1. EL PROCESO DE ETL (EXTRACT, TRANSFORM AND LOAD)

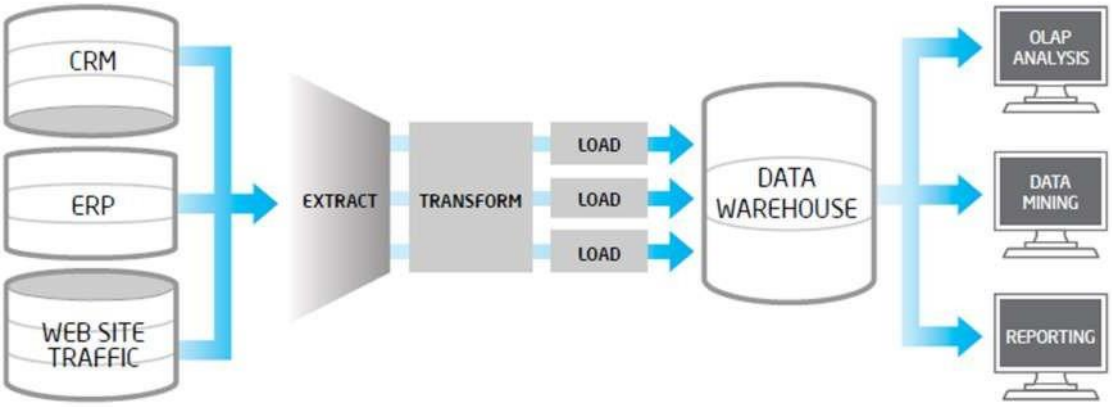
1.1 DEFINICIÓN

El proceso de ETL es una tecnología que tiene la función de integración de los datos, de modo que debe ofrecer una única visión de los datos. ETL corresponde a las siglas en inglés:

- Extract: Extracción.
- Transform: Transformación.
- Load: Carga.

Por lo tanto, como su propio nombre indica, este proceso se encarga de la extracción, transformación y carga de los datos. También tiene la función de gestionar estos datos. Debe asegurar su integridad, coherencia y disponibilidad en el destino.

Permite extraer datos del entorno de origen; según nuestras necesidades de negocio, transformarlos y cargar los datos en los entornos destino. Habitualmente, los entornos tanto de origen como de destino son ficheros y/o bases de datos, pero también pueden ser mensajes de cierto middleware, así como otras fuentes no estructuradas, semiestructuradas o estructuradas.



**Representación gráfica del proceso de ETL. Fuente: http://www.dataprix.com/blog-it/business-intelligence/integracion-datos/guia-procesos-eleccion-herramientas-etl*

Precisamente este proceso consta de estas tres fases ya mencionadas:

- **Fase de extracción**: se conectan sistemas de fuentes de datos y según los objetivos que hemos marcado, se analizan, recogen y procesan los datos. Los pasos que se deben seguir en esta fase son los siguientes:
  - Extraer los datos del sistema de origen.
  - Analizar los datos que han sido extraídos y chequearlos.
  - Interpretar ese chequeo para comprobar así si los datos cumplen con nuestros requisitos y lo que habíamos establecido previamente.
  - Si por el contrario, no cumplen nuestras pautas, esos datos serán rechazados.
  - Convertir los datos al formato adecuado para poder pasar a la siguiente fase; la transformación.

  En cuanto a esta fase, debemos diferenciar tres tipos de extracción:

  - **Full extract o extracción total**: se refiere a extraer la totalidad de los datos. Aquí se barren tablas completas que pueden llegar a tener millones de registros.

- **Incremental extract o extracción incremental**: en esta modalidad se va procesando por lotes únicamente lo que fue modificado o agregado. Además, puede haber filas que se borren por estar duplicadas o por tratarse de datos erróneos.
- **Update notification o notificación de actualizaciones**: en esta modalidad solo se van extrayendo los datos a medida que se produce una actualización, como puede ser un insert.

Es de gran importancia que la extracción de esos datos cause el menor impacto posible en el sistema de origen. Cuando se tienen que extraer muchos datos es posible que el sistema se ralentice o colapse, provocando que ya no pueda tener el mismo uso cotidiano que tenía. Por ello, cuando hay que extraer gran cantidad de datos, se suelen programar horarios o calendarios en los que se provoque el mínimo impacto.

- **Fase de transformación**: los datos mediante una serie de reglas, se convierten en formatos normalizados que pueden ser tratados. Estas reglas pueden estar basadas en excepciones o restricciones o pueden ser declarativas pero, para potenciar e incrementar su eficacia y pragmatismo, debemos asegurarnos que sean:
  - Declarativas.
  - Independientes.
  - Claras.
  - Inteligibles.
  - Útiles para el negocio.

  Veamos las diferentes acciones o procesos de transformación que podemos llevar a cabo:

  - **Reformateo de datos**.
  - **Conversión de unidades**: Por ejemplo, podría ser la conversión de distintas monedas (euros, libras, etc.) en un único valor estándar. Otro ejemplo sería la conversión de millas a kilómetros por hora o viceversa.

Esto es algo muy común al extraer datos de países con unidades métricas diferentes.

- **Selección de columnas para su carga posterior**: un ejemplo sería hacer que no se carguen las columnas con valores nulos.
- **Agregación de columnas**: un ejemplo sería añadir una columna con el lugar de procedencia de ciertos automóviles.
- **Dividir una columna en varias**: esto es de gran utilidad para, por ejemplo, dividir en tres columnas la identificación de una persona que está en un solo campo. Así, se puede usar una columna para el nombre y otras dos columnas para los apellidos.
- **Traducir códigos**: si por ejemplo, se almacena una "H" para hombres y una "M" para mujeres en la fuente de origen, hay que dar las instrucciones necesarias para que en destino se guarde un "1" para hombres y un "2" para mujeres.
- **Obtener nuevos valores calculados**.
- **Unir datos de diversas fuentes**.
- **Lookups**: tiene lugar cuando se compara un dato con otro tipo de datos, cruzando información. Un ejemplo sería capturar un código de cliente de una base de datos y cruzar este con otra base de créditos concedidos para saber si ese cliente disfruta o no de ese préstamo.
- **Pivoting**: es un proceso parecido al anterior, pero con mayor complejidad, puesto que se cruzan datos de diferentes fuentes de información.

- **Fase de carga**: es el proceso más complejo. En este, se importan los datos ya transformados a la estructura de almacenamiento que hemos seleccionado.

  Esta fase de carga puede desarrollarse de dos formas:

  - **Acumulación simple**: se realiza un resumen de las transacciones realizadas en el periodo seleccionado y se transporta el resultado hacia el Data WareHouse como una única transacción. Se almacenará un valor calculado que resultará la suma o promedio de la magnitud

considerada. Esta es la forma más sencilla de desarrollar el proceso de carga.

- **Rolling**: se almacena la información resumida a diferentes niveles, que corresponden a diferentes agrupaciones de la unidad de tiempo o distintos niveles jerárquicos en varias o alguna de las dimensiones de la magnitud almacenada. Por ejemplo, totales semanales, totales mensuales, etc.

Es de gran importancia tener presente que esta fase de carga interactúa con la base de datos de destino y en esa base de datos se aplicarán las restricciones que se hayan definido en este proceso. La calidad de los datos en el proceso de ETL estará garantizada si esas restricciones están bien definidas.

Finalmente, y aunque no se considere parte del proceso ETL, es interesante mencionar la fase de limpieza de datos. Se trata de una fase previa a todo el proceso ETL y de gran importancia, ya que nos permite asegurar que todos los datos de que disponemos son correctos.

Aunque trataremos esta fase más adelante, creemos interesante introducir sus ventajas. Estas son:

- Nos permite asegurar la calidad de los datos que vamos a procesar.
- Evita información no veraz y errónea.
- Ahorra costes de espacio en el disco debido a que gracias a esta fase podemos eliminar la información duplicada.
- Nos permite agilizar las consultas por la ausencia de datos repetidos o inservibles.
- Ayuda en la toma de decisiones estratégicas.

Comentar que no es posible lograr un buen resultado en un proceso ETL en sintonía con los objetivos marcados si antes no hemos llevado a cabo este proceso de limpieza de datos que nos proporcionará una base de datos de calidad que nos permitirá poder tomar decisiones acertadas en los niveles estratégico y ejecutivo.

Antes de pasar a ver las herramientas que podemos utilizar para nuestros procesos ETL, resulta interesante presentar los beneficios que puede reportar a nuestra empresa llevar a cabo este proceso.

Dichos beneficios son:

- Nos permite crear un Master Data Management; un repositorio central estandarizado de todos los datos de la organización. De este modo, los mismos datos pertenecientes a un cliente y disponibles en varias bases de datos de la empresa se unificarán en una única base de datos, de modo que ello nos permita disponer de toda la información en un mismo sitio.

- Unificar todos los datos en un único sitio, permite a los directivos tomar decisiones estratégicas basadas en el análisis de los datos cargados en las nuevas bases de datos.

- Este proceso sirve para integrar sistemas. Las empresas y organizaciones crecen de manera orgánica, y de cada vez se van agregando más fuentes de datos, provocando ello que surjan nuevas necesidades en relación a estos datos, como puede ser la integración de los datos nuevos con los antiguos.

- Nos permite tener una visión global de todos los datos consolidados en el Data Warehouse, lo que nos permitirá una buena planificación estratégica.

1.2 HERRAMIENTAS ETL

Conforme los Data WareHouse iban ganando importancia en las grandes corporaciones, la programación de los procesos ETL empezó a estar compuesta de un gran número de líneas de código que los hacía muy difíciles de mantener. Esta dificultad para mantenerlos y la lenta curva de aprendizaje que tienen los mismos causó que se buscasen alternativas.

Las empresas más importantes en el sector de los sistemas de información deciden, a mediados de los 90, invertir y desarrollar sus propias herramientas. Empresas como Informativa, IBM, SAS u Oracle empiezan a lanzar herramientas potentes orientadas al

desarrollo y diseño de procesos ETL sin necesitar específicamente programas en código. De este modo, nacen Informativa PowerCenter, ODI (Oracle Data Integrator), IBM Datastage o SAS Data Integrator.

Estos software despuntan por su alto coste de licencias, limitando así su nicho de mercado a las grandes empresas, destacando por su fiabilidad. No obstante, el incremento de los sistemas de Business Intelligence (BI) dentro de las compañías de presupuestos más modestos provocó que las empresas dedicadas al Software OpenSource se centrasen en el mundo de las ETL.

Las herramientas ETL documentan cómo los datos son transformados (si lo son) entre el origen y el destino, almacenando la información en un catálogo propio de metadatos. Estos metadatos los intercambian con otras aplicaciones que puedan necesitarlos y administran todos los procesos y ejecuciones de la ETL: log de errores, planificación de la transportación de datos, log de cambios y estadísticas asociadas a los procesos de movimientos de datos.

Las herramientas ETL permiten diseñar, administrar y controlar todos los procesos del entorno ETL.

Ejemplos de herramientas ETL OpenSource son KETL, Talend, Jaspersoft ETL, Scriptella, y la herramienta OpenSource por excelencia, Kettle (Pentaho Data Integrator):

**Representación gráfica de distintas herramientas ETL. Fuente: http://pentahotutorial.blogspot.com.es/*

Ventajas de estas herramientas:

- **Entorno intuitivo y visual:** al permitir seguir y diseñar el flujo y transformación de datos de este modo, se incrementa la velocidad de desarrollo de los procesos.
- **Agilidad en la depuración de errores de desarrollo.**
- **Mantenimiento**: la interfaz gráfica de las herramientas hace más sencillas las tareas de mantenimiento.
- **Operaciones y capacidades de administración**: la administración de errores tiene lugar mediante logs y estadísticas de ejecución.
- **Conectividad**: estas herramientas hacen más fácil la conexión a los distintos sistemas de origen. Bases de datos, ficheros XML, páginas web, etc.
- **Manejo de metadatos y modelos**: pueden haberse creado por herramientas externas o por la propia herramienta.
- **Planificación global de conjuntos de procesos**: permiten la programación en tiempo real o batch, administración de excepciones o lanzamiento de eventos disparadores entre otras cosas.
- **Interfaces con sistemas Frontoffice**.

- **Interfaces de datos con sistemas externos**: envío de información a proveedores, clientes, recepción, proceso e integración de la información que se recibe.
- **Capacidades SOA**: es la arquitectura orientada a servicios, que establece una estructura de diseño para integrar aplicaciones y que permite a las organizaciones o compañías unir sus objetivos de negocio en cuanto a flexibilidad de integración con alineación directa a los procesos de negocio, con la infraestructura TI y con sistemas legados. Esto también permite reducir los costes de implementación, adaptación rápida frente a los cambios y reacción ágil ante la competitividad e innovación de los servicios a clientes. Todo esto se da gracias a que las nuevas tecnologías combinan fácilmente con aplicaciones independientes, permitiendo de este modo que los componentes del proceso se puedan coordinar e integrar de modo efectivo y rápido.
- **La descentralización del control de la ejecución y de todos los procesos**.

Las herramientas ETL pueden ser útiles para diferentes propósitos y no únicamente para entornos Data Warehousing o en la construcción de un Data WareHouse, como por ejemplo:

- **Tareas de Bases de datos:** se usan para consolidar, sincronizar y migrar bases de datos operativas.
- **Migración de datos en distintas aplicaciones** debido a cambios de versión o cambio de aplicativos.
- **Sincronización entre diferentes sistemas operacionales.**
- **Consolidación de datos:** los sistemas con volúmenes grandes de datos se consolidan en sistemas paralelos para procesos de borrado en los sistemas originales o para mantener así históricos.
- **Interfases de datos con sistemas externos:** envío de información a proveedores y clientes. Recepción, proceso e integración de la información recibida.
- **Interfases con sistemas Frontoffice:** interfaces de subida y bajada con sistemas de venta.

- **Otros:** preparación de procesos masivos como newsletter o mailings, actualización de usuarios a sistemas paralelos, etc.

Según Gartner, las características más importantes que debe incluir un software ETL son las que siguen:

- **Conectividad / capacidades de adaptación (con soporte a orígenes y destinos de datos):** habilidad para conectar con distintos tipos de estructura de datos, tanto bases de datos relacionales como no relaciones, diferentes formatos de ficheros, XML, colas de mensajes, páginas web, emails, repositorios de contenido, aplicaciones ERP, CRM o SCM o herramientas de ofimática.
- **Capacidades de entrega de datos:** habilidad para facilitar datos a otras aplicaciones, bases de datos o procesos en varias formas, con capacidades para programación de procesos batch, mediante el lanzamiento de eventos o en tiempo real.
- **Capacidades de transformación de datos:** habilidad para transformar los datos, ya sean transformaciones básicas como conversión de tipos, cálculos simples o manipulación de cadenas; transformaciones intermedias como sumarización, agregaciones o lookups; o transformaciones más complejas como análisis de texto en formato libre o texto enriquecido.
- **Capacidades de Metadatos y Modelado de Datos:** recuperación de los modelos de datos desde los inicios de datos o aplicaciones, creación y mantenimiento de modelos de datos, mapeo de modelos físico a lógico, repositorio de metadatos abierto con la posibilidad de interactuar con otras herramientas, sincronización de los cambios en los metadatos en los diferentes componentes de la herramienta, documentación, etc.
- **Capacidades de diseño y entorno de desarrollo:** representación gráfica de los modelos de datos, objetos del repositorio y flujos de datos, soporte para test y debugging, gestión de workflows de los procesos de desarrollo, capacidades para trabajar en equipo, etc.
- **Capacidades de gestión de datos:** perfiles, calidad de datos y minería de datos.

- **Adaptación a las distintas plataformas hardware y sistemas operativos existentes:** mainframes (IBM Z/OS), AS/400, HP Tandem, Unix, Wintel, Linux, Servidores Virtualizados, etc.
- **Las operaciones y capacidades de administración:** habilidades de gestión, control de los procesos de integración de datos y monitorización, como recolección de estadísticas de ejecución, controles de seguridad, gestión de errores, etc.
- **La arquitectura y la integración:** grado de compactación, interoperabilidad y consistencia de los distintos componentes que forman la herramienta de integración de datos con un mínimo de productos, un único repositorio, interoperabilidad con otras herramientas o vía API, un entorno de desarrollo común, etc.
- **Capacidades SOA:** como ya hemos dicho antes, SOA es la arquitectura orientada a servicios, que establece una estructura de diseño para integrar aplicaciones y que permite a las compañías unir sus objetivos de negocio, en cuanto a flexibilidad de integración con alineación directa a los procesos de negocio, con la infraestructura TI y con sistemas legados. Esto también permite reducir los costes de implementación, adaptación rápida frente a los cambios y reacción ágil ante la competitividad y la innovación de los servicios a clientes. Todo esto se da gracias a que las nuevas tecnologías combinan fácilmente con aplicaciones independientes, permitiendo de este modo que los componentes del proceso se puedan coordinar e integrar de manera efectiva y rápida.

Las herramientas ETL han ido evolucionando y ahora incluyen más funcionalidades propias de una herramienta de integración de datos. Podemos destacar las siguientes:

- Servicios de entrega/acceso de datos (mediante conectores o adaptadores).
- Gestión de servicios.
- Data profiling.
- Data quality.
- Procesos operacionales.
- Servicios de transformación: CDC, SCD, validación, agregación.

- Servicios de acceso a tiempo real.
- Extract, Transform and Load (ETL).
- Enterprise Information Integration (EII).
- Enterprise Application Integration (EAI).
- Capa de transporte de datos.
- Gestión de metadatos.

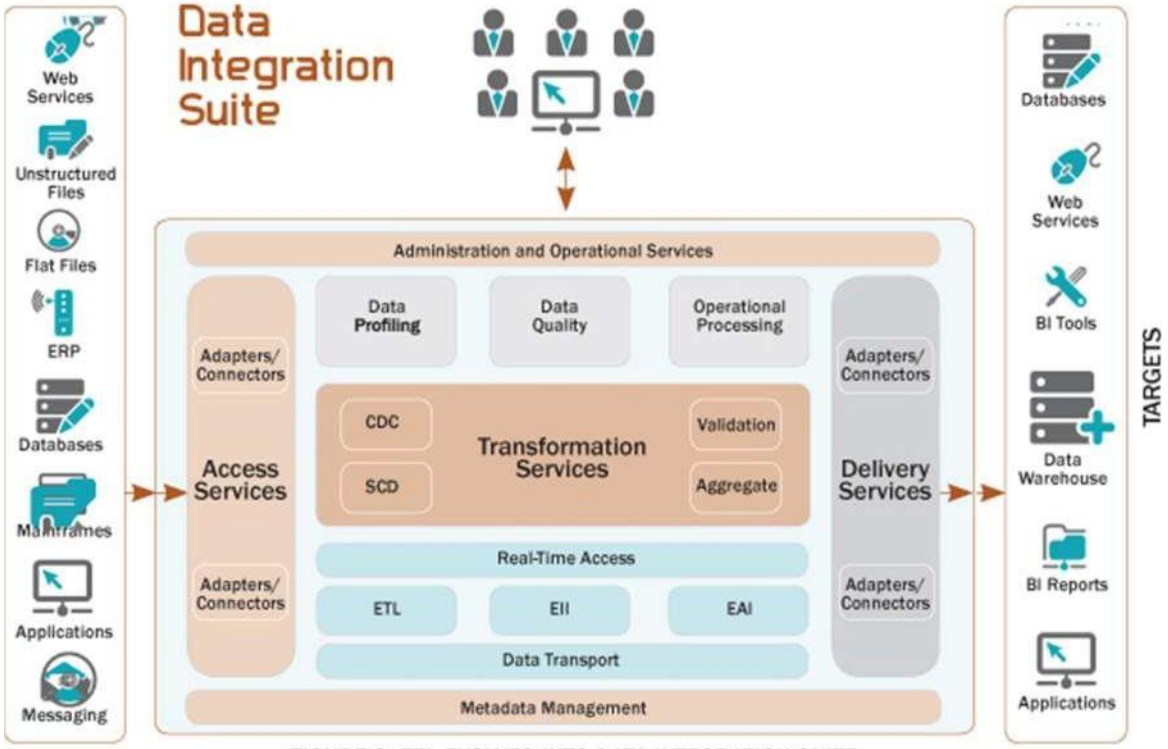


FIGURE 2: ETL EVOLVES INTO DATA INTEGRATION SUITE

**Representación gráfica de Integración de datos. Fuente: https://gerardnico.com/wiki/_media/data/processing/data_integration_suite.gif?cache=*

Algunas herramientas ETL:

- Ab Initio.
- Beneti.
- BI Tool – ETL Software.
- CloverETL.
- Cognos Decisionstream (IBM).

- Data Integrator (herramienta de Sap Business Objects).
- ETI Solution (anteriormente era ETI*Extract).
- IBM Webshere DataStage (anteriormente Ascential DataStage).
- Microsoft Integration Services.
- Oracle Wrehouse builder.
- WebFocus-iWay DataMigrator Server.
- Pervasive.
- Informatica PowerCenter.
- Oxio Data Intelligence ETL full web.
- SmartDB Workbench.
- Sunopsis (Oracle).1
- SAS Dataflux.
- Sybase.
- Syncsort: DMExpress.
- Opentext (anteriormente Genio, Hummingbird).

1.2.1 Procesamiento en herramientas ETL

Para mejorar el rendimiento de los procesos ETL en grandes volúmenes de datos se ha desarrollado en el software ETL la aplicación de procesamiento paralelo. Existen tres tipos de paralelismos que se pueden implementar en las aplicaciones.

Estos son:

- **De datos**: se divide un único archivo secuencial en pequeños archivos de datos para así proporcionar acceso paralelo.
- **De segmentación (pipeline):** permite el funcionamiento simultáneo de diversos componentes del mismo flujo de datos.
- **De componente:** permite el funcionamiento simultáneo de múltiples procesos en distintos flujos de datos en el mismo puesto de trabajo.

Estos tres tipos de paralelismo pueden combinarse para realizar una misma operación ETL.

Es necesario que en un sistema de ETL se puedan detener ciertos datos hasta que todas las fuentes estén sincronizadas. También cuando un almacén de datos debe ser actualizado con los contenidos en un sistema de origen, se necesitan establecer puntos de actualización y sincronización. Las múltiples y distintas bases de datos de origen tienen distintos ciclos de actualización (unas pueden actualizarse cada pocos minutos y otras pueden tardar semanas o días). La dificultad reside en asegurar que los datos que se cargan son relativamente consistentes.

1.3 RETOS EN LOS PROCESOS Y HERRAMIENTAS ETL

Hay que tener en cuenta que los procesos ETL pueden ser muy complejos, y es que un sistema ETL que está mal diseñado puede provocar problemas operativos notorios.

Es posible que en un sistema operacional la calidad de los datos o el rango de valores no coincida con las expectativas de los diseñadores en el momento de especificar las reglas de transformación o validación. Por ello, es importante y recomendable realizar una evaluación completa de la validez de los datos (Data profiling) del sistema de origen durante el análisis para identificar así las condiciones que se precisan para poder tratar los datos de manera adecuada por las reglas de transformación especificadas. Esto nos llevará a modificar las reglas de validación implementadas en el proceso ETL.

Los Data WareHouse, normalmente, son alimentados de forma asíncrona desde diferentes fuentes, que sirven a propósitos distintos. El proceso ETL es vital para lograr que los datos extraídos de manera asíncrona de orígenes heterogéneos se integren en un entorno homogéneo.

Durante el análisis se debe establecer la escalabilidad de un sistema ETL durante su vida útil. Esto también incluye la comprensión de los volúmenes de datos que deberán ser procesados según los acuerdos de nivel de servicio (SLA: Service level agreement). Es posible que el tiempo para realizar la extracción de los sistemas de origen cambie, hecho que implica que la misma cantidad de datos deba ser procesada en menos tiempo. Hay algunos sistemas ETL que son escalados para poder procesar diversos

terabytes de datos con el objetivo de actualizar un Data WareHouse que es posible que contenga decenas de terabytes de datos. Los lotes que se procesaban a diario pueden pasar a procesarse en micro-lotes durante un mismo día o incluso a la integración con colas de mensajes o a la captura de datos modificados (CDC: change data capture) en tiempo real para una transformación y actualización continua. Todo esto es debido al incremento de volúmenes de datos.

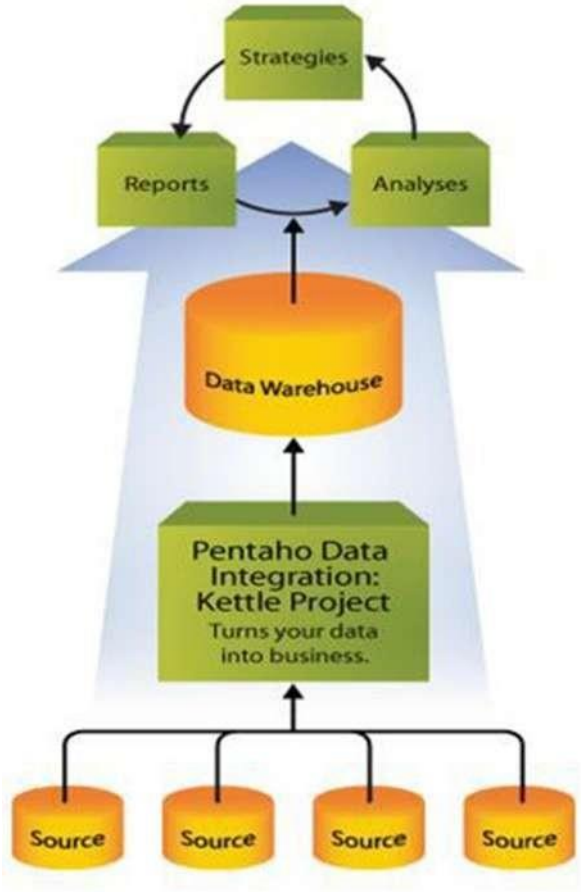
1.4 ETL EN CONTEXTO PENTAHO

Matt Casters inició Kettle en 2001 y en 2006 Pentaho Data Integration (PDI) adquirió Kettle y lo renombró después de que este pasara a ser open source.

Pentaho Data Integration (PDI) es una solución de integración de datos programada en java orientada por completo al usuario y que se basa en un enfoque de metadatos. Los procesos ETL son encapsulados en metadatos, los cuales se ejecutan mediante el motor ETL.

**Representación gráfica icon de Pentaho Data Integration (PDI). Fuente: http://www.dataversity.net/pentaho-and-mongodb-announce-native-integration/*

PDI nos permite cargar datos de varias fuentes de origen en un Data WareHouse para que después la información consolidada nos sea útil a nivel táctico, operativo y estratégico.



**Representación gráfica de la arquitectura Pentaho Data Integration. Fuente: https://gravitar.biz/pentaho/*

Las principales características de Pentaho Data Integration son las siguientes:

- Entorno gráfico orientado al desarrollo ágil y rápido basado en el área de trabajo y la de vista o diseño.
- Multiplataforma.
- Consta de varios conectores a bases de datos, tanto comerciales como propietarias. También incluye conectores a ficheros planos, XML, Excel u otros.
- Arquitectura extensible por medio de pluguins.
- Soporta procesos ETL en paralelo, uso de cluster y arquitecturas servidor maestro-esclavo.
- Integrado por completo con la suite de Pentaho.

- Basado en el desarrollo de dos tipos de objetos:
  - **Trabajos**: permiten la administración y la gestión de procesos ETL a un alto nivel.
  - **Transformaciones**: permiten definir las operaciones de transformación de datos.
- Está formado por cuatro componentes:
  - **Spoon**: entorno gráfico para desarrollar trabajos y transformaciones.
  - **Pan**: permite ejecutar transformaciones.
  - **Kitchen:** permite ejecutar trabajos.
  - **Carte**: servidor remoto que permite ejecutar trabajos y transformaciones.
- Pasos disponibles para trabajos:
  - **Generales**: permite comenzar un trabajo, ejecutar trabajos o transformaciones entre otras operaciones.
  - **Correo**: permite enviar correos, validar cuentas o recuperarlas.
  - **Gestión de fichero**: permite llevar a cabo operaciones con ficheros como crear, comparar, borrar o comprimir.
  - **Condiciones**: permite llevar a cabo comprobaciones necesarias para procesos ETL como la existencia de una carpeta, una tabla o un fichero.
  - **Scripting**: permite crear scripts de SQL, JavaScript y Shell.
  - **Carga Bulk**: permite llevar a cabo cargas Bulk a MSSQL, Acces, MySQL y ficheros.
  - **XML**: permite validar XML y XSD.
  - **Envío de ficheros**: permite coger o enviar ficheros desde SFTP y FTP.
  - **Repositorio**: permite llevar a cabo operaciones con el repositorio de trabajos y transformaciones.
- Pasos disponibles para transformaciones:
  - **Entrada**: permite la recuperación de datos desde bases de datos (JDBC), CSV, Acces, Excel ficheros, Mondrian, RSS, LDAP u otros.
  - **Salida**: permite la carga de datos en bases de datos y en otros formatos de salida.

- **Transformar**: permite llevar a cabo operaciones con datos como filtrar, mapear, ordenar, añadir nuevos campos, etc.
- **Utilidades**: permite operar con columnas o filas y también otras operaciones como escribir a un log o enviar un email.
- **Flujo**: permite llevar a cabo operaciones con el flujo de datos como fusionar, la detección de flujos vacíos, llevar a cabo operaciones distintas en función de una condición, etc.
- **Scripting:** permiten la creación de scripts de JavaScript, SQL, expresiones regulares, fórmulas y expresiones java.
- **Búsqueda de datos**: permite incorporar información al flujo de datos por medio de la búsqueda en bases de datos y otras fuentes.
- **Uniones**: permite la unión de filas dependiendo de distintos criterios.
- **Almacén de datos**: permite trabajar con dimensiones SCD.
- **Validación:** permite validar datos, direcciones de correo, tarjetas de crédito o XSD.
- **Estadística**: permite llevar a cabo operaciones estadísticas sobre un flujo de datos.
- **Trabajos**: permite llevar a cabo operaciones propias de un trabajo.
- **Mapeado**: permite llevar a cabo el mapeo entre campos de entrada y campos de salida.
- **Embebido:** permite llevar a cabo operaciones con sockets.
- **Experimental**: incluye aquellos pasos que están en fase de validación.
- **Obsoleto**: incluye aquellos pasos que en la siguiente versión del producto desaparecerán.
- **Carga bulk**: permite llevar a cabo cargas bulk a Infobright, LucidDB, Oracle y MonetDB.
- **Historial**: recopila los pasos habitualmente usados por el desarrollador.

1.4.1 Carga de los datos (Data Load)

En este apartado vamos a ver cómo se llevaría a cabo una carga de datos partiendo de una fuente de datos y cómo cargar estos datos en un entorno conocido.

Ya hemos presentado Kettle y también hemos comentado que funciona mediante las llamadas Transformaciones y Trabajos o Jobs.

Transformaciones

Una transformación es un conjunto de pasos que pueden conectarse o no entre sí, dependiendo de su relación por medio de lo que llamamos saltos. Estos pasos terminan en uno o varios destinos y pueden ser ejecutados de modo secuencial o en paralelo. Estos pasos son los que mueven los datos entre sí.

Para la creación de transformaciones se dispone de las opciones siguientes:

- Presionar las teclas CTRL-N.
- Seleccionar en el menú principal la opción: Fichero- Nuevo- Transformación.
- Hacer click en el botón que se encuentra en la barra de herramientas.

Cuando nos aparezca la pantalla para crear una nueva transformación dispondremos de distintas opciones para la creación de pasos, que veremos en el menú que aparecerá a nuestra izquierda. Las principales y más destacables opciones son:

- **Input**: son las fuentes de información u orígenes a las que nos conectamos para obtener datos. Por ello, esta es la parte principal de extracción de esta herramienta ETL. Pentaho ofrece distintas posibilidades como origen de datos. Aquí veremos los más habituales para llevar a cabo procesos ETL:
    - **Table input**: para leer los datos en una base de datos. Estas bases de datos acostumbran a ser bases de datos relacionales que tenemos en nuestra organización.
    - **CSV input**: son ficheros de tipo CSV que se usan para la extracción de información por distintos programas.

  - **Text file input**: es semejante al anterior, pero en este no tenemos la limitación por tipo de fichero.

- **Output**: son los destinos de los datos, es decir, la parte principal de la carga de esta ETL. Veamos los destinos de datos más habituales que nos ofrece para realizar procesos ETL:
  - **Table output**: para escribir datos en una base de datos que escojamos nosotros.
  - **Text file output**: permite escribir los datos en un archivo de texto plano.
  - **Insert/Update**: nos sirve para escribir los datos en una base de datos, sobrescribiendo los datos si el registro ya ha sido insertado o insertando el registro si no ha sido insertado aún. Esto es útil ante las posibles actualizaciones de registros.
  - **Update**: Si el registro ya ha sido insertado, sobrescribe los datos.
  - **JSON Output**: escribe en formato de texto compatible con WebServices.
  - **XML Output**: escribe en formato de texto que es compatible con WebServices.
- **Transform**: pasos para transformar los datos de modo que nos sean útiles dependiendo de nuestras necesidades. Se pueden hacer cálculos con los datos que recibimos, añadir datos estáticos, secuencias numéricas, concatenar distintos campos, etc. Debido a esto, esta es la parte más importante de las transformaciones de esta ETL. Veamos las transformaciones de datos más habituales que nos ofrece para realizar procesos ETL:
  - **Calculator**: permite llevar a cabo operaciones de fechas en los campos de tipo fecha que vienen de origen u operaciones matemáticas con los campos numéricos que también nos vienen en origen.
  - **Concat Fields**: concatena uno o varios campos en uno solo. Un ejemplo serían los nombres y apellidos de un cliente, que podemos tenerlos en dos campos y querer juntarlos en uno. En dicho caso, se utilizaría esta opción.
  - **Replace in string**: remplaza una cadena de texto por otra que escojamos nosotros.

  - **Select values**: permite seleccionar los campos que queramos en el caso que el origen nos dé mucha más información de la que queramos.
  - **Sort rows**: permite ordenar los campos.
  - **Unique rows**: nos permite desechar de forma automática los registros o filas iguales que nos lleguen de origen y quedarnos únicamente con un registro.
- **Big Data**: encontramos aquí los pasos principales para escribir y leer datos en bases de datos consideradas Big Data. Los pasos anteriores también sirven para Big Data, pero estos se centran exclusivamente en bases de datos para Big Data y los otros sirven para leer múltiples fuentes distintas y controlar los datos y transformarlos como mejor nos vaya dependiendo de nuestras necesidades.

Veamos los pasos más habituales para transformar los datos:

- **Flow – Filter Rows**: para filtrar los registros que queremos o no para nuestro resultado final, o que queremos hacer pasar por otros pasos para llevar a cabo distintas transformaciones a los mismos.
- **Flow – Switch/Case**: es similar al anterior, pero nos permite enviar cada registro a un paso distinto en función de si cumplen o no una condición. Filter Rows puede desechar los registros que no cumplan la condición, en cambio aquí siempre pasarán todos los registros a un paso.
- **Scripting – Execute SQL script**: permite la ejecución de una consulta SQL[2] escrita manualmente.
- **Joins – Merge Join**: permite unir el resultado de dos pasos en uno mediante un campo común.
- **Statistics – Group by**: permite agrupar distintas filas en una, agrupando por un campo o varios en común y pudiendo llevar a cabo operaciones matemáticas en los campos que no agrupamos.

Trabajo o Job

Por su parte, un Trabajo o Job es un conjunto de transformaciones que se ejecutan de modo secuencial. Por tanto, un Trabajo no mueve los registros o datos, sino secuencias de tareas.

1.4.2 Calidad de los datos (Data Quality)

Cuando hablamos de la calidad de los datos, nos referimos a las técnicas, métodos, procesos, algoritmos y operaciones orientadas a una mejora en la calidad de los datos existentes. Con esto conseguimos:

- Potenciar acciones de marketing.
- Optimizar la fidelización de nuestros clientes.

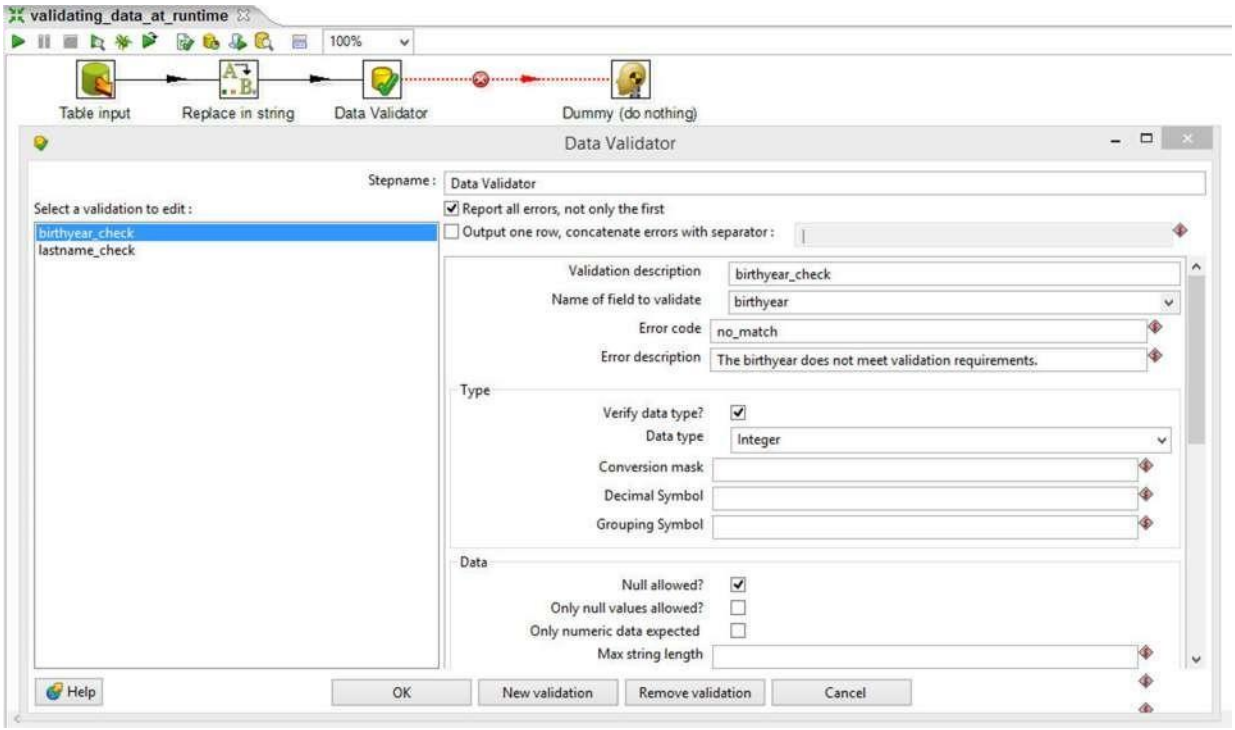
La calidad de los datos no solo hace referencia al hecho de que no haya defectos en estos, sino que los datos deben:

- Proporcionar una única visión.
- Ser consistentes.
- Ser completos.
- Ser adecuados para su función.
- Estar relacionados de forma correcta con todas las fuentes.
- Cumplir las leyes y normativas.

Al hablar de calidad de datos, debemos diferenciar entre dos vertientes: la validación de los datos y la limpieza de datos.

La **validación de los datos** intenta rechazar los registros erróneos durante la entrada al sistema. Por su parte, el proceso de **limpieza de los datos**, del que hablaremos en el siguiente apartado, consta también de la validación de datos, así como de la corrección de los datos o eliminación de estos para lograr datos de calidad.

En cierto modo, podemos utilizar Pentaho como herramienta de Data Quality. Pentaho, por defecto, ya tiene distintos validadores de datos que nos permiten hacer validaciones sencillas con las que podremos transformar, aceptar o rechazar distintos registros en función de los datos que tengamos. De estos pasos, el más completo a utilizar es Data Validator, porque nos ofrece una gran cantidad de validaciones como, por ejemplo, definir un rango de valores numéricos aceptados (Maximum Value-Minimum Value) con lo que se puede, por ejemplo, validar si la edad es correcta. Data Validator también nos ofrece, por ejemplo, validar si no aceptamos valores en blanco (Null Allowed).



**Representación gráfica ejemplo Data Validator. Fuente: https://anonymousbi.wordpress.com/tag/etl/*

1.4.3 Limpieza de los datos (Data Cleaning)

A la hora de analizar grandes volúmenes de información o no tan grandes volúmenes, uno de los problemas con los que podemos encontrarnos es la calidad de los datos. Es fundamental una buena calidad de datos para lograr buenos informes, cuadros de mandos, etc.

Para solucionar estos problemas de calidad tenemos lo que llamamos Data Cleaning o Data Scrubbing, que es el proceso de limpieza de datos de los registros erróneos o equívocos que podemos tener. Hay múltiples problemas con los que podemos encontrarnos, por ejemplo, datos incorrectos o inexactos, incompletos, no lógicos, etc. Por ello, hay que buscar esos datos y modificarlos, eliminarlos o sustituirlos según las necesidades que tengamos.

De este modo, cuando encontramos que tenemos alguno o varios de estos problemas, debemos llevar a cabo un proceso de limpieza de la base de datos. Para ello tenemos que llevar a cabo un proceso de Data Cleaning para aquellas tablas en las que hayamos detectado problemas.

Estos problemas o errores se producen, normalmente, por:

- **Error en la introducción de los datos por parte del usuario**: son los más habituales al ser un error humano. Puede ser intencionado o no, pero ocurre a menudo. Pueden ser unas direcciones de email incorrectas o nombres mal escritos, por ejemplo.
- **Error en la transmisión de los datos o el almacenamiento de los mismos**: nos podemos encontrar con datos erróneos debido a los fallos en los equipos o un error al haber insertado el dato en el almacenamiento. Por ejemplo, si hemos definido que el campo *Email* del cliente puede ocupar 50 caracteres en la base de datos, y un cliente inserta una que sobrepasa esa cifra, pueden suceder dos cosas; o bien el problema se controla y no le permite al cliente hacer el registro mostrándole un mensaje que se lo señala, o que no se controle el problema y los datos del cliente se graben recortados, creando así un registro erróneo.

- **Error por distintas definiciones de datos en los diccionarios**: por ejemplo, pongamos que accedemos a una fuente de información en la que tenemos el registro de los clientes que se han dado de alta en nuestra página web en Estados Unidos. Al tener la página web en inglés, cuando seleccionan país de origen hemos cargado un diccionario de datos en inglés en el que aparecen los siguientes registros: USA/SPAIN/FRANCE. Y en nuestra página web en Argetnina, tenemos cargados para el mismo registro los siguientes datos: EEUU/ESPAÑA/FRANCIA. Al hacer la carga de los usuarios dados de alta en ambos países, tendremos datos inexactos que nos pueden llevar al error al ver las altas por países.

Para poder encontrar estos errores y anomalías en nuestra base de datos, vamos a llevar a cabo una **auditoría de datos**, que se refiere a la revisión de estos datos. Mediante la utilización de diccionarios de datos y métodos estadísticos vamos a encontrar esas anomalías en nuestra base de datos.

La auditoría de base de datos es llevada a cabo por los auditores de sistemas y consiste en realizar una evaluación de los accesos a los datos almacenados en las bases de datos con el objetivo de poder monitorear, medir y tener constancia de los accesos a la información almacenada en esas bases de datos. El fin o el objetivo primordial en todos los casos, es lograr la seguridad corporativa. Hacer este control de datos es más complejo debido a las nuevas tecnologías, por ello normalmente es llevado a cabo por expertos que, por lo general, son externos a la organización o compañía.

Para que una auditoría sea realizada de manera correcta será necesario seguir los siguientes pasos:

- Detección de datos incorrectos.
- Limpieza.
- Normalización.
- Deduplicación.
- Integración.

Lo primero que debemos hacer es determinar las posibles casuísticas de datos erróneos. El hecho de que la mayoría de datos e informaciones provienen, normalmente, de una base de datos operativa, lo más habitual es que estas casuísticas estén ya identificadas. No obstante, es importante ejecutar procesos tales como conteos de datos nulos o acumulados, ya que ello puede evitarnos sorpresas de última hora. Facilitando así nuestro trabajo.

La fase de limpieza se refiere a eliminar todos aquellos registros que no son válidos. Se aconseja que toda la información que se elimina se guarde como copia de seguridad, por si en un futuro dicha información se requiere. Como registros no válidos nos referimos a información no válida con campos nulos o incorrectos o que presentan datos aislados.

Seguidamente, deberemos llevar a cabo una fase de normalización, que nos permitirá que todos los valores que hagan referencia a un mismo dato tengan la información unificada en un solo valor. Por ejemplo, en el campo referido al país de origen, deberá constar dicho país en un único idioma, así, optaremos por España o Spain, nunca por los dos.

El proceso referido a deduplicación se refiere a la identificación de posibles datos duplicados. Estos datos duplicados los dejaremos en cuarentena hasta que sean desestimados y eliminados según criterios determinados. En algunas ocasiones, será necesario realizar procesos de cálculo paralelos, como sumas, conteos, medias y porcentajes, con el fin de agrupar los diferentes valores contenidos en cada uno de los distintos códigos que tiene el cliente.

Si realizamos correctamente todos estos pasos que conforman la auditoría de datos, aseguraremos la integridad y coherencia de los datos, de modo que podremos proceder a la fase de carga en nuestro proceso de ETL.

Comentar que para llevar a cabo todas estas fases, disponemos de toda una serie de herramientas en el mercado que nos facilitan estos procesos. Dichas herramientas nos

permiten automatizar estos procesos, permitiéndonos diseñar un flujo que hace que los cambios en la ETL tengan un mínimo impacto en el coste de desarrollo.

Una vez hayamos realizado estos procesos, se finaliza la parte de ETL y obtención de datos. De este modo, estaremos en disposición de tratar y analizar la información. Comentar que en esta primera fase, el objetivo clave es asegurar que los datos se depuran e importan de manera correcta para su posterior exploración y aplicación.

Hecha la auditoría y una vez hayamos detectado el problema, definiremos el motivo de esa anomalía, qué la ha producido y de qué modo podemos corregirlo. Para corregir estos problemas existen diferentes métodos:

- **Transformación**: se puede cambiar el valor de un valor a otro en función de aquello que nosotros parametricemos mediante el uso de algoritmos matemáticos básicos o de un diccionario. Si por ejemplo, en nuestra base de datos nos aparece como país de origen ESPAÑA y nos viene SPAIN, podemos tener en el proceso de limpieza el cambio de un valor por el otro.
- **Eliminación**: habitualmente, se realiza la eliminación de registros a causa de registros duplicados. Estos pueden suceder por fallos al introducir los datos, ya sea un error automático o manual. Normalmente, las tablas tienen un registro o más que actúan como clave primaria que no se puede duplicar, pero puede ocurrir que la clave primaria no se haya activado en la tabla o que no sea correcta.
- **Estadística**: el uso de funciones matemáticas que contienen desviaciones, rangos, promedios o algoritmos y que son llevadas a cabo por expertos en la materia, nos puede llevar a comprobar si los datos son correctos o no, aunque es complejo, su valor estadístico lo determina. Del primer punto que hemos comentado, la transformación, se diferencia porque en la primera sabemos que un dato es erróneo de manera fehaciente. Si por ejemplo, la edad de una persona es -2, no precisamos un algoritmo muy complejo para saber que ese dato es erróneo.

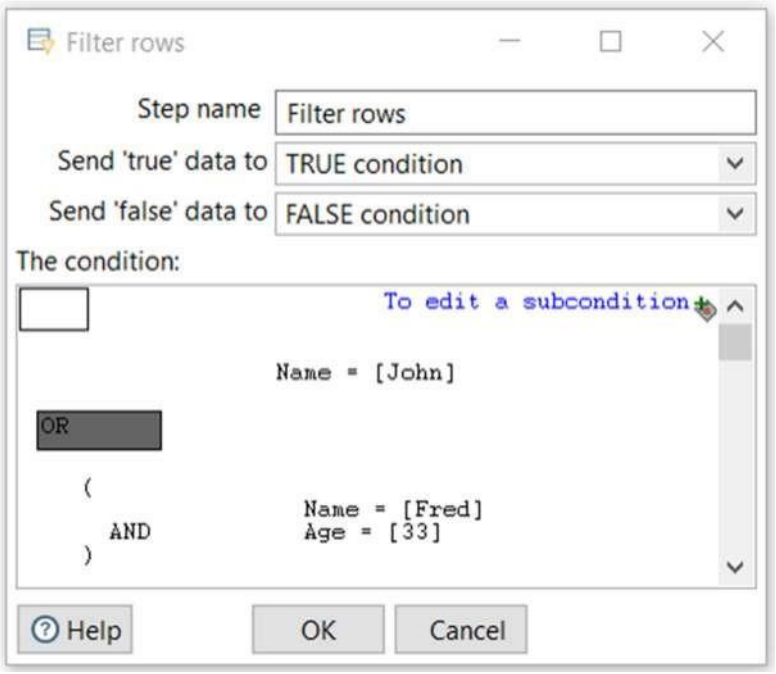
- **Análisis**: para detectar los errores de sintaxis se usa un analizador gramatical que decide si una frase es aceptable o permitida. Es parecido a lo que usamos para revisar la ortografía o la sintaxis con un programa de tratamiento de texto.

Cuando ya hemos definido el problema, debemos ejecutar el proceso para corregir esos errores o eliminar registros. Hay que tener en cuenta cómo se ha planteado el proceso ya que es posible que un proceso se haya planteado perfectamente, pero que la ejecución se alargue más de la cuenta.

El proceso de Data Cleaning no es un proceso que solo se ejecute una vez, sino que debemos mantener un control. Habitualmente, se establece cada cuánto tiempo vamos a hacer el proceso de Data Cleaning. Este siempre será abierto a modificaciones por si aparecen nuevas anomalías o casuísticas que debamos controlar en nuestros sistemas.

Podemos usar la herramienta Pentaho Data Integration como herramienta de Data Cleaning. Anteriormente ya hemos comentado que esta incluye la validación de datos, por lo que podemos seguir los mismos pasos para un proceso de limpieza u otros de diferentes:

- **Flow – Filter Rows**: ya hemos comentado anteriormente este paso. Nos sirve para filtrar los registros que queremos o no para nuestro resultado final, o qué queremos hacer para pasar por otros pasos para llevar a cabo distintas transformaciones a los mismos.

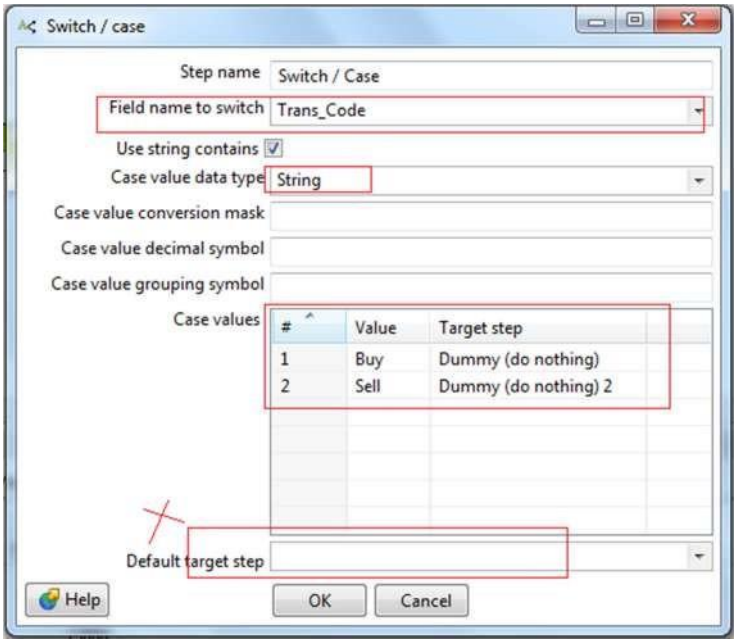


**Representación gráfica de un ejemplo Filter Rows. Fuente:*

http://kettle.bleuel.com/2017/02/16/metadata-injection-examples-for-special-scenarios/

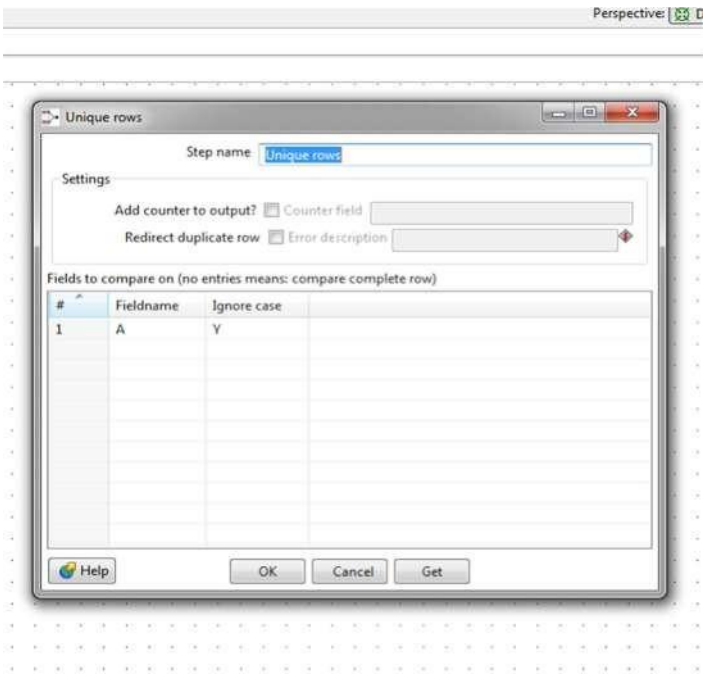
- **Flow – Switch / Case**: como sucede con el anterior, podemos usar este paso para llevar los registros que tengan o no cierto valor hacia un camino u otro.

  Es similar al anterior, pero nos permite enviar cada registro a un paso distinto en función de si cumplen o no una condición. Filter Rows puede desechar los registros que no cumplan la condición, en cambio aquí siempre pasarán todos los registros a un paso.



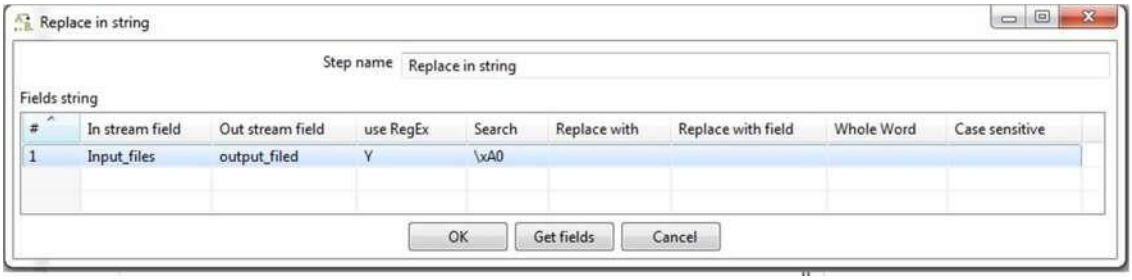
**Representación gráfica de un ejemplo Switch/Case. Fuente: http://pentaho-bi-suite.blogspot.com.es/2015/05/integration-work-out-use-case-solved.html*

- **Transform – Unique Rows**: este paso nos permite, en caso de tener registros repetidos, eliminar los sobrantes.



**Representación gráfica de un ejemplo Unique Rows. Fuente: http://pdiby.blogspot.com.es/2015/04/the-difference-between-unique-rows-and.html*

- **Transform – Replace in string**: con este paso podemos reemplazar una parte de un campo que veamos o consideremos erróneo por otro que no lo sea.

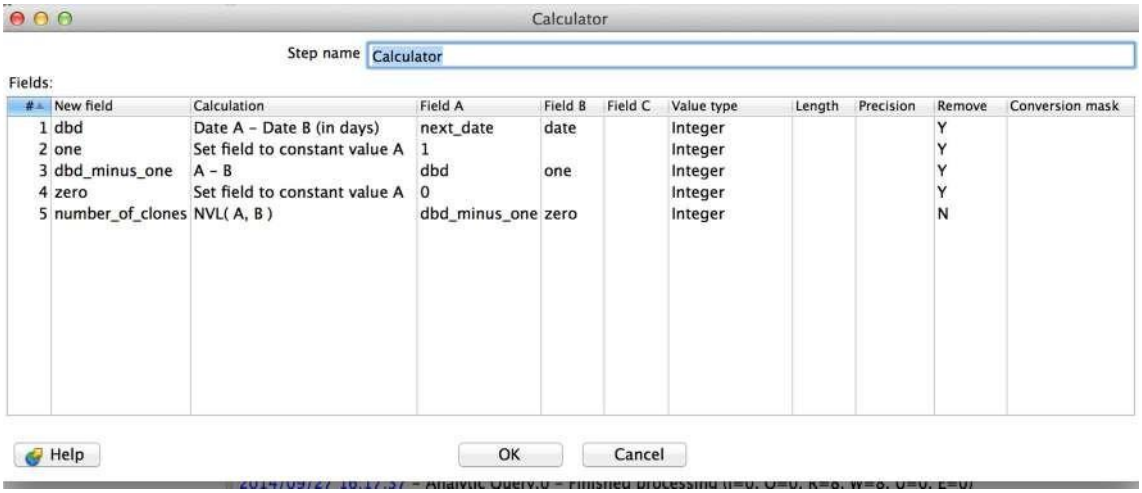


**Representación gráfica de un ejemplo de Replace in string. Fuente: http://data-analytics4u.blogspot.com.es/2015/06/remove-non-breaking-space-in-data.html*

- **Transform – String Operations**: nos permite llevar a cabo diversas operaciones con un campo. Como por ejemplo, poner el campo entero en mayúsculas o en minúsculas, eliminar caracteres en blanco o eliminar caracteres especiales.

**Representación gráfica de un ejemplo String Operations. Fuente: https://codigosdejavo.wordpress.com/2014/03/09/3-cosas-de-pentaho-data-integration-que-me-hubiera-gustado-saber-desde-un-principio/*

- **Transform – Calculator**: nos permite llevar cabo distintas operaciones con los datos que después podremos utilizar para evaluar los mismos.



**Representación gráfica de un ejemplo de Calculator. Fuente: https://stackoverflow.com/questions/26040054/filling-data-gaps-in-a-stream-in-pentaho-data-integration-is-it-possible*

Estos que hemos comentado aquí, tan solo son algunos de los pasos más destacados para llevar a cabo un proceso de limpieza de datos, pero no son los únicos.

Pentaho, como hemos comentado, está programado en java y por lo tanto, admite la creación de módulos de java específicos para lo que se adapte mejor a las necesidades que tenemos por medio de el paso User Defined Java Class.