional. Sin embargo la tecnología de la base de datos donde se implementa puede ser multidimensional o relacional, dando origen a tres categorías de productos OLAP: MOLAP, ROLAP y HOLAP que se detallan a continuación.

Tipos de Motor OLAP

Los cubos multidimensionales representan la información plana de filas y columnas en estructuras multidimensionales o hipercubos.

- ROLAP: Como se aclaró anteriormente, el almacén de datos se organiza a través de una base de datos multidimensional, sin embargo, puede ser soportado por un SGBD Relacional. Para lograr esto se utilizan los diferentes esquemas, en estrella, copo de nieve y constelación, los cuales transformarán el modelo multidimensional y permitirán que pueda ser gestionado por un SGDB Relacional, ya que solo se almacenarán tablas. Se utiliza cache para optimizar el tiempo de respuesta. No requiere de otro software adicional para gestionar el cubo. Utiliza índices de mapas de bits. Utiliza índices de Join. Posee optimizadores de consultas. Cuenta con extensiones de SQL (drill-up, drill-down, etc).
- MOLAP: Refiere a una tecnología de bases de datos multidimensional y propietaria, también denominada cubo de datos. La representación externa e interna coinciden y la performance es la característica principal. La información se almacena en estructuras multidimensionales específicas, denominadas Arrays. Disponen de técnicas de compactación que favorecen el rendimiento. Tanto los datos detallados (o fuente) como los datos agregados o precalculados residen en el mismo formato multidimensional. Optimiza las consultas, pero suele requerir más espacio de disco y diferente software. MOLAP requiere que en una instancia previa se generen y calculen los cubos multidimensionales, para que luego puedan ser consultados. Este proceso se puede resumir a través de los siguientes pasos:
 - Se seleccionan los indicadores, atributos, jerarquías, etc., que compondrán el cubo multidimensional.

- Se precisan los datos del cubo.
- Se ejecutan las consultas sobre los datos precalculados del cubo.

Precalcular los datos de los cubos multidimensionales favorece los tiempos de respuesta y le quita flexibilidad dado que en caso de realizar cambios sobre algún cubo se debe tener que recalcularlo totalmente para que se reflejen las modificaciones llevadas a cabo, provocando de esta manera una disminución importante en cuanto a flexibilidad. Se precisa más espacio físico para almacenar los datos (esta desventaja no es tan significativa) y tiene limitaciones en cuanto al tamaño máximo de los cubos.

- HOLAP y OLAP Híbrido: Toma lo mejor de los anteriores. Implementa MOLAP para la confección de tableros y ROLAP para navegar y explorar los datos, brindando de esta manera flexibilidad y altos tiempos de respuesta. Es imprescindible realizar un buen análisis para identificar los diferentes tipos de datos.
- Estructuras in-memory: Brinda mayor rapidez de consultas, reportes y análisis, flexibilidad para generar prototipos hacen que esta arquitectura sea una de las más relevantes de la arquitectura OLAP. Brinda los siguientes enfoques:

In-memory Analytic	Ventajas	Desventajas
MOLAP Classic	Resuelve las consultas rápido, se pueden introducir valores en los reportes y usarlos	Modelo dimensional tradicional Limitado al espacio de memoria física
In memory Metadata ROLAP	Metadatos en memoria entonces tiene mayor velocidad en las consultas. No tiene problemas de memoria	Modelo dimensional tradicional Solo los metadatos están en memoria.
Índice invertido in-memory: índice con datos que se cargan en memoria	Rápida generación de reportes y consultas. Requiere menos modelamiento	Limitado por la memoria Requiere modelamiento de los índices
Índice asociativo in-memory: arreglo/ vector/ índice con cada entidad/atributo correlacionando con cualquier otra entidad/atributo.	Rápida generación de reportes y consultas. Requiere menos modelamiento. La presentación de informes y consultas es más rápida	Limitado por la memoria física. Algunas secuencias de comandos/modelamiento son necesarias
Hoja de cálculo in-memory	Los reportes o análisis son tan sencillos como ordenar y filtrar	Limitado por la memoria física

Tabla 4.1 – OLTP in-memory

Los productos comerciales y open source soportan alguna de todas las características mencionadas previamente. El siguiente cuadro comparativo tomado de Wikipedia, toma las características de cada uno y los productos comerciales que le dan soporte. Es interesante analizar también las nuevas características de las herramientas como el soporte de acceso a través de APIs, técnicas de procesamiento de datos y técnicas de modelado.

OLAP Server	MOLAP	ROLAP	HOLAP	In-Memory	Offline
IBM Cognos TM1	Yes	No	No	Yes	Cognos Insight Distributed mode
Essbase	Yes	No	No	No	
IcCube	Yes	No	No		Offline Cubes
Infor BI OLAP Server	Yes	No	No	Yes	Local cubes
Jedox OLAP Server	Yes	No	No	Yes	No
Microsoft Analysis Services	Yes	Yes	Yes	Yes	Local cubes, PowerPivot for Excel, Power BI Desktop
MicroStrategy Intelligence Server	Yes	Yes	Yes	No	MicroStrategy Office, Dynamic Dashboards
Mondrian OLAP server	No	Yes	No	No	
Oracle Database OLAP Option	No	Yes	No	No	
SAS OLAP Server	Yes	Yes	Yes	No	
IBM Cognos BI	Yes	Yes	Yes	No	
SAP NetWeaver BW	Yes	Yes	No	No	
Cubes (OLAP server)	No	Yes	No		
Druid	Yes	Yes	Yes	Yes	Yes

Tabla 4.1 - [Comparativa de productos OLAP según tipo de almacenamiento](#)

Visualización OLAP. MDX

La herramienta de visualización OLAP permite al usuario final realizar consultas, reordenar y filtrar los datos de una estructura multidimensional para la toma de decisiones.

Se dice que navegan la información, realizando cortes (Slicing & Dicing) y subiendo y bajando por jerarquías de las dimensiones (Drill Up / Drill Down).

Por ejemplo, en la siguiente imagen se ve un problema en términos de distintas vistas, componentes o dimensiones. Los usuarios toman cualquier componente para analizar, una o varias a la vez.

	05/10/2016			
Producto	Región			
	Argentina	Brasil	Uruguay	
	Hidrocarburo	345	45	6
	Minería	123	12	78
	Software	23	5	765
	Manufactura	45	456	54
Vitivinícola	567	67	5	

Reporte diario

	2017 - Junio			
Producto	Región			
	Argentina	Brasil	Uruguay	
	Hidrocarburo	1345	745	67
	Minería	1230	1265	784
	Software	235	55	7651
	Manufactura	645	4562	542
Vitivinícola	5867	678	51	

	2017 - Julio			
Producto	Región			
	Argentina	Brasil	Uruguay	
	Hidrocarburo	1345	745	67
	Minería	1230	1265	784
	Software	235	55	7651
	Manufactura	645	4562	542
Vitivinícola	5867	678	51	

	2017 - Agosto			
Producto	Región			
	Argentina	Brasil	Uruguay	
	Hidrocarburo	1345	745	67
	Minería	1230	1265	784
	Software	235	55	7651
	Manufactura	645	4562	542
Vitivinícola	5867	678	51	

Reporte Sumarizado

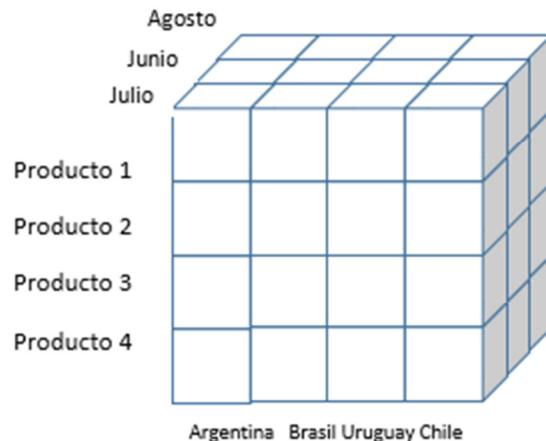


Fig. 4.2 - Cubo multidimensional

Dicing / Slicing

El gerente de ventas de Argentina puede estar interesado en analizar el comportamiento de las ventas en su país para todos los años, ídem para el gerente de Brasil y los otros países, como se puede observar en la figura 4.2. Este tipo de navegación se denomina **slicing** donde se selecciona un único valor para una de sus dimensiones, creando un nuevo cubo con una sola dimensión, como una rebanada del cubo original.

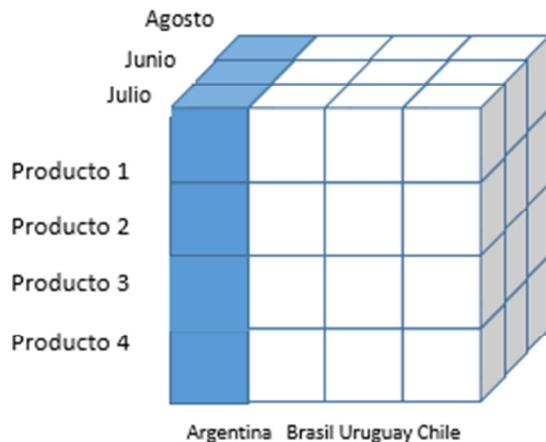


Fig. 4.3 - Cubo multidimensional - Navegación slicing

La navegación de tipo **dicing** produce un sub cubo para facilitar el análisis de múltiples dimensiones. La navegación se realiza tomando varios miembros de distintas dimensiones. Por ejemplo para analizar las ventas de un país en un período determinado.

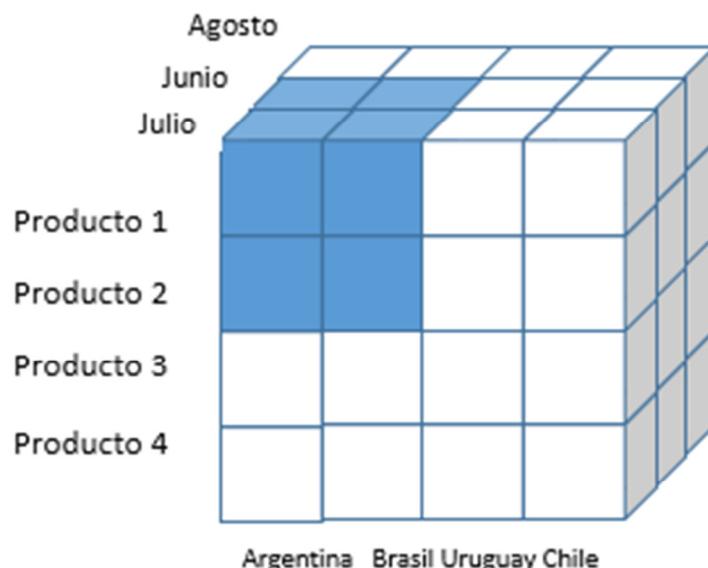


Fig. 4.4 - Cubo multidimensional - Navegación dicing

Drill Down / Drill Up

Con estas operaciones se puede navegar en las jerarquías de las dimensiones.

Con la navegación Drill Down se desciende por la jerarquía de niveles de la dimensión, yendo de lo general a lo particular, obteniendo así mayor detalle.

Por ejemplo, si se visualiza una celda con las ventas de todos los vehículos en el año actual, se puede hacer un “drill down” sobre la dimensión vehículo para conocer el monto discriminado por tipo: Moto, Auto, Pickup, Camión, etc

La operación “drill up”, tiene el efecto contrario, va de lo particular a lo general, resumiendo la información. Por ejemplo, si se visualiza la información de ventas de Junio, se puede hacer un “drill up” sobre la dimensión tiempo para visualizar las ventas de todo el año.

Pivot

La navegación pivot permite rotar el cubo en el espacio para analizarlo desde diferentes perspectivas.

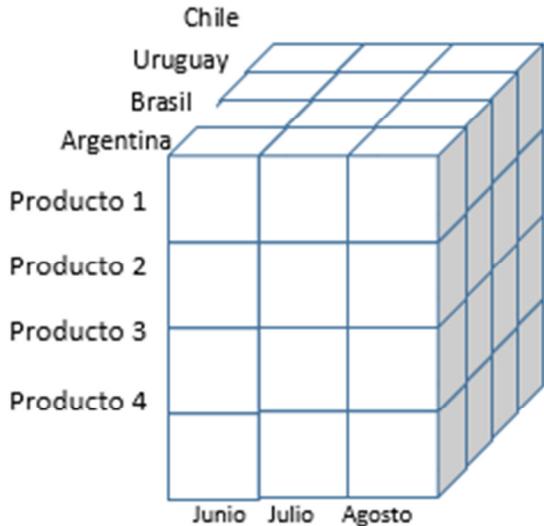


Fig. 4.6 - Cubo multidimensional - Navegación pivot

MDX

Las operaciones se traducen a consultas MDX (Multidimensional eXpression) que trabajan sobre la estructura dimensional.

Así como SQL es el lenguaje de consulta de las bases de datos relacionales, MDX lo es para las bases multidimensionales.

Mientras SQL recupera los datos de la consulta en 2 dimensiones (formato de tabla, filas y columnas), MDX lo hace en N dimensiones (D1=filas, D2=columnas, D3, paginas, D4,...,DN)

Para entender la potencia de este tipo de herramientas, pensemos en un cubo como el de la figura 4.1 La sintaxis de MDX tiene el siguiente formato:

```
SELECT {[Measures].[Cantidad Vendida], [Measures].[Monto]} ON COLUMNS,
{[Productos].members} ON ROWS
FROM [Ventas]
WHERE [Tiempo].[2017].[01]
```

En este caso, visualiza dos columnas (“cantidad vendida” y “monto”), agrupado por “Producto” del cubo “Ventas”, filtrando solo aquellos en el periodo Enero 2017.

MDX fue creado en 1997 por Microsoft, pero la mayoría de los fabricantes de OLAP lo incorporan, convirtiéndose en un estándar de facto de la industria. Sin embargo, cada implementación agrega sus propias extensiones.

Las últimas versiones de Microsoft Excel incluye componentes para poder utilizarlo. También los navegadores de cubos como Saiku de Pentaho, brindan accesos directos a las consultas MDX.

The screenshot shows the Saiku Analytical Engine interface. On the left, there are navigation panels for 'Cubos' (Olympics) and 'Dimensiones' (Age, Athlete, Date). The main area displays an MDX query:

```

1 SELECT
2 NON EMPTY {Hierarchize({[Athlete].[Country].Members})} ON COLUMNS,
3 NON EMPTY {Hierarchize({[Measures].[Sum of Gold medals]})} ON ROWS
4 FROM [olympics]
    
```

A circled button labeled 'Cambiar a modo MDX' is visible. Below the query, a table shows the sum of gold medals for various countries:

MeasuredLevel	Afghanistan	Algeria	Argentina	Armenia	Australia	Austria	Azerbaijan	Bahamas	Bahrain
Sum of Gold Medals	0	2	49	0	163	26	6	11	0

Fig 4.7 - Visualizar consulta MDX

Se puede obtener una documentación completa en las siguientes direcciones:

<https://docs.microsoft.com/en-us/sql/analysis-services/multidimensional-models/mdx/querying-multidimensional-data-with-mdx>

<https://mondrian.pentaho.com/documentation/mdx.php>

Mondrian

Mondrian es un motor OLAP de tipo ROLAP con cache, que forma parte de la suite Pentaho.

Es open source escrito en JAVA y ejecuta consultas MDX leyendo los datos de una base de datos relacional.

Su arquitectura está organizada en 4 capas: dimensional, estrella y almacenamiento

- Capa de presentación: la capa de presentación, se encarga de lo que se presenta al usuario y como este puede interactuar con ella. Esto incluye cubos naveables, gráficos, etc.
- Capa dimensional: La capa dimensional se encarga de ejecutar las consultas MDX. Puede llevarlas a cabo en distintas etapas, evaluando primero los ejes y luego las celdas, valiéndose de datos precargado en caché de la capa inferior.
- Capa Estrella: La capa estrella se encarga de mantener un caché para optimizar las consultas y solicitar los datos a la capa de almacenamiento.
- Capa de almacenamiento: La capa de almacenamiento es encargada de consultar el motor relacional.

En el siguiente gráfico puede visualizarse su arquitectura según su documentación:

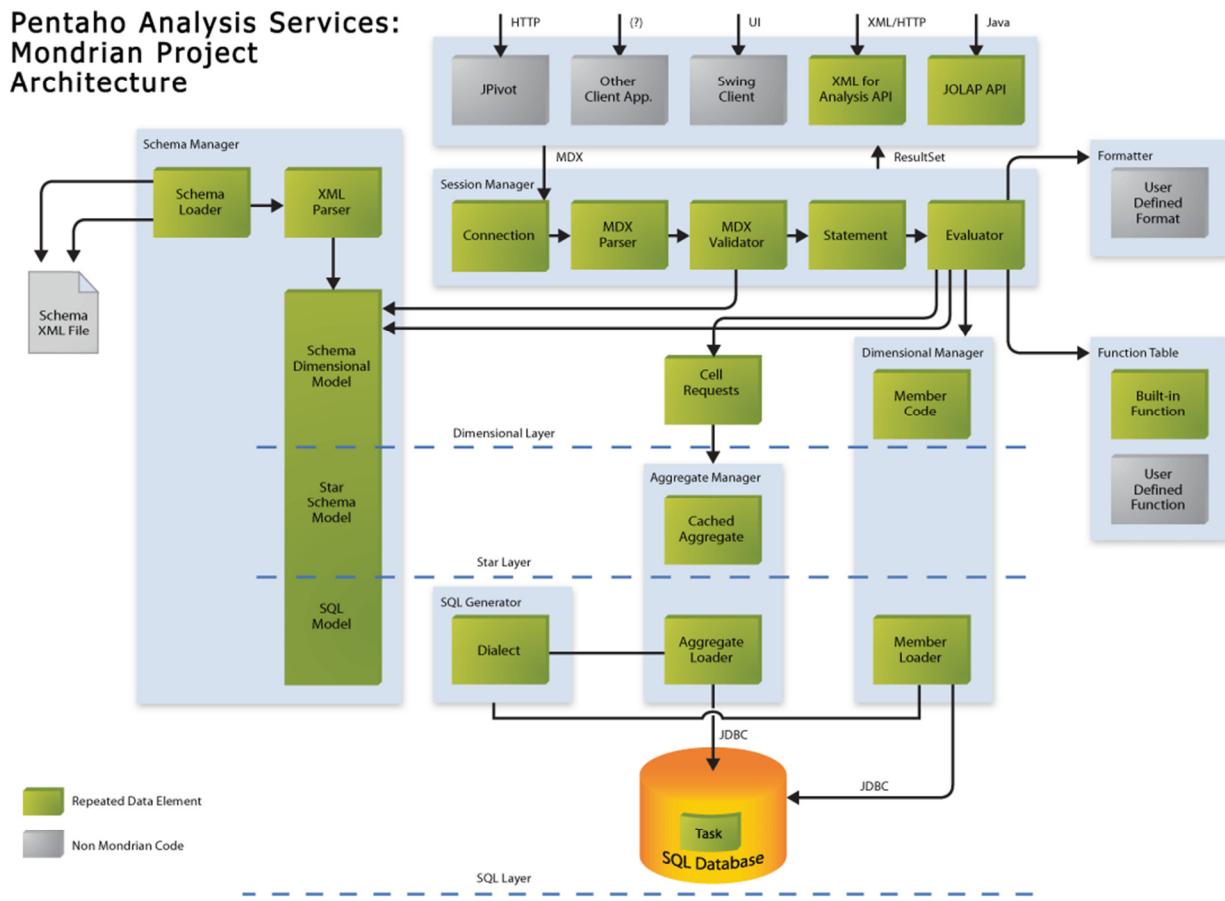


Fig. 4.8 - Mondrian

Mondrian requiere un esquema donde se explice los cubos con sus respectivas dimensiones y medidas. Este esquema no es más que un archivo XML, que relaciona las tablas y columnas del motor relacional a las dimensiones y medidas correspondientes del modelo dimensional.

La ventaja de este esquema es que si uno quisiera podría armar un modelo dimensional sobre una base de datos transaccional sin alterar la base de datos, es decir, sin necesidad de crear tablas o vistas particulares, dado que es posible invocar consultas SQL directamente desde el esquema.

El esquema puede escribirse con un editor de textos, o bien utilizar una herramienta provista por la suite Pentaho llamada “Schema Workbench”

Pentaho Schema Workbench

Esta herramienta permite editar visualmente el archivo xml que contiene la definición del esquema dimensional.

Se la puede descargar gratuitamente desde el sitio comunidad de Pentaho

<https://sourceforge.net/projects/mondrian/files/schema%20workbench/>

Un esquema dimensional cuenta con las siguientes estructuras básicas:

- Dimensiones compartidas:
Cada dimensión puede tener 1 o más jerarquías, organizadas en niveles
- Cubos
Cada cubo puede usar dimensiones compartidas o contener dimensiones propias, debe contener 1 o más dimensiones y 1 o más medidas.
- Cubos virtuales
Un cubo virtual es una conjunción de cubos.

Schema Workbench es multiplataforma y no requiere instalación, pero necesita contar con una JVM de Java instalada en el sistema. También requiere, previamente a iniciarla ,que se copie el driver jdbc del motor de base de datos que se desea utilizar a su carpeta “lib”.

Se inicia con el comando “./workbench.sh” (en Linux) o “workbench.bat” (En Windows)

Esta es su ventana principal:

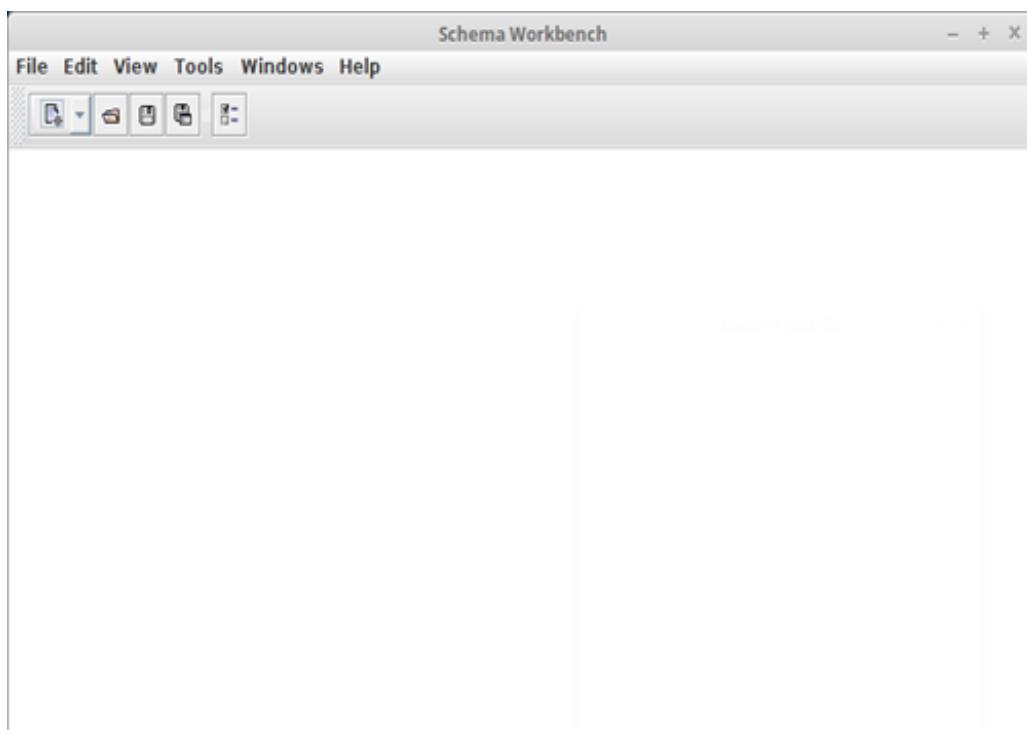


Fig. 4.9 - Pentaho Workbench

La primera vez que se inicia, es necesario definir la conexión a la base de datos.

Se requiere que el driver jdbc de la base de datos se encuentre previamente en la carpeta “./lib”.

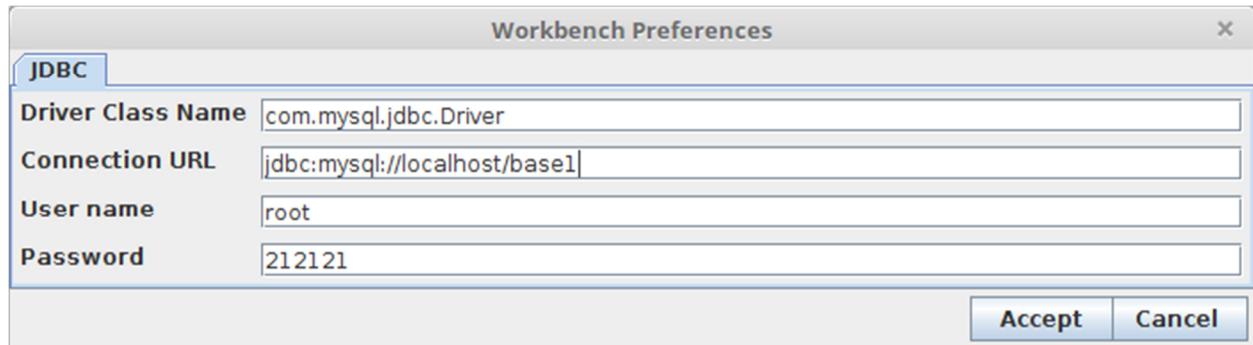


Fig. 4.10 - Pentaho Workbench. Configuración de JDBC

Se debe agregar al menos un cubo, y este debe contener al menos una dimensión y al menos una medida.

Cada dimensión debe contener también al menos una jerarquía con al menos un nivel.

Un esquema mínimo deberá contener las siguientes entidades y los siguientes atributos :

- < Esquema > (name)
 - < Cubo > (name)
 - < Tabla > (name)
 - < Dimension > (name, foreign-key)
 - < Jerarquia > (primary-key)
 - < Tabla > (name)
 - < Nivel > (name, column)
 - < Medida > (name, column, aggregator)

La jerarquía anterior representa un esquema con un cubo, que contiene una dimensión con un único nivel y una sola medida.

Las dimensiones compartidas (aquellas que se utilizan en más de un cubo), se definen antes de los cubos.

A continuación se presenta un esquema con 1 dimensión compartida y 2 cubos donde se utiliza

- < Esquema > (name)
 - < Dimension > (name, foreign-key)
 - < Jerarquia > (primary-key)
 - < Tabla > (name)
 - < Nivel > (name, column)
 - < Cubo1 > (name)
 - < Tabla > (name)
 - < DimensionUsage > (name, source, foreign-key)
 - < Medida > (name, column, aggregator)

- < Cubo2 > (name)
 - < Tabla > (name)
 - < DimensionUsage > (name, source, foreign-key)
 - < Medida > (name, column, aggregator)

Ejemplo

Supongamos que tenemos una base de datos con información de proyectos y personas que participan en los mismos. Se necesita poder analizar los proyectos, identificando la tipificación de los mismos, los períodos en que se crearon, el presupuesto asignado, etc. También se necesita analizar información de las personas. edad, proyecto asociado, género, etc.

Con este caso práctico, vamos a definir un esquema con dos cubos. Uno enfocado en los proyectos y otro en las personas. A continuación se irá estableciendo paso a paso el esquema en formato XML. En cada paso se comentará el código añadido, hasta llegar al final de la definición.

Definición principal del esquema

```
<Schema name="proyectos_y_personas">
```

Se definen las dimensiones compartidas

```
<Dimension name="Tiempo">
  <Hierarchy hasAll="true" primaryKey="id">
    <table name="d_tiempo"/>
    <Level name="Año" column="anio" type="Integer"></Level>
    <Level name="Mes" column="mes" type="Integer"></Level>
    <Level name="Dia" column="dia" type="Integer"></Level>
  </Hierarchy>
</Dimension>

<Dimension name="Proyecto">
  <Hierarchy hasAll="true" primaryKey="id">
    <table name="d_tipo"/>
    <Level name="Tipo" column="tipo" type="String"></Level>
    <Level name="Subtipo" column="subtipo" type="String"></Level>
    <Level name="Proyecto" column="proyecto" type="String"></Level>
  </Hierarchy>
</Dimension>
```

Luego se define el primer cubo. Cuenta con 2 dimensiones compartidas que se definieron previamente, y dos medidas : “Cantidad” y “Presupuesto”.

La función de agregación de “Cantidad” es “distinct-count”, es decir que cuando se haga un “drill up”, o sea, cuando se resuma información, de los registros que involucra va a contar la cantidad de valores distintos de la columna “proyecto_id”

Mientras que la función de agregación de “Presupuesto” suma los valores de la columna “presupuesto” para los registros que incluye el resumen.

```
<Cube name="proyectos">

    <!-- Dimensiones Compartidas -->
    <DimensionUsage name="Fecha de creacion" source="Tiempo" foreignKey="fecha_ini"/>
    <DimensionUsage name="Proyecto" source="Proyecto" foreignKey="proyecto_id"/>

    <!-- Medidas -->
    <Measure name="Cantidad" column="proyecto_id" aggregator="distinct-count" />
    <Measure name="Presupuesto" column="presupuesto" aggregator="sum" />

</cube>
```

También se especifica el cubo de personas, con solo una dimensión y dos medidas

```
<Cube name="personas">

    <!-- Dimensiones Compartidas -->
    <DimensionUsage name="Proyecto" source="Proyecto" foreignKey="proyecto_id"/>

    <!-- Medidas -->
    <Measure name="Cantidad" column="persona_id" aggregator="distinct-count" />
    <Measure name="Edad" column="edad" aggregator="avg" />

</cube>
```

Finalmente, se pueden agregar roles de visualización.

Estos roles pueden asignar o no permisos al cubo a ciertos cubos, o incluso a ciertas dimensiones. Un caso común, es cuando se dan permisos por región, de manera que cada responsable solo puede visualizar datos de su región,

En este caso, vamos a dar permiso a todos los cubos a quienes tengan el rol “personal”

```
<Role name="personal">
    <SchemaGrant access="all">
        </SchemaGrant>
    </Role>
```

Al final, va la etiqueta de cierre del esquema

```
</Schema>
```

Reporting

Los informes operativos constituyen el punto inicial de una herramienta de inteligencia de negocios en el contexto de una organización. La información que se genera de la tarea diaria crece de forma exponencial, que se almacena en bases de datos o en archivos con diferentes formatos.

Es importante contar con reportes para los distintos niveles operativos, mandos intermedios y estratégicos para conocer el estado de la organización que se construyan con distintas fuentes y distintos tipos de datos. Sin embargo, la generación de informes impacta en forma significativa en la operatoria diaria.

Las herramientas para generar informes no son novedosas y existen en el mercado soluciones maduras que brindan esta funcionalidad, incluso algunas permiten al usuario independizarse del área de TI para obtener sus propias reportes. Las últimas tendencias hacen mucho hincapié en las visualizaciones atractivas y versátiles, así como también brindar posibilidades para exportarlos en diferentes formatos como csv, HTML, PDF, MS Excel, entre otros.

Los informes intentan dar respuesta a ¿qué pasó? Dado que es básicamente la primer pregunta que se realiza al iniciar un análisis con los datos existentes, y todas las soluciones de BI incluye un motor de reporting. Suelen contener texto y gráficos que faciliten el análisis, así como también tablas, mapas para información geolocalizada, métricas que visualizan el rendimiento del negocio, alertas visuales o automáticas que estén incluidas como en un cuadro de mando. Sus usuarios necesitan información sumarizada según dis-

tintos criterios. Una herramienta de reporting permite diseñar y gestionar (planificar, administrar y distribuir) los informes en el contexto de una organización.

Los informes incluyen distintos tipos de métricas como indicadores de rendimiento (KPI) para analizar un proceso por ejemplo, o los indicadores de metas.

Al realizar un informe, uno de los puntos más difíciles es la selección del tipo de gráfico. [Extreme Presentation](#) brinda un método para diseñar presentaciones de datos complejos, que sean efectivas y conduzcan rápidamente a la acción. Brinda muchísimo material y en particular este diagrama en función de la información disponible para analizar.

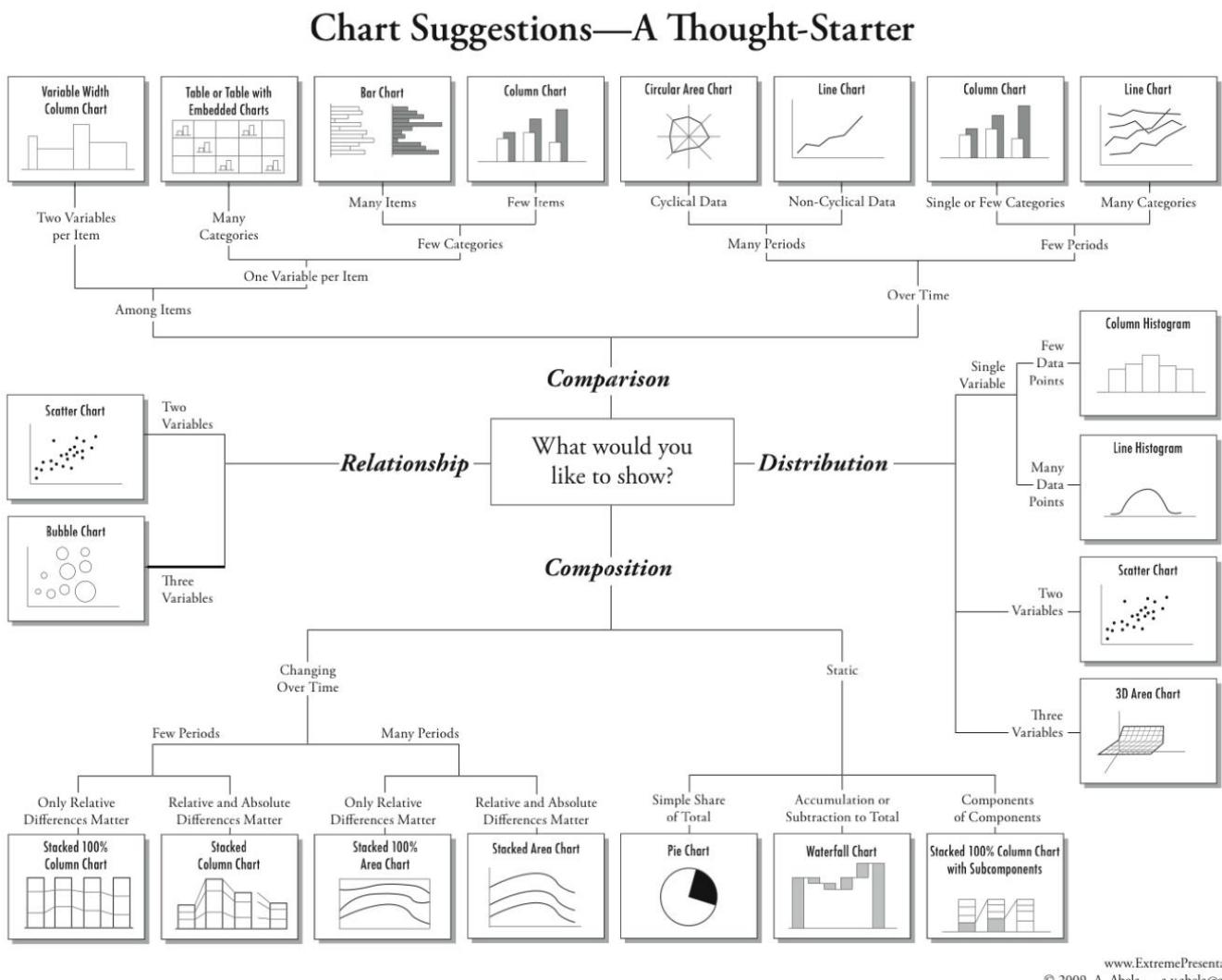


Fig. 4.10 - Sugerencias de uso de los distintos tipos gráficos

La suite de Pentaho open source brinda una herramienta de reporting denominada Report Designer. Ofrece la posibilidad de conectarse a múltiples fuentes de datos, incluso transformaciones generadas en Kettle. La herramienta está formada por tres componentes principales:

- Report Designer: aplicación de escritorio que provee un entorno visual para crear reportes sofisticados.

- Report Engine: librería codificada en Java para hacer reportes, puede ser usada server-side o client-side.
- Reporting SDK: para embeber los reportes de Pentaho en la aplicación.

Vendor	Bar Code	SKU	Name	Scale	On Hand	Cost	MSRP
AUTOART STUDIO DESIGN		S12_1099	1968 Ford Mustang	1:12	68 units	\$ 95	\$ 195
Description: Hood, doors and trunk all open to reveal highly detailed interior features. Steering wheel actually turns the front wheels. Color dark green.							
68 units							
CAROUSEL DIECAST LEGENDS		S24_1628	1966 Shelby Cobra 427 S/C	1:24	8.197 units	\$ 29	\$ 50
Description: This diecast model of the 1966 Shelby Cobra 427 S/C includes many authentic details and operating parts. The 1:24 scale model of this iconic lightweight sports car from the 1960s comes in silver and it's own display case.							
8.197 units							
		S24_2840	1958 Chevy Corvette Limited Edition	1:24	2.542 units	\$ 16	\$ 35
Description: The operating parts of this 1958 Chevy Corvette Limited Edition are particularly delicate due to their precise scale and require special care and attention. Features rotating wheels, working steering, opening doors and trunk. Color dark green.							
2.542 units							
		S700_2824	1982 Camaro Z28	1:18	6.934 units	\$ 47	\$ 101
Description: Features include opening and closing doors. Color: White. Measures approximately 9 1/2 Long.							
6.934 units							
CLASSIC METAL CREATIONS		S10_1949	1952 Alpine Renault 1300	1:10	7.305 units	\$ 99	\$ 214

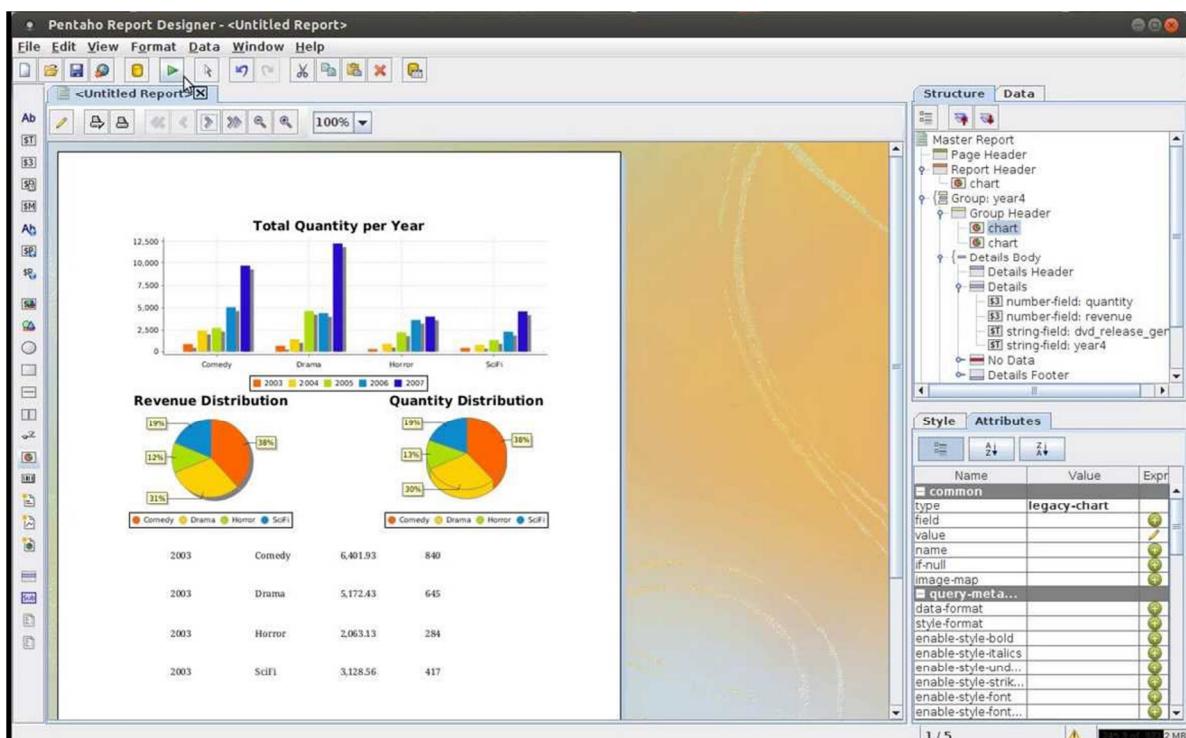


Fig. 4.11 - Ejemplos de Report Designer

Como mencionamos previamente, en el mercado de productos open source de reporting es muy amplio y maduro. El siguiente [link de Wikipedia](#) mantiene una lista actualizada de herramientas open source y propietarias para la construcción de reportes.

[BIRT Project](#)

[D3.js](#)

[JasperReports](#)

[KNIME](#)

[LibreOffice Base](#)

[OpenOffice Base](#)

[Pentaho](#)

[SpagoBI](#)

Cada una de ellas brinda mayor o menor capacidad de análisis y conocimientos técnicos para sacar el máximo provecho de ellas.

La analítica de los datos requiere de herramientas cada vez más versátiles que permitan realizar análisis en línea, gestionar las distintas variables y tomar decisiones a partir de ellas.

CAPÍTULO 5

Introducción a Big Data & Ciencia de los Datos

Cuando hablamos de Big Data nos referimos a conjuntos de datos o combinaciones de conjuntos de datos cuyo tamaño (volumen), complejidad (variabilidad) y velocidad de crecimiento (velocidad) dificultan su captura, gestión, procesamiento o análisis mediante tecnologías y herramientas convencionales, dentro del tiempo necesario para que sean útiles. Las 3Vs tradicionales se siguen extendiendo a Veracidad (que los datos procesados sean certeros) y Valor (retorno de la inversión). Las soluciones de Big Data suelen ser costosas, ya sea que las empresas cuentan con equipamiento propio o requieren de computación en la nube para su procesamiento. Esto implica que el retorno de inversión debe ser considerado en cualquier proyecto de estas características. La analítica de los datos pasó de ser una innovación a ser una necesidad imperativa. Según el informe de IBM del 2014, "[Storwize family delivers data ready infrastructure. An ideal storage solution for business analytics](#)" los CIOs ubican a la analítica de datos como la primera entre las estrategias prioritarias a llevar adelante en una empresa.

Aunque el tamaño utilizado para determinar si un conjunto de datos determinado se considera Big Data no está firmemente definido y sigue cambiando con el tiempo, la mayoría de los analistas y profesionales actualmente se refieren a conjuntos de datos que van desde 30-50 terabytes a zettabytes. Según el diccionario de IT de Gartner, Big Data is "is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation." El informe de IBM hace referencia al explosivo crecimiento de los datos en los últimos años, que se espera que alcance al menos los 40 zettabytes en el 2020, 300 veces más que los niveles del 2005.



Fig. 5.1 – Crecimiento de los datos.

La naturaleza compleja de Big Data se debe principalmente a la naturaleza no estructurada de gran parte de los datos generados por las tecnologías modernas, como los weblogs, la identificación por radiofrecuencia (RFID), los sensores incorporados en dispositivos, la maquinaria, los vehículos, las búsquedas en Internet, las redes sociales como Facebook, computadoras portátiles, teléfonos inteligentes y otros teléfonos móviles, dispositivos GPS y registros de centros de llamadas.

Por ejemplo, en la siguiente imagen tomada de Wikipedia representa la actividad de edición de un usuario, en este caso el bot Pearle, en el marco de un proyecto donde el objetivo fue analizar, en un espacio colaborativo, como los usuarios realizan la edición de los documentos a fin de identificar patrones y tendencias por ejemplo para sugerir automatizar algunas tareas o identificar ataques informáticos sobre la plataforma.

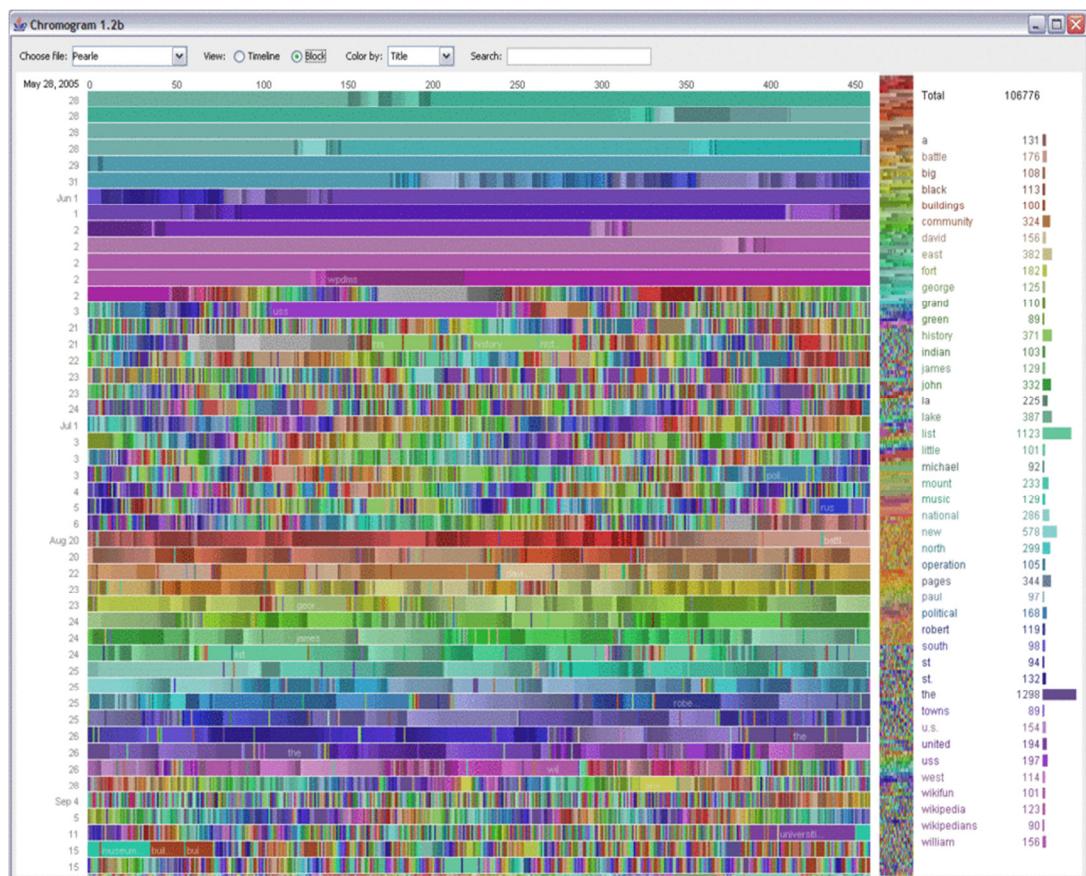


Fig. 5.2 - Actividad de un usuario en Wikipedia. By Fernanda B. Viégas

Lo que hace que Big Data sea tan útil es el hecho de que proporciona respuestas a muchas preguntas que las organizaciones ni siquiera sabían que tenían. En otras palabras, proporciona un punto de referencia. Con una cantidad tan grande de información, los datos pueden ser moldeados o probados de cualquier manera que la empresa considere adecuada. Al hacerlo, las organizaciones son capaces de identificar los problemas de una forma más comprensible.

El análisis de Big Data ayuda a las organizaciones a aprovechar sus datos y utilizarlos para identificar nuevas oportunidades. Eso, a su vez, conduce a movimientos de negocios más inteligentes, operaciones más eficientes, mayores ganancias y clientes más felices.

Muchas fuentes y tipos de datos

Con tantas fuentes, tipos de datos y estructuras complejas, la dificultad de integración de datos aumenta. Las fuentes de datos de big data son muy amplias:

- Datos de internet y móviles.
- Datos de Internet de las Cosas.
- Datos sectoriales recopilados por empresas especializadas.
- Datos experimentales.

Y los tipos de datos involucrados también lo son, desde datos estructurados en bases de datos hasta datos no estructurados de documentos, videos y audios, semi-estructurados como software, hojas de cálculo, informes. Solo el 20% de información es estructurada y eso puede provocar muchos errores si no tenemos un proyecto de calidad de datos.

La siguiente infografía desarrollada por Lori Lewis & Chadd Callahan of [Cumulus Media](#), que realiza cada año, se presenta la increíble escala de volumen de datos de e-commerce, social media, email, y otras fuentes de generación de contenidos que ocurren en la Web en 1 minuto!



Fig. 5.3 - Generación de datos de Internet en 1 minuto

O los siguientes datos tomados de WorldMeters ilustran este crecimiento impresionante de datos a través de Internet.

SOCIETY & MEDIA	
2,249,453	New book titles published this year [+]
204,338,125	Newspapers circulated today [+]
277,738	TV sets sold worldwide today [+]
2,450,494	Cellular phones sold today [+]
\$ 90,683,216	Money spent on videogames today [+]
3,774,170,649	Internet users in the world today [+]
99,420,597,035	Emails sent today [+]
2,161,727	Blog posts written today [+]
283,365,362	Tweets sent today [+]
2,302,562,147	Google searches today [+]

Fig. 5.4. <http://www.worldometers.info/> consultado el 13 de noviembre de 2017

Big Data requiere Big Analytics

Estamos acostumbrados a saber lo que pasó, pero hoy nos interesa más conocer lo que pasará: si la organización, con su producto o servicio, seguirá siendo el gusto de los consumidores o si se hablará de ella bien o mal en las redes sociales .Todos estos aspectos requieren de nuevos modelos de análisis mucho más complejos que lo que se podía experimentar hasta ahora.

Esta nueva realidad ha motivado nuevos requerimientos por parte de las organizaciones en relación con el análisis de datos. Y, precisamente, para poder analizar toda esa información de que hoy se dispone, requiere de un nuevo modelo de análisis: Big analytics, el único que permite dar forma al Big Data.

Esta necesaria aparición tiene que ver con el hecho de que, para poder hacer un análisis predictivo o una “clusterización” de los perfiles de clientes, por ejemplo, se requiere de algo más que la propia inteligencia de negocio, y ese algo más se concreta precisamente en Big Analytics.

Analytics permite la aplicación de procesos matemáticos complejos. Sus técnicas hacen posible trabajar con los datos al nivel de granularidad más bajo disponible, en crudo, y empleando para ello modelos mucho más ágiles que los actuales modelos de BI. Por esto han surgido un conjunto de herramientas y tecnologías que hacen al big data landscape, que se pueden caracterizar en:

- Infraestructura
 - Analítica de datos
 - Aplicaciones
 - Cross-Infrastructure/Analytics
 - Open Source
 - Fuentes de Datos & APIs

Y se ilustran en la siguiente infografía.

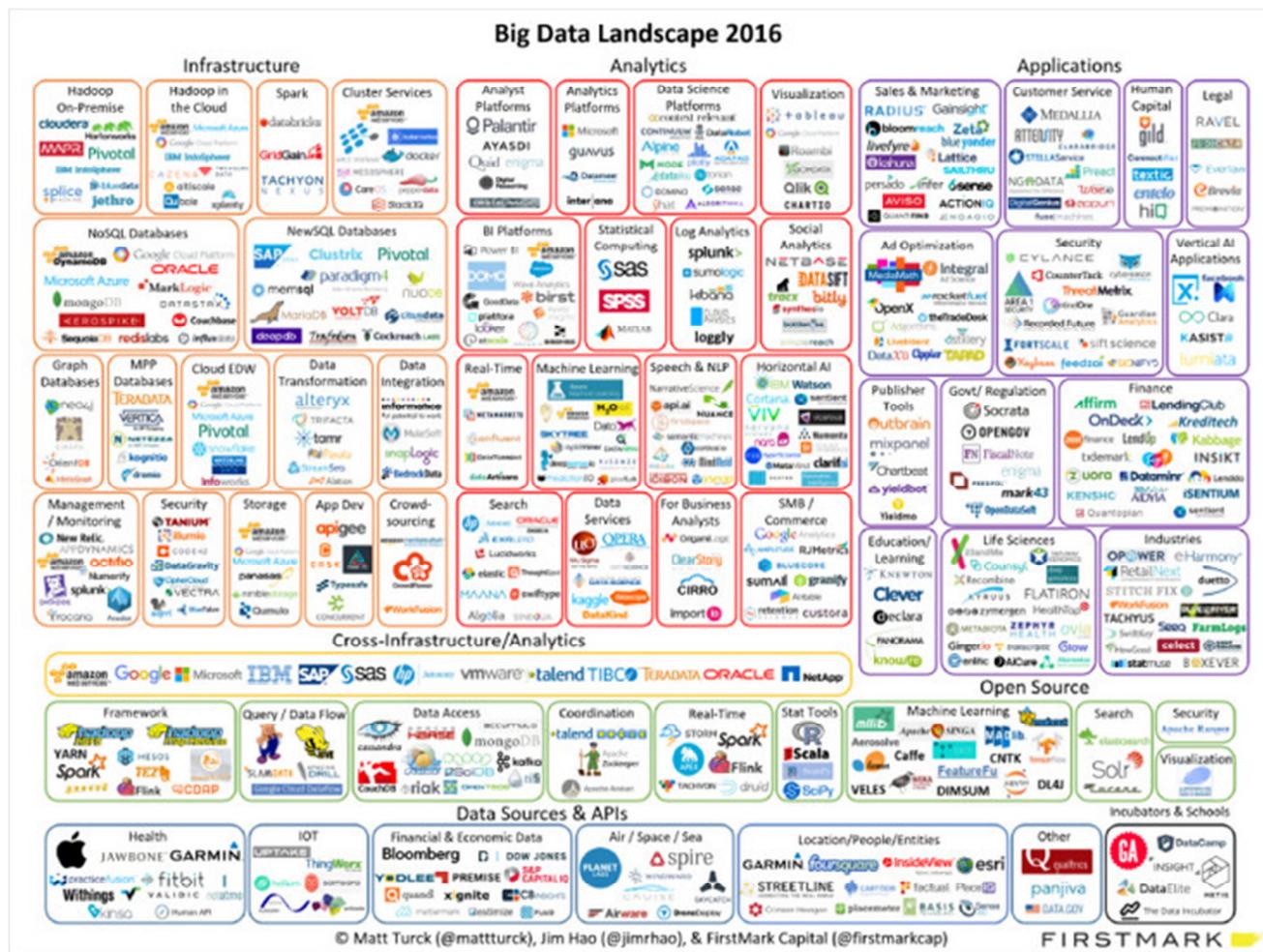
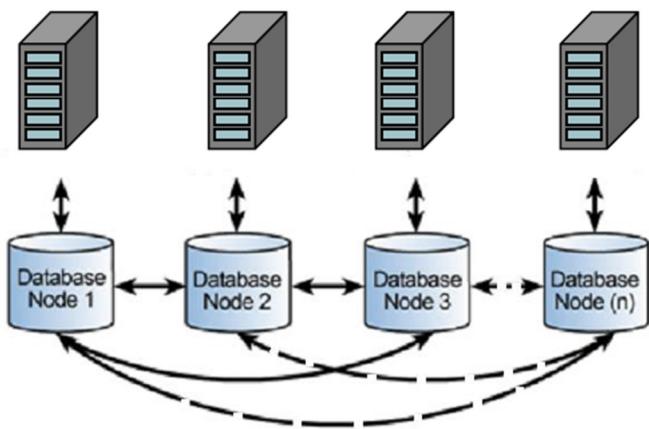


Fig. 5.5. Big Data Landscape. Created by @Mattturk, @Jimrhao and @firstmarkcap

Hadoop: Una breve historia

La historia de Hadoop está necesariamente unida a la de Google. De hecho, podría decirse que Hadoop nace en el momento en que Google precisa urgentemente de una solución que le permita continuar procesando datos al ritmo que necesita, en una proporción que repentinamente ha crecido de forma exponencial.

Google se ve incapaz de poder indexar la web al nivel que exige el mercado y por ello decide buscar una solución, que se basa en un sistema de archivos distribuidos.



"La falla del hardware es la norma más que la excepción"

Fig. 5.6 - Arquitectura Distribuida. Curso Big Data & Da Science de IBM en 2017

Esta solución, que posteriormente se denominará Hadoop, se basa en un gran número de pequeñas computadoras, donde cada uno de los cuales se encarga de procesar una porción de información. Lo bueno del sistema es que, a pesar de que cada uno de ellos funciona de forma independiente y autónoma, todos actúan en conjunto, como si fueran una sola computadora de dimensiones increíbles. En 2006, Google publica todos los detalles acerca de su nuevo descubrimiento, compartiendo su conocimiento y experiencia con todos los usuarios que anhelaban acceder a esta información. Entre el conjunto de beneficiarios, destaca el interés de la comunidad Open Source que, apasionados por la idea y el nuevo horizonte que se abre frente a ellos, explotan sus posibilidades desarrollando una implementación a la que denominan Hadoop. A partir de ese momento, es Yahoo quien toma el relevo impulsando su expansión, para lograr alcanzar a grandes e icónicas empresas en el mundo de la informática, como Facebook, que empiezan a incorporarlo a sus rutinas, a beneficiarse de su uso y a participar en su desarrollo, junto con la comunidad Open Source.

Hadoop es un sistema de código abierto que se utiliza para almacenar, procesar y analizar grandes volúmenes de datos. Sus ventajas son muchas, entre las que se destacan:

- Aísla a los desarrolladores de todas las dificultades presentes en la programación paralela.
- Cuenta con un ecosistema que sirve de gran ayuda al usuario, ya que permite distribuir el archivo en nodos, que no son otra cosa que computadoras con commodity-hardware.
- Es capaz de ejecutar procesos en paralelo en todo momento.
- Dispone de módulos de control para la monitorización de los datos.

- Presenta una opción que permite realizar consultas.
- También potencia la aparición de distintos add-ons, que facilitan el trabajo, manipulación y seguimiento de toda la información que en él se almacena.

La siguiente figura 5.4 presenta el ecosistema Hadoop

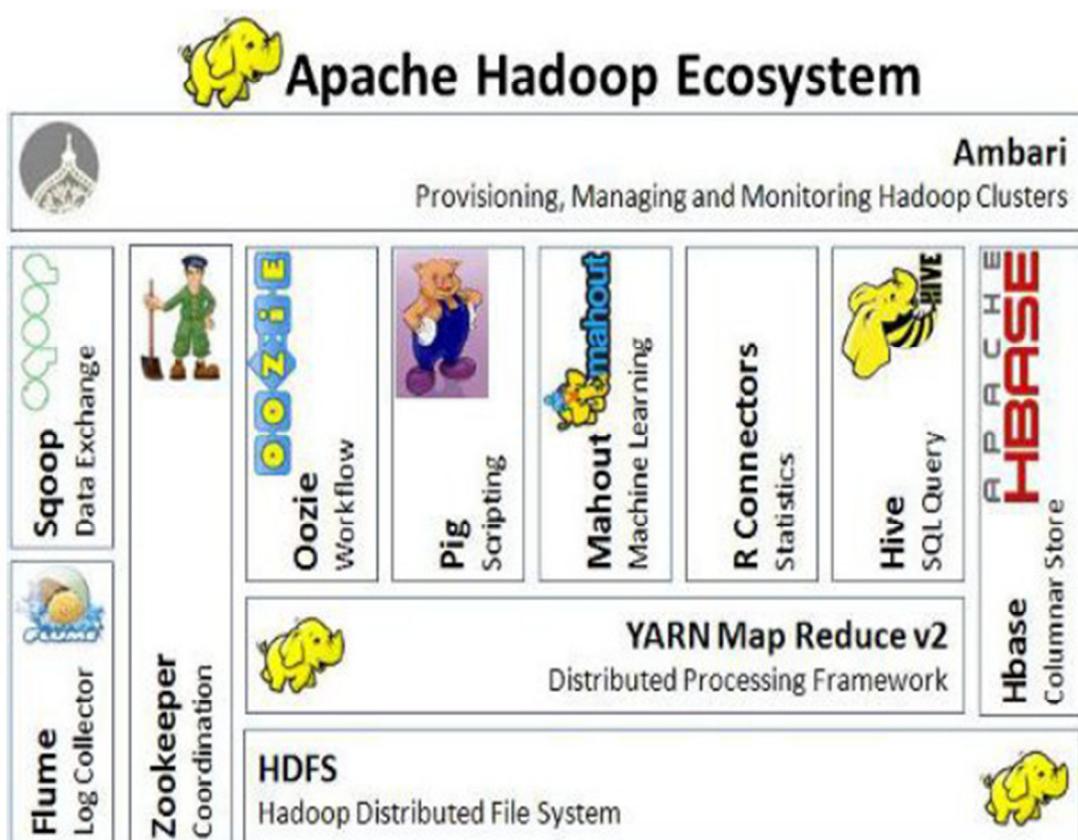


Fig. 5.7 - Ecosistema Hadoop. Curso Big Data & Da Science de IBM en 2017

Hadoop es un sistema que se puede implementar sobre hardware a un costo relativamente bajo, siendo a su vez totalmente gratuito para software. Esta circunstancia supone que, aquella información que antes las empresas no podían procesar debido a los límites de la tecnología existente o a barreras de tipo económico, que se hacían insalvables en muchos casos; hoy pueda ser almacenada, gestionada y analizada, gracias a esta herramienta.

Cualquier organización que utilice Hadoop puede obtener información nueva, al mismo tiempo que descubre y aplica cualquier otro tipo de análisis a sus datos, como por ejemplo una regresión lineal sobre millones de registros de su histórico.

Los **componentes básicos** de Hadoop son los siguientes:

- **HDFS**: consiste en un sistema de archivo distribuido, que permite que el fichero de datos no se guarde en una única máquina sino que sea capaz de distribuir la información a distintos dispositivos.

- **MAPREDUCE:** Se trata de un framework de trabajo que hace posible aislar al programador de todas las tareas propias de la programación en paralelo. Es decir, permite que un programa que ha sido escrito en los lenguajes de programación más comunes, se pueda ejecutar en un cluster de Hadoop.

La gran ventaja es que hace posible escoger y utilizar el lenguaje y las herramientas más adecuadas para la tarea concreta que se va a realizar. El objetivo es mover el procesamiento más que mover los datos. Se estructuran en nodos, de los cuales se distribuyen en maestros y esclavos, y se sincronizan para tener amplia tolerancia a fallas.

La función Map procesa pares clave-valor generando un conjunto de pares-valores intermedios. Los nodos ejecutan la misma tarea en paralelo, procesando datos del mismo o cercanos de acuerdo al archivo sobre el cual van a trabajar.

Por su parte la función Reduce recibe, une y combina los valores intermedios asociados a la misma clave intermedia entregados por el Mapper. Esquemáticamente, es muy ilustrativa la siguiente imagen tomada del curso de Big Data & DataScience de IBM

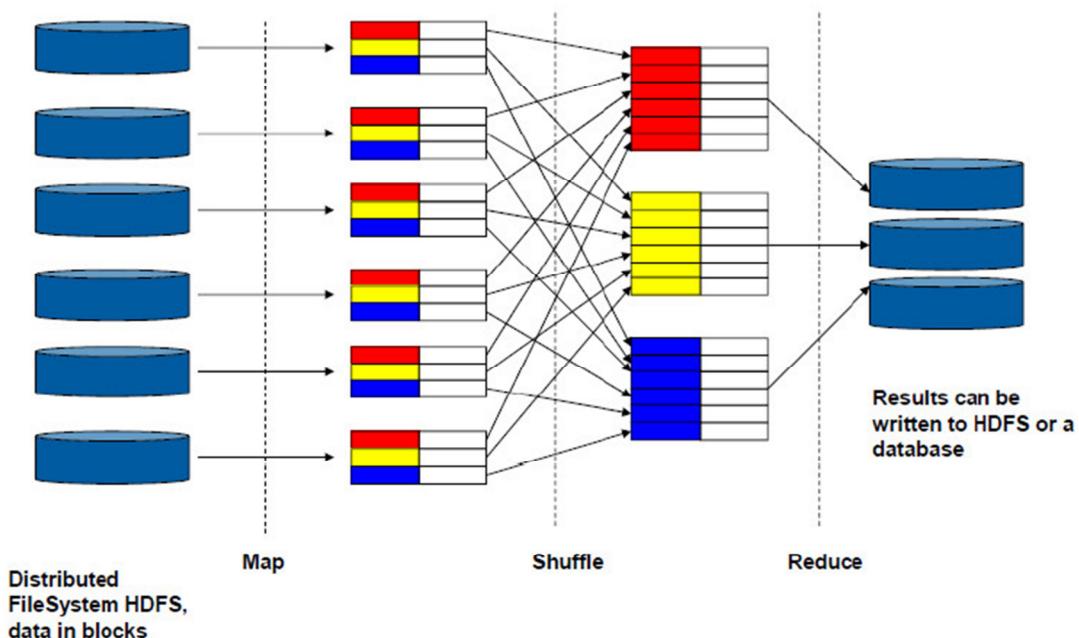


Fig. 5.8 - Hadoop MapReduce. Curso Big Data & Da Science de IBM en 2017

Instanciado en un ejemplo concreto, por ejemplo un contador de palabras. Contamos con miles de archivos y el objetivo es analizar las ocurrencias de todas las palabras para luego realizar un ranking de las más utilizadas. Gráficamente, la solución podría ser:

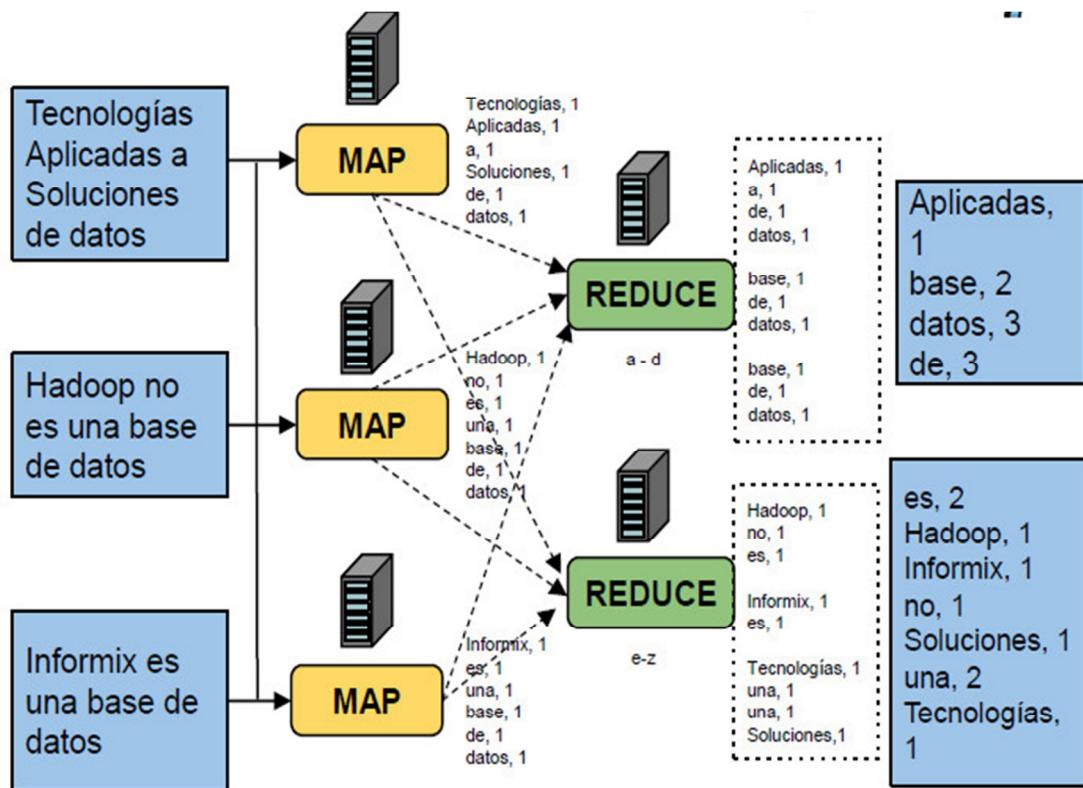


Fig. 5.9 - Hadoop MapReduce. Un ejemplo. Curso Big Data & Da Science de IBM en 2017

Como se puede observar, a la izquierda presentamos las fuentes de datos, que serían por ejemplo archivos planos. Luego, la función Map aplica el algoritmo de contar palabras sobre cada archivo y el resultado lo envía a la función Reduce, quien se encarga de unir todos los resultados, distribuyendo el trabajo a fin de optimizarlo y por tolerancia a fallas. Una vez con el resultado, puede utilizarse cualquier técnica de visualización para presentar los resultados. Y esto lo abordaremos en la siguiente sección de este capítulo.

Ciencias de los Datos

En la ciencia de los datos se estudia la extracción de conocimiento a partir de los datos. Es un campo interdisciplinario que involucra métodos científicos, procesos y sistemas para obtener información valiosa o realizar un mejor entendimiento de los datos. Está apareciendo debido a la necesidad de trabajar con conjuntos inmensos de datos (conocidos como Big data), formados tanto por datos estructurados como por datos semi estructurados o no estructurados. Estos conjuntos proceden de los datos generados por los dispositivos electrónicos, las redes sociales, la web, etc como mencionamos previamente. Es una continuación de algunos campos de análisis de datos como la estadística, minería de datos, aprendizaje automático y analítica predictiva. Sin embargo, los procesos y herramientas para procesarlos son muy diferentes a los utilizados en el pasado, como son las ecuaciones, algoritmos, evaluación e interpretación de resultados.

Actualmente compañías como Google, Amazon, Facebook, Netflix, Walmart, GAP, IBM y General Electric, entre otras, están liderando procesos de gestión masiva de información y reclutando “Data Scientists” en la nueva era que deviene de Big Data.

Es interesante mencionar que mientras BI trabaja sobre datos completos, archivos de datos limpios, de un volumen acotado, que informa sobre todo lo que paso en el pasado la ciencia de los datos trabaja sobre datos actuales, por lo tanto pueden estar incompletos, desordenados, se analizan para ver con qué información se cuenta y sus hallazgos impulsan decisiones sobre operaciones y productos. Todo esto involucra un gran conjunto de datos que es un desafío administrar y explotar.

El trabajo de un buen Científico de Datos es hacer descubrimientos “buceando” en un océano de datos cada vez más masivo y complejo. Identifican potenciales fuentes ricas en datos valiosos y acercan la brecha entre los datos, la información y finalmente las decisiones de negocio.

Jake Moody presenta en [Datacamp](#), una plataforma que brinda una Maestría en línea de Ciencia de Datos, una infografía muy interesante para ilustrar las diferencias entre [Data Engineering vs. Data Science](#). Ambos trabajan en conjunto y si bien en varios puntos sus funciones se solapan, cada vez más y más se van especializando.



Fig. 5.10 – Data Engineer vs Data Scientist

El *data engineer* es el responsable de la arquitectura que dará soporte al análisis de datos como las bases de datos en gran escala, computación en la nube, distribuida, utilizando herramientas específicas. Mientras que el *data scientist* es quien organiza los datos, los limpia y “masajea” para realizar estadísticas descriptivas o descubrir insights, construye **modelos** a partir de las necesidades del negocio a resolver.

Según el ciclo de vida de un proyecto de análisis de datos, cada rol se ve afectado en las distintas etapas, como se puede observar en la figura 5.6. Por ejemplo, el *data engineer* requiere comprender los datos con el fin de importarlos y limpiarlos y así garantizar la calidad de los mismos con técnicas específicas y en menor medida su modelado, machine learning, reporte y visualización. Estos últimos 4 aspectos son abordados más en profundidad por el *data scientist*, quien requiere comprender las reglas del negocio, realizar análisis de los datos y presentarlos de manera adecuada, correcta y completa, en tiempo y forma para que sea útil para la toma de decisiones.



Fig. 5.11 – Data engineer vs Data scientist Workflow

Sus skills también difieren y utilizan distintas herramientas, aunque comparten algunas. El *data engineer* requiere más habilidades relacionadas con bases de datos, modelos predictivos, computación distribuida; mientras que el *data scientist* requiere habilidades para gestionar herramientas ETL, machine learning, estadísticas, operaciones de búsqueda, storytelling y visualización, entre otras.

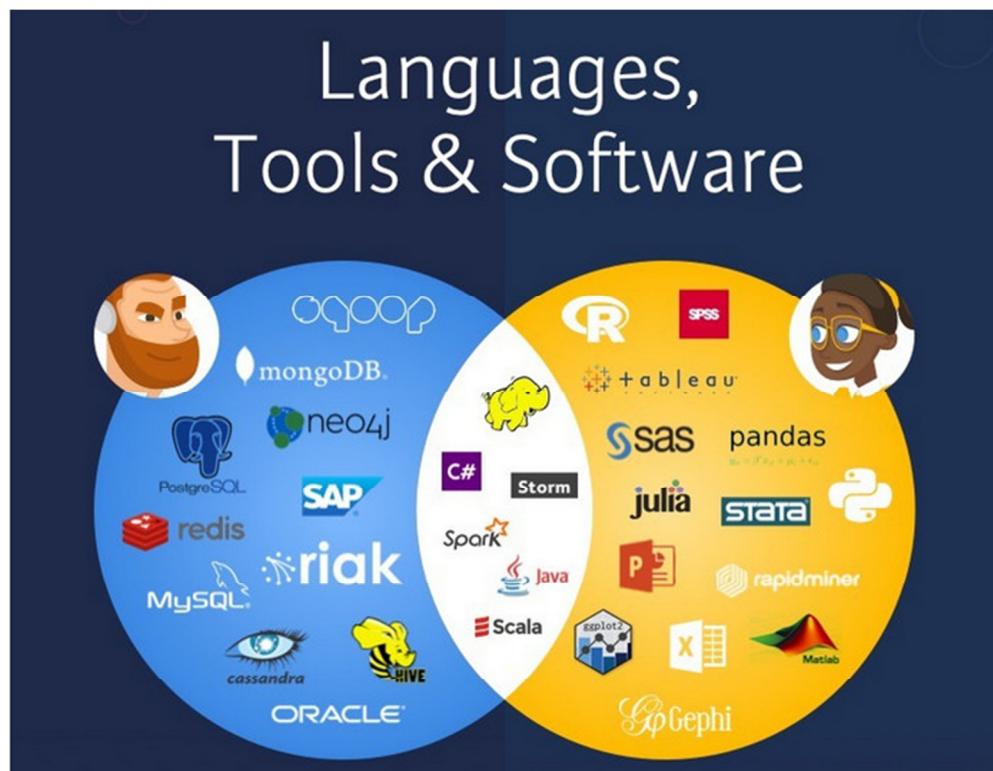


Fig. 5.12 – Data Engineer vs Data Scientist. Herramientas y software

Las responsabilidades también difieren, como se puede observar en el siguiente cuadro comparativo:

Data Engineer	Data Scientist
Desarrollar, testear y mantener la arquitectura de gran escala.	Dar respuesta a las preguntas del negocio y de la industria
Garantizar y asegurar la disponibilidad de la arquitectura del negocio.	Tomar ventaja del gran volumen de datos que se dispone para dar respuesta a las necesidades del negocio.
Descubrir oportunidades para la adquisición de datos.	Realizar modelados predictivos y prescriptivos utilizando las herramientas adecuadas como programas analíticos, machine learning y estadística.
Desarrollar procesos para obtener datasets necesarios para realizar las tareas de modelado, minería y análisis predictivo.	Encontrar patrones ocultos en los datos. Explorar y examinando los datos
Integrar los datos de distintas fuentes, estructurados, semi estructurados y no estructurados, a través de la gestión de una amplia variedad de lenguajes y herramientas.	Automatizar tareas para realizar la analítica de los datos.
Realizar recomendaciones para brindar datos de calidad, en forma eficiente.	Presentar storytelling para los gerentes claves de la organización, basada en datos.

Tabla 5.1 – Responsabilidad del Data engineer vs Data Scientist

Es de destacar que si bien ambos puestos tienen amplia demanda, el Data Scientist se encuentra mejor remunerado y existe mayor demanda.



Fig. 5.13 – Data Engineer vs Data Scientists. Propuestas laborales y remuneración.

Metodología

Para hacer un buen análisis de datos lo que primero debemos armar es una base de conocimiento sobre lo que queremos estudiar. Esta base debe estar armada con variables independientes y debemos tener una variable de clasificación que es la variable dependiente y la que se va a predecir. Por ejemplo, para saber si un tumor es cancerígeno o benigno:

Caso	Ubicación	Tamaño	Forma	Aspecto	Color	Clasificación
P1	U1	Y	G	A	C	C
P2	U1	Z	N	A	S	C
P3	U2	Y	G	B	S	B

Tabla 5.1 – Ejemplo de Ciencia de Datos

Las variables dependientes del ejemplo serían ubicación, tamaño, forma, aspecto y color. Una vez que tenemos los datos armamos el modelo, este modelo es usado con los datos nuevos de cada una de las instituciones las cuales lo aplican. Es importante tener en cuenta que una vez que confeccionamos el modelo no lo entrenamos más, dado que el mismo ya aprendió

y está validado de ese modo. La Ciencia de Datos lo que hace es encontrar el mejor modelo a aplicar. Entrenar al modelo para que aprenda y luego prediga. Pero ¿qué es un modelo? Existen dos grandes grupos de modelos dentro de la ciencia de los datos:

- Machine learning: proviene de la inteligencia artificial (IA), de los años 60 hacia acá, mucho de los años 80 y 90, se basa más que nada en entrenamiento, algunas de las técnicas serían las redes neuronales y support vector machine.
- Data mining : proviene de la estadística, hay métodos desde el año 1700, se basa más que nada en fórmulas, por ejemplo regresión lineal que es el caballo de batalla de DM, análisis de discriminantes, sólo para clasificación y PCA o análisis de componentes principales. No son conjuntos disjuntos, hay una intersección entre estos grupos que aún no está definido ni cerrado. Algunos modelos necesitan fórmulas y entrenamiento. En la intersección de ambos grupos, algunas de las técnicas serían arboles de decisión, K-means y Bayes. Minería de Datos es una disciplina en sí misma y será abordada en próximas versiones del libro.

Por su parte las **variables** pueden ser **cualitativas o categóricas**, por ejemplo podemos poner Argentina, Chile, Bolivia o 1, 2, 3 donde cada número representa un país; o **cuantitativas** las cuales son numéricas, y pueden ser discretas(entero) o continuas (flotantes). A su vez pueden ser **nominales**, por ejemplo Argentina, Chile, Bolivia u **ordinales** (primario, secundario, etc) son categorías relacionadas entre ellas por un orden.

Si la tarea a realizar predice variables cualitativas estamos ante un problema de **clasificación**, por ejemplo si vamos a predecir si una persona es buena pagadora. Las tareas que predicen variables cuantitativas suelen identificar un problema de **regresión**, por ejemplo, si vamos a predecir la temperatura de mañana.

Se denomina **outlier** a un valor o conjunto de valores de la muestra que llama la atención, puede ser un error o no. Por ejemplo, en la base de conocimiento es algo mal clasificado o información errónea.

Dependiendo del problema, se realizan ajustes del modelo aplicando distintas técnicas. En el anexo se presentan distintas situaciones de acuerdo al tipo de problema a analizar.

Hoy en día, las herramientas indispensables para el análisis de datos incluyen librerías de Python y R Studio.

- Python es un lenguaje de programación interpretado, multiplataforma y multiparadigma ya que soporta programación imperativa, orientada a objetos y en cierta medida funcional. Es dinámicamente tipado y open source. La potencia del lenguaje es también su posibilidad de extensión. Si bien es posible construir una gran variedad de aplicaciones, en los últimos años se desarrollaron muchas librerías para ciencia de los datos, generándose una gran comunidad y experiencias. Más información <https://docs.python.org/3/tutorial/index.html>

- R es un lenguaje de programación estadístico, muy potente para obtener información de los datos. Hoy en día es imprescindible para el científico de datos. La librería Shiny facilita la publicación rápida y sencilla de los resultados. Más información en: <https://www.rstudio.com/resources/webinars/data-science-essentials/>

En el anexo de este capítulo se presentan dos casos de ejemplos para trabajar con R y con las librerías de Python.

Visualización de Datos Big Data: Recomendaciones Básicas de Diseño

Hoy en día, la sociedad genera mucha información a través de su amplia participación e intervención en búsquedas, transacciones, intercambio de información, mensajes en diferentes aplicaciones y redes sociales en Internet. Esto, junto al aumento de las capacidades de almacenamiento, da lugar a que las distintas organizaciones cuenten con la disponibilidad de una masividad de información, una gran cantidad y variedad de datos sobre las personas.

La analítica de datos permite estructurar este caudal de información recogido desde distintos medios, realizar con ello un tratamiento y análisis exhaustivo de los datos para su mejor explotación y aprovechamiento, utilizando métodos de inteligencia artificial, aprendizaje automático, análisis predictivo. Posibilita obtener una visión analítica, descriptiva, estadística de grandes conjuntos de datos, como también prospectiva sobre tendencias y evolución de los datos, que es esencial para la toma de decisiones.

Uno de las etapas de la analítica de datos es la visualización de los mismos. La visualización de los datos diseñada correctamente, no solo permite focalizar valores, mediciones, métricas más importantes, sino que ayuda a descubrir patrones y observaciones muy difíciles de detectar con otros sistemas.

Pero las representaciones mal diseñadas, la elección de tipos de gráficos incorrectos, la falta de contexto, la mala organización y estructuración puede provocar efectos adversos reduciendo y restringiendo las potencialidades previamente mencionadas. El mal diseño en la visualización puede provocar imprecisiones más que certezas, mala interpretación de los datos, confusión, representación incorrecta de la información registrada, como también demoras, intentos fallidos e interacciones infructuosas, entre otros.

En este sentido, en las próximas secciones se presentarán algunas recomendaciones básicas a considerar en la fase de visualización de datos. A la hora de realizar la visualización de datos Big Data, deberemos tener en cuenta una serie de reglas:

- Favorecer una correcta percepción y comprensión de los datos

Esto constituye uno de los pilares de la visualización de los datos por lo que el diseño debe permitir lograrlo. Por lo general se apunta a que la visualización sea atractiva, pero lo que se debe trabajar es en lograr que sea realmente efectiva.

La percepción es el proceso por el cual el cerebro humano recibe estímulos externos de su entorno. En este contexto constituye un proceso por el cual el cerebro capta la información representada visualmente, como por ejemplo en un tablero. De acuerdo a la forma de representación utilizada, la percepción puede implicar un procesamiento casi subconsciente e instantáneo. La imagen como la posición, el color, el tono e intensidad, la alineación, el tamaño y la forma se pueden procesar sin atención consciente, lo que se conoce como procesamiento previo a la atención. Hay otras representaciones como el texto y los números que requieren un enfoque consciente para procesar, por lo que la combinación apropiada y balanceada de ambos puede ayudar a la percepción.

- Beneficiar la intuición y demás factores humanos aplicados en la interacción

La visualización de datos debe ser visual obviamente pero también intuitiva. Si está diseñada en forma apropiada va a permitir que el usuario comprenda lo que necesita saber en forma directa, donde el reconocimiento, el entendimiento y la adquisición de la información transmitida sea sin costo ni esfuerzo mental.

Brindar posibilidades al usuario de intervención, interacción y control sobre los datos que se visualizan en forma intuitiva puede optimizar la experiencia del usuario. Por ejemplo se puede poner a disposición metadatos para que el usuario pueda aplicar y filtrar, para generar predicciones y deducir relaciones, generando nueva información.

Recomendaciones de HCI, usabilidad como de la psicología cognitiva deben considerarse en pos de aprovechar adecuadamente cuestiones de atención, percepción, creatividad, pensamiento, concentración, entendimiento, captación y demás factores humanos que intervienen en la interacción.

- Proveer claridad y expresividad

La visualización como la interacción si se provee, debe ser simple, clara y fácilmente entendible por los usuarios. Se debe usar títulos y etiquetas coherentes para los datos que se muestran y que sean fáciles de leer. El uso de elementos adicionales como anotaciones o puntos de la historia, deben agregar valor.

Debe proveerse mecanismos de asistencia y guía bajo demanda, para los usuarios menos experimentados. Tener en cuenta que los usuarios de las herramientas de visualización de Big Data, pueden tener conocimientos diferentes en cuanto al uso y manejo de recursos tecnológicos como también del análisis empresarial, estadístico, analítico y predictivo, que puede adquirir con estas herramientas.

- Asegurar precisión y confiabilidad

Se debe mostrar los datos en un contexto visual y textual para asegurar mejor interpretación y precisión de la visualización. Se debe tener acceso a las fuentes, información sobre la fecha de recolección de los datos, la de última actualización.

El contexto es importante para mejorar la interpretación. Por ejemplo, si se informa que hubo 500 accidentes vehiculares en una zona de la ciudad, se puede interpretar diferente si se le adiciona como contexto que dicha zona es céntrica y muy poblada, o por el contrario que es una zona rural.

La confiabilidad no solo se da en la precisión sino además en cuestiones de consistencia, donde los recursos visuales como textuales que se utilicen como los títulos, los sectores, el color, sector, tamaño, efectos, señales o marcas, deben proveerse de manera pensada, bien diseñada, apropiada para el contexto, estado o fin para lo que se lo aplique. Además, esto debe mantenerse y ser homogéneo en el resto de las representaciones que se utilicen.

- Priorizar los datos relevantes

El diseño más apropiado es el que permite enfatizar la información importante, sin el empleo de recursos visuales que distraigan, entorpezcan o hagan engorrosa la visualización.

La componente más importante de la visualización de datos son los datos, no el diseño. El diseño solo debe guiar estratégicamente y, en forma transparente, a los usuarios a la focalización de los datos relevantes.

Una narración bien planificada, una buena organización agrega profundidad, y ayuda a que los datos se manifiesten en un contexto más amplio y significativo, siendo más fácil de encontrar, distinguir y memorizar.

Por ejemplo, se puede priorizar las métricas más críticas en un tablero de instrumentos e incluso dentro de esa lista de información crítica, se puede clasificar en niveles de importancia. Por lo tanto, los datos pueden ser ponderados de manera diferente y se pueden contrastar sus visualizaciones de manera tal de resaltar información importante a transmitir.

Priorizar no significa colorear. El uso de colores debe ser analizado con cuidado y utilizado con moderación. No deben aparecer en todo el tablero, llamando la atención y obstaculizando así la clasificación de lo prioritario y el enfoque del usuario.

- Aplicar una Metodología

La etapa de visualización debe ser un proceso de ingeniería, donde se debe considerar los requisitos de diseño, el tipo de los datos a transmitir, la audiencia, analizar las tendencias de diseño y el impacto en los usuarios.

Es importante conocer a los destinatarios a la que está dirigida la información. Se debe modularizar a la audiencia. No todos los usuarios finales percibirán la misma información de la misma manera, por lo que se deberá trabajar desde esa perspectiva, visualizar lo que su audiencia necesita saber. También, es importante saber el rol que jugarán los mismos, si adquieren un rol de sólo monitor, consumidor y observador de los datos o tendrán acciones de intervención.

Para garantizar calidad en la visualización, es necesario diseñar distintas formas de visualización de datos y ponerlas a prueba frente a usuarios finales como también frente a las recomendaciones existentes. Mediante un proceso metodológico adecuado se podrá lograr asegurar calidad y efectividad de representación, como se explica en el siguiente punto.

- Diseñar iterativamente

La visualización requiere una comprensión del concepto visual adquirida por los usuarios finales que debe ser evaluada. Esto es algo muy difícil de lograr, por lo que se recomienda diseñar pruebas de concepto y prototipos y ponerlos bajo evaluación de los usuarios.

Es muy importante para entender sus reacciones y nivel de aceptación, contar con la participación de los mismos, obtener su retroalimentación en una configuración interactiva, de revisación, puesta en consideración, ajuste y refinamiento.

El proceso de análisis visual consiste entonces, en un ciclo continuo que se inicia en los datos y sus posibles transformaciones, que conducen a la visualización y la construcción de modelos. Existe una conexión, diálogo entre ellos, con el objetivo de extraer conocimiento que pueda ser usado para iterar el proceso de análisis, con un mayor nivel de detalle y/o complejidad, tal y como muestra en la siguiente figura.

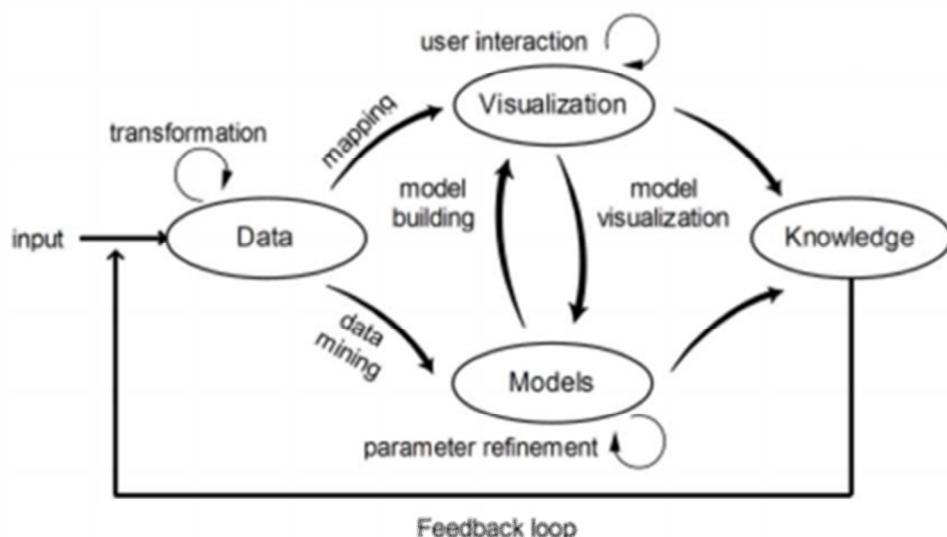


Fig.5.14: Proceso de análisis visual. Imagen extraída en
Visual analytics: Definition, process, and challenges. In Information visualization.

Desde una perspectiva de análisis visual, las dos primeras etapas son la transformación o adaptación de los datos y su visualización, incluyendo en ésta la interacción. La capacidad de interacción debe permitir al usuario realizar, al menos las operaciones básicas sobre vista general, zoom, filtro y selección.

Una vez establecido el objetivo del análisis visual de los datos, se trata de seleccionar un tipo de visualización interactiva que permita realizar dicha exploración preliminar.

- Identificar el tablero de instrumentos

Un tablero es un recurso muy potente que posibilita eficacia en el relato y narración de la historia de los datos.

Existen distintos tipos de paneles de instrumentos, entre los principales se cuentan con los paneles operativos, estratégicos o ejecutivos y analíticos. Cada uno, presenta determinados rasgos a considerar en el proceso de diseño.

El tablero operacional, generalmente se caracteriza por una línea de investigación o consultas que regularmente son actualizadas para el monitoreo de eventos efectuados en ciertos lapsos de tiempo y frecuencia.

El tablero estratégico o ejecutivo, es una vista en alto nivel de una pregunta o línea de investigación que generalmente se responde de forma rutinaria y específica, y generalmente presenta los KPI de una manera mínimamente interactiva.

En cambio, el tablero analítico, se caracteriza por una vista altamente interactiva que posibilita una variedad de perspectivas de análisis y visiones contextuales de un tópico central.

- Analizar el tipo de datos a visualizar

Si bien casi todo puede convertirse en datos y codificarse visualmente, conocer el tipo y origen de los datos es tan importante como entender los datos en sí. Hay tres tipos de datos categórico, ordinal y cuantitativo. Los datos categóricos son datos que lógicamente se conjugan como América del Sur, Europa y Asia. Los ordinales son aquellos que presentan una secuencia lógica como medallas de oro, plata y bronce. Y los cuantitativos son aquellos que denotan cantidad como 1 millón de ventas, 300 suscripciones, entre otros.

En este sentido, distintas representaciones visuales funcionan mejor con diferentes tipos de datos. Por ejemplo, las gráficas de dispersión funcionan bien con los datos cuantitativos, mientras que los gráficos de línea funcionan mejor para datos ordinales de fechas más que para los datos categóricos no ordinales ya que implican continuidad. También hay estudios que relacionan los recursos visuales apropiados para cada tipo de datos. Por ejemplo, aplicar colores discretos a datos categóricos es un error común que cometan los visualizadores.

- Elegir los diagramas y gráficos apropiados

Hay representaciones que funcionan mejor para diferentes tipos de datos que otros. Es necesario estudiar las fortalezas de cada tipo de gráfico y qué características de datos visualizan mejor.

Por ejemplo, un gráfico de embudo crea una visualización del progreso de los datos a medida que pasa por etapas secuenciales, como se muestra en la figura. Cada etapa del embudo representa un porcentaje de todo el proceso que se rastrea. El cuadro es útil para comprender el estado del proceso en forma inmediata y para identificar posibles áreas problemáticas.

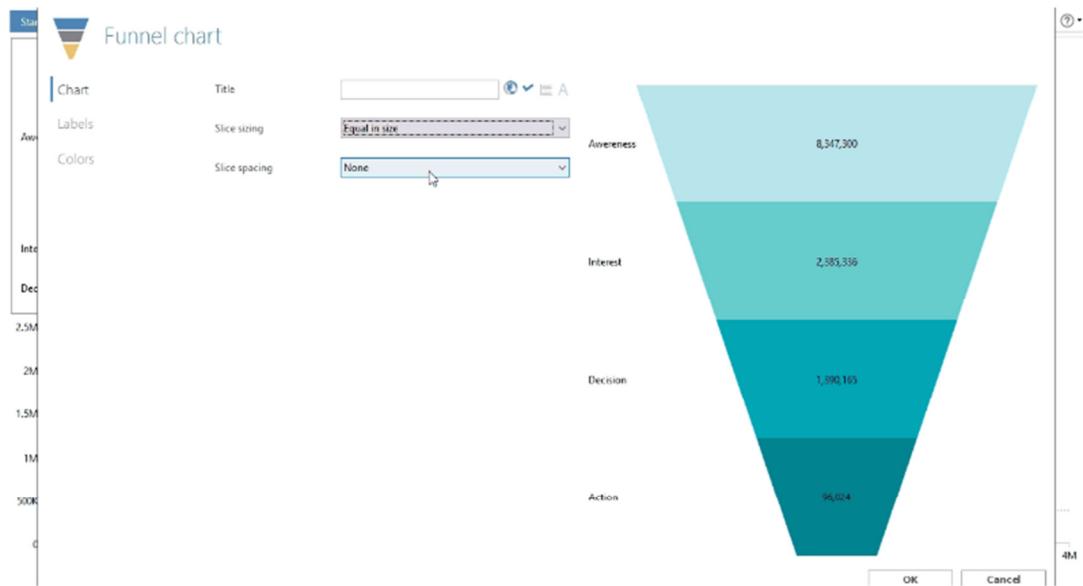


Fig.5.15: Gráfica de embudo. Imagen extraída en Best Practices in Data Visualization. Disponible en: www.targit.com.

Se lo puede utilizar para rastrear datos secuenciales, para revelar problemas potenciales en un proceso lineal o para rastrear el progreso de eventos tales como campañas de marketing y procesos de ventas.

Otro ejemplo, es el gráfico de pasos que es una variación del gráfico de líneas tradicional con la línea formando una serie de pasos entre los puntos de datos. Los gráficos de pasos son visualizaciones de datos particularmente útiles cuando se busca una tendencia general, cambios en los datos que ocurren en intervalos irregulares, cuando se quiere comparar la magnitud de los cambios en los datos o un seguimiento del aumento en los precios por ejemplo, o tasas variables como las tasas de interés y las tasas impositivas, como se muestra en la siguiente figura.

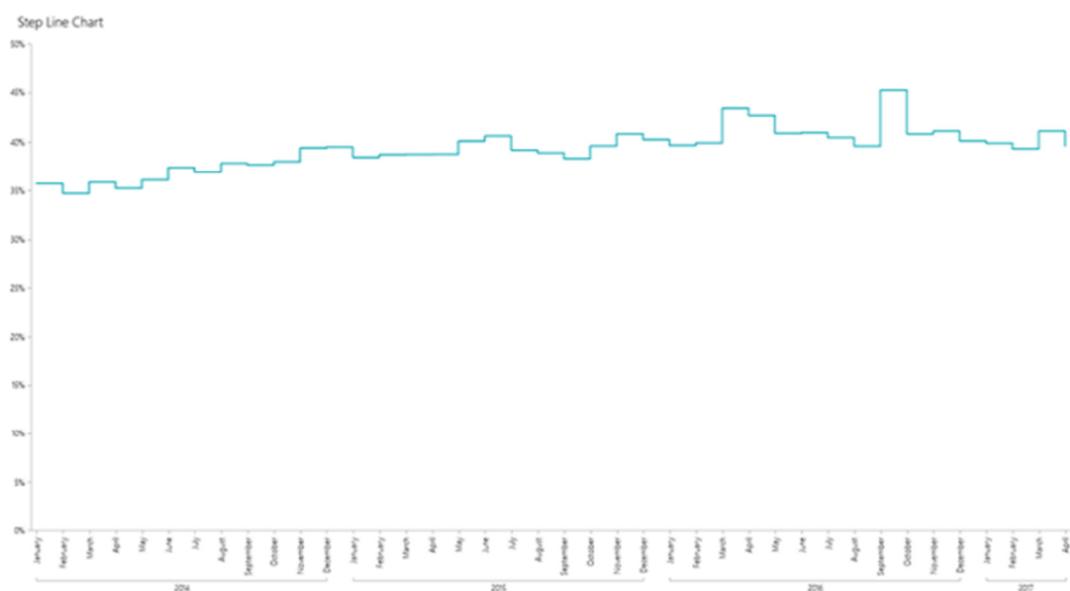


Fig.5.16: Gráfico de Pasos. Imagen extraída en Best Practices in Data Visualization. Disponible en: www.targit.com

En el caso de los diagramas de spline, los mismos representan otra variación del gráfico de líneas que traza una línea curva a través de cada punto de datos en una serie, como se muestra en la Figura 5.17.

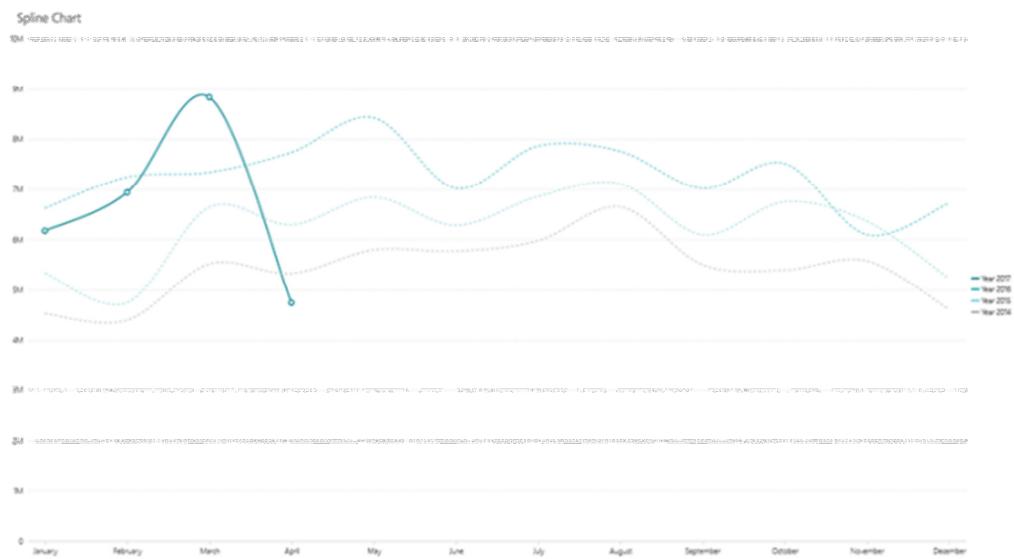


Fig.5.17: Diagrama de spline. Imagen extraída en Best Practices in Data Visualization. Disponible en: www.targit.com.

Este tipo de diagramas, son útiles para visualizar cambios suaves y graduales en los datos, en vez de picos.

Un último ejemplo, para comprender los tipos de gráficos según la información a mostrar, se encuentra el gráfico de cascada, como la Figura 5.18. El mismo muestra el efecto acumulativo de los valores de datos introducidos secuencialmente representados por columnas suspendidas en el aire.

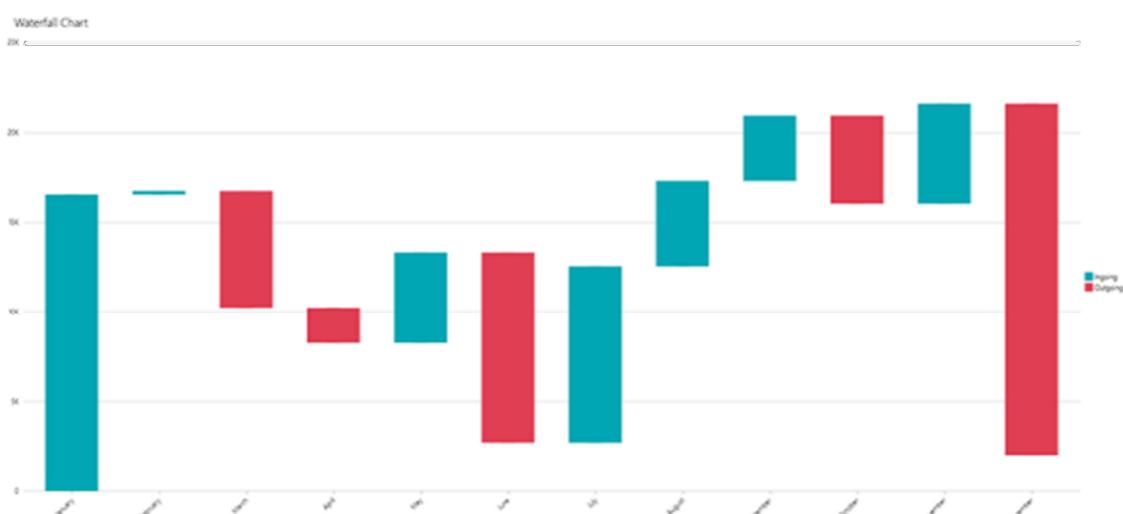


Fig.5.18: Gráfico de cascada. Imagen extraída en Best Practices in Data Visualization. Disponible en: www.targit.com.

Los gráficos de cascadas son ideales para visualizar cambios entre un punto de inicio y final específico, o cuando se desea visualizar valores que cruzan el eje 0.

Hay que tener en cuenta que las distintas representaciones para realizar distintos aspectos de la información pueden complementarse en una sucesión de historia o compartir espacio en un tablero, pero no debe generar ruido visual ni una sobrecarga de información.

- Considerar la Accesibilidad Web

En caso que la visualización de datos, se encuentre como contenido de páginas web hay que considerar las recomendaciones de accesibilidad especificadas por la W3C.

La página web, con su contenido HTML y sus hojas de estilo CSS que determinan el aspecto de los elementos de la página, permite mediante código JavaScript manipular el DOM de la misma, es decir, la estructura de la propia página web entendida como un documento estructurado jerárquicamente, generando nuevos contenidos que se incrustan dinámicamente.

Hay librerías JavaScript como D3 (o también D3.js) que permite manipular datos en diferentes formatos (tablas, CSV o JSON, entre otros), generar gráficos vectoriales, desde gráficos de barras hasta complejas visualizaciones combinando diferentes elementos gráficos. Los mismos pueden ser incrustados en forma dinámica, en la página web para su manipulación, incluyendo elementos de interactividad, tanto por lo que respecta a la interfaz del usuario como al uso de transiciones que aportan dinamismo a la visualización.

En este caso recomendaciones de percepción, robustez, operabilidad y comprensión debe ser tenidas en cuenta, como también cuestiones de diseño responsive. Esto permitirá entre otras cosas, que la página como la visualización de datos masivos que se despliegue allí, pueda accederse utilizando tecnologías asistivas, operarse desde distintos dispositivos móviles, y manejarse desde distintos dispositivos de interacción.

La masividad de la información está transformando la manera en que se observa y se entiende a la sociedad y al mundo, la forma en que se realizan los negocios, la toma de decisiones y se aplican las políticas públicas.

La minería de datos va más allá de la estadística tradicional como cálculo de medias geométricas, análisis de varianza, entre otros. La minería de datos presenta posibilidades de transformación, recopilación de distintas fuentes y medios, procesamiento, modelación de conjuntos masivos de datos, como también visualización.

Entonces, el Big Data y más concretamente las herramientas de análisis, gestión y visualización de datos están desempeñando un papel cada día más central y preponderante.

Pero, las potencialidades de utilizar e implementar estrategias de Big Data no debe verse empañado por malos diseños en las visualizaciones de datos realizadas, que se agudizan aún más, frente a la masividad de información que se potencia, la complejidad de los datos a transmitir, como a la heterogeneidad de los mismos.

La introducción a las mejores prácticas para visualizar datos que se presentó en este artículo, debe ser considerado y aplicado para poder garantizar calidad en la representación, y posibilitar el acceso y monitoreo de Big Data por parte de una comunidad de usuarios cada vez más diversa y multitudinaria.

Bibliografía

- Wattenberg , M.. Viégas, A ,Hollenbach, K. (2007) Visualizing Activity on Wikipedia with Chromograms <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.91.6785>
- Ciencia de Datos https://es.wikipedia.org/wiki/Ciencia_de_datos.
- García González, F. J.(2003). Aplicación de técnicas de Minería de Datos a datos obtenidos por el CEAMA. Universidad de Granada.
- Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In Visual Languages Proceedings, IEEE.
- Hollinsworth, David (2016). Storelling with data. Disponible en: <https://www.targit.com/en/blog/2016/01/data-visualization-best-practices-part-1>
- Thomsen, Kasper S. (2017). Best Practices in Data Visualization. Disponible en: www.targit.com.
- Keim, D., Andrienko, G., Fekete, J. D., Görg, C., Kohlhammer, J., & Melançon, G. (2008). Visual analytics: Definition, process, and challenges. In Information visualization. Springer Berlin Heidelberg.
- W3C N(2008). Web Content Accessibility Guidelines (WCAG) 2.0. Disponible en: <https://www.w3.org/TR/WCAG20/>
- Murray, S. (2013). Interactive data visualization for theWeb. O'ReillyMedia, Inc.

Bibliografía en línea

- <https://www.analyticsvidhya.com/blog/2014/07/statistics/>
- <https://jakevdp.github.io/PythonDataScienceHandbook/>
- <https://datahack.analyticsvidhya.com/contest/practice-problem-loan-prediction-iii/>
- <http://pandas.pydata.org/pandas-docs/stable/cookbook.html>
- <https://www.analyticsvidhya.com/blog/2016/01/12-pandas-techniques-python-data-manipulation/>
- <http://pbpython.com/pandas-pivot-table-explained.html>
- <https://www.analyticsvidhya.com/blog/2014/09/data-munging-python-using-pandas-baby-steps-python/>
- <https://www.analyticsvidhya.com/blog/2016/01/guide-data-exploration/>
- <https://docs.scipy.org/doc/numpy-dev/user/quickstart.html>

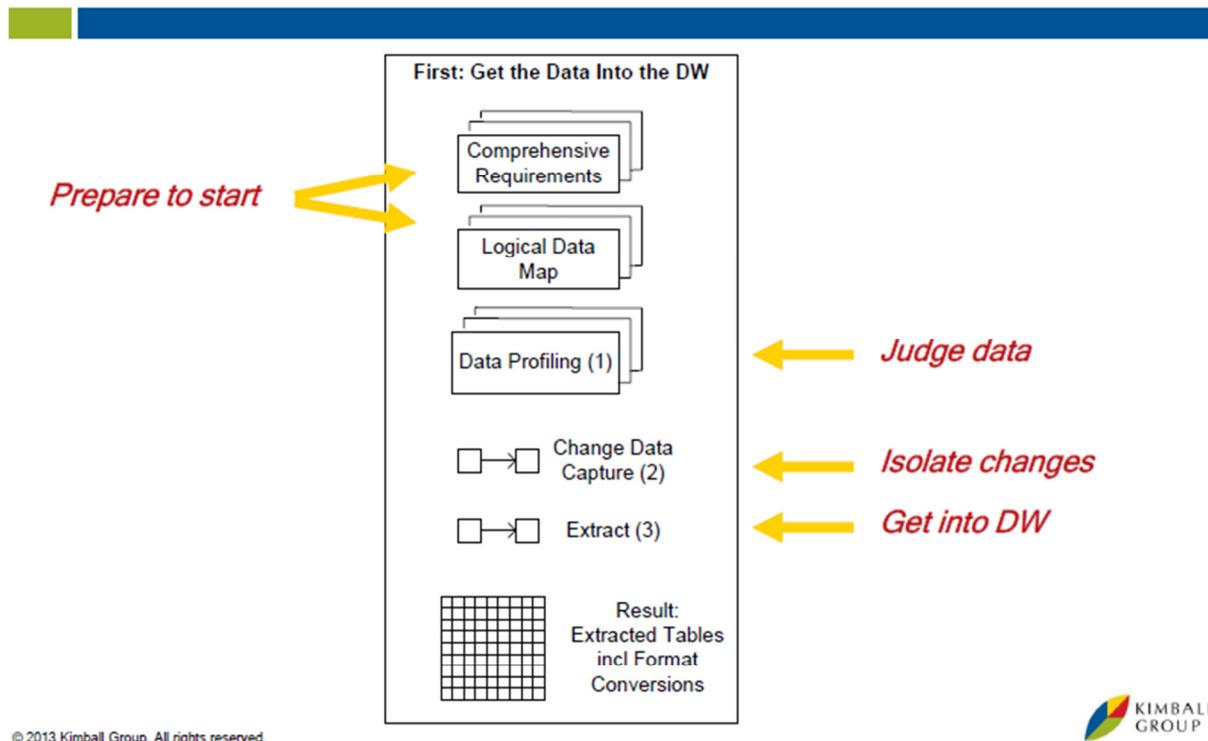
<https://github.com/apache/hadoop>
<https://matplotlib.org/users/index.html>
<https://docs.scipy.org/doc/scipy/reference/>
<https://github.com/jonathan-bower/DataScienceResources#python-notebook-tutorials>
<https://github.com/ujjwalkarn/DataSciencePython#data-science-with-python>
<https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/>
<https://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/>
<https://www.analyticsvidhya.com/blog/2015/11/improve-model-performance-cross-validation-in-python-r/>
<https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/>
<https://www.analyticsvidhya.com/blog/2015/01/decision-tree-simplified/>
<https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/>
https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

Capítulo 3 (Anexo)

Kimball Group estructura el proceso de ETL en 4 subsistemas que se detallan a continuación. Cada uno de ellos se [encuentra disponible en el sitio Web del grupo](#), el cual es muy útil para futuras referencias también.

E: Getting the Data Into the DW

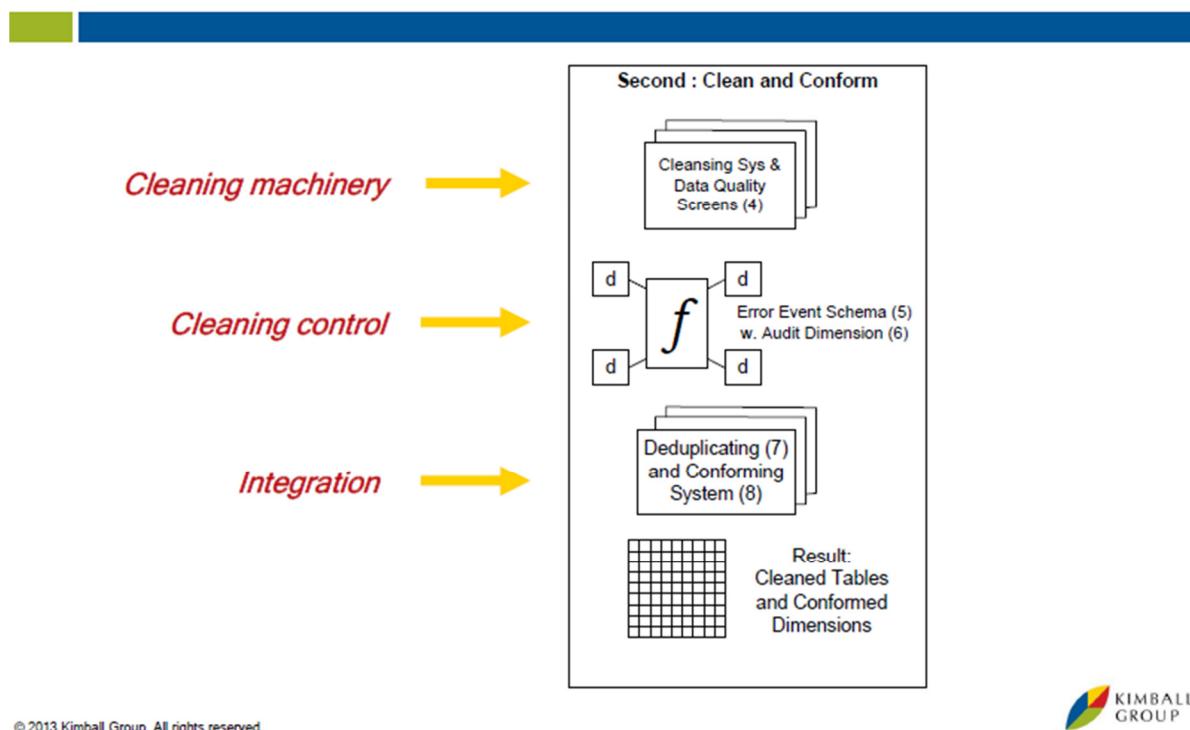
Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



Categoría 1: foco en extraer los datos del sistema fuente.

T: Clean and Conform

Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



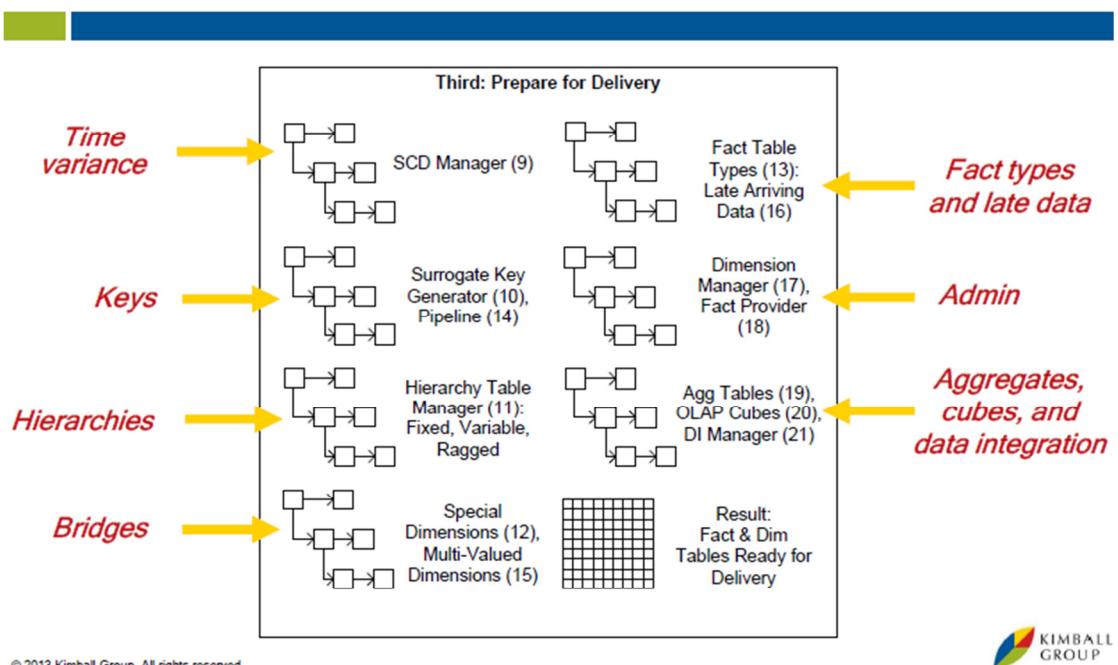
© 2013 Kimball Group. All rights reserved.



Categoría 2: Calidad y Limpieza

L: Prepare for Presentation

Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



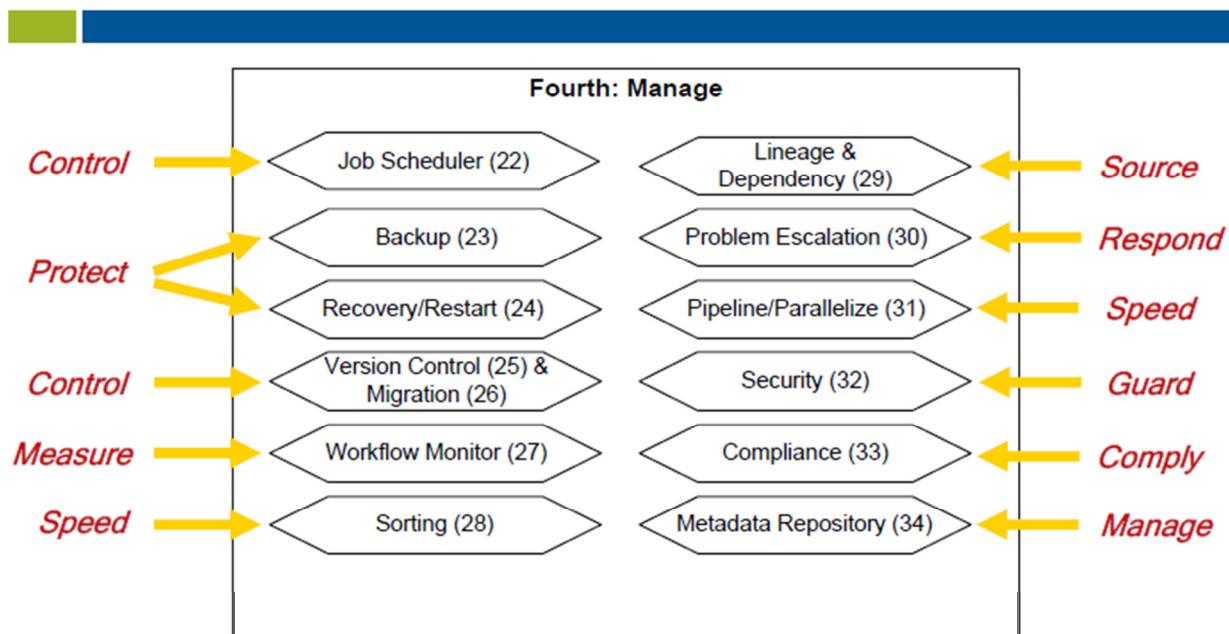
© 2013 Kimball Group. All rights reserved.



Categoría 3: Carga de los datos en la estructura

M: Manage All the Processes

Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



© 2013 Kimball Group. All rights reserved.



Categoría 4: gestión del proceso de ETL.

Capítulo 5 (Anexo)

Ejemplo práctico con Python y R Studio

A continuación veremos un ejemplo utilizando un stack de tecnologías comúnmente utilizado para la inicialización en la temática con el lenguaje de programación Python. Algunas bibliotecas interesantes:

- **Jupyter Notebook:** es una aplicación web que permite crear y compartir documentos que contienen código fuente, ecuaciones, visualizaciones y texto explicativo. Entre sus usos está la limpieza y transformación de datos, la simulación numérica, el modelado estadístico, el aprendizaje automático.
- **NumPy** (Numerical Python): La característica más poderosa de NumPy es la matriz n-dimensional. Esta biblioteca también contiene funciones básicas de álgebra lineal, capacidades avanzadas de números aleatorios y herramientas para la integración con otros lenguajes de bajo nivel como Fortran, C y C++.
- **SciPy** (Scientific Python) está basado en NumPy. Es una de las librerías más útiles para la variedad de módulos de ciencia e ingeniería de alto nivel, álgebra lineal, optimización y matrices dispersas.
- **Matplotlib** para trazar una gran variedad de gráficos, comenzando desde histogramas a gráficos de líneas.
- **Pandas** para operaciones de datos estructurados y manipulaciones. Se usa ampliamente para la manipulación y preparación de datos.
- **Scikit Learn** para aprendizaje automático. Basada en NumPy, SciPy y matplotlib, esta biblioteca contiene una gran cantidad de herramientas eficaces para el aprendizaje automático y el modelado estadístico, que incluye clasificación, regresión, clustering y reducción de dimensionalidad.
- **Statsmodels** para modelado estadístico. Statsmodels es un módulo de Python que permite a los usuarios explorar datos, estimar modelos estadísticos y realizar pruebas estadísticas.
- **Seaborn** para la visualización de datos estadísticos. Seaborn es una biblioteca para hacer gráficos estadísticos atractivos e informativos en Python. Está basado en matplotlib.
- **Bokeh** para la creación de gráficos interactivos, paneles y aplicaciones de datos en modernos navegadores web. Le permite al usuario generar gráficos elegantes y conci-

sos al estilo de D3.js. Además, tiene la capacidad de interactividad de alto rendimiento en conjuntos de datos muy grandes o de transmisión.

- **Blaze** para ampliar la capacidad de Numpy y Pandas a los conjuntos de datos distribuidos y de transmisión. Se puede usar para acceder a datos de una multitud de fuentes, incluyendo Bcolz, MongoDB, SQLAlchemy, Apache Spark, PyTables, etc.
- **Scrapy** para rastreo web. Es un marco muy útil para obtener patrones de datos específicos. Tiene la capacidad de comenzar en la url de inicio de un sitio web y luego buscar en páginas web dentro del sitio web para recopilar información.

Ahora que estamos familiarizados con las bibliotecas adicionales, hagamos una inmersión profunda en la resolución de problemas a través de Python. En el proceso, utilizamos algunas librerías poderosas y también encontramos el siguiente nivel de estructuras de datos. Lo guiamos a través de 3 fases clave:

1. Exploración de datos: descubriendo más sobre los datos que tenemos
2. Data Munging: limpiar los datos y manipularlos para adaptarlos mejor al modelado estadístico
3. Modelado Predictivo: ejecutar los algoritmos reales y divertirse.

Análisis exploratorio en Python usando Pandas

Pandas es una de las bibliotecas de análisis de datos más útiles en Python. Han sido fundamentales para aumentar el uso de Python en la comunidad de ciencia de datos.

Antes de cargar los datos, vamos a entender las 2 estructuras de datos clave en Pandas: Series y DataFrames

Introducción a Series y Dataframes

La serie se puede entender como una matriz unidimensional etiquetada / indexada. Puede acceder a elementos individuales de esta serie a través de estas etiquetas.

Un dataframe es similar al libro de Excel: tiene nombres de columnas que hacen referencia a columnas y tiene filas, a las que se puede acceder mediante el uso de números de fila. La diferencia esencial es que los nombres de columna y los números de fila se conocen como índice de columna y fila, en el caso de dataframe.

Las series y dataframes forman el modelo de datos básico para Pandas en Python. Los conjuntos de datos se leen primero en estos cuadros de datos y luego se pueden aplicar fácilmente varias operaciones (por ejemplo, agrupar, agregación, etc.) a sus columnas.

<http://pandas.pydata.org/pandas-docs/stable/10min.html>

Ahora usaremos Pandas para leer un conjunto de datos de una competencia de [Analytics Vidhya](#), realizar un análisis exploratorio y construir nuestro primer algoritmo básico de categorización para resolver el problema.

Conjunto de datos de práctica: problema de predicción de préstamos

La compañía Dream Housing Finance se ocupa de préstamos hipotecarios. Tienen presencia en todas las áreas urbanas, semi urbanas y rurales. El cliente primero solicita un préstamo hipotecario luego de que la compañía valida la elegibilidad del cliente para el préstamo.

La empresa desea automatizar el proceso de elegibilidad del préstamo (en tiempo real) en función del detalle del cliente que se le proporcionó al completar el formulario de solicitud en línea. Estos detalles son Género, Estado civil, Educación, Número de dependientes, Ingresos, Monto del préstamo, Historial de crédito y otros. Para automatizar este proceso, han dado un problema para **identificar los segmentos de clientes que son elegibles para el monto del préstamo** para que puedan dirigirse específicamente a estos clientes.

Variable	Description
Loan_ID	Unique Loan ID (ID préstamo)
Gender	Male/ Female (Mas/Fem)
Married	Applicant married (Y/N)
Dependents	Number of dependents
Education	Applicant Education (Graduate/ Under Graduate)
Self_Employed	Self employed (Y/N)
ApplicantIncome	Applicant income (Ingresos)
CoapplicantIncome	Coapplicant income
LoanAmount	Loan amount in thousands (Monto del préstamo en miles)
Loan_Amount_Term	Term of loan in months (Duración en meses del préstamo)
Credit_History	credit history meets guidelines
Property_Area	Urban/ Semi Urban/ Rural
Loan_Status	Loan approved (Y/N)

Dataset: <https://drive.google.com/file/d/1x0x7IoyAyH94PugfASgofc256QOxxbA2/view>

Manos a la obra

Este ejemplo se realizará sobre el sistema operativo Linux, siendo su distribución Ubuntu 16.04, y se utilizará como lenguaje de programación Python 2.7

Instalación de bibliotecas a utilizar:

```
sudo apt-get install python-dev python-pip  
sudo pip install pandas jupyter matplotlib sklearn scipy
```

Una vez instalado las herramientas anteriores ejecutaremos en la terminal la siguiente orden:

```
jupyter notebook
```

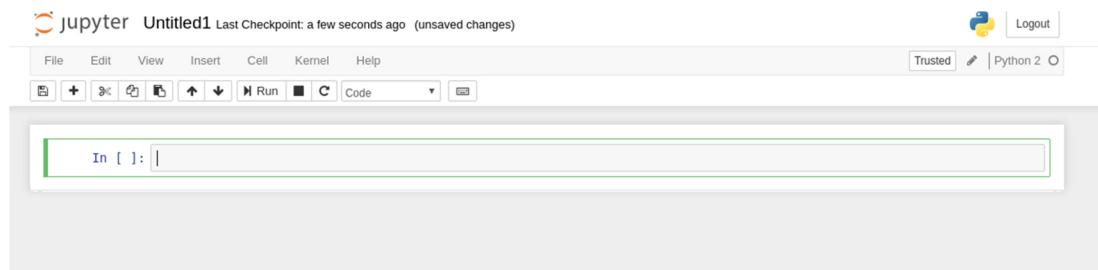
y aparecerá una pantalla similar a la siguiente :



Iremos a “New” y Elegiremos “Python 2”



Luego aparecerá una pantalla como la siguiente:



Esta pantalla nos da la posibilidad de ir colocando código en la entrada “In” y ejecutarlo con “Run” o la combinación de teclas Ctrl + Enter.

Comencemos:

1 - Importación de bibliotecas y lectura de nuestro dataset:

```
In [ ]: import pandas as pd
import numpy as np
import matplotlib as plt

df = pd.read_csv("path_download_file/train.csv")
```

2 - Viendo algunos registros de la cabecera del dataset (método head()):

```
In [4]: df.head(5)
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoaapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0
1	LP001003	Male	Yes	1	Graduate	No	4532	1000.0	120.0	360.0	1.0
2	LP001004	Male	Yes	0	Graduate	Yes	3600	0.0	60.0	360.0	1.0
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0

3- Método describe: proporcionará count, mean, standard deviation (std), min, cuartiles y max en su salida

```
In [5]: df.describe()
Out[5]:
   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  Credit_History
count      614.000000      614.000000  592.000000      600.000000      564.000000
mean       5403.459283     1621.245798  146.412162      342.000000      0.842199
std        6109.041673    2926.248369    85.587325      65.12041      0.364878
min        150.000000      0.000000    9.000000      12.00000      0.000000
25%       2877.500000      0.000000  100.000000      360.00000      1.000000
50%       3812.500000    1188.500000    128.000000      360.00000      1.000000
75%       5795.000000    2297.250000    168.000000      360.00000      1.000000
max       81000.000000   41667.000000   700.000000      480.00000      1.000000
```

Aquí hay algunas inferencias, puede analizar mirando el resultado del método describe ():

1. Monto del préstamo (LoanAmount) tiene (614 - 592) 22 valores perdidos.
2. Meses del préstamo (Loan_Amount_Term) tiene (614 - 600) 14 valores faltantes.
3. Credit_History tiene (614 - 564) 50 valores perdidos.
4. También podemos observar que alrededor del 84% de los solicitantes tienen una credit_history. ¿Cómo? La media del campo Credit_History es 0.84 (Recuerde, Credit_History tiene valor 1 para aquellos que tienen un historial de crédito y 0 de lo contrario)
5. La distribución de ingresos () del solicitante parece estar en línea con las expectativas. Lo mismo con CoapplicantIncome

Tenga en cuenta que podemos tener una idea de un posible sesgo en los datos al comparar la media con la mediana, es decir, la cifra del 50%.

Para los valores no numéricos (por ejemplo, Property_Area, Credit_History, etc.), podemos ver la distribución de frecuencias para comprender si tienen sentido o no. La tabla de frecuencias se puede imprimir con el siguiente comando:

```
In [6]: df['Property_Area'].value_counts()
Out[6]:
Semiurban    233
Urban        202
Rural         179
Name: Property_Area, dtype: int64
```

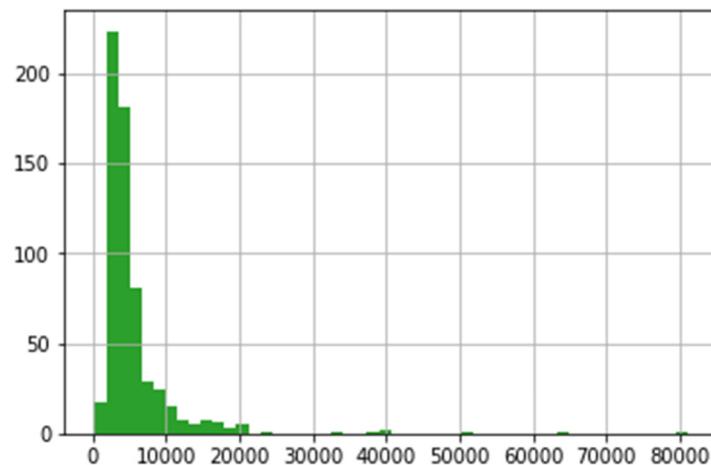
Del mismo modo, podemos mirar los valores únicos del historial de crédito. Tenga en cuenta que df ['column_name'] es una técnica de indexación básica para acceder a una columna particular del dataframe.

Análisis de distribución

Ahora que estamos familiarizados con las características básicas de los datos, estudiemos la distribución de diversas variables. Comencemos con las variables numéricas, a saber, ApplicantIncome y LoanAmount

Comencemos trazando el histograma de ApplicantIncome usando los siguientes comandos:

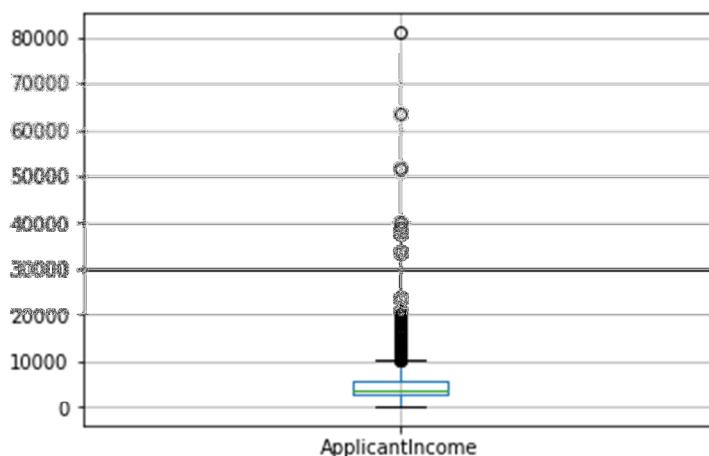
```
In [10]: df['ApplicantIncome'].hist(bins=50)  
plt.pyplot.show()
```



Aquí observamos que hay pocos valores extremos. Esta es también la razón por la cual se requieren 50 bins para representar claramente la distribución.

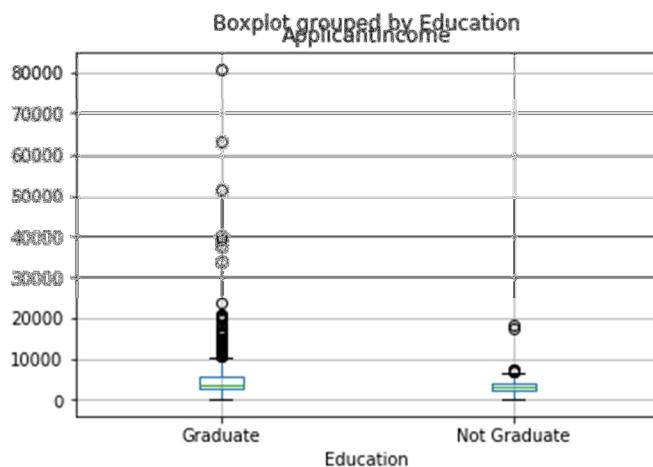
A continuación, observamos los diagramas de caja para comprender las distribuciones. El diagrama de caja se puede trazar de la siguiente manera:

```
In [4]: df.boxplot(column='ApplicantIncome')  
plt.pyplot.show()
```



Esto confirma la presencia de muchos valores atípicos / extremos. Esto se puede atribuir a la disparidad de ingresos en la sociedad. Parte de esto puede ser impulsado por el hecho de que estamos viendo personas con diferentes niveles de educación. Vamos a segregarlos por educación:

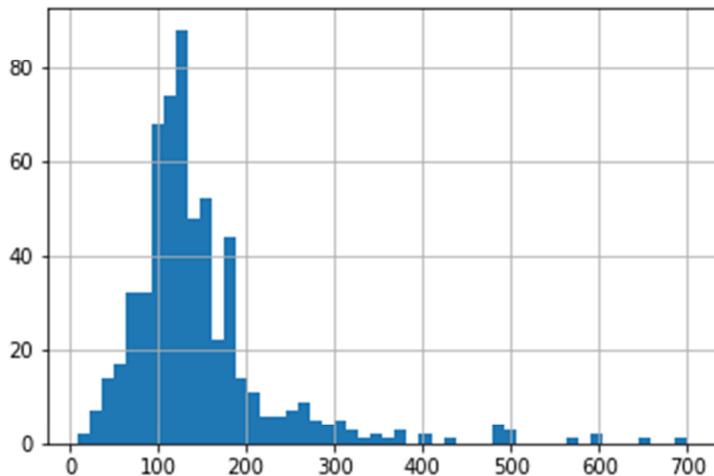
```
In [5]: df.boxplot(column='ApplicantIncome', by = 'Education')
plt.pyplot.show()
```



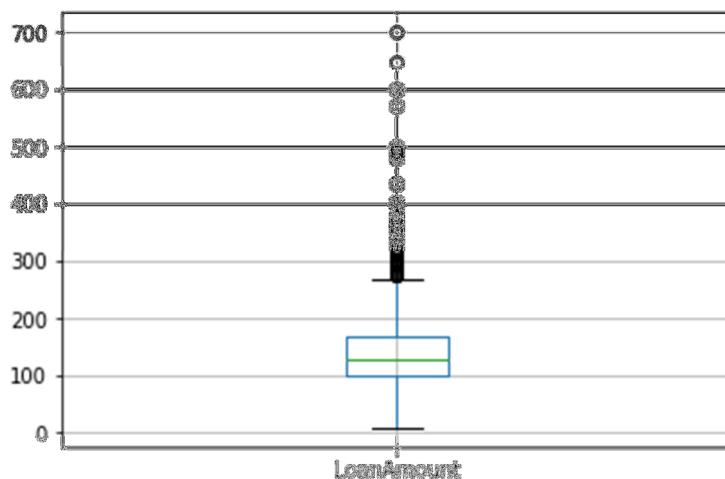
Podemos ver que no hay diferencias sustanciales entre los ingresos medios de los graduados y los no graduados. Pero hay un mayor número de graduados con ingresos muy altos, que parecen ser los valores atípicos.

Ahora, veamos el histograma y el diagrama de caja de LoanAmount usando el siguiente comando:

```
In [6]: df['LoanAmount'].hist(bins=50)
plt.pyplot.show()
```



```
In [7]: df.boxplot(column='LoanAmount')
plt.pyplot.show()
```



Nuevamente, hay algunos valores extremos. Claramente, tanto ApplicantIncome como LoanAmount requieren una cierta cantidad de munging de datos. LoanAmount tiene valores valores perdidos y también extremos, mientras que ApplicantIncome tiene algunos valores extremos, que exigen una comprensión más profunda.

Análisis de variables categóricas

Ahora que entendemos las distribuciones para ApplicantIncome y LoanIncome, vamos a comprender las variables categóricas en más detalle. Utilizaremos la tabla dinámica de estilo de Excel y la tabulación cruzada. Por ejemplo, veamos las posibilidades de obtener un préstamo basado en el historial crediticio.

Nota: aquí el estado del préstamo ha sido codificado como 1 para Sí y 0 para No. Por lo tanto, la media representa la probabilidad de obtener un préstamo.

```
In [14]: temp1 = df['Credit_History'].value_counts(ascending=True)
temp2 = df.pivot_table(values='Loan_Status',index=['Credit_History'],aggfunc=lambda x: x.map({'Y':1,'N':0}).mean())
print 'Tabla de frecuencia para historial de crédito:'
print temp1
print
print 'Probabilidad de obtener un préstamo para cada clase de historial de crédito:'
print temp2
```

Tabla de frecuencia para historial de crédito:

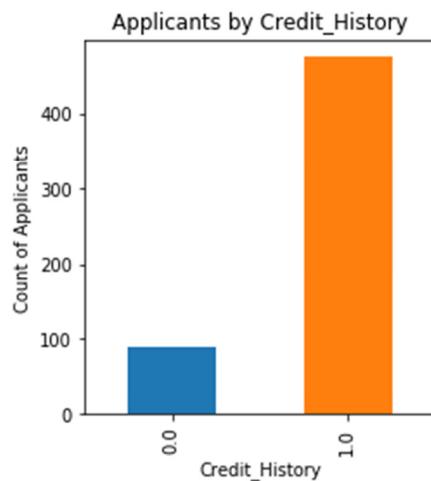
Credit_History	Count
0.0	89
1.0	475

Probabilidad de obtener un préstamo para cada clase de historial de crédito:

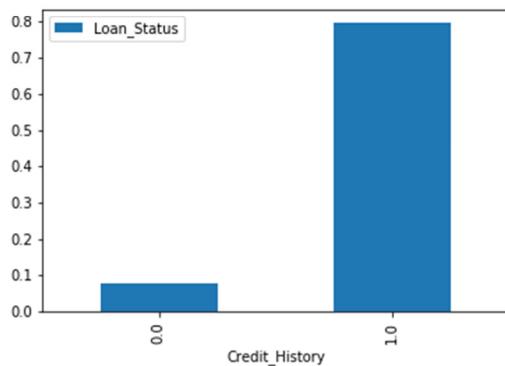
Credit_History	Loan_Status	Probability
0.0	Y	0.078652
1.0	Y	0.795789

Esto se puede trazar como un gráfico de barras usando la biblioteca "matplotlib" con el siguiente código:

```
In [7]: import matplotlib.pyplot as plt
fig = plt.figure(figsize=(8,4))
ax1 = fig.add_subplot(121)
ax1.set_xlabel('Credit_History')
ax1.set_ylabel('Count of Applicants')
ax1.set_title("Applicants by Credit_History")
temp1.plot(kind='bar')
plt.show()
```



```
In [90]: temp2.plot(kind = 'bar')# temp2 tabla pivot
plt.show()
```



Esto muestra que las posibilidades de obtener un préstamo son ocho veces mayores si el solicitante tiene un historial crediticio válido. Podríamos trazar gráficos similares por Casado, Independiente, Propiedad_Area, etc.

Alternativamente, estos dos gráficos también se pueden visualizar combinándolos en un gráfico apilado:

```
In [91]: temp3 = pd.crosstab(df['Credit_History'], df['Loan_Status'])
temp3.plot(kind='bar', stacked=True, color=['red','blue'], grid=False)
plt.show()
```



Si aún no nos hemos dado cuenta, acabamos de crear dos algoritmos básicos de clasificación, uno basado en el historial de crédito, mientras que otro en 2 variables categóricas (incluida la educación).

Acabamos de ver cómo podemos hacer un análisis exploratorio en Python usando Pandas. Espero que el amor por los pandas (el animal) se haya incrementado a esta altura, dada la cantidad de ayuda que la biblioteca puede brindarle al analizar los conjuntos de datos.

A continuación, exploraremos las variables ApplicantIncome y LoanStatus, realizando data munging y creando un conjunto de datos para aplicar diversas técnicas de modelado.

Data Munging en Python: usando Pandas

Para aquellos que han estado siguiendo, aquí están los zapatos que deben usar para comenzar a correr.

Mientras exploramos los datos, encontramos algunos problemas en el conjunto de datos, que deben resolverse antes de que los datos estén listos para un buen modelo. Este ejercicio se conoce generalmente como "Data Munging". Aquí están los problemas, de los que ya somos conscientes:

- Faltan valores en algunas variables. Deberíamos estimar esos valores sabiamente según la cantidad de valores perdidos y la importancia esperada de las variables. Al observar las distribuciones, vimos que ApplicantIncome y LoanAmount parecían contener valores extremos en cada extremo. Aunque pueden tener un sentido intuitivo, pero deben tratarse adecuadamente.

Además de estos problemas con los campos numéricos, también debemos ver los campos no numéricos, es decir, Género, Área de la propiedad, Casado, Educación y Dependientes para ver si contienen información útil.

- Verificar los valores perdidos en el conjunto de datos: veamos los valores perdidos en todas las variables porque la mayoría de los modelos no funcionan con datos faltantes. Entonces, revisemos el número de nulos / NaN en el conjunto de datos.

```
In [92]: df.apply(lambda x: sum(x.isnull()),axis=0)
Out[92]: Loan_ID          0
Gender           13
Married          3
Dependents      15
Education        0
Self_Employed   32
ApplicantIncome  0
CoapplicantIncome 0
LoanAmount       22
Loan_Amount_Term 14
Credit_History   50
Property_Area    0
Loan_Status       0
dtype: int64
```

Aunque los valores perdidos no son muy altos en número, muchas variables los tienen y cada uno de ellos debe estimarse y agregarse en los datos.

Nota: Recordemos que los valores perdidos pueden no ser siempre NaN. Por ejemplo, si Loan_Amount_Term es 0, ¿tiene sentido o consideraría que falta? Supongo que la respuesta es falta y con razón. Entonces debemos verificar valores que no son prácticos.

¿Cómo completar los valores perdidos en LoanAmount?

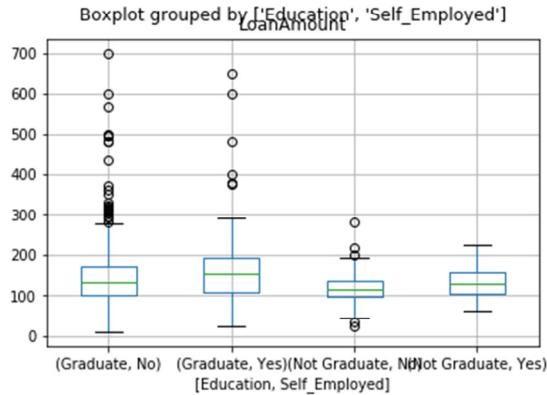
Existen numerosas formas de completar los valores faltantes del monto del préstamo, siendo el más simple el reemplazo por la media, que se puede hacer mediante el siguiente código:

```
df['LoanAmount'].fillna(df['LoanAmount'].mean(), inplace=True)
```

El otro extremo podría ser construir un modelo de aprendizaje supervisado para predecir el monto del préstamo sobre la base de otras variables y luego usar la edad junto con otras variables para predecir la supervivencia. Dado que, el objetivo ahora es resaltar los pasos en la reducción de datos, preferiría adoptar un enfoque, que se encuentra en algún lugar entre estos 2 extremos. Una hipótesis clave es que si una persona es educada o trabaja por cuenta propia puede combinarse para dar una buena estimación del monto del préstamo.

Primero, veamos el diagrama de caja para ver si existe una tendencia:

```
In [97]: df.boxplot(column='LoanAmount', by = ['Education', 'Self_Employed'])
plt.show()
```



Por lo tanto, observamos algunas variaciones en la mediana del monto del préstamo para cada grupo y esto puede usarse para imputar los valores. Pero primero, debemos asegurarnos de que cada una de las variables de Self_Employed y Education no deben tener valores faltantes.

Como decimos antes, Self_Employed tiene algunos valores faltantes. Veamos la tabla de frecuencias:

```
In [99]: df["Self_Employed"].value_counts()
```

```
Out[99]: No      500
          Yes     82
          Name: Self_Employed, dtype: int64
```

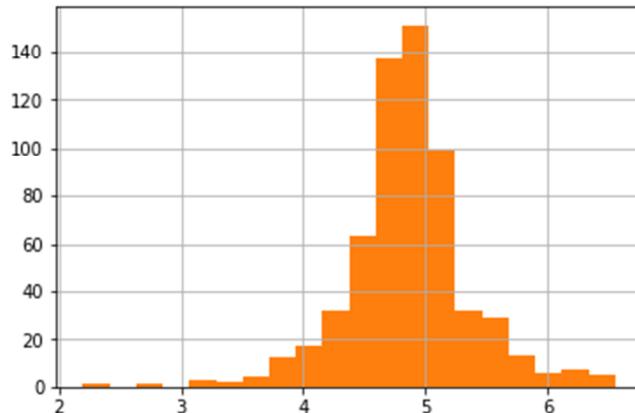
Como el 86% de los valores son "No", es seguro imputar los valores faltantes como "No", ya que hay una alta probabilidad de éxito. Esto se puede hacer usando el siguiente código:

```
df['Self_Employed'].fillna('No', inplace=True)
```

¿Cómo tratar los valores extremos en la distribución de LoanAmount y ApplicantIncome?

Analicemos LoanAmount primero. Dado que los valores extremos son prácticamente posibles, es posible que algunas personas soliciten préstamos de alto valor debido a necesidades específicas. Entonces, en lugar de tratarlos como valores atípicos, probemos una transformación de registro para anular su efecto:

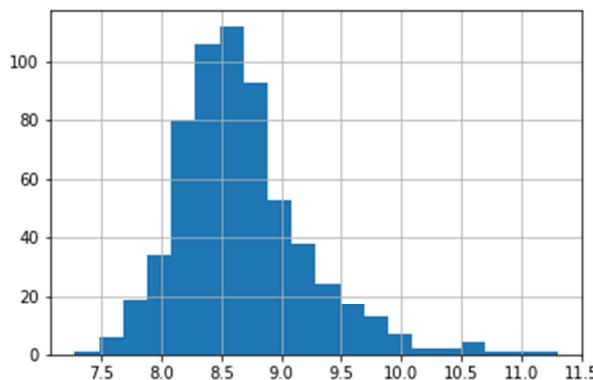
```
In [168]: df['LoanAmount_log'] = np.log(df['LoanAmount'])
df['LoanAmount_log'].hist(bins=20)
plt.show()
```



Ahora la distribución se ve mucho más cerca de lo normal y el efecto de los valores extremos ha disminuido significativamente.

Una intuición puede ser que algunos solicitantes tienen un ingreso más bajo pero son fuertes solicitantes con Co-applicants. Por lo tanto, podría ser una buena idea combinar ambos ingresos como ingreso total y tomar una transformación de registro de los mismos.

```
In [12]: df['TotalIncome'] = df['ApplicantIncome'] + df['CoapplicantIncome']
df['TotalIncome_log'] = np.log(df['TotalIncome'])
df['TotalIncome_log'].hist(bins=20)
plt.show()
```



Ahora vemos que la distribución es mucho mejor que antes. Lo dejo para que pueda imputar los valores que faltan para Género, Casado, Dependientes, Loan_Amount_Term, Credit_History. Además, lo aliento a que piense en la posible información adicional que puede derivarse de los datos. Por ejemplo, crear una columna para $\text{LoanAmount} / \text{TotalIncome}$ podría tener sentido, ya que da una idea de qué tan bien el solicitante puede pagar su préstamo.

Algunos conceptos teóricos:

La validación cruzada o cross-validation es una técnica utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de la partición entre datos de entrenamiento y prueba. Consiste en repetir y calcular la media aritmética obtenida de las medidas de evaluación sobre diferentes particiones. Se utiliza en entornos donde el objetivo principal es la predicción y se quiere estimar la precisión de un modelo que se llevará a cabo a la práctica. Es una técnica muy utilizada en proyectos de inteligencia artificial para validar modelos generados [\[**\]](#).

Construyendo un modelo predictivo en Python

Después, hemos hecho que los datos sean útiles para el modelado, veamos ahora el código Python para crear un modelo predictivo en nuestro conjunto de datos. Skicit-Learn (sklearn) es la librería más utilizada en Python para este propósito y seguiremos el camino.

Como, sklearn requiere que todas las entradas sean numéricas, debemos convertir todas nuestras variables categóricas en numéricas codificando las categorías. Esto se puede hacer usando el siguiente código:

Aclaración: recuerde aplicar “data munging” a las variables utilizadas.

```
In [15]: from sklearn.preprocessing import LabelEncoder
var_mod = [
    'Gender',
    'Married',
    'Dependents',
    'Education',
    'Self_Employed',
    'Property_Area',
    'Loan_Status'
]
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i])
df.dtypes
```

```
Out[15]: Loan_ID          object
Gender            int64
Married           int64
Dependents        int64
Education         int64
Self_Employed     int64
ApplicantIncome   int64
CoapplicantIncome float64
LoanAmount        float64
Loan_Amount_Term  float64
Credit_History    float64
Property_Area     int64
Loan_Status        int64
LoanAmount_log    float64
TotalIncome       float64
TotalIncome_log   float64
dtype: object
```

A continuación, importaremos los módulos requeridos. Luego definiremos una función de clasificación genérica, que toma un modelo como entrada y determina los puntajes de Precisión y Validación Cruzada. Dado que este es un artículo introductorio, no entraremos en los detalles de la codificación.

```
In [16]: #Import models from scikit learn module:  
from sklearn.linear_model import LogisticRegression  
from sklearn.cross_validation import KFold #For K-fold cross validation  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.tree import DecisionTreeClassifier, export_graphviz  
from sklearn import metrics  
  
#Generic function for making a classification model and accessing performance:  
def classification_model(model, data, predictors, outcome):  
    #Fit the model:  
    model.fit(data[predictors],data[outcome])  
  
    #Make predictions on training set:  
    predictions = model.predict(data[predictors])  
  
    #Print accuracy  
    accuracy = metrics.accuracy_score(predictions,data[outcome])  
    print "Accuracy : %s" % "{0:.3%}".format(accuracy)  
  
    #Perform k-fold cross-validation with 5 folds  
    kf = KFold(data.shape[0], n_folds=5)  
    error = []  
    for train, test in kf:  
        # Filter training data  
        train_predictors = (data[predictors].iloc[train,:])  
  
        # The target we're using to train the algorithm.  
        train_target = data[outcome].iloc[train]  
  
        # Training the algorithm using the predictors and target.  
        model.fit(train_predictors, train_target)  
  
        #Record error from each cross-validation run  
        error.append(model.score(data[predictors].iloc[test,:], data[outcome].iloc[test]))  
  
    print "Cross-Validation Score : %s" % "{0:.3%}".format(np.mean(error))  
  
    #Fit the model again so that it can be referred outside the function:  
    model.fit(data[predictors],data[outcome])
```

Logistic Regression

Hagamos nuestro primer modelo de logistic regression. Una forma sería tomar todas las variables en el modelo, pero esto podría resultar en un ajuste excesivo (no nos preocupemos si todavía no conocemos esta terminología). En palabras simples, tomar todas las variables puede hacer que el modelo comprenda relaciones complejas específicas a los datos y no generalice bien.

Podemos hacer fácilmente algunas hipótesis intuitivas para poner todo en marcha. Las posibilidades de obtener un préstamo serán mayores para:

- Los solicitantes tienen un historial de crédito (¿recuerdan que observamos esto en la exploración?)
- Solicitantes con mayores ingresos de solicitantes y co-solicitantes

- Solicitantes con nivel de educación superior
- Propiedades en áreas urbanas con altas perspectivas de crecimiento

Así que vamos a hacer nuestro primer modelo con 'Credit_History'.

```
In [21]: predictor_var = [
    'Credit_History',
    'Education',
    'Married',
    'Self_Employed',
    'Property_Area'
]
classification_model(model, df,predictor_var,outcome_var)

Accuracy : 80.945%
Cross-Validation Score : 80.946%
```

En general, esperamos que la precisión aumente al agregar variables. Pero este es un caso más desafiante. La precisión y la puntuación de validación cruzada no se ven afectadas por variables menos importantes. Credit_History está dominando el modo. Tenemos dos opciones ahora:

1. Feature Engineering: derivada de información nueva e indirecta e intentar predecirla.
Dejaré esto a tu creatividad.
2. Mejores técnicas de modelado. Exploraremos esto a continuación.

Árbol de decisión

El árbol de decisiones es otro método para hacer un modelo predictivo. Se sabe que proporciona una mayor precisión que el modelo de regresión logística.

```
In [23]: model = DecisionTreeClassifier()
predictor_var = [
    'Credit_History',
    'Gender',
    'Married',
    'Education'
]
classification_model(model, df,predictor_var,outcome_var)

Accuracy : 81.270%
Cross-Validation Score : 80.784%
```

Aquí el modelo basado en variables categóricas no puede tener un impacto porque el historial de crédito está dominando sobre ellos. Probaremos algunas variables numéricas:

```
In [26]: #We can try different combination of variables:
predictor_var = [
    'Credit_History',
    'Loan_Amount_Term',
    'LoanAmount_log'
]
classification_model(model, df,predictor_var,outcome_var)
```

Accuracy : 89.414%
Cross-Validation Score : 67.909%

Aquí observamos que, aunque aumentó la precisión al agregar variables, el error de validación cruzada disminuyó. Este es el resultado del modelo que sobreajusta los datos. Probemos con un algoritmo aún más sofisticado y veamos si ayuda:

Random Forest

El bosque aleatorio es otro algoritmo para resolver el problema de clasificación.

Una ventaja de Random Forest es que podemos hacer que funcione con todas las características y devuelve una matriz de importancia de características que se puede usar para seleccionar características.

```
In [28]: model = RandomForestClassifier(n_estimators=100)
predictor_var = [
    'Gender',
    'Married',
    'Dependents',
    'Education',
    'Self_Employed',
    'Loan_Amount_Term',
    'Credit_History',
    'Property_Area',
    'LoanAmount_log',
    'TotalIncome_log']
classification_model(model, df,predictor_var,outcome_var)
```

Accuracy : 100.000%
Cross-Validation Score : 78.181%

Aquí vemos que la precisión es del 100% para el conjunto de entrenamiento. Este es el último caso de sobreajuste y se puede resolver de dos maneras:

- Reducir el número de predictores
- Ajuste de los parámetros del modelo

Probemos ambos. Primero vemos la matriz de importancia de las características de la cual tomaremos las características más importantes.

```
In [30]: #Create a series with feature importances:
featimp = pd.Series(
    model.feature_importances_,
    index=predictor_var).sort_values(
        ascending=False)
print featimp
```

Credit_History	0.275683
TotalIncome_log	0.256707
LoanAmount_log	0.220565
Dependents	0.057488
Property_Area	0.045958
Loan_Amount_Term	0.043757
Gender	0.029176
Married	0.027488
Education	0.023005
Self_Employed	0.020174

dtype: float64

Usemos las 5 variables principales para crear un modelo. Además, modificaremos un poco los parámetros del modelo de bosque aleatorio:

```
In [31]: model = RandomForestClassifier(
    n_estimators=25,
    min_samples_split=25,
    max_depth=7,
    max_features=1
)
predictor_var = [
    'TotalIncome_log',
    'LoanAmount_log',
    'Credit_History',
    'Dependents',
    'Property_Area']
classification_model(model, df,predictor_var,outcome_var)
```

Accuracy : 82.899%
Cross-Validation Score : 80.297%

Tenga en cuenta que se redujo la precisión, pero la puntuación de validación cruzada está mejorando, lo que demuestra que el modelo se está generalizando bien. Recuerde que los modelos de bosque aleatorio no son exactamente repetibles. Las diferentes ejecuciones darán lugar a ligeras variaciones debido a la aleatorización. Pero la salida debería permanecer en el estadio.

Habrá notado que incluso después de un ajuste básico de parámetros en bosque aleatorio, hemos alcanzado una precisión de validación cruzada ligeramente mejor que el modelo de regresión logística original. Este ejercicio nos brinda un aprendizaje muy interesante y único:

- Usar un modelo más sofisticado no garantiza mejores resultados.
- Evite utilizar técnicas complejas de modelado como una caja negra sin entender los conceptos subyacentes. Si lo hace, aumentará la tendencia de sobreajuste, lo que hará que sus modelos sean menos interpretables.

Caso Práctico con R

R es un lenguaje orientado a comandos. En la siguiente imagen podemos observar como es el ambiente de trabajo que brinda

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, and Addins. The left sidebar has tabs for 'trabajando con objetos.R', 'datos', 'mi_csv', and 'Tarea_2'. The main area contains two code snippets:

```
18 #####
19
20 rm(list=ls())
21
22 atendidos <- c(728,632,596,689,745,865,694,583,657,643,794
23 #atendidas mensualmente en el servicio de urgencias de un
24 atendidos
25
26 class(atendidos) # nos devuelve la clase del objeto
27
28 x=1
29 class(x)
30
31 y="a"
32 class(y)
33
34 #####
35 # Además R dispone de otras clases de datos más específica
36 # para números enteros, complex para números complejos, lo
37 # o matrix para matrices #####
38 |
39 <-->
38:1 o matriz para matrices
```

```
0
> letras
[1] "a" "b" "c" "d" "e"
> class(letras)
[1] "character"
> mezcla=c(1,2,3,"a","b","c") # si en un vector se combinan ele
mentos de clase numeric y character el vector resultante es de
clase character
> mezcla
[1] "1" "2" "3" "a" "b" "c"
> x=c(1,4,5,2,4,5,4,3,2,2,3,2,2,4,4,5,5,6,6,7)
> length(x) # La función length() devuelve la longitud (número
de elementos) del vector
[1] 20
> peso=c(77,68,85,NA,73) # se ha perdido el cuarto valor
> |
```

The right panel shows the 'Environment' browser with variables and their types:

Variables	Type	Value
atendidos	num	[1:12] 728 632 596 689 745 865 6...
edad	num	[1:10] 22 34 29 25 30 33 31 27 2...
letras	chr	[1:5] "a" "b" "c" "d" "e"
mezcla	chr	[1:6] "1" "2" "3" "a" "b" "c"
peso	num	[1:5] 77 68 85 NA 73
x	num	[1:20] 1 4 5 2 4 5 4 3 2 2 ...
y	char	"a"

Ejecutando comandos en R, versión 3.5.1.

En este ejemplo vamos a tomar una base de datos que provee R en su distribución estándar, y realizar la regresión lineal de los datos. Usaremos [Mammals de la librería MASS](#), que incluye información de animales considerando el peso del cuerpo y del cerebro de 62 especies.

Antes de comenzar, tengamos en cuenta los comandos más comunes utilizados en R.

Comando	Descripción
library(faraway)	Librería a utilizar
data(worldcup)	Base de datos de la librería
mydata=worldcup	Creamos la referencia mydata para la BD
fix(mydata)	Muestra los datos de la BD
names(mydata)	Muestra los nombres de las columnas de la BD
attach(mydata)	Indicamos que solo usamos los datos de este paquete, para no poner mydata\$Time por ejemplo
install.packages("httpuv")	Para instalar un paquete
table(muydata\$Team)	Vemos los valores que hay en la columna Team de mydata
summary(mydata)	Nos da un resumen de los datos que tenemos, como promedio, media, etc
mydata=mydata[-34,]	Es para eliminar el elemento que esta en la posición 34 en la BD, se debe tener en cuenta el orden de eliminado
> distancia=c(3,2,1,4)	Genera una matriz en R
> tiempo=c(7,5,3,9)	
> base=cbind(distancia,tiempo)	
> base	
> mydata\$index=NULL	para sacar una columna de los datos

Los siguientes son comandos para trabajar con diagramas:

plot(mydata\$Time, mydata\$Tackles, col=mydata\$Position)	Muestra un gráfico con colores según Position
identify(mydata\$Time, muydata\$Tackles)	Debe haber un plot antes de identify, hace que clickeando sobre un punto veamos a que dato está representando
plot(muydata\$Time, muydata\$Tackles, col=muydata\$Position, pch=19, cex=1, lty="solid", lwd=10)	Muestra un diagrama en la cual se configuran los tamaños, estos comandos se utilizan para exploración de datos
plot(md\$Bwt, md\$Hwt, col=md\$Sex)	
plot(regresion)	
plot3d(Time, Tackles, Passes)	Muestra el gráfico en 3d y se puede girar con el mouse
hist(temp, col="red")	Dibuja el histograma de la variable en color rojo
par(mfrow=c(2,2))	Divide la pantalla en 4
abline(regresion)	Muestra la línea de regresión
legend('topright', levels(Sex), lty=1, col=1:2)	Muestra las referencias en el gráfico

En R, se cuenta con diferentes bases de datos para realizar distintos modelos. En este ejemplo vamos a utilizar dentro de la librería MASS los datos de mammals. R es case sensitive (es importante usar correctamente mayúsculas y minúsculas).

Para comenzar ejecutamos R Studio, y se cargan las siguientes librerías:

```
> library(MASS)
> data(mammals)
> mydata=mammals
> attach(mydata)
```

Este ejemplo tiene dos tipos de OUTLIER, entonces vamos a:

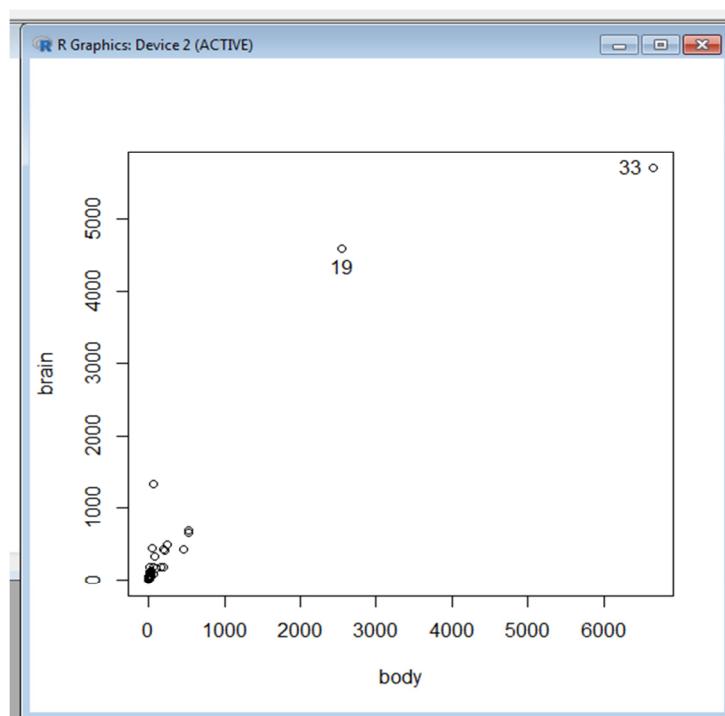
- sacar a los dos elefantes y hacer la regresión
- sacar al humano y ver si mejora la regresión

a)

```
summary(mydata)

  body        brain
Min. : 0.005  Min. : 0.14
1st Qu.: 0.600  1st Qu.: 4.25
Median : 3.342  Median : 17.25
Mean   :198.790  Mean   :283.13
3rd Qu.: 48.203  3rd Qu.:166.00
Max.   :6654.000  Max.   :5712.00
```

Los que tienen valor de body y brain mayor a Max. :6654.000 Max. :5712.00 son 19 y 33, como se observa en el siguiente gráfico, que logramos ejecutando plot(mydata)



Entonces, sacamos de la muestra los datos de las posiciones 33 y 19, en ese orden. Esto es muy importante porque los datos están almacenados en un arreglo.

```
> mydata=mydata[-33,]
> mydata=mydata[-19,]
> regresion=lm(body~brain, mydata)
> regresion
Call:
lm(formula = body ~ brain, data = mydata)

Coefficients:
(Intercept)      brain
10.9751        0.3406
```

```
> attach(mydata)
The following objects are masked from mydata (pos = 3):
```

body, brain

```
> cor(body, brain)
[1] 0.6505592
> cor(brain, body)
[1] 0.6505592

> regresion=lm(body~brain, md)
> regresion
Call:
lm(formula = body ~ brain, data = md)

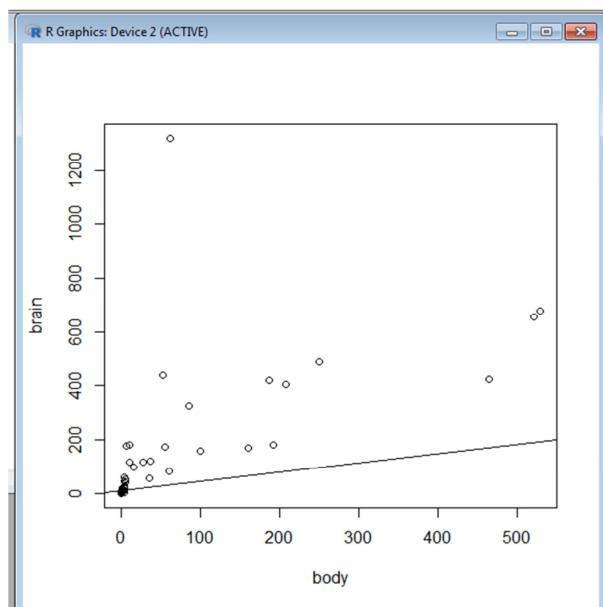
Coefficients:
(Intercept)      brain
10.9751        0.3406
```

Body = 0.3406 * brain + 10.9751

```
> plot(body, brain)
> abline(regresion)

> summary(regresion)
```

```
Call:
lm(formula = body ~ brain, data = md)
```



Residuals:

Min	1Q	Median	3Q	Max
-398.52	-15.59	-12.11	-11.05	309.96

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	10.9751	13.3881	0.820	0.416
brain	0.3406	0.0522	6.524	1.84e-08 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 91.51 on 58 degrees of freedom

Multiple R-squared: **0.4232**, Adjusted R-squared: 0.4133

F-statistic: 42.56 on 1 and 58 DF, p-value: 1.84e-08

b) sacamos al humano de la muestra

```
> fix(mydata)
> mydata=mydata[-31,]
> attach(mydata)
```

The following objects are masked from mydata (pos = 3):

body, brain

The following objects are masked from mydata (pos = 4):

body, brain

```
> cor(brain, body)
[1] 0.8884084
```

```
> regresion=lm(brain~body, mydata)
```

```
> regresion
```

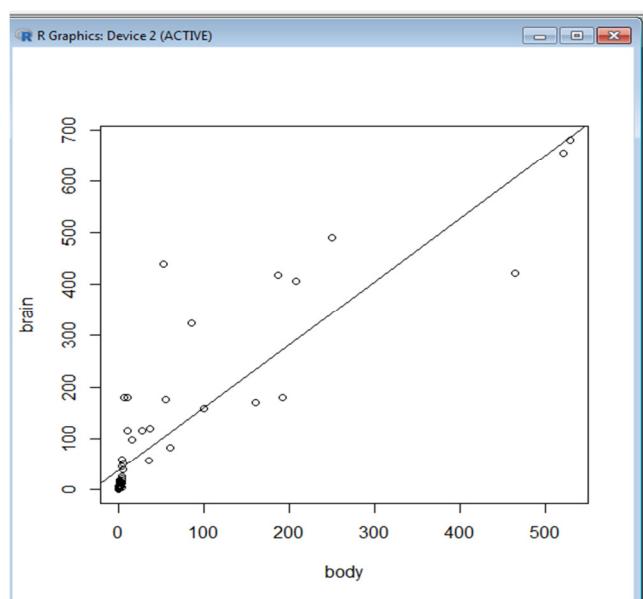
Call:

```
lm(formula = brain ~ body, data = mydata)
```

Coefficients:

(Intercept)	body
36.572	1.228

```
> plot(brain, body)
```



```

> abline(regresion)
> plot(body,brain)
> abline(regresion)

> summary(regresion)

```

Call:

```
lm(formula = brain ~ body, data = mydata)
```

Residuals:

Min	1Q	Median	3Q	Max
-184.81	-34.52	-27.16	0.67	339.35

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	36.57231	10.95089	3.34	0.00148 **
body	1.22847	0.08408	14.61	< 2e-16 ***

Signif. codes:	0 ‘***’	0.001 ‘**’	0.01 ‘*’	0.05 ‘.’
	0.1 ‘ ’	1		

Residual standard error: 77.15 on 57 degrees of freedom

Multiple R-squared: **0.7893**, Adjusted R-squared: 0.7856

F-statistic: 213.5 on 1 and 57 DF, p-value: < 2.2e-16

Conclusión:

Sacando al humano mejora Multiple R-squared ya que está más cerca de 1 y Pr(>|t|) ya que es menor de 0.05 Si tenemos un nuevo bichito y queremos saber el peso de su cerebro, podemos reemplazar el valor en la fórmula de la recta que tuvimos o bien utilizar la función predict

```

> prediccion=predict(regresion,data.frame(body=4))
> prediccion

```

41.4862 //valor del peso de cerebro del bichito nuevo

```

> summary(prediccion)
Min. 1st Qu. Median Mean 3rd Qu. Max.
41.49 41.49 41.49 41.49 41.49 41.49

```

Si hacemos

```

> prediccion=predict(regresion,data.frame(body=100)) nos da 159.4 que es similar al que
está en la posición 42 de fix(md)

```

Haciendo un resumen de lo visto hasta ahora:

Hasta ahora tenemos variables independientes y dependiente, si la variable dependiente (es la que queremos predecir) es cualitativa entonces tenemos un problema de clasificación, si la variable es cuantitativa entonces el problema es de regresión. Si no hay variable dependiente el tipo de problema es de agrupamiento o segmentación.

Regresión lineal, si no ajusta la recta obtenida podemos hacer:

- Sacar registros outliers.
- Agregar registros para mejorar la muestra.
- Cambiar de método, por ejemplo una red neuronal. La mayoría de los métodos permiten hacer regresión y clasificación, excepto que la regresión lineal solo permite hacer problemas de regresión.
- Transformar variables, por ejemplo elevar al cuadrado una de las variables.
- Sacar variables, por ejemplo por correlación (no son significativas). Si tengo dos problemas de regresión, los dos ajustan iguales, pero uno tiene menos variables, entonces es conveniente dejar el que tiene menos variables por el principio de parsimonia.

Medidas de ajuste de un modelo de regresión lineal

Para dos variables podemos medir su correlación lineal con el coeficiente de correlación r (generalmente se usa el de Pearson,

https://es.wikipedia.org/wiki/Coeficiente_de_correlaci%C3%B3n_de_Pearson):

El valor del índice de correlación varía en el intervalo $[-1,1]$, indicando el signo el sentido de la relación:

Si $r = 1$, existe una correlación positiva perfecta. El índice indica una dependencia total entre las dos variables denominada relación directa: cuando una de ellas aumenta, la otra también lo hace en proporción constante.

Si $0 < r < 1$, existe una correlación positiva.

Si $r = 0$, no existe relación lineal. Pero esto no necesariamente implica que las variables son independientes: pueden existir todavía relaciones no lineales entre las dos variables.

Si $-1 < r < 0$, existe una correlación negativa.

Si $r = -1$, existe una correlación negativa perfecta. El índice indica una dependencia total entre las dos variables llamada relación inversa: cuando una de ellas aumenta, la otra disminuye en proporción constante.

Cuando tenemos dos o más variables, si hacemos la regresión usamos como ejemplo de la medida de ajuste usamos:

- R², coeficiente de determinación, esperamos que sea alto
- Significancia de las variables independientes o regresoras, buscamos que p<0.05
- Análisis de residuos queremos que
 - sean de distribución normal (gráfico qq plot o qq nom)
 - Tengan aproximadamente media 0 y no presenten estructura, es decir que sean independientes

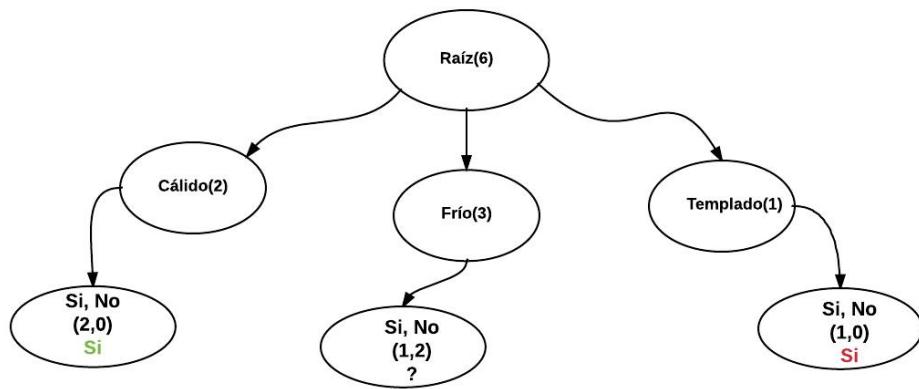
Debemos tener en cuenta que la regresión lineal se utiliza para casos de regresión y la regresión logística se utiliza para casos de clasificación.

Árboles de decisión

Los árboles de decisión se pueden utilizar tanto para casos de regresión como de clasificación. Por ejemplo, tenemos una base de compras ¿Compran o no el producto? Variables: clima, publicidad, precio, publicidad por redes sociales, compra?

Se saca, nunca se incluye en el análisis					Variable dependiente a predecir, como es cualitativa es de clasificación
individuo	Clima	publicidad	precio	publicidad por redes sociales	, compra?
1	cálido	fuerte	alto	poca	Si
2	cálido	fuerte	alto	muchas	si
3	frio	normal	bajo	poca	no
4	frio	normal	bajo	muchas	si
5	frio	baja	alto	poca	no
6	templa-doo	fuerte	alto	poca	si

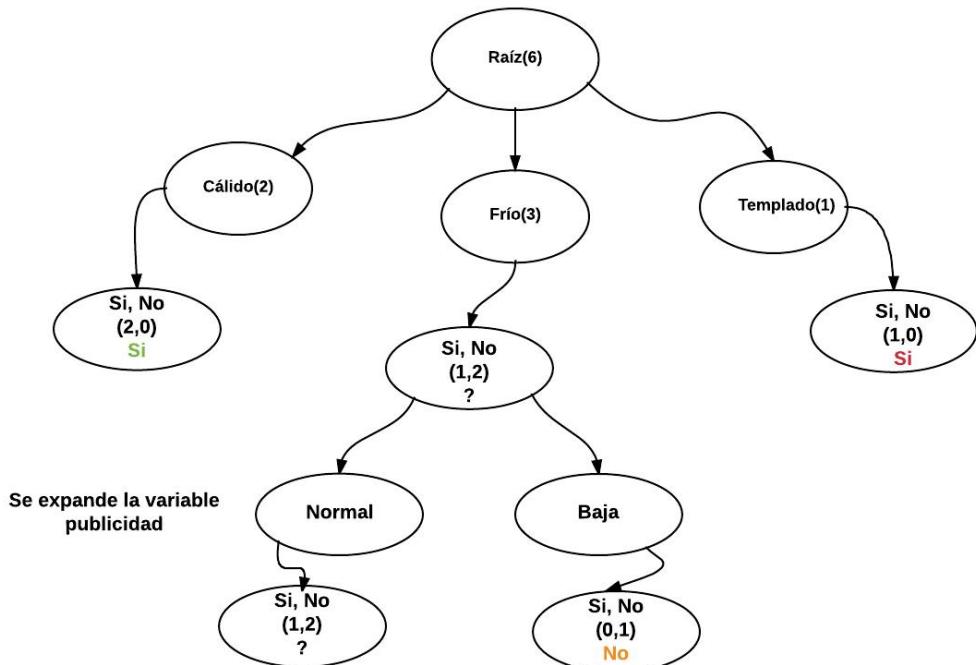
Lo primero que debemos hacer es elegir la raíz del árbol, para este caso, vamos a elegir el clima como raíz. Dentro del diagrama del árbol, los números entre () representan la cantidad de casos para esa categoría:



Primera regla: Si el clima es cálido entonces compra (Nodo terminal)

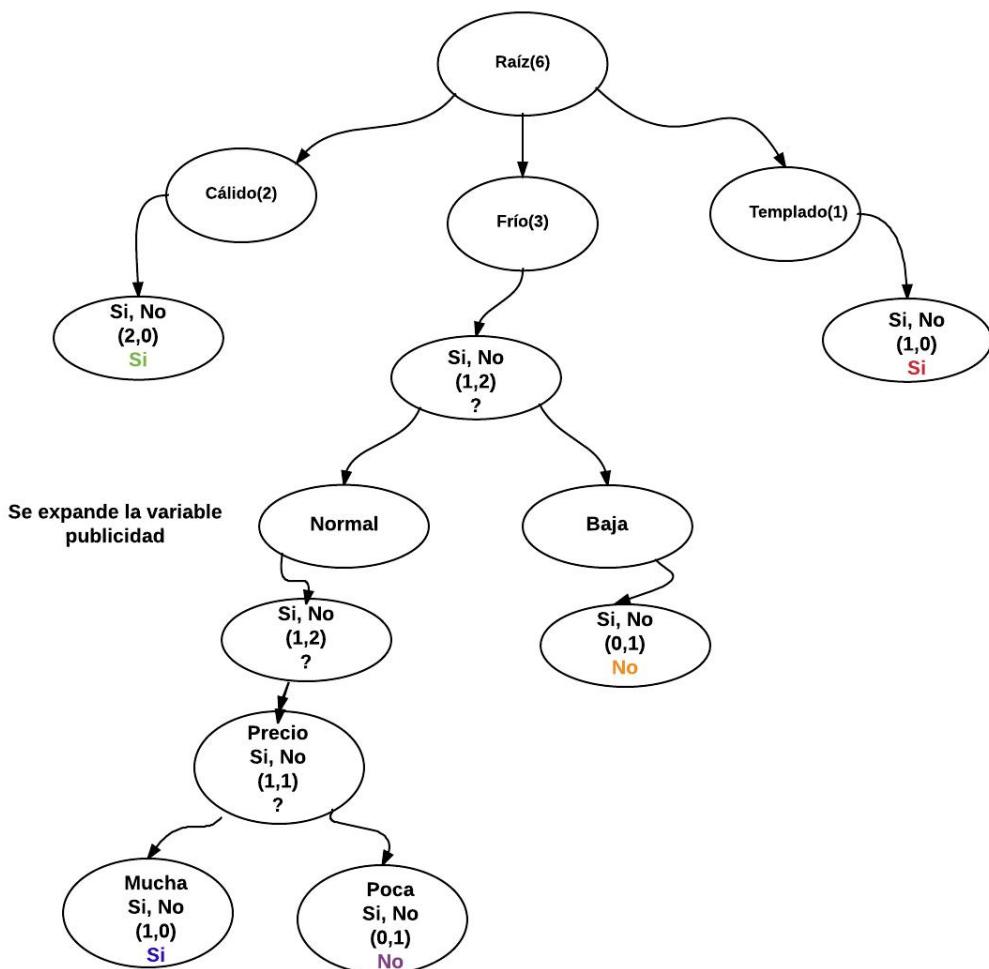
Segunda regla: Si el clima es templado entonces compra (Nodo terminal)

Para los casos de clima es Frío, no podemos decir si compran o no, entonces abrimos con la variable publicidad.



Tercer regla: Si clima frío y publicidad baja entonces NO compra

Como con frío y normal no podemos deducir nada, expandimos la variable precio y luego publicidad en redes



Cuarta regla, si el clima es frío, la publicidad es normal, el precio es bajo y publicidad en redes es mucha entonces SI compra.

Quinta regla, si el clima es frío, la publicidad es normal, el precio es bajo y publicidad en redes es poca entonces NO compra.

Para que el cliente venda mucho buscamos los SI, esto está representado por la regla uno, dos y cuatro

Cuando hay 0 posibilidades de un caso (encontramos una regla) entonces es un Nodo puro o homogéneo, es lo que buscamos, para la raíz tenemos que elegir aquella variable cuyo nodos sean la mayormente puros o homogéneos.

En general:

- Debemos buscar variables que aporten información
- Buscar nodos que sean homogéneos
- Buscar la menor cantidad de reglas

Las técnicas más utilizadas para realizar árboles de decisión son:

- CART: es el árbol de clasificación y regresión, es muy utilizado, e R la librería que tiene implementado a CART es rpart
- CHAID: es el usado por SPSS, trabaja con las distancias chi cuadrado
- ID3: utiliza el concepto de entropía, solamente permite problemas de clasificación

- C4.5: del mismo creados de ID3, es una mejora de este

Todos estos métodos tienen un índice que nos permiten obtener nodos homogéneos o puros. Para buscar nodos puros en ID3, considera que son los nodos con la menor entropía y más ganancia. Cuando va a abrir un nodo lo hace por aquella variable que logre menos entropía y mayor ganancia.

Para el ejemplo de si compra o no, vamos a calcular la entropía:

1- Calculamos la entropía del sistema, que es la de la variable a predecir:

$$\text{entropia} = -\text{prob(si)} * \log_2(\text{prob(si)}) - \text{prob(no)} * \log_2(\text{prob(no)})$$

Sistema: 6 casos, 4 si y 2 no

$$\begin{aligned}\text{Entropia(Sistema)} &= - (4/6) * \log_2(\text{prob}(4/6)) - \text{prob}(2/6) * \log_2(\text{prob}(2/6)) \\ &= - 0.67 * \log_2(0.67) - 0.33 * \log_2(0.33) \\ &= - 0.67 * -0.58 - 0.33 * -1.6 \\ &= 0.916\end{aligned}$$

2- Calcular la ganancia de cada una de las variables:

Tomanos variable clima y sacamos la probabilidad y entropía de las categorías de la misma.

$$\text{ganancia(clima)} = \text{entropia(sistema)} - \text{prob(calido)} * \text{entropia(calido)} - \text{prob(frio)} * \text{entropia(frio)} - \text{prob(templado)} * \text{entropia(templado)}$$

En R, no usa entropía usa el índice de GINI

```
> library(rpart)
```

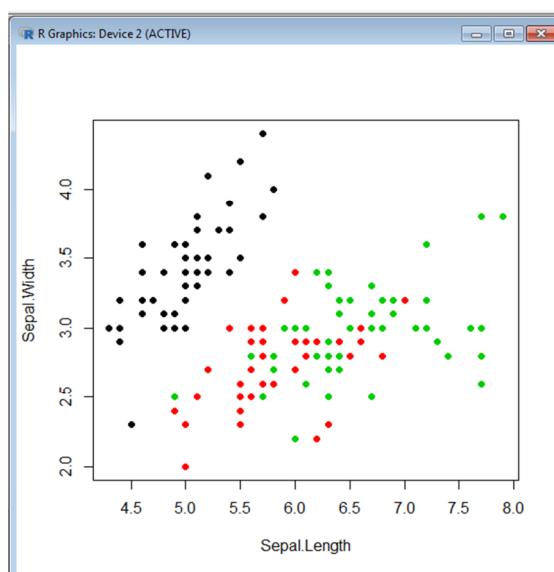
```
> m=iris
```

```
> fix(m)
```

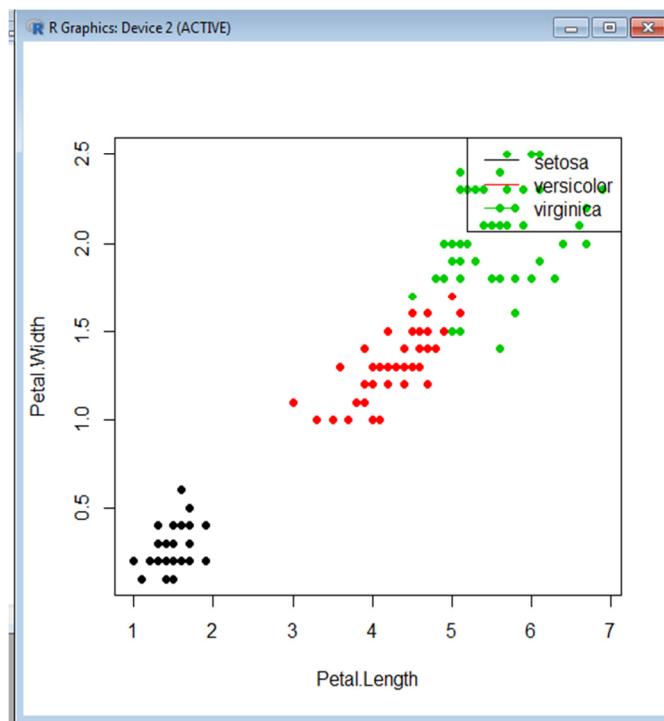
```
> plot(Sepal.Length, Sepal.Width, col=Species, pch=19, cex=1, lty="solid")
```

Warning messages:

```
> legend('topright', levels(Species), lty=1, col=1:3)
```

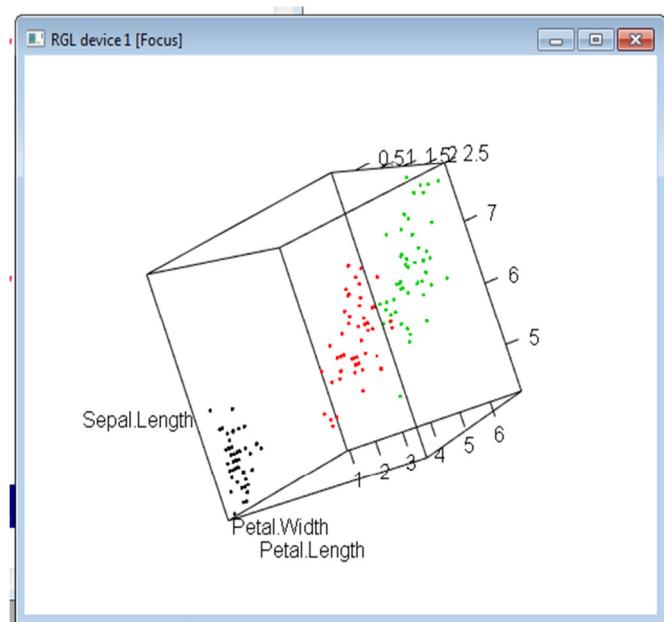


```
> plot(Petal.Length, Petal.Width, col=Species, pch=19, cex=1, lty="solid")
```



Los pétalos son mejores clasificando que los sépalos

```
> library(rgl)
> plot3d(Petal.Length, Petal.Width, Sepal.Length, col=as.integer(Species))
```



```
> arbol=rpart(Species~.,m,method="class")
```

```
> arbol
```

n= 150

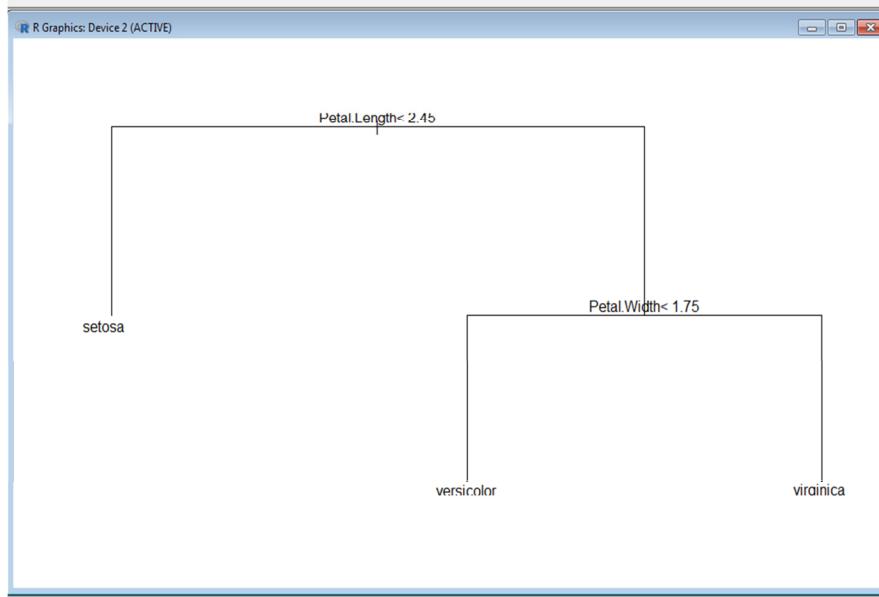
```
node), split, n, loss, yval, (yprob)
```

* denotes terminal node

```

1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
2) Petal.Length< 2.45 50  0 setosa (1.00000000 0.00000000 0.00000000) *
3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.50000000)
   6) Petal.Width< 1.75 54  5 versicolor (0.00000000 0.90740741 0.09259259) *
   7) Petal.Width>=1.75 46  1 virginica (0.00000000 0.02173913 0.97826087) *
> plot(arbol)
> text(arbol)

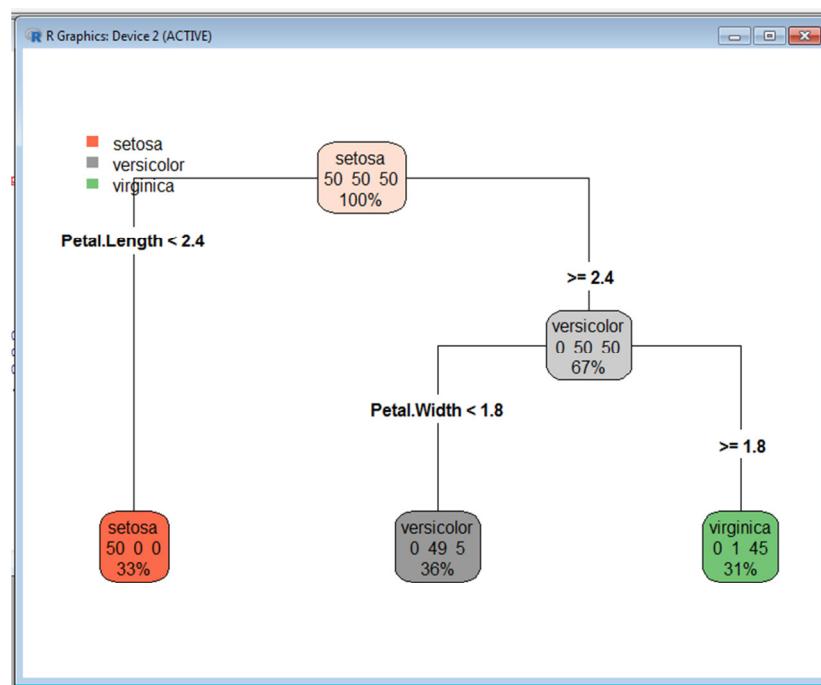
```



```

> library(rpart.plot)
> rpart.plot(arbol, type=4, extra=101)
> rpart.plot(arbol, type=4, extra=101, cex=2)

```



Reglas:

- 1-Si el largo del pétalo es menor a 2.4 entonces es setosa
- 2-Si el largo del pétalo es mayor o igual a 2.4 y el ancho del pétalo es menor a 1.8 entonces es versicolor (acá quedaron 5 virginicas mal clasificados), podría haber seguido dividiendo el árbol pero como calculo la ganancia de{j}cio que no valía la pena
- 3-Si el largo del pétalo es mayor o igual a 2.4 y el ancho del pétalo es mayor o igual a 1.8 entonces es virginica (acá quedó 1 versicolor mal clasificada)

Accuracy(exactitud), es el porcentaje de acierto, es el equivalente al r*2 en regression

En este caso es $144/150 = 0.96\%$ porque quedaron 6 flores mal clasificadas

summary(arbol)

Call:

rpart(formula = Species ~ ., data = m, method = "class")

n= 150

	CP	nsplit	rel error	xerror	xstd
1	0.50	0	1.00	1.19	0.04959167
2	0.44	1	0.50	0.75	0.06123724
3	0.01	2	0.06	0.07	0.02583280

Variable importance

Petal.Width Petal.Length Sepal.Length Sepal.Width

34 31 21 14

Node number 1: 150 observations, complexity param=0.5
predicted class=setosa expected loss=0.6666667 P(node) =1
class counts: 50 50 50
probabilities: 0.333 0.333 0.333
left son=2 (50 obs) right son=3 (100 obs)
Primary splits: //Como definiendo la bifurcacion de las ramas
Petal.Length < 2.45 to the left, improve=50.00000, (0 missing)
Petal.Width < 0.8 to the left, improve=50.00000, (0 missing)
Sepal.Length < 5.45 to the left, improve=34.16405, (0 missing)
Sepal.Width < 3.35 to the right, improve=19.03851, (0 missing)

Surrogate splits:

Petal.Width < 0.8 to the left, agree=1.000, adj=1.00, (0 split)
Sepal.Length < 5.45 to the left, agree=0.920, adj=0.76, (0 split)
Sepal.Width < 3.35 to the right, agree=0.833, adj=0.50, (0 split)

Node number 2: 50 observations

predicted class=setosa expected loss=0 P(node) =0.3333333

```
class counts: 50 0 0
probabilities: 1.000 0.000 0.000
```

```
Node number 3: 100 observations, complexity param=0.44
predicted class=versicolor expected loss=0.5 P(node) =0.6666667
class counts: 0 50 50
probabilities: 0.000 0.500 0.500
left son=6 (54 obs) right son=7 (46 obs)
```

Primary splits:

```
Petal.Width < 1.75 to the left, improve=38.969400, (0 missing)
Petal.Length < 4.75 to the left, improve=37.353540, (0 missing)
Sepal.Length < 6.15 to the left, improve=10.686870, (0 missing)
Sepal.Width < 2.45 to the left, improve= 3.555556, (0 missing)
```

Surrogate splits:

```
Petal.Length < 4.75 to the left, agree=0.91, adj=0.804, (0 split)
Sepal.Length < 6.15 to the left, agree=0.73, adj=0.413, (0 split)
Sepal.Width < 2.95 to the left, agree=0.67, adj=0.283, (0 split)
```

```
Node number 6: 54 observations
predicted class=versicolor expected loss=0.09259259 P(node) =0.36
class counts: 0 49 5
probabilities: 0.000 0.907 0.093
```

```
Node number 7: 46 observations
predicted class=virginica expected loss=0.02173913 P(node) =0.3066667
class counts: 0 1 45
probabilities: 0.000 0.022 0.978
```

```
> printcp(arbol) //indica cuales son las variables que uso del arbol
```

```
Classification tree:
rpart(formula = Species ~ ., data = m, method = "class")
```

```
Variables actually used in tree construction:
[1] Petal.Length Petal.Width
```

```
Root node error: 100/150 = 0.66667
n= 150
      CP nsplit rel error xerror   xstd
1 0.50     0    1.00  1.19 0.049592
```

```

2 0.44    1    0.50  0.75 0.061237
3 0.01    2    0.06  0.07 0.025833
> predicción=predict(arbol, m, type="class") //para predecir los datos, en este caso utilizamos el mismo conjunto de entrenamiento
> table(predicción, m$Species) //matriz de confusión, nos indican cuales quedaron mal clasificadas, nos permite observar cantidad de aciertos y cantidad de mal clasificados, en la diagonal encontramos los aciertos que son los verdaderos , los que están fuera de la diagonal están mal clasificados, son los falsos

```

Predicción setosa versicolor virginica //Es lo esperado, las filas son lo predicho

setosa	50	0	0
versicolor	0	49	5
virginica	0	1	45

> confusionMatrix(predicción, m\$Species)

Confusion Matrix and Statistics

Reference

Prediction setosa versicolor virginica

setosa	50	0	0
versicolor	0	49	5
virginica	0	1	45

Overall Statistics

Accuracy : 0.96

95% CI : (0.915, 0.9852)

No Information Rate : 0.3333

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.94

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: setosa Class: versicolor Class: virginica

Sensitivity	1.0000	0.9800	0.9000
Specificity	1.0000	0.9500	0.9900
Pos Pred Value	1.0000	0.9074	0.9783
Neg Pred Value	1.0000	0.9896	0.9519
Prevalence	0.3333	0.3333	0.3333
Detection Rate	0.3333	0.3267	0.3000

Detection Prevalence	0.3333	0.3600	0.3067
Balanced Accuracy	1.0000	0.9650	0.9450

Estos datos son importantes para verificar si el árbol es correcto.

Matriz de confusión

		Esperado	
Predicho	Si	No	
Si	VP	FP	
No	FN	VN	

Accuracy = Verdadero positivo + verdadero negativos / Verdadero positivo + verdadero negativos + falso positivos + falso negativos

Sensitivity = verdadero positivo / verdadero positivo + falso negativo

Specificity = verdadero negativo / verdadero negativo + falso positivo

Estos 3 valores son imprescindibles para evaluar el modelo. Existen más técnicas pero estas 3 son las mínimas utilizadas. Accuracy es exactitud y no precisión.

Cada modelo que hago lo debo testear, ¿cómo lo hago? Si cambias el conjunto de entrenamiento del modelo obtenemos otro modelo. Veamos para el caso de clasificación

1- Primero entreno el modelo (armarlo, modelo el problema), se limpia la base se sacan outlier y demás

2- Mido su performance con resultados del modelo y con un testeo, este testeo me va a determinar si el modelo ajusta bien o no

¿Cómo hago el testeo?

De toda la base, le doy ejemplos que no haya testeado en el entrenamiento y de la misma magnitud y que sepa la respuesta, se puede usar parte de lo que ya se tiene y no utilizar todo para el conjunto de entrenamiento, NO SE PUEDE USAR LO QUE USAMOS PARA ENTRENAR!!!

Conjunto de testeo:

- El conjunto de entrenamiento y de testeo DEBEN ser independientes
- Debe ser de formato similar o igual al de entrenamiento
- Debe ser aleatorio, se deben sacar datos de la muestra en forma aleatoria.
- Debe tener una cantidad suficiente de cada categoría, no necesariamente la misma cantidad de cada categoría
- De la muestra 80% de entrenamiento y 20% de testeo, en big data la proporción en testeo puede ser menor

Una vez que tenemos el conjunto de testeo armamos la matriz de confusión:

Para cada caso, le pregunto como da, si acierta es VP, y así vamos armando a través de lo esperado (es lo que sabemos) y el predicho (lo que indica el modelo):

		Esperado
Predicho	Si	No
Si	VP	FP
No	FN	VN

Accuracy (recall o exactitud) = $(VP + VN) / Total = \% = 0.6$ //Esto es cuanto le acierta

Sensibilidad= $VP / (VP + FN) = \% = 0.66$ //Mide el error con respecto a la variable Si, esto pesa más cuando no quiero muchos FN

Especificidad = $VN / (VN + FP) = \% = 0.5$ //Cuánto se equivoca en la parte de los negativos, esto pesa cuando no quiero muchos FP

En la [Teoría de detección de señales](#) una **curva ROC** (acrónimo de **Receiver Operating Characteristic**, o Característica Operativa del Receptor) es una representación gráfica de la sensibilidad frente a la especificidad para un sistema clasificador binario según se varía el umbral de discriminación.

En R

```
> library(caret)
> library(rgl)
> m=iris
set.seed(nro) //para que utilice la misma semilla para armar la partición y así siempre tener la misma
```

```
> particion=createDataPartition(y=m$Species, p=0.75, list=FALSE) //Armamos un conjunto de indices con los datos que queremos entrenar
```

```
> entrenamiento=m[particion,]
> test=m[-particion, ]
> summary(test)
> arbol=rpart(Species~., entrenamiento, method="class") //Armamos el árbol en base al conjunto de entrenamiento armado aleatoriamente
> rpart.plot(arbol, type=4, extra=101)
```

```
> pred=predict(arbol, test, type="class")
> table(pred, test$Species) //Comparamos los valores predichos y esperados con el conjunto de test
```

pred setosa versicolor virginica

```
setosa      12      0      0
versicolor   0     12      2
virginica    0      0     10
```

>

accuracy=34/36=0.94

sensibilidad=

```
> confusionMatrix(pred, test$Species)
```

Confusion Matrix and Statistics

Reference

Prediction setosa versicolor virginica

```
setosa      12      0      0
versicolor   0     12      2
virginica    0      0     10
```

Overall Statistics

Accuracy : 0.9444 //es alto en la realidad lo máximo es 70%

95% CI : (0.8134, 0.9932)

No Information Rate : 0.3333

P-Value [Acc > NIR] : 1.728e-14

Kappa : 0.9167

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: setosa Class: versicolor Class: virginica

Sensitivity	1.0000	1.0000	0.8333
Specificity	1.0000	0.9167	1.0000

**Esto indica que para la setosa lo va a acertar y con el resto puede haber confusión,
cuanto más cerca de 1 esta mejor es**

Pos Pred Value	1.0000	0.8571	1.0000
----------------	--------	--------	--------

Neg Pred Value	1.0000	1.0000	0.9231
----------------	--------	--------	--------

Prevalence	0.3333	0.3333	0.3333
Detection Rate	0.3333	0.3333	0.2778
Detection Prevalence	0.3333	0.3889	0.2778
Balanced Accuracy	1.0000	0.9583	0.9167

Ejemplo de los cangrejos

```
>library(crabs)

> fix(c)

> c$index=NULL

> names(c)

[1] "sp" "sex" "FL" "RW" "CL" "CW" "BD"

> set.seed(8)

> particion=createDataPartition(y=c$sp, p=0.75, list=FALSE)

> entrenamiento=m[particion,]

> test=m[-particion,]

> summary(test)

> arbol=rpart(sp~., entrenamiento, method="class")

> rpart.plot(arbol, type=4, extra=101)

> pred=predict(arbol, test, type="class")

> table(pred, test$sp)

> confusionMatrix(pred, test$sp)
```

Confusion Matrix and Statistics

Reference

Prediction B O

B 21 8

O 4 17

Accuracy : 0.76

95% CI : (0.6183, 0.8694)

No Information Rate : 0.5

P-Value [Acc > NIR] : 0.0001529

Kappa : 0.52

Mcnemar's Test P-Value : 0.3864762

Sensitivity : 0.8400 //reconoce mejor los azules que los naranjas

Specificity : 0.6800 //No reconoce muy bien los naranjas

Pos Pred Value : 0.7241

Neg Pred Value : 0.8095

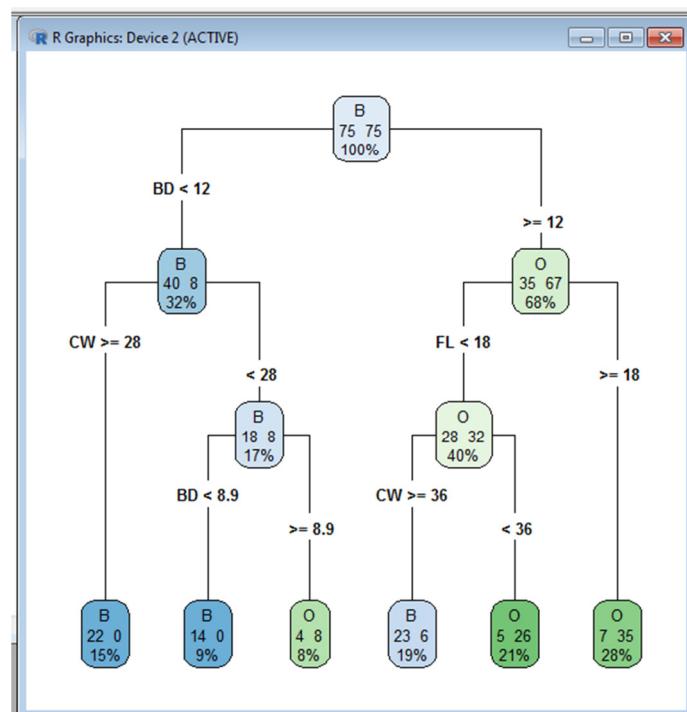
Prevalence : 0.5000

Detection Rate : 0.4200

Detection Prevalence : 0.5800

Balanced Accuracy : 0.7600

'Positive' Class : B //la sensibilidad está relacionada con la clase positiva



Los Autores



Javier Díaz

Licenciado en Matemáticas de la Universidad Nacional de La Plata actualmente es el Secretario de Relaciones Institucionales de la UNLP, Director Científico y Técnico del Centro Superior para el Procesamiento de la Información de la UNLP y Decano de la Facultad de Informática en distintos períodos desde su creación, es Profesor Titular Ordinario de más de 10 cátedras siendo una de ellas Tecnologías Aplicadas a Business Intelligence. Es también Director del Laboratorio de Investigación de Nuevas Tecnologías en Informática (LINTI) de la Facultad de Informática de la UNLP, en el que trabajan más de 50 personas dedicadas a la investigación y es la base de 10 cátedras pertenecientes a la carrera de Licenciatura en Informática y también es el sostén de estudios prácticos en el Master de Redes de Datos, del que el Lic. Díaz es también director.

<http://www.linti.unlp.edu.ar> jdiaz@unlp.edu.ar



Alejandra Osorio

Es Licenciada en Informática y Magister en Entornos Virtuales de Aprendizaje de la Universidad de Panamá actualmente es Directora de Desarrollo de Software en el CeSPI UNLP desde el año 2007, Directora de Sistemas Académicos desde el año 2007 a 2012 y Jefe de Proyectos desde el año 1990, cuenta con una amplia experiencia en el planeamiento estratégico, diseño, implementación y mantenimiento de tecnología, comunicaciones y sistemas para una amplia variedad de aplicaciones., especialista en Sistemas Académicos y gestión universitaria desde liquidación de sueldos a análisis de datos para la toma de decisiones, obteniendo una Diplomatura en Indicadores y Estadísticas Educativas, Universidad Nacional de Educación a Distancia de España en el año 2016.

aosorio@cespi.unlp.edu.ar



Ana Paola Amadeo

Es Licenciada en Informática y Magister en Entornos Virtuales de Aprendizaje de la Universidad de Panamá, es Profesora de la catedra Tecnologías Aplicadas a Business Intelligence desde el año 2011 y Jefe de Trabajos Prácticos de la materia Diseño de Software centrado en el Usuario desde el año 1999 al 2007. Docente investigadora desde el año 1999 en el Laboratorio de Investigación en Nuevas Tecnologías Informáticas. Coordinadora en la Dirección de Sistemas Académicos del CeSPI, participa en numerosos proyectos de análisis de datos sobre la realidad universitaria, informes periódicos a instituciones internas y externas, perspectivas de género, entre otros.

pamadeo@linti.unlp.edu.ar @pulamadeo



Diego Graselli

Es Licenciado en Informática en la Universidad Nacional de La Plata. Dedico mi tiempo al desarrollo de sistemas, investigación de nuevas tecnologías y docencia. Trabajo desde hace 15 años en el desarrollo de sistemas operacionales y analíticos. Colaboro en el portal de información del Centro Superior para el Procesamiento de la Información (CeSPI). Ayudante diplomado de Tecnologías Aplicadas a Business Intelligence desde hace 3 años.

diegograselli@gmail.com



Ariel Martínez

Es Licenciado en Informática de la UNLP, actualmente se desempeña como Líder de Proyecto, con sólidos conocimientos en Ingeniería de Software, procesos de negocios, manejo de herramientas de análisis y de desarrollo, pruebas de aplicaciones e implementación de software. Experiencia: análisis funcional, definición de proyectos de software, estimaciones de proyectos de software, definición y análisis de métricas, aseguramiento y control de la calidad de los productos, liderazgo y gestión, planificación y Organización. Especialidades: Gestión de equipos, Análisis Funcional y de dominio, definición de procesos aplicables a proyectos de software. Docente universitario con 15 años de experiencia, desempeñándose como colaborador, ayudante alumno, ayudante diplomado y profesor en CRESTA, actualmente Jefe de Trabajos Prácticos de Tecnologías Aplicadas a Business Intelligence

arielmarthinez@gmail.com



Ivana Harari

Es Licenciada en Informática de la Universidad Nacional de La Plata y Especialista en Docencia Universitaria (UNLP/2012). Se encuentra desarrollando la tesis final de la Maestría en Redes de Datos. Trabaja en el de tema de investigación denominado “Interfaces Hombre Computadora, Accesibilidad y Usabilidad”. Como integrante del Laboratorio de Investigación en Nuevas Tecnologías Informáticas de la UNLP, ha participado de numerosos proyectos de evaluación de sistemas desde el punto de vista de comunicación hombre–máquina, para distintas organizaciones públicas y privadas. Ha dirigido tesinas de Licenciatura en Informática y de Maestría. Ha dictado cursos sobre la temática en distintas universidades nacionales. Ha publicado artículos en prestigiosos eventos científicos nacionales e internacionales. Actualmente se desempeña como Profesor Adjunto de la cátedra Diseño Centrado en el Usuario y de Interfaces adaptadas para dispositivos móviles de la Facultad de Informática de la Universidad Nacional de La Plata.



Dalila Romero

Licenciada en Informática de la UNLP, actualmente trabajo en el Centro Superior para el Procesamiento de la Información (CeSPI) de la Universidad Nacional de La Plata y soy docente de la Facultad de Informática de la misma Universidad. Entre las tareas que realizo habitualmente se encuentra el análisis de datos, gestión de estadísticas e indicadores, resolución de requerimiento de análisis de información, administración y gestión de portal de estadísticas. Con respecto a la docencia soy profesor adjunto en dos cátedras en las cuales continuamente estamos actualizando los contenidos teniendo en cuenta las tendencias actuales.

dromero@info.unlp.edu.ar



Néstor Armando López

Egresado de la Facultad de Ciencias Exactas de la UNLP como Calculista Científico, ejerció su profesión de informático en el ámbito público de la Provincia de Buenos Aires (Ministerio de Economía, Dirección General de Cultura y Educación, Instituto Provincial de Lotería y Casinos, Arba) desde el año 1980. En el ámbito privado, brindó soporte y capacitación de software IBM a través de la empresa MEGA Computer S.A. a diversos clientes entre los años 1992 y 1999. Participó en distintos proyectos del Centro Superior para el Procesamiento de la Información (CeSPI) en la UNLP entre 2002 y la ac-

tualidad. En el ámbito docente, fue profesor titular de una materia de la carrera de Técnico Superior en Análisis de Sistemas en el Instituto Superior de Formación Docente y Técnica Nro. 12 de la Provincia de Buenos Aires entre 1983 y 1987; instructor certificado de DB2 y Visual Age Generator en IBM entre 1997 y 2001; instructor certificado por Proydesa y Oracle Argentina para la carrera de Administrador de Bases de Datos Oracle entre 2002 y 2006. Actualmente colabora en la cátedra de Tecnologías Aplicadas a Business Intelligence aportando toda su experiencia.

nlopez@cespi.unlp.edu.ar



Emanuel Borda

Egresado de la Facultad de Informática de la UNLP de Analista Programador Universitario. Creador del sistema operativo Fuentux.

Amante de la tecnología y la innovación. Docente en la Facultad de Informática de la UNLP. Ayudante diplomado de Tecnologías Aplicadas a Business Intelligence.

Director en la Dirección de Servicios Digitales al ciudadano de la Dirección Provincial de Gobierno Digital perteneciente a la Subsecretaría de Innovación y Experiencia Ciudadana del Gobierno de la Pcia. de Bs As.

Tecnologías para el análisis de datos basadas en software libre : guía de desarrollo y aplicación /

Francisco Javier Díaz ... [et al.] ; coordinación general de Javier Díaz ; María Alejandra Osorio ;

Ana Paola Amadeo. - 1a ed . - La Plata : Universidad Nacional de La Plata ; La Plata : EDULP, 2019.

Libro digital, PDF - (Libros de cátedra)

Archivo Digital: descarga
ISBN 978-950-34-1774-4

1. Software Libre. 2. Análisis de Datos. I. Díaz, Francisco Javier II. Díaz, Javier, coord. III. Osorio, María Alejandra, coord. IV. Amadeo, Ana Paola, coord.

CDD 005.4

Diseño de tapa: Dirección de Comunicación Visual de la UNLP

Universidad Nacional de La Plata – Editorial de la Universidad de La Plata
47 N.º 380 / La Plata B1900AJP / Buenos Aires, Argentina
+54 221 427 3992 / 427 4898
edulp.editorial@gmail.com
www.editorial.unlp.edu.ar

Edulp integra la Red de Editoriales Universitarias Nacionales (REUN)

Primera edición, 2019
ISBN 978-950-34-1774-4
© 2019 - Edulp

e
exactas


Editorial
de la Universidad
de La Plata



UNIVERSIDAD
NACIONAL
DE LA PLATA