



Data warehousing

Del dato al
conocimiento
organizacional

Julian Grijalba F
2020-06-01

Data warehousing

Del dato al conocimiento
organizacional

01 Introducción

02 ¿Cómo desarrollar modelos de información en un Data warehouse?

03 Procesos de datos (ETL) - Extracción, Transformación y Carga de datos.

04 Del dato al Conocimiento

El contenido de este ebook está orientado a dar los primeros pasos para que una organización sin importar su tamaño o enfoque de negocio pueda convertir a los datos en su activo más valioso de la organización. Esto desde un enfoque de crear un único punto de verdad para la toma de decisiones, definido como un Data warehouse o Bodega de datos.

El ebook inicia con un recorrido de la importancia de una buena toma de decisiones, como se puede modelar nuestra información para poderla centralizar y así llegar al objetivo común, convertir los datos en conocimiento para la organización.

Su contenido se estructura así:

1. Introducción
 - Conceptos
2. ¿Cómo desarrollar modelos de información en un Data warehouse?
 - Tipos de Modelos de información
 - Metodologías de modelado de información
 - Creación de un data warehouse en la nube
 - Buenas practicas en la generación de modelos de data warehouse
3. Procesos de datos (ETL) - Extracción , Transformación y Carga
 - Introducción
 - Conceptos de ETL en Azure Data Factory
 - Estrategias de cargas de datos
 - Buenas practicas en el diseño de procesos de datos
4. Del dato al Conocimiento
 - Reportes y Dashboard
 - Definición de indicadores
 - Diseño modelo de autoservicio

1.Introducción

¿No escuchas a los datos de tu organización?, en esta guía vamos a ser que hablen.

Con este fin vamos a introducirlos en un mundo de datos, donde nuestro principal objetivo deber ser la confianza de la organización en la información que generamos.

Comenzaremos con un concepto clave: Inteligencia Empresarial (Business Intelligence / BI), es un conjunto de herramientas y procesos que le ayudan a tomar decisiones con base en datos precisos, ahorrando tiempo y esfuerzo. La función principal de una herramienta de BI, es brindar la posibilidad de analizar fácilmente los datos basados en conceptos de negocio, sin tener conocimientos técnicos sobre herramientas de bases de datos u otras fuentes que contengan los datos.

Las herramientas de BI pretenden extraer conocimiento de nuestros datos almacenados usualmente en una bodega de datos (data warehouse/DWH) con un enfoque preciso en tres principales pilares: fiabilidad, disponibilidad y experiencia de usuario.

Ahora, describamos un caso que nos permita contextualizar la importancia de los datos:

Imagina que eres el director general de una pequeña empresa dedicada a la fabricación de zapatos y, con base en los informes de ventas por producto que estás analizando, detectas que, han ido disminuyendo las ventas mensuales de los zapatos deportivos azules . Las cifras que ves ahora corresponden a, aproximadamente, la mitad de lo que se vendía a principios de año. Como director general tienes diferentes posibilidades:

- Eliminar los zapatos deportivos color azul del catálogo.
- Cambiar el diseño de los zapatos deportivos color azul del catálogo.
- Crear una bonificación para el departamento comercial, para promover la venta de este producto.
- Despedir al responsable de la marca de zapatos porque su departamento está causando pérdidas, como se refleja en el análisis de pérdidas y ganancias.

Pero ¿qué ocurre si el verdadero problema no está allí?, este se genera debido a que se le ha incluido un nuevo atributo a los zapatos deportivos azules, lo cual generó un cambio en su código interno y, el catálogo de productos no está correctamente actualizado en los sistemas transaccionales, por ello no se están reflejando correctamente las ventas con el nuevo código.

Desde esta perspectiva, sus decisiones anteriores serían erróneas porque se están basando en un análisis incorrecto de los datos. Este es un ejemplo del por qué, la fiabilidad, es un requisito básico en todos los proyectos relacionados con la información, puesto que, esta se utiliza en la toma de las principales decisiones, desde la gestión estratégica de la empresa, hasta las actividades operativas básicas. Por consiguiente, es obligatorio para nosotros como gestores de datos, garantizar la consistencia de estos, para que cada dimensión de análisis asegure resultados correctos basados en la calidad de la información utilizada.

Ahora analicemos otro escenario, estás trabajando en una cadena de comidas rápidas; es viernes a las 6 de la tarde y, necesitas hacer el pedido de los diferentes insumos alimenticios antes de irte a casa. Estás ejecutando el reporte de tu almacén que sugiere la cantidad de insumos que necesitas pedir, pero, identificas que la fuente de datos de este reporte tiene información del miércoles por la mañana porque el proceso encargado de actualizarlo

diariamente aún no ha terminado. ¿Qué sucede?, la siguiente semana posiblemente sufrirás falta de insumos para la preparación de los productos en la cadena de comidas rápidas o, por otro lado, podrás tener un almacén completamente saturado porque pediste 1.000 hamburguesas, que no habrías pedido si el sistema te hubiera mostrado que éstas, llegaron el jueves pasado.

Podrías afrontar un problema similar, si no puedes acceder al sistema en el momento requerido, debido a alguna actividad de mantenimiento en la plataforma y, debes, con base en tu intuición y no en los datos, realizar el estimativo de lo que hace falta, para realizar el pedido. Desde la perspectiva de la información, todos los productos de datos deben estar disponible para los usuarios cuando ellos necesiten usarlos. Esto parece una condición obvia, pero es importante enfatizar en dos factores principales que nos permiten lograr este objetivo. Primero, el sistema debe ser estable, funcionar correctamente durante el horario comercial, y segundo, los datos deben estar actualizados en función de los usuarios objetivo y sus necesidades.

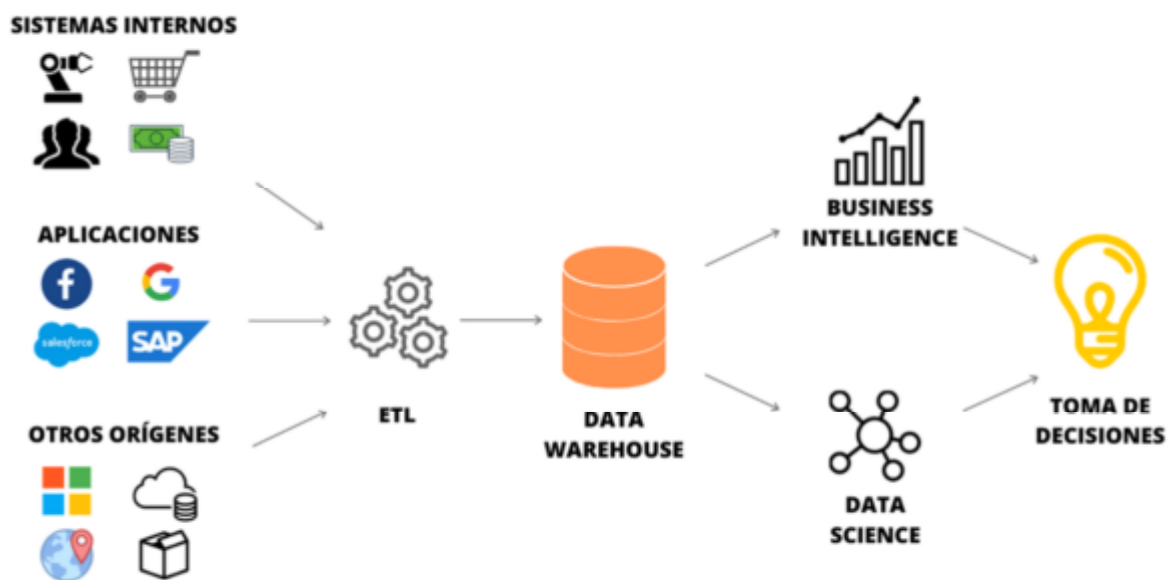


Ilustración 1-Flujo de datos a conocimiento¹

En los siguientes capítulos estaremos abordando conceptos, herramientas y, arquitecturas de información, que nos permitirán tener un soporte para la toma de decisiones con base en datos en cualquier tipo de organización.

Conceptos

a) ERP

La fuente de información más común en una solución de Data Warehouse y normalmente la primera a la que se acude es, la Enterprise Resource Planning (ERP). Este tipo de herramienta transaccional se utiliza para controlar y guardar las principales operaciones que se producen dentro de la empresa.

¹ Data Warehouse. (s. f.). [Grafico]. mistralbs. <https://www.mistralbs.com/wp-content/uploads/2020/01/dwh-2.png>

Dependiendo de las características del ERP habilitado en su empresa, podremos extraer la información principal sobre ventas, finanzas, operaciones, recursos humanos o stock.

b) ETL

El componente ETL, acrónimo de Extracción, Transformación y Carga, tiene un nombre bastante descriptivo. Como puede imaginar, sus funciones principales son Extraer, Transformar y Cargar. ¿Qué significa? Usted no utilizará sus tablas transaccionales para desarrollar el análisis del modelo con su herramienta de Inteligencia de negocios; utilizará su datawarehouse para ese propósito. Así que tendrá que extraer la información de su origen (p.e. ERP) y cargarla en su data warehouse. En medio de este proceso tendrá que adaptar la información a la estructura requerida por el data warehouse, creando las diferentes tipologías de objetos de información, calendarizarlos y darle seguimiento a estos.

c) Data Warehouse (DWH)

Es la base de cualquier sistema de información en una corporación, porque es el lugar donde residen los datos que se quieren analizar, normalmente apoyados en un sistema de base de datos. La idea debajo de un DWH es que se pueden recoger datos de múltiples fuentes, limpiarlos, asegurar la consistencia y su integridad, para tener una implementación de BI fiable.

El suministro de información de su organización puede tener múltiples fuentes, por ejemplo, datos procedentes de su sistema transaccional, datos de pedidos/entregas/facturas, datos procedentes de la herramienta de almacenamiento que controla el stock que tiene dentro de su almacén, datos procedentes de la empresa logística que está entregando productos, la agrupación manual de clientes o productos realizada por sus analistas, los datos procedentes de sus clientes minoristas que muestran cómo están vendiendo sus productos, datos de la organización mundial de la salud referente a estadísticas de Covid-19, datos de movilidad de Google, etc. Puede tener datos heterogéneos en estas fuentes, pero la base de las técnicas de DWH es conseguir que todos estos datos estén vinculados de alguna manera y se hablen entre ellos.

d) Tablas

Son objetos de la base de datos, diseñados para almacenar y manipular registros de información.

e) Hechos

Suelen ser las tablas más grandes del modelo (Fact tables), ya que contienen la historia de los datos de negocio que deben ser analizados, resumidos y agregados según los campos requeridos. Contienen el detalle de las ventas, los costes, los pasos operativos, los movimientos contables, datos de recursos humanos, visitas de la fuerza de ventas, o cualquier dato que se quiera analizar. No es necesario tener el máximo nivel de detalle en todas las tablas de hechos, es más, se recomienda tener algunas tablas agregadas precalculadas para tener un mejor tiempo de respuesta de la interfaz de informes. Los hechos, al ser solos números e identificadores, no pueden ser interpretados directamente, es por ello que, se tienen las tablas de dimensiones que se describen a continuación.

f) Dimensiones

Es la interpretación de los números que nos llegan de las tablas de Hechos. Estas también denominadas como, Dimension tables, contienen principalmente el identificador del concepto y su descripción. Estos campos también se denominan, Datos descriptivos, porque detallan cómo se distribuyen los datos. En este caso podemos pensar en la fecha, el cliente, el producto, la planta, la región, el país, la oficina de ventas, o cualquier otro campo que permita dividir la información.

g) Reportes

Al final del camino, los resultados principales de una herramienta de inteligencia de negocios serán los informes, a un nivel que permita a los usuarios finales, analizar la información de forma detallada. Dependiendo de la herramienta de BI, pero también de las características y complejidad de los informes, podemos encontrar diferentes conceptos como Informe, Documento, Cuadro de Mando (dashboards), Vista de Análisis, Auto-servicios, etc. Todos ellos, se refieren a la interfaz final que se utilizará para analizar la información. Pueden ir desde una simple tabla de ventas por categoría, a un conjunto de visualizaciones interactivas entre sí, que pueden paginar, desglosar (drill up/down), pivotar, imprimir o enviar por correo electrónico.

h) Inteligencia de negocios

Inteligencia de negocio (Busines Intelligence / BI) tiene múltiples definiciones en diferentes publicaciones. La definición de Wikipedia para BI es la siguiente "La inteligencia de negocios es un conjunto de teorías, metodologías, arquitecturas y tecnologías que transforman los datos brutos en información significativa y útil para los propósitos del negocio". Esta es una definición interesante debido a que muestra una imagen completa de una solución de BI, y no sólo el enfoque habitual en las herramientas de visualización de datos que algunas definiciones destacan.

Avanzar con una solución de BI implica seguir unas teorías en la definición del proceso tales como, el modelo de datos que cubre las necesidades del negocio, aplicar metodologías ágiles que le ayuden a construir de forma incremental la solución y su mantenimiento una vez la solución esté al alcance de los usuarios finales. Todo lo anterior, cubierto por una arquitectura que proporcione un adecuado Retorno de la Inversión (ROI) en función del beneficio que genera el proyecto en la organización.

2. ¿Cómo desarrollar modelos de información en un Data warehouse?

Tipos de Modelos de información

Los modelos de datos se basan en un modelo relacional en la medida en que suelen ubicarse en una base de datos relacional. Se llama relacional porque la base del modelo son las relaciones entre los datos, normalmente guardados en tablas. Un ejemplo sencillo de esta clase de modelo es una tabla de ventas en la que tienes un código de producto, un código de cliente y el importe de las ventas realizadas durante este mes. De esta tabla se pueden obtener todas las relaciones entre producto y cliente que tengan alguna venta relacionada. También puedes unir tablas usando consultas en lenguaje SQL (Structured Query Language) para unir información adicional y obtener nueva información. Existen múltiples clasificaciones de modelos de datos, pero en este momento vamos a hablar de los principales tipos de modelos utilizados en el data warehousing, modelos normalizados, desnormalizados, modelos en estrella y en copo de nieve.

a. Modelo normalizado

Un modelo normalizado pretende reducir al mínimo la redundancia de datos, optimizando los costes de almacenamiento al evitar repetir los mismos datos varias veces. Este tipo de modelo es muy recomendable para las soluciones transaccionales y también se puede utilizar en los modelos DWH, pero hay que tener en cuenta sus restricciones. Estas se dan porque, debido a su concepto, la forma en que se accede normalmente a la información no es eficiente, pues, requiere que se unan muchas tablas en el proceso, generándose un alto impacto en el costo de procesamiento de la información, por consiguiente, demoras en el tiempo de respuesta de las consultas que se realicen. Su lado positivo, es que es muy eficiente a la hora de guardar información y tiene unos altos niveles de integridad en sus datos.

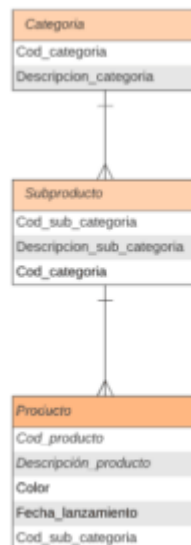


Figura 1- Modelo normalizado

b. Modelo desnormalizado

El modelo desnormalizado quiere mejorar el rendimiento de la consulta evitando las uniones en tiempo de ejecución. Para ello, repite los datos a lo largo de las tablas para minimizar el número de uniones necesarias para tener toda la información junta.

Producto
Cod_producto
Descripción_producto
Color
Fecha_lanzamiento
Cod_sub_categoria
Descripcion_sub_categoria
Cod_categoria
Descripcion_categoria

Figura 2- Modelo desnormalizado

La situación más habitual en un DWH es que se necesite un enfoque intermedio entre una alta normalización y desnormalización. Dependiendo de la naturaleza de la jerarquía identificadas en los datos (p.ej. categoría -> sub-categoría->Producto) y sus atributos se podrá definir este. En el ejemplo anterior (Figura 2), podría tener el Cod_Categoría y el Cod_subcategoría en una nueva tabla con sus descripciones, pero entonces tendría que unirse con las tablas Producto para obtener la descripción.

c. Modelo en estrella

El modelo en estrella es un tipo de modelo relacional muy utilizado en el data warehousing, especialmente para los DataMart pequeños, en el que se tienen tablas de hechos (de las que hablamos en los conceptos) que contienen la información, por ejemplo, de sus datos de ventas, financieros, etc., los cuales son analizados desde las diferentes perspectivas de información (dimensiones) que dispone nuestro modelo.

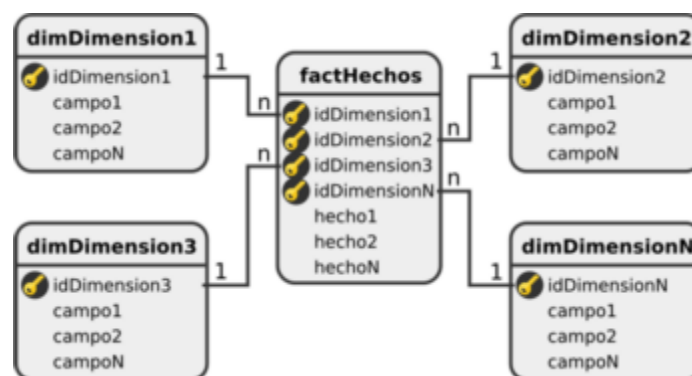


Figura 3- Modelo en Estrella ²

d. Modelo de copo de nieve

Piense ahora en un copo de nieve. Visualizará un copo con un gran núcleo y luego ramas que se dividen en otras más pequeñas. De manera similar, puede pensar en un modelo de datos de copo de nieve en el que, tiene grandes tablas de hechos en el centro, vinculadas por algunas tablas clave que contienen la base para extraer jerarquías de atributos

² Bernabeu, D. (s. f.). Estrella. <https://troyanx.com/Hefesto/estrella.html>.

principales, luego, puede encontrar directamente alguna tabla de dimensiones o tablas clave más pequeñas que relacionan los datos con otras tablas de dimensiones. En esencia, en el copo de nieve, las dimensiones tienen sus jerarquías en diferentes tablas, como se puede ver en la Figura4, dimDimension2 y dimDimension3.

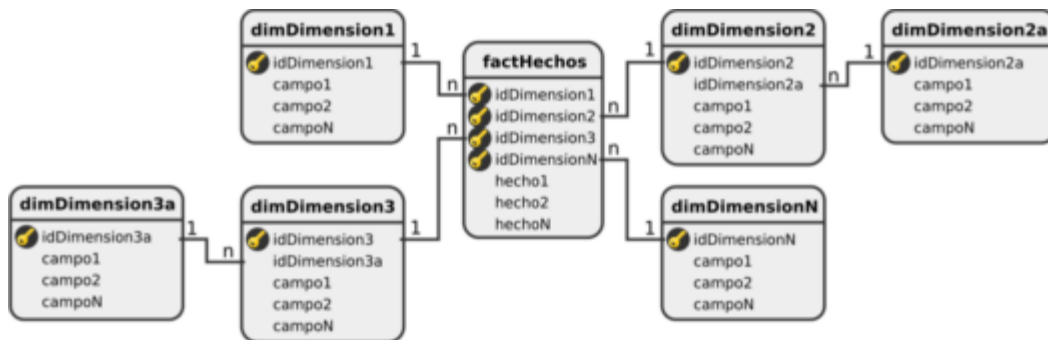


Figura 4- Modelo Copo de Nieve³

e. Modelo físico de datos

Una vez que tengas el modelo lógico definido con los anteriores modelos expuestos pasarás a la parte técnica y, tendrás que definir qué campos contienen tus tablas; cuáles estarán relacionados directamente con las entidades del modelo lógico; cuál es el tipo de datos para todos estos campos; cuáles son los campos que se utilizarán para unir tablas; cuáles serán las claves únicas de las tablas; si tu modelo reforzará la integridad de los datos con el uso de claves foráneas o si prefieres mantener la restricción física fuera de tu modelo y desarrollar controles lógicos en los procesos de datos (ETL).

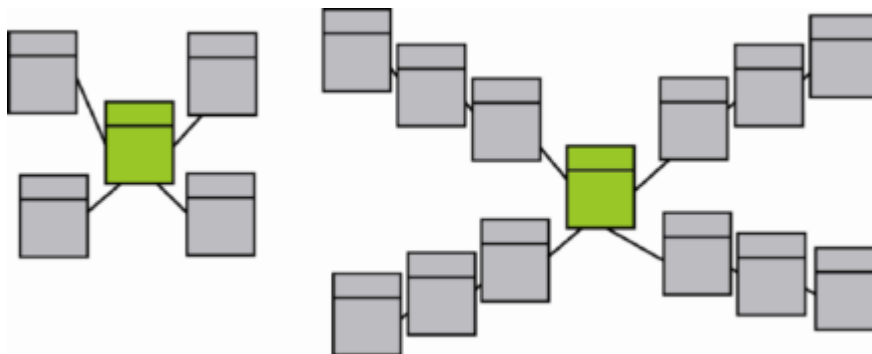


Ilustración 2 - Modelos Copo de Nieve - Estrella⁴

Tenga en cuenta que es importante definir una nomenclatura para las tablas y los campos. Esta nomenclatura puede ser muy estricta, pero recomendamos tener un nivel de nomenclatura intermedio entre los nombres muy descriptivos y los simplemente técnicos. Algo que permite fácilmente distinguir a las tablas es el origen y su objetivo, para ello se pueden usar la siguiente nomenclatura:

- fac_nombre_tabla: información relacionada a un hecho de datos
- dim_nombre_tabla: información relacionada a una dimensión
- stg_tipotabla_nombre: Tabla de trabajo

³ Bernabeu, D. (s. f.). Copo de Nieve. <https://troyanx.com/Hefesto/estrella.html>. <https://troyanx.com/Hefesto/copo-de-nieve.html>

⁴ Star schema (left) vs. Snowflake schema (right). (s. f.). [Ilustración]. Star schema (left) vs. Snowflake schema (right). https://www.researchgate.net/figure/Star-schema-left-vs-Snowflake-schema-right_fig4_227246694

En cuanto a la nomenclatura de los campos, se recomienda seguir estas dos sencillas reglas:

1. No poner el mismo nombre de campo a diferentes conceptos: normalmente las herramientas de BI reconocen cuando se tiene el mismo nombre de campo e intentan unir la información a través de este campo, provocando alguna pérdida de datos. Hay que tener en cuenta que algo que parece ser similar, como Región, puede provocar desajustes en los datos si se trata de región de cliente o región de empleado, ya que pueden ser diferentes en algunos casos.

2. Definir el mismo nombre para el mismo concepto de información: Normalmente las herramientas de BI reconocerán de forma automática un campo que se llama igual en diferentes tablas como si este fuese el mismo, esto reducirá complejidad del modelo pues facilitará la búsqueda de información en la bodega de datos.

Resaltaremos la relevancia de esto con el siguiente ejemplo: la entidad cliente tiene una característica de ubicación geográfica y a este campo lo definimos como ubicación, otra entidad como sucursal tiene esa misma característica y la definimos también como ubicación, ¿Qué ocurrirá cuando intentábamos hacer un análisis de las ventas por ubicación de la sucursal? La herramienta detectaba que ubicación estaba disponible en ambas tablas e interpreta que son la misma, por lo que unirá ambas tablas a través de este campo generando una confusión de datos entre el cliente y la sucursal que claramente es un error, es por ello que siempre se deben distinguir los campos claramente, para este ejemplo deberán ser `ubicacion_cliente` y `ubicacion_sucursal`.

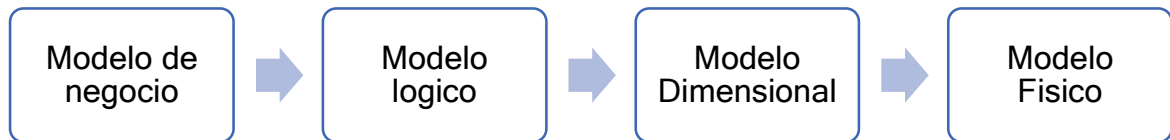
Para los campos de datos también se sugieren unos estándares básicos:

- `id_nombre_campo`: Campo identificador de la tabla
- `desc_nombre_campo`: Campo descriptivo de la tabla
- `fec_nombre_campo`: Campo fecha en la tabla
- `num_nombre_campo`: Campo numérico en la tabla
- `imp_nombre_campo`: Campo de valores relacionados con dinero (importes)

Finalmente a la hora de construir su modelo físico evite mientras pueda las relaciones entre las tablas que tengan características de muchos a muchos, estas confunden a las herramientas de BI y en ocasiones a usted mismo. Para evitar esto intégreles por medio de ETL o utilice unas tablas de relación.

Metodología de Modelado de información

Con los tipos de Modelo de Información, se dio una visión general del modelado de datos considerando principalmente dos pasos en el proceso, definir el modelo lógico estableciendo un conjunto de entidades, tipos de relación entre ellas y atributos que existen en cualquier tabla, luego, seguir con el modelo físico cuando ya definimos todos los nombres de los campos, tipos, precisión y otras consideraciones físicas para la creación y ubicación de las tablas. Ahora vamos a profundizar en las diferentes etapas que tenemos que superar desde nuestros requerimientos del negocio hasta la creación del modelo físico del data warehouse.



a. Modelo de negocio

El modelado de negocio está directamente relacionado con el equipo de negocios, los usuarios clave, los propietarios de productos, o el interlocutor del lado del negocio. Una herramienta útil para realizar este análisis es, una matriz de granularidad, en la que se puede cruzar qué dimensiones se utilizan en cada análisis y a qué nivel; como se visualiza a continuación:


	Date	Product	Store	Employee	...
Sales	X	X	X	X	
Discounts	X	X	X		
Deliveries	X		X	X	
Purchases	X	X		X	

Figura 4-Matriz de granularidad⁵

b. Modelo lógico

Una vez que hayamos verificado que todos los requisitos estén en el modelo de nuestro análisis de negocio, podremos definir qué entidades deben ser tenidas en cuenta en nuestro sistema. Para ello tendremos que, examinar en detalle cuales son los conceptos que nuestros usuarios clave esperan obtener para ser analizados, validando la posibilidad de acceder a ellos en nuestro sistema fuente. Parece bastante lógico que si tienen algún requerimiento de análisis es porque lo están utilizando en los datos fuente (por ejemplo, un

⁵ Data Warehousing Guide - Core Concepts - 1/4. (2020, 21 agosto). <https://qimia.io/>. <https://qimia.io/en/blog/Data-Warehousing-Guide-Core%20Concept>

ERP, archivos planos, sistemas transaccionales, etc.), no obstante, no hay que aceptar esto como una premisa hasta que no se valide.

En este modelo lógico definiremos cómo se relacionan las entidades, si es una relación uno a uno, uno a muchos, o muchos a muchos.

Finalmente, en un modelo lógico necesitamos definir qué atributos se colocan en cada entidad, en otras palabras, qué características de los diferentes conceptos de negocio estarán disponibles para cada una de estas. Hablando de productos podemos pensar en el color, el tamaño, la categoría, la sub-categoría, la fecha de lanzamiento o cualquier otro concepto relacionado con el producto. Hablando de clientes podemos pensar algo como país, nombre, segmento de negocio, etc. Para la dimensión tiempo podemos tener día, mes, año, día de la semana, mes del año, u otros atributos relacionados con el tiempo.

c. Modelo dimensional

Durante el proceso de modelado lógico será necesario identificar qué hechos deberemos tener en cuenta y cuáles son nuestras dimensiones que pueden considerarse como grupos de atributos relacionados con una determinada relación entre ellos. A veces esto se considera como parte del modelo lógico, pero a veces, especialmente en casos de gran complejidad, puede hacerlo en un análisis separado obteniendo como resultado el modelo dimensional para los datos. Al definir los tipos de relación entre atributos verá que puede tener los mismos tipos de relación que para las entidades, uno a uno, uno a muchos, y muchos a muchos, pero en este caso estamos teniendo en cuenta el nivel de detalle (granularidad) esperada para los campos dentro de las tablas en lugar de la granularidad de las tablas.

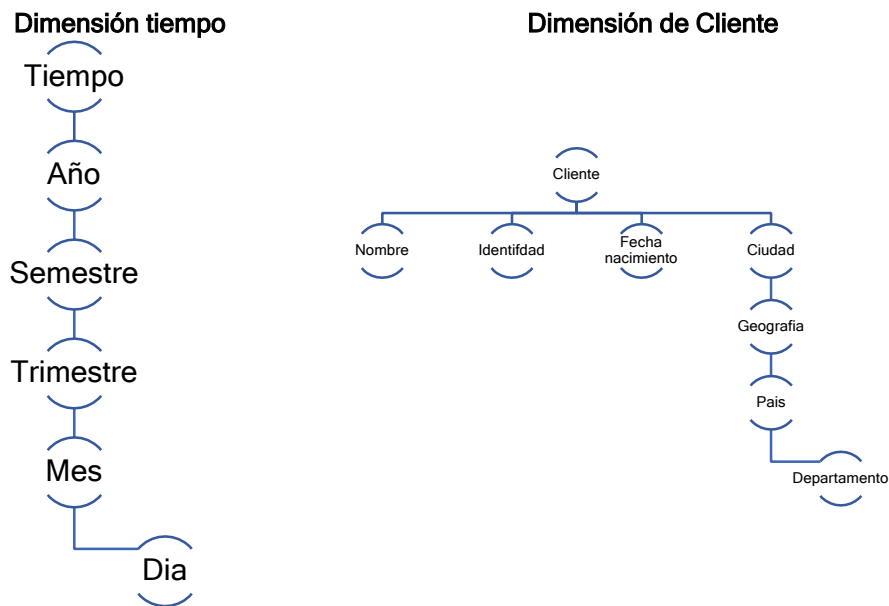


Figura 5 – Modelo Multidimensional

El resultado más claro del análisis dimensional es un esquema jerárquico, como se ve en la figura 5. Cuando se tiene una relación de uno a muchos entre dos atributos, significa que, por ejemplo, en el caso de Geografía vs. Cliente, se pueden tener múltiples clientes que se encuentran en una Geografía, pero sólo una ciudad asignada a cada cliente. En el caso de la dimensión del tiempo es totalmente jerárquica, un mes solo pertenece a un semestre y un semestre puede tener varios meses asociados. Además, añadido a este análisis gráfico deberíamos obtener la lista de hechos que queremos analizar, con una descripción sobre la fuente y las condiciones.

Entidad	Hechos	Descripción
Ventas	Cantidad Ordenada	Cantidad ordenada en un pedido por un cliente
	Cantidad Enviada	Cantidad enviada en un pedido a un cliente
	Valor de impuestos	Valor impuestos aplicados a la factura
	Valor total de factura	Valor total de la factura con impuestos
Producto	Valor unitario producto	Valor unitario del producto

Figura 6 - Listas de hechos de una venta

Hay unas características específicas para las dimensiones que tenemos que considerar a la hora de hacer nuestros diseños:

- **Dimensiones lentamente cambiantes (SCD)**

Es muy importante garantizar que en el data warehouse se refleje el historial de los cambios que han sucedido durante un lapso de tiempo, es allí donde aparece el término de Dimensiones lentamente cambiantes (Slowly Changing Dimensions). El enfoque de estas se centra en los cambios históricos en los datos (por ejemplo, vendedores) y, que la información se refleje de la manera correcta. Un ejemplo de esto puede ser, el vendedor Julian de la sucursal de chapinero, está asociado a una jefe de zona de ventas en Bogota.

Julian a trabajado durante los primeros 6 meses del 2021 en la zona de Bogota y en el mes 7 a él lo mueven a la zona de ventas de Cali, ¿Qué ocurre con sus ventas?, ¿se deberían ir con él a Cali, aunque no haya vendido nada aun?, ¿Bogota perdería los primeros seis meses de las ventas de Julian? Esto no es real, e imaginamos que su antiguo jefe de zona no estará nada feliz con perder todas esas ventas, hecho que no es verdadero. Para estudiar con más detalle las técnicas asociadas, pueden acceder a este enlace: What are Slowly Changing Dimensions? (s. f.). Xplenty. Recuperado 7 de mayo de 2021, de <https://www.datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html>

- **Huérfanos**

Los huérfanos son unos de los errores más comunes en las dimensiones cuando son analizadas por herramienta de BI y, corresponden a valores que aparecen en la tabla de hechos (como un hijo), pero que no tienen una referencia en la dimensión (sin papá), y por ello se llaman huérfanos. Para gestionarlos se pueden aplicar algunas de las siguientes estrategias:

- Crear un padre ficticio para los huérfanos:** en los procesos de carga se buscarán los valores que se encuentran en los hechos y no tienen referencias en las dimensiones, como resultado los valores encontrados serán asignados a un padre “otros” o “huérfanos” para que sean gestionados
- Eliminar huérfanos:** No es una práctica adecuada debido a que la eliminación de valores constituye un cambio en la verdad de los datos entregados. Esto solo podrá ser permitido si es aprobado por el negocio, sin embargo, no se recomienda por el sesgo que aplica.

d. Modelo físico

En este momento ya se tienen claras las necesidades del usuario, se ha definido qué entidades lógicas estarán involucradas en el modelo y cuál es la relación entre ellas, se ha analizado qué dimensiones y hechos son requeridos, como también sus atributos.

El último paso es la creación de objetos en la base de datos. Una herramienta de modelado de datos es especialmente útil en este paso, ya que después de definir la estructura de las tablas, columnas, tipos de columnas, claves primarias, claves foráneas, etc. se tendrá en la mayoría de ellas la posibilidad de generar scripts de creación de todas las entidades que hacen parte del modelo de la data warehouse. Se podrán definir puntos adicionales que permitan una mejor gestión y optimización del data warehouse como son al menos el particionamiento, el esquema o el propietario, la compresión y otras características que variarán dependiendo del motor de base de datos que se utilice.

El siguiente paso es crear las tablas y definir sus campos en el data warehouse. Para las columnas de las tablas se establece el nombre siguiendo las convenciones de nomenclatura definidas; el tipo de campo normalmente será numérico; de caracteres o de fecha, se define también el tamaño y la precisión del campo y otros parámetros para determinar si son nulos o no. No podemos dejar a un lado, temas importantes como el mantenimiento y el buen performance(rendimiento) del data warehouse tales como las claves primarias, los índices y las claves foráneas debido a que:

- Un índice mejora el tiempo de respuesta del sistema cuando buscamos un valor determinado, tal y como lo hacemos cuando queremos leer un capítulo específico de un libro, en lugar de buscar en todas sus páginas, por medio del índice, llegamos al que buscamos leer.
- Una clave primaria es una restricción que permite que un valor (o combinación de valores) sea único en la tabla y no sea vacío. En una tabla de dimensiones como la de producto, por ejemplo, la llave primaria será el código de producto, ya que es único para cada uno de estos. Por otro lado, en una tabla de hechos es posible que no podamos definir una clave primaria, dependerá de nuestro diseño, pero normalmente en entornos de Data warehouse tendrás información agregada al nivel deseado, lo que provocará implícitamente que tu clave sea el conjunto de columnas que hayas incluido en la cláusula group by.
- Una llave foránea es un valor que reside en otra tabla y que asegura que cada vez que ingresemos un dato, ese dato exista en la tabla que hacemos referencia. El uso de claves foráneas tiene sus ventajas e inconvenientes, por lo que, si quieres implementarlas debes saber de antemano qué inconvenientes se te pueden presentar. La principal ventaja es que garantizan la integridad de los datos entre las tablas. Si en la dimensión de Producto tienes una clave foránea a subcategoría, se garantiza que, si llega un nuevo producto la referencia a subcategoría exista en la tabla de subcategoría, si no es así, el registro será rechazado. Una de las principales desventajas es que pueden complicar el proceso ETL y también causar cierta lentitud durante la carga de datos. Cuando se tiene una clave foránea habilitada no se pueden eliminar los datos de la tabla referenciada, ni borrar registros que tengan información relacionada en las tablas dependientes, algo que parece lógico, pero

como veremos, a veces en los procesos de datos (ETL) es más fácil de gestionar si se elimina y recarga completamente algunas tablas.

Creación de un Data warehouse on Cloud

Introducción

Antes de empezar a trabajar con bases de datos y, probablemente, con cosas aún más complicadas, necesitas estar familiarizado con el lenguaje estándar que se utiliza para interactuar con ellas. En esta actividad veremos una introducción a SQL. Nos concentraremos sólo en los temas básicos, para que no te pierdas en detalles que no son necesarios para esta actividad. Para los que ya poseen conocimientos sobre el funcionamiento de SQL, pueden pasar directamente a construir los entregables con base en lo diseñado en la actividad 1.

SQL son las siglas de Structured Query Language (lenguaje de consulta estructurado). Como su nombre indica, es un lenguaje utilizado para consultar una base de datos, pero tiene muchas otras funciones aparte de la consulta. Una cosa que debemos saber sobre SQL es que no es similar a otros lenguajes de programación como Java, C# y Python. Esto se debe a que es un lenguaje declarativo, en lugar de uno procedimental. Esto significa que en lugar de describir los cálculos que hay que hacer, especifica lo que se pretende obtener.

SQL se creó originalmente teniendo en cuenta el modelo relacional. Sin embargo, las modificaciones posteriores añadieron características que no forman parte del modelo relacional, por lo que puede hacer las cosas de una manera diferente si lo necesita. Muchas de las versiones actuales de las bases de datos admiten tanto el modo relacional como el relacional de objetos, que incluyen soporte para la programación orientada a objetos y tipos de datos más complejos que los habituales primarios.

El SQL, apareció por primera vez en 1970 y su última versión es del 2011. No todos los motores de bases de datos implementan la misma versión; incluso no todas implementan una versión completa. Esto genera en ocasiones inconvenientes al querer migrar la información de un motor de base de datos a otro.

Esta guía referencia en su material complementario, información a detalle para la manipulación de base de datos a profundidad con el fin de generar los entregables requeridos en esta actividad.

Creación y gestión de infraestructura en la nube

Una de las mejores opciones que se tienen en la actualidad, es la posibilidad de crear infraestructura en la nube sin requerir de ningún centro de datos, servidores u otros dispositivos, simplemente se gestionan y crean a través de un portal web. Este tipo de soluciones nos ofrecen muchas herramientas, no solamente la infraestructura, las cuales detallaremos a continuación:

- **IaaS (Infraestructura as a Service):** Este servicio nos permite tener disponibilidad de equipos de hardware de redes, almacenamiento y procesamiento, como podría ser desde un disco duro virtual de 10 TB y unos servidores muy potentes, hasta tener la disponibilidad de crear grandes granjas de servidores a unos pocos clics.

- **PaaS (Platform as a Service):** Este servicio proporciona una plataforma completa, con una infraestructura y software que está listo para ser usado. Este servicio facilita el desarrollo y ejecución de una solución, sin tener que planear una infraestructura.
- **SaaS (Software as a Service):** Este permite suministrar aplicaciones a través de Internet, como un servicio. En lugar de instalar y mantener el software, simplemente se accede a él a través de Internet. El proveedor gestiona el acceso a la aplicación, incluida la seguridad, la disponibilidad y el rendimiento.

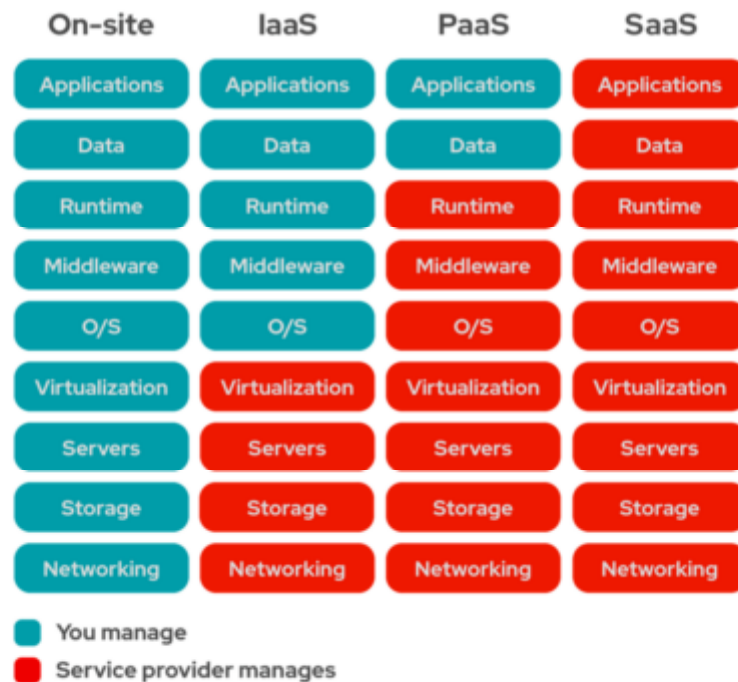


Figura 7- Servicios en la Nube⁶

Buenas prácticas en la generación de modelos de data warehouse

El data warehouse es el corazón para la toma de decisiones en la mayoría de las organizaciones, es por ello que tenemos que aplicar una serie de buenas prácticas en el diseño de nuestro modelo, previo a la creación del mismo. A continuación, se describen las principales:

1. Utilice los tipos de datos más pequeños posibles

Utilice siempre el tipo de datos más pequeño posible. Lo que esto significa es que nunca deberíamos usar un tipo de datos de cadena cuando se podría usar un número entero. El uso del tipo de datos más pequeño posible optimizará el almacenamiento de datos, ETL (extraer - transformar - cargar), generar informes y procesar el modelo semántico.

⁶ redhat. (s. f.). Diferencias entre IaaS, PaaS y SaaS. Diferencias entre IaaS, PaaS y SaaS. Recuperado 15 de mayo de 2021, de <https://www.redhat.com/cms/managed-files/iaas-paas-saas-diagram5.1-1638x1046.png>

2. Defina los atributos de las dimensiones de una forma descriptiva

Los mejores atributos de dimensión son aquellos que son de naturaleza descriptiva. Los atributos descriptivos son más fáciles de entender desde la perspectiva del usuario porque los atributos de dimensión se utilizan para describir, filtrar, controlar, ordenar y proporcionar contexto para las medidas cuantitativas. Los atributos descriptivos permiten a los usuarios aprovechar el valor de las métricas de manera más completa.

3. Utilice claves subrogadas

Las claves subrogadas, se utilizan para relacionar las tablas de dimensiones con las tablas de hechos. Las claves sustitutas (SK) no tienen ningún significado para el negocio ni un significado intrínseco. Los SK generalmente se asignan en el momento en que se carga un registro en la tabla de dimensiones y generalmente se mantienen a través del proceso ETL. Los SK se utilizan normalmente como clave principal en una tabla de dimensiones determinada y son diferentes a la clave empresarial. El uso de claves sustitutas tiene varias ventajas, puesto que, nos permiten rastrear el historial de registros de dimensiones. Los campos de clave sustituta también pueden proporcionar un rendimiento superior en comparación con el uso de una clave comercial, que podría usar un tipo de datos de cadena de caracteres.

4. Utilice esquemas en estrella, mientras sea posible

Los modelos de datos normalizados suelen ser muy confusos, pero un diseño desnormalizado, como un esquema en estrella, es muy simple y requiere muy pocas combinaciones para producir una consulta significativa. El objetivo de un almacén de datos es proporcionar grandes volúmenes de datos a un usuario para informes analíticos y un esquema en estrella simple y optimizado nos ayuda a lograr este objetivo.

5. Permita que en los hechos se tengan diferentes tipos de métricas que permitan agregar la información

El mejor tipo de medida para guardar información en el almacén de datos es aquella que se puede agregar por completo. Es decir, una que se puede resumir en cualquier dimensión o en todas las dimensiones y seguir siendo significativa. Por ejemplo, una medida de Monto de ventas se puede resumir por Producto, Fecha, Geografía, etc. y aún proporcionar información valiosa para el cliente. Las medidas que no se pueden agregar por completo, como las proporciones u otros cálculos porcentuales, deben manejarse en el modelo semántico o en la herramienta de generación de informes.

6. Evite cometer los siguientes errores:

- a. Diseñar las tablas a partir del informe final esperado: El fundamento de un modelo multidimensional es poder soportar muchas preguntas de negocio, si esto no se cumple y lo ajustamos a un reporte, el modelo no funcionará

- b. Número excesivo de Joins para acceder al resultado: El objetivo de un modelo multidimensional es procesar la mayor cantidad de información en el menor tiempo posible, y esta última variable tiene relación directa con la cantidad de saltos (joins) que hacemos entre las tablas. Si se identifica que una consulta requiere muchos joins, es necesario revisar seriamente el modelo.

3. Procesos de datos (ETL) - Extracción , Transformación y Carga de datos.

Introducción

Antes de diseñar los procesos de datos, es fundamental entender cuáles son las opciones que tendremos de procesamiento en la herramienta seleccionada en la nube. En este orden de ideas primero abordaremos los conceptos y funcionalidades comunes, luego crearemos un ambiente de Azure Data Factory, que nos permita materializarlos. Una vez, esto quede claro, se describirán las buenas prácticas en el diseño de los procesos de datos, esto con el fin de guiar al estudiantes para generar los procesos de datos adecuados según los requerimientos de negocio y la realidad de los datos.

Conceptos de ETL en Azure Data Factory

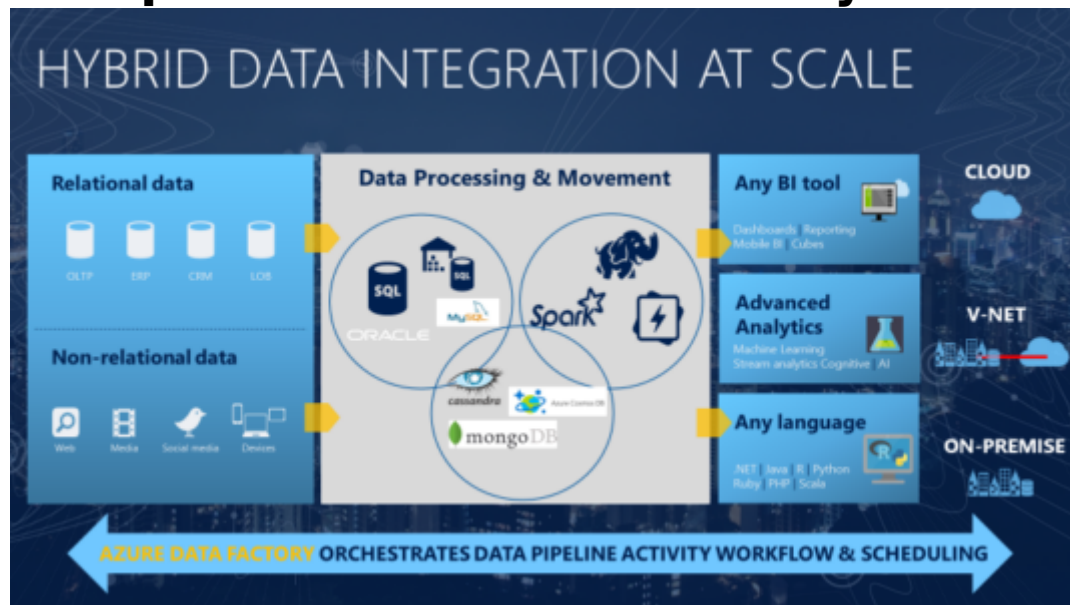


Ilustración 3 - Arquitectura ADF⁷

Azure Data Factory (ADF) es la herramienta de facto para construir soluciones de datos en Azure, debido a que puede manejar procesos ETL complejos y se integra de forma nativa con todos los servicios de Azure, y nos quita un dolor de cabeza en nuestro plan de trabajo.

Su interfaz de usuario permite arrastrar y soltar objetos preconstruidos para agilizar los desarrollos y proporciona una solución de completa, garantizando de esta forma la trazabilidad de los procesos y sus ejecuciones.

⁷ Wickaramarathna, R. S. (2019, 30 enero). A Quick Guide to Azure Data Factory - Ruwan Sri Wickaramarathna. Medium. <https://medium.com/@ruwansriw/about-azure-data-factory-604eb1028702>

1. Objetos de ADF

Azure Data Factory cuenta con cuatro componentes esenciales, con los cuales se proporcionan los cimientos para construir los flujos de datos para extraer, transformar, cargar datos o ejecutar tareas personalizadas. A continuación, se describen estos componentes claves:

- **Activity:** Representa una acción o un paso de procesamiento. Por ejemplo, se genera una actividad para copiar datos entre una fuente y su destino.
- **Pipeline:** Es una agrupación lógica de actividades como si tuviéramos un Workflow. Las actividades de un pipeline pueden encadenarse para operar secuencialmente, también de forma independientemente en paralelo. Por ejemplo, se podría implementar que día a día copie los datos de ordenes desde la fuente al área de trabajo (staging), se ejecuten transformaciones en el área staging y finalmente se copie el resultado en el data warehouse.
- **Dataset:** Es la materia prima y el destino que esta tendrá después del procesamiento. Los conjuntos de datos representan estructuras de datos que simplemente apuntan o hacen referencia a las fuentes de información que desea utilizar en sus actividades como entradas o salidas. Por ejemplo, apuntar a una tabla de dimensión que se creó en la anterior guía.
- **Linked Service:** Este componente gestiona los detalles de conexión tanto a las fuentes origen, de trabajo y destino. Por ejemplo, la conexión a la base de datos que se desplego en la primera guía.
- **Integration runtime:** Se refiere a la infraestructura informática subyacente utilizada por ADF. Es la computación donde se produce el movimiento de datos, el envío de actividades o la ejecución de paquetes realizados en SQL Server Integration Services.

2. Creación de procesos ETL

Para la creación de procesos de datos se sigue un estándar de proceso que se describirá a continuación:

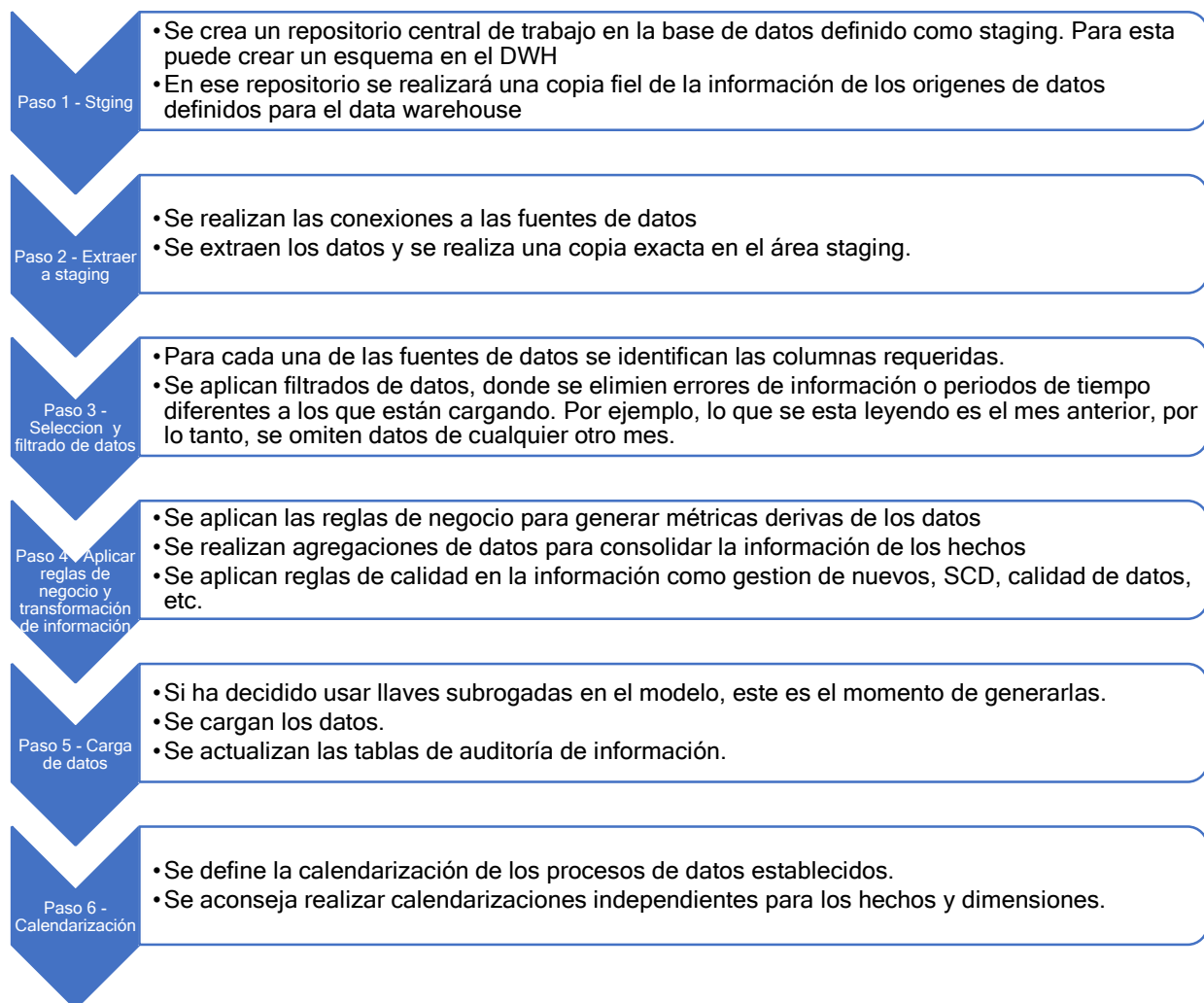


Ilustración 4 - Flujo estándar procesos de datos estructurados

Se tienen los siguientes videos de apoyo:

- **Video Creación ambiente Data Factory:** En solo 2 minutos pueden ver como se generó el ambiente de ADF <https://youtu.be/ZZ81n1w5bjE>
- **Video ADF Crea automáticamente tablas:** En solo 5 minutos, se puede ver como se carga una fuente de datos del proyecto y genera automáticamente una tabla en el área staging del DWH <https://youtu.be/HTXBHrfs-78>
- **Documentación creación ambiente Data factory:** Microsoft. (2020, 14 diciembre).

Create an Azure data factory using the Azure Data Factory UI - Azure Data

Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/quickstart-create-data-factory-portal>"

- **Documentación copia de archivos planos desde un blob a una base de datos:**
Microsoft. (2021a, febrero 18). Copy data from Azure Blob storage to SQL using Copy Data tool - Azure Data Factory. Microsoft Docs.
<https://docs.microsoft.com/en-us/azure/data-factory/tutorial-copy-data-tool>

Estrategias de carga de datos

Es algo fundamental para tener en cuenta, aún más si decidimos usar llaves foráneas. Se tendrán principalmente, tres tipos de tablas en función de cómo se cargan:

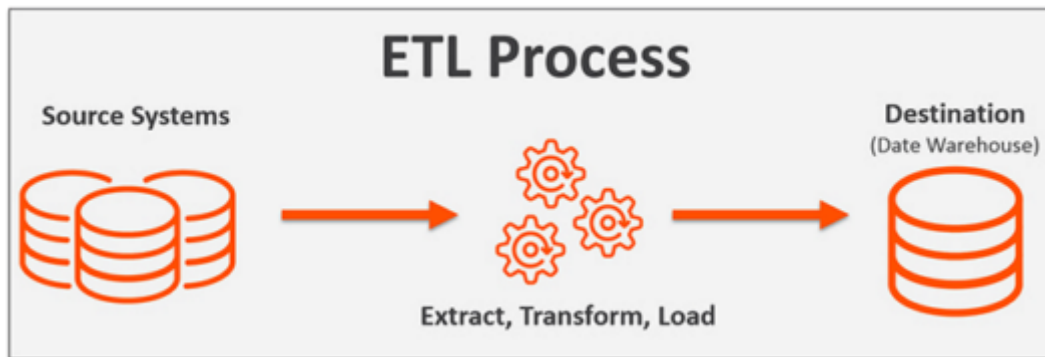


Ilustración 5 - Procesos de datos - ETL⁸

- **Cargas manuales/fijas:** No se incluirán en ningún proceso de ETL, tendrán información fija hasta que la actualicemos manualmente. Son útiles para atributos estáticos o muy poco cambiantes, como la empresa, el país o, en nuestro modelo, las tablas de regiones, que veremos en la Guía 1.
- **Cargas incrementales:** En este caso subiremos nuevos registros que, aparecen en cada carga. Puede que sólo tengamos nueva información disponible o puede que tengamos todo el histórico, pero preferiríamos sólo cargar las nuevas filas en lugar de toda la tabla desde cero cada vez. Para poder tener información fiable en la tabla, necesitaremos tener una clave primaria de la tabla o fechas de auditoria (creación, modificación, eliminación) para poder detectar qué información es nueva y cuál ya ha sido insertada en nuestra tabla. Para los registros existentes tendremos dos opciones: actualizar los antiguos con información nueva para los campos no clave o mantenerlos como estaban. La primera es la opción más habitual en este caso.
- **Cargar completamente las tablas:** Esta suele ser la forma más rápida y sencilla de desarrollar un ETL y, es especialmente útil para recargar tablas de dimensiones.

⁸ Watts, S. (2017, 14 diciembre). ETL Basics [Ilustración]. ETL Basics: Extract, Transform & Load. <https://s7280.pcdn.co/wp-content/uploads/2017/11/ETL-Process.jpg>

También puede ser necesario utilizar un proceso de recarga completa para algunas tablas de hechos agregados si podemos recibir información pasada o si la agregación se basa en relaciones de atributos cambiantes. Su enfoque es sencillo eliminar todo y recargar todo al mismo tiempo.

Buenas prácticas en el diseño de procesos de datos

1. **Reglas de nomenclatura:** La convención de nomenclatura ayudará en gran medida a estandarizar los procesos de datos de la misma manera como se hizo con las estructuras del Data warehouse. Estas reglas se deben definir para los siguientes objetos:
 - a. Servicios vinculados
 - b. Conjuntos de datos
 - c. Pipelines
 - d. Actividades dentro de los pipelines

Debe dejar claro en este estándar, cual es la periodicidad del pipeline, cual es la tecnología de los Linked Service usada y, aún más, en los Datasets con el objetivo de reusarlos en otros procesos, describiendo al menos si estos son fuentes de datos, de donde vienen, y donde estaban estos datos, por ejemplo una fuente de datos de actividades que está en Sql server, se podría describir así, DS_SQL_actividades o, el destino en el área staging de la bodega de datos así, TG_SQL_staging_actividades. Microsoft también comparte unos estándares descritos en el siguiente enlace: Microsoft. (2020a, octubre 15). Rules for naming Azure Data Factory entities - Azure Data Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/naming-rules>

2. **Trabajar con carpetas:** Se pueden utilizar carpetas y subcarpetas para organizar los componentes de Data Factory, ya que nos facilitan la navegación.
3. **Servicios vinculados dinámicos:** Reutilizar el código siempre es un gran ahorro de tiempo y significa que a menudo tiene una huella más pequeña para actualizar cuando se necesitan cambios, como puede parametrizar el servicio vinculado en su Azure Data Factory y hacer que el servicio vinculado sea reutilizable. Por ejemplo, puede parametrizar el nombre de la base de datos en el linked service en lugar de crear 5 separados correspondientes a las 5 bases de datos que tenga.
4. **Uso de plantillas:** Las plantillas le permiten crear e implementar una infraestructura de Azure completa, segura y rápida. Se puede acceder a este tutorial: Microsoft. (2020c, diciembre 17). Tutorial - Create & deploy template - Azure Resource Manager. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/template-tutorial-create-first-template?tabs=azure-powershell>.

5. **Orden lógico de creación de los procesos de datos:** Como se describió en el numeral a) en el paso a paso para creación de procesos de datos, igualmente, se refuerza que la forma lógica de construir estos para poblar un DWH son:
- Extraer datos de la fuente y copiarlos en área staging
 - Transforma los datos
 - Carga de Dimensiones
 - Cargar de hechos
6. **Creación jerárquica de Pipelines:** Paul Andres, en su blog, recomienda crear Pipelines de forma jerárquica con el fin de facilitar la asignación de alcance y responsabilidades, para estos, se puede acceder a este enlace: Mrpaulandrew, V. A. P. B. (2019, 25 septiembre). Azure Data Factory - Pipeline Hierarchies (Generation Control). Welcome to the Community Blog of Paul Andrew. <https://mrpaulandrew.com/2019/09/25/azure-data-factory-pipeline-hierarchies-generation-control/>
7. **Documentación de Pipelines:** Es muy importante documentar los objetos que hemos creado siempre, esto orienta y agiliza actividades de mantenimiento o revisión de incidencias pues ayuda a entender cómo se creó, la idea aplicada o las razones detrás de alguna actividad.
8. **Automatización del proceso:** El uso de desencadenadores (Triggers) es una forma de administrar y programar la ejecución de los procesos de datos. Hay tres tipos diferentes de disparadores para elegir: evento, programado y ventana giratoria.
- **Eventos:** Este inicia cuando se crea, elimina o modifica algún objeto en el entorno. No se aconseja usarlo para las necesidades del caso de negocio, debido a que podría generar mucha carga al ejecutarse por cada evento que suceda en el ETL.
 - **Programado:** Permite definir del mes o día de la semana, hora y minutos en que iniciará el proceso. Sin embargo, el disparador programado no funciona para ningún escenario de reabastecimiento. El desencadenante parece una elección perfecta, ya que puede definir claramente una frecuencia (en minutos, horas, días, semanas, meses) en la que desea ejecutar su ETL.
 - **Por eventos:** Permite realizar ejecuciones de dependencia de otros desencadenantes o políticas de reproceso. Desafortunadamente, la elección de la frecuencia es muy limitada y solo permite elegir entre minutos y horas. Además, no admite activar el primer o último día de un mes ni ejecutarlo en varios días específicos de la semana. Por lo tanto, un escenario como ese tendría que resolverse con necesidades de ejecución en un mismo día.

4. Del dato al Conocimiento

Los reportes y dashboards se consideran a veces como la solución para una organización, pero sin todo el trabajo previo que se ha realizado, difícilmente podríamos tener algo que analizar sin contar con un data warehouse. El objetivo de esta guía no es cubrir temas referentes a buenas prácticas de reporting y de visualización de datos, su enfoque radica en cómo, con un data warehouse, podemos realizar informes así como, temas muy interesantes de explotación de información ad-hoc de una forma sencilla y ágil para que el usuario de negocio final pueda contestar todas las preguntas que se formulen en el transcurso del uso de esta solución.

a) Reportes y Dashboard

La herramienta básica para el análisis clásico son los informes y documentos donde los usuarios pueden ver los datos planos filtrados, ordenados, con la posibilidad de exportarlos a Excel u otros entornos como Power Point, PDF, etc.

Microsoft ofrece una herramienta gratuita de BI en la nube, que permite crear informes y cuadros de mando llamada Power BI. Por medio de esta, se puede generar una capa visual y cubrir los requerimientos definidos por el negocio. También se tiene disponible Power BI PRO, la cual tiene un costo asociado y ofrece la posibilidad de trabajar en un entorno compartido, con seguridad y características adicionales a la gratuita.

Al igual que lo aplicado en bases de datos y en los procesos de datos, la estandarización juega un rol protagónico en este entorno visual, ya que garantiza que todos los reportes tengan el mismo Look & Feel y facilita la gestión del cambio en su utilización, pero esto no solo queda allí, siempre piense que tiene que contar una historia con sus datos y así cubrir las necesidades del negocio.

Con este fin se describen algunas prácticas para la definición del estándar visual de los reportes y Dashboard:

- **Video de buenas prácticas y consejos para diseñar un reporte en Power BI:** Dna, E. (2019, 25 marzo). Power BI Report Design Tips And Inspiration. YouTube. <https://www.youtube.com/watch?v=m5Qk4NLQNVA&feature=youtu.be>
- **Consejos de diseño en Dashboard:** M. (2019, 14 agosto). Tips for designing a great Power BI dashboard - Power BI. Microsoft Docs. <https://docs.microsoft.com/en-us/power-bi/create-reports/service-dashboards-design-tips>
- **7 errores comunes:** Milek, Z. (2021, 25 marzo). Power BI Dashboard Design: Avoid These 7 Common Mistakes. Zebra BI Financial Reporting in Power BI and Excel. <https://zebrabi.com/power-bi-dashboard-design/>

b) Definición de indicadores

Los indicadores clave de rendimiento (KPI) cuantifican el rendimiento de una organización o de sus procesos para alcanzar los objetivos empresariales. Ejemplos típicos de KPI son el número de defectos de un producto, la satisfacción del cliente con un servicio, el margen de beneficio de un producto, etc. Todos estos ejemplos se refieren en cierto modo a una actividad o a conjuntos de actividades, en las que se presenta la interacción de múltiples objetos.

Como se explicó anteriormente, un Data Warehouse consiste en hechos multidimensionales que representan observaciones medibles sobre entidades en el tiempo. Las entidades y el tiempo forman las dimensiones, y las medidas representan las observaciones sobre las entidades. Si se reduce a una sentencia, un Data Warehouse es esencialmente una gran colección de medidas que cubren una determinada parte de la realidad, las cuales, en la mayoría de las veces se refieren a procesos.

Lo interesante aquí es, como diseñar los KPI a partir de los eventos del Data Warehouse y que cumplan con las características de estos:

- Una premisa de los KPI es su naturaleza resumida. Un KPI no se basa en una única observación arbitraria, sino que agrega un gran número de observaciones sobre las mismas entidades (o actividades) para que sean estadísticamente significativas. El concepto de observación es el bloque de construcción atómico de los KPI (mínimo nivel de detalle en los datos usados).
- Un KPI simple, se fundamenta en un único tipo de observación donde se tienen una o varias entidades que participan en la observación y una métrica de valor asociada a la observación, normalmente un número. Por lo tanto, una observación es un hecho, las dimensiones, las entidades y, el valor, las métricas disponibles.
- Los KPI derivados se generan con base en cálculos de los KPI más simples.
- Los KPI tienen asociado al menos un objetivo y unos umbrales (positivo, aceptable y negativo).

Se recomienda las siguientes lecturas asociadas a la implementación del concepto de KPI en la herramienta.

1. **Visualizaciones KPI en Power BI:** M. (2020, 30 enero). Key Performance Indicator (KPI) visuals - Power BI. Microsoft Docs. <https://docs.microsoft.com/en-us/power-bi/visuals/power-bi-visualization-kpi>
2. **Uso de KPI en Power BI:** Erkec, E. (2018, 18 diciembre). Use of Key Performance Indicators in Power BI. SQL Shack - Articles about Database Auditing, Server Performance, Data Recovery, and More. <https://www.sqlshack.com/use-of-key-performance-indicators-in-power-bi/>

c) Diseño modelo de autoservicio

El auto-servicio es el principal entregable generado desde el entorno del data warehouse al entregable final, debido a que, por sus características, le permitirá gestionar los grandes volúmenes de datos de una forma sencilla y amigable usando herramientas como Excel.

Para que este modelo sea exitoso, se deberán cubrir los siguientes aspectos en el diseño de este:

- **Eliminar todas las columnas innecesarias:** En cada consulta, elimine todas las columnas innecesarias. No se preocupe si elimino una de más, en los

pasos que dejan las consultas puede volver a uno anterior y realizar el cambio.

- **Cambie el nombre para todos los nombres de tablas y columnas que serán visibles en el modelo de autoservicio:** los nombres técnicos en un modelo de análisis de información impiden el uso de los usuarios finales y requiere que estos tengan sesiones de entrenamiento que se pueden omitir con descripciones concisas y en lenguaje de negocio, por ejemplo: en vez de desc_clientes_nombre_apellido a Nombre Cliente
- **Establecer explícitamente el tipo de datos para todas las columnas:** Revise que las columnas se encuentran bien clasificadas, esto le puede evitar errores en conversión de datos o tipos de datos no compatibles, como puede ser tratar de sumar una variable numérica como Saldo que ha sido erróneamente clasificada como tipo String (cadena de caracteres).
- **Oculte los campos llave:** Como su nombre lo indica, los campos clave no deberían ser visibles, a menos que sea un valor que utiliza el negocio como puede ser el documento de identidad de un cliente.
- **Genere jerarquías en las dimensiones:** Esto permite de forma nativa que el usuario utilice un análisis drill down/up de los datos.
- **Genere carpetas para los indicadores:** Cree una carpeta independiente donde se incluyan todas las métricas en ese único lugar. Si tiene de diferentes tipos puede crear sub-carpetas y así facilitar el acceso por parte de los usuarios finales.
- **Defina el proceso de actualización:** Una vez se publique el modelo en power bi Service, calendarizar la periodicidad de actualización del Dataset.
- **La conexión al auto-servicio debe ser de fácil acceso:** Cree un entorno como teams, sharepoint o una página web donde el usuario pueda descargar el autoservicio de una forma amigable y sencilla.

Se recomiendan las siguientes lecturas asociadas al despliegue de esta solución.

1. **Power BI en Excel:** Microsoft. (2021f, mayo 28). Analyze in Excel for Power BI - Power BI. Microsoft Docs. <https://docs.microsoft.com/en-us/power-bi/collaborate-share/service-analyze-in-excel>
2. **Video de Introducción a Analyze in Excel.** SQLBI. (2021, 25 enero). DAX Tools - Analyze in Excel for Power BI 1 - Introduction. YouTube. <https://www.youtube.com/watch?v=BAOijlKE1UI>

Bibliografía

- Rahul, B. (2018). SQL Primer - An Accelerated Introduction to SQL Basics [Libro electrónico]. <https://doi.org/10.1007/978-1-4842-3576-8>
- Meier, A., & Kaufmann, M. (2019). SQL & Nosql Databases: Models, Languages, Consistency Options and Architectures for Big Data Management (2019 ed.). Springer Vieweg.
- Zimányi, E., & Abelló, A. (Eds.). (2016). Business Intelligence. Lecture Notes in Business Information Processing. Published. <https://doi.org/10.1007/978-3-319-39243-1>
- Prakash, N., & Prakash, D. (2018). Data Warehouse Requirements Engineering. Requirements Engineering for Data Warehousing. Published. <https://doi.org/10.1007/978-981-10-7019-8>
- Data Warehouse Requirements Engineering: A Decision Based Approach. (2018). Springer Nature Singapore. <https://doi.org/10.1007/978-981-10-7019-8>
- Microsoft. (s. f.). Azure Data Factory Documentation - Azure Data Factory. Microsoft Docs. Recuperado 6 de junio de 2021, de <https://docs.microsoft.com/en-us/azure/data-factory/>
- Microsoft. (2021, 16 marzo). Azure Data Factory tutorials - Azure Data Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/data-factory-tutorials>
- Adam Marczak - Azure for Everyone. (2019, 22 julio). Azure Data Factory Tutorial | Introduction to ETL in Azure. YouTube. <https://www.youtube.com/watch?v=EpDkxTHAhOs>
- Data Platform Discovery Day. (2020, 5 mayo). Paul Andrew: A Complete Introduction to Azure Data Factory at Data Platform Discovery Day April 2020. YouTube. <https://www.youtube.com/watch?v=CtDTJPbXVpQ>
- Microsoft. (2018, 5 julio). Pipeline execution and triggers in Azure Data Factory - Azure Data Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/concepts-pipeline-execution-triggers>

- Microsoft. (2021b, mayo 20). Mapping data flows - Azure Data Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/concepts-data-flow-overview>
- Microsoft. (2021b, abril 1). Expression functions in the mapping data flow - Azure Data Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/data-flow-expression-functions>
- Microsoft. (2021c, abril 19). Parameterizing mapping data flows - Azure Data Factory. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/data-factory/parameters-data-flow>