UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

"DESARROLLO DE UN SISTEMA INFORMACIONAL INMOBILIARIO PARA ENTIDADES BANCARIAS"

TRABAJO FIN DE GRADO

Septiembre - 2024

AUTOR: Vidal Urbaneja González
DIRECTOR: Jesús Javier Rodríguez Sala

RESUMEN

En este trabajo de fin de grado se abordará el diseño e implementación de un sistema informacional referente a la gestión de inmuebles para una entidad bancaria. Este sistema se basa en la construcción de un almacén de datos o Data Warehouse (DW), junto con el conjunto de procesos y actividades dirigidas a la construcción y gestión de un repositorio para los datos sobre las propiedades inmobiliarias de la entidad. Dichos datos se cargarán en el sistema recopilándolos desde diversos orígenes de datos que proporcionan toda la información relevante. Esta integración de datos desde los mencionados orígenes hacia el repositorio central se realizará mediante la implementación de una batería de procesos ETL (extracción, transformación y carga) diseñados a tal efecto.

La finalidad última del DW implementado es su posterior explotación con la ayuda de técnicas de inteligencia de negocios (BI) que alimentarán herramientas de cuadros de mandos y reportes para apoyar la toma de decisiones analíticas de los usuarios finales del banco.

AGRADECIMIENTOS

Quisiera agradecer a todas las personas y organizaciones que me han apoyado y guiado durante la realización de este trabajo.

En primer lugar a la empresa Teralco por brindarme la oportunidad de realizar este proyecto, sin su confianza, colaboración y recursos depositados en mí, no habría sido posible el desarrollo y finalización del mismo.

A mi tutor Jesuja, por su paciencia, comprensión, esfuerzo y ayuda durante años, que a pesar de las diferentes vicisitudes adversas que me han sucedido, me ha orientado y guiado hasta culminar el trabajo en una etapa complicada por la que estaba pasando mi vida.

A mis padres, que no fallaron en su motivación, amor y apoyo incondicional e inquebrantable a lo largo de mi vida académica.

Finalmente, a mi amiga Lorena, que desde que entré en la carrera siempre ha estado a mi lado hasta el día de hoy, por las innumerables horas pasadas en la biblioteca de Altabix y Tabarca, por la posterior difícil entrada juntos al mundo laboral y por seguir acumulando tiempo en el mismo trabajo, este camino habría sido muy solitario sin ella.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ALGORITMOS

Capítulo 1
Introducción

1.1.- EL MERCADO INMOBILIARIO

El sector inmobiliario es un soporte fundamental en la economía de cualquier país, constituye un motor impulsivo de la misma, además de ser un buen indicativo de salud del mismo. Al ser un sector importante en indicadores de riqueza, puede ser influenciado tanto de manera externa, ya sea debido a la situación económica mundial, o incluso por factores más cercanos como el encarecimiento de los materiales por diferentes causas como guerras o crisis sanitarias, como de manera interna, donde la especulación en los precios de los inmuebles, puede encarecer excesivamente el coste y sumergir a un país a una crisis económica significativa, como sucedió en España en el 2010 en la que se denominó la *"crisis del ladrillo"*[1], cuando los bancos concedían créditos sin ningún tipo de supervisión. Por otro lado, este ámbito abarca varias acciones que se pueden realizar sobre un bien inmobiliario además de la compra-venta, como el alquiler y la gestión de los mismos y una larga lista de actividades, ya que engloba diferentes tipos de bienes, desde

viviendas o suelos, hasta grandes superficies comerciales, creando así un flujo de constante movimiento de dinero y por tanto de información que debe de ser tratada.

Concretamente uno de los aspectos más importantes del sector inmobiliario por parte de la población es la adquisición de una vivienda. Ya sea como bien de primera necesidad que brinda seguridad y bienestar a las personas, como un medio de inversión a largo plazo. Todas estas viviendas, además de otros inmuebles usualmente relacionados tales como trasteros o garajes, son vendidos y comprados todos los días siendo probablemente las transacciones más importantes que haga un individuo en su vida.

Cuando una persona va a comprar o vender un inmueble como por ejemplo una vivienda, lo primero que hace es consultar los diferentes tipos de páginas web o portales dónde se publicitan este tipo de inmuebles. En ellas nos encontramos una gran variedad de inmuebles, categorizados y filtrados por una infinidad de parámetros, tales como número de habitaciones, baños, armarios, etc., hasta la inclusión de mapas donde se puede visualizar la media de los precios por zonas o barrios .

A esto se le une diferentes características y acciones que la persona que está buscando un inmueble puede realizar, nos referimos a solicitar visitas, creación de ofertas, y por supuesto, el contacto con el propietario del inmueble o entidades bancarias a los que se le quiere comprar.

Y es que un gran número de inmuebles pertenecen a la banca ya que es un método más de obtención de capital y cobro de deudas. Estos inmuebles procedentes en su mayoría de ejecuciones hipotecarias o de garantías en caso de impago, representan una apetitosa parte del mercado inmobiliario porque en muchos casos se venden a un precio inferior al que lo vendería un particular, aunque en algunas ocasiones sean inmuebles que puedan estar en buenas condiciones, esto es debido a que se pueden encontrar inmuebles en disputas legales porque contienen ocupas, o activos que la entidad bancaria los vendan en lotes, ya que generalmente desean deshacerse lo antes posible de ellos porque se trata de capital que tienen atado y quieren recuperar parte de las pérdidas de riqueza que haya podido presentar para volver a tener liquidez. En consecuencia estos inmuebles también son puestos a la venta en los diferentes portales inmobiliarios y depositados en agencias inmobiliarias para su venta.

1.2.- JUSTIFICACIÓN DEL PROYECTO

Como se puede intuir, la cantidad de datos que se deben procesar y gestionar en el mercado inmobiliario es muy grande, aún más si pertenecen a una entidad bancaria, porque ya no estamos en el supuesto de que un individuo compra a otro un activo, sino que una entidad,

la cual tiene decenas de miles de inmuebles en venta, puede llegar a vender en un día cientos de inmuebles, y no solo eso sino que a su vez genera más información producto de la propia venta. Por tanto, nos encontramos en la tesitura de tratar un gran volumen de información donde el coste humano de manipular diariamente todos estos datos sería insalvable.

Ante este volumen de información, muchas entidades bancarias optan por externalizar la gestión de sus inmuebles a otra empresa u empresas que se especializan en este sector. Tenemos ejemplos como Solvia para el banco Sabadell, Servihabitat para CaixaBank o Altamira para el Santander[2], son algunos de los casos que podemos encontrar en el mercado inmobiliario. Si bien es una ayuda para la gestión de clientes y comercialización de esos activos, se crea un nuevo problema derivado, correspondiente a tener una empresa externa que se encarga también de la gestión de la base de datos de estos activos. Este problema se puede escalar a diferentes niveles, porque externalizar la gestión de los activos no termina aquí, también se dan casos de varias inmobiliarias enviando información, y no solo una inmobiliaria principal, se puede dar la circunstancia de que otra inmobiliaria gestione solo la cartera de unos determinados activos porque en tema de costes sea más rentable, o por no referirnos a otras empresas muy ligadas a la compra de inmuebles que son las tasadoras, haciendo cada vez más complicado el tratamiento de esta información y desembocando en la necesidad de disponer de un modelo propio de información de esos inmuebles en los sistemas de la entidad, que recoja y permita realizar la trazabilidad de dicha información y una gestión centralizada de la misma.

Como podemos imaginar, para empezar nos encontramos con un primer problema que se traduce en tener diversas entradas de datos y la calidad a nivel del propio dato. Al recibir información de varias fuentes, es necesario introducirla en un mismo sistema gestor de base de datos, ya sea que la información llegue en ficheros planos tipo .csv o .txt o que sea mediante “inserción de datos” [3] en otros gestores de base de datos diferentes a los utilizados en la propia entidad. En cuanto al nivel de dato, la discrepancia entre el formato y consistencia que nos podemos encontrar puede ser muy alta, por eso se necesita una estandarización y normalización [4] que se lleva a cabo mediante un procesado preliminar en unos valores y referencias preestablecidas, o en las que se hayan llegado a un consenso antes. Un ejemplo claro de este problema, podría tratarse de que una inmobiliaria use el valor “*Alq*” para referirse a un inmueble alquilado y la tasadora utilice “*Alquilado*”, a la hora de buscar inmuebles alquilados en estas dos fuentes, es necesario introducir ambos valores, lo que es más costoso, o peor todavía, que no se encuentren la totalidad de inmuebles alquilados porque solamente estemos introduciendo uno de los dos valores.

Una vez se tienen los datos limpios y estandarizados, qué pasa si se requiere tener campos calculados de diferentes orígenes, por ejemplo los gastos que haya podido tener un inmueble de ibi, luz, agua, o cuándo se efectuó la venta. Nos damos cuenta que tener valores importantes de diferentes orígenes concentrados en una misma tabla, nos ahorra

primero el conocimiento de a qué fuentes acudir y cómo relacionarlas para obtener el dato en cuestión. Además de lo ya comentado de tener campos calculados relevantes, también es posible hacer que dicha tabla guarde información histórica, así se puede averiguar los diferentes estados en los que estuvo el inmueble y si hubo cambios en el pasado. Para finalizar, si a este destino, añadimos campos únicos de otras tablas, que si en un principio no son tan relevantes para estar en nuestra tabla principal, pero sí lo suficiente para ciertas casuísticas, podremos acceder a esas otras tablas sin tener que pasar por otras relaciones intermedias que dificultan conseguir dicha información. De esta forma, se consigue una plataforma centralizada y optimizada para el análisis de datos y la generación de informes, que podrá ser enviada a otros departamentos de la entidad bancaria u otras externas como por ejemplo la CIRBE[5] (Central de información de Riesgos del Banco de España). Por supuesto, todos estos datos también son el punto de partida inicial para alimentar cuadros de mandos que muestren toda esta información de una forma más sencilla y fácil de interpretar para los tomadores de decisiones.

Cabe esperar que todos estos datos lleguen por diferentes orígenes todos los días a la entidad, y alguien debería ejecutar los procesos que traten toda la información, entonces ¿hay que tener a una persona ejecutando los cientos de procesos de carga de información que pueda haber todos los días?, qué sucede si se quiere que estos procesos se ejecuten de madrugada porque tardan mucho tiempo en procesar y se necesitan los datos por la mañana, más importante todavía, si empieza a ejecutarse el proceso que carga nuestra tabla centralizada y limpia de datos relevantes para la empresa, pero las tablas que la alimentan no tienen información porque todavía no se han ejecutado, claramente no se cargaría nada. Esto hace pensar que se debería poder ejecutar automáticamente todos los procesos, y diseñar una estrategia de ejecuciones sucesivas en orden y con criterio para cargar de una forma óptima todos los datos.

1.3.- OBJETIVOS

El objetivo que persigue este proyecto es el tratamiento y gestión de activos inmobiliarios para una entidad bancaria, para ello se recurrirá a la creación de un modelo interno de datos inmobiliarios que se resumen a continuación:

- Por un lado, se hará la extracción de datos de diferentes fuentes e integración de los mismos en los sistemas de la entidad bancaria conocidos como "*Staging Area*" [6] o almacenamiento interno, este término se refiere a un área o entorno donde los datos se preparan y procesan antes de ser cargados en su destino final, en nuestro caso serán tablas intermedias que reunirán diferentes características de los inmuebles.

- A continuación se realizarán las diversas transformaciones que sufrirán los datos para su limpieza, validación, normalización, enriquecimiento y además de la creación de nuevos campos calculados a partir de otros.

- Por último, la carga de todos estos datos finales en un "*Data Warehouse"*[7], utilizando métodos de optimización para que el tiempo de procesado sea el mínimo posible.

Es decir, se van a crear diferentes procesos ETL [8] (en español extracción, transformación y carga de datos), mediante el uso de una de las herramientas líderes en integración de datos y gestión de procesos ETL que es Powercenter.

También se diseñará el "*Data Warehouse"* (o DW) que recogerá la información final de las operaciones ETLs, este almacén de datos será un repositorio de datos inmobiliarios centralizado, para integrar y organizar grandes volúmenes de información procedentes de diversas fuentes, lo que proporcionará a su vez un histórico de los datos.

Por último y a partir del DW, se abordará la implementación de diferentes informes como pueden ser inventarios de activos vendidos, además de alimentar diferentes cuadros de mando. Todos estos procesos serán automatizados y ejecutados diariamente, además de que implementarán las respectivas validaciones para comprobar que todo está funcionando correctamente.

En conclusión, se pretende crear un modelo de base de datos para el tratamiento y gestión de activos inmobiliarios, mostrando como los datos discurren por las diferentes etapas por las que viajan, desde que son enviados por la agencia inmobiliaria, pasando por diferentes transformaciones y validaciones hasta que finalmente son consumidos por el usuario final de forma depurada y clara.

1.4.- LÍMITES DEL PROYECTO

Queda fuera de alcance de este proyecto los siguientes puntos:

- En lo referente a la parte de "*back end*" [9], la fase de modelado de tablas y la planificación automática de procesos ETL e integración de datos, hasta hace poco la etapa de modelado también se realizaba y estaba incluida para hacerse por el mismo equipo de inmuebles a la hora de crear nuevas entidades o tablas que aumentaran el modelo interno, pero esta parte se decidió externalizar a otro equipo solo dedicado a este tema, por lo tanto se abordará el diseño y normativa de las tablas sin entrar a cómo funciona el software modelador de bases de datos utilizado,

de igual forma pasa con la planificación de procesos, se mostrará cómo se diseñan, configuran y ejecutan los diferentes elementos que se pueden automatizar, sin entrar en la creación de los mismos.

- Respecto a toda la parte que implica la visualización de los datos, tales como los cuadros de mandos o tableros de control tampoco se profundizará en su creación. Este proyecto comprende únicamente los entresijos del "*back end*" desde la llegada del dato hasta su puesta a punto para ser visualizado, pero sin mostrar el desarrollo relativo al *"front end"* [9], aunque sí que se mostrará cómo es la visualización por la parte de negocio que corresponde al cliente final, dando a conocer como se suele presentar la información de manera visual utilizando tablas, mapas y otros elementos gráficos para resumir de manera fácil y comprensible el arduo trabajo realizado en la parte del "*back end*" y poder evaluar la evolución del datos en su última etapa.

Capítulo 2 Antecedentes y estado de la cuestión

2.1.- SITUACIÓN ACTUAL DE LA EMPRESA

Como se ha comentado con anterioridad, en una empresa se crea un volumen de datos constante que guardan sus transacciones y acciones capturados de distintas formas. Estos datos en el peor de los casos son guardados en extensas hojas de cálculo, con laboriosas fórmulas, y deben ser introducidos de manera manual, dejando la información inconexa, además de incrementar el trabajo que debe realizar la persona encargada de tratar esta información. Otro método más común, es que las firmas los administren construyendo un conjunto de sistemas transaccionales[11] que recolectan y almacenan los datos generados de las distintas operaciones de la entidad, para luego entrar en consonancia con bases de datos relacionales[12] donde se cargan (Figura 2.1).

Si esta es una forma más beneficiosas de tratar los datos, en muchas ocasiones descubrimos que con el transcurso de los años, los datos se van acumulando de una forma

desestructurada y desordenada, sin darles ningún uso o propósito, donde solo se generan gigabytes de datos, agrandando información inútil en términos de memoria en la base de datos de la empresa. Sumado a encontrarnos sistemas mixtos, en los que la empresa tiene montadas bases de datos relacionales para ciertas casuísticas y un sin fin de hojas de cálculo para otras, debido a que el usuario final se siente más seguro utilizando estas últimas, aunque sea un método más laborioso.

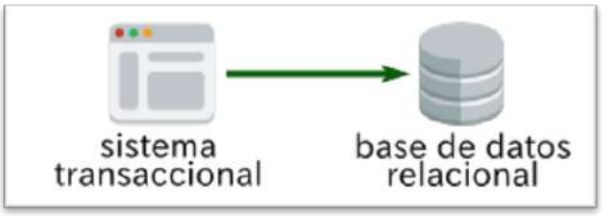

Figura 2.1 Sistema transaccional a base de datos relacional

Esta última era la situación en la que se encontraba la entidad bancaria en relación a los datos de inmuebles antes de haberse construido el modelo, se había creado un sistema mixto entorno al gestor de bases de datos "Microsoft Access" donde se guardaba la información de los inmuebles en un gran número de tablas, y el programa de hojas de cálculo "Microsoft Excel" con el que algunos empleados, que a lo mejor pertenecía a otras partes del negocio y no tenían tantos conocimientos en la creación de consultas avanzadas de SQL (lenguaje de consulta estructurada)[13], consultaban de forma simple la base de datos Access, para culminar añadiendo unas últimas pinceladas de cálculos finales y transformaciones mediante el uso del Excel a los datos obtenidos del Access.

En esta tesitura aparece el concepto del Data Warehouse (Figura 2.2) que ayuda a solventar este problema (se ejemplifica y detalla de forma más visual en el capítulo 3 usando el modelo de datos de inmuebles), se podría definir de forma sencilla, como un almacenamiento de datos orientado al negocio, no volátil, variante en el tiempo, integrada y que da un soporte en la toma de decisiones, método que es mundialmente atribuido a William H. Inmon[14].

Figura 2.2 Data Warehouse

Bill Inmon enfocó su diseño para construir un gran almacén de datos que contuviera todos los datos relevantes de diferentes fuentes de la parte específica del negocio que se estaba tratando, para después darles algún sentido o uso y entenderlos mejor, en nuestro caso consistiría en reunir todos los datos relacionados con inmuebles de los diferente proveedores de información y guardarlos en un mismo repositorio bajo ciertas características. Otro ejemplo claro en una entidad bancaria sería el Data Warehouse de "*Deuda*", que concentraría todos los datos relacionados a esa temática.

2.2.- HERRAMIENTAS DISPONIBLES EN EL MERCADO

Pero ¿Cómo se construye el Data Warehouse? ¿Cómo se carga el modelo interno de inmuebles de la empresa?, para construirlo y cargarlo se debe realizar una acción de integración, de todos los datos de una forma directa y clara en los sistemas de la empresa, este acto se lleva a cabo mediante procesos ETL (en el capítulo 3 se profundizará en ello) donde a su vez estos desarrollos son creados a partir de herramientas ETL.

Una herramienta ETL es un software especializado para la creación de procesos que permiten la gestión y tratamiento de grandes cantidades de datos. Estos programas posibilitan configurar los procesos para ordenar y elegir de qué fuentes va a "beber" nuestra base de datos, por ejemplo: otras bases de datos relacionales, archivos planos, servicios web, sistemas CRM[15], entre otros. Para su posterior transformación y carga en el Data Warehouse.

Igualmente estas herramientas facilitan la optimización de los procesos ETL, demostrando un gran rendimiento para el procesamiento de datos, ya que en su mayoría se puede configurar para que corran en paralelo, permitiendo una gran eficiencia en los tiempos de ejecución, así mismo, también dan un alto grado de escalabilidad, pudiendo modificar los procesos para añadir nuevas funcionalidades una vez creados o consiguiendo desarrollos modulares, donde diferentes programadores pueden trabajar para añadir nuevas funcionales en vista de alcanzar un objetivo en común en el menor tiempo posible.

Entre sus principales beneficios se pueden encontrar herramientas muy visuales, es decir, de un simple vistazo es posible hacerse una idea aproximada del funcionamiento de un proceso ETL, no es necesario que el programador se tenga que sumergir en un mar de líneas de código para discernir su funcionamiento, facilitando la creación y sobre todo el mantenimiento de los flujos de datos a otros programadores que, en caso de fallo del propio proceso, puedan solventarlo de una manera rápida y sencilla, sin llegar a tener que parar otros procesos que dependan de él.

Por último y sin entrar en detalles de todo el aprovechamiento que nos pueden ofrecer estas herramientas, cabe destacar la ayuda en el apartado de programación y planificación a la hora de correr estos procesos, ya que permiten ejecutarlos en horarios específicos y en función de eventos, a la par de no tener a una persona dedicada en exclusiva a darle a un simple botón de ejecutar, una ejemplificación de esta característica podría ser un proceso que se ejecuta a las ocho de la mañana después de que haya terminado la ejecución de otra ETL que cargue las fuentes que va a leer, esto facilita la automatización de los flujos de trabajo en horas de bajo trabajo que se traduce en menos picos de carga computacional, además de tener los datos ya procesados en determinadas horas, en particular en el horario laboral de los empleados, para así facilitar que se puedan consumir por otras aplicaciones o usuarios de una manera efectiva.

Dado que el modelo de inmuebles de la entidad está construido a base de un gran número de procesos ETLs y es por tanto tema principal de este trabajo, es imperativo analizar algunas de las herramientas disponibles que existen en el mercado.

2.2.1.- Informatica - Powercenter[16]

Powercenter es la herramienta líder en el mercado de integración de datos empresariales con una tasa alta de fidelización e implementación entre sus usuarios, además de ser la aplicación con la que se ha desarrollado este trabajo (Figura 2.3). Algunas de sus características[17] son:

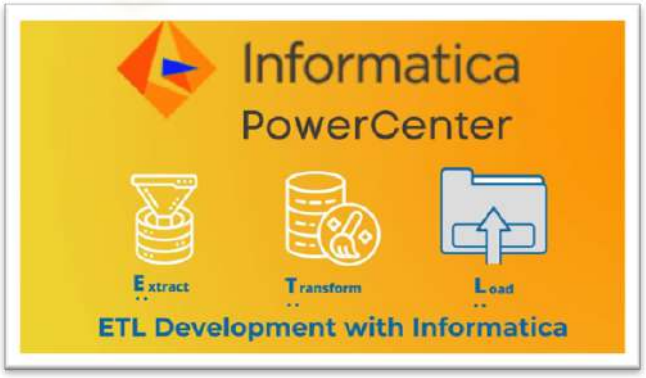

Figura 2.3 Herramienta ETL Powercenter

- Interfaz intuitiva que permite al usuario arrastrar y añadir componentes para después configurarlos y conectarlos entre sí (Figura 2.4).

- Su escalabilidad le permite manejar grandes volúmenes de datos y flujos de trabajo, aportando un rendimiento óptimo en el procesamiento de datos gracias a técnicas avanzadas de ejecución en paralelo.

- Cuenta con una amplia conectividad que permite la integración de datos desde una variedad de fuentes, añadido a las diferentes transformaciones, seguridad en la entrada a sus procesos y un sistema de control de fallos, aumenta la productividad a la hora de desarrollar y disminuye los riesgos en la integración.
- El soporte y la comunidad es activa, ayudando tanto en el aprendizaje como en la documentación y asistencias a los usuarios.

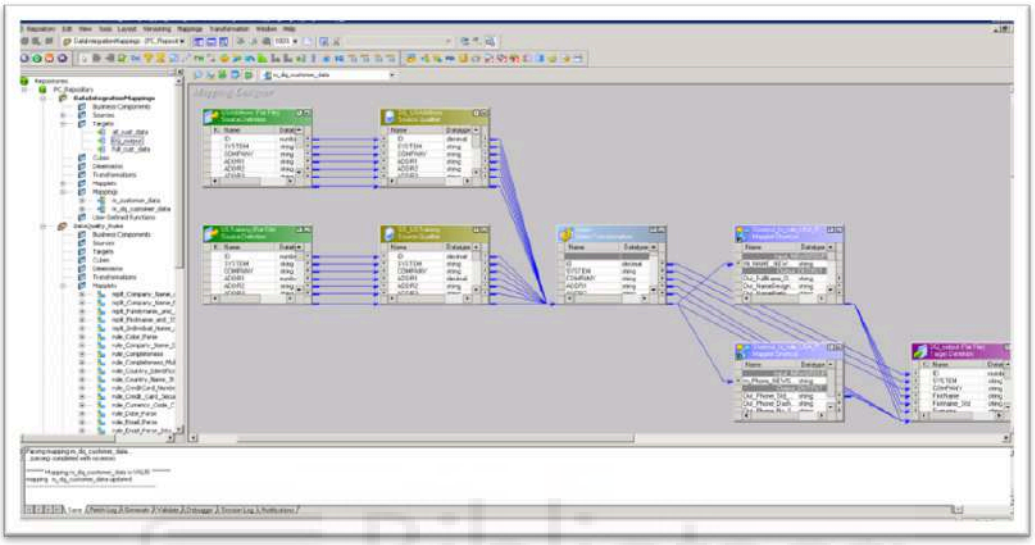

Figura 2.4 Interfaz gráfico Powercenter

2.2.2.- Talend - Open Studio[17]

Lanzada en el 2006, es una versión de código abierto de la suite de productos de Talend (Figura 2.5) que proporciona herramientas para la integración de datos, algunas características[18] y funcionalidades incluyen:

Figura 2.5 Talend - Open Studio

- Como Powercenter, su interfaz gráfica es intuitiva y visual, “tipo” arrastrar y soltar (Figura 2.6), facilita la creación de procesos sin necesidad de escribir código.

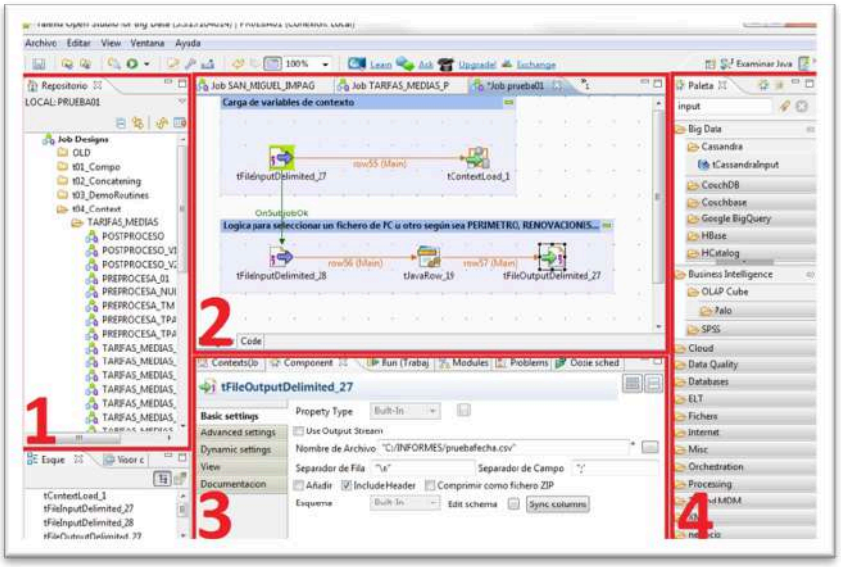

Figura 2.6 Interfaz gráfica Talend Studio

- Conectividad con diferentes orígenes como base de datos, archivos planos, aplicaciones en la nube, sistemas ERP, entre otros.

- Transformaciones y depuraciones que permiten generar y escribir código Java y consultas SQL, además de operaciones de limpieza de datos gracias a un repertorio de componentes pre-construidos, listos para facilitar las necesidades del usuario.

- Su administrador de metadatos permite crear plantillas (metadatos) sobre la estructura, el tipo y el formato de los datos que se utilizan en sus procesos ETL, que facilitan la reutilización, mantenibilidad y portabilidad de un proceso de integración a otros, no teniendo que volver a crearlos para posteriores procesos. (Figura 2.7).

Figura 2.7 Configuración metadatos Talend Studio

2.2.3.- IBM - InfoSphere DataStage[20]

InfoSphere DataStage es una herramienta ETL creada por la empresa IBM para la integración de datos en entornos empresariales, entre sus características nos encontramos:

- Como las herramientas ETL anteriores, contiene una interfaz gráfica para el diseño de flujos de trabajo (Figura 2.8), amplia conectividad con diferentes orígenes de datos, transformaciones dedicadas a limpiar, normalizar, enriquecer, sumado a una escalabilidad y depuración de los datos.

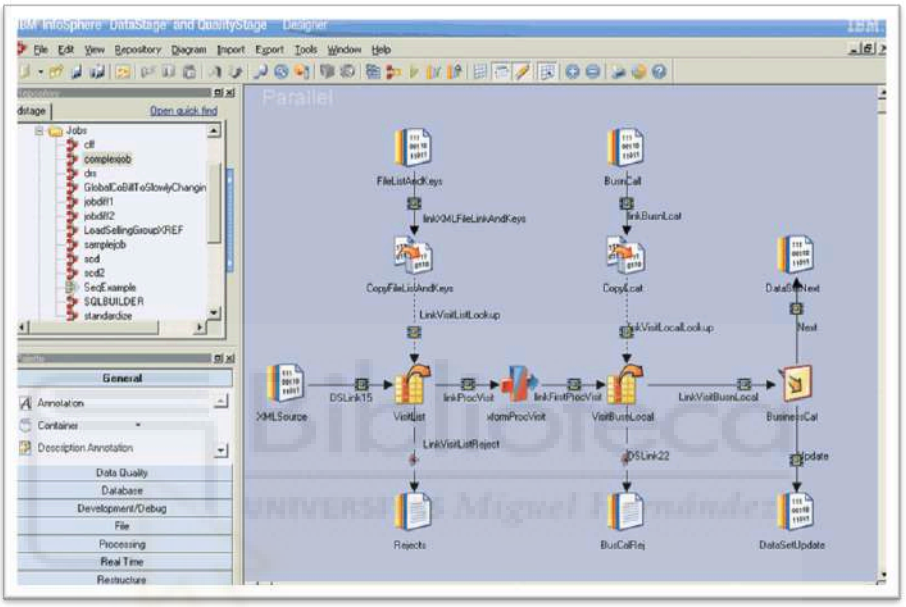

Figura 2.8 Interfaz gráfica de IBM - InfoSphere DataStage

- DataStage pertenece a la suite de productos de InfoSphere dedicada a la creación de ETLs, en esta suite también encontramos[21]:

    - InfoSphere QualityStage: Destinada a solventar la calidad de los datos, identificando y haciendo una limpieza de datos incorrectos como duplicados, o datos inconsistentes como corrección de formatos, valores no válidos, e incluso errores ortográficos.
    - InfoSphere Information Analyzer: Solución que ayuda a evaluar patrones, tendencias o problemas almacenados en los datos integrados.
    - InfoSphere Information Governance Catalog: Finalmente Governance Catalog es como una gran biblioteca digital donde los diferentes datos sobre diferentes temáticas de una empresa son “etiquetados” para ser encontrados rápidamente qué significan y cómo están relacionados entre sí, es decir, organiza, protege y ayuda entender todos los datos de la empresa de manera eficiente.

Figura 2.9 Características Suite - InfoSphere

Esta característica implica un entorno familiar para los desarrolladores, si se contrata toda la suite al completo de IBM.

2.2.4.- Microsoft - SQL Server Integration Services (SSIS)[22]

SSIS es la herramienta de Microsoft para integrar y cargar datos de manera automatizada entre diferentes sistemas (Figura 2.10). Algunas de sus características son:

Figura 2.10 Microsoft - SQL Server Integration Services

- Dispone de un diseñador gráfico para la creación de flujos de trabajo, transformaciones, automatización, monitorización y depuración.

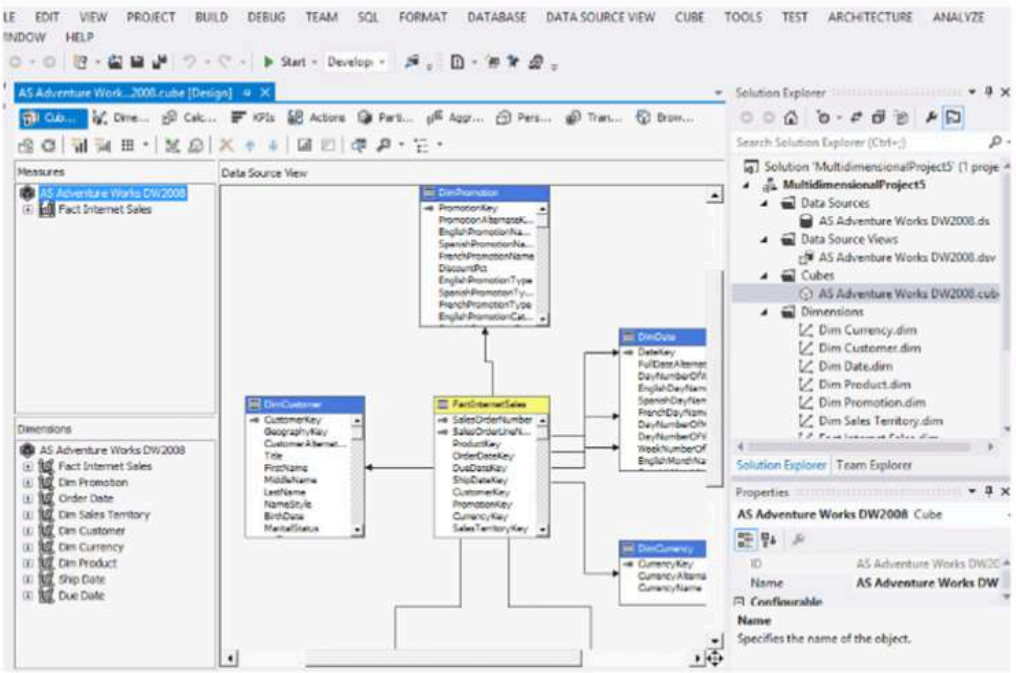

Figura 2.11 Interfaz gráfica de SSIS

- Es programable y extensible, permite personalizar y ampliar sus flujos mediante el uso de scripts de código en diversos lenguajes como C o Visual Basic .NET.

- Especial mención a la conectividad flexible a fuentes, ya no solo las habituales, como ficheros planos, base de datos relaciones, etc sino a servicios web o aplicaciones empresariales, facilitando la lectura de datos de diferentes entornos.

- Se puede integrar con otros ecosistemas de Microsoft como SQLServer, Azure o Visual Studio.[23]

2.2.5.- Apache - NiFi[24]

Apache Nifi es una herramienta de código abierto dedicada a la extracción, transformación y carga de datos (ETL). Desarrollada por la empresa Apache Software Foundation, permite automatizar de forma visual y eficiente la carga de grandes cantidades de datos[25]. Entre sus características destacan:

- Como los anteriores, incorpora una interfaz gráfica (Figura 2.12) con ejecuciones en tiempo real muy técnicas (Figura 2.13), transformaciones programables en Java, alta modularidad, incorpora una auditoría del dato para cumplir regulaciones.

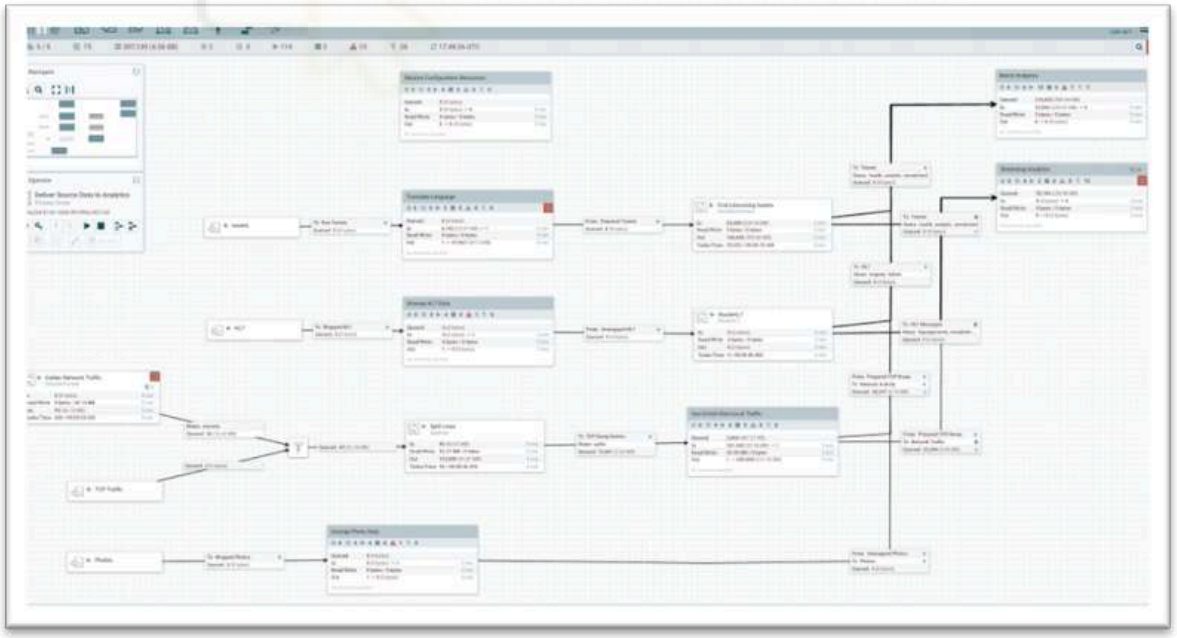

Figura 2.12 Interfaz gráfica Apache - NiFi

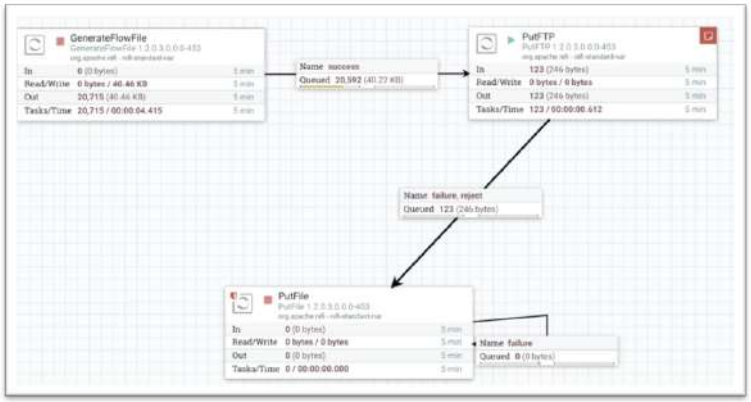

Figura 2.13 Ejecución en tiempo real Apache - NiFi

- Es posible programar nuevos elementos y componentes programados con la API de Java ya que está basada en esta tecnología.

- Integrado con Cloudera Data Platform (CDP) para administración y análisis de datos, y Cloudera Flow Management (CDF), plataforma destinada a la ingesta y procesamiento de los mismos.

2.2.6.- Hitachi - Pentaho Data Integration (Kettle)[26]

Pentaho, también conocido por Kettle, es un software de código abierto para la integración de datos mediante técnicas de ETL y basado en metadatos, entre sus características encontramos:

- Ofrece una interfaz gráfica intuitiva (Figura 2.14), muy visual y fácil de usar para el diseño de flujos sin necesidad de escribir código, pero con la posibilidad de escribir consultas SQL para componentes personalizados.

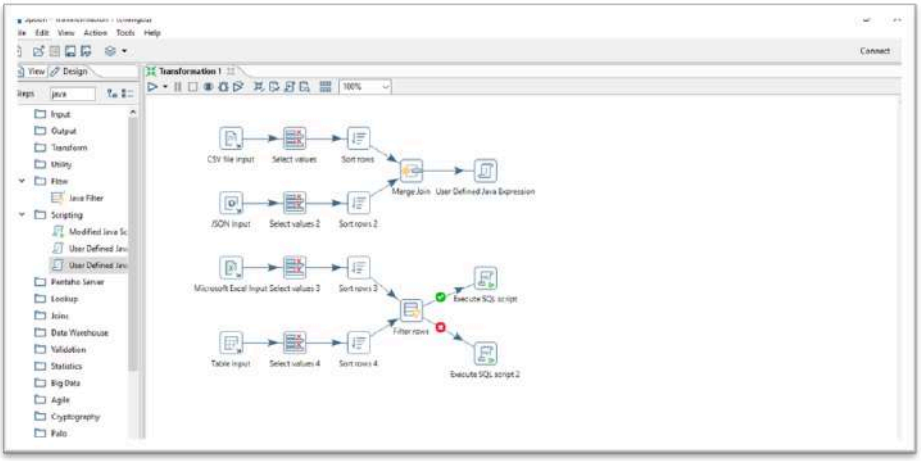

Figura 2.14 Interfaz Pentaho

- Cuenta con características comunes a otras herramientas ETL, como la integración de datos desde múltiples orígenes, la programación de ejecución de tareas, seguridad controlando el acceso a la información, entre otros.

- Multiplataforma ya que se puede usar en sistemas de Microsoft, Mac o Linux [27]

2.2.7.- CloverDX [28]

Clover es una plataforma basada en Java que permite la migración, preparación y automatización de datos mediante el uso de ETL (extracción, transformación y carga) de manera eficiente y escalable en entornos empresariales[29] (figura 2.15).

Figura 2.15 CloverDX

Entre sus cualidades nos encontramos:

- Como es habitual en estas herramientas, tiene una interfaz gráfica (Figura 2.16) que permite diferentes tipos de transformaciones, con lectura de diferentes orígenes a diferentes destinos.

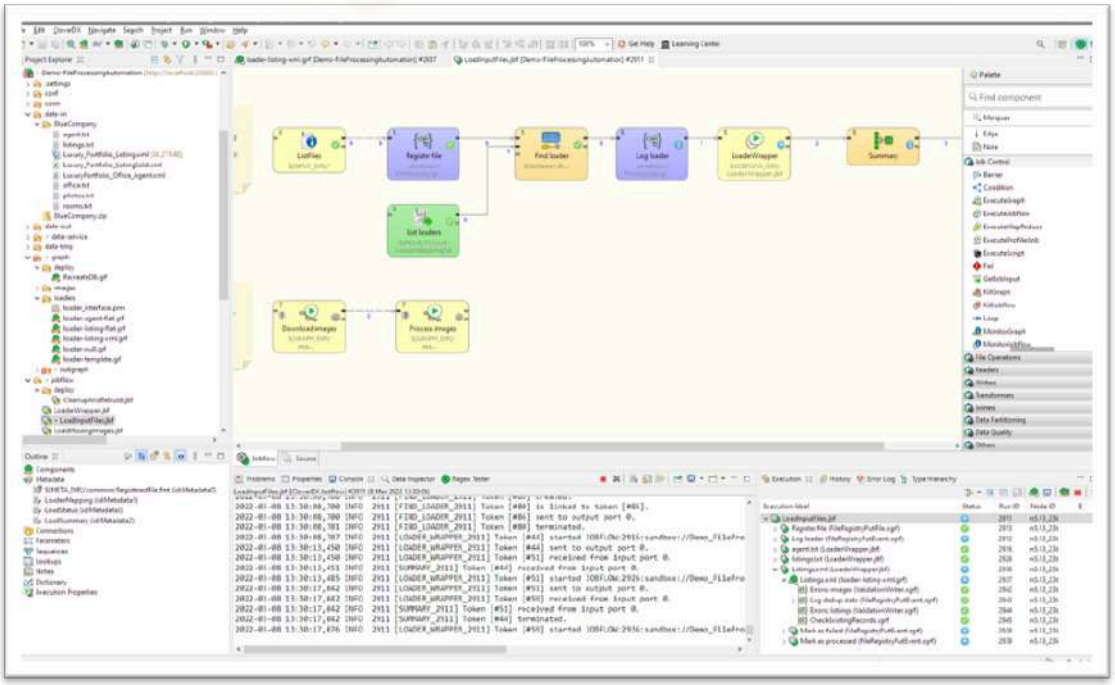

Figura 2.16 Interfaz gráfica CloverDX

- Utiliza tecnologías y estándares de la industria para la integración de datos como XML y SQL entre otras.

- Automatización del movimiento de los datos entre diferentes bases de datos.

2.2.8.- Oracle - Data Integrator (ODI)[30]

ODI es una herramienta de integración de datos que permite diseñar, ejecutar y supervisar flujos de trabajo [31], entre sus cualidades:

Figura 2.17 Oracle Data integrator

- Están las habituales, como admitir una amplia gama de fuentes de datos, tales como bases de datos, archivos planos, aplicaciones web, servicios en la nube entre otros.

- La interfaz de usuario es gráfica (Figura 2.18) ayudando a definir la lógica y las reglas de integración de los datos para el programador.

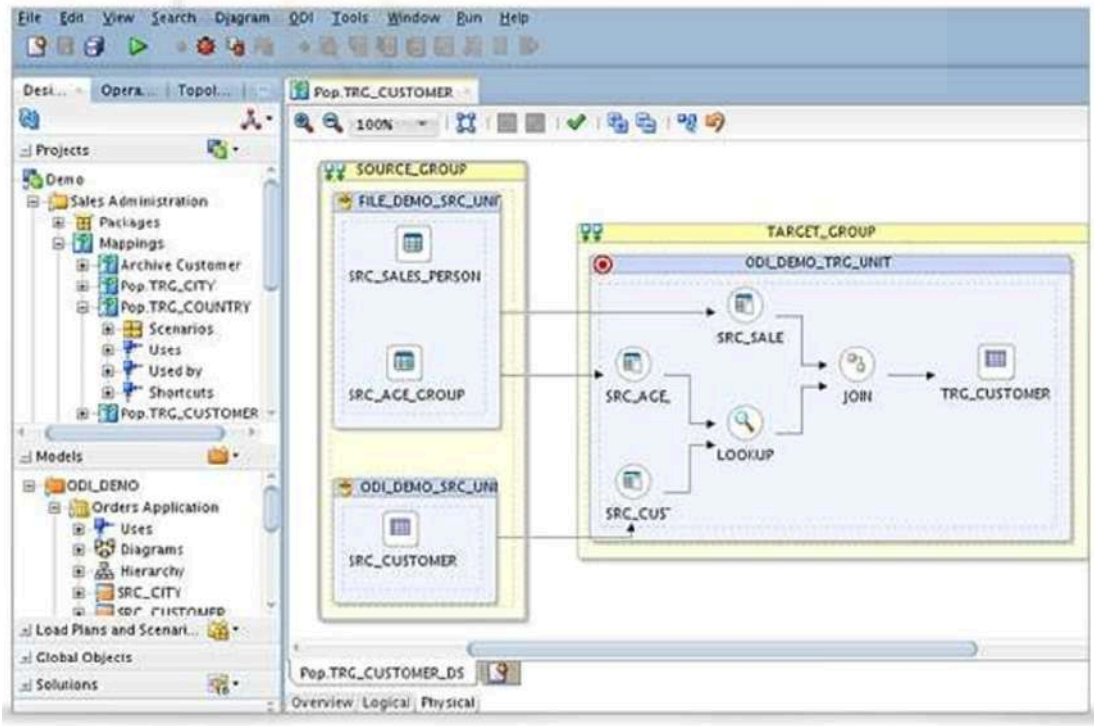

Figura 2.18 Interfaz gráfica Oracle Data Integrator

- Cuenta con una idea llamada “módulos de conocimiento”, que son elementos reutilizables usados para cargar, transformar o validar datos en los diferentes procesos.

- Dispone de una arquitectura declarativa donde los programadores no tienen que definir los detalles de implementación como el orden de ejecución o los métodos de transformación de datos, sino que marcan las tareas de integración mediante la especificación de los objetivos y las relaciones entre los datos, ya que ODI interpreta estas especificaciones, genera automáticamente el código y los flujos para después ejecutarlos de una forma optimizada.

- Se integra con otras aplicaciones de Oracle como por ejemplo Oracle Fusion Middleware.

2.2.9.- Resumen

Como se describe al comienzo de este apartado, todas las herramientas ETL presentadas tienen una serie de características en común:

- Interfaz gráfica: Todas cuentan con una interfaz gráfica para el desarrollo, que ayuda en la curva de aprendizaje y mejora la facilidad de uso, de tal manera que, de un vistazo rápido, se puede intuir el funcionamiento de un proceso ETL.
- Transformaciones: Las herramientas facilitan un gran repertorio de elementos (en muchos casos personalizables) para limpiar, normalizar o directamente operar con los datos.
- Planificación: Todas las herramientas ETL anteriormente mencionadas permiten definir horarios de ejecución, gestionar dependencias, controlar la carga de trabajo o programar tareas.
- Conexiones con distintos orígenes: En mayor o menor medida, dependiendo la herramienta y el entorno de trabajo, ofrecen la lectura de diferentes tipos de orígenes de datos.
- Depuración de errores: A la hora de solventar procesos fallidos es necesaria una interfaz que ayude a identificar posibles errores. Algunas de estas herramientas destacan por ser más amigables con el programador, pero en definitiva todas tienen algún tipo de característica que ayuda a identificar y corregir dichos errores.
- Escalables y modulares: En definitiva hay herramientas que admiten más escalabilidad y modularidad para grandes integraciones, pero hasta la herramienta ETL más sencilla permite cierto grado escalabilidad, ayudando a manejar eficientemente grandes volúmenes de datos, o en modularidad, en la creación de componentes independientes y reutilizables que faciliten el diseño y mantenimiento de los desarrollos.

Pero al igual que con los lenguajes de programación, donde encontramos una gran variedad en el mercado, dependiendo del estado de los datos, las tecnologías con que se montan las bases de datos y su finalidad, es posible que convenga implementar unas u otras en nuestros sistemas. Para ello se presenta a continuación, una tabla a modo de revisión para hacer una comparativa rápida de las herramientas analizadas de sus puntos destacables:

Tabla 2.1: Resumen de aspectos clave de las herramientas ETL analizadas

Herramienta ETL	Principales Ventajas	Principales Inconvenientes
Informatica - Powercenter	- Depuración: Errores son más sencillos de solventar puesto que cuenta con un sistema que describe los fallos. - Escalabilidad: Es capaz de manejar grandes volúmenes de datos. - Conectividad: Da la posibilidad de integrar datos a partir de una gran variedad de fuentes. - Seguridad: Ofrece diferentes métodos de control de acceso a los procesos para proteger los datos sensibles.	- Licencias: con alto costo y dependiendo del usuario (número de usuarios, volumen de datos, servidores). - Aprendizaje: Si bien tiene una interfaz amigable, es necesario un alto conocimiento de consultas SQL. -Proveedor: El usuario queda atado a actualizaciones y cambios por parte de la empresa Informática.
Talend - Open Studio	- Herramienta ETL “Open Source”. - Tiene como punto de partida Eclipse lo que admite desarrollar para aplicaciones dedicadas a Java, aportando un entorno familiar para usuarios que estén habituados a este lenguaje. - Gestión de metadatos, que permite ser reusados para futuros procesos.	- Aunque la herramienta es gratuita, existen funciones sólo disponibles en la versión de pago. - Rendimiento bajo en grandes volúmenes de datos, si bien es capaz de manejarlos experimenta ciertas limitaciones respecto a otras herramientas de la competencia. - Al ser gratuita, el soporte y actualizaciones depende en gran medida de la comunidad activa de usuarios.
IBM - InfoSphere DataStage	- Especial importancia a la integración en el ecosistema IBM de las otras soluciones InfoSphere existentes: QualityStage, Analyzer y Governance. dentro de un mismo sistema abarca	- Coste de licencias. - Al pertenecer a suite de otras aplicaciones, puede presentar problemas de compatibilidad con sistemas de terceros.
Microsoft - SQL Server Integration Services	- Uso de script para modificar o crear flujos de integración de datos en lenguajes de programación. - Al ser de Microsoft se integra con otras tecnologías de la compañía como SQLServer, Azure y Visual Studio.	- Al pertenecer a Microsoft no está disponible para otras plataformas. - Está diseñado para ser usado con otras tecnologías de Microsoft, como SQLServer y Windows Server
Apache - NiFi	- Herramienta de código abierto basada en Java, acumulando las ventajas que ello conlleva. - Amplia configuración, desde los propios elementos de la herramienta hasta la modificación en tiempo de ejecución de la configuración del proceso. - Aporta facilidades para auditorías.	- Las versiones por debajo de 1.x no cuentan con soporte técnico. - Consumo de recursos computacionales elevados en función de la carga de procesamiento.

Hitachi - Pentaho Data Integration	- Usa tecnologías comunes (Java, XML o JavaScript) además de ser de código abierto y multiplataforma. - Tiene una instalación y configuración sencilla. - Permite escribir consultas en Javascript.	- Rendimiento bajo en casos de grandes volúmenes de datos. - Limitaciones en la disponibilidad de características avanzadas para la versión gratuita. - Documentación limitada aunque cuenta con un gran comunidad.
CloverDX	- Puede funcionar como librería Java lo que permite instalarse en cualquier sistema operativo que utilice esta tecnología. - Los elementos están guardados con formato XML, tanto los componentes como los propios metadatos de la aplicación proporcionando portabilidad y compatibilidad con una amplia gama de sistemas operativos y plataformas.	- Al estar basada en Java es posible que requiera de puentes en compatibilidad para entornos que no usen esta tecnología. -Documentación más escasa respecto a herramientas más utilizadas.
Oracle - Data Integrator (ODI)	- Diseñado especialmente para integrarse de manera nativa con las bases de datos de Oracle. -Dispone de elementos configurables y reutilizables. -Arquitectura de flujo de datos declarativa. -Integración con otras aplicaciones de la compañía como Oracle Fusion Middleware.	- Coste de licencias que para pequeñas organizaciones con presupuestos limitados pueden presentar un problema. - Flexibilidad limitada en entornos que no corren con Oracle, al ser una herramienta diseñada para integrar datos en entornos de Oracle se pueden producir limitaciones en términos de compatibilidad en otros entornos.

2.3.- CASO DE USO: EJEMPLO DE MALAS Y BUENAS PRÁCTICAS

Por último, en este apartado vamos a valorar la importancia de las herramientas ETL, mostrando un ejemplo real, que si bien no forma parte del Data Warehouse de inmuebles, sí que está estrechamente relacionado puesto que bebe de él y es una praxis aún utilizada en la entidad que, a día de hoy, se sigue subsanando, en las propias palabras de un alto cargo del banco, se va tratar de quitar y automatizar un "*chiringuito*"; en el caso que nos atañe, un "*chiringuito*" es un artificio o práctica que algunos empleados crean para obtener cierta información, pero que no sigue ningún tipo de directriz, método, integración, optimización, automatización, etc., es decir, se trata de un tipo de consulta que, si bien es cierto que proporciona el resultado deseado, se hace aplicando una metodología inadecuada que no cumple con ciertos criterios de eficiencia a la hora de gestionar los datos. A continuación se describe en qué consiste esta práctica del "*chiringuito*". El usuario envía una consulta SQL en un documento de Word de siete páginas, que ha construido en Access (ver Figura 2.19). Esta consulta genera un informe de las tasaciones de los activos inmobiliarios que están "vivos", es decir sin vender, la ejecuta un número indeterminado de veces al mes, por lo que no sigue ningún tipo de patrón a la hora de poder planificarla.

Figura 2.19 Consulta SQL Access información de tasaciones.

Una vez revisado y comprobado que los resultados proporcionados al ejecutar la consulta son iguales, a los campos de una de nuestras tablas del DW, procedemos a usar la herramienta Powercenter para crear una ETL que emule la consulta SQL proporcionada (Figura 2.20).

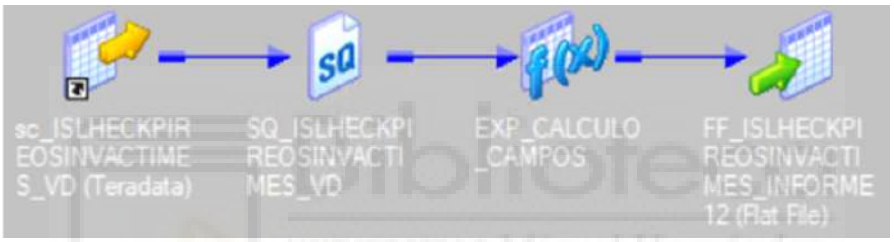

Figura 2.20 ETL tasaciones creada con Powercenter.

Sin tener ninguna noción sobre la herramienta Powercenter, ni en construcción de ETLs, ni en lenguaje SQL, resumir siete páginas de consulta SQL, en cuatro elementos (dos de lectura, uno de cálculos y el último el propio informe en sí), efectivamente da a entender que existía una carencia acusada en el tratamiento de los datos, pero ¿cómo ha sido posible?, para entender mejor y valorar lo ocurrido se va explicar de manera sencilla el flujo de los datos en ambos métodos. El usuario hasta ahora consultaba mediante Access las bases de datos de inmuebles en sus orígenes y los copiaba a un excel (Figura 2.21).

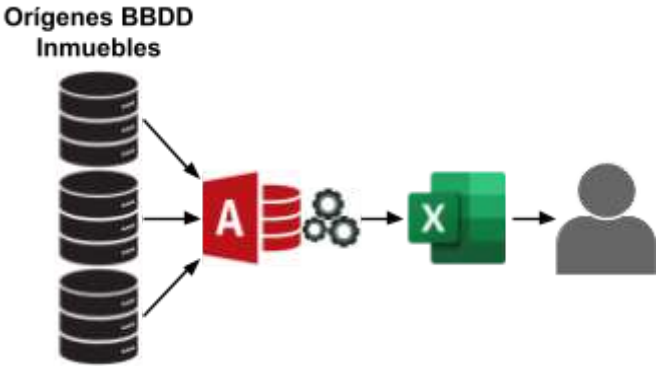

Figura 2.21 Método que ejemplifica el procedimiento que seguía el usuario.

Para obtener los datos de los inmuebles era necesario acudir a un gran número de tablas, lo que generaba nuevos problemas, por ejemplo, si se necesita la información respecto a las tasaciones directas (existen diferentes tipos de tasaciones) de un inmueble, pero en la tabla de información general de inmuebles (A) no se dispone de esa información y se tiene que consultar otra tabla distinta que sí que la tiene (C), pero que casualmente no dispone de las claves (campos comunes entre tablas para poder enlazar las dos tablas entre sí), se debe encontrar una tercera tabla (B) la cual que sí tenga las claves necesarias, para hacer de conexión entre la que contiene la información de los inmuebles (A), con la tabla que tiene las tasaciones directas de los inmuebles (C) (Figura 2.22).

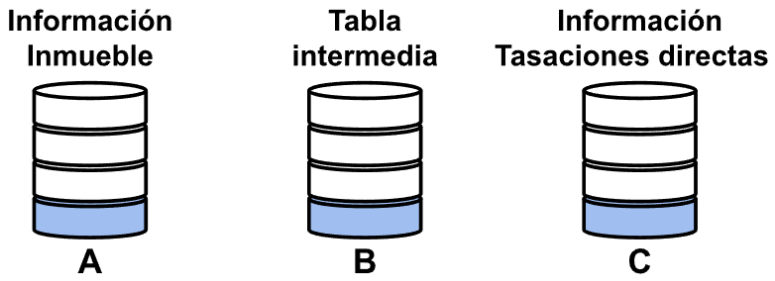

Figura 2.22 Puente entre tablas

De esta manera es como el usuario solo para poder obtener la información de diferentes tablas orígenes, necesitaba líneas y líneas de código SQL para enlazar las tablas entre sí.

```
FROM        MAESTRO.INV_ACTIVOS_CONT LEFT OUTER JOIN
                    (SELECT      TOP (100) PERCENT PRINEX.FUNHIP.UHNREG AS IdUnico,
TAS1.VTTASACI AS IdTasacion, TAS1.VTTIPOTASA AS TipoTasacion, TAS1.VTVALTAS AS
ValorTasacionDirecta,
                                TAS_AUX1.TEFECHA AS FechaTasacionDirecta,
TAS_AUX1.TECODEMPTA AS Codigtasadora, TAS_AUX1.TEEMPTAS AS [nom Tasadora],
TAS_AUX1.TECIFTASA AS [Cif Tasadora],
                                TAS_AUX1.TEEXPEDI AS Expediente, TAS_AUX1.TETASADOR AS
Tasador, PRINEX.FEMPTASA.ETNOM AS Tasadora, PRINEX.FEMPTASA.ETCIF AS CifTasadora
                    FROM        PRINEX.FUNHIP LEFT OUTER JOIN
                                PRINEX.FF_VALTASA AS TAS1 ON TAS1.VTIDOBJETO =
PRINEX.FUNHIP.UHNREG AND TAS1.VTOBJETO = 'INM' AND TAS1.VTTIPOTASA IN ('DIRECTA',
'ACTUAL', 'BDE', 'COMERCIAL', 'COMPRA', 'COMPRA DIR',
                                'PERMUTA', 'COSTE') LEFT OUTER JOIN
                                PRINEX.FF_TASACI AS TAS_AUX1 ON TAS1.VTTASACI =
TAS_AUX1.TETASACI LEFT OUTER JOIN
                                PRINEX.FEMPTASA ON PRINEX.FEMPTASA.ETCOD =
TAS_AUX1.TECODEMPTA) AS TasacionDirecta ON MAESTRO.INV_ACTIVOS_CONT.IdUnico =
TasacionDirecta.IdUnico LEFT OUTER JOIN
                    (SELECT      FUNHIP_1.UHNREG AS IdUnico, TAS1.VTTASACI AS IdTasacion,
TAS1.VTTIPOTASA AS TipoTasacion, TAS1.VTVALTAS AS ValorTasacionEstadistica,
TAS_AUX1.TEFECHA AS FechaTasacionEstadistica,
                                TAS_AUX1.TECODEMPTA AS Codigtasadora,
TAS_AUX1.TEEMPTAS AS [Nom Tasadora], TAS_AUX1.TECIFTASA AS [Cif Tasadora],
TAS_AUX1.TEEXPEDI AS Expediente, TAS_AUX1.TETASADOR AS Tasador,
                                FEMPTASA_1.ETNOM AS Tasadora, FEMPTASA_1.ETCIF AS
CifTasadora
                    FROM        PRINEX.FUNHIP AS FUNHIP_1 LEFT OUTER JOIN
```

Figura 2.23 Sentencia "LEFT OUTER JOIN" usada para enlazar tablas entre sí

La Figura 2.23 (captura de una consulta en un documento de Word) muestra la evidencia, es decir un fragmento de la consulta utilizada en el access, en la que se aprecian hasta siete "LEFT OUTER JOIN" que son las sentencias que se usan en SQL para enlazar tablas, todo ello sin haber llegado a realizar ni un simple cálculo, además sin pensar en que se deben tener los conocimientos necesarios para enlazarlas, o el tiempo necesario dedicado si se tuviera que realizar un nuevo informe similar. Una vez ejecutada la consulta en el Access ya se obtendría el informe para ser consumido por el propio usuario o otros compañeros del departamento (Figura 2.24):

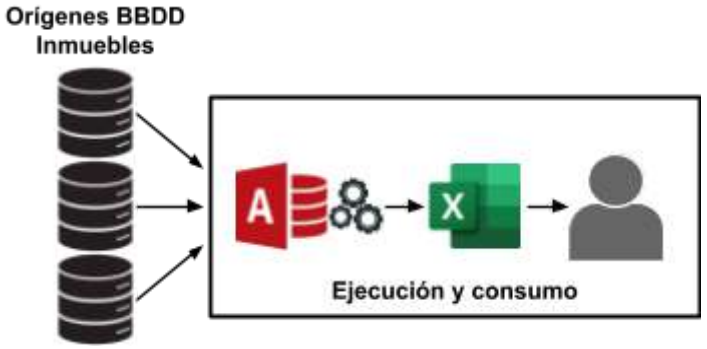

Figura 2.24 Informe final del procedimiento antiguo.

Esta parte que parece sencilla, se complica aún más si a la hora de ejecutar la consulta, la persona encargada, por diferentes motivos, no está presente para hacerlo, y es que en algo tan simple como dar a un botón en una aplicación, se puede complicar si asumimos que todos los compañeros tienen conocimiento de Access, y todo se complica todavía más si deben ejecutar una consulta SQL que por algún motivo de error porque simplemente haya habido algún problema al copiarla.

Para ver la foto completa del problema se pasa a describir el flujo del proceso ETL construido con Powercenter. El secreto por el que se consigue una mejor eficiencia con está acción respecto a la consulta SQL es porque se dispone de una tabla del Data Warehouse que ya integra la información de la consulta (ver Figura 2.25).

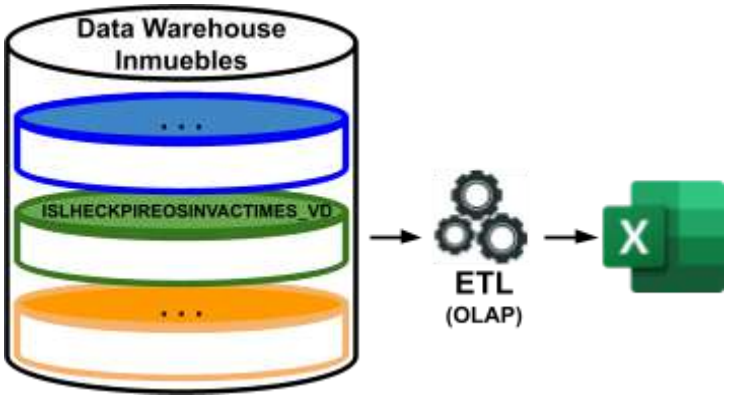

Figura 2.25 Nuevo procedimiento para generación del informe

Esta tabla contiene todos los campos necesarios, ya tratados (o semi-tratados), es decir , la labor de relacionar tablas e integrar datos unificándolos en un mismo destino ya se ha hecho con anterioridad, por lo que solo hay que consultar este origen único para la mayoría de los campos con nuestro proceso ETL, y si se tiene que realizar algún cálculo, este se realizará sobre los campos de la propia tabla mediante una pequeña transformación en el proceso, generando finalmente el informe deseado.

De esta forma se ha solventado el esfuerzo a la hora de obtener los datos pero no la parte en la que cualquier usuario pueda generar el informe sin tener conocimientos previos de la aplicación. Aquí entra en juego otro factor importante de las herramientas ETL, que es la conexión con otras tecnologías como podría ser una página web que contuviera un formulario a modo de puente entre Powercenter y el usuario, con un diseño amigable, que ejecutase el proceso para una determinada fecha en específico, recordemos que una característica del Data Warehouse es la historificación de los datos por lo tanto el informe se podría generar para todo el histórico de datos. Añadiendo a su vez, nuevas ventajas, ya que se podría hacer que a este formulario solo tuviera acceso un determinado número de usuarios de la entidad, y una vez generado el informe, configurar el envío del mismo a un directorio específico, controlado así el acceso a datos delicados (Figura 2.26).

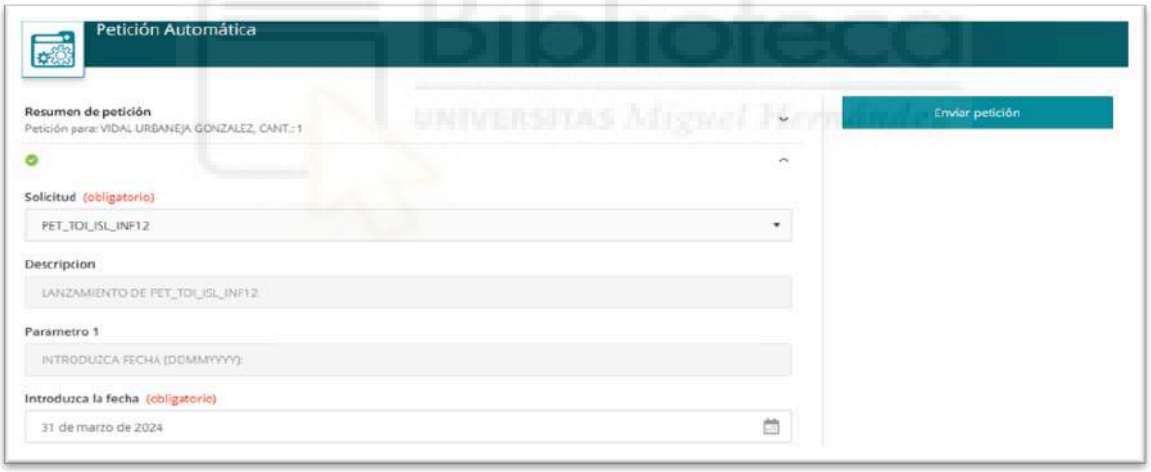

Figura 2.26 Formulario web para ejecución proceso ETL

Por tanto, aunque el diagrama final (Figura 2.27) internamente parezca más complicado y farragoso, para el usuario final que va a consumir los datos, tan solo es un sencillo informe que pueden generar él y otros usuarios, concluyendo y demostrando como la integración de los datos ayuda a resolver el "*chiringuito*", solventando una carencia en el tratamiento de los datos en la entidad, que es la razón del presente trabajo.

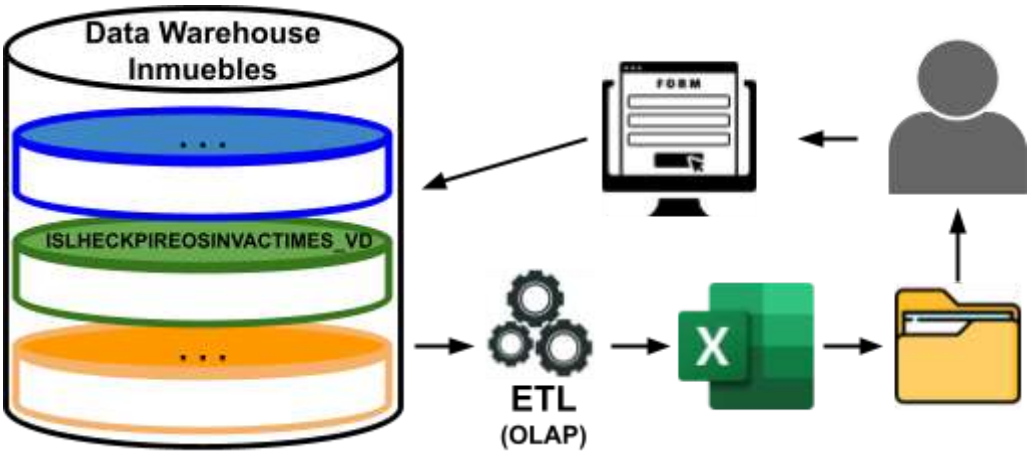

Figura 2.27 Nuevo procedimiento final.

Capítulo 3
Hipótesis de trabajo

3.1.- ETL[8]

Uno de los conceptos que se abordaban en los anteriores capítulos era la ETL (siglas en inglés de *Extract*, *Transform* y *Load*), se refiere a un proceso mediante el cual se consigue la integración, calidad y consistencia de los datos en el almacén de la empresa u otros destinos, cómo se verá en el apartado del Data Warehouse.

Figura 3.1 Fases de una ETL

Estas operaciones desempeñan un papel fundamental en la construcción y mantenimiento de estos Data Warehouse, garantizando a los usuarios que lo consultan, datos de alta calidad y fiables, si bien existen muchas más técnicas para la integración, es la más importante en este trabajo y en muchas ocasiones constituyen el proceso de integración en sí, a continuación se hará un análisis de sus fases (Figura 3.1) y las labores que desempeñan.

3.1.1.- Extracción

En esta fase los datos son recolectados de múltiples fuentes de origen, en la mayoría de casos, otras bases de datos relacionales, aunque también puede haber: archivos planos y servicios web entre otros, estos datos pueden estar estructurados, semiestructurados o no estructurados. La etapa crítica en este apartado radica en garantizar la disponibilidad de los datos desde los orígenes, pudiendo perjudicar los siguientes tramos de la ETL, si llegaran con demora. Por eso, la programación y sincronización con las fuentes orígen es fundamental.

3.1.2.- Transformación

Una vez extraídos los datos, se ejecutan una serie de procesos de preprocesamiento de datos, en los que se limpian y filtran de errores y duplicados, se aplican reglas de negocio, se realizan cálculos incluyendo agregaciones, se enriquecen y se cambian su estructura o formato (si fuera necesario). La finalidad es garantizar la calidad del dato y su compatibilidad con el resto de datos del almacén.

3.1.3.- Carga

Después de haber sido extraídos y transformados, son cargados en el almacén de datos (Data Warehouse) u otro repositorio de datos para su posterior consulta o análisis. La carga tiene la posibilidad de hacerse parcial o completa, además de tener una frecuencia puntual o periódica. En esta fase se procede a informar las marcas de tiempo para que el Data Warehouse adquiera su característica de ser un repositorio de datos histórico.

3.2.- DATA WAREHOUSE[7]

En el capítulo 2 se introducía el concepto de Data Warehouse como infraestructura o almacén usado en la entidad para el tratamiento de los datos inmobiliarios , en este

apartado se va a describir todos sus puntos técnicos, características, procedimientos y resto de elementos que participan y están relacionados con él.

3.2.1.- Características

Haciendo un pequeño recordatorio del capítulo 2, el Data Warehouse es un depósito de datos centralizado que, usando la definición del Sr. William Harvey Inmon cumple (Figura 3.2), cumple las siguientes características:

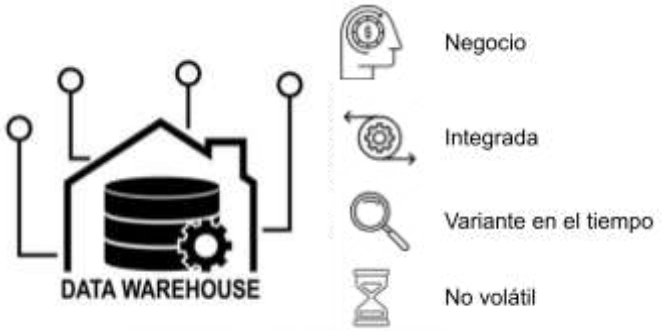

Figura 3.2 Propiedades del Data Warehouse

- Datos integrados: El almacén de datos recibe la información de diferentes orígenes tales como: otras bases de datos relacionales, ficheros planos como .csv o .txt, servicios web, etc., para ser tratados y limpiados.

- Orientados al negocio: Una vez son normalizados y organizados son una fuente de orígen fiable para la toma de decisiones y como punto de partida para programas de *Business Intelligence[47]* (en este mismo capítulo se presentarán algunas en el apartado de herramientas de visualización de datos) que expondrán estos datos de forma visual.

- Variante en el tiempo: Este depósito de datos mantendrá un histórico de datos, marcando cada partición con sellos de tiempo (fechas) para así poder avanzar y retroceder y tener una foto completa de la evolución de los mismos como si de un "*timelapse*" se tratara.

- No volátil: Esta característica aparece con la necesidad de tener un depósito de tablas estable y que no varíe modificando o eliminando datos en las diferentes ejecuciones. Por lo tanto, a la hora de operar el Data Warehouse las acciones se suelen limitar a consultar o ingresar nuevos datos, si se quiere modificar o eliminar siempre es mejor ingresar una copia de esos con algún campo que identifique esta nueva versión y así los datos no sufren ninguna pérdida a la hora de ser analizados.

Estas cualidades permiten manejar un gran volumen de datos recogidos a lo largo de los años en un mismo entorno centralizado y evitando la redundancia de recibir el mismo dato de diferente orígenes, en contrapartida presenta una serie de desventajas que tenemos que tener en cuenta:

- Al ser un depósito de datos centralizado, se necesitan los recursos computacionales tanto de hardware como de software, suficientes para dar cabida a todos los datos y futuras ampliaciones.
- Tener todos los datos unificados es una hoja de doble filo, puesto que las modificaciones en los procesos que lo cargan pueden ser operaciones muy delicadas, además de tener muchos datos sensibles ubicados en un mismo sitio acarreando problemas de seguridad.
- Los beneficios de su implantación son a medio y largo plazo, como se apreció en el último apartado del capítulo 2, una vez implementado se reduce el esfuerzo considerablemente, pero el trabajo de analizar, diseñar y construir el Data Warehousing [32] no se hace visible hasta posteriores desarrollos.

Ya vista las características, se va a continuar explicando sus componentes y la construcción del mismo o Data Warehousing (DWH), este término, usado erróneamente para referirse al Data Warehouse, alude al proceso de construir y mantener el almacén de datos (DW), mientras que el Data Warehouse es el almacén de datos en sí mismo, para ello, se va a seguir con el esquema de la arquitectura de un Data Warehousing clásico como es el usado en inmuebles (Figura 3.3):

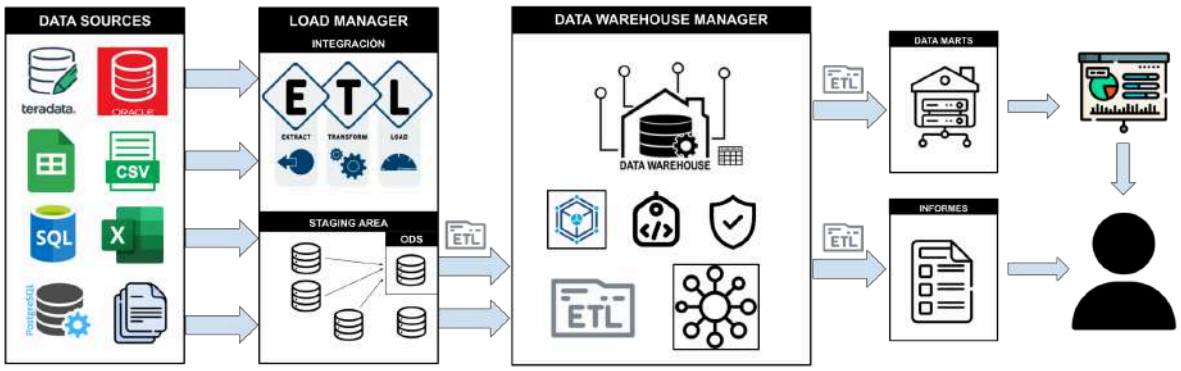

Figura 3.3 Arquitectura del Data Warehousing

3.2.2.- Data Sources[39]

Los Data Sources (Figura 3.4) son el orígen de los datos del Data Warehouse. Estas fuentes de datos pueden proceder de distintos lugares y en formatos diferentes, algunas de las más comunes son:

- Archivos de texto planos (ejemplo más habitual archivo con formato .txt)
- Bases de datos transaccionales SQL y NoSQL (pueden estar alojadas en diferentes tipos de gestor de datos).
- Hojas de cálculo.
- Páginas web o Redes sociales.
- Servicios Web.
- Fuentes de datos abiertos (Open Data).

Figura 3.4 Data Sources

En el modelo de inmuebles de la entidad los orígenes llegarán de dos tipos de bases de datos fundamentalmente: de Oracle y Teradata, además de la lectura de diferentes tipos de ficheros planos (Figura 3.5).

Figura 3.5 . Orígenes consultados en inmuebles

3.2.3.- Load Manager[40]

El elemento Load Manager será el encargado de administrar todos los procesos de integración, y abarca desde que se obtienen los datos en los diferentes orígenes hasta que se cargan en el Data Warehouse. Esta *integración* consiste en una serie de técnicas y procesos ETL que se encargan de llevar a cabo todas las tareas relacionadas con la modificación, extracción, control, depuración, carga y actualización del almacén de datos. Para facilitar este mecanismo aparece el *Staging Area[6]*, que hace referencia a un

almacenamiento auxiliar, en el que poder llevar a cabo las técnicas de integración para homogeneizar los orígenes en un entorno común de datos estructurado (es decir, tablas). Además, el *ODS[33]* facilita el acceso rápido a datos operacionales del día. Este elemento permitirá la unión de consultas en un mismo gestor de bases de datos, y la eficiencia en el desarrollo de procesos ETL que carguen el Data Warehouse, puesto que todos sus orígenes partirán de un mismo ecosistema, y por consiguiente se podrán implementar técnicas de optimización a la hora de ejecutarlos. (Figura 3.6).

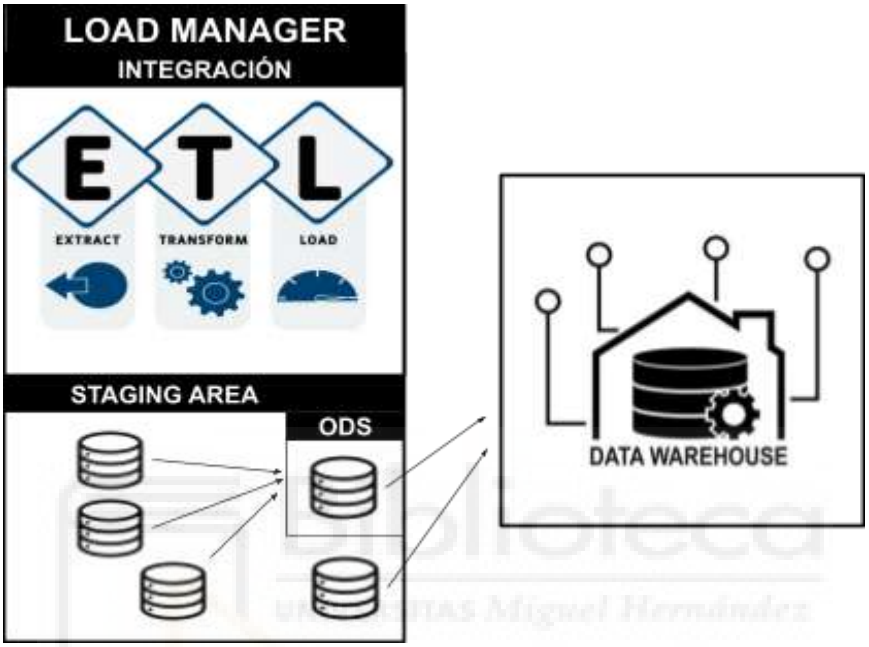

Figura 3.6 Load Manager

En el apartado 3.1 se comentaba qué es una ETL y como el proceso de integración es prácticamente el mismo (extracción, transformación y carga), en consecuencia, este apartado se va a enfocar en las otras técnicas usadas en la integración como son:

- Codificaciones: Un problema común que nos encontramos en la fase de transformación, es que los datos al venir de diferentes fuentes, cada orígen, codifica los atributos de una forma distinta, un ejemplo podría ser un indicador de ocupación, que puede llegar como "S" o "N", pero también ser codificado como "1" o "0", incluso siendo la misma fuente la que envía la información nos encontramos con esta casuística. En el Data Warehouse de inmuebles existe además otra forma de codificación una vez recibidos los datos, se trata de una *tabla diccionario* con un campo clave y otro valor, para los casos en que los datos esperados de las fuentes siempre son los mismos. Imaginando las ciudades de España, para *Alicante* se codificaría el "01*"*, *Valencia* el "02*"*, *Castellón* "03*"* y así sucesivamente, siendo estos valores numéricos los que se grabarán en las tablas del almacén en vez de sus significados. Esta práctica, además de lo evidente, que es no sobrecargar de información en el Data Warehouse, también ayuda a limpiar los

datos, puesto que en un caso de codificación de entidad bancarias el dato “Banco Santander” y “Santander” sería el mismo, con esta solución no perderíamos registros con el mismo significado y permitiendo una agilidad a la hora de leer y hacer consultas SQL, como sería en la codificación de valores más largos como “Juzgado de Primera Instancia e Instrucción Nº1 de Alicante” donde se transformaría a “01”.

- Medida de Atributos: Algo similar puede llegar a pasar a la hora de informar las medidas de las unidades, tal vez en un origen nos encontremos puntos para separar los decimales, cuando en otro origen diferente, nos llegan usando comas, en teoría dos separadores decimales válidos, pero a la hora de integrar hay que decidir por el uso de uno u otro, y transformar a la elección el no elegido.

- Redundancia: En la parte de características del DW, se describía como ventajas la eliminación de tener múltiples campos con información idéntica, es en esta fase de integración, donde se deberá elegir la fuente más apropiada y fiable para eliminar la redundancia. En el modelo de inmuebles, en estos casos se suele informar el origen que esté mejor informado, y en los casos que no estén informados, mediante sentencias SQL, usar la lógica *CASE WHEN* que son expresiones condicionales sumado a la sentencias *IS NULL*, que sirve para ver si el campo llega sin valores, conseguir que en el caso donde el origen principal viene a nulo, se intente informar de los otros orígenes restantes, esta es una práctica usualmente utilizada para que las fuentes secundarias ayuden a completar la información de la principal.

- Entorno: Uno de los puntos más importantes de la integración, es unificar las distintas fuentes en un mismo entorno, en este punto es cuando entran en juego el Staging Area y el *ODS (Operational Data Store)[33]*, ya que aparte de servir como contenedores auxiliares de los datos, antes de la incorporación al Data Warehouse también se pueden realizan operaciones adicionales. Algunos autores sitúan este almacenamiento auxiliar dentro del DW, en el modelo de inmuebles esta característica se presenta de forma híbrida.

3.2.4.- Data Warehouse Manager[41]

El Data Warehouse Manager (Figura 3.7) se refiere a los elementos necesarios para el diseño, gestión y mantenimiento para la administración del Data Warehouse, como bien recordaremos en la definición del Data Warehousing, el DW tan solo representa el almacén de datos en sí, pero en su construcción y manejo aparecen nuevas tecnologías. Para una mejor comprensión de este término, se realizará un símil con la construcción de un coche. Hasta ahora el concepto de Data Source podría interpretarse como los diferentes proveedores de piezas que suministran a la fábrica automotriz, en cuanto a la integración,

el *Data Staging area* junto al *ODS*, sería el envío y recepción en el almacén de la fábrica de todas las piezas necesarias, ordenadas y etiquetadas, donde finalmente el Data Warehouse equivaldría al coche ya fabricado que usa el cliente. Lógicamente para llegar al producto final existen una serie de tecnologías y etapas implementadas que son lo que corresponde a esta fase del Data Warehousing, tales como el diseño del coche (estructura seguida para el montaje del DW), las conexiones tanto eléctricas como estructurales del automóvil (modelo que sigue el depósito de datos), un chasis sobre el que irán los elementos (el *SGBD* o gestor de base de datos que arropará el almacén), el motor *OLAP*[34] correspondiente a las consultas que explotarán el DW, un sistema de seguridad y etiquetado (metadatos) por si hubiera que hacer reparaciones, y por último, máquinas y operaciones que ensamblen todos los componentes y añadan nuevas funcionalidades en un futuro (ETLs), a continuación se describen las características anteriormente explicadas:

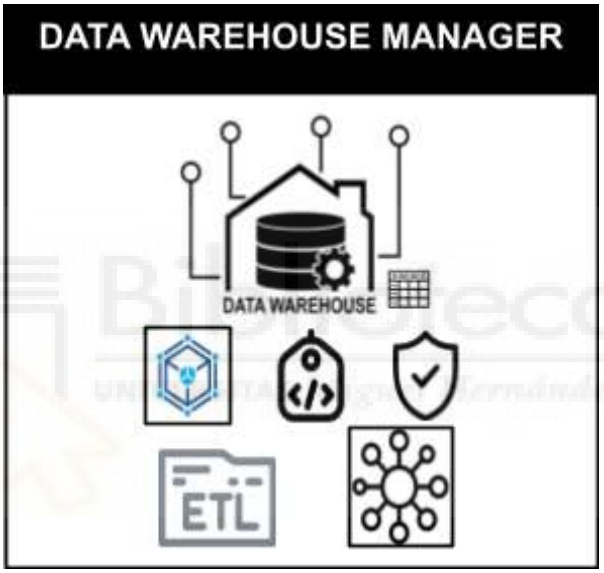

Figura 3.7 Data Warehouse Manager

Arquitectura del DWH

Con anterioridad se ha comentado que se considera a Bill Inmon como el padre del DW, aunque este concepto, al tener una evolución a lo largo del tiempo y con la superación de nuevos desafíos, es difícil asignarle la totalidad de su invención. Otro pionero en este campo es Ralph Kimball que introdujo importantes cambios y una nueva metodología alternativa. Las diferencias más notorias entre ambos modelos se pueden resumir en dos:

- Enfoque general: Mientras que Kimball abogaba por un enfoque bottom-up (Figura 3.8), donde los datos se integran primero en *Data Marts[38]*, considerados como almacenes temáticos más pequeños y específicos del DW, para luego combinarse en el propio Data Warehouse. Inmon proponía un enfoque top-down en el que primero va el Data Warehouse y después los Data Marts (sentido opuesto).

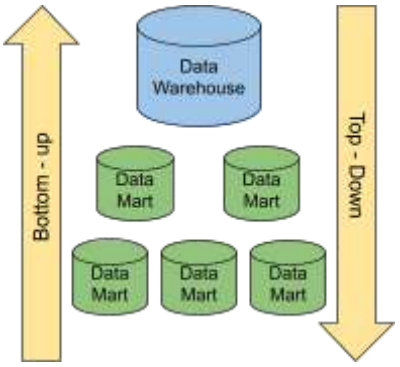

Figura 3.8 Estrategia procesamiento Bottom-Up y Top Down

- Modelo de datos: Inmon propone un modelo de datos relacional normalizado, sin embargo Kimball introduce un nuevo diseño llamado modelo dimensional, que se centra en el diseño de esquemas de estrella o copo de nieve para representar los datos de manera intuitiva y fácil de entender para los usuarios finales (Figura 3.9).

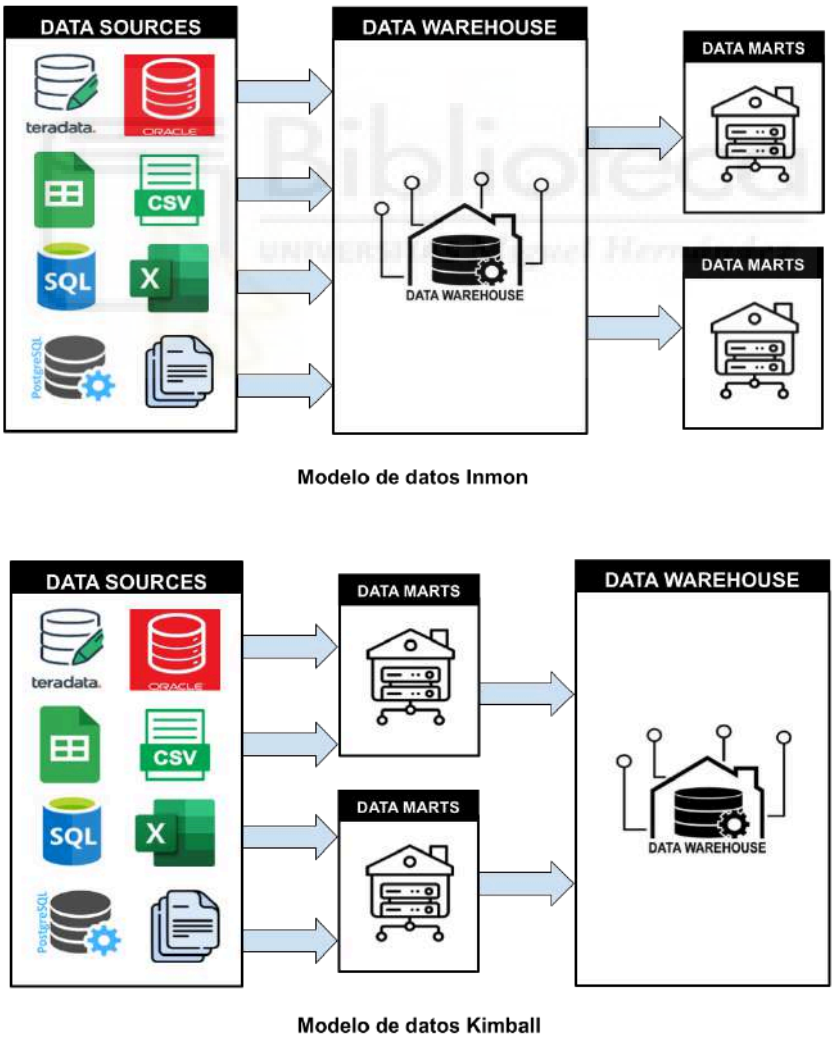

Figura 3.9 Diferencias modelo de datos Inmon y Kimball

Aunque ambas tecnologías sirven para construir el Data Warehouse, cada una tiene sus ventajas y desventajas, el motivo por el que se explican es debido a que el modelo de inmuebles de la entidad opta por desarrollar un modelo híbrido, parte de la construcción descrita por Inmon, pero siguiendo un modelo dimensional y en estrella definida por Kimball dando lugar al siguiente apartado.

Metodología del Data Warehousing

El Data Warehouse de inmuebles está interconectado de forma híbrida siguiendo un *modelo de estrella[42]* (Figura 3.11) con características del modelo de *copo de nieve[45]* (Figura 3.12). En el esquema de estrella, su estructura de datos está desnormalizada o parcialmente desnormalizada (Figura 3.10).

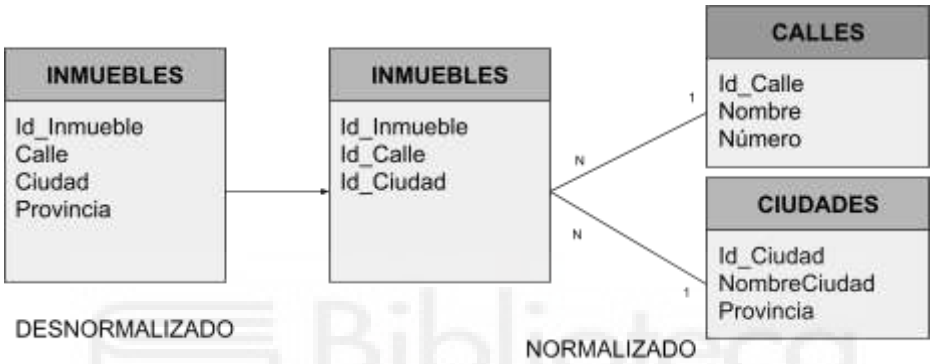

Figura 3.10 Ejemplo tablas desnormalizada y normalizada

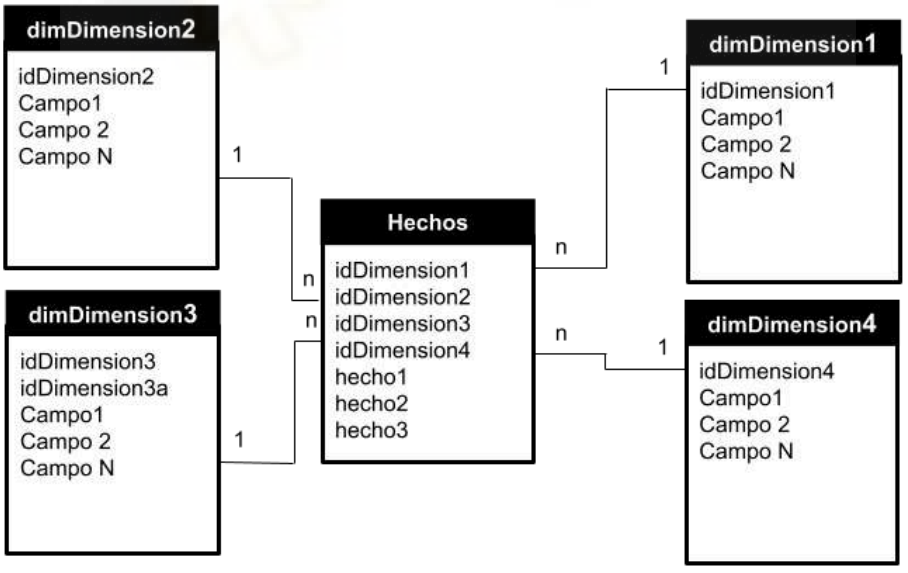

Figura 3.11 Esquema en estrella con tablas de hecho y dimensión

Esto evita desarrollar más uniones para la obtención de la información. Consta de dos elementos, la tabla de *hechos[43]* y las tablas de *dimensión[44]*. La tabla de *hechos* es una tabla central que almacena datos cuantitativos muy interesantes en el negocio como:

importes, claves foráneas, atributos de tiempo, etc. y medidas agregadas como: sumas, promedios, mínimos, máximo etc., que son utilizados para el análisis y la generación de informes. Por otro lado, las tablas de dimensión están destinadas a proporcionar información adicional a las tablas de hecho, diseñadas para ser estáticas y no cambiar con frecuencia, se enfocan en atributos descriptivos que ayudan y dan contexto a las medidas de las tablas de hecho, un ejemplo de estas tablas suelen ser: clientes, tiempo (día, mes año), ubicaciones etc.

Existen otros modelos como el anteriormente mencionado de *Copo de Nieve* o *Constelación[46]*, el primero es una extensión del modelo en estrella pero con la diferencia de que sus tablas están normalizadas traduciéndose en más uniones aunque consigue una menor redundancia, y el segundo se compone de múltiples modelos de estrella (Figura 3.12).

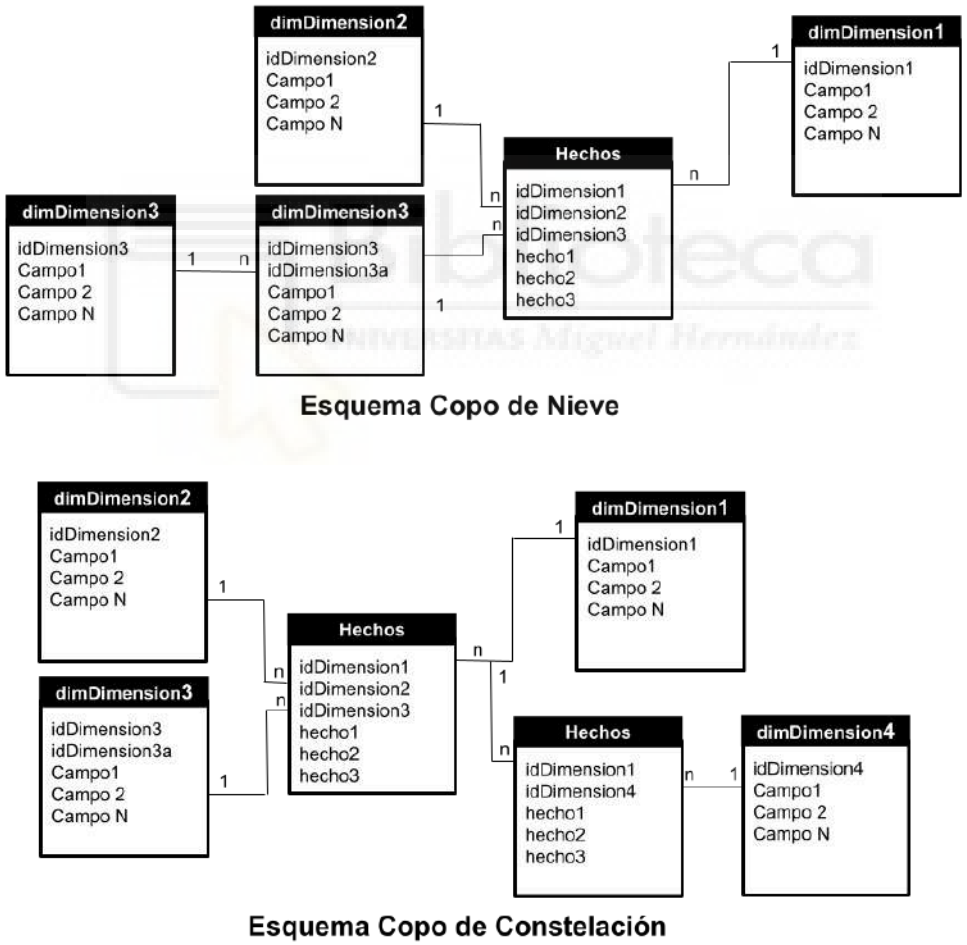

Figura 3.12 Modelo de Copo de Nieve y Constelación

Implementación del Data Warehousing

Recorriendo lo definido anteriormente, en una empresa, los datos de las transacciones realizadas parten del entorno operacional (Data Sources), al sistema *ODS* del *Staging Area*

(recordemos que es un almacén auxiliar del operacional) del cual bebe el Data Warehouse en el ecosistema informacional, todas las etapas transcurren gracias a sus procesos ETL. En ese punto, en el símil del coche se introducía una nueva tecnología que hacía de motor y era *OLAP (Online Analytical Processing)*, se describe como una capa adicional utilizada para consultas y análisis multidimensionales de los datos almacenados en el Data Warehouse organizada en cubos *OLAP*. (Figura 3.13)

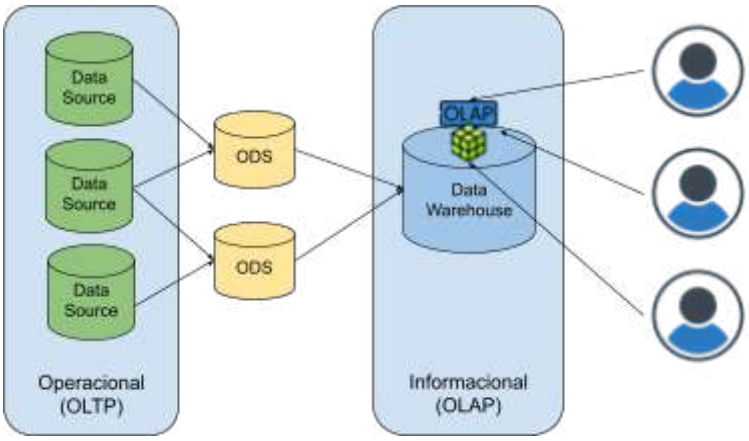

Figura 3.13 Tecnología OLAP

Un cubo OLAP (Figura 3.14) es la estructura en la que se organiza esta tecnología, este diseño permite realizar las dos operaciones más comunes: "drill-down" (desglosar el detalle de los datos) y "roll-up" (acotar los datos a niveles más generales).

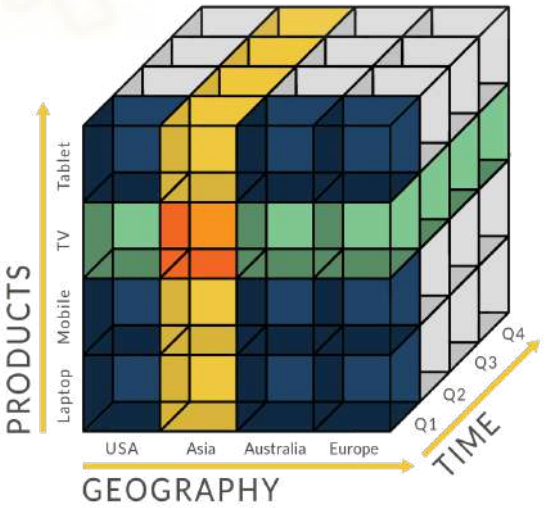

Figura 3.14 Ejemplo de Cubo Olap

Tomando el cubo de la Figura 3.14 como ejemplo, la operación de *drill-down* en el tiempo, podría seguir el detalle de : Cuatrimestre -> Trimestre -> Mes -> Día o para la geografía: Europa-> España -> Comunidad Valenciana -> Alicante-> Santa Pola, si quisiéramos cambiar a una operación de tipo *roll-up* en productos tal vez encontraríamos: A14 5G ->

Samsung -> Teléfono, como se observa el cubo está formado a su vez por pequeños cubos, en el ejemplo anterior, el cubo más pequeño sería un móvil A14 5G vendido en un determinado día en Santa Pola, un cubo más grande podría ser el móvil A14 5G vendido en España en un mes determinado. Con estos ejemplos, evidenciamos que el uso de los cubos OLAP facilita el análisis de los datos y mejora la eficiencia con la que se consulta el almacén. Existen tres tipos de implementaciones OLAP:

- ROLAP[35]: En los sistemas ROLAP (Relational On Line Analytic Processing), no existen estructuras físicas separadas sino que los cubos se implementan en el momento en que se realizan las consultas SQL a las tablas relacionales. Se organizan en tablas de *hechos* y *dimensión* (recordemos el *esquema de estrella*), con una generación dinámica de los cubos en respuesta a la consulta SQL utilizando vistas y agregaciones, que a diferencia de la siguiente implementación (MOLAP), los datos no están precalculados, es decir, las agregaciones se calculan en tiempo real. Usando como ejemplo el cubo OLAP de la Figura 3.14, en un sistema ROLAP, los datos de los productos vendidos se almacenarán en una tabla de hecho, mientras que los de geografía y tiempo en tablas de dimensión respectivamente. La consulta ROLAP resultante, generaría en tiempo real un cubo virtual con las ventas de los productos agregados por mes, producto y geografía permitiendo analizar los datos en el momento sin tenerlos almacenados. La desventaja de este tipo de implementación viene dada por tiempos de respuesta mayores para consultas de cierta complejidad puesto que los cálculos se hacen sobre la marcha.

- MOLAP[36]: MOLAP (Multidimentional On Line Analytic Processing) a diferencia de ROLAP, precomputa los cubos multidimensionales y almacena los datos, lo que acelera el tiempo de respuesta de consultas complejas sobre los datos precalculados del cubo y la optimización en entornos con grandes volúmenes de datos. Tomando como ejemplo el cubo OLAP de la Figura 3.14, si se quisiera analizar los teléfonos vendidos mensualmente en Europa, en un sistema MOLAP esta selección ya se almacenaría en un cubo, además de otros cálculos como ventas totales en los diferentes países. A la hora del análisis, estos datos ya estarían listos para ser analizados. Las desventajas de implementar este sistema, radican en una flexibilidad limitada, puesto que si la selección varía se tendría que reconstruir un nuevo cubo, un tiempo de carga inicial de los cálculos en el almacenamiento, y por su puesto, más recursos de almacenamiento para los datos y estructuras precalculadas.

- HOLAP[37]: Como sus siglas indican (Hybrid On Line Analytic Processing) es un sistema híbrido que combina ROLAP y MOLAP. Sumando sus beneficios, para consultas de datos simples, sin un alto nivel de agregaciones, se usará las bases relacionales de ROLAP, y para consultas rápidas de datos agregados y

precalculados se usará MOLAP, pudiendo combinar al mismo tiempo los dos sistemas. Siguiendo con el ejemplo del cubo OLAP de la Figura 3.14, las ventas de todos los móviles en Europa durante un año se podría almacenar en una estructura multidimensional (MOLAP), y para consultas más detalladas, como ventas individuales en un determinado país ser almacenadas en una base de datos relacional (ROLAP), así a la hora de analizar ventas totales en grandes volúmenes de datos de regiones completas se podrá utilizar datos precalculados y almacenados y para ver las ventas específicas por cliente hacer consultas directas a la base de datos relacional. Esta combinación solventa los problemas anteriores de la alta espera en la respuesta de las consultas por parte de ROLAP y la menor necesidad de almacenamiento al no tener que precomputar todas las posibles agregaciones. En contra, la implementación y mantenimiento de un sistema HOLAP es de mayor complejidad y tiene un costo adicional.

3.2.5.- Análisis del Data Warehouse

La finalidad de todo Data Warehouse es ser explotado en su etapa final por parte de los usuarios, en este apartado se describirá algunas de las herramientas (Figura 3.15) que hacen de nexo entre el DW y los clientes en el modelo de inmuebles.

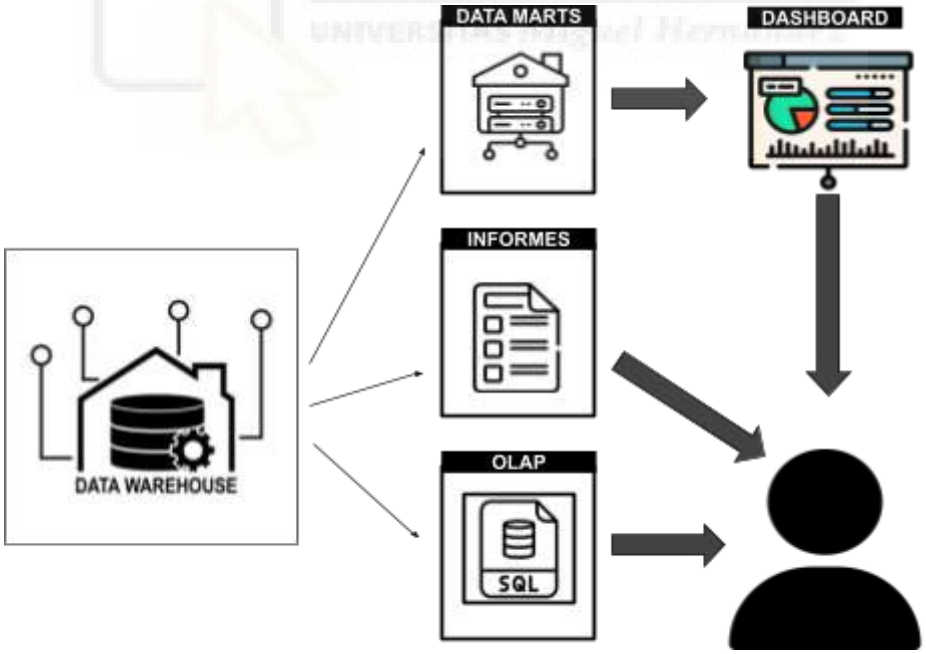

Figura 3.15 Ejemplos usos DataWarehouse

- OLAP: Descrito en el apartado anterior, es el componente más usado en el modelo, mediante consultas sobre el DW, proporciona respuestas rápidas a preguntas complejas, además de utilizar operados como el Drill-up o Drill-down, entre otros, para analizar datos históricos o en más detalle comportamientos y evoluciones.

- Reporting (informes): Esta herramienta fue de las primeras en implantarse en el modelo, la generación por ejemplo de informes de inmuebles vendidos, es de los más importantes, tanto a nivel de análisis como en el departamento de contabilidad.

- Dashboards[45] (tableros de control o cuadros de mando): Son herramientas muy visuales, que recopilan, organizan y presentan los datos por pantalla de forma comprensible para el usuario haciendo un seguimiento de los famosos indicadores clave de desempeño (o *KPIs*[46]) como: saldos, importes, cantidades, entre otros. En este punto entrarían a formar parte todos los programas de *Business Intelligence (BI) [47]* que faciliten la creación de Dashboards.

3.3.- GESTORES BASES DE DATOS (SGBD)[48]

Los sistemas de gestión de bases de datos (componente que forma parte del Data Warehouse Manager visto en el apartado anterior) son fundamentales para el funcionamiento de un Data Warehouse, ya que albergan y organizan las tablas estructuradas utilizadas en el mismo, además de permitir a los usuarios consultar y modificar mediante lenguaje SQL su información, añadiendo funcionalidades de seguridad y control de acceso para proteger la confidencialidad de los datos. En este apartado se va a exponer las dos herramientas utilizadas que administran las bases de datos en este trabajo.

3.3.1.- Teradata SQL Assistant[50]

Teradata SQL Assistant (Figura 3.16) es una herramienta de desarrollo y consulta de bases que pertenece a la compañía Teradata Corporation. Algunas características clave incluyen:

Figura 3.16 Teradata SQL Assistant

- Editor consultas SQL (Figura 3.17): Facilita un entorno de edición de texto que permite al desarrollador escribir y editar consultas SQL, cuenta con ayuda para destacar los errores de sintaxis, autocompletado y sugerencia de código SQL.

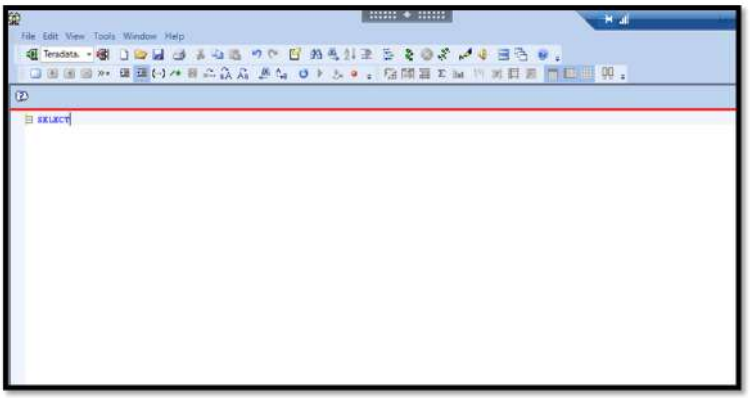

Figura 3.17 Editor de consultas SQL Teradata

- Ejecución de consultas: Permite al usuario la ejecución de consultas y la visualización de los datos en forma de tablas, también ayuda en la exportación de los datos en ficheros para su análisis de forma local.
- Seguridad: Las bases de datos están protegidas por permisos y roles, protegiendo los datos según el nivel de acceso que tenga el usuario.
- Explorador: Muestra y facilita la navegación dentro de las estructuras de las bases de datos a los que los usuarios tienen acceso.
- Historial: Permite a los usuarios guardar un historial de las consultas anteriormente ejecutadas.

3.3.2.- Oracle SQL Developer[51]

Desarrollada por Oracle Corporation (Figura 3.18), es una herramienta para administración de base de datos, entre sus características encontramos:

Figura 3.18 Oracle SQL Developer

- La exploración de la estructura de la base de datos y sus objetos como tablas, vistas, procedimientos almacenados entre otros.
- Creación y gestión de usuarios, roles y privilegios, además de copias de seguridad y restauración de bases de datos

- Contiene un editor de código SQL (Figura 3.19) que permite a sus usuarios escribir, editar y ejecutar consultas SQL.
- Cuenta con una interfaz amigable, y un depurador de errores SQL además de un motor de sugerencias de funciones y tablas muy necesario si todavía no se dispone de cierta desenvoltura en el entorno.
- También dispone, como Terada, de un histórico de las consultas ejecutadas con anterioridad.

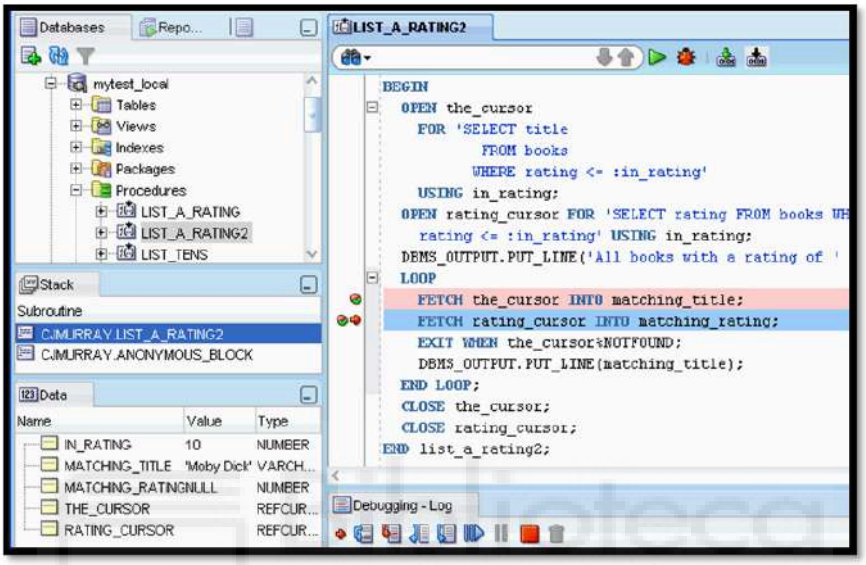

Figura 3.19 Editor SQL Oracle SQL Developer

En el modelo de inmuebles, las acciones que se pueden realizar en el gestor de bases de datos tanto de Teradata como el de Oracle, vienen dadas por los permisos que el usuario en cuestión tenga. Se configura de esta forma para no producir la pérdida o robo de datos delicados de la entidad.

Especial mención a los diferentes *entornos* que podemos encontrar en las base de datos de la entidad, no son pocas las veces que se oye e incluso se toma de forma cómica que alguien de la empresa ha borrado "sin querer" las bases de datos, por ello, en la mayoría de empresas ponen a la disposición de los desarrolladores diferentes servidores para la prueba de programas y procesos, así encontramos servidores para entornos de producción, preproducción y desarrollo, estos dos últimos pueden contener una copia o parte de los datos que se van a usar, lo que permite poner ciertas limitaciones tanto en seguridad a la hora de dar permisos o implementar estrategias que protejan los datos, por ejemplo, si los datos son sensibles posiblemente estén ofuscados en los servidores de desarrollo, para que de esta manera, probar consultas y procesos en entornos seguros donde el borrado o modificado de los datos no sea un peligro. Tanto en los gestores de bases de datos que se usan en este trabajo como la herramienta ETL Powercenter que se conecta a ellos dispone de estos entornos.

3.4.- INFORMATICA POWERCENTER[14]

En el capítulo 2, se describía las características que tiene Powercenter para la creación de ETLs, al ser la herramienta utilizada en la entidad para la integración de datos, se hará una breve explicación de cómo es su área de trabajo y los componentes que utiliza para una mejor compresión del capítulo 4. Powercenter se divide principalmente en tres componentes: Designer, Workflow Manager y Monitor (Figura 3.20). En pocas palabras, se podría decir que Designer es donde se van a desarrollar las ETLs, Workflow Manager tratará de su configuración y ejecución y el Monitor tendrá el trabajo de terminal, que arrojará los datos del funcionamiento y depuración de errores. A continuación, se describen estos componentes con más detalle.

Figura 3.20 Componentes de Powercenter

3.4.1.- Designer

El primer punto que el desarrollador se fija al entrar a Powercenter es cómo está organizado el Designer (el Workflow Manager y Monitor siguen la misma estructura). Se presenta como forma de directorios o en forma de esquemas y propietarios si fuera una base de datos (Figura 3.21):

Figura 3.21 Organización Powercenter

Como se observa, se parte de un símbolo de base de datos llamado “Repositories” para entrar en el entorno de desarrollo “REP_DESA” (como se comentaba en el apartado de gestores de bases de datos Powercenter también cuenta con tres entornos, desarrollo, preproducción y producción), lo siguiente que se encuentra son diferentes carpetas o aplicativos donde normalmente se organizan los “*Mapping*”, entendamos el mapping como

el desarrollo que detalla el proceso de transformación de los datos, desde las fuentes hasta los destinos, es decir, son representaciones ETL que se están organizadas para un mismo Data Warehouse o temática (Aplicativos) (Figura 3.22):

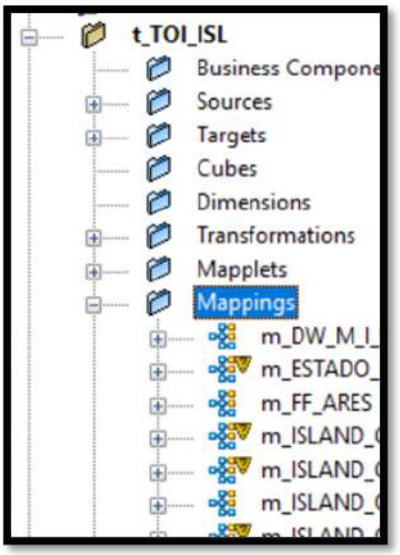

Figura 3.22 Aplicativos y Mappings

A la derecha de esta organización se encuentra la zona de trabajo del Designer, esta área de trabajo se reparte a su vez en otras cinco (Figura 3.23):

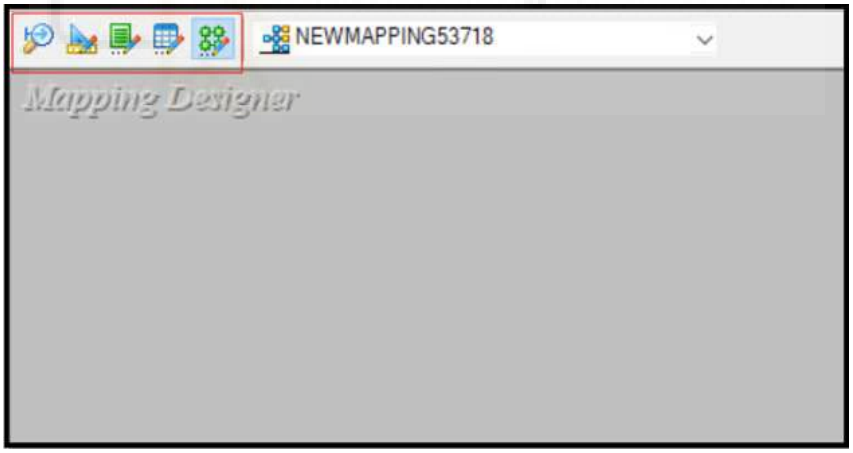

Figura 3.23 Áreas trabajos del Designer

En orden de izquierda a derecha, se presenta el Source Analyzer, destinado a la creación y modificación de los componentes del Data Source, el Target Designer para formar y modificar el destino final de los datos, Transformation Developer usado para construir componentes preestablecidos de Powercenter a nuestro antojo, el Mapplet Designer con el que se configuran procesos reciclables para ciertos cálculos preestablecidos, y finalmente el Mapping Designer donde se realizarán los desarrollos de las ETLs o parte de ellas, cada área hace que se puede configurar y personalizar elementos que irán encapsulados en la

ETL. Para no divagar en las diversas herramientas que ofrece Powercenter, nos centraremos en el área de desarrollo del Mapping Designer que es sin duda la más importante.

El Mapping Designer cuenta con una serie de componentes (Figura 3.24) ya prefabricados que formarán la ETL, son de fácil uso ya que tan solo se arrastran al área de trabajo y por el icono del elemento da una idea aproximada de su función.

Figura 3.24 Algunos elementos del Mapping Designer

A continuación, se hará una breve exposición de algunos de estos elementos, centrándonos en los que se usan con mayor frecuencia en el trabajo. En el capítulo 2, una de las características que se describían de PWC era la necesidad de tener conocimientos de lenguaje SQL (preferiblemente del gestor de base de datos que usa mayormente para la lectura y escritura de datos en la empresa), esta afirmación viene dada puesto que cualquier consulta SQL se puede traducir a "lenguaje Powercenter" y viceversa, la traducción final de esta cualidad es que la mayoría de sus componentes se pueden explicar usando SQL:

- *Source y Target Shortcut* (Figura 3.25): Estos dos elementos *shorcut* (en inglés atajo) comprenden la lectura de las fuentes orígen (*source*) y la carga en destino (*target*), es decir, las partes iniciales y finales de un proceso ETL. El elemento *Source shortcut*, dispone a su vez de dos componentes, el primero (SC), representa el orígen de dónde se lee, en este caso FTABACI_V que es una tabla alojada en Teradata, en cuanto al SQ que va ligado a ella, es un componente donde podremos empezar a filtrar datos.

Figura 3.25 Powercenter - Componentes Source y Target Shorcut

La carga de todos los campos de este origen con el filtro indicado traducido a consulta SQL: SELECT * FROM FTABACI WHERE FECHA_EXTRACCION = CAST('$$FECHAVIRTUAL' AS DATE FORMAT 'DDMMYYYY') (Figura 3.26).

Figura 3.26 Filtro SQ Powercenter del Source

Los siguientes tres componentes: *Expression*, *Aggregator* y *Filter* (Figura 3.27) son en gran medida los más usados en el proceso de integración de los datos:

Figura 3.27 Componentes Powercenter Expression, Aggregator y Filter

- *Expression:* Usado para operaciones (sin agregaciones) como sumas o restas , transformación de datos tales como sentencias lógicas de if-else (Consulta SQL la sentencia CASE) a nivel de fila.
- *Aggregator*: Equivalente a las funciones de agregado en SQL: SUM, MAX, MIN, AVG, etc seguido de un GROUP BY . Muy utilizada pues informan campos en su mayoría *KPI*, utilizados para alimentar cuadros de mando por su relevancia.
- *Filter*: Como su nombre indica en inglés, filtra siguiendo la condición que se interponga en su configuración. Su sinónimo en SQL sería la cláusula WHERE. A diferencia con el componente powercenter *Source Shorcut* que el filtro se hace en el propio origen antes de entrar al flujo de la ETL, este componente puede filtrar los datos en cualquier etapa de la ETL. Su traducción a consulta SQL vendría dada en el resultante de querer eliminar los datos devueltos producidos por diferentes uniones o subconsultas realizadas.

Por último, entre los componentes más importantes que nos encontramos en Powercenter no puede faltar el *Joiner* (Figura 3.28). Es el elemento encargado de unir los diferentes orígenes y unificar los flujos de datos.

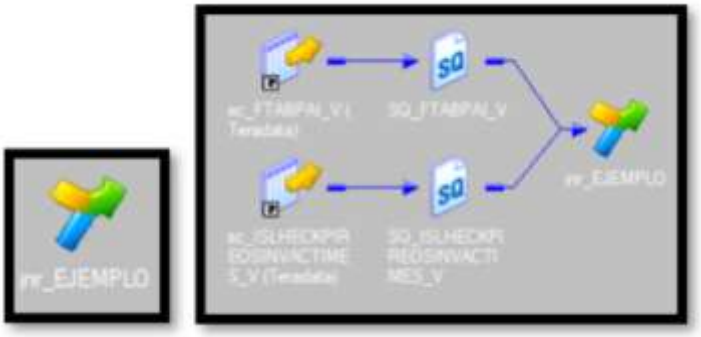

Figura 3.28 Componente Joiner Powercenter

Es sin duda el componente de Powercenter más utilizado y su operador en SQL es el mismo que da nombre a este componente: *JOIN*. Los principales tipos de JOINs[52] en SQL son los que se presentan en la Figura 3.29.

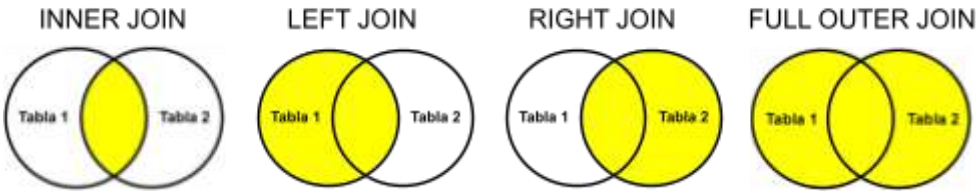

Figura 3.29 Diferentes tipo de Joins SQL

Para sintetizar, tanto el *left join* como el *right join* añaden información de la tabla secundaria a la tabla principal sin que esta pierda información en caso de tener registros comunes en los campos que hacen de unión. Caso diferente del *inner join* donde las dos tablas son principales y el resultado de la unión devolverá los registros que tengan datos en común, en los campos que hagan de enlace en las dos tablas. También hace de filtro para casos en los que tan solo se quiera el perímetro de una de las dos tablas, descartando el resto de registros, puesto que los datos que no sean iguales se descartarán del flujo. Por último, el *full outer join* añade la información de ambas tablas al flujo principal sin perder datos. Este tipo de Joins también existen en el componente Joiner de powercenter y se puede configurar entre sus propiedades (Figura 3.30).

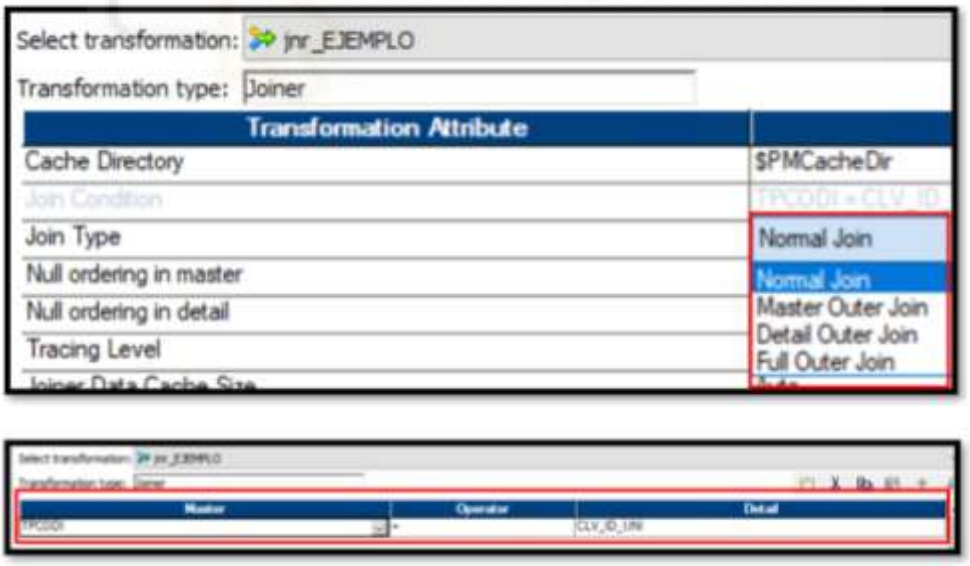

Figura 3.30 Configuración componente Joiner Powercenter

Estos componentes son los más usados en el trabajo y en el modelo de inmuebles, aunque existen otros elementos también usados en menor medida como: "*Union*"(función idéntica a la sentencia SQL UNION), "*Sorter*" (ORDER BY en SQL), "*LookUp*" (para hacer búsquedas de ciertos valores en campos configurados) entre otros.

3.4.2.- Workflow Manager

El Workflow Manager es el lugar donde se configuran los mappings para ser ejecutados, está organizado de igual manera que el Designer, donde el desarrollador primero se conecta al servidor del entorno que vaya a trabajar, seguido del aplicativo (donde se guardan las ETLs con misma temática) y por último los workflows: (Figura 3.31)

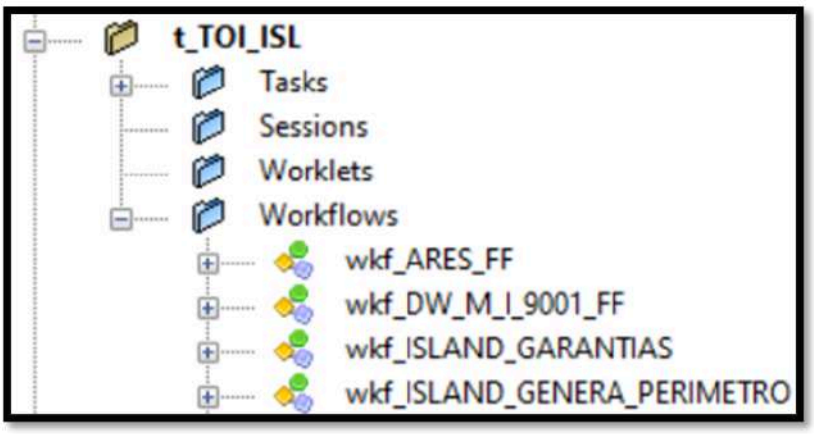

Figura 3.31 Organización Workflow Manager aplicativos y workflows

Cuenta con una área de trabajo visual similar al Designer donde se pueden arrastran elementos llamados *Tasks* o Tareas (Figura 3.32).

Figura 3.32 Task componentes Workflow Manager

De estos elementos el componente más importante es la session que es el encargado de hacer de enlace entre el mapping y el workflow, además de permitir configurar todos los elementos del mismo, algunas otras peculiaridades que se pueden configurar:

- Modificación de las conexiones tanto de orígenes como destinos que se leen y escriben en el mapping.
- Parámetros o variables, calculadas en el mapping o que serán enviadas a sesiones posteriores.
- Diferentes parámetros de depuración y memoria usada en el proceso..
- Especial mención a los diferentes métodos de carga: Relacional, MLOAD (MultiLoad) , FLOAD (Fast Load) o PDO (Push Down Optimization) son las más usadas en el modelo, entre otras.

En la siguiente figura 3.33 se presenta un ejemplo del entorno del Workflow Manager de cómo se relacionan las Task de las sesiones, con otras usadas para el cálculo de parámetros o decisiones en flujo.

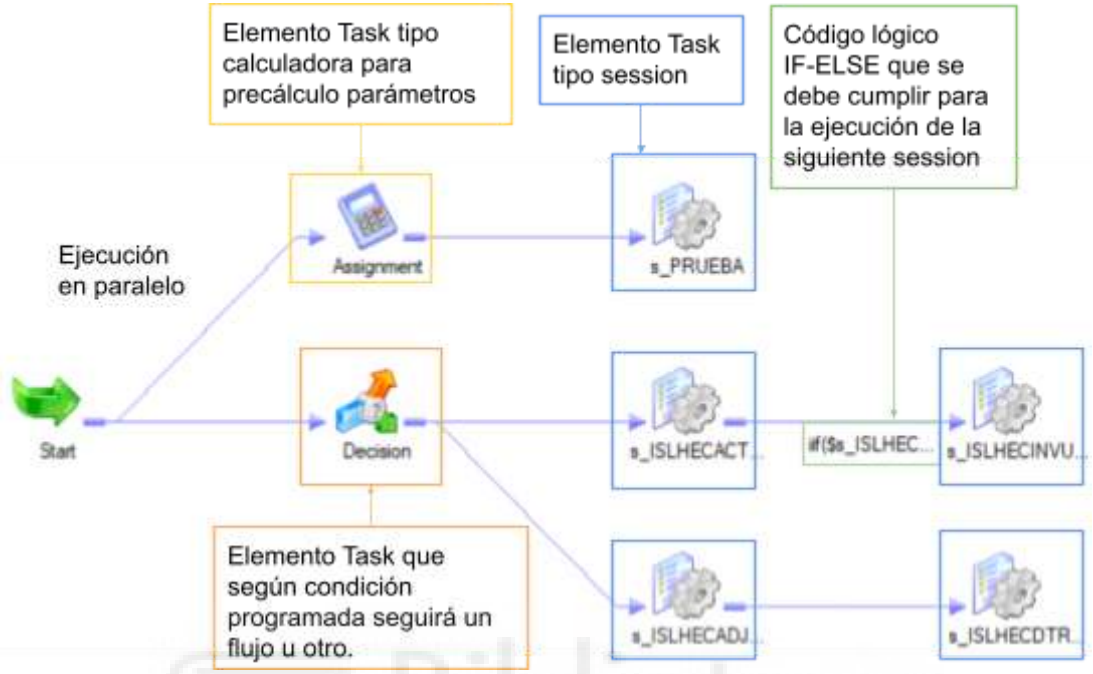

Figura 3.33 Entorno ejemplo Workflow Manager

3.4.3.- Monitor Manager

El Monitor Manager es la interfaz que usa Powercenter para supervisar, controlar y gestionar las ejecuciones de los workflow y sus sesiones, su organización es idéntica al Designer y Workflow Manager por aplicativos. Entre sus características más importantes se encuentran:

- Monitorización de ejecuciones: Permite ver el estado de los workflow y sesiones en ejecución, en las que localizamos, completados (*Succeeded*), fallidos (*Failed*), Deshabilitado (*Disabled*) etc (Figura 3.34), como los recursos que consumen, la hora de ejecución o cuando empiezan y terminan.

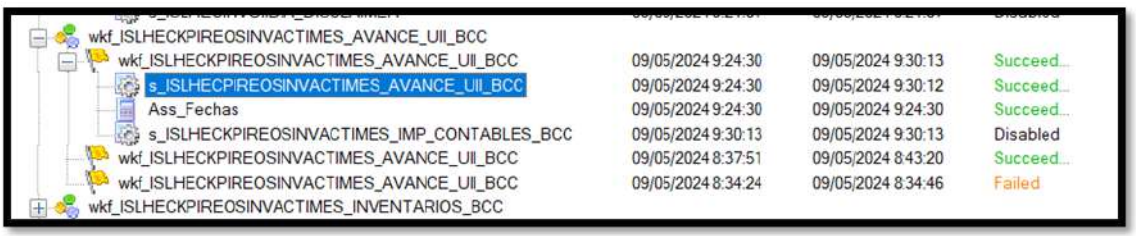

Figura 3.34 Estados Workflows y sesiones

- Administración de worflows y sesiones: Posibilita iniciar, detener, pausar y reanudar workflows y sesiones, además de la programación de los mismos para su ejecución en un determinado horario. A su vez, la configuración de alertas para notificar de workflows acabados o fallidos.
- Depuración y visualización de errores: Proporciona detalles e información de los errores encontrados durante la ejecución ayudando al programador a detectar fallas en los procesos ETL (Figura 3.35).

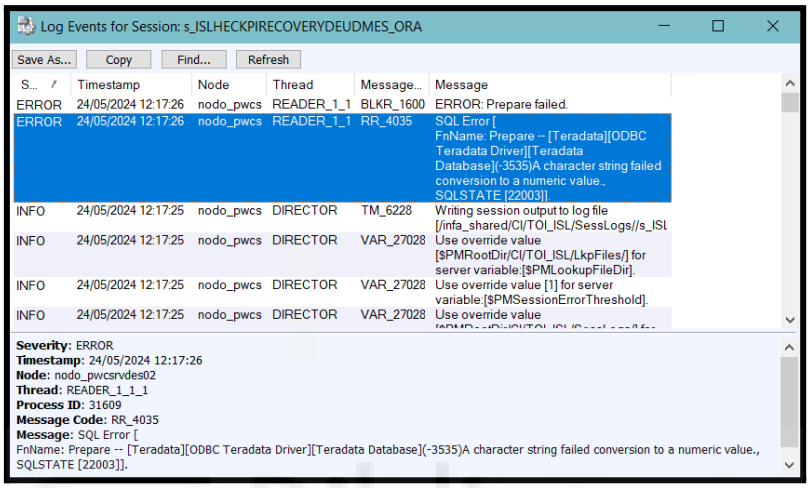

Figura 3.35 Depurador de errores Monitor Powercenter

3.5.- HERRAMIENTAS DE VISUALIZACIÓN DE DATOS

Las herramientas de *Business Intelligence [47]* son las encargadas de dar un sentido a los datos mediante la presentación de la información de forma útil y clara, ayudando a las empresas a la toma de decisiones y al análisis de datos históricos y actuales. A pesar de que en este trabajo no se abarca la creación de Dashboards, informes o gráficos, sí que se usan las aplicaciones descritas a continuación para validar que los datos en estas herramientas sean correctos, ya que los orígenes de la información que se presenta en ellas provienen, directa o indirectamente, del Data Warehouse de inmuebles y debe corresponderse con la misma.

3.5.1.- Salesforce[53]

Salesforce no es una herramienta pura BI, sino un CRM que se centra en la gestión de ventas, servicio al cliente y marketing, aunque sí incluye funcionalidades y servicios

dedicados al reporting y la creación de dashboards para la visualización y análisis de datos (Figura 3.36).

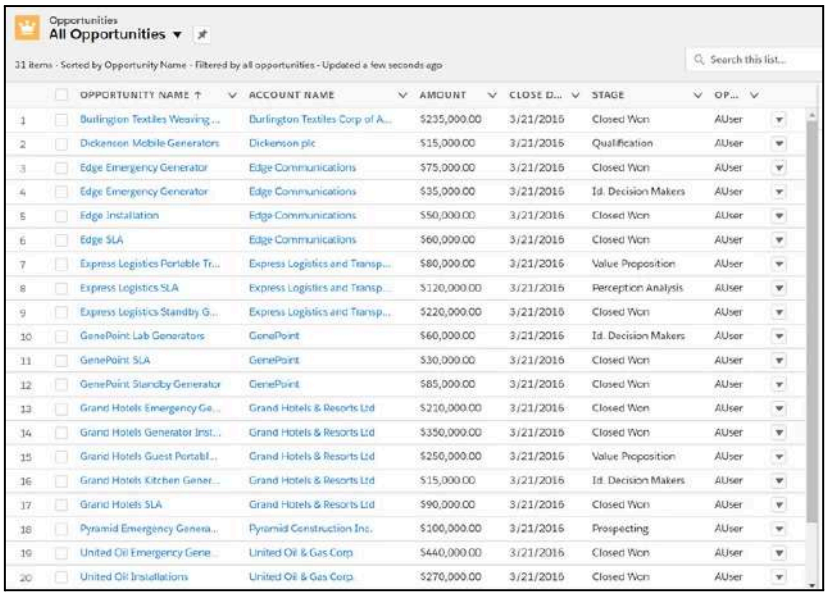

Figura 3.36 Dashboard Salesforce

En inmuebles, esta herramienta se utiliza por el departamento de comercialización, así que tendrá un uso más dedicado a la visualización de los datos más que de balances o contabilidad.

3.5.2.- QlikView[54]

QlikView es la herramienta de la empresa Qlik destinada a la analítica de datos de la empresa que facilita la toma de decisiones mediante una interfaz intuitiva, con un modelo de datos asociados que facilita la navegación rápida entre los datos, y la obtención de "*insights*" (Figura 3.37) o descubrimientos relevantes obtenidos derivados del análisis de los datos.

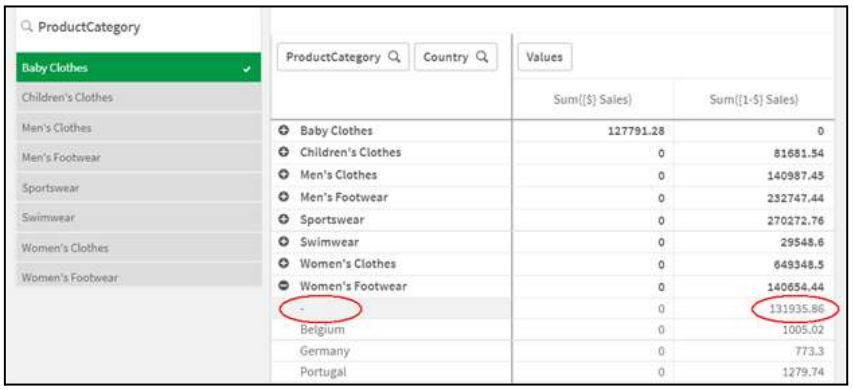

Figura 3.37 Visualización de datos relevantes en QlikView

3.6.- OTROS PROGRAMAS USADOS

En este apartado se van a describir herramientas que se utilizan con mucha frecuencia en el trabajo, pero por ser de uso general, véase como un ejemplo las herramientas de ofimática, no son descritas en apartados individuales por carecer de valor o interés especial para el lector, aunque sí que es necesaria una breve reseña por su uso continuado y función en el presente trabajo.

3.6.1.- FileZilla[55]

FileZilla es una aplicación FTP de código abierto que se usa para transferir archivos entre un cliente y un servidor. Teniendo en cuenta que los procesos ETL de powercenter generan archivos de datos como informes o reciben archivos para informar sus bases de datos, es necesaria la transferencia segura de archivos entre diferentes ubicaciones, algunas de sus características son:

- Integración de datos externos: En algunas ocasiones, como en el caso del modelo de inmuebles que recibe ficheros de la inmobiliaria, como fuente de información en ubicaciones externas, es necesario un programa FTP para facilitar la transferencia de estos ficheros al servidor donde reside Powercenter.
- Depuración y parametrización: Por otro lado, Powercenter genera archivos de tipo registro (.log), estos archivos que se generan automáticamente y son una fuente de datos para depurar errores en los procesos ETL (Figura 3.38), de igual manera recibe archivos paramétricos donde el usuario puede configurar fechas u otras variables para sus procesos ETL.

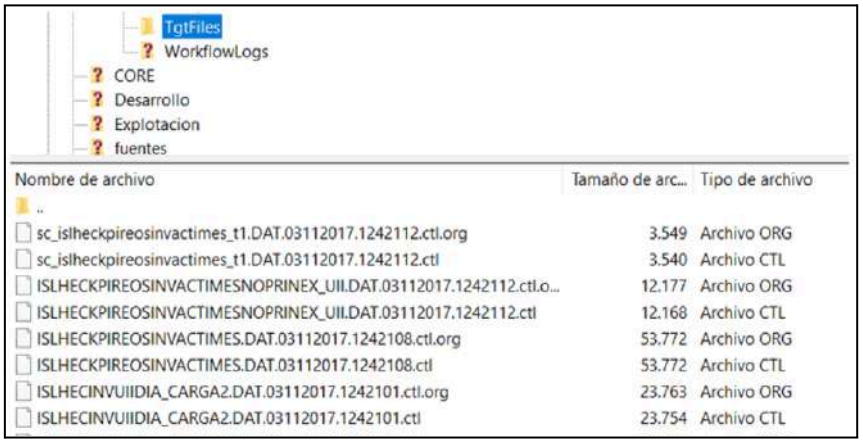

Figura 3.38 Log generados Powercenter

- Automatización de procesos: FileZilla puede programarse para la transferencia de estos archivos de forma automática, así cuando se genera un informe en un

determinado momento en el servidor donde se aloja Powercenter no es necesaria la acción humana para su transferencia.

- Estructura Jerárquica: Como se vio en PWC se puede ordenar los procesos que seguían una misma temática por aplicativos, FileZilla permite esta misma organización usando directorios para cada aplicativo (Figura 3.39)

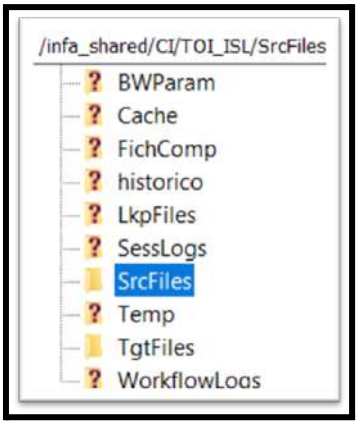

Figura 3.39 Organización directorios FileZilla

- Transferencia de archivos y seguridad: Powercenter puede generar una variedad de tipos de archivos, además de leer esa misma variedad, lo que es un riesgo para el robo de datos, FileZilla además de mover o alimentar Powercenter, crea una nueva capa de seguridad en los datos que son generados o recibidos por PWC gracias a mecanismos de autentificación (Figura 3.40) .

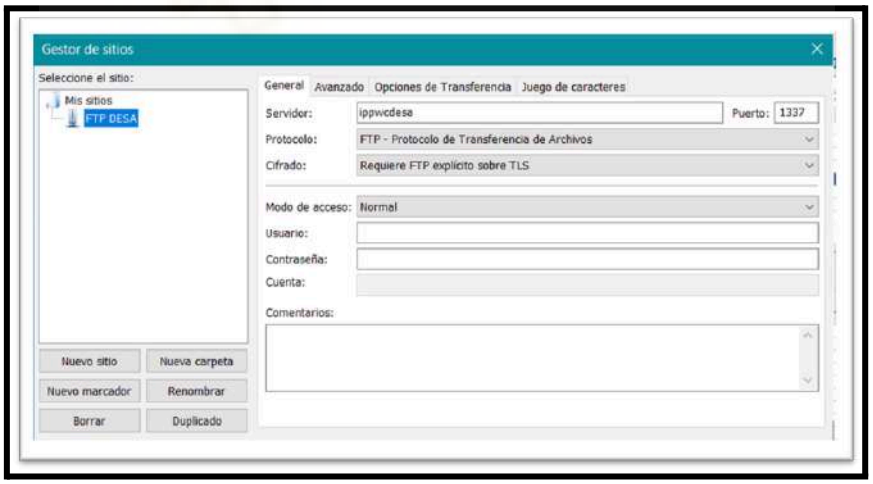

Figura 3.40 Seguridad FileZilla

Adicionalmente en este apartado de aplicaciones FTP, mencionar que en la entidad, al tratar datos de suma importancia, tener acceso al entorno de producción para subir o descargar archivos está muy restringido y se siguen otras vías para tener un historial de entrada además de la consiguiente autorización, aunque sí que dejan a la disposición de los

desarrolladores una aplicación web que permite visualizar los archivos que alimentan a los procesos ETL de producción o los archivos generados por los mismos (Figura 3.41):

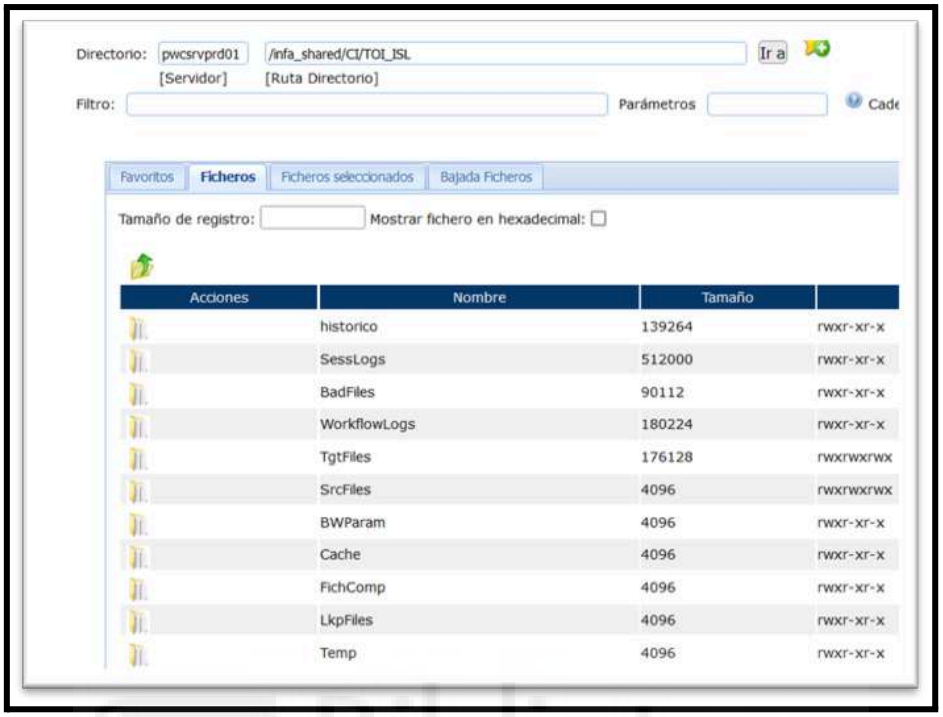

Figura 3.41 Herramienta Web visualización de ficheros

3.6.2.- Erwin Data Modeler[56]

Erwin es una herramienta de modelado de datos utilizada para el diseño de bases de datos. Permite a los usuarios, a partir de un modelo de datos conceptual, crear un modelo de datos lógicos, aunque el uso en la entidad es bien distinto. (Figura 3.42)

Figura 3.42 Erwin Data Modeler

Como se ha visto en apartados anteriores, los datos del almacén son estructurados y por tanto se guardan en tablas que a su vez contienen campos donde se guardarán los registros. ¿Dónde entra en juego Erwin? Debido a que el programador en la entidad no tiene permisos para la creación, modificación o eliminación de tablas del Data Warehouse sino que se delega en otro departamento de bases de datos que realiza estos trabajos, de alguna manera se debe indicar la acción a realizar sobre las tablas del modelo, ya sea para crear nuevas tablas, modificar sus campos o eliminar sus características como claves principales

u otras. En esta coyuntura entra el Erwin Data Modeler, sirve como documento para indicar al departamento de bases de datos, la operación que se va a realizar, puesto que si el analista o programador que es el que diseña las tablas del Data Warehouse no tiene la autorización necesaria por motivos de seguridad para crearlas, es necesario el uso de un programa que indica con la mayor exactitud posible la operación a realizar.

Su uso es muy sencillo, se organiza por Diagramas (Figura 3.43) que contendrá las tablas o "*Entities*" (en la Figura 3.44 vemos las usadas en Inmuebles) que se hayan creado o modificado con anterioridad de un mismo Data Warehouse o temática.

Figura 3.43 Creación de un Diagrama

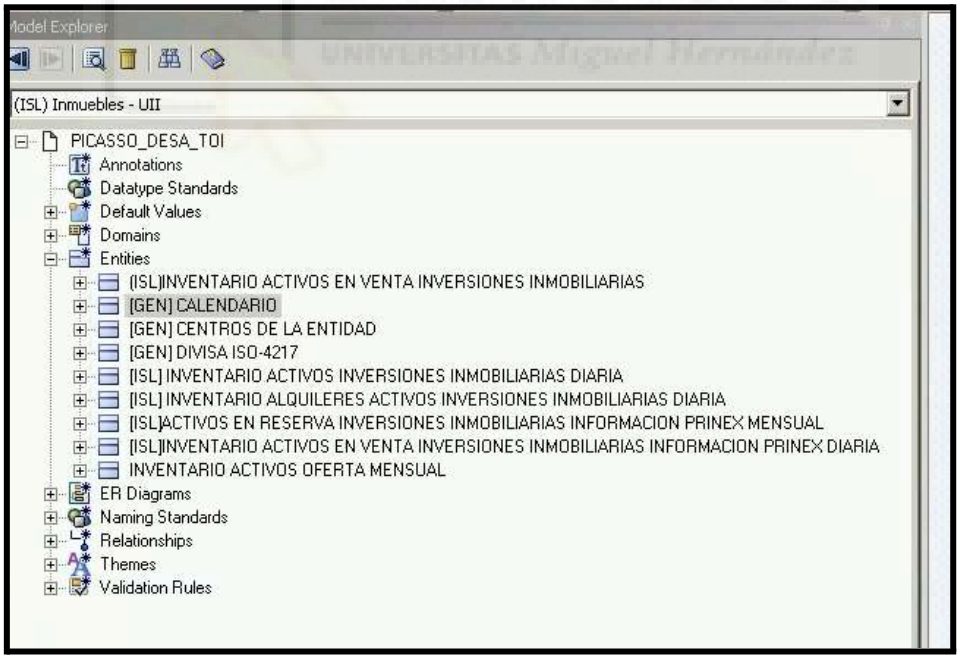

Figura 3.44 Tablas a modelar en Erwin

Cuenta con un área de trabajo en la que se podrán arrastrar las tablas a crear o modificar, en la Figura 3.45 se presenta esta situación en la que se va a modelar la tabla INVENTARIO. Como esta tabla tiene campos de dimensión como un código de tiempo identificador e importes, se añaden las tablas de dimensión CALENDARIO y DIVISA.

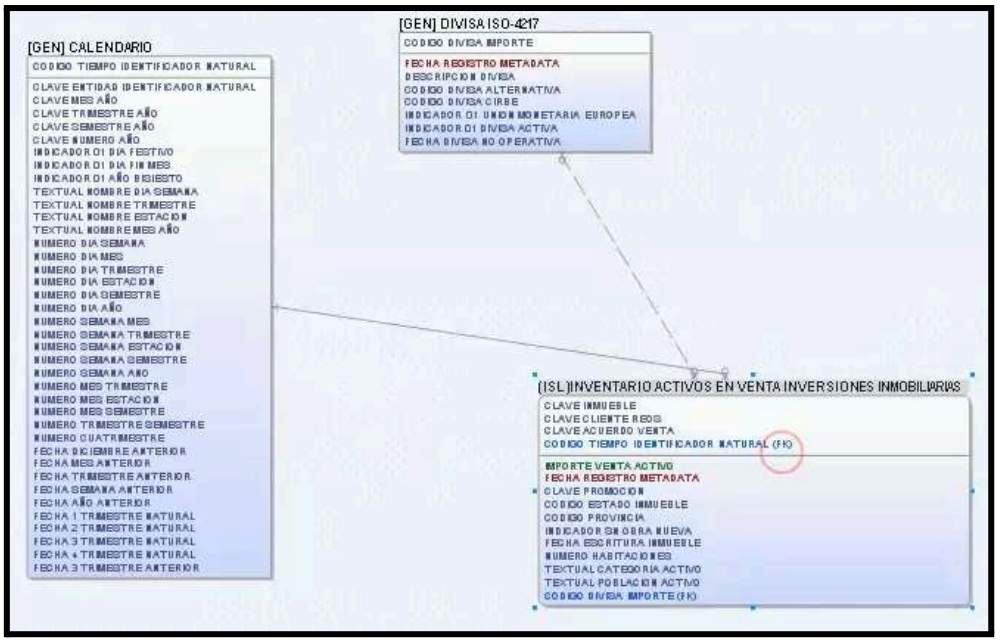

Figura 3.45 Área de trabajo Erwin

Tan solo se debe ingresar (Figura 3.46) los campos que portará la tabla bajo unas normas vistas en el apartado de normativa y buenas prácticas para conceder al Data Warehouse de una mayor normalización.

Figura 3.46 Modelado de tabla con Erwin

El uso de este programa a parte de como hoja de trabajo, también colabora en el histórico de las modificaciones realizadas sobre las tablas del almacén.

3.6.3.- Control-M[56]

Control M es una herramienta que permite automatizar flujos de trabajo a través de una interfaz de usuario intuitiva y muy fácil de entender (Figura 3.47) que sigue los colores de un semáforo para describir el estado en el que se encuentran los "*jobs*" o cajas, verde para finalizado, amarillo procesando y rojo fallido.

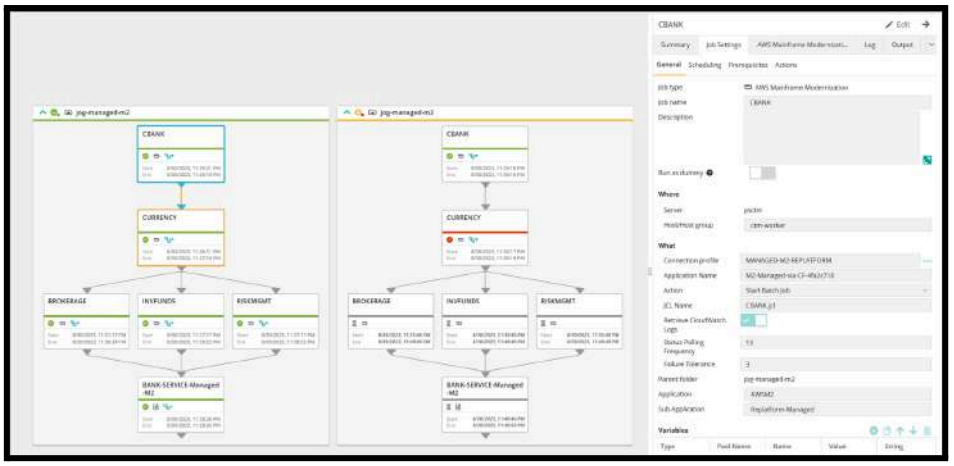

Figura 3.47 Visualización de “jobs” en control-M

Estos “*jobs*” son componentes que se pueden configurar de distintas formas según la operación, por ejemplo junto a Powercenter se pueden programar la ejecución de un Workflow que se ejecute bajo un desencadenante o que de paso a otro posterior, también agrupar la ejecución de varios de ellos para la carga del Data Warehouse en un horario en específico, de forma que con un simple vistazo podamos vislumbrar si ha habido algún fallo en los procesos. Esta última opción es la más interesante, puesto que la información que se integra en muchas tablas son a la vez el origen o fuente de datos de otra y usar una aplicación que muestre mediante diagramas el estado de los mismos, es fundamental para el diseño de carga del DW, además que permite ver la ejecución en tiempo real mejorando la optimización de los procesos junto a una sincronización con el propio depurador de Powercenter, hace que el programador no tenga que ir al Monitor de Powercenter a buscar los errores en caso de fallo haciéndolo muy útil para el mantenimiento.

Sumado a otros trabajos, como la posibilidad de configurarlos para enviar archivos o transferir archivos entre directorios emulando al FileZilla, la importación y exportación de bases de datos, la interacción con otras herramientas BI que generen informes o cumplimenten los dashboard, una vez haya terminado la carga del DW, entre otras acciones son las posibilidades que permite la automatización por parte de Control-M.

3.7.- NORMATIVA Y BUENAS PRÁCTICAS

El desarrollo de procesos ETL o modelado de la entidad no suele seguir normas ISO, puesto que son desarrollos para una empresa privada, pero sí que siguen una serie de estándares y buenas prácticas que se deben aplicar en sus desarrollos, hasta el punto de existir aplicaciones propias de verificado que analizarán nuestros procesos y en caso de no pasar las comprobaciones no dejarán la subida de estos desarrollos a entornos más importantes como preproducción o producción. Este tipo de normativa no es exclusiva de la entidad, sino que la mayoría de empresas usan alguna variante para dotar a los

desarrollos de cierta organización. Ya que esta normativa afecta al trabajo realizado, se darán unas pequeñas pinceladas sobre los puntos más importantes.

3.7.1.- Normativa Powercenter

Debido a que los procesos están desarrollados con Powercenter para normalizar e integrar sus ETLs en los diferentes entornos se deben seguir una serie de puntos:

- Nomenclaturas de objetos: Va destinada a cómo se nombran los componentes de Powercenter acompañando ciertos prefijos, es decir, "m_" (mapping) seguido del nombre de la tabla destino, resultando en m_ISLHECKPIREOSINVACTIMES, de igual manera sucederá con los workflow y las sesiones que acogerán los prefijos wkf_ (wkf_ISLHECKPIREOSINVACTIMES) o en caso de las sesiones s_ (s_ISLHECKPIREOSINVACTIMES). De forma similar para el resto de componentes se muestran algunos ejemplos en la Tabla 3.1:

Tabla 3.1 Normativa Powercenter Prefijos

Tipo de Objeto	Prefijo
Aggregator	agg_
Expression	exp_
Filter	fil_
Joiner	jnr_
Mapping	m_
Session	s_
Shorcut	sc_
Source Qualifer	SQ_
Union	u_
Workflow	wkf_

- Descripciones: Todos los componentes de Powercenter, incluidos mapping y workflows deberán tener sus descripciones debidamente completadas; resumiendo en pocas palabras, la función que desempeña ese elemento en el proceso. Esta recomendación ayuda a la compresión de procesos por parte de otros programadores (Figura 3.48).

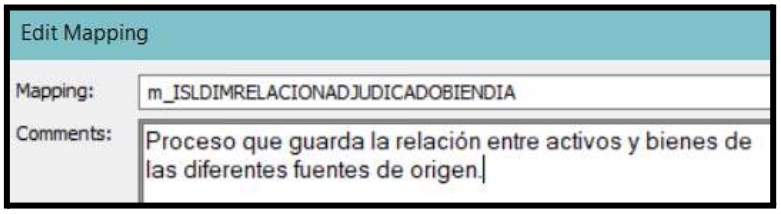

Figura 3.48 Descripción breve de la funcionalidad de un mapping

3.7.2.- Normativa modelado de tablas en Erwin Data Modeler

Aunque menos importante que la normativa seguida en Powercenter, para el modelo de nuevas entidades o ampliación de las mismas explicado en el apartado de la herramienta Erwin Data Modeler, se exige una serie de consideraciones que persiguen un orden y estructuras a la hora de crear las tablas del Data Warehouse, esta normativa consiste en los siguientes puntos:

- La notación numérica solo se modela para métricas de tipo: importes, porcentajes, secuencial etc., no por ejemplo para códigos, esta norma corresponde para un posterior mejor análisis, eficiencia en el almacenamiento u operaciones aritméticas, los códigos se suele representar mejor con una notación alfanumérica que los describan de forma única.
- Los atributos se deberán modelar con términos funcionales y siguiendo un orden. Para ello, la primera partícula del atributo deberá contener ciertas expresiones que indicarán su tipo, en cuanto al orden deberán agruparse por el tipo de atributo, las partículas y jerarquía se muestra a continuación:

    1. Claves (CLV_).
    2. Códigos (COD_).
    3. Indicadores S y N o 0 y 1 (IND_): En la entidad solo existen dos tipos de indicadores el SN (Sí o No) y el 0 y 1 (de tipo booleano).
    4. Importes (IMP_).
    5. Fechas (FEC_).
    6. Descripción (DSC_) (para todos aquellos atributos que describan conceptos).

Por último, recordar que en la fase de integración del Data Warehouse se sigue la práctica de codificar ciertos valores y la inclusión de una tabla diccionario, para clasificar estos códigos, Erwin dispone de un campo llamado "*Referencial*" en el que se informa la palabra clave con la que se guardará el código para que posteriormente se pueda buscar en el diccionario el significado de ese código, por ende, siempre que se dé de alta un código el referencial también deberá ser documentado.

Capítulo 4
Metodología y resultados

4.1.- PLANIFICACIÓN DEL PROYECTO

En la planificación de cualquier diseño de software existen fases que se basan en algún ciclo de vida, y el desarrollo del almacén de datos no es distinto. Esta tecnología se refiere a las etapas que un objeto o sistema atraviesa desde sus inicios hasta la finalización del mismo. Proporciona una estructura que ayuda a planificar y gestionar los recursos de manera eficiente, facilita la identificación de problemas en fases tempranas y permite una mejora y evaluación continua del proceso. A su vez, cada fase incluye diferentes actividades específicas de documentación y de controles de calidad, que en una entidad privada financiera se acentúan ya que se debe dejar constancia de una traza para el cumplimiento de auditorías tanto externas como internas. De igual forma, los mismos datos también tienen su propio ciclo de vida ya que cruzan una serie de etapas o fases. En este apartado se describe la evolución de estas etapas.

4.1.1.- Ciclo de vida del Data Warehouse[57]

El ciclo de vida del Data Warehouse sigue las fases para el desarrollo de un software normal, es decir, una toma de requisitos, un desarrollo y un mantenimiento posterior, pero con ciertos puntos adaptados al repositorio de datos. La figura 4.1 muestra las etapas seguidas que se describen más adelante:

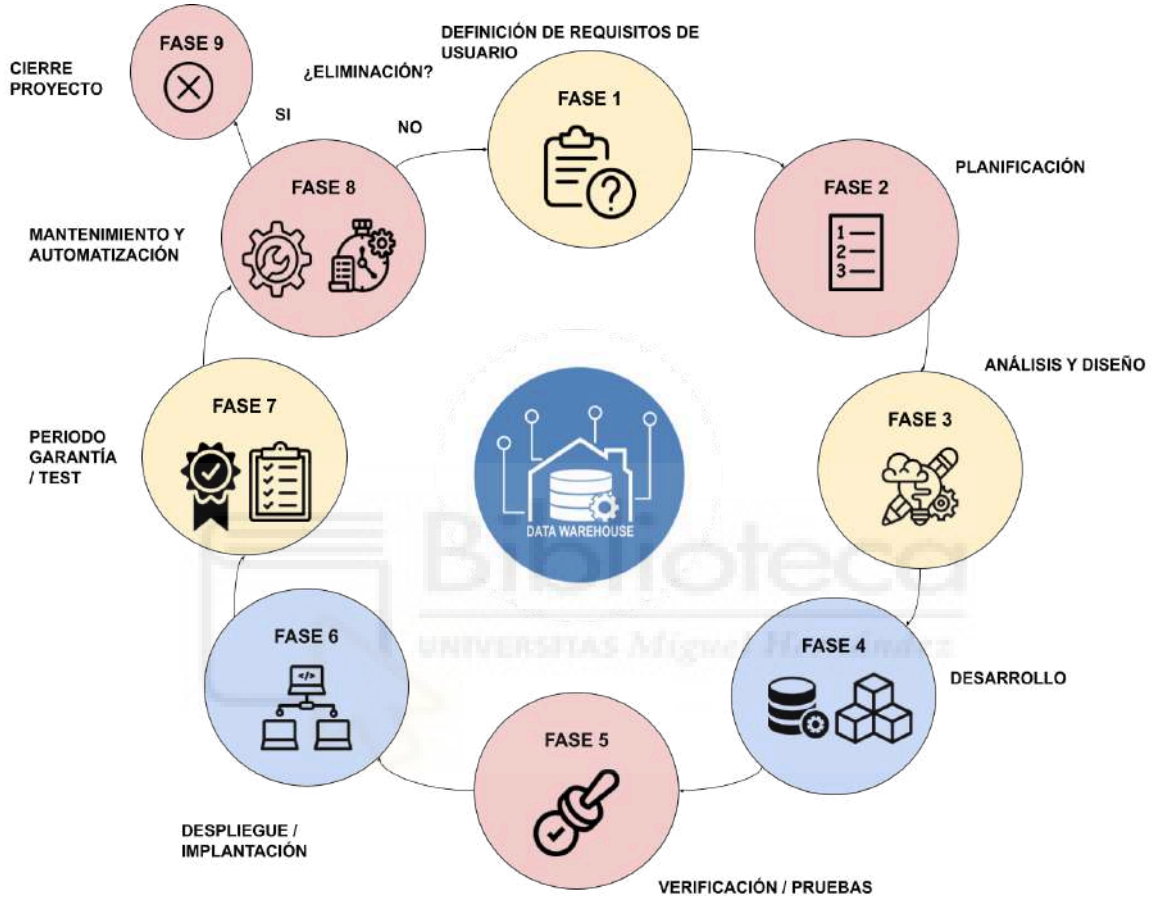

Figura 4.1 Ciclo de vida del Data Warehouse

Fase 1: Definición de requisitos/ Estudio de requisitos y Planificación

En esta fase se hace un análisis de requisitos de los usuarios, sus necesidades y expectativas. También se especificarán todos los requisitos técnicos como los datos que se incluirán y las fuentes del DW, además de limitar el alcance del proyecto. En esta etapa podemos tener entrevistas con el usuario y en caso de usuarios más técnicos recoger memorias o documentación más técnica.

Fase 2: Planificación

Con los datos de la definición de requisitos en la mano, la planificación se centrará en realizar una estimación para completar los objetivos, apoyada por diagramas de tareas (Tabla 4.1). En esta fase se investigarán los riesgos existentes que puedan hacer peligrar el

desarrollo del mismo, además de crear informes de seguimiento o documentación para tener registrados los cambios en los requisitos que se hagan con posterioridad.

Tabla 4.1 Diagrama Gantt tareas realizadas

		2019	2020		2021		2022		2023	
	Módulos		S1	S2	S1	S2	S1	S2	S1	S2
Proyecto Inmuebles	Estacionamiento de datos contables y comerciales en el ecosistema de la entidad.									
	Integración de la entidad: INVENTARIOS UII									
	Integración de la entidad: KPI REOS									
	Integración de la entidad: BIENES									
	Integración de entidades secundarias: - ENTIDAD ANEJOS - ENTIDAD ALQUILER - ENTIDAD GEOLOCAL									
	Estacionamiento vía fichero de Datos Comerciales									
	Desarrollo Data Mart Salesforce									
	Desarrollo Data Mart de Garantías									
	Integración de la entidad : TASACIONES									
	Integración de la entidad : GASTOS									
	Integración de datos de Demandas									
	Validaciones									

Tareas propias | Tareas de equipo | Tareas de terceros

En la tabla 4.1 se muestra la planificación de las tareas realizadas en el modelo de inmuebles, tanto propias como en colaboración con otros miembros del equipo o las tareas realizadas por terceros.

Fase 3: Análisis y diseño

En esta etapa se diseñan los modelos funcionales y analíticos, el diseño de las ETL como de los cubos OLAP, se desarrollan requisitos técnicos de aprovisionamiento y procesamiento. Es decir, se traducen los requisitos del usuario a un modelo lógico para traducirlo a un diseño físico que documente la construcción del DWH.

Fase 4: Desarrollo

Se refiere al desarrollo de todos los procesos ETL necesarios para la construcción del almacén, además de la carga del modelo dimensional de los cubos OLAP con datos iniciales de desarrollo.

Fase 5: Verificación / Pruebas

Se toma un muestreo de los datos iniciales y se verifica en diferentes tipos de pruebas como: pruebas unitarias de carga y enriquecimiento, de valor, de contraste, coherencia, entre otros, probando que la calidad de los datos está en consonancia con los requisitos recogidos y permite que los clientes hagan pruebas de usuario (UAT[64]).

Fase 6: Despliegue / Implantación
Implantación del almacén e integraciones de datos en el entorno de producción. Se asegura que los usuarios tengan acceso y puedan utilizar el sistema, además de informarles sobre cómo utilizar el DW y las herramientas OLAP de consulta.

Fase 7: Periodo y garantía / Test
Etapa intermedia destinada a estabilizar el DW en el entorno de producción, perfilar errores y añadir nuevas necesidades que hayan surgido, también sirve para dar soporte a usuarios rezagados.

Fase 8: Mantenimiento y automatización
Se implementan procesos regulares que actualicen el almacén de datos en la periodicidad requerida, además se monitorean los procesos y se solucionan fallos ocasionales. Igualmente se documenta y se registran los cambios, para tener constancia de que las modificaciones realizadas de mejoras o evoluciones no afecten al mal funcionamiento del repositorio y de serlo, poder volver al estado original.

Fase 9: Cierre Proyecto
Esta fase solo queda reservada en caso de la retirada del DW o partes del mismo, ya sea porque no cumplen con los requisitos del negocios o porque se tengan que migrar los datos a un nuevo sistema.

4.1.2.- Ciclo de vida de los datos[58]

El ciclo de vida del dato define las etapas por las que pasan los datos, ayudando a saber donde aplicar seguridad, calidad y aplicar diferentes políticas, es decir contribuye a conocer dónde poner la atención, para identificar las carreteras que utilizan los datos y ver comportamientos “raros” o atípicos. La diferencia con el ciclo de vida del DW que se centra en el diseño, desarrollo, implementación y mantenimiento de una infraestructura de almacenamiento de datos, es que este se enfoca en la gestión de datos individuales además de los procesos y actividades relacionados a ellos, algo esencial puesto que los datos del DW no son temporales sino perennes en el tiempo y necesitan un tratamiento añadido o especial (Figura 4.2).

Fase 1: Creación
Los datos son generados o recopilados de diferentes fuentes primarias u operacionales, o simplemente pueden provenir del propio sistema.

Fase 2: Almacenamiento
Esta etapa hace referencia a todas las bases de datos o repositorios donde son almacenados.

Figura 4.2 Ciclo de vida del dato

Fase 3: Uso

Los datos son accesibles para su consulta y utilizados para análisis, informes, toma de decisiones y otras operativas.

Fase 4: Distribución

Fase en la que los datos se comparten entre diferentes sistemas y departamentos, es decir, el dato se democratiza y está disponible para cualquier usuario que lo necesite, siempre respetando las políticas de seguridad y privacidad.

Fase 5: Archivado

Cuando los datos ya no son de un uso activo o periódico, pero deben conservarse por motivos históricos, legales o de cumplimiento, es necesario crear procedimientos de archivado que liberen los sistemas de estos datos en desuso.

Fase 6: Eliminación

En el momento en que los datos ya no tienen valor o cuyo periodo de guardado ha finalizado se eliminan de forma segura o pasan a una zona de "Almacenamiento en Frío", descargándolos de los sistemas para descartarlos definitivamente.

4.2.- CAPTURA DE REQUISITOS DEL NEGOCIO

En este apartado se van a describir cuántos usuarios intervienen en el ecosistema del DWH de inmuebles, los roles que ejercen y los casos de uso que pueden desempeñar.

4.2.1.- Roles de usuarios

Los roles en un Data Warehouse se refiere a las diferentes responsabilidades y funciones que toman sus usuarios para diseñar, construir, mantener y utilizar el almacén. En la siguiente tabla se muestran los roles más importantes que han participado para la construcción, mantenimiento y explotación del repositorio de inmuebles.

Tabla 4.2: Roles de usuarios

ROL	USUARIO	DESCRIPCIÓN
Ingeniero de datos	Externo y de la propia entidad.	El ingeniero de datos es la base de desarrollo y análisis del almacén, sus responsabilidades abarcan desde la construcción y mantenimiento hasta la automatización del sistema.
Jefe de Proyecto Inmuebles	Propia entidad	En el caso del modelo de inmuebles que se compone de un grupo pequeño, puede abarcar todas las funciones del ingeniero de datos, pero se enfoca en la planificación, seguimiento y toma de requisitos de los usuarios.
Desarrollador BI	Externo y de la propia entidad.	Se encarga de transformar los datos en información visual comprensible para la toma de decisiones.
Administrador de base de datos	Externo y de la propia entidad.	Rol que asegura que el sistema esté bien estructurado, optimizado, además de ser el encargado de crear sus tablas y administrar las bases de datos que las contienen.
Operador de Soporte	Externo y de la propia entidad.	Es el encargado de hacer guardias y arreglar fallos en los procesos ETL, rol crítico puesto que si los procesos automatizados que se ejecutan se interrumpen la información no estará disponible.
Usuario de Negocio	Propia entidad	Usuario final que explota el DW y usa los datos contables para la toma de decisiones financieras y operativas.
Usuario Riesgos	Propia entidad	Usuario final que analiza los datos recogidos en el Data Mart de Garantías que se informa a partir del Data Warehouse de inmuebles.
Usuario Comercial	Externo y de la propia entidad.	Uso de los paneles de mando para la gestión de inmuebles y la ejecución de órdenes de saneamiento para su mantenimiento.
Usuario Abogado Gestión Inmobiliaria.	Propia entidad	Este usuario pertenece a la parte legal del banco que visualizará las demandas en específico de los inmuebles.
Administrador de Seguridad	Externo y de la propia entidad.	Rol que afianza que solo los usuarios autorizados tengan acceso a los datos y sistemas de la entidad.

4.2.2.- . Casos de uso de los usuarios

En este punto se van a describir los diferentes casos de uso que pueden desempeñar los roles mencionados, cabe destacar que estos casos de uso solo son los más importantes que están relacionados con el DWH de inmuebles, es decir, a menudo los usuarios tendrán otro tipo de casos que no se verán en este punto, seguramente el usuario comercial atenderá a clientes y el administrador de seguridad desempeñará otras funcionan más allá de proporcionar permisos, por otro lado habrá usuarios que como el ingeniero de datos o el jefe de proyectos de inmuebles puedan realizar trabajos que van más allá de su rol, pero al tratarse de un grupo pequeño algunos casos de uso se acaban sobreponiendo entre los dos roles cuando existe alta carga de trabajo. Los roles de la tabla 4.2 se podrían clasificar en tres grupos: (Figura 4.3)

- Grupo de desarrolladores: En este grupo entrarían el Ingeniero de datos, el Jefe de Proyecto (manera opcional) y el Desarrollador BI.
- Usuarios finales: Como los roles de Abogado, Comercial, Riesgos y de Negocio que serán los perfiles que explotarán el DW y sus Data Marts.
- Grupo Mantenimiento: Que velarán por el almacén, en él se puede englobar al Administrador de BBDD, al Operador y al Administrador de Seguridad.

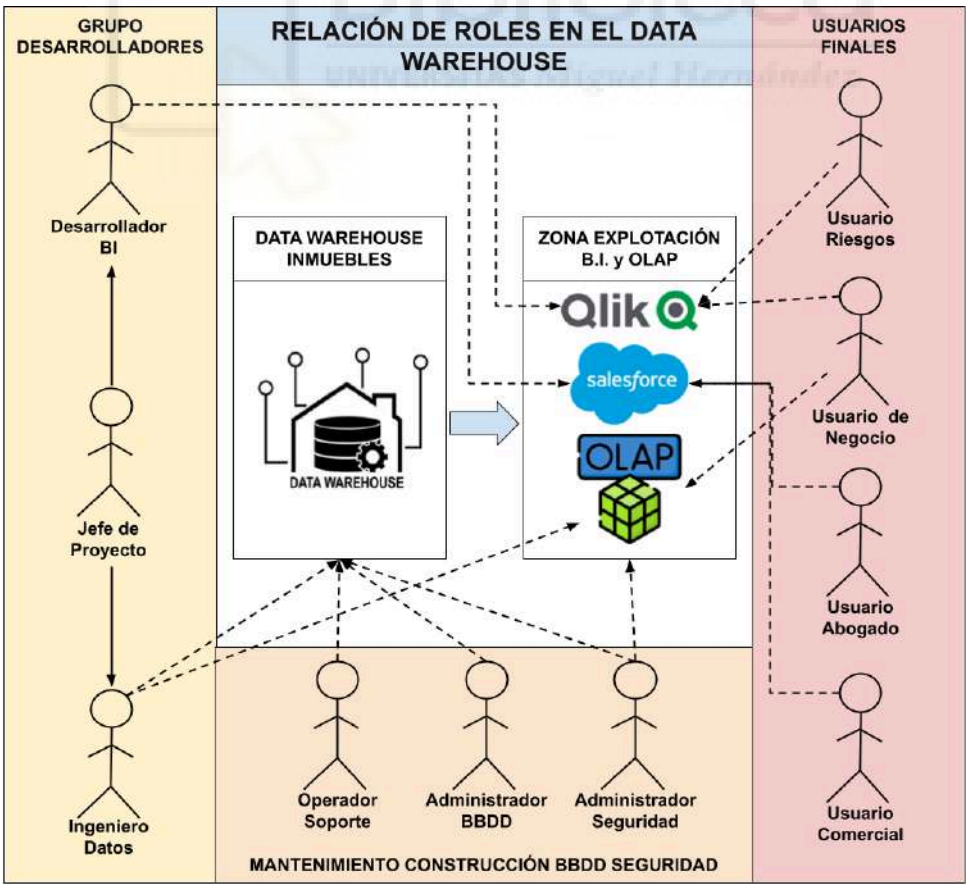

Figura 4.3. Relación de los roles en el modelo de inmuebles

Usuarios finales

Son los usuarios que van a explotar la información, en estos momentos existen cuatro tipos:

- El *usuario de negocio*: pertenece al departamento de contabilidad de la entidad, es el encargado de hacer los balances a partir de la información enviada en los informes diarios que recibe, al ser un usuario un poco más técnico también realiza consultas OLAP sobre el DW ya sean hechas por él o proporcionadas por el grupo de inmuebles. Cada día revisa ciertos campos KPIs importantes y envía un informe con las observaciones encontradas (Figura 4.4).

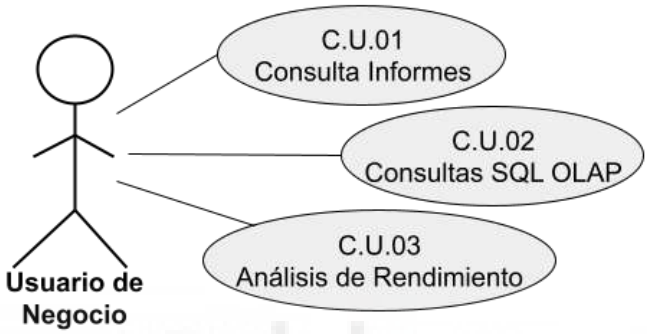

Figura 4.4 Usuario de negocio

- El *usuario de riesgos*: es un usuario final que nace del Data Mart de garantías donde se calcula el importe final del coste de un inmueble desde que entra por costas judiciales, mientras está en el inventario con los gastos de mantenimiento hasta otros costes en la venta respecto a la recuperación de deuda del mismo. Este usuario tendrá acceso a un reporte en la herramienta BI Qlik que leerá sobre el Data Mart de Garantías que se ejecuta mensualmente, además de poder hacer consultas OLAP sobre el propio DM. (Figura 4.5)

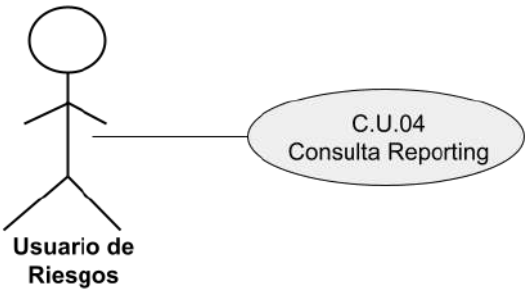

Figura 4.5 Usuario de Riesgos

- El *usuario comercial*: puede ser tanto interno como externo a la entidad, se encarga de gestionar el "saneamiento" (se refiere al mantenimiento físico de los inmuebles se verá en el apartado 4.4 - Implementación) , el monitoreo y administración de los

mismos, adicionalmente podrá descargar archivos como documentos del catastro, facturas, etc. (Figura 4.6).

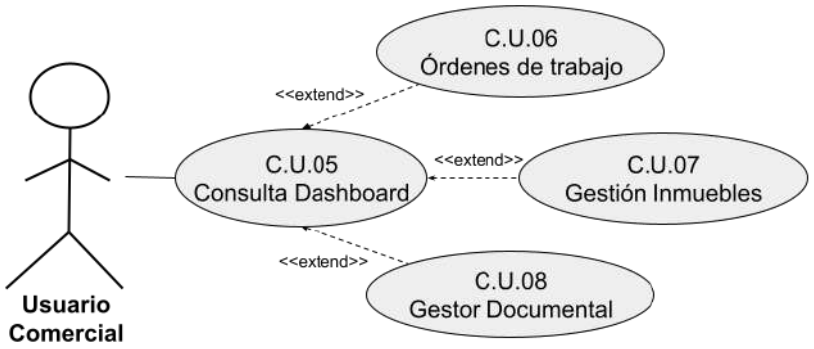

Figura 4.6 Usuario Comercial

- El *usuario abogado*: especializado en la gestión de inmuebles que corresponde al departamento legal de la empresa, podrá ver las demandas interpuestas relacionadas con los inmuebles y los intervinientes que participan en la misma, además de poder descargar documentos de importancia desde el panel de mandos al igual que el usuario comercial. (Figura 4.7).

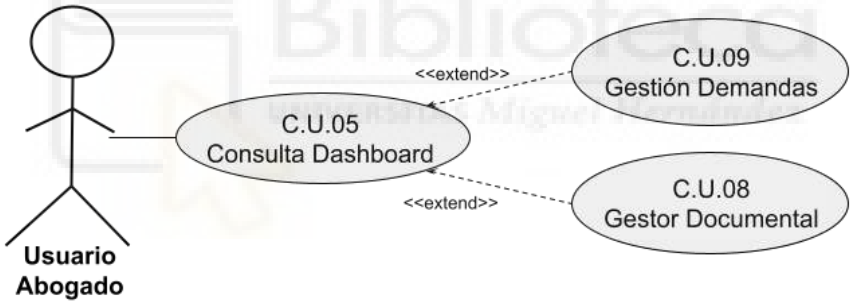

Figura 4.7. Usuario Abogado

En cuanto a la relación *<<extend>>* que se utilizará en diferentes diagramas de casos de uso UML, sirve para modelar un comportamiento opcional que existe bajo ciertas condiciones a partir de un caso de uso base, por ejemplo dependiendo del usuario final que consulte el dashboard podrá acceder a un caso de uso u otro.

Grupos de desarrollo

Este grupo va a ser el motor tanto de la parte *front end* como *back end* de los datos, conformará el grupo de inmuebles en la parte del *back* que en nuestro caso será un jefe de proyecto y un ingeniero de datos (excepcionalmente puede haber dos ingenieros en las etapas de estrés del proyecto como se verá en el apartado 4.4 de implantación). En cuanto al *front* está el rol del desarrollador BI que trabajará en la visualización de los datos, destacar que este rol también tiene su respectivo Project Manager BI o Jefe de Proyectos BI, pero se omitirá por no ser el centro de estudio de este trabajo y no duplicar

información. Con respecto a los casos de uso anteriores se puede apreciar por ejemplo en la consulta del dashboard que ya aparecen estos actores, se debe a la validación y pruebas por parte del desarrollador BI, y al mantenimiento y calidad de los datos de los cuadros de mandos, por parte del grupo de inmuebles, es decir, si un dato esperado no es representado correctamente o es erróneo, la responsabilidad del mismo recae en la parte del *back*, puesto que los datos que se visualizan son fuente directa del Data Mart de Salesforce o Garantías, por eso, también tienen acceso a las consolas, a continuación se hará un breve resumen de estos roles:

- *Desarrollador BI*: Este rol es el que desarrollará los cuadros de mandos en Salesforce y Qlik, para los usuarios finales. Enmarca varias destrezas que serán repetidas a su vez por el Ingeniero de datos. (Figura 4.8)

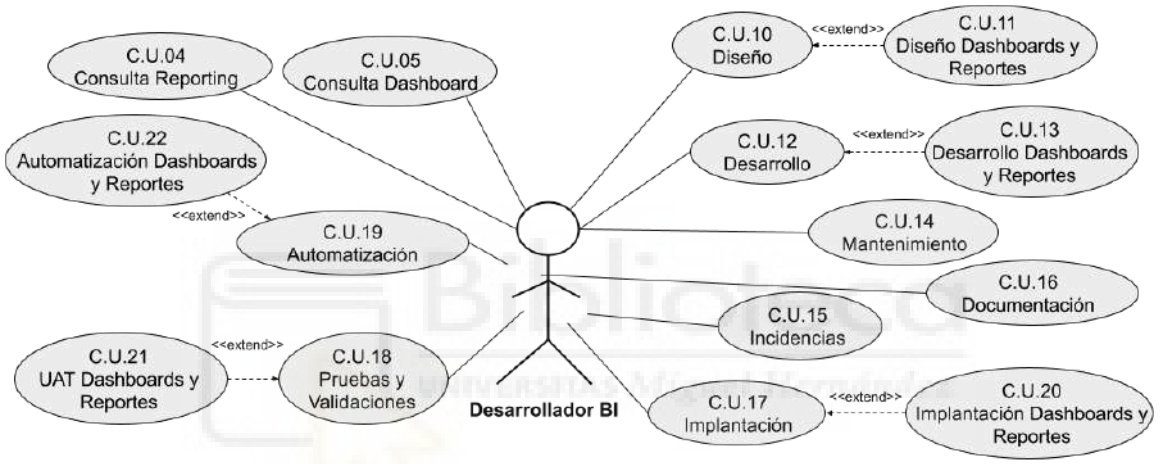

Figura 4.8 Rol Desarrollador BI

- *Ingeniero de datos*: Corresponde a mi rol, y sus casos de uso van relacionados con el tratamiento de los datos. Hereda los casos de uso comunes del desarrollador BI pero cambia a los específicos de su rol en cuanto a Diseño, Desarrollo, Implantación, pruebas y Automatización (Figura 4.9)

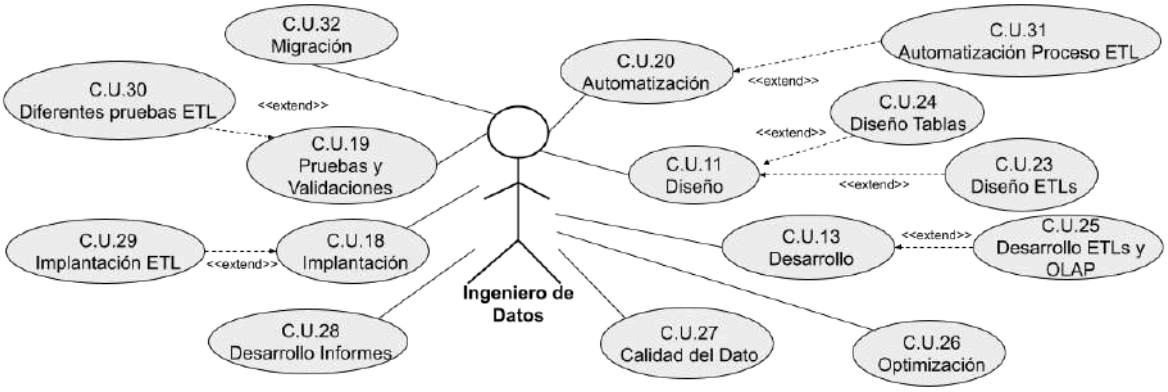

Figura 4.9 Rol Ingeniero de Datos

- *Jefe de Proyecto*: Responsable del Ingeniero de datos, sus casos de uso están relacionados con el análisis de requisitos de los clientes y velar por la correcta evolución del proyecto, aunque opcionalmente puede heredar los casos de uso del ingeniero. (Figura 4.10)

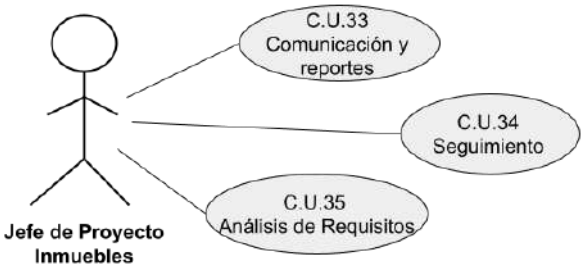

Figura 4.10. Rol Jefe de Proyecto

Grupos de Mantenimiento

Roles que darán soporte a todas las etapas del Data Warehouse, nos podemos encontrar:

- Administrador de Seguridad: Proporciona permisos de acceso al resto de usuarios desde aplicaciones hasta bases de datos (Figura 4.11).

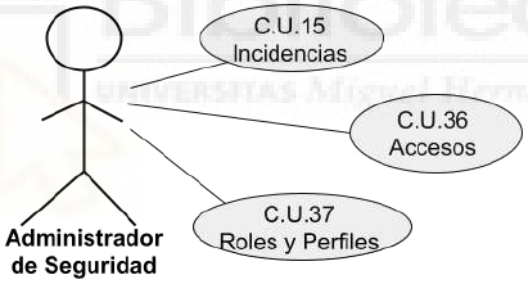

Figura 4.11 Rol Administrador de Seguridad

- Administrador de BBDD: Gestiona la estructura de todos los elementos de almacenamiento que forman parte del DWH (Figura 4.12)

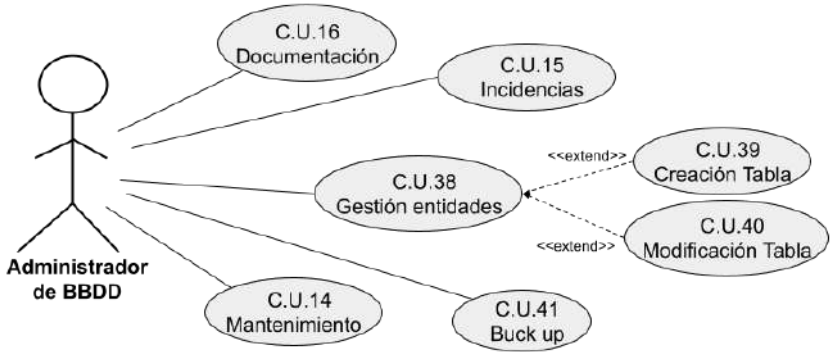

Figura 4.12 Rol Administrador de BBDD

- Operador Soporte: Encargado de monitorear los procesos críticos que se ejecutan automáticamente y la resolución de incidencias de los mismos. (Figura 4.13)

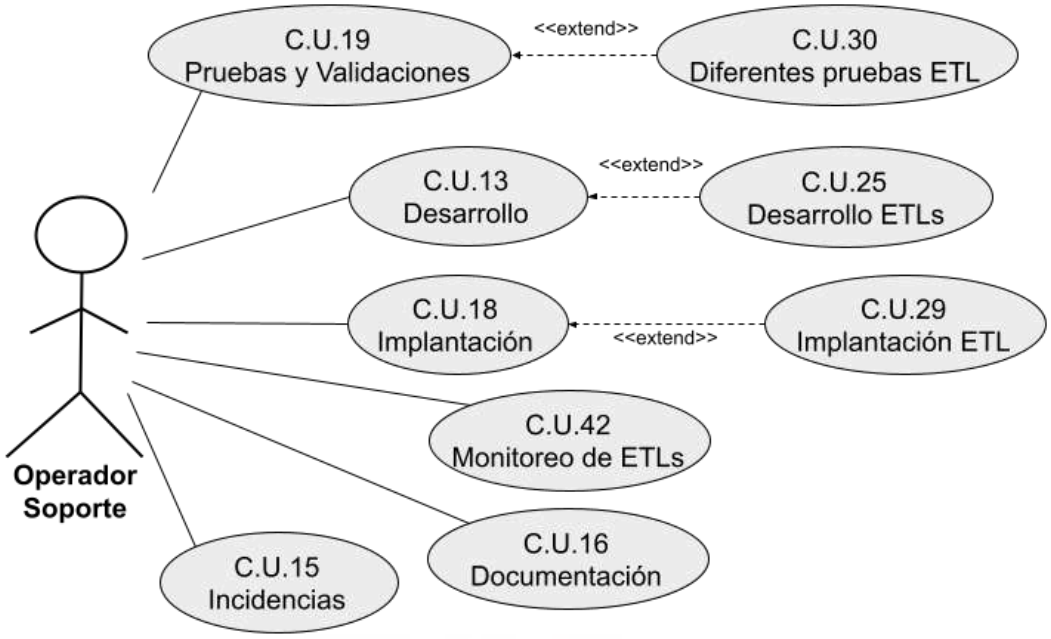

Figura 4.13 Rol Operador Soporte

En el Anexo I se puede consultar una descripción detallada de cada uno de los casos de uso mostrados en las figuras anteriores, dichas descripciones se encuentran en la forma de tablas/plantillas estándar empleadas comúnmente en ingeniería del software.

4.3.- DISEÑO

En este apartado se afrontará el diseño, las relaciones y el flujo seguido para la carga de datos en las tablas del almacén y en sus Data Marts desde sus orígenes.

Comenzando por el DW en la figura 4.14 podemos observar el modelo conceptual de carga del almacén de inmuebles y cómo consta de cuatro fuentes origen (se explicará más extensamente en el siguiente apartado 4.4). La BDD de la inmobiliaria corresponde a datos contables y comerciales, y que se interpretará como fuente principal de los datos de inmuebles, los archivos recibidos por la inmobiliaria de datos comerciales, complementarán y añadirán información comercial a la fuente primaria, el Data Warehouse de deuda que marca los activos con el acuerdo de deuda al que pertenecen y por último las inversiones inmobiliarias que llegan desde la propia entidad al igual que el propio DW de deuda.

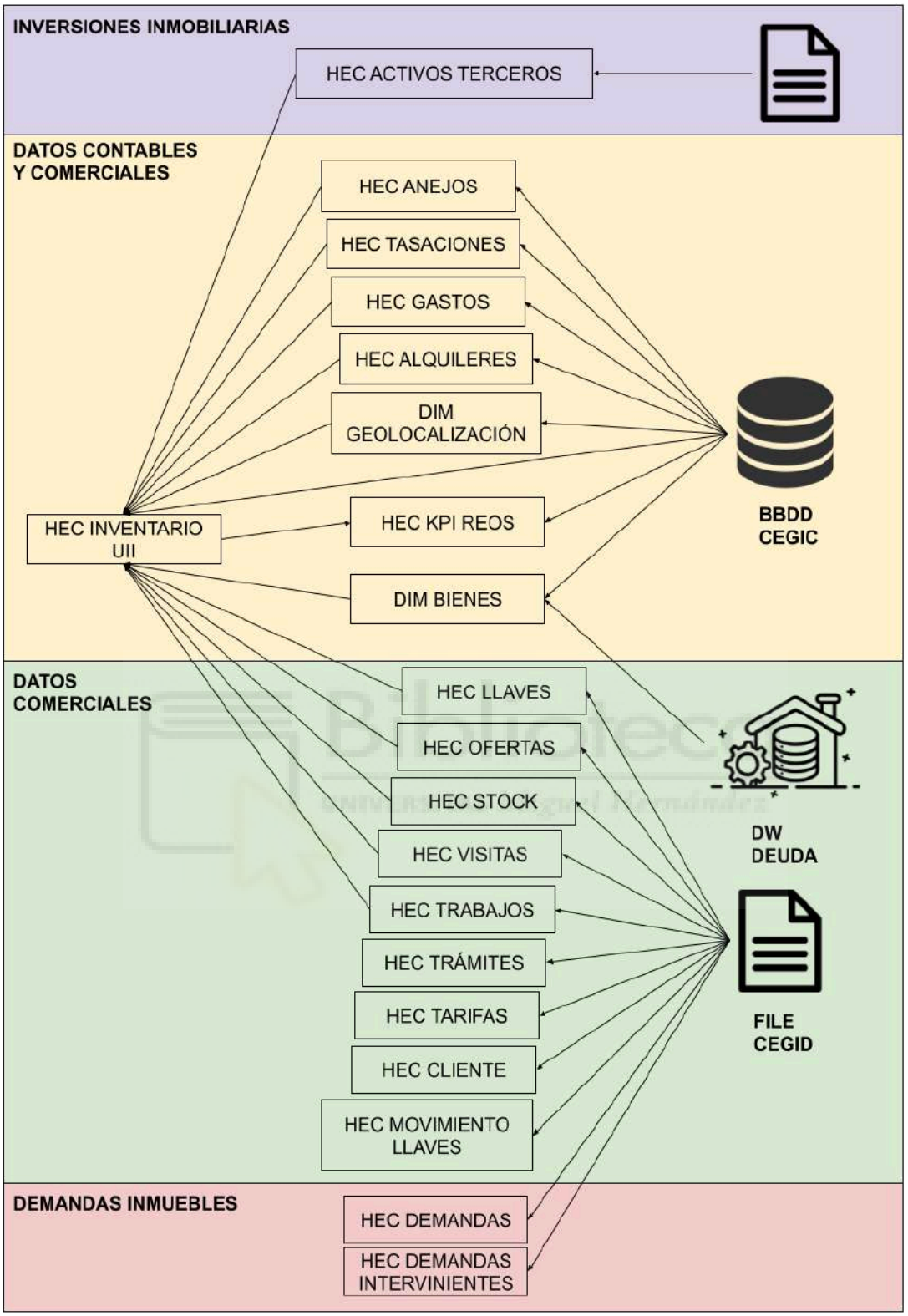

Figura 4.14 Modelo Conceptual del flujo de datos del Data Warehouse de inmuebles.

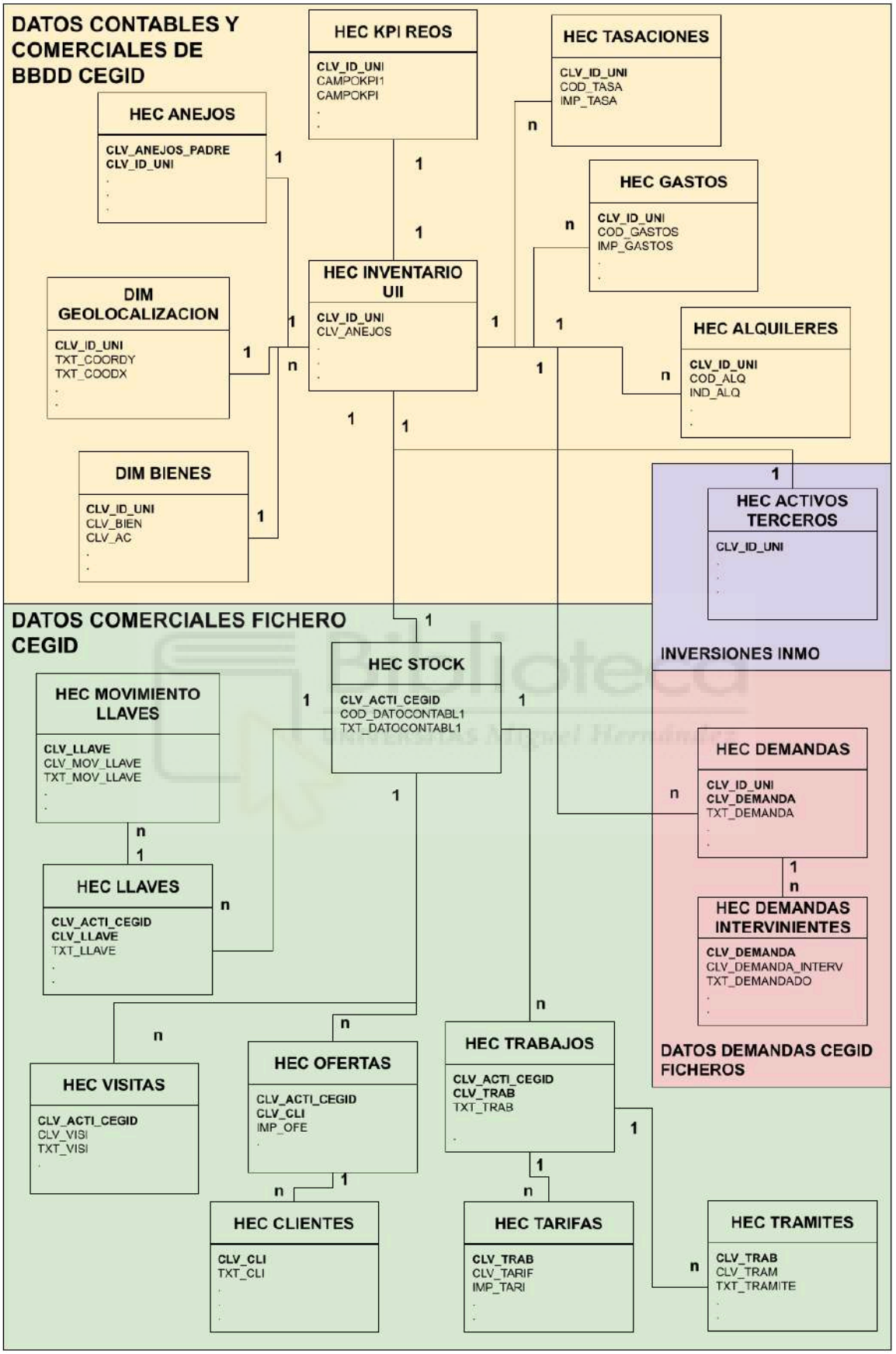

Figura 4.15 Modelo relacional del Data Warehouse de inmuebles

Todas las tablas que se presentan en esta Figura 4.15 pertenecen al propietario TI, (tablas integradas), dentro de este propietario se empieza a intuir que estamos ante un modelo de estrella con alguna variación, pues la tabla de “integración de inventarios” es donde confluye el resto de entidades, y las que no lo hacen es porque son detalles muy específicos de otras, por ejemplo el movimiento de las llaves que detalla el traslado de llaves que residen en la tabla homónima “llaves”, siendo la tabla “kpi reos” la única que se carga a partir de la de “inventarios”, pues en ella se calcularán los campos más importantes a nivel contable y la división de los activos en inventarios, además será el punto de partida para OLAP, reportes e informes y los Data Mart de Salesforce y Garantías. Resaltar que la carga de la tabla “inventarios” debe ser la última en la automatización seguida de “kpi reos".

Respecto a la Figura 4.15 representa el modelo relacional del almacén pero con ciertos matices. Como se anticipó en el capítulo tres, el almacén de inmuebles sigue un esquema híbrido de estrella con características del modelo de copo de nieve, ¿cómo se traduce esto en el repositorio?, como observamos la tabla principal de hecho es la de “inventarios integración” y siguiendo la teoría del modelo de estrella el resto deberían de ser tablas de dimensión, sin embargo no es así, sino que son lo que se llaman tablas de hecho agregadas o detalladas, la función de presentar el modelo conceptual de flujo de la Figura 4.14, es hacer entender que el resto de tablas, parte de su información más importante se integra en la tabla de hecho principal (por eso se debe de ejecutar la última), por ejemplo la tabla de “tasaciones” carga la última fecha e importe de tasación y otros campos, dejando el resto de tablas secundarias para consultas más detalladas de ese tema y así conseguir una desnormalización parcial, la excepción pasa por ejemplo con la tabla de “movimientos de llaves” que complementa a la tabla de “llaves” en los datos comerciales, este tipo de tablas son muy pocas en el almacén y lo más importante es que su uso es muy limitado.

Cabe preguntarse cómo contribuye el modelo de copo de nieve al repositorio. Se aprecia que todas las entidades del repositorio cuentan con una misma clave id única, es decir, si queremos unir la tabla de “tasaciones” con la de “gastos”, se puede perfectamente sin pasar por la principal de “inventarios”. Esta característica no es casualidad, el diseño de base de datos donde todas las tablas comparten una misma clave primaria de unión se conoce como modelo de identificador único, facilita la integración y simplifica la relación entre diferentes entidades. Este método es común en Data Warehouses con esquemas de copo de nieve combinado con modelos en estrella. Como vimos en el capítulo anterior una de las finalidades del DW era simplificar las consultas SQL, sin tener que recurrir a tablas intermedias de tipo diccionario para las uniones, pues se realizan sobre una clave primaria común. Este diseño se consigue introduciendo la tabla que contiene esta clave primaria en todas las integraciones y usando este campo como clave primaria en conjunción con otras, por lo tanto, aunque en la Figura 4.15 solo se muestran las relaciones con el punto central que es la inventario para no saturar la figura de conexiones, pero eso no significa que el resto de tablas no se pueden relacionar entre ellas sin tablas intermedias. En cuanto a los datos comerciales recibidos por fichero de CEGID su clave activo CEGID es igual a la

clave id única, de igual manera que lo explicado se ha decidido representar la tabla de "stock" como un punto auxiliar a modo de esquema de constelación ya que ella residen los activos que son recibidos en datos comerciales y que existen en la tabla de "inventarios", además de portar la mayor cantidad de campos recibidos por la inmobiliaria de datos comerciales, pero en ningún momento se debe pasar por ella, para interconectar el resto de datos comerciales ni se integran datos a partir de otras integraciones, como en la de "inventarios" es solo una forma de representar las relacionas de forma limpia, en otras palabras, la tabla de "inventarios" se puede unir directamente con la de "ofertas" para saber el número de ofertas recibidas en un determinado inmueble, al igual que con el resto de tablas menos con las que detallan alguna información como la mencionada anteriormente de las "llaves" y sus "movimientos".

El último punto recae sobre las tablas de dimensión, ver en qué se diferencian ahora de las de hecho, detalles o agregadas. Las tablas de dimensión van a ser utilizadas para datos estables, y por la naturaleza de sus datos la historificación de los mismos no representa ningún tipo de interés. Para estos ejemplos se han seleccionado dos entidades muy representativas: la dimensión de "bienes" y la dimensión de "geolocalización", la primera lleva los acuerdos de deuda que están ligados a los inmuebles, el estudio de estos acuerdos aunque es muy importante porque hacen de traza de la procedencia del inmueble a la hora de historificarlos no representan ninguna ventaja, no van a cambiar (a no ser que se cierre la oficina donde se concedió el préstamo y se traspasen a otra). La segunda es más representativa todavía, corresponde a las coordenadas de los inmuebles, hacer un histórico de su localización no tiene ningún sentido, puesto que los inmuebles no van a cambiar de sitio.

El Data Mart de Salesforce (Figura 4.16) no tiene apenas complejidad:

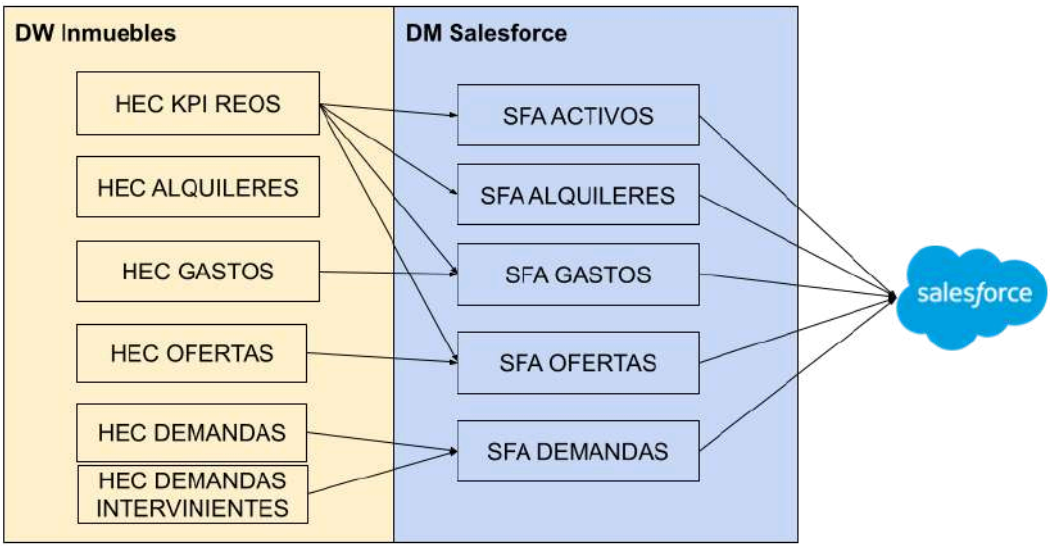

Figura 4.16 Modelo Conceptual del flujo de datos Data Mart Salesforce

Las entidades que lo conforman cargan los datos a partir de las tablas del DW de inmuebles sobre el tema en particular que trate la tabla en cuestión (la hecho de gastos con la SFA de gastos y sucesivamente), la tabla "kpi" al tener los mismos campos que la de "inventarios", además de otros calculados, cargará la entidad "sfa de activos" que es la más importante del modelo y servirá como filtro para el resto, como se verá en el siguiente apartado de implementación.

Por último el Data Mart de Garantías (Figura 4.17 y 4.18) se puede llegar a considerar un DM compartido pues su carga viene del DW de deuda asimismo como el DW de inmuebles:

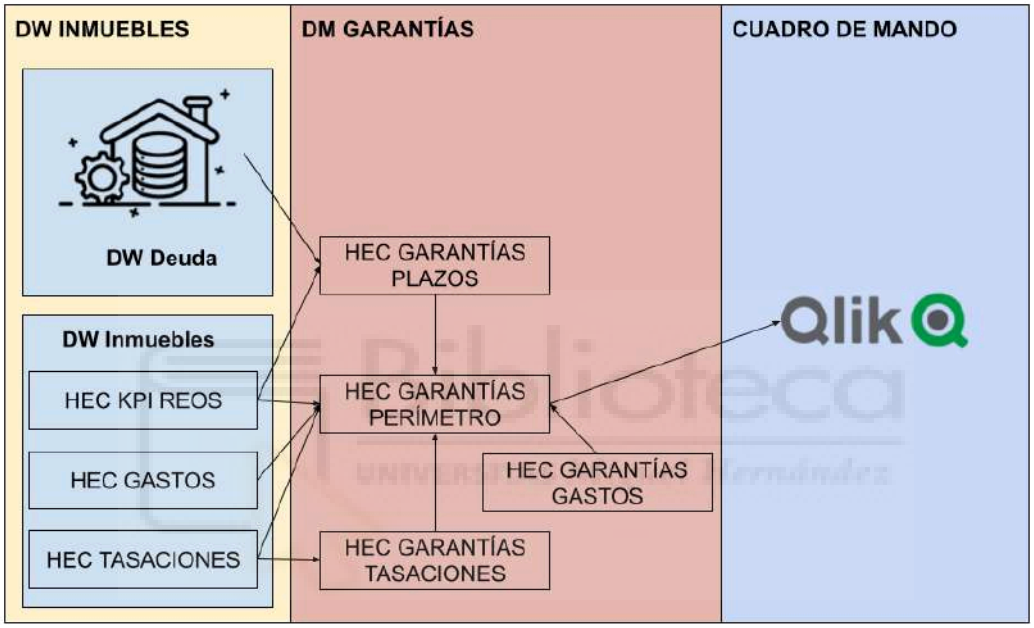

Figura 4.17 Modelo Conceptual del flujo del Data Mart Garantías

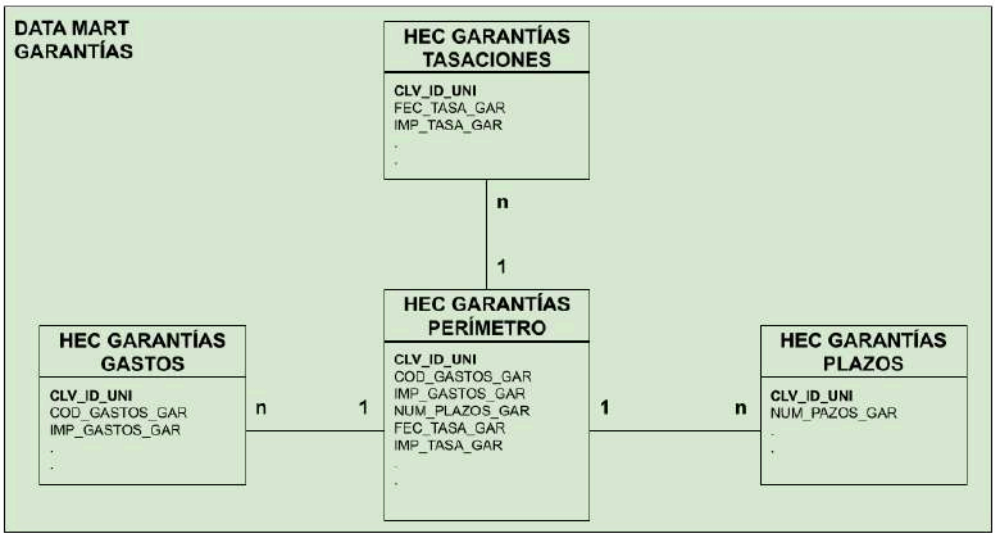

Figura 4.18 Modelo Relacional del Data Mart Garantías

Su modelado es igual al Data Warehouse de inmuebles, un modelado en estrella usando características del modelo de copo de nieve, los datos importantes de las tablas de detalle se comprimen en la tabla central ("*perímetro*"), que será de dónde lea el reporte QLik.

4.4.- IMPLEMENTACIÓN

Como se ha ido presentando en el resto de capítulos de este trabajo donde se describe cómo tratar y organizar los datos para darles un uso útil en la toma de decisiones, en específico para el ámbito inmobiliario en una entidad financiera. En este apartado de implementación, se van a recorrer las etapas más importantes en las que me he visto implicado, tomando como referencia el diagrama Gantt de la tabla 4.1, para recrear la fabricación del almacén de datos de inmuebles y sus usos, al ser mi entrada en el proyecto en etapas muy tempranas, se podrá usar este trabajo como un caso práctico en un ámbito empresarial real, de la construcción y explotación de un sistema informacional (DW), puesto que he tenido una participación completa en todas las fases del Data Warehousing , ya sea tanto en su creación como su modificación y ampliación.

Se mostrará desde un punto de vista de análisis, diseño y desarrollo las diferentes formas que ha ido adoptando el repositorio a lo largo del tiempo, focalizando la mirada en los puntos importantes. Cabe destacar que al ser un trabajo realizado en años, no se enseñará toda la implementación realizada sino que se seleccionará casos de interés que sean sustanciales en el tema que estamos abordando, omitiendo o mostrando brevemente los puntos con un menor aporte por ser repetitivos, de poco interés o que no se hayan realizado por mi persona. También indicar que lo expuesto serán arquitecturas y desarrollos para el tema principal de este trabajo, ofuscando datos y nombres importantes de la entidad por motivos de seguridad y privacidad.

4.1.1.- Introducción y antecedentes

En este apartado se va explicar cómo se encontraba el almacén antes de mi llegada, además de dar contexto a los orígenes de la información que si bien tienen una función ornamental e informativa ponen en situación a las fuentes del repositorio y organización del equipo. También se aprovechará para explicar mis primeros procesos destinados a reunir la información en un mismo ecosistema.

<u>El orígen de los inmuebles</u>

Para entender el orígen de los datos debemos remontarnos a cómo "nacen" estos inmuebles en la entidad financiera, y la forma más habitual es que vengan del propio Data Warehouse de deuda, es decir, el cliente del banco que ha dado un bien como garantía entra en

morosidad por impagos o de igual forma un acuerdo procedente de una compra de carteras de deuda (en ocasiones las entidades se venden deuda entre ellas a un menor precio que el pago de la misma para conseguir liquidez), en ese instante se activan unos mecanismos, en los que el banco, como palanca de recuperación de la deuda, convierte esa garantía mediante una dación (dación en pago) que es un acuerdo voluntario en el que el deudor entrega el bien acordado para saldar la deuda al acreedor, o adjudicación en la que el resultado es el mismo pero viene mediante un proceso judicial, subastas o litigios, entre otros, y después de haber seguido una serie de procesos de saneamiento, como quitar cargas, hacer la toma de posesión (ir con un cerrajero, cambiar llaves…), inscripción en el registro, etc., pasa a ser un activo propio de la entidad (Figura 4.19).

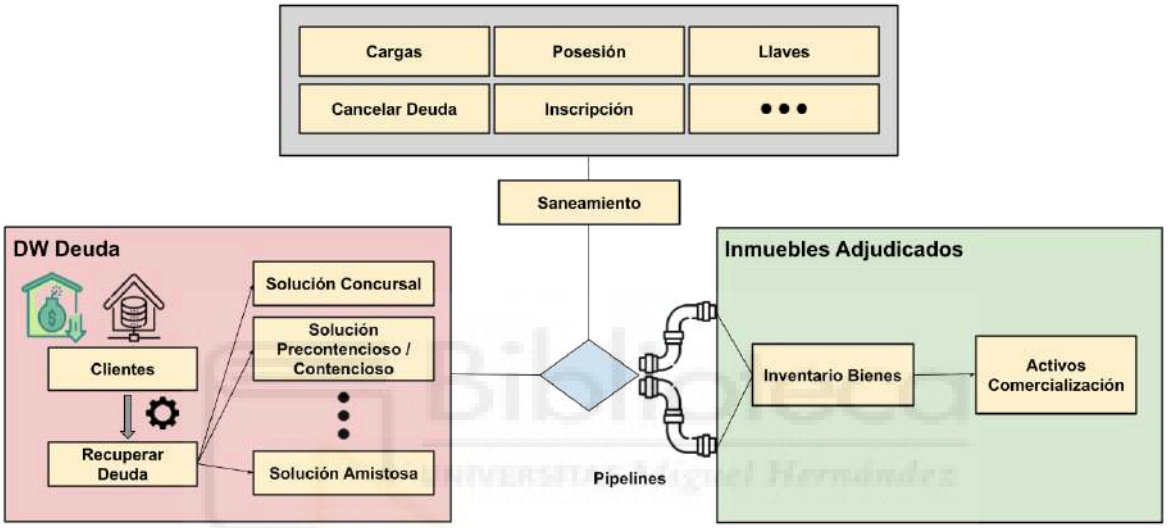

Figura 4.19 Orígenes de un activo inmobiliario

En este punto aparece la figura de la inmobiliaria, o como la nombran en la entidad, el "service provider" (proveedor de servicios) que gestionará y comercializará los inmuebles captados en este proceso (depende del banco puede ser un departamento propio dedicado). En el caso que nos atañe, pongamos que se trata de "Altamira Inmuebles" que mediante una colaboración con la empresa, esto es, empleados de la propia entidad se envían a trabajar a la propia inmobiliaria para crear un nexo entre las dos empresas, con el propósito de alumbrar una base de datos relacional donde se darán de alta estos activos, mediante *pipelines[57]* (conjunto de procesos y utilizados para mover datos desde diferentes sistemas) o tubos, de forma manual.

En respuesta a este funcionamiento empieza a germinar el proyecto de inmuebles, en un principio además de la organización de los datos en un almacén superior (DW), su objetivo principal era crear un modelo interno paralelo a la base de datos de CEGID. Sobre CEGID aclarar que es el software usado por la inmobiliaria para la gestión de activos inmuebles donde incluso la licencia es pagada por la propia entidad financiera y que dará nombre a la fuente principal de base de datos que informará el DW de inmuebles, por ello a partir de

ahora en adelante, se usará CEGID para referirse a las bases de datos origen y no a la aplicación. En cuanto a los detalles de esta BBDD, comentar que es una base de datos que solo se puede consultar desde los terminales financieros de la entidad mediante usuarios dados de alta, es decir, el desarrollador no tiene acceso a ella, los datos de los inmuebles que guarda no son solo de sus características comerciales, también su código postal, calle, etc., así como datos contables como importes de venta o importes de amortización entre otros. Como veremos más adelante, esta BBDD está alojada en Oracle.

Volviendo al punto anterior, para desempeñar todas las tareas de desarrollo que supone estos hitos, en la entidad se crea un pequeño grupo en el campo del data correspondiente a inmuebles (Figura 4.20).

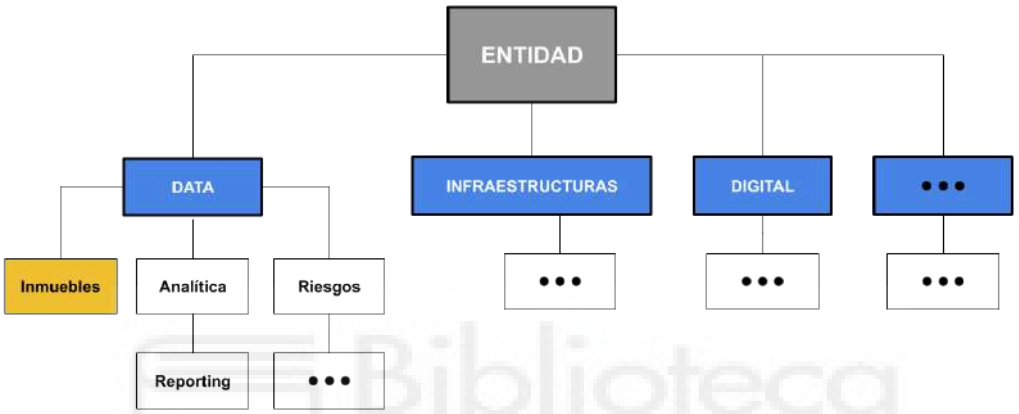

Figura 4.20 Organigrama de la empresa

A este grupo se le asigna un jefe de proyecto, que será un empleado de la parte informática de la entidad; en cuanto a las personas que estarán a su cargo, como en el caso de la inmobiliaria, la institución tiene externalizado los servicios de desarrollo IT en otras empresas tecnológicas, en un sistema que ellos llaman “Modelo de Spotify” (Figura 4.21), este modelo se basa en tener un bolsa de ingenieros de datos y, según las necesidades, fechas y presupuesto, el jefe de proyecto pedirá la incorporación de X personas para llevarlo a cabo, en el caso de inmuebles se solicitó una persona para el comienzo del proyecto.

Así es como se constituyó el grupo de inmuebles provisto de un jefe proyecto que participaría en la toma de requisitos de los clientes (trabajadores de la propia entidad) y la planificación. Este tipo de usuario tiene un perfil técnico, conocimiento sobre el negocio y la particularidad que conlleva en específico los inmuebles, y aunque su trabajo casi a tiempo completo sea acudir a reuniones para toma de requerimiento, tiene conocimientos extensos sobre lenguaje SQL. De igual manera aunque en menor medida también tiene conocimientos sobre la creación de ETLs con Powercenter, la herramienta usada en la entidad. El perfil del ingeniero de datos que aglutina diferentes ámbitos que no son solo la propia de desarrollador de ETLs y código SQL, sino que también es analista de datos, en muchos casos tiene trato con los usuarios en tareas que ya están en un lienzo con los

marcos definidos, y los diferentes casos de uso detallados en el apartado 4.2.2. En cuanto a la etapa de diseño y arquitectura suele ser un entendimiento entre ambos, ya que al ser un grupo muy reducido de dos personas, la toma decisiones en el modelaje se hace mucho más simple.

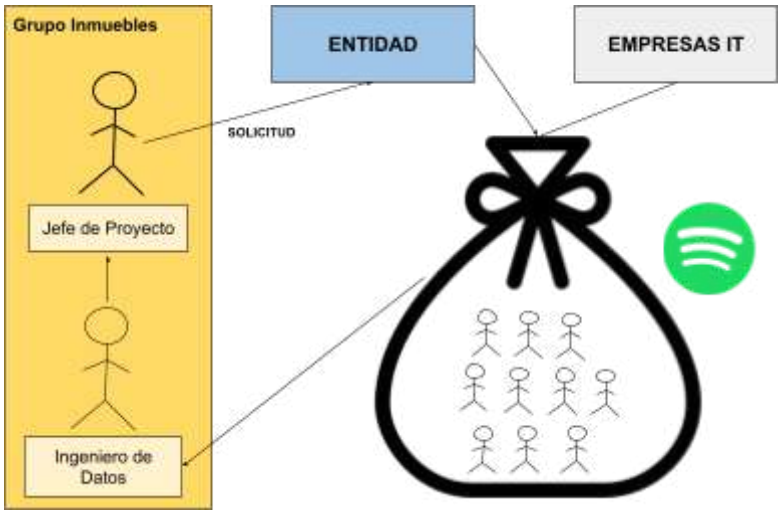

Figura 4.21 Modelo Spotify

Aunque el proyecto empezó en torno al 2018, entre requerimientos, análisis y otros contratiempos como la propia burocracia de la entidad, no sería hasta el 2019 cuando comenzaron los primeros desarrollos.

Los Data Source de Inmuebles y el Staging Area

En un primer momento el compañero que empezó el proyecto eliminó ciertos desarrollos que existían, y empezó el estacionamiento de CEGID (la BBDD de la inmobiliaria) de Oracle en las bases de datos de Teradata. (Tabla 4.3)

Tabla 4.3 Planificación de estacionamientos

	2019	2020		2021		2022		2023	
Módulos		S1	S2	S1	S2	S1	S2	S1	S2
Estacionamiento de datos contables y comerciales en el ecosistema de la entidad.									

Tareas propias | Tareas de equipo | Tareas de terceros

Como se observa, este proceso de "estacionamiento" se realiza a lo largo de la vida del proyecto hasta la actualidad, así que siendo una parte fundamental en la ingesta de datos y para un mejor entendimiento se describirá detalladamente los pasos que componen estos desarrollos.

Ya se ha mencionado que las bases de datos de CEGID de la inmobiliaria Altamira están montadas en Oracle, y este va a ser el origen principal del almacén de datos. Todos los días la inmobiliaria envía esta base de datos al completo con la información actualizada del día anterior, es decir, la inmobiliaria hace una exportación de las base de datos en oracle de CEGID, se comprime en un fichero, se envía a la entidad, cuando llegan a esta existe un proceso automático que descomprime el fichero, y luego hace una importación a las bases de datos de Oracle del propio banco, en otras palabras todos los días se crean y destruyen todas las tablas de la BBDD de Oracle de la entidad para ser reemplazadas por las nuevas procedentes de la inmobiliaria (Figura 4.22).

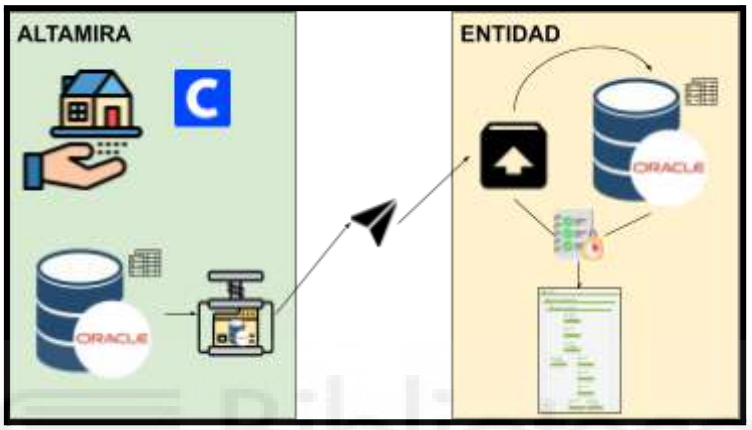

Figura 4.22 Envío de las bbdd Oracle desde la inmobiliaria

El proceso de descompresión e importación lo mostraremos mediante el uso control-M por su forma visual de encapsular los trabajos en diagramas (Figura 4.23).

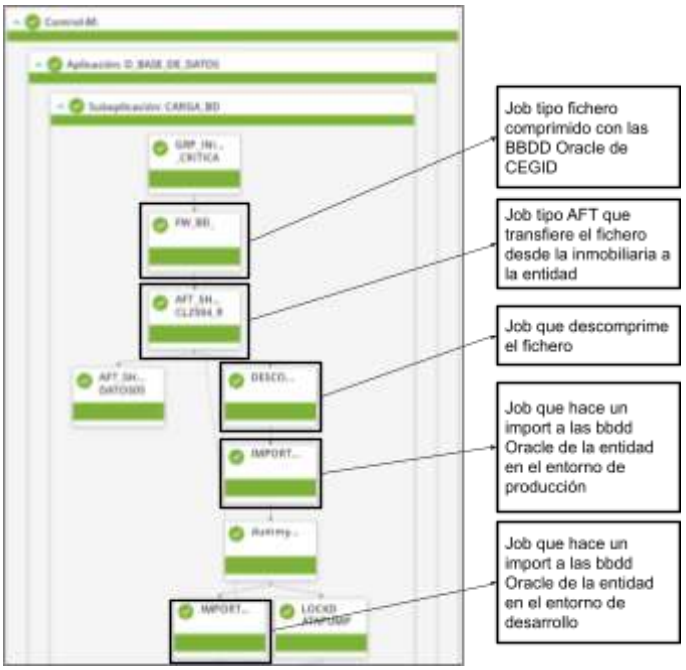

Figura 4.23 Proceso de importación de las BBDD en la entidad.

Recordemos que control-M, permite ver también el inicio y fin de los procesos, en nuestro caso este proceso se ejecuta de madrugada y suele tardar unas tres horas en finalizar (Figura 4.24).

Figura 4.24 Tiempos de carga del proceso de importación

En este momento empieza el trabajo de estacionamiento en el *Staging Area*, este proceso preliminar a la integración, consiste en pasar la información de las tablas de Oracle a tablas de Teradata, añadiendo ciertas transformaciones (Figura 4.25).

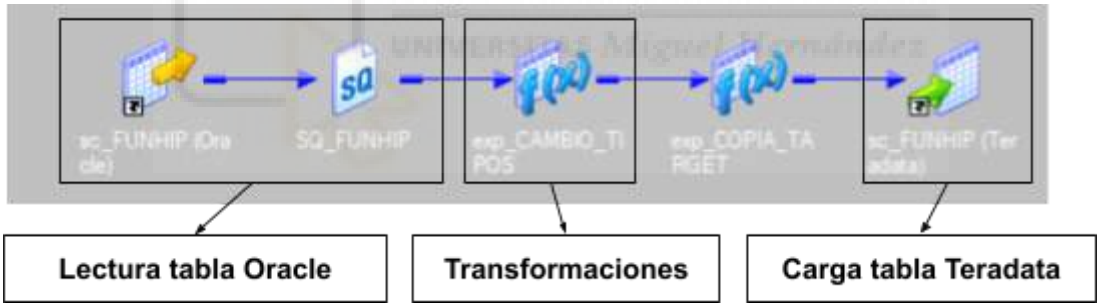

Figura 4.25 Proceso de migración de información entre bases de datos

Lo que se hace es leer todos los datos de la tabla de Oracle sin ningún tipo de filtro, para después realizar algunas transformaciones destinadas a normalizar los tipos de datos que se usará posteriormente en el DW. En este tipo de transformaciones es habitual encontrarse:

- Cambio de numéricos a alfanuméricos (Figura 4.26): Este tipo de conversión puede parecer inocuo pero da más trabajo de lo que parece, todos los campos provenientes de CEGID son numéricos si los datos que portan son de ese tipo, el problema es que un campo numérico limita muchos sus operaciones si los valores que lleva no representan números, no existe mejor ejemplo para explicar este caso que el acuerdo de deuda que está vinculado al bien que representa, ese acuerdo puede tener una apariencia de números como: 897564281928, pero operar con él haciendo sumas, restas, divisiones etc, no aportan ningún valor substancial, sin embargo si es

alfanumérico se puede usar funciones de padding (añadir caracteres hasta completar un número de caracteres), como veremos más adelante, tal vez añadirle una A o D al acuerdo, para diferencia si es de dación o adjudicación. En el caso mostrado (UHNFDH) se trata del número de la finca, que se hace de tipo varchar porque posteriormente será concatenado a otro campo, resultando en un campo nuevo.

- Acortamiento precisiones y calidad del dato: En la misma figura 4.26, también observamos que si los datos de ese numérico como máximo van a llegar a ser un varchar(8) es innecesario guardar más memoria en cambiarlo a varchar (15), esta tesitura también nos encontramos casos correspondiente a la calidad del dato como el número de dormitorios o número de baños, que sabemos por parte de la inmobiliaria que si llegan con un número mayor a 99 es que no están informados (en ocasiones los comerciales rellenan estos campos con números atípicos para señalar estas casuísticas, otro ejemplo sería poner una fecha con año 2099 o 1990), luego esta información servirá para que en el DW marquemos este tipo de casuísticas buscando los números 99 o fechas específicas (Figura 4.27).

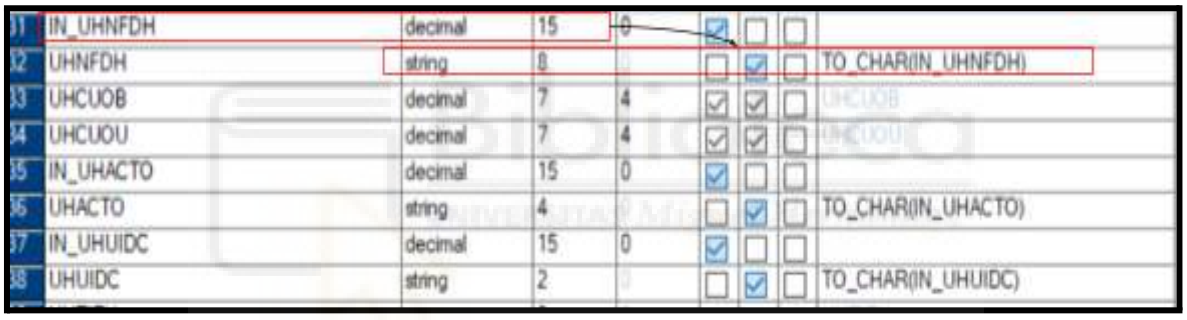

Figura 4.26 Cambio de tipo de dato de numérico a alfanumérico

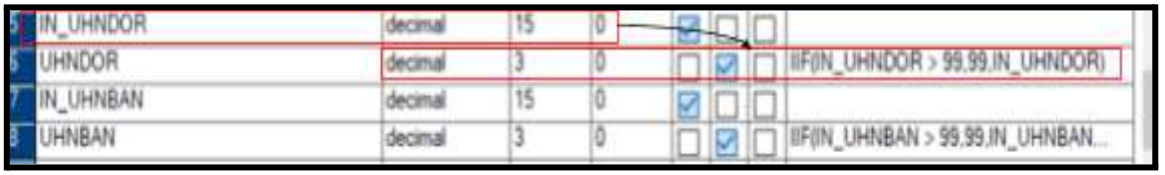

Figura 4.27 Precisiones y ciertos controles en la calidad del dato.

- Padding (relleno): Se ha mencionado como ejemplo en la transformación de alfanuméricos, esta función rellena con un carácter tanto por la derecha como por la izquierda, y es de las más usadas para normalizar datos que ya están en la entidad (como por ejemplo claves), además de estos orígenes el modelo de inmuebles se va a retroalimentar con datos del banco que ya están normalizados (el Data Warehouse de Deuda), si esos campos con los que vamos a operar ya usan el padding para rellenar por ejemplo con ceros un campo de unión, usar este tipo de funciones en las primeras etapas, ahorrará tener que hacerlo en la integración o que por olvido ocasione problemas porque los registros no se estén uniendo.

Con este proceso ya tendríamos la información en una tabla idéntica a la de Oracle pero en Teradata y en el *Staging Area* de la entidad. En el capítulo 3 explicamos que, tanto Powercenter como Teradata y otras aplicaciones, se organizaban por propietarios o aplicativos, en la entidad los propietarios que guardan las tablas del *Staging Area* son: TA (tablas diarias) y TH (tablas históricas), la figura 4.25 de la migración mostraba la carga en una tabla del propietario TA, pero en ocasiones existe un proceso hermano (Figura 4.28) , que desde la tabla del propietario TA hace un copia en la tabla del propietario TH.

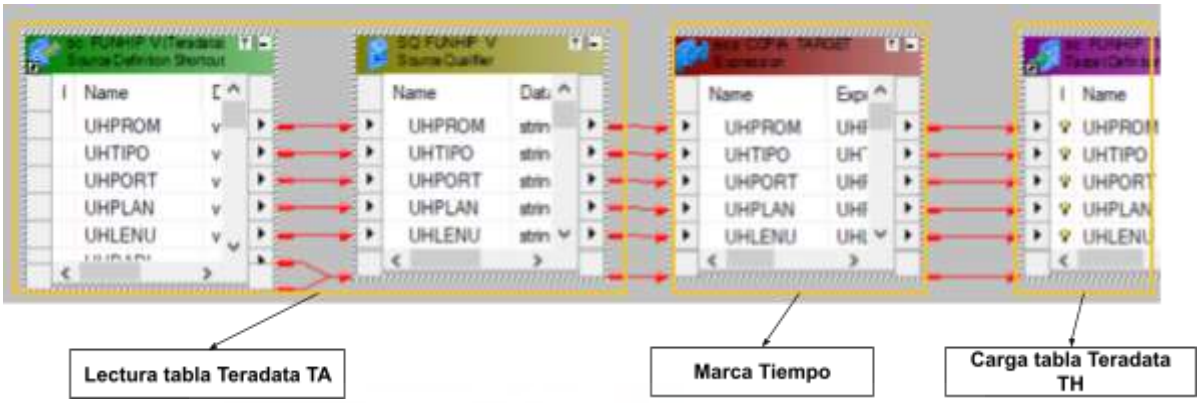

Figura 4.28 Proceso de carga de tabla TH a partir de una TA

El motivo de este proceso, es que el *Staging Area* contempla hacer históricos de fuentes orígenes que pueden ser interesantes por su evolución o de las que se necesita acceder a datos históricos para cálculos más adelante, por ende las TA no realizan ningún históricos y su información se borra cada día, sin embargo en la TH sí contiene un campo tiempo mediante el cual marcan la carga realiza en determinado día, aportando la misma característica que tienen las tablas del Data Warehouse en cuanto a su historificación (Figura 4.29).

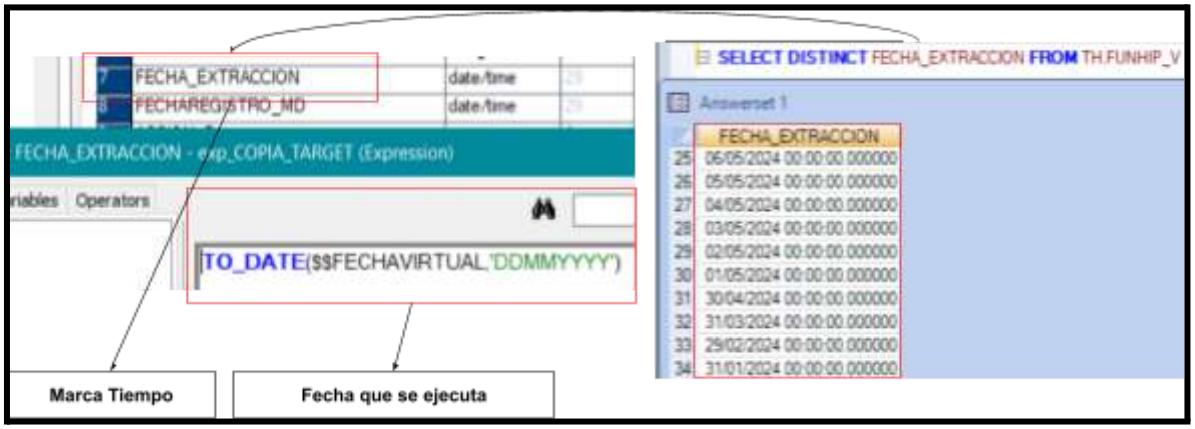

Figura 4.29 Marcado de la partición cargada en una tabla del propietario TH

Este tipo de proceso que guarda información histórica en el *Staging Area* que es un almacenamiento auxiliar, no es común, por eso también se realiza un ventilado periódico

(eliminación de datos), guardando sólo los datos del último día del mes como se observaba en la figura 4.29. Esto se puede conseguir de dos formas: la primera es añadir el nombre de esta tabla, la periodicidad con la que se desea la eliminación de los datos y otras características menores en una tabla de ventilaciones que deja a disposición la entidad para estos menesteres, después se lanzará un proceso Powercenter que leerá la información de esa tabla de ventilados y hará las eliminaciones correspondientes. La otra forma es automáticamente cuando se ejecute el proceso, esto se consigue introduciendo una consulta SQL de tipo DELETE en el workflow que borrará los datos de la tabla según la condición programada (Figura 4.30):

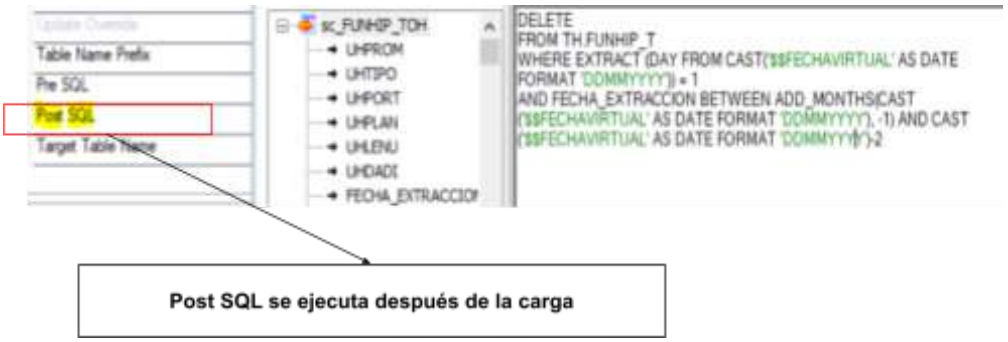

Figura 4.30 Delete de la tabla TH

Esta condición hace que cada día 1 de mes, parámetro "*$$FECHAVIRTUAL*", se eliminen todos los registros del mes anterior salvo el último día de dicho mes (véase SQL en figura 4.30).

Aprovechando el desarrollo de estos dos procesos también se va a explicar dos de los tres tipos de carga que se han usado en el trabajo, ya que estos procesos de estacionamiento son ejemplos excepcionales que ayudan a entender el uso de estos métodos. En el capítulo 3 ya se nombraban, estos son:

- Fload (*Fast Load*) [59]: Es un método de carga que aparece seleccionando la opción de "*Bulk Load*" (carga masiva)(Figura 4.31), como su nombre indica es un método más rápido que los tradicionales como por ejemplo el relacional[65] que no se usa en este trabajo pero es indicado para un volumen de carga más pequeño o mediano porque procesa cada registro individualmente. Cuenta con un menor uso de recursos de red y de sistema, lo que mejora la migración de grandes cantidades de datos. Su uso es especialmente recomendado para las tablas del propietario TA por diferentes causas:
    - Requiere de tablas vacías: *FastLoad* hace de forma predeterminada un borrado total de la tabla que va a cargar porque necesita que esté vacía, ideal para tablas que no se van a historificar y que necesitan de un borrado optimizado constante.

- Solo permite operaciones INSERT: Necesitamos asegurar que los datos de orígen no son modificados, al tener solo una operación soportada de inserción nos aseguramos la integridad de los mismos.
- Altamente Optimizado: Está pensado para la migración de grandes volúmenes de datos, lo que es idóneo para este tipo de carga diaria.

Figura 4.31 Configuración carga Fload

- PDO (*Pushdown Optimization*)[60]: Este método de carga es el más rápido de que se dispone, puesto que en su funcionamiento no llega a entrar en la integración del Powercenter sino que se usan los recursos del propio gestor de base de datos para hacer la carga. Nos podríamos preguntar por qué no se utiliza este método en todos los desarrollos, y es que PDO tiene ciertas limitaciones, debe utilizarse con orígenes y fuentes de una misma base de datos, por ejemplo para los procesos del propietario TH que lee una tabla de Teradata del propietario TA y hace una copia en otra tabla de Teradata en el propietario TH, además al operar directamente las transformaciones contra la base de datos, es decir, lo que hace es traducir la lógica de transformaciones de Powercenter en consultas SQL y las envía a la base de datos (esta consulta se puede leer en el depurador Figura 4.32)

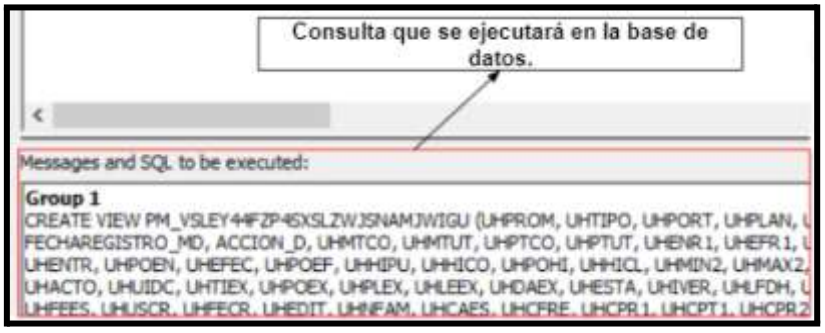

Figura 4.32 Consulta que crea PDO

Quedan excluidos todos esos elementos de integración que no sean compatibles en este caso con Teradata (que Powercenter sí tiene). Por otro lado, este método de

carga es más complicado a la hora de desarrollar, ya que si bien tiene un pequeño depurador (Figura 4.33) los mensajes de error son más difíciles de comprender, no llegando a ser aconsejable para integraciones con un alto grado de complejidad.

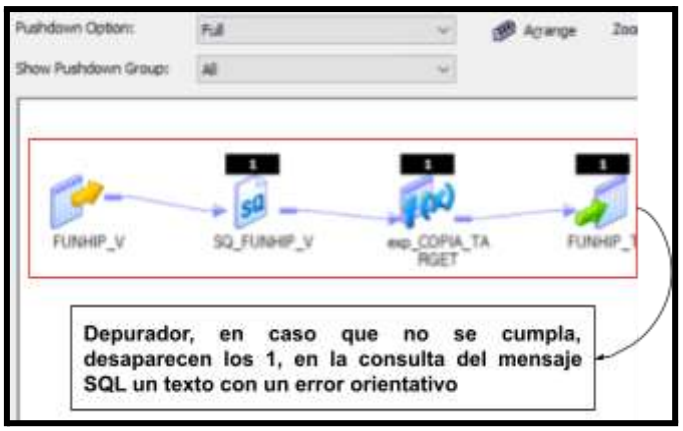

Figura 4.33 Depurador PDO errores

Como puede verse, este proceso de "estacionar" es como una capa intermedia que representa una migración entre bases de datos a un almacén intermedio (*Staging Area*) para después ser integrados en las tablas finales del Data Warehouse. Aunque hay que aclarar ciertos puntos conceptuales antes de seguir, no todos los procesos que crea Powercenter se podrían a llegar a definir como "ETL", si bien el proceso que carga la tabla TA podría llegar a considerarse como tal (lee información, la transforma y la carga) sería con ciertas comillas pues es más bien una migración de datos entre bases de datos, ya que realmente consiste en copiar los datos de las tablas de Oracle a Teradata (ambas pertenecientes ya a la entidad), porque en esta última es dónde se va a ubicar el Data Warehouse debido a que tiene los servidores computacionalmente más desarrollados en esta tecnología de BBDD, para después aprovecharla para realizar ciertas transformaciones dedicadas más a la estructura y tipo de los datos que se sabe que nunca van a cambiar. Por otro lado, el proceso de la TH no es una ETL porque su función radica en crear un histórico de esa tabla si fuese necesario (en este caso es una copia de datos en todos los sentidos), pero finalmente se usa Powercenter porque ambos procesos se van a automatizar con posterioridad (Figura 4.34), comenzando su ejecución una vez que haya terminado la importación.

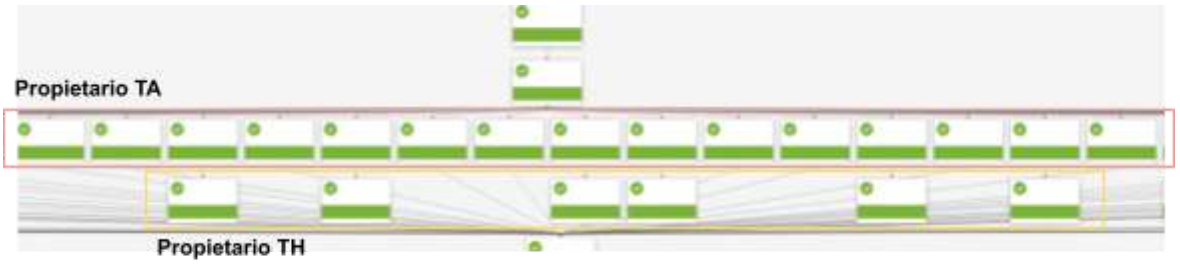

Figura 4.34 Diagrama en control -M de los estacionamientos de las tablas TA y TH

Además hay que destacar que estas tablas son estacionadas para un fin (Figura 4.35), es decir, si se va a integrar cierta información en el DW, primero se analiza de dónde va a proceder esta información, y en el caso que dicha información no esté estacionada en alguna tabla, se piden datos mediante Erwin (herramienta de modelado vista en el capítulo 3) al departamento de BBDD para su creación. Si en las primeras etapas del proyecto había cerca de veinte tablas estacionadas, a día de hoy superan las cincuenta, pero existen más del doble todavía sin estacionar, hay que entender que si se hace este procedimiento es porque si se hubieran estacionado todas, la mitad de la información no se estaría usando y solo acarrearía costes computacionales y de almacenamiento.

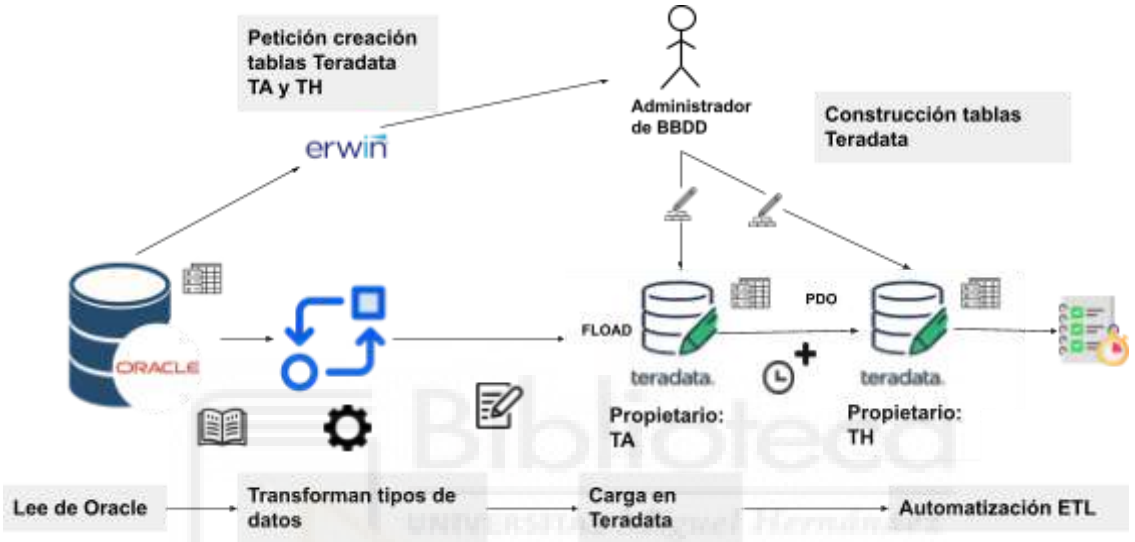

Figura 4.35 Diagrama del flujo para la creación de estacionamientos

Otro punto a aclarar, son las pocas transformaciones realizadas para la normalización de los datos en cuanto a errores se refiere, esto es así porque partimos de una situación en la que la base de datos ya está normalizada y en funcionamiento desde hace tiempo, no son datos que llegan a partir de formularios sin ninguna supervisión, hay personas que han recolectado los datos, los han enviado por las tuberías y ha habido otras que los han insertado (en este caso a mano), este proceso hace que sea difícil de encontrar errores y la convierte en una base de datos muy fiable. La dificultad radica en que son datos desperdigados en más de cien tablas, en algunas de las figuras observamos como el nombre de la tabla y sus campos son muy poco o nada orientativos (carencia que se corregirá en el Data Warehouse) complicando más el entendimiento de la información que guardan.

4.1.2.- Entrada en el proyecto, orígenes y el DW.

Mi entrada en el proyecto de inmuebles viene dada por un cambio de rumbo, en lo que en un principio tan solo iba a ser un modelo interno que fuera en paralelo con el de la inmobiliaria, es decir, Altamira era el proveedor de información inmobiliaria de la entidad

y gestionaba tanto el alta de los activos como la base de datos de CEGID, y aunque había empleados de la propia entidad colaborando con ella, por si acaso, se consideró relevante disponer de un modelo propio de información en los sistemas del banco. Se aprovechó esta tesitura junto con la incorporación de los datos comerciales y las inversiones inmobiliarias en el modelo, para convertirlo en un modelo centralizado y explotable por la entidad, crear un propio CRM de inmuebles e informes diarios que contabilicen los activos. Siguiendo el modelo de Spotify, el jefe de proyecto solicitó un nuevo ingeniero de datos a la entidad, y como el compañero anterior pertenecía a la misma empresa que yo, entré en el proyecto de inmuebles (Figura 4.36):

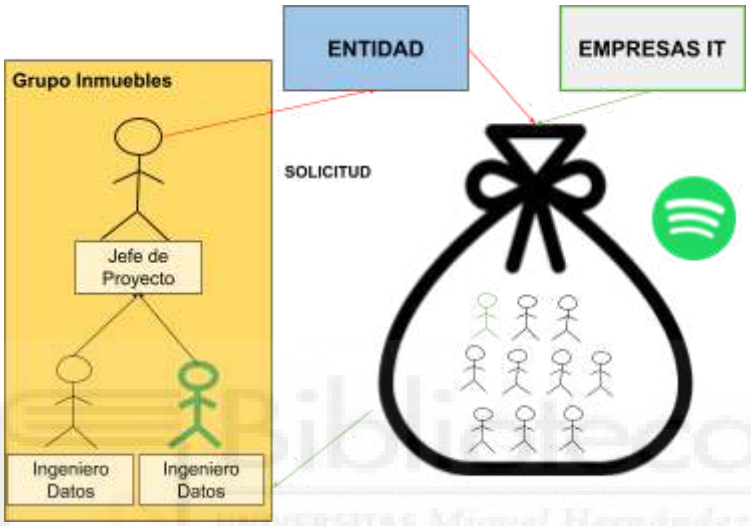

Figura 4.36 Modelo de Spotify Inmuebles

En el año 2020 (como se observa en el Gantt), se formarán los cimientos de inmuebles y se estabilizará el modelo, para que en el 2021 con la salida del compañero, volver al grupo de dos: un jefe proyecto y un ingeniero de datos (yo), siendo el grupo más pequeño formado en la entidad. En el momento de la entrada el DW de inmuebles correspondía a la siguiente estructura (Figura 4.37)

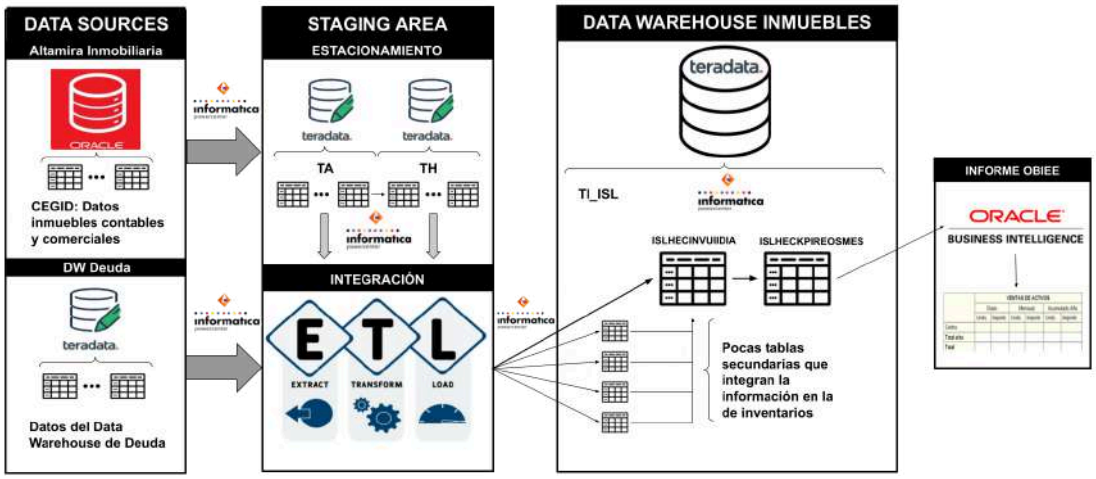

Figura 4.37 DWH de Inmuebles Etapa 1

Sus fuentes eran dos, la inmobiliaria que proporcionaba los datos comerciales y contables de los inmuebles, y cómo podríamos sospechar del apartado de los orígenes de los inmuebles, el DW de deuda que proporcionaba la información de los acuerdos que están ligados a los bienes, en el caso de la BBDD de CEGID se hacía un estacionamiento en los propietarios TA (información diaria) y TH (información histórica) como se ha explicado en el apartado anterior, para luego ser integrada mediante ETLs a las tablas del DW de inmuebles (principalmente a la tabla de los inventarios diarios) que se ubican en el propietario TI (TI hace referencia a "Tablas Integradas"), en el caso de la información de deuda no es necesario ese estacionamiento puesto que la información ya está integrada en el ecosistema de la entidad que es Teradata. Por último, un pequeño informe de OBIEE, que aunque pudiera parecer esquelético en su día, pertenecía a un informe mayor donde iba acompañado con otros datos del DW de deuda.

Tabla 4.4 Planificación integración de la Inventarios UII

Comenzaremos con la tabla de inventarios (Tabla 4.4). Analizando su nombre: HEC indica que se trata de una tabla de hecho, INV por Inventarios, UII se refiere a la integración de diferentes bases de datos, y DIA quiere decir que será automatizada para ser ejecutada diariamente. La selección de estudio de esta tabla entre otras del modelo inmueble sigue tres objetivos: el principal, es la tabla central del modelo estrella donde se hace el proceso de integración de más campos del modelo, en segundo lugar, se mostrará también como Powercenter permite la escalabilidad de los procesos, puesto que es una ETL que empezó el compañero integrando la información de cerca de 200 campos en sus inicios y que yo he ido incrementando hasta llegar al día de hoy a los 360 campos. Y el tercer objetivo es la inclusión de dos nuevas fuentes, los nuevos inmuebles procedentes de inversiones inmobiliarias cuyos orígenes están en la entidad y la de datos comerciales que se envían y se integrarán mediante ficheros desde la inmobiliaria.

Abriremos este proceso entendiendo cómo se puede escalar una ETL, ya sea para añadir integraciones a la misma o trabajar de forma modular, y es mediante el desarrollo de varios mapping que integrarán los datos en un mismo destino, para luego enlazarlos a un mismo workflow (recordemos que el workflow permite la ejecución de diferentes sesiones que son la configuración del mapping). De esta forma si se realizó una integración inicial de los primeros campos en la tabla de inventarios, para nuevas integraciones tan solo hay que crear nuevos mappings e introducir sus sesiones en el workflow en serie (si se introducen datos a la vez, la tabla se bloqueará) para que se ejecuten después del inicial y así no

modificar el primer mapping que ya integró información correctamente y anulando la posibilidad de modificarlo accidentalmente (Figura 4.38).

Figura 4.38 Sesiones permiten escalabilidad en Powercenter

Este es el proceso para insertar nuevos registros, sin embargo, si hay que integrar información en nuevos campos en la tabla destino, hay que realizar un paso más consistente en configurar la sesión para que realice una operación UPDATE (en vez de la configurada por defecto que es INSERT, Figura 4.39), esto es, si la primera sesión ya tenía el número de activos deseados, para integrar nuevos campos en esa tabla, si hacemos un segundo INSERT con la información de los nuevos campos el proceso fallará por duplicados porque estará intentando introducir las mismas claves principales y la misma marca de tiempo que ya ha insertado en la primera sesión (esta tabla ya pertenece al DW por ende contendrá un campo de tiempo para historificar), no se están introduciendo nuevos registros, sino que se están informando mejor los registros ya existentes. En esta situación, tan solo hay que leer el nuevo mapping desde la propia tabla, para obtener las claves principales, y a este flujo ir añadiendo la información (UPDATE) de los campos nuevos que se quieren integrar.

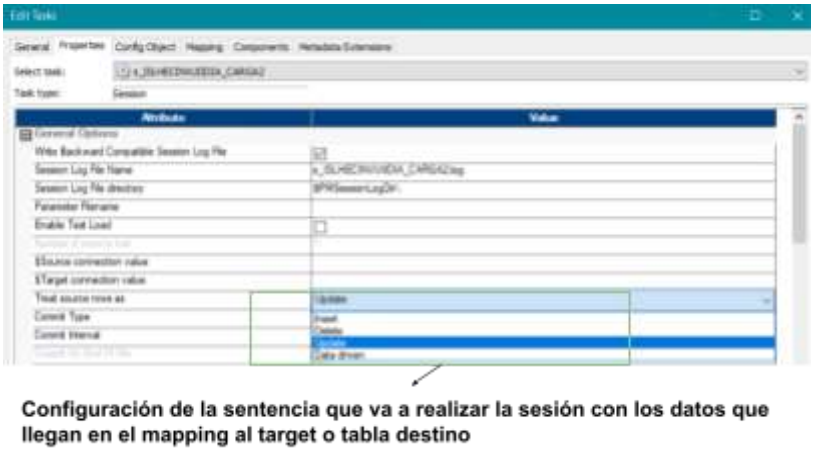

Figura 4.39 Configuración del Update

Por otro lado, Powercenter también admite funciones como DELETE o DATA DRIVEN, si se configura la primera borrará todos los registros que tengan la misma clave principal en

el flujo que llega a la tabla destino que coincidan con los registros guardados en la misma, el segundo se usa para tomar decisiones dinámicas basadas en los datos del flujo principal, por ejemplo junto a la transformación de "Update Strategy" se puede conseguir un UPSERT, traducido de forma más sencilla, es la posibilidad de actualizar (UPDATE) o agregar (INSERT) según una condición, este tipo de función son muy usadas en inmuebles para modificar datos o insertarlos según si un campo está vacío o no.

Aprovechando esta temática abordaremos la inclusión de un nuevo origen en el Data Warehouse, los datos de inversiones inmobiliarias, al igual que el banco compra deuda, también compra bienes inmobiliarios. Como vimos en el apartado anterior de diseño, esta información viene de la propia entidad mediante un fichero que se deposita en un directorio interno, este fichero se estaciona (Figura 4.40), se integra y se une con los que vienen de la inmobiliaria (las transformaciones realizadas se describirán en integración de datos comerciales más adelante pues son las mismas).

Figura 4.40 Mapping integración inversiones inmobiliarias.

Para unir estos activos con los de la inmobiliaria podemos, aunque el mapping sea el mismo, realizar dos opciones a nivel de ejecución, o hacer una sesión con un INSERT y añadirla al workflow que carga la tabla "inventarios integración", o como se ha hecho finalmente en este caso crear un nuevo workflow que se automatice después de la ejecución de la propia tabla de "inventarios" (Figura 4.41).

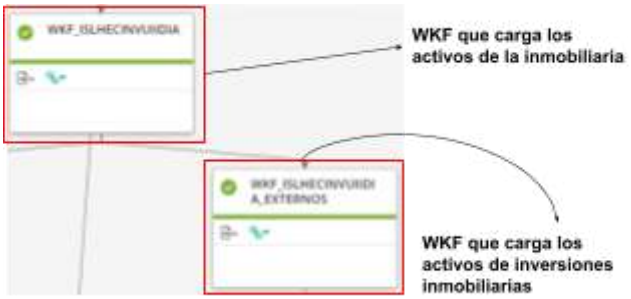

Figura 4.41 Control-M diagrama ejecución de los procesos de la Inventario

La decisión de diseñar la integración de esta forma radica en primer lugar en que en las primeras etapas de implantación del proceso, cuando se llegaba a esta ejecución podría no estar disponible el origen porque que no había llegado el fichero todavía, que en muchos casos tardaba desde las ocho a las doce del mediodía en ser recibido, cuando el propio proceso de la tabla "inventario" terminaba sobre las cinco de la mañana, y la carga de los

activos de la inmobiliaria tenía prioridad sobre estos. Y segundo, ya que estos activos proceden de un origen distinto a los de la inmobiliaria, separar los workflows de carga es interesante por una mera cuestión de organización, estructuración y validación de los procesos en el apartado de pruebas.

Llegados a este punto cabría preguntarse, cómo es posible que otro departamento tenga ya integrados los datos y nosotros necesitemos un proceso faraónico para este propósito, la respuesta se ilustra la Figura 4.42. Efectivamente, solo vienen informados los campos únicamente necesarios para la contabilidad, por dar contexto a esta integración, resulta que la inmobiliaria por cada activo que gestiona cobra al banco un cierto dinero, en este caso el banco decidió que gestionaría él mismo estos activos, por lo tanto, aunque sí que se comercializan, esto no se hará por las vías comunes, ni aparecerán en los cuadros de mando del usuario comercial, estos activos tiene la peculiaridad de que se suelen vender por lotes y cuando se venden se recibe otro archivo con ciertos valores contables que se cargan en la tabla "Kpi" que se comentará más adelante.

Figura 4.42 Datos normativos carga de campos

Prosiguiendo con la ETL de la tabla inventarios, los tipos más comunes de integración que se realizan en el proceso se podrían dividir según la dificultad del campo en:

- *Campos Directos*: Es la información más fácil de obtener y validar en una ETL, se consideran de este tipo todos los campos que se unen al flujo principal mediante un join sin hacer ninguna transformación en sus datos directamente desde la tabla origen. En la Figura 4.43 se ha elegido esta muestra de integración porque representa varias cualidades, se quiere representar como al flujo principal (1) que es la propia tabla de "Inventario" con la información y la tabla "perímetro" cargada previamente, se le van añadiendo diferentes campos de las tablas (2) y (3) mediante muchos *left joins*, se trata de campos directos sin ninguna transformación. Los elementos de tipo filtro antes de la unión se deben al método de desarrollo destino a tablas (Algoritmo 4.1) en las que desde un mismo campo se puede obtener varios.

La inmobiliaria tiene algo parecido a tablas de tipo “cajón de sastre”, en otras palabras, guardan información muy variopinta que no tiene relación entre sí en una misma tabla (son las marcadas con dos y tres en la Figura 4.43) para que cada registro se convierta en un campo hay dos opciones, la primera es repetir este origen varias veces en el mapping y filtrar por el campo nombre para luego unirlo al flujo principal, o leer la tabla sin ningún tipo de filtro y posteriormente filtrar por el campo nombre, esta explicación es la que aplica en el Algoritmo 4.1 usando la tabla 4.5 como guía.

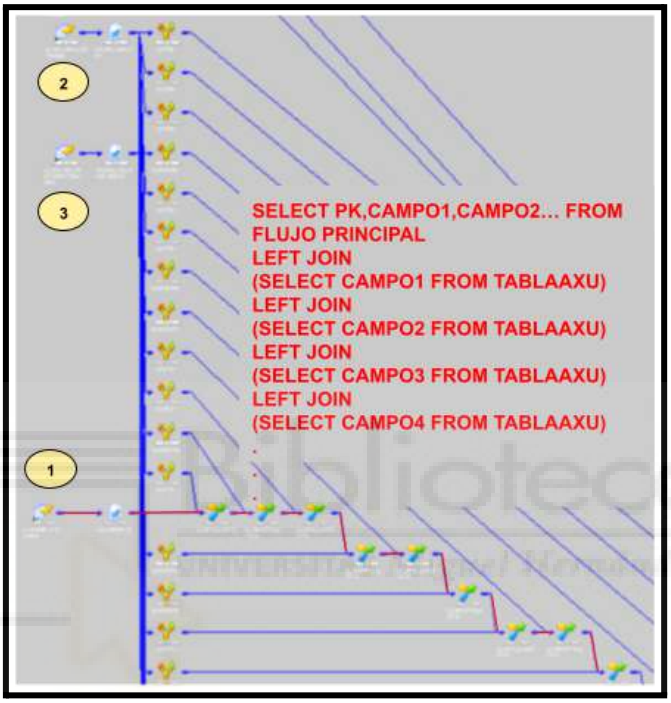

Figura 4.43 Campos directos que se unen al flujo principal

Tabla 4.5. Ejemplo tabla “cajón de sastre”

PK Activo	Campo Nombre	Campo Valor
1	Tapiado	S
2	Alquilado	S
3	Ocupado	N
4	Suministro eléctrico	S

Algoritmo 4.1: Integración de campos con lectura completa de tabla origen

```
1 SELECT
2 CASE WHEN CAMPONOMBRE='Tapiado' THEN CAMPOVALOR END CAMPOVALORTAPIADO,
3 CASE WHEN CAMPONOMBRE='Alquilado' THEN CAMPOVALOR END CAMPOALQUILADO, ...
4 FROM TABLASASTRE
```

su símil si hubiéramos leído varias veces de la misma tabla 4.44 pero filtrando sería igual al pseudocódigo del Algoritmo 4.2.

Algoritmo 4.2: Integración de campos con lecturas repetidas de la misma tabla

```
SELECT PK, CAMPOVALORTAPIADO, CAMPOVALORALQ,...
FROM FLUJOPRINCIPAL
LEFT JOIN (
SELECT PK, CAMPOVALORTAPIADO FROM TABLASASTRE WHERE CAMPONOMBRE='Tapiado')
TABLASASTRETAP ON ...
LEFT JOIN (
SELECT PK, CAMPOVALORALQ FROM TABLASASTRE WHERE CAMPONOMBRE='Alquilado')
TABLASASTREAL ON ...
```

- *Campos Semidirectos*: Son campos en los que el origen no cambia pero reciben alguna transformación o aún filtrando con una cláusula WHERE, no se puede añadir al flujo principal porque duplicaría sus registros. Un ejemplo representativo de este caso es la fecha de tasación, un inmueble puede tener distintas fechas de tasación y aunque es una información interesante, para la tabla central del modelo tan solo se necesita integrar la última, además de existir otra tabla para estos menesteres que es la tabla de hechos de "tasaciones". Para conseguirlo se deberá realizar un MAX(FEC_TASA) para reducir los registros a una tasación por activos y así poder unir al flujo principal, un ejemplo también habitual y más difícil es el de la Figura 4.44, en ocasiones existen varias fechas de venta que se ingresan al sistema (ver Tabla 4.6 como guía), y la correcta no es la máxima fecha de venta sino que es la última ingresada (que puede no ser la máxima) y ese valor lo da la secuencia, el problema es que si se hace un MAX(SECUENCIA) a la hora de agrupar no funcionará pues las fechas de venta serán diferentes.

  En estos casos siguiendo la Figura 4.44, se lee del mismo origen, en uno de ellos (1) se hace la agregación de MAX(SECUENCIA) pero solo nos llevaremos las claves principales y la secuencia sin la fecha venta, para luego más tarde unir (2) con el otro flujo que sí tiene la fecha de venta mediante un INNER JOIN que filtrará la fecha correcta (Algoritmo 4.3).

Tabla 4.6. Ejemplo tabla campo semicalculado

PK Activo	Fecha Venta	Secuencia
1	22/03/2024	3
1	25/03/2024	2
1	21/03/2024	1

Algoritmo 4.3: Integración de campos semicalculado basado en la tabla 4.6

```
SELECT B.PK,B.SECUENCIA, FEC_VENTA FROM
(SELECT PK  SECUENCIA,FEC_VENTA TABLA)A
INNER JOIN
(SELECT PK,MAX(SECUENCIA) TABLA GROUP BY SECUENCIA)B
ON A.PK=B.PK AND A.SECUENCIA=B.SECUENCIA
```

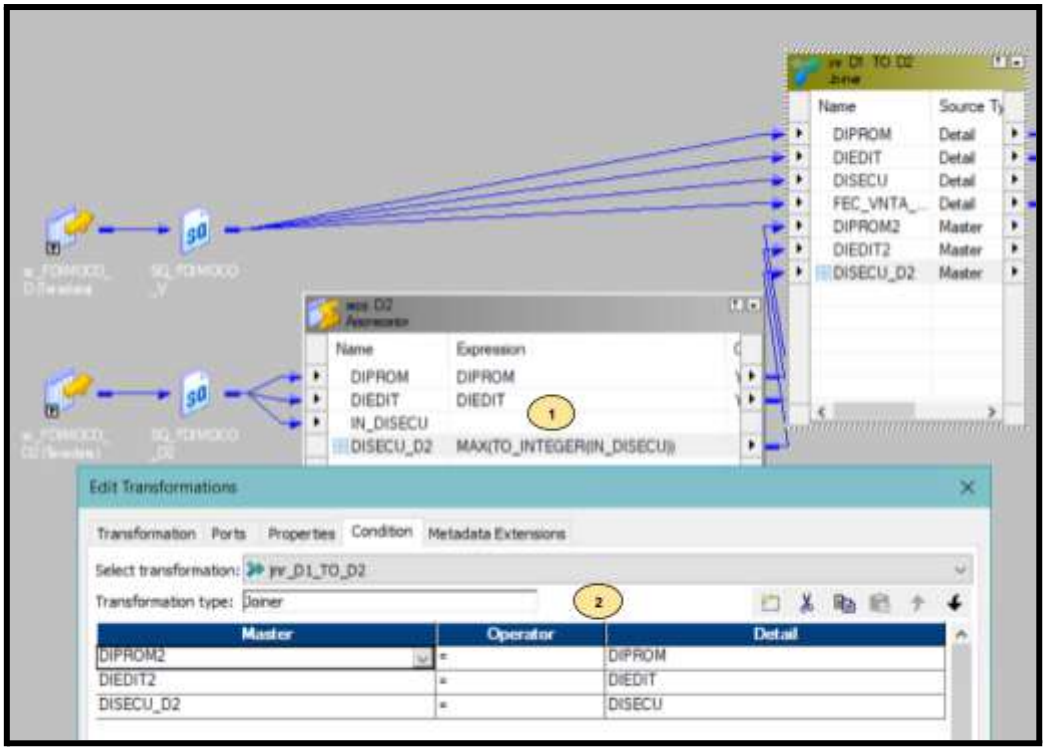

Figura 4.44 Ejemplo campo semi calculado

En cuanto a los campos que reciben alguna transformación, pertenecen a esta categoría también, hacemos alusión a los campos que son directos de una tabla pero por ejemplo se codifican (Tabla 4.7) o se hacen un control de nulos para que, cuando el campo sea nulo, insertar por ejemplo una arroba u otro valor acordado. La codificación es el mecanismo para transformar datos a otros más sencillos por ejemplo las provincias (como se explicó en el capítulo 3).

Tabla 4.7 Ejemplo codificación

PK Activo	Fecha Venta	Secuencia
REFPROVINCIAS	ÁLAVA	01
REFPROVINCIAS	ALBACETE	02
REFPROVINCIAS	ALMERIA	03
REFPROVINCIAS	ASTURIAS	04
REFPROVINCIAS	ÁVILA	05

A estos códigos se les asignará un referencial (*REFPROVINCIAS*) y se insertarán en una tabla dimensión (*DIM DICCIONARIO*), cuando se necesiten los valores que traducen tan solo será necesario realizar una consulta a esta tabla como se indica en el Algoritmo 4.4.

Algoritmo 4.4: Consulta de la tabla diccionario mediante SQL

```
1 SELECT * FROM TI.DIMDICCIONARIO_V WHERE REFERENCIAL= 'REFPROVINCIAS'
```

- *Campos Calculados*: Son todos los campos para cuya creación participan más de un campo o campos agregados que no se puede encontrar en origen, de este tipo puede ser tan simple como una unión entre dos (Figura 4.45)

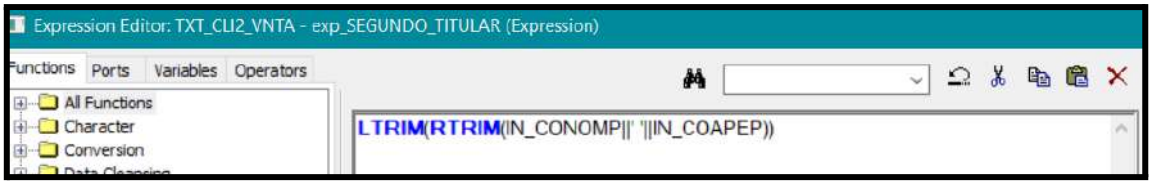

Figura 4.45 Unión de dos campos en Powercenter

Simplemente una agrupación para obtener otro campo agregado como el número de visitas que ha recibido un inmueble: (Figura 4.46)

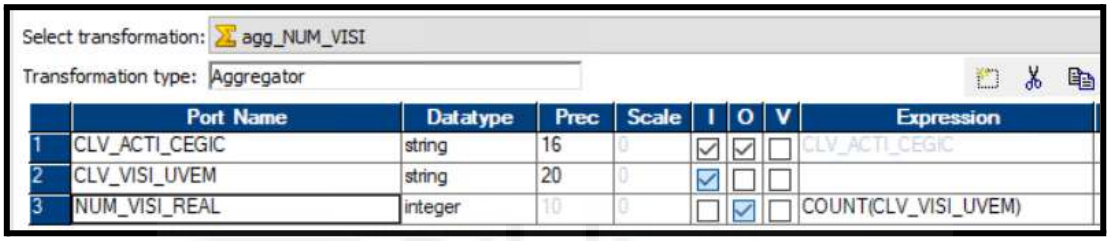

Figura 4.46 Campo calculado agregado

O cálculos mucho más complejos como el de la Figura 4.47 que calcula el estado de un inmueble a partir de la descodificación de dos campos.

Figura 4.47 Campo calculado Powercenter

Su traducción en lenguaje SQL usando la función *CASE-WHEN* se corresponde con el Algoritmo 4.5.

Algoritmo 4.5: Codificación de valores tomando como ejemplo el de alquiler

```
1 CASE WHEN ALESTA='25' THEN 'Alquilada'
2 WHEN UHESTA='0' AND URESTA='5' THEN 'Disponible Alquiler'
3 ...
4 ELSE '' END COD_ECV_ESTADO_AQUILER
```

También se puede usar otras funciones como el *DECODE*, aunque existen muchas más, estas son las integraciones más típicas que podemos encontrar en la tabla de hecho principal de "inventarios" de integración.

El último método de carga utilizado en esta entidad es Mload (Figura 4.48). El método *MultiLoad*[63] de Powercenter es para la carga masiva de datos usando en el proceso las capacidades de carga paralelas de Teradata para optimizar la integración, soporta las mismas operaciones mencionadas anteriormente de INSERT, UPDATE, DELETE y UPSERT. Emplea tablas de trabajo como almacenamientos intermedios que se pueden configurar otorgándoles más memoria en el propio workflow. Este método de carga es más interesante que PDO a la hora de depurar, porque cuando falla genera ciertas tablas de error, como *LOADERROR.UV_PROCESO* y *LOADERROR.ET_PROCESO*, ambas se pueden consultar en el gestor de BBDD de Teradata. La primera guarda, en su caso, todos los registros que se han duplicado, muy interesante ya que ahorra el ejecutar la consulta traducción en su búsqueda. La segunda guarda el error que se ha producido. Su configuración es idéntica a la figura 4.31 de *Fload* en cuanto a la tabla origen y destino.

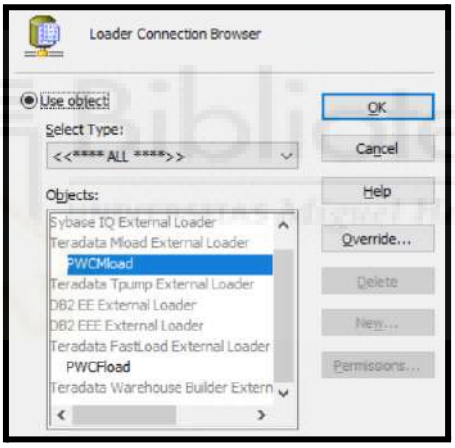

Figura 4.48 Método carga Powercenter Mload

A continuación se describe el trabajo con los dos últimos orígenes del DW de inmuebles, los datos comerciales y las demandas (Tabla 4.8).

Tabla 4.8 Planificación integración de los datos comerciales

	2019	2020		2021		2022		2023	
Módulos		**S1**	**S2**	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**
Integración vía fichero de Datos Comerciales									
Integración vía fichero de Datos de Demandas									

Tareas propias | Tareas de equipo | Tareas de terceros

Hasta ahora los datos comerciales como el código postal, la calle, la provincia, etc. se reciben por parte de las tablas de la BBDD de CEGID, en este caso los datos comerciales,

además de recibir la misma información para completar mejor la obtenida de CEGID, se recibirán los datos comerciales propiamente dichos, como las ofertas recibidas por los inmuebles, las visitas realizadas, las llaves y el movimiento de las llaves que tienen los comerciales para enseñar los inmuebles, los trabajos de mantenimiento realizados, etc. Esta información procede de las propias bases de datos de la inmobiliaria, no de CEGID. Estos ficheros equivalen cada uno a una tabla de integración, es decir, no existe como tal una ETL como en el caso de la tabla "inventarios" que integre la información de muchas tablas a una única, sino que a partir, por ejemplo, del fichero de ofertas se carga la tabla de hecho de "ofertas". La causa de este tipo de volcado radica en que el fichero recibido es una información pedida a la "carta" donde el jefe de proyecto, después de diferentes reunione,s solicita determinados campos a la inmobiliaria, lo que se traduce en que esta, ya ha pasado un proceso de ETL en sus propias tablas y el resultado es ese fichero para cargar en nuestros sistemas, es más, su formato es parecida a una tabla ya importada (llega en registros separados por un carácter especial que hace de división entre campos). Las transformaciones que se harán en estos procesos obedecerán primero a la calidad de los datos, algunos ejemplos:

- *Campos mal informados*: Son campos sin sentido respecto a la descripción del campo en sí, como teléfonos con letras o códigos postales con más de cinco dígitos.
- *Formato de los ficheros*: Las tablas de Teradata están configuradas en formato ANSI y aunque Powercenter permite el cambio de formato en sus archivos de entrada, en ocasiones no acaba de funcionar del todo bien, se requiere traducción al formato UTF-8 para carácteres especiales como acentos o eñes que su conversión de formato son símbolos ilegibles.
- *El separador*: Como se ha mencionado, los datos del fichero vienen separador por un separador que hace limitador de campos, en este caso si el separador es un carácter habitual como un punto coma, es posible que alguno de sus registros los usen para separar información, y a la hora de leer, el proceso se equivoque porque piense que un delimitador cuando no lo es (Figura 4.49).

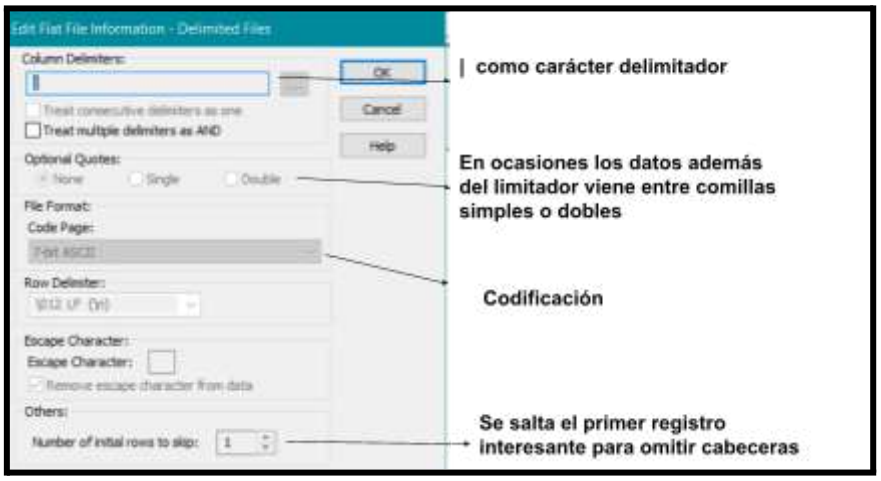

Figura 4.49 Configuración archivo entrar en Powercenter

- *Saltos de línea*: El fichero puede traer saltos de línea que provoquen errores en la lectura por parte de Powercenter:
- *Formato de fechas*: En ocasiones, en distintos registros hay fechas con formato DD-MM-YYYY o DD/MM/YYYY, Powercenter permite sin transformaciones leer un formato en concreto (Figura 4.50), en estos casos la mejor solución es pedir en los orígenes que hagan los cambios necesarios en sus procesos para adoptar el mismo formato.

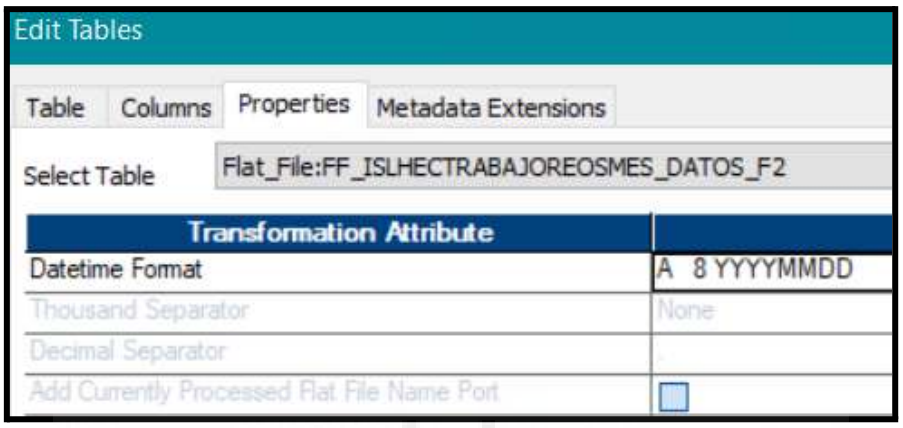

Figura 4.50 Configuración fecha de entrada en fichero Powercenter

- *Códigos con diferentes codificaciones para un mismo valor*: Aquí entrarían todos los valores que representan lo mismo, pero se envían de diferentes formas por ejemplo: Banco Sabadell o simplemente Sabadell, el valor es el mismo pero si integramos esta información en una consulta habría que considerar todos los casos para no incurrir en errores por doble codificación.
- *Recepción de datos con más precisión que el campo que los contiene*: Error muy común en el que se cortan los datos porque la estructura enviada del campo a la hora de crear la entidad no coincide con los datos, perdiendo información en el proceso.

Con respecto a los datos de demandas, hay que formatearlos con las reglas del ecosistema de la entidad, que son las ya vistas en la tabla de "inventarios" como la codificación de valores, el control de nulos, entre otros. De igual forma las demandas se envían y depositan junto a los datos comerciales vía fichero (Figura 4.51), es el último de los orígenes y cuenta con las mismas características que el de comerciales.

Figura 4.51 Ficheros demandas

Tanto la integración de los ficheros de los datos comerciales como los de demandas son automatizados para que el almacén pueda leer estos orígenes (Figura 4.52).

Figura 4.52 Automatización carga entidades de Demandas

La entidad más importante que existe en inmuebles se carga a partir de la tabla de "inventarios" principalmente, donde los desarrollos sufridos en su integración concuerdan con el diagrama Gantt (Tabla 4.9).Se trata de la KPI REOs (*Real Estate Owned*), como su nombre indica, se trata de los indicadores clave de desempeño (KPI) de los activos inmobiliarios que el banco ha adquirido debido a ejecuciones hipotecarias (REOs).

Tabla 4.9 Planificación integración entidad KPI REOs e informes

	2019	2020		2021		2022		2023	
Módulos		S1	S2	S1	S2	S1	S2	S1	S2
Integración de la entidad: KPI REOS									

Tareas propias | Tareas de equipo | Tareas de terceros

De esta tabla beberán los cuadros de mandos y reportes, además de los Data Marts, y también se generarán los informes para el usuario de contabilidad. Este proceso se inició previa a mi participación porque es de donde también se cargaba el pequeño reporte de OBIEE. Su integración es más sencilla que la de "inventarios", aunque también es más interesante, consta de tres sesiones enlazadas a tres mapping (Figura 4.53).

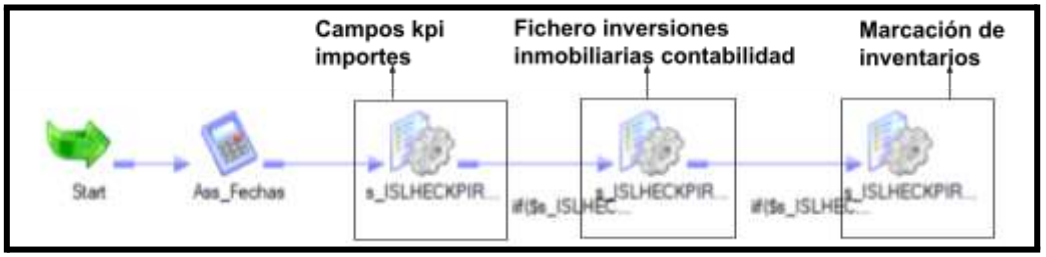

Figura 4.53 Sesiones de la Kpi

La primera calcula ciertos campos kpi de la Figura 4.54 a partir de la tabla de “inventarios” junto con algunas tablas de CEGID, en un simple vistazo puede verse que son importes contables por lo que se espera el uso de agrupaciones y, por supuesto, todos los campos de “inventarios” que serán campos directos que se integran en esta.

Etiquetas de fila	Valores Cuenta de IdUnico	Suma de TotalCosteAdquisicionOriginal	Suma de TotalCosteAdquisicion	Suma de Neto Contable	Suma de Neto Contable Caja	Suma de ValorContableBruto	Suma de CorrInicial	Suma de CorrPosterior	Suma de CorreccionTotal	Suma de Deterioro Filial	Suma de Deterioro Subestandard
Variación	0	0	0	0	0	0	0	0	0	0	0

Figura 4.54 Campos kpi que conforman el informe de inmuebles

La integración sigue los mismos patrones que los vistos en la tabla “inventarios”, seleccionando todos los campos como flujo principal, pues ambas van a tener el mismo número de activos, se va uniendo la información a través de left joins al resto de información que ya va siendo transformada con agregaciones y cambios, para en última instancia, realizar las últimas modificaciones pertinentes (Figura 4.55).

Figura 4.55 Mapping de la primera sesión cálculo de los campos KPI

Algoritmo 4.6: Diseño de ETL para añadir información usando UPDATE.

```
1 SELECT INV.*, CAMPOKPI1,...
2 FROM TI.ISLHECINVUIIDIA_V
3 LEFT JOIN( SELECT CAMPOPK1...) ON...
```

Esta es la sesión que se empezó previamente a mi participación, pero como a lo largo del tiempo se crearon nuevos campos contables opté por modificarla en vez de crear un nuevo mapping, en un intento de tener estos valores contables todos calculados en uno mismo y por ser una ETL mucho más pequeña comparada comparada con "inventarios".

La segunda sesión (Figura 4.56) carga el importante fichero de inversiones inmobiliarias mencionado anteriormente. Este fichero actualiza los mismos campos contables que los descritos en la sesión 1, es decir, que la primera sesión se usa para los activos procedentes de la inmobiliaria, mientras que la segunda es para los activos de inversiones inmobiliarias, así pues, se une a la propia tabla KPI. Este método es el mismo que se usaba en la sesión dos de "inventarios" pero de una forma más esclarecedora ya que el mapping entero entra en una captura, donde la primera sesión insertaba el "perímetro" y la segunda y sucesivas actualizaban los campos con la información faltante, al ya tener los activos cargados tan solo modifica los campos importes con UPDATE para no producir errores de duplicados.

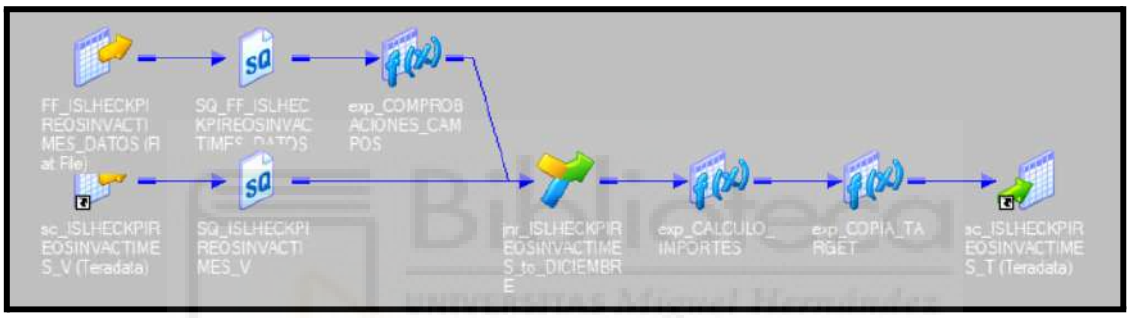

Figura 4.56 Mapping de la segunda sesión de la KPI

Por último, la tercera sesión es el cálculo del campo más importante del modelo, anque antes hay que poner un poco de contexto porque sin él los datos sólo serían números. Los ejes del cubo OLAP de inmuebles representan el tiempo y también el campo del ciclo de vida del inmueble. Un inmueble cuando se da de alta y entra al DW está en el estado de "vigentes", es decir, sin vender, de este estado puede cambiar al estado "vendido" o al estado de "provisionadas". Este último estado simplificando mucho, se refiere a un hábito contable de gestionar el riesgo, por ejemplo, imaginemos que el banco ha dado un dinero por una casa y resulta que cuando se queda la casa para recuperar el valor total del préstamo mediante una tasación del inmueble prevé que no va a recuperar ese dinero, lo que hace es guardar (provisionar) el importe restante que faltaría, y una vez vendido ver si ha existido alguna pérdida o no (Figura 4.57).

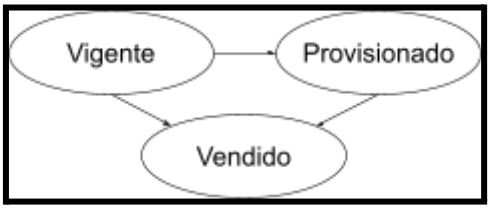

Figura 4.57 Estados de un activo

Sobre estos tres estamos los activos se van a clasificar en tres inventarios (Figura 4.58):

- *Inventario de activos inmuebles "normales":* corresponde a los activos que no están en los otros dos inventarios.
- *Inventario de activos inmueble "TR"*: aquí se engloban los activos que están alquilados o de VPO y por esa situación tienen el derecho de tanteo y retracto, una medida legal para la persona que habita en ese inmueble en el que en caso de venta se le ofrezca las mismas condiciones que al comprador, eso corresponde al derecho de tanteo o una vez vendido el inmueble poder adquirirlo reembolsando al comprador original el dinero que sería el derecho a retracto.
- *Inventario de activos "fallidos"*: Este inventario es muy pequeño y describe los inmuebles que llevan tanto tiempo sin venderse que el banco contablemente los da como pérdidas, no son tanto activos provisionados que no se vaya a recuperar su valor, sino que son activos que nadie los quiere, por tanto, siguen teniendo los estado de vendidos, provisionados y ventas por si se diera el caso en el que alguno se vende.

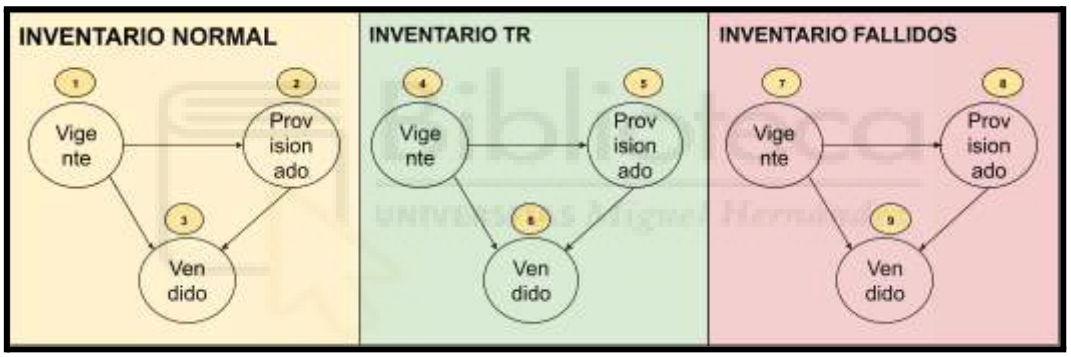

Figura 4.58 Inventarios de un activo

El marcado de estos estados y de los inventarios se realiza en el siguiente mapping (Figura 4.59). El diseño de este cálculo es el siguiente, se lee nueve veces (una vez por cada inventario) la tabla "kpi" y se filtra en cada lectura un estado de cada inventario, por ejemplo para el estado de vendidos del inventario normal, se buscará que el importe de venta sea mayor que 0, que exista una fecha de venta o fecha de escritura diferente de null, etc, y un campo que indique que no es ni "*TR*" ni "*Fallido*". En la siguiente transformación se marca este flujo con un simple dígito: '3' que corresponderá a un activo vendido de este inventario, (en la Figura 4.57 ya se describía este código), así sucesivamente para conseguir el resto. Una vez marcados tan solo hay que volver a unir todos los flujos en uno, usando un elemento de tipo unión y actualizar este campo en la tabla destino. Por último, hay un elemento de tipo agregación después de la unión, esta es una casuística especial que surge debido a activos que estaban vendidos y volvían al estado de vigentes por diferentes razones, lo que hacía que para un mismo activo haya dos registros uno en vigentes o provisionadas y el mismo pero en el estado de vendidos, con esa agregación lo

que se realiza es un MIN() sobre el código de vida del inmueble para obtener el activo "vigente", cuando se dé esta casuística.

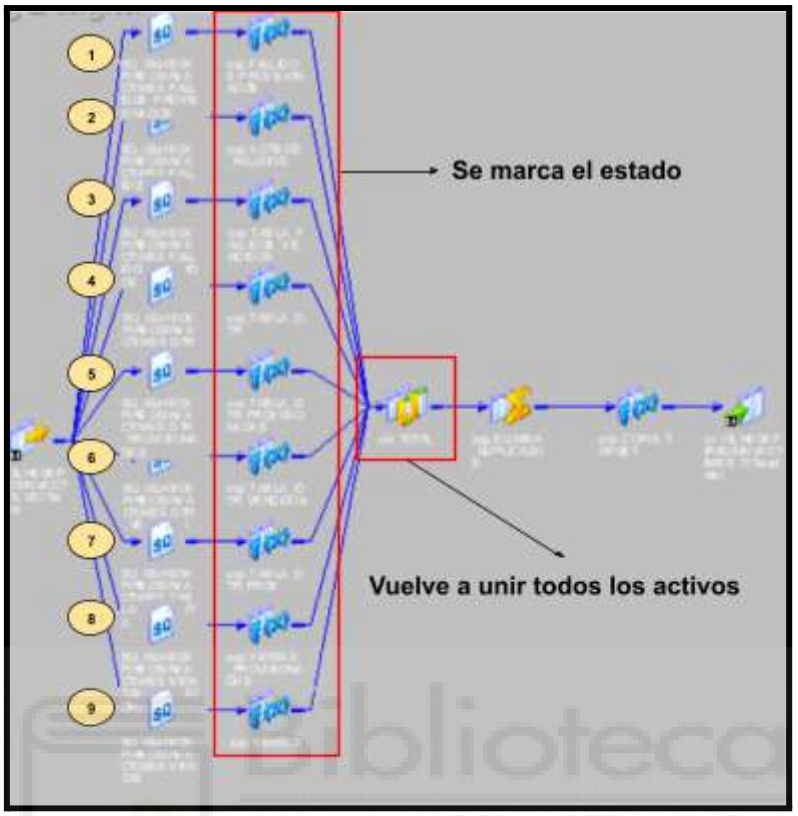

Figura 4.59 Mapping tercero con el marcado de los inventarios

Ya solo faltaría generar los informes, así que filtrando por el nuevo campo y la marca de tiempo, se pueden enviar los datos de cada estado de los inventarios (Figura 4.60)

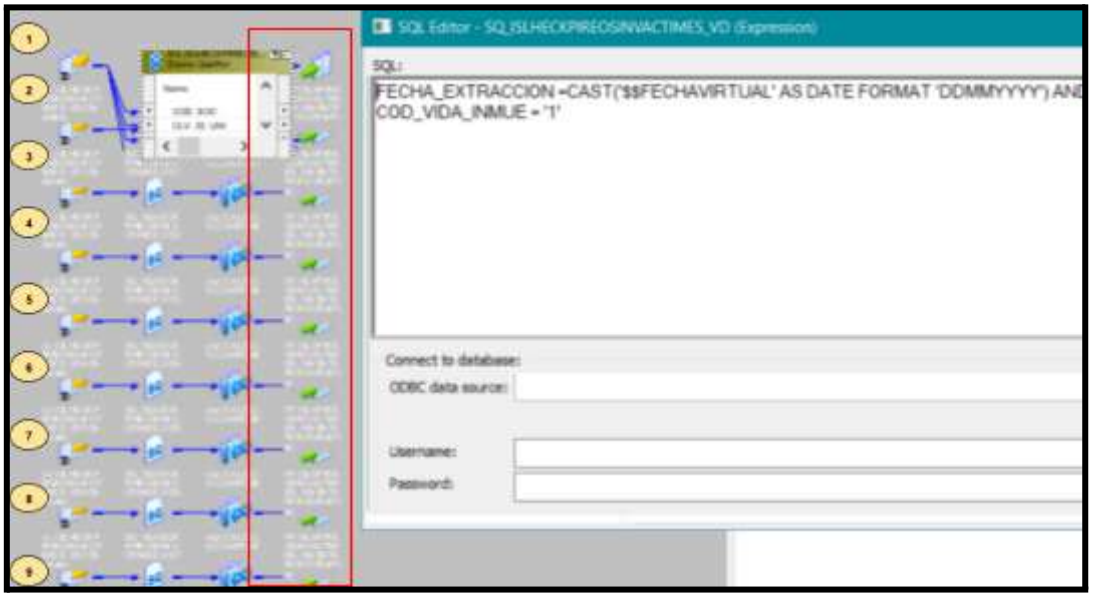

Figura 4.60 Generación de los informes de los inventarios

Se usa este caso (el parámetro de *$$FECHAVIRTUAL* Figura 4.62) para explicar los parámetros y variables de los procesos de integración. Powercenter admite el uso de parámetros a través de archivos externos alojados en el servidor (para esto se usa el cliente *FTP Filezilla*). Su funcionamiento es muy sencillo, se vincula el archivo en formato .ini en las propiedades del workflow (Figura 4.61), este archivo (Figura 4.62) contiene todos los parámetros que se quieren usar.

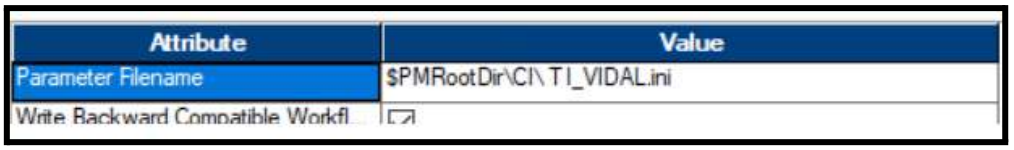

Figura 4.61 Ruta del fichero de parámetros

Figura 4.62 Fichero de parámetros

Como en ocasiones anteriores, se automatizan tanto el proceso de carga de la tabla "Kpi" como los informes que se envían (jobs AFT) (Figura 4.63).

Figura 4.63 Automatización de la KPI e informes

Para cerrar las entidades mostradas en el apartado de diseño que participan en el almacén, se van a aportar algunos datos de las mismas (Tabla 4.10).

Tabla 4.10 Planificación entidades detalle del DW de inmuebles

	2019	2020		2021		2022		2023	
Módulos		S1	S2	S1	S2	S1	S2	S1	S2
Integración de la entidad: BIENES									
Integración de entidades secundarias: - ENTIDAD ANEJOS - ENTIDAD ALQUILER - ENTIDAD GEOLOCAL									
Integración de la entidad : TASACIONES									
Integración de la entidad : GASTOS									

Tareas propias | Tareas de equipo | Tareas de terceros

La inclusión de estas tablas en esta memoria es para completar el modelo y que se pueda visualizar el diseño completo del DW, ya que su integración es análoga a las que ya se han contado y no aporta más novedades a la estructura, así pues, se darán solo algunos detalles y cualidades de las mismas:

- DIM BIENES: Es una dimensión que guarda los acuerdos de deuda que permiten hacer una traza de los activos de los inventarios, un bien puede pertenecer a varios activos, por ejemplo si se da el préstamo sobre una casa que tiene un garaje y un trastero, después estos tres activos se darán de alta por separado (si se pueden vender por separado) aunque parten del mismo bien. Su creación se realizó por terceras personas, aunque con la migración de los acuerdos de deuda del servidor financiero a otro servidor, se tuvieron que renombrar en un nuevo campo que es el momento en el que la modifiqué. La inclusión de una dimensión en el modelo también da ocasión a explicar cómo se carga, recordemos que una tabla de dimensión guarda información estable que no representa ningún tipo de ventaja el historificarla, por eso, su carga se hace mediante el método *SCD (Slowly Changing Dimensions)* de tipo 2 [62], en este método de carga si el registro ya existe en la tabla no se realiza ninguna acción, si el registro cambia respecto a uno que ya existía, se marca el antiguo con un código y se ingresa el nuevo, por último, en el caso que no exista simplemente se inserta.
- HEC ANEJOS: La tabla "hecho de anejos" proporciona al inventario el detalle de los activos padre respecto a los hijos, los activos son segregados si se pueden vender por separado, lo que hace esta tabla es dar el detalle del activo "padre" en el ejemplo anterior el padre sería la casa y el garaje y el trastero los hijos.
- HEC ALQUILER: Proporciona detalle sobre los activos alquilados, la información que carga a la tabla de inventarios son campos como el indicador de si está alquilado o el estado del alquiler como: Alquilada, Disponible, con opción a compra, etc.

- DIM GEOLOCAL: Con igual carga que la dimensión de bienes, integra a la tabla principal de “inventario” las coordenadas de los inmuebles.
- HEC TASACIONES: Contiene el detalle de todas las tasaciones realizadas sobre los inmuebles, tanto directas, método de valoración que se basa en la comparación de un bien con otros bienes similares puestos a la venta, como estadísticas en las que se utilizan modelos matemáticos y estadísticos para estimar el valor aproximado de un inmueble. Esta entidad integra información en la de “inventarios” como: la última fecha de tasación, el último importe de tasación, el nombre de la empresa tasadora, etc.
- HEC GASTOS: Detalla todos los gastos que tiene un inmueble hasta su venta, se integran algunos de estos gastos en la de inventarios como los de gestoría, registro , ITP, entre otros.

4.1.3.- Data Marts y desarrollos BI

Los Data Marts de inmuebles (Tabla 4.11) son: el de Salesforce, que es de dónde se informarán los cuadros de mandos de visualización de los datos, y el de garantías, que junto con el de DW de deuda se utilizará para tareas de tipo predictivas del departamento de riesgos de la entidad.

Tabla 4.11 Planificación Data Marts de inmuebles

	2019	2020		2021		2022		2023	
Módulos		S1	S2	S1	S2	S1	S2	S1	S2
Desarrollo Data Mart Salesforce									
Desarrollo Data Mart de Garantías									

Tareas propias | Tareas de equipo | Tareas de terceros

Salesforce se aloja en el propietario SFA y su automatización es diaria, se llama así porque la herramienta BI que visualizará estos dato será Salesforce, se compone de las entidades:

- SFA ACTIVOS: Se carga ciertos campos kpi de la tabla del DW Kpi.
- SFA ALQUILERES: Informará los indicadores de alquileres.
- SFA GASTOS: Proveerá el desplegable de los gastos
- SFA OFERTAS: Detalle de las ofertas realizadas en la pestaña de comercialización.
- SFA DEMANDAS: Visualización de las demandas sobre los activos

Estos procesos van a tener muy poca complejidad (Figura 4.64). Se implementan en colaboración con el desarrollador BI y los usuarios finales seleccionando los campos kpi que se mostrarán en los dashboards, su origen suele ser la tabla destinada a este ecosistema

(la entidad Kpi Reos) junto alguna otra tabla del Data Warehouse que detalle el panel a montar.

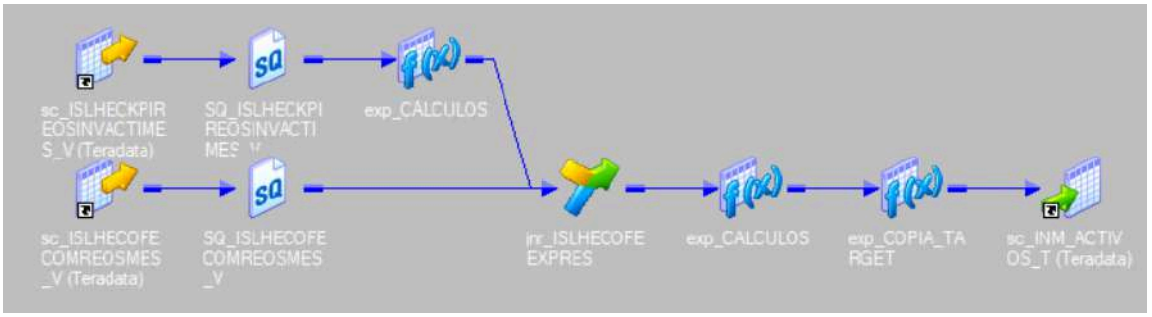

Figura 4.64 Mapping de carga de la SFA de ACTIVOS

Las transformaciones realizadas en estos procesos, buscan facilitar el trabajo al desarrollador BI, de hecho la más habitual es cambiar todos los indicadores de tipo S o N, a tipo 1 y 0 (Figura 4.65) o descodificar algún campo para que se visualice como texto.

```
IIF(IN_IND_SN_ES_PROM = 'S', '1', IIF(IN_IND_SN_ES_PROM = 'N', '0', ''))
```

Figura 4.65 Transformación de indicador SN a indicador 01

La figura 4.66 muestra un ejemplo visual que representa algunos de los campos kpi de las entidades, contiene varios apartados como un mapa para localizar los activos en venta muy parecido al de un portal web inmobiliario, este mapa se carga a partir de los campos de coordenadas de su origen estaba en la tabla “dim geolocalizacion”.

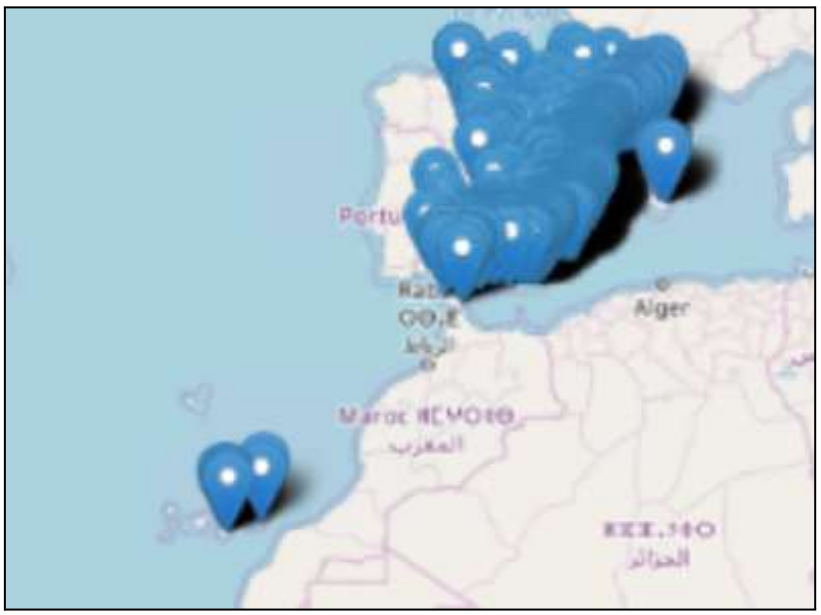

Figura 4.66 Salesforce mapa de activos

Clicando sobre alguno de los activos mostrado en el mapa, o si se localiza a través del buscador del panel, el cuadro de mando mostrará el detalle del mismo (Figura 4.67) donde, en diferentes pestañas, se encontrará el resto de información perteneciente a las tablas del propietario SFA.

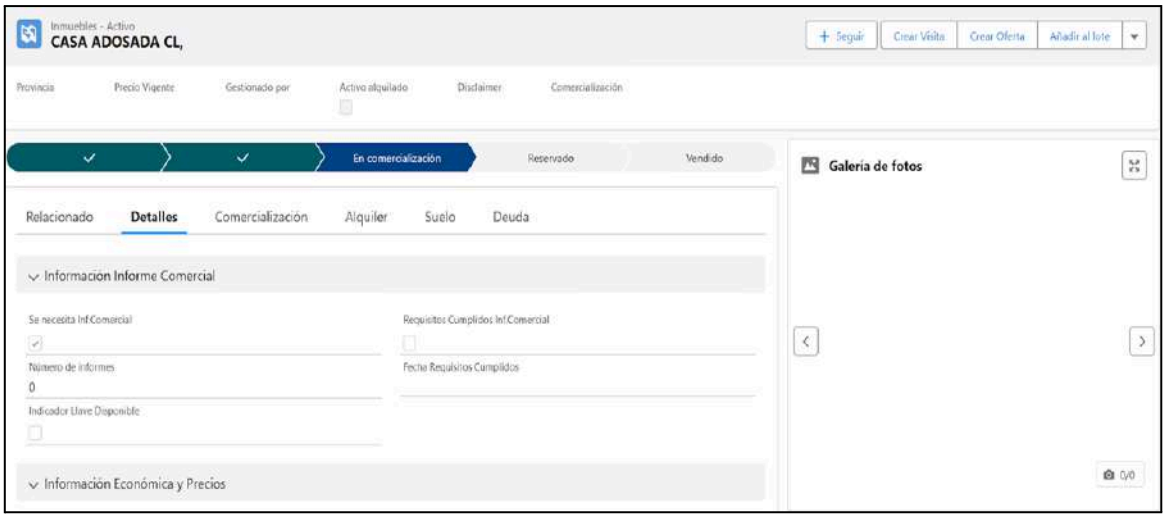

Figura 4.67 Detalles Salesforce Activo

En la Figura 4.68 y Figura 4.67 en los diferentes desplegables se pueden consultar los campos kpi que llegan de la entidad SFA.ACTIVOS, además de otros, como el desplegable de Gastos de inmuebles que será informado a partir de la tabla SFA.GASTOS.

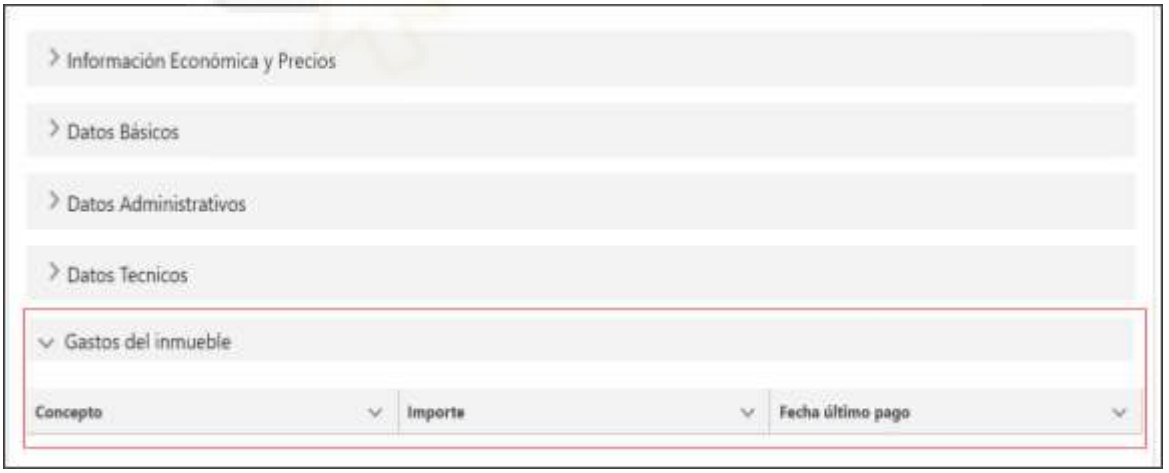

Figura 4.68 Desplegable panel detalle activo Salesforce

Navegando por estas vistas se puede llegar, por ejemplo, al apartado de comercialización, donde se puede observar el detalle de las ofertas que se cargan a partir de la SFA.OFERTAS, o la pestaña contigua que muestra información de alquileres que se cargan con la tabla SFA.ALQUILERES (Figura 4.69).

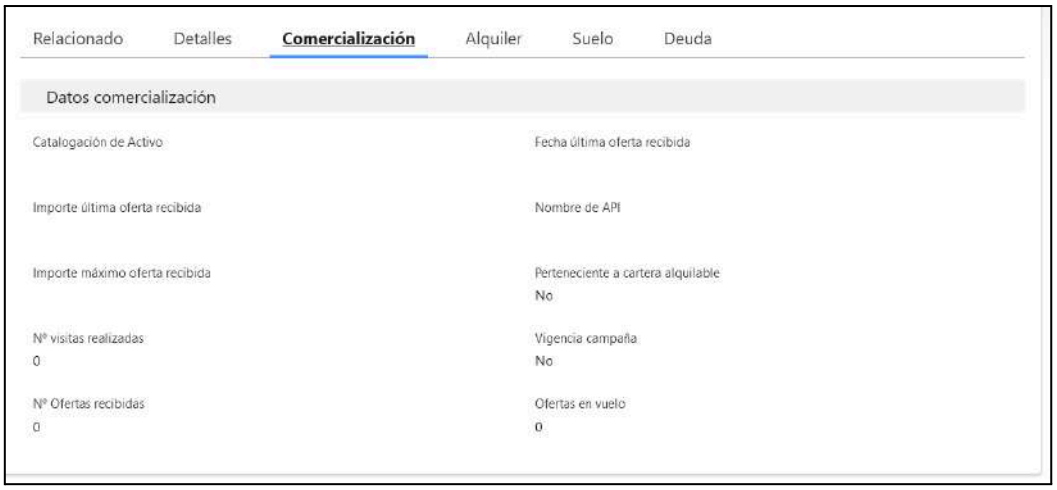

Figura 4.69 Pestaña de comercialización activos Salesforce

Cambiando al panel "judiciales" se pueden visualizar las demandas de los inmuebles, que se informará a partir de la tabla SFA.DEMANDAS (Figura 4.70).

Figura 4.70 Panel de demandas en Salesforce

En cuanto al Data Mart de garantías, la finalidad que se quería analizar era el cálculo de los costes que ha tenido un activo, desde las costas judiciales sumado al mantenimiento y el coste de venta, para ver, una vez vendido, si ha habido una caída de valor respecto al préstamo concedido (Figura 4.71).

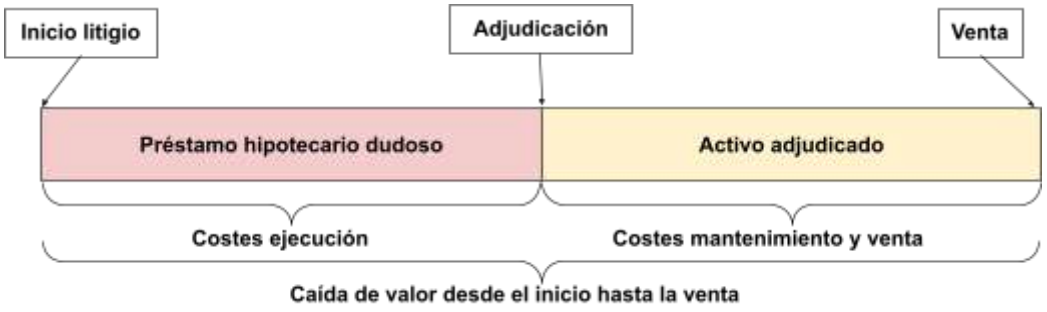

Figura 4.71 Explicación Data Mart Garantías

Este Data Mart de garantías, se compone de las entidades:

- HECHO GARANTÍAS PLAZOS: Esta tabla guarda el número de plazos en el que está un acuerdo en morosidad antes de entrar al proceso judicial. Los orígenes de

esta tabla son de la tabla Kpi del DW de inmuebles y de la tabla principal del DW de deuda que contiene las fechas de los acuerdos dudosos.

- HECHO GARANTÍAS TASACIONES: Guarda las tasaciones que se van recogiendo para predecir los costes que generará el inmueble, se informa a partir de la tabla de Tasaciones del DW.

- HECHO GARANTÍAS GASTOS: Guarda los gastos asociados y los sumatorios desde el inicio del proceso judicial hasta la venta, su información parte de la tabla de gastos del DW.

- HECHO GARANTÍAS PERÍMETRO: Almacena la información más importante de las otras tablas y contiene el perímetro seleccionado de inmuebles que parte de la tabla kpi del Data Warehouse además de ser la tabla a consultar mayormente (a no ser que necesiten más detalle) por el departamento de riesgos.

La integración en el propietario BI de este Data Mart es muy sencilla y sigue el mismo diseño que el del Data Warehouse de inmuebles, debido a que los campos de las tablas "tasaciones", "gastos" y "perímetro" en su mayoría son campos directos que provienen del almacén de inmueble, por lo tanto no se entrará en detalle de sus integraciones.

Aunque de este DM de garantías se hizo un pequeño panel en Qlik en el que básicamente se mostraba la información que tenía la tabla principal del submodelo (la tabla GARANTIAS PERÍMETRO Figura 4.18 del apartado de diseño). Esta es una información más bien destinada para el departamento de riesgo que está automatizada con un periodo mensual en el que se usan sus tablas como orígenes para hacer un modelo predictivo con los activos ya vendidos, así como consultas sobre el propio modelo.

Por último, sobre el informe de OBIEE y la figura 4.3 en la que se mostraban los roles de usuario, donde el usuario de contabilidad accedía a Qlik para ver ciertos reportes, estos se muestran en los paneles ilustrados en las figuras 4.72, 4.73 y 4.74, y su información procede de la tabla Kpi de integración.

Figura 4.72 Panel que calcula los campos kpi de la tabla Kpi según inventario y tiempo

Figura 4.73 Panel con el importe de ventas respecto al año anterior

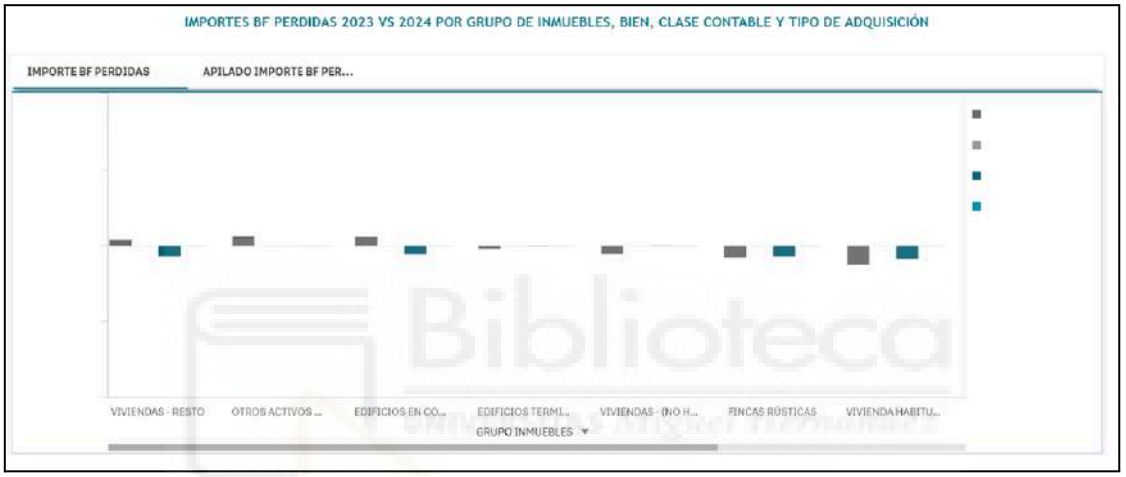

Figura 4.74 Panel Importe beneficio-Pérdidas en los ejercicios 2023 y 2024

A modo de resumen y para finalizar se vuelve a presentar el modelo de inmuebles (Figura 4.75), para tener una comparación de las diferentes partes implementadas del almacén a día de hoy.

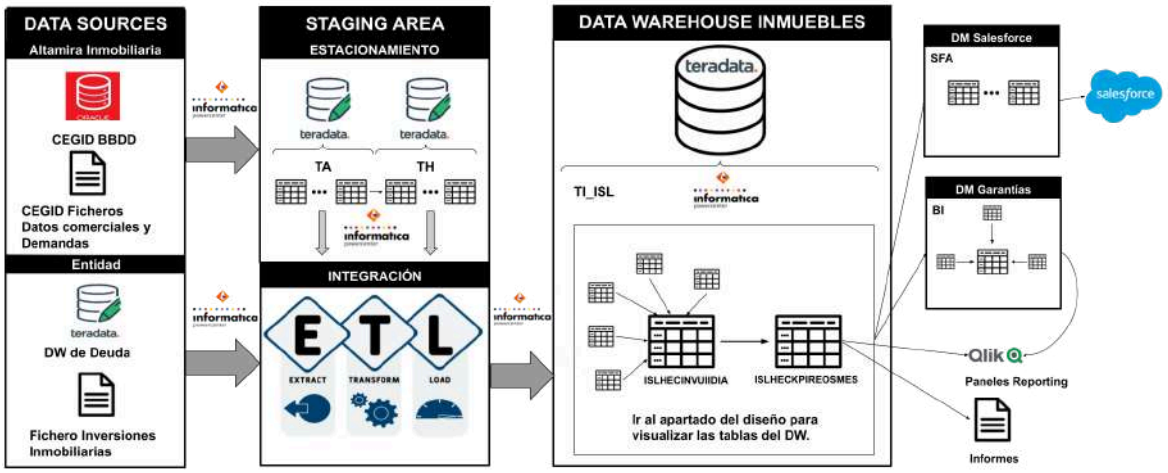

Figura 4.75 Modelo Inmuebles a día de hoy

4.5.- PRUEBAS

El apartado de validación es uno de los puntos más importantes a tener en cuenta, especialmente cuando se trabaja en la integración de datos. Es el mecanismo para saber que se están cumpliendo los objetivos y tener "feedback" de los usuarios. Además de ser una parte fundamental en la documentación de carga del Data Warehouse y depuración en caso de fallo del proceso ETL, para finalizar también con el último punto del diagrama (Tabla 4.12)

Tabla 4.12 Planificación de validaciones

Existen varias formas de validar si los datos se han cargado correctamente o comprobar el funcionamiento de los mismos, aunque dependiendo en qué parte del ciclo de vida estemos, si en pruebas o verificación donde se comprobará que las integración realizadas en el entorno de desarrollo cumplen ciertos requisitos de incorporación, o en la etapa de periodo garantía o test, que se situará en el entorno de producción y se enfocará más, en las validaciones de aceptación de los datos por parte del usuario.

4.5.1.- Pruebas de integración - Fase 5 del ciclo de vida del DW

Las pruebas de integración, según sus orígenes se pueden dividir en: las que tienen el mismo origen que el entorno donde se hospeda el Data Warehouse, y las que sus fuentes proceden de diferentes ecosistemas respecto al entorno del almacén.

Respecto a los mismos orígenes, todo proceso y por extensión todo campo del cual se quiera integrar su información, lleva su traducción de lenguaje Powercenter a consulta SQL, y viceversa. Por lo tanto, el proceso de validación se acorta enormemente si tanto los orígenes de la ETL como el destino se encuentran en Teradata recurriendo a nuestro caso práctico de la entidad. Este es uno de los motivos principales de por qué existe un paso intermedio que es el estacionamiento de la información antes de la integración, ahora tan solo debemos comparar el cálculos de los orígenes contra los datos cargados en la tabla destino, mediante consultas SQL.

Entre las pruebas que se pueden realizar, en la entidad se distinguen principalmente tres (como mínimo, existen más), que se han nombrado en ciclo de vida del Data Warehouse:

- Pruebas de carga: Estas pruebas consisten en dos validaciones de carga, primero ratificar que el perímetro almacenado (Nº de registros cargados) en la tabla destino respecto al Nº de registros esperados en la consulta SQL que carga la tabla destino son los mismos, así como en el tiempo empleado en la carga (Figura 4.76), en este caso no se busca tanto la optimización del proceso sino validar que no existen problemas a la hora de cargar los datos, es decir, comprobar que la ejecución de una sesión o workflow esté procesando más de una hora o se quede "colgado" esto nos indicaría que existe algún elemento "raro" en el proceso, no se ha configurado o se está ejecutando mal el método de carga, este tipo de pruebas se emplean en la integración de nuevas tablas como primera validación para ver que, al menos, el número de datos es correcto respecto a la consulta, y que no existe ningún tipo de error en la ejecución del proceso que se subirá posteriormente a producción.

  Un ejemplo de este tipo de validación sería una clausula simple de COUNT(*) (Algoritmo 4.7).

Algoritmo 4.7: Conteo de número de registros de la consulta que traduce Powercenter

```
1 SELECT COUNT (*) NREGISTROS FROM
2 ( SELECT * FROM FLUJOPRINCIPAL WHERE ...)CONSULTATRADUCCION
```

  debe ser igual al número de registros de la tabla destino (Algoritmo 4.8).

Algoritmo 4.8: Conteo de número de registros de la tabla destino

```
1 SELECT COUNT(*) NREGISTROS FROM TABLADW WHERE ...
```

  o una unión entre la consulta y la tabla destino mediante un inner join entrelazado por sus claves principales que devolverá los registros iguales al perímetro deseado (Algoritmo 4.9).

Algoritmo 4.9: Mismo número de registros entre la consulta y la tabla destino

```
1 SELECT
2 TABLADW.CLAVEPRINCIPAL, CONSULTATRADUCCION.CLAVEPRINCIPAL FROM
3 TABLADW INNER JOIN CONSULTATRADUCCION ON
4 TABLADW.CLAVEPRINCIPAL=CONSULTATRADUCCION.CLAVEPRINCIPAL
```

Figura 4.76. Carga de un proceso TA

- Pruebas de valor: Estas pruebas van destinadas al control de campos cargados con valor en la tabla destino del DW. Es decir, es un control de nulos que nos indica qué campos pueden tener deficiencias y se hace sobre la propia tabla cargada. Por

ejemplo si un campo está totalmente a nulo puede deberse a que ocurrió algún problema y que su integración se está haciendo incorrectamente. Hay que tener especial atención a aquellos campos que sin estar a nulos en origen, son finalmente nulos, por ejemplo los campos de codificación o indicadores que si eran nulos se cargaban como "@", fechas que están informadas pero su valor es 01/01/2099, etc. La razón de hacer estas pruebas en el entorno de desarrollo a pesar de su simpleza, es que en grandes integraciones o en particular, cuando se va a modificar un proceso que ya está funcionando en producción, desconectar un campo en el mapping para realizar el cambio y luego no volver a acordarse de conectarlo es un problema que hasta al desarrollador más experto le puedo llegar a pasar. Un ejemplo de esta validación corresponde a una simple cláusula de SUM() que junto a un CASE (cláusula con lógica de tipo IF-ELSE) y un IS NULL (valores nulos) o el valor que represente el NULL, devuelve el número de nulos que tiene el campo en cuestión (Algoritmo 4.10)

Algoritmo 4.10: Validación de control de nulos

```
1 SELECT
2 SUM (CASE WHEN CAMPO1 IS NULL THEN 1 ELSE 0) END CAMPO1NULOS,
3 SUM (CASE WHEN CAMPO2 IS NULL THEN 1 ELSE 0) END CAMPO2NULOS,
4 SUM (CASE WHEN CAMPO3='@' THEN 1 ELSE 0) END CAMPO3NULOS,...
5 FROM TABLADW WHERE ...
```

- Pruebas de contraste: Las validaciones de contraste si que se focalizan en que los datos cargados sean los esperados, para ello se une la consulta que es la traducción de Powercenter con la tabla final donde están los datos cargados mediante un joiner. Para después comparar el campo o campos entre la consulta y la tabla final. Es relevante resaltar que las validaciones de contraste se realizan sobre los campos de las tablas de forma individual o en conjunto dependiendo de la dificultad de la integración del campo, en caso de que se validen campos que tienen un mismo origen o de cálculo directo, se pueden validar todos juntos puesto que no suelen representar graves problemas, su validación en SQL sería igual al apartado anterior de valor, es decir, con las cláusulas como SUM y CASE WHEN, pero esta vez uniendo la tabla origen a la tabla destino (Algoritmo 4.11). Para campos calculados o que tienen algún tipo de transformación la prueba se debe realizar individualmente para ese campo, uniendo otra vez la tabla destino pero esta vez a una subconsulta SQL (la traducción SQL de Powercenter de ese campo). Si se quiere validar un indicador de ocupaciones o que ese indicador se está cargando de forma errónea, una forma de validarlo sería el Algoritmo 4.12.

Algoritmo 4.11: Validación de contraste entre valores de la consulta y la tabla final

```
1 SELECT
2 SUM (CASE WHEN TABLADW.CAMPO1<>TABLAORIGEN.CAMPO1  THEN 1 ELSE 0) END
3 CAMPO1VALIDACION,
```

```
4 SUM (CASE WHEN TABLADW.CAMPO2<>TABLAORIGEN.CAMPO2 THEN 1 ELSE 0) END
5 CAMPO2VALIDACION...
6 FROM TABLADW LEFT JOIN TABLAORIGEN
7 ON TABLDW.CLAVEPRINCIPAL=TABLAORIGEN.CLAVEPRINCIPAL
8 WHERE ...
9
```

Algoritmo 4.12: Validación de campo individual entre la consulta y la tabla final

```
1  SELECT TABLADW.ACTIVO, TABLADW.IND_OCUPACIONES,
2  CONSULTATRADUCCION.IND_OCUPACIONES
3  FROM
4  ( SELECT CLAVEUNION, IND_OCUPACIONES FROM TABLAFINAL WHERE...) TABLADW
5  INNER JOIN
6  (SELECT FLUJOPRINCIPAL.CLAVEUNION,
7  CASE WHEN TABLAOCUPACIONES.OCUPACIONES='VERDADERO' THEN 'S' ELSE 'N' END
8  IND_OCUPACIONES
9  FROM FLUJOPRINCIPAL
10 LEFT JOIN
11 TABLAOCUPACIONES ON ...) CONSULTATRADUCCION ON
12 CONSULTATRADUCCION.CLAVEUNION=TABLADW.CLAVEUNION
13 WHERE
14 TABLADW.IND_OCUPACIONES<>CONSULTATRADUCCION.IND_OCUPACIONES
15
16
```

Cuando los orígenes están en otro ecosistema respecto al del repositorio, en este caso, aunque las pruebas serían idénticas a las anteriores, cambia la forma de validarlo al no poder realizar una consulta SQL contra la tabla del DW, pues los orígenes no están en la misma base de datos que el almacén, ejemplo de ellos sería el fichero de los datos contables de inversiones inmobiliarias que se carga en la tabla “kpi”, la información de los ficheros de datos comerciales recibidos u otras bases de datos diferente a Teradata en nuestro caso, entre otros.

La correcta validación de los datos, dependerá más de la perspicacia del programador a la hora de seleccionar la calidad de la muestra que se va a validar más que la cantidad de muestra que se escoja. Una muestra es un subconjunto de datos extraídos de un conjunto mayor, en nuestro caso algunas recomendaciones vendrían de seleccionar activos que estén informados lo máximo posible, activos de diferentes inventarios para obtener diferentes casos, también activos que su estado haya cambiado hace poco, por ejemplo inmuebles que acaban de entrar a los inventarios o que han cambiado de inventario como de vigentes pasando a provisionadas y a vendidas.

En definitiva todos esos activos que el programador intuya que pueda haber una problemática especial, para poder contrastarlos con los activos cargados en la tabla integrada del almacén. En este tipo de casos se pueden hacer muy buenas validaciones usando herramientas simples como un editor de texto en el caso de ficheros para las pruebas de carga donde se puede abrir el archivo y ver el número de registros existente. O

herramientas de ofimática como excel, donde se puede copiar la muestra de los orígenes y su igual de la tabla que carga el proceso, y realizar las pruebas de contraste y valor usando las propias herramientas que da la hoja de cálculo.

4.5.2.- Periodo garantía o test - Fase 7 del ciclo de vida del DW

En cuanto a la etapa de periodo garantía o test, a parte de volver a ejecutar las pruebas del apartado anterior realizadas para validar la integración, pero ahora en el entorno de producción, entorno que muchos casos al contar con datos reales más informados, presenta nuevos problemas y retos, también se incluye un catálogo de pruebas indicadas por el cliente a elaborar en este ambiente destinadas más a la calidad del dato en sí (como veíamos en los casos de uso las pruebas de integración van dirigidas a las ETLs, aunque se validen los datos, realmente se está validando que los elementos de integración han funcionado correctamente), puesto que esta información luego en gran parte formará parte de los campos *KPIs*. Entre las pruebas que se pueden realizar, en este catálogo podemos diferenciar:

- Pruebas de negocio.- representa el número de registros que cumplen una condición definida por el cliente.
- Pruebas estadísticas.- buscan valores en un campo que difieren en un tanto por ciento en el valor absoluto del valor medio.
- Pruebas de sonda.- representan los valores que no existen entre dos tablas.
- Pruebas de perfilado.- se focaliza en encontrar ciertos tipos de datos como número de negativos en los importes.
- Pruebas de versionado.- hace un histórico de los resultados en las reglas definidas con anterioridad.

Todo este abanico de reglas de calidad se recogen y una vez que el cliente ha dado su aprobación, se solicitan mediante un documento técnico para que sean creadas en un reporting por la herramienta de Qlik que se ejecutará todos los días (Figura 4.77) después del proceso ETL al que vaya asociado.

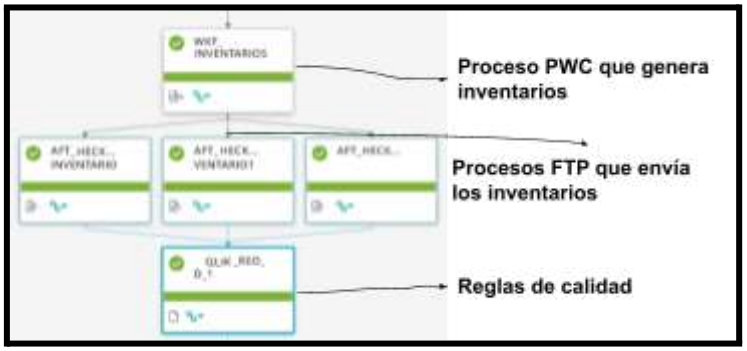

Figura 4.77 Control -M. Automatización reglas de calidad.

A continuación, se va a abordar un ejemplo de esta práctica para el campo clave cif inquilino de la tabla "kpi" de inmuebles en el que se aplicará una regla de perfilado (Figura 4.78).

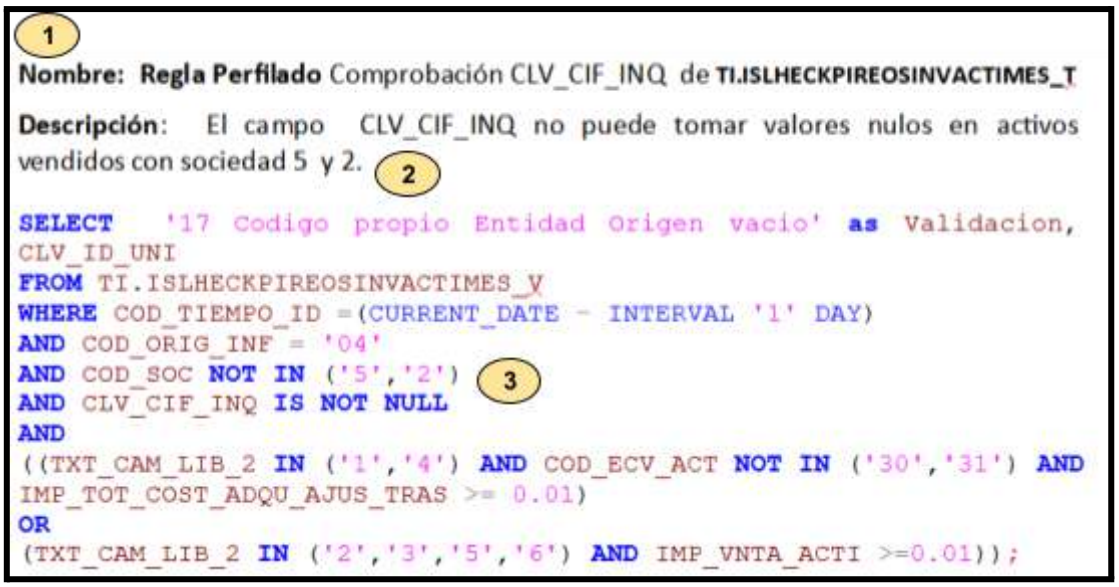

Figura 4.78 Documento con las consultas de reglas de calidad

(1) En esta regla de perfilado que es parecida a la prueba de integración de valor (recordemos que esas pruebas nos alertaban si un campo tenía alguna problemática sobre si estaba informado o no), (2) en este caso, se da un paso más y específicamente se busca el número de activos con el campo clave cif inquilino informado, y que (3) cumple con ciertas condiciones que ha hecho saber el cliente, que son los activos vendidos y los activos sin vender pero que tienen el importe total de adquisición en positivo, además de no pertenecer a ciertos códigos de sociedad como son el 5 y el 2.

La regla de calidad vista desde el apartado de visualización de Qlik se traduce en la Figura 4.79, donde se puede observar el campo sobre el que se hace la regla (1) y una breve explicación de la condición del mismo (2) (si se pincha sobre este campo aparecerá la consulta OLAP de la Figura 4.78 aunque es una información destinada para usuarios más técnicos o para el ingeniero de datos), además de cómo ha evolucionado el perímetro (3) de la regla a lo largo de los meses.

En este caso se ha filtrado por cada último día del mes en el propio Qlik, es decir, se ha hecho un "roll-up" en el cubo OLAP, (ya se percibe en la consulta pero tenemos tres ejes, el del tiempo, el de los inventarios vendidos y vigentes, y los campos kpi por los que se filtra). Aunque se ejecuta todos los días, su validación es cien por cien porque no hay discrepancias respecto al perímetro total, pero en caso contrario, que no se hubiera cumplido la regla, habría bajado del 100 por cien, y en este caso se habría enviado un correo al grupo de Inmuebles para alertar que algo está sucediendo.

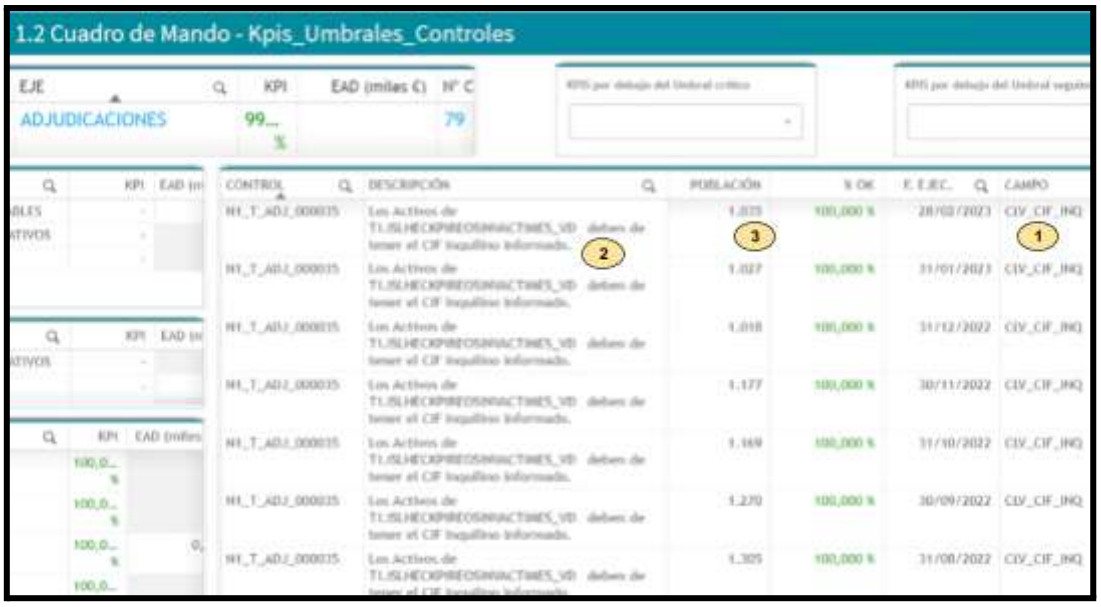

Figura 4.79 Qlik reglas de calidad

También se puede encontrar un histórico del perímetro y su porcentaje de cumplimiento en las diferentes gráficas (Figura 4.80) dando como resultado una idea rápidamente de la evolución de la regla de calidad.

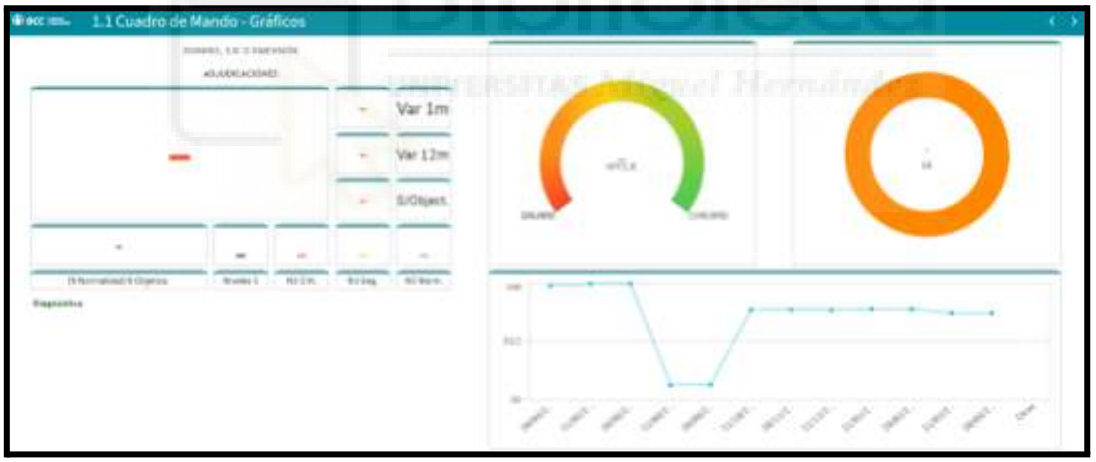

Figura 4.80 Gráficas reglas de calidad sobre la marca de tiempo

4.6.- IMPLANTACIÓN

La implantación de las ETLs en el entorno de producción, está estrechamente relacionada con las pruebas y los test o período de garantía , de hecho en el ciclo de vida se sitúa entre ambas y es el mecanismo mediante el cual se van a probar los procesos de integración con datos delicados de la empresa, a su vez tiene repercusiones en la fase de mantenimiento. Dado que la fase de implantación debe seguir una serie de pasos “especiales” se va a

explicar como es el recorrido que, en mi caso como desarrollador, debe recorrer para implementar un proceso en producción. (Figura 4.81)

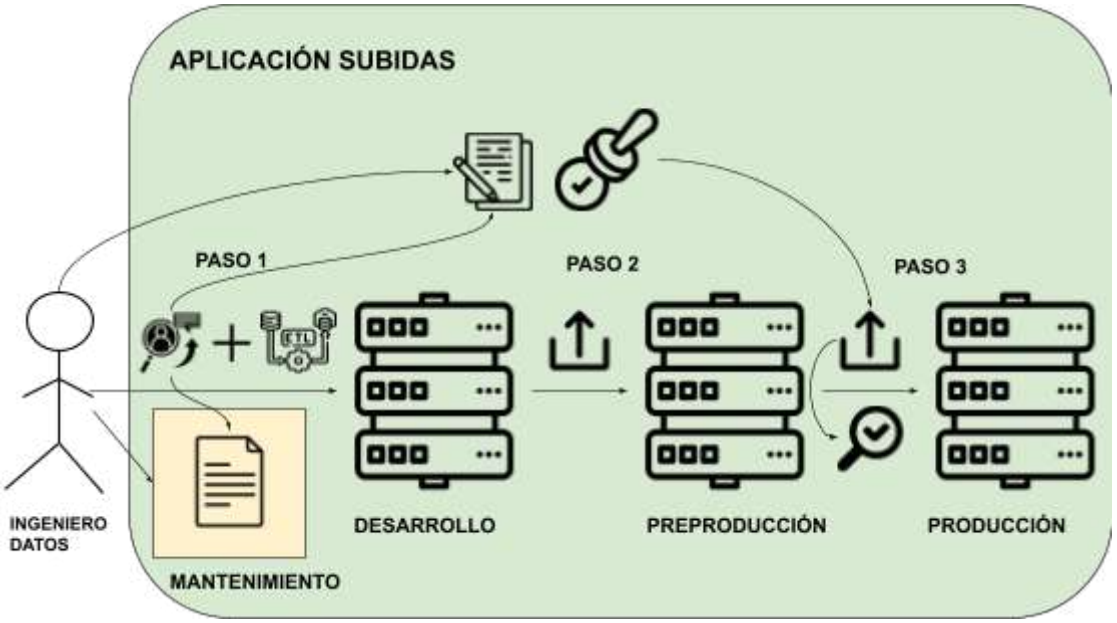

Figura 4.81 Proceso de subida de una ETL

La entidad dispone de una aplicación propia para la subida de desarrollos entre los diferentes entornos, el funcionamiento de esta aplicación no tiene tanta importancia como la funcionalidad que desempeña, ya que su objetivo es dejar constancia y una trazabilidad de los procesos que son subidos entre los diferentes ecosistemas, dando cabida a los siguientes pasos:

1. La persona encargada de subir el proceso debe crear en esta herramienta una referencia (es un código único generado para cada subida que funciona como clave para encontrar la documentación y el workflow al cual se asocia en casos posteriores), a la cual se la asociará el workflow o ETL que se desea implantar en producción. Esta referencia es necesaria para que otras personas tengan la documentación necesaria, y en caso de fallo lo puedan arreglar, por ello se deja plasmada en un archivo que se aloja en un directorio común de la empresa con la referencia del proceso subido.

2. Ahora se cambia del entorno de desarrollo al de preproducción, este nuevo entorno en el caso práctico que se expone no se suele usar frecuentemente para la ejecución de integraciones, sino que más bien funciona como un medio intermedio (una especie de "*staging area*" para procesos Powercenter) para que en el caso de que el proceso en producción por algún motivo se corrompa solo se deba volver a subir el mismo proceso desde este entorno.

3. Por último, el puente que se hace entre la fase de pruebas del ciclo de vida y la implantación, para subir al entorno de producción se deben adjuntar las pruebas de integración obligatoriamente de esa fase, a estas pruebas también se las añade la referencia de la subida. En esta fase también se validará por parte de la aplicación de la entidad de forma automática, que los apartados de buenas prácticas en Powercenter descritos en el capítulo 3 de esta memoria se han llevado a cabo y las sentencias DELETE del workflow tienen una cláusula WHERE. Se debe hacer una pequeña mención a este último punto, como se indicó en el apartado práctico del *staging area* se mostraba como a los procesos del propietario TH, se les podía incorporar un ventilado en una zona del workflow dedicado a ello (PostSql), en prácticamente todos los procesos existe otro tipo de DELETE que se sitúa en el apartado superior a éste, en PreSql (Figura 4.82)

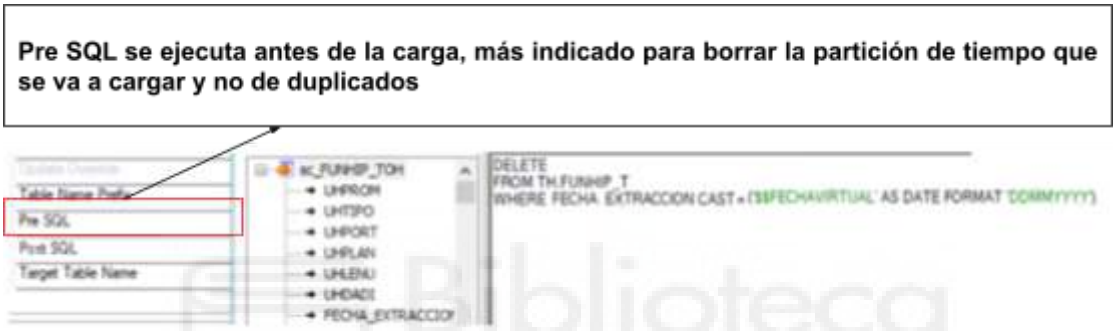

Figura 4.82 Pre Delete de la partición cargada

El Algoritmo 4.13 se usa en todos los desarrollos para eliminar los datos de la marca de tiempo que se está cargando, por ejemplo, si vamos a ejecutar el proceso marcando esa partición con la fecha 31052024 (31/05/2024), se creará esa partición de tiempo en la tabla del almacén, si hubiera que realizar una modificación en el proceso y se vuelve a ejecutar, como la fecha de la marca de tiempo es siempre una clave principal en la tabla destino, nos aparecerá el error de duplicados y fallará el proceso, ahora bien, si se borra la partición de tiempo con fecha 31052024 antes de hacer una nueva carga, al volver a ejecutar el proceso se puede volver a ingresar los datos sin problemas, por esta razón es que este borrado se realiza en el apartado de Pre SQL, porque borra los datos de esa partición de tiempo que es la misma que se va a cargar en el repositorio.

Algoritmo 4.13: Borrado de una marca de tiempo

```
1 DELETE * FROM TABLADW WHERE CAMPOMARCATIEMPO = ( '$$PARAMETROTIEMPO' AS
2 DATE FORMAT 'DDMMYYYY')
```

Una vez realizada la implementación final del proceso, en caso de tratarse de desarrollos que para la explotación del Data Warehouse como el que se vió en el apartado 2, se crearán los respectivos manuales, ya sea en caso de simples consultas OLAP para usuarios más

técnicos, o cómo sería el procedimiento para la ejecución de informes desde la propia aplicación web de la entidad como el visto en el ejemplo del "*chiringuito*". También se tendrá que pedir acceso al departamento de ciberseguridad para que los usuarios finales puedan acceder tanto a la ejecución de los informes como a las bases de datos donde estén alojadas las tablas del repositorio que van a explotar.

4.7.- MANTENIMIENTO

El objetivo final de todo desarrollo es automatizarlo, ya sea para su ejecución diaria, mensual o en una frecuencia determinada, por esta razón el mantenimiento va ligado a la automatización ya que al ejecutar procesos de Powercenter sin supervisión pueden fallar por cualquier error, desde una desconexión puntual en los servidores, hasta errores en los datos de lectura en orígen, pasando por desarrollos que no contemplen todos los casos.

La entidad contempla dos mantenimientos, críticos y no críticos. Los críticos se refieren a la ejecución de ETLs o procesos de Powercenter que, en caso de fallo, puedan producir paradas hasta la próxima jornada laboral. En este caso estarían todas las integraciones para la contabilidad de los inmuebles, tanto desde los orígenes hasta su carga en el almacén, este mantenimiento se lleva a cabo por el departamento de "guardias" como su nombre indica es un departamento dedicado a tratar procesos fallidos a cualquier hora del día. Siguiendo el diagrama de la (Figura 4.83), imaginemos que la cadena que carga el staging area de inmuebles que empieza a ejecutarse en la madrugada (aunque el fallo se produzca en horario laboral también es gestionado por la guardía) y por alguna razón uno de ellos falla.

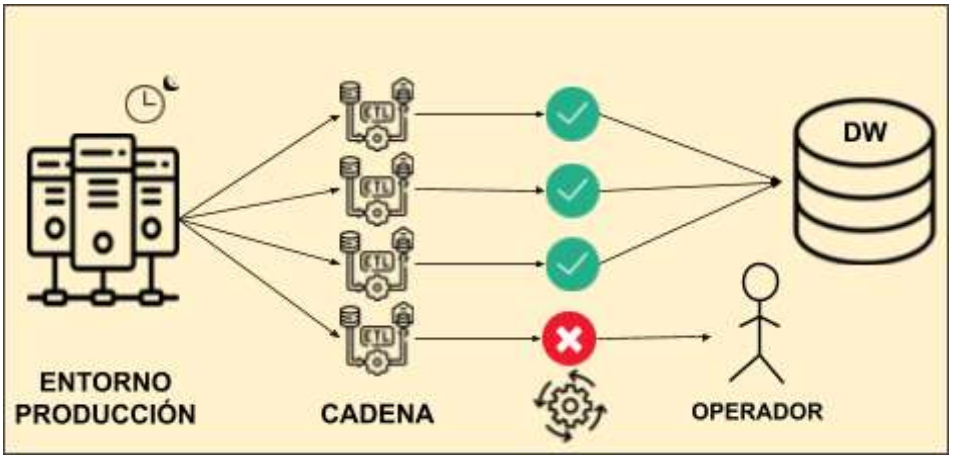

Figura 4.83 Proceso generación de una incidencia por fallo

Se genera una incidencia, entendiendo este fenómeno como un evento no planeado que interrumpe o reduce el proceso de carga, que es asignada automáticamente por el gestor de incidencias a un técnico de guardia. A la hora de abordar la incidencia este técnico tendrá varias opciones, ir a la herramienta Powercenter donde está a alojada la ETL y usar los

metadatos (Figura 4.84) de la misma para intentar averiguar si el problema ha sido un pequeño error por una modificación reciente, ya que Powercenter permite el versionado de sus procesos y pone a disposición del programador un comparador de versiones con los cambios realizados de una forma aproximada entre ellas.

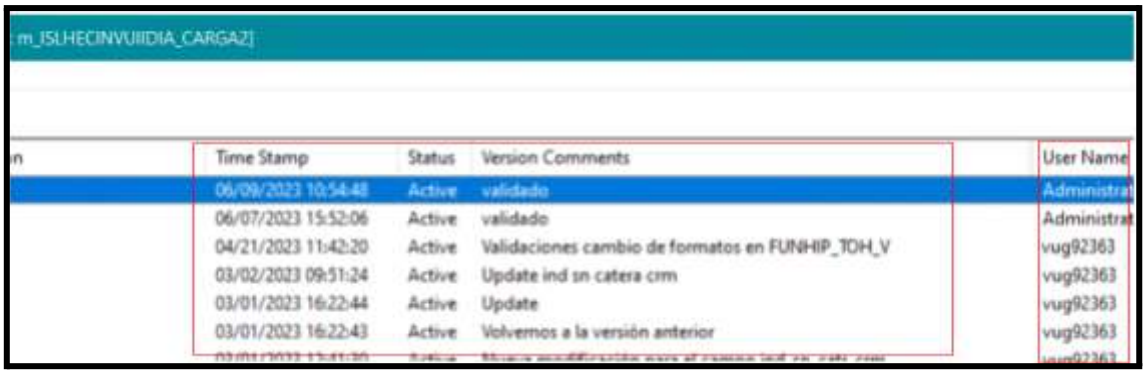

Figura 4.84 Metadatos cambios realizados Powercenter

Si no encuentra nada, buscará en la carpeta compartida de mantenimiento el nombre de la cadena, y obtendrá el plan de mantenimiento que es un histórico de los cambios que han recibido los desarrollos a lo largo del tiempo ya sea por modificación o nueva integración (Figura 4.85).

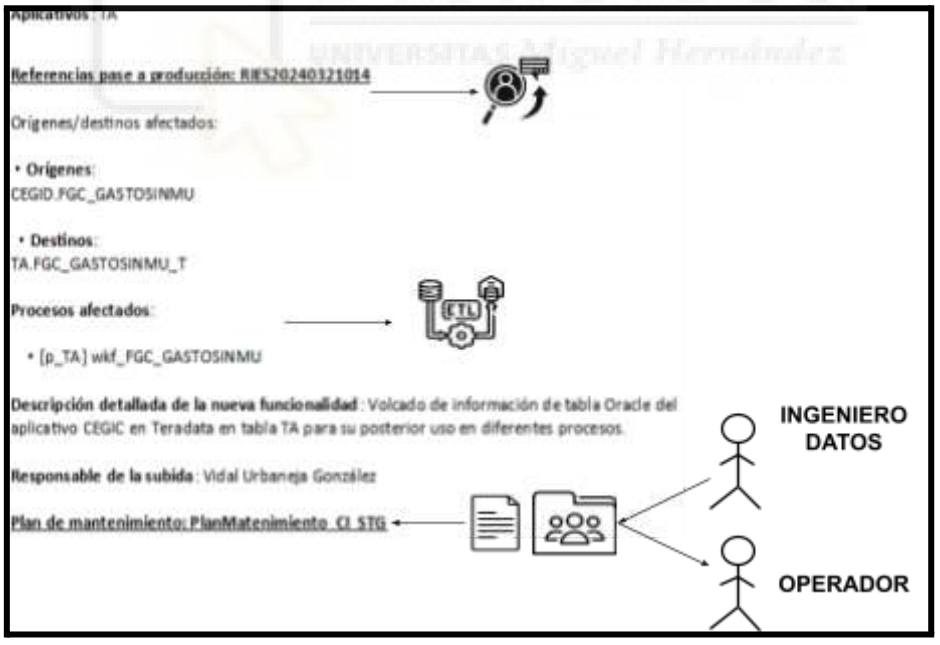

Figura 4.85 Extracto del plan de mantenimiento

En esta Figura 4.85 se representa uno de los cambios realizados en la cadena del *staging area* que consiste en la agregación de un nuevo proceso de STG. En este documento el técnico también encontrará las referencias usadas para la implantación de los diferentes procesos en producción. Su siguiente paso es buscar la última referencia que exista en el

documento del proceso que ha fallado, recordemos que a esta referencia se le vinculaba tanto el workflow como las pruebas de integración, por lo tanto, solo tiene que ir a la aplicación de subidas, ingresar la referencia y la aplicación le devolverá los documentos de integración y validación.

Los documentos de integración describen detalladamente el proceso ETL y lo más importante, contienen la consulta SQL traducción del desarrollo Powercenter, que ayudado del error que arroja el Monitor de PWC, facilita el depurado de errores. En la siguiente Figura 4.86, se representa los pasos seguidos por el operador desde que falla el proceso, su depurado, la modificación del mismo y cómo vuelve a hacer el ciclo de despliegue para subsanar el fallo. Una vez el proceso en producción con el fallo subsanado solo se deberá relanzar para una nueva ejecución.

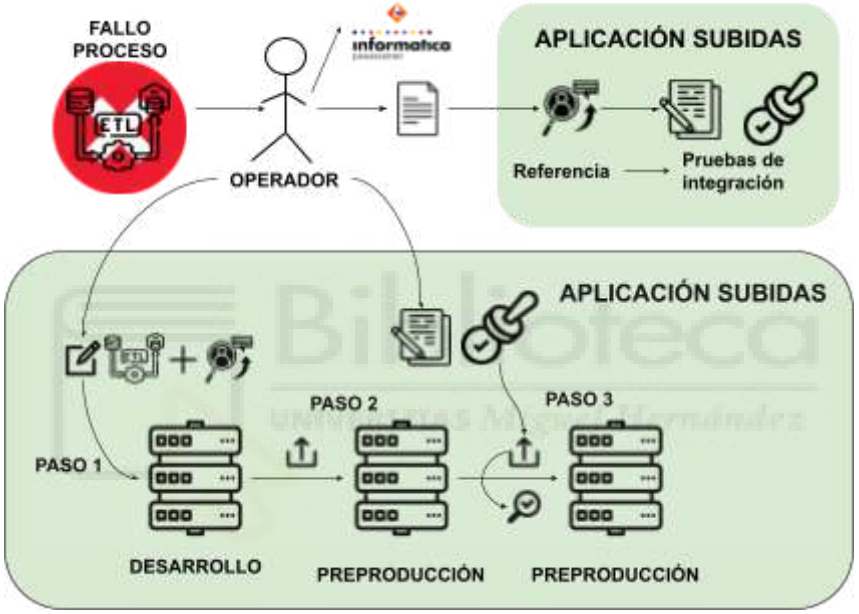

Figura 4.86 Procedimiento para gestionar una incidencia por fallo de ETL

Por su parte, los procesos no críticos son aquellos en los que los datos no son prioritarios en el día, estos son mantenidos, en cada caso, por el propietario del proceso que pudiera fallar (u otro compañero de departamento) lo antes posible en su jornada laboral.

Capítulo 5
Conclusiones y trabajo futuro

5.1.- CONCLUSIONES

La creación del sistema informacional de inmuebles a día de hoy ha cumplido todos los hitos para los que en un principio fue pensado y ha seguido ampliándose con nuevas funcionalidades a lo largo del tiempo, prueba de ello es el constante flujo de trabajo enviado por usuarios que viendo los beneficios del mismo siguen solicitando nuevas tareas y evolutivos. El modelo ha pasado por diferentes etapas, una primera en la en que se consolidaba el almacén, en vistas de crear un duplicado de las bases de datos de la inmobiliaria, centralizando e integrando mediante la herramienta ETL de Powercenter toda la información importante en un mismo repositorio que residiera en la propia entidad bajo sus reglas, y cuya finalidad era la contabilidad, como baremo de salud de la cual gozaba la parte de recobro de deuda mediante la venta de bienes principalmente. Esta etapa se construyó en poco tiempo, el Data Warehouse guardaba las entidades esenciales, pero afinarlo y prepararlo para la explotación por otras partes de la entidad, llevaría mucho más

tiempo y comprendería las sucesivas etapas, como la inclusión de nuevas fuentes de datos y la creación de diferentes Data Marts de los cuales beberían los cuadros de mando que ayudarían a dar soporte a los usuarios comerciales o proveer de información a otras áreas del banco como el departamento de riesgos.

En conclusión, el desarrollo de este sistema ha proporcionado al banco una solución integrada y eficiente para la gestión de inmuebles y su posterior explotación, facilitando la ingesta de datos comprensibles, y ofreciendo una base sólida para nuevas funcionalidades en un futuro.

5.2.- POSIBLES DESARROLLOS FUTUROS

Como hemos visto el modelo de inmuebles está en constante evolución y aunque las integraciones que se pueden realizar son finitas todavía quedan muchos posibles desarrollos por materializar, entre ellos nos encontramos la retroalimentación de datos a partir de la información ingresada por los usuarios finales en los cuadros de mandos de Salesforces a las propias tablas del almacén, hasta ahora estos cuadros de mando solo tenían la funcionalidad de visualizar la información, esta nueva funcionalidad permitirá a los usuarios finales enviar órdenes de trabajo como limpieza, cerrajería, etc., para el mantenimiento de los inmuebles, toda esta información serviría como nuevos orígenes y se grabaría en las tablas de datos comerciales pertenecientes al almacén.

También están pendientes realizar modificaciones profundas en los procesos de integración del repositorio debido a la división de los orígenes en datos comerciales y datos contables. Ya no se recibirá por la base de datos de oracle de la inmobiliaria los datos contables y comerciales y posteriormente por fichero los datos comerciales, sino que la base de datos de oracle se enfocaría únicamente en los datos contables y se recibiría por fichero solo los datos comerciales. Esta modificación va en paralelo, a lo más apremiante que se realiza en la actualidad, que es el envío de datos y nuevas conexiones provocadas por el cambio de inmobiliaria que daba soporte y gestionaba los activos del banco. En este cambio de proveedor de servicios, se realizarán cambios en los orígenes, la migración de datos y ejemplos para que la nueva inmobiliaria prepare sus bases de datos en una estructura parecida a la recibida hasta ahora en el almacén, y desarrollos secundarios como la información sobre el movimiento de llaves ya que la nueva inmobiliaria recibirá de la antigua todas las llaves de los activos y se realizará una trazabilidad sobre los mismos.

Sin duda, mientras la entidad perdure el sistema informacional de inmuebles seguirá activo, ya que siempre será necesario un mantenimiento y mejora que desemboque en nuevos desarrollos futuros.

Bibliografía

[1] Crisis del ladrillo
https://www.pibank.es/que-fue-burbuja-inmobiliaria-espana/
Fecha de consulta: 03-03-2024

[2] Ejemplos inmobiliarias que gestionan inmuebles de bancos
https://www.helpmycash.com/hipotecas/inmobiliarias-de-bancos/
Fecha de consulta: 03-03-2024

[3] Inserción de datos
https://www.campusmvp.es/recursos/post/Fundamentos-de-SQL-Insercion-de-datos-INSERT.aspx
Fecha de consulta: 03-03-2024

[4] Calidad del dato
https://www.astera.com/es/type/blog/data-standardization/
Fecha de consulta: 05-03-2024

[5] CIRBE
https://clientebancario.bde.es/pcb/es/menu-horizontal/productosservici/relacionados/cirbe
Fecha de consulta: 20-03-2024

[6] Staging Area
Libro Data Warehousing - Hefesto V3 Pág 41. Apartado Extracción
Fecha de consulta: 20-03-2024

[7] Data Warehouse
Libro Data Warehousing - Hefesto V3 Pág 49. Capítulo 3: Arquitectura Data Warehousing
Fecha de consulta: 01-04-2024

[8] ETL
Libro Data Warehousing - Hefesto V3 Pág 30. Apartado Integración procesos ETL
Fecha de consulta: 01-04-2024

[9] Back End y Front End
https://aws.amazon.com/es/compare/the-difference-between-frontend-and-backend/#:~:text=El%20back%2Dend%20son%20los,las%20aplicaciones%20para%20los%20usuarios.
Fecha de consulta: 05-04-2024

[10] Front End
Libro Data Warehousing - Hefesto V3 Pág 30. Apartado Integración procesos ETL
Fecha de consulta: 05-04-2024

[11] Sistemas transaccionales
https://sistema-de-informacion-de-contabilidad-y-finanzas.fandom.com/es/wiki/Sistemas_Transaccionales
Fecha de consulta: 05-04-2024

[12] Base de datos relacional
https://www.oracle.com/es/database/what-is-a-relational-database/
Fecha de consulta: 05-04-2024

[13] Sistemas CRM
https://www.oracle.com/es/cx/what-is-crm/
Fecha de consulta: 05-04-2024

[14] Informatica Powercenter
https://www.informatica.com/es/products/data-integration/powercenter.html
Fecha de consulta: 05-04-2024

[15] W. H. Inmon
https://www.astera.com/es/type/blog/data-warehouse-concepts/#:~:text=Bill%20Inmon%2C%20el%20padre%20del,%2C%20productos%2C%20proveedores%2C%20etc
Fecha de consulta: 17-04-2024

[16] Informatica - PowerCenter
https://www.informatica.com/es/products/data-integration/powercenter.html
Fecha de consulta: 17-04-2024

[17] Características Informatica - PowerCenter
https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/beneficios-de-utilizar-informatica-powercenter-como-herramienta-etl
Fecha de consulta: 17-04-2024

[18] Talend - Studio
https://www.talend.com/products/talend-open-studio/
Fecha de consulta: 17-04-2024

[19] Características Talend - Studio
https://www.modus.es/talend-open-studio-plataforma-de-integracion/
Fecha de consulta: 17-04-2024

[20] IBM - InfoSphere DataStage
https://www.ibm.com/es-es/products/datastage
Fecha de consulta: 17-04-2024

[21] IBM - InfoSphere Suite
https://www.ibm.com/docs/es/iis/11.5?topic=server-components-in-suite
Fecha de consulta: 17-04-2024

[22] Microsoft - SQL Server Integration Services
https://learn.microsoft.com/es-es/sql/integration-services/sql-server-integration-services?view=sql-server-ver16
Fecha de consulta: 17-04-2024

[23] Características Microsoft - SQL Server Integration Services
https://blog.bismart.com/que-diferencia-etl-y-ssis
Fecha de consulta: 24-04-2024

[24] Apache - Nifi
https://nifi.apache.org/
Fecha de consulta: 24-04-2024

[25] Características Apache - Nifi
https://aprenderbigdata.com/introduccion-apache-nifi/
Fecha de consulta: 24-04-2024

[26] Hitachi - Pentaho Data Integration
https://www.hitachivantara.com/pentaho/
Fecha de consulta: 24-04-2024

[27] Características Hitachi - Pentaho Data Integration
https://www.incentro.com/es-ES/blog/que-es-pentaho
Fecha de consulta: 24-04-2024

[28] Cloverdx
https://www.cloverdx.com/
Fecha de consulta: 24-04-2024

[29] Características Cloverdx
https://newsmatic.com.ar/big-data/revision-de-cloverdx
Fecha de consulta: 24-04-2024

[30] Oracle - Data Integrator (ODI)
https://www.oracle.com/es/middleware/technologies/data-integrator.html
Fecha de consulta: 24-04-2024

[31] Características Oracle - Data Integrator (ODI)
https://www.oracle.com/mx/a/ocom/docs/10-beneficios-de-la-plataforma.pdf
Fecha de consulta: 24-04-2024

[32] Data Warehousing
Libro Data Warehousing - Hefesto V3 Pág 26. Data Warehousing vs Data Warehouse
Fecha de consulta: 02-05-2024

[33] ODS
https://www.techtarget.com/searchoracle/definition/operational-data-store
Fecha de consulta: 02-05-2024

[34] OLAP
https://troyanx.com/Hefesto/olap.html
Fecha de consulta: 02-05-2024

[35] ROLAP
https://troyanx.com/Hefesto/rolap.html
Fecha de consulta: 02-05-2024

[36] MOLAP
https://troyanx.com/Hefesto/molap.html
Fecha de consulta: 02-05-2024

[37] HOLAP
https://troyanx.com/Hefesto/holap.html
Fecha de consulta: 02-05-2024

[38] Data Marts
https://troyanx.com/Hefesto/data-mart.html
Fecha de consulta: 02-05-2024

[39] Data Sources
https://troyanx.com/Hefesto/1-datasources.html
Fecha de consulta: 02-05-2024

[40] Load Manager
https://troyanx.com/Hefesto/2-load-manager.html
Fecha de consulta: 02-05-2024

[41] Data Warehouse Manager
https://troyanx.com/Hefesto/3-data-warehouse-manager.html
Fecha de consulta: 02-05-2024

[42] Modelo de estrella
https://troyanx.com/Hefesto/estrella.html
Fecha de consulta: 02-05-2024

[43] Tabla Hechos
https://troyanx.com/Hefesto/tablas-de-hechos.html
Fecha de consulta: 02-05-2024

[44] Tabla Dimensión
https://troyanx.com/Hefesto/32-tablas-de-dimensiones.html
Fecha de consulta: 02-05-2024

[45] Dashboard
https://troyanx.com/Hefesto/dashboards.html
Fecha de consulta: 02-05-2024

[46] KPI
https://troyanx.com/Hefesto/12-indicadores-y-perspectivas.html
Fecha de consulta: 02-05-2024

[47] Business Intelligence
https://troyanx.com/Hefesto/definiendo-al-bi.html
Fecha de consulta: 02-05-2024

[48] Gestor bases de datos (SGBD)
https://troyanx.com/Hefesto/sgbd.html
Fecha de consulta: 02-05-2024

[49] Copo de nieve
https://troyanx.com/Hefesto/copo-de-nieve.html
Fecha de consulta: 02-05-2024

[50] Gestor base de datos - Teradata
https://www.teradata.com/
Fecha de consulta: 02-05-2024

[51] Gestor base de datos - Oracle
https://www.oracle.com/es/database/
Fecha de consulta: 02-05-2024

[52] Principales tipos Joins en SQL
https://diego.com.es/principales-tipos-de-joins-en-sql
Fecha de consulta: 02-05-2024

[53] Salesforce
https://www.salesforce.com/
Fecha de consulta: 02-05-2024

[54] Qlikview
https://help.qlik.com/es-ES/qlikview/May2023/Content/QV_HelpSites/what-is.htm
Fecha de consulta: 15-05-2024

[55] Erwin Data Modeler
https://www.danysoft.com/erwin/
Fecha de consulta: 15-05-2024

[56] Control -M
https://www.bmcsoftware.es/it-solutions/control-m.html
Fecha de consulta: 15-05-2024

[57] Pipeline
https://www.purestorage.com/es/knowledge/what-is-a-data-pipeline.html
Fecha de consulta: 03-06-2024

[58] Ciclo de vida del dato
https://segment.com/blog/data-life-cycle/
Fecha de consulta: 03-06-2024

[59] Método de carga Powercenter - Fload
https://docs.informatica.com/es_es/data-security-group/test-data-management/10-4-0/guia-del-administrador/conexiones/conexiones-de-teradata-fastload.html
Fecha de consulta: 03-06-2024

[60] Método de carga Powercenter - PDO
https://docs.informatica.com/data-integration/powercenter/10-5/advanced-workflow-guide/pushdown-optimization/pushdown-optimization-overview.html
Fecha de consulta: 03-06-2024

[61] OBIEE
https://www.mindstreamanalytics.com/esp/articulos/que-es-obiee.html
Fecha de consulta: 03-06-2024

[62] Método carga Dimensión SCD
https://troyanx.com/Hefesto/slowly-changing-dimensions.html
Fecha de consulta: 03-06-2024

[63] Método de carga Powercenter - Mload
https://docs.informatica.com/es_es/data-security-group/test-data-management/10-4-0/guia-del-administrador/conexiones/conexiones-de-teradata-multiload.html
Fecha de consulta: 03-06-2024

[64] UAT
https://www.wearetesters.com/investigacion-ux/pruebas-aceptacion-usuario-uat-que-son/
Fecha de consulta: 03-06-2024

[65] Método de carga Powercenter - Relacional
https://docs.informatica.com/data-integration/powercenter/10-5/advanced-workflow-guide/pushdown-optimization/pushdown-optimization-overview.html
Fecha de consulta: 03-06-2024

Anexo I
Casos de Uso

En este anexo se presentan las tablas que describen en detalle cada caso de uso del proyecto.

Tabla AI.1: Caso de uso 01.

C.U. 01	Consultar Informes
Actores	Usuario de Negocio, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Consulta de informes financieros procedentes del Data Warehouse de inmuebles por parte del usuario de contabilidad.
Dependencias	Carga primera del DW de inmuebles
Frecuencia	Diaria
Importancia	Alta
Urgencia	Alta
Comentarios	Analiza y revisa los indicadores clave de rendimiento (KPI) como márgenes de beneficio, liquidez y endeudamiento.

Tabla AI.2: Caso de uso 02.

C.U. 02	Consultar SQL OLAP
Actores	Usuario de Negocio, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Consulta mediante lenguaje SQL el OLAP del DW de inmuebles
Dependencias	Debe estar cargado el Data Warehouse
Frecuencia	Diaria
Importancia	Alta
Urgencia	Alta
Comentarios	Analiza y revisa los indicadores clave de rendimiento (KPI) en la propia BBDD del DW

Tabla AI.3: Caso de uso 03.

C.U. 03	Análisis de Rendimiento
Actores	Usuario de Negocio
Descripción	Analiza las variaciones contables y reporta desviaciones significativas en los valores contables de los activos vendidos.
Dependencias	Consultar Informes
Frecuencia	Diaria
Importancia	Alta
Urgencia	Alta
Comentarios	El usuario hace un análisis y envía un correo comunicando los resultados.

Tabla AI.4: Caso de uso 04.

C.U. 04	Consultar Reporting
Actores	Usuario de Riesgo, Usuario de Negocio, Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Consulta en la aplicación BI Qlik reportes de los activos
Dependencias	Debe estar cargado el Data Warehouse de inmuebles y el Data Mart de Garantías
Frecuencia	Mensual
Importancia	Alta
Urgencia	Alta
Comentarios	Analiza y revisa los reportes generados de los activos

Tabla AI.5: Caso de uso 05.

C.U. 05	Consultar Dashboard
Actores	Usuario Comercial, Usuario Abogado, Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Visualización de los paneles de mando que muestran información y permiten realizar acciones sobre ellos.
Dependencias	Carga del DW de inmuebles, Carga del Data Mart de Salesforce
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Si el Data Mart no se ha cargado, podrán seguir accediendo a los paneles pero solo verán la información correspondiente a los últimos datos disponibles.

Tabla AI.6: Caso de uso 06.

C.U. 06	Órdenes de Trabajo
Actores	Usuario Comercial
Descripción	Genera y envía órdenes de trabajo para servicios específicos como limpieza, reparaciones, cerrajería,etc.
Dependencias	Carga del DW de inmuebles, Carga del Data Mart de Salesforce
Precondición	Visualizar el Dashboard
Postcondición	Se genera una orden de trabajo.
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Esta parte de “saneamiento” a día de hoy está en desarrollo pero la intención es que retroalimente las tablas de datos comerciales de trabajos, tarifas y trámites del DW que se verán en capítulos posteriores.

Tabla AI.7: Caso de uso 07.

C.U. 07	Gestión Inmuebles
Actores	Usuario Comercial
Descripción	Visualiza el estado de los inmuebles bajo su gestión
Dependencias	Carga del DW de inmuebles,Carga del Data Mart de Salesforce
Precondición	Visualizar el Dashboard
Postcondición	Se muestran los activos gestionados
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Si el Data Mart no se ha cargado podrá seguir accediendo a los paneles pero solo verá la información correspondiente a los últimos datos disponibles.

Tabla AI.8: Caso de uso 08.

C.U. 08	Gestor documental
Actores	Usuario Comercial, Usuario Abogado
Descripción	Documentos que están disponibles para la descarga por parte de los usuarios finales.
Dependencias	Carga del DW de inmuebles,Carga del Data Mart de Salesforce
Precondición	Visualizar el Dashboard
Postcondición	Se descarga el documento
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	En la parte de inmuebles no se guardan estos documentos, sino que en sus tablas se carga la ruta al almacenamiento donde están guardados. Este caso de uso es compartido por el usuario comercial y el usuario abogado ya que se usa el mismo procedimiento para ambos.

Tabla AI.9: Caso de uso 09.

C.U. 09	Gestión Demandas
Actores	Usuario Abogado
Descripción	Visualiza el estado y progreso de cada demanda, además de los intervinientes que están relacionados a los inmuebles.
Dependencias	Carga del DW de inmuebles,Carga del Data Mart de Salesforce
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Si el Data Mart no se ha cargado podrá seguir accediendo a los paneles pero solo verá la información correspondiente a los últimos datos disponibles.

Tabla AI.10: Caso de uso 10.

C.U. 10	Diseño
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Diseño de elementos que participan en el DWH.
Dependencias	Depende
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Estos elementos se describen en otros casos de uso.

Tabla AI.11: Caso de uso 11.

C.U. 11	Diseño de Dashboards y Reportes
Actores	Desarrollador BI
Descripción	Diseño de paneles de mando o reportes de Inmuebles para los usuarios finales.
Dependencias	Carga del DW de inmuebles,Carga del Data Mart de Salesforce, Carga del Data Mart de Garantías
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Los diseños vienen dados después de la toma de requisitos a los usuarios y en conjunto con el Ingeniero de Datos y Jefe de Proyecto de Inmuebles.

Tabla AI.12: Caso de uso 12.

C.U. 12	Desarrollos
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Desarrollo de elementos que participan en el DWH.
Dependencias	Ninguna
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Estos elementos se describen en otros casos de uso.

Tabla AI.13: Caso de uso 13.

C.U. 13	Desarrollo Dashboards y Reportes
Actores	Desarrollador BI
Descripción	Se desarrollan los paneles y Reportes con las herramientas BI de Salesforce y QLik.
Dependencias	Carga del DW de inmuebles,Carga del Data Mart de Salesforce, Carga del Data Mart de Garantías
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Se informarán sus paneles desde los Data Marts Inmuebles, se requiere una colaboración con Ingeniero de Datos o el Jefe de Proyectos de Inmuebles para su correcto desarrollo.

Tabla AI.14: Caso de uso 14.

C.U. 14	Mantenimiento
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles, Administrador de base de datos, Operador de Soporte, Administrador de Seguridad
Descripción	Mantenimiento de los diferentes elementos del DWH.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	El mantenimiento es realizado por todos los actores que no son usuarios finales, corresponde a uno de los casos más comunes del día a día entre los roles.

Tabla AI.15: Caso de uso 15.

C.U. 15	Incidencias
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles, Administrador de base de datos, Operador de Soporte, Administrador de Seguridad
Descripción	Resolución de incidencias ocasionadas por la interrupción o parada de elementos del DWH.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	La resolución de incidencias es común en todos los actores dependiendo su rol (a excepción de los usuarios finales), desde bloqueo en tablas para los administradores de base de datos o ser el caso de uso más repetitivo para los operadores.

Tabla AI.16: Caso de uso 16.

C.U. 16	Documentación
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles, Administrador de base de datos, Operador de Soporte, Administrador de Seguridad
Descripción	Documentación de los diferentes elementos del DWH.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	La documentación describe y deja constancia de cualquier elemento, consiguiendo que sea comprensible por otros usuarios, es común a todos los roles a excepción de los usuarios finales.

Tabla AI.17: Caso de uso 17.

C.U. 17	Implantación
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles, Administrador de base de datos, Operador de Soporte
Descripción	Representa el despliegue de un elemento del DWH en producción
Dependencias	Desarrollo de elementos del DWH en desarrollo.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Según el rol la implantación será de un elemento en concreto.

Tabla AI.18: Caso de uso 18.

C.U. 18	Pruebas y Validaciones
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles, Operador de Soporte
Descripción	Caso de uso que verifica el buen funcionamiento del DWH.
Dependencias	Desarrollo de elementos del DWH que validar.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Según el rol las pruebas y validaciones serán diferentes.

Tabla AI.19: Caso de uso 19.

C.U. 19	Automatización
Actores	Desarrollador BI, Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Acción que permite que los desarrollos del Ingeniero de Datos y Desarrollador BI sean ejecutados en determinada franja horaria automáticamente.
Dependencias	Tienen que existir elementos del DWH en producción.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	La automatización puede crear incidencias.

Tabla AI.20: Caso de uso 20.

C.U. 20	Implantación Dashboards y Reportes
Actores	Desarrollador BI
Descripción	Despliegue en el entorno de producción desarrollos BI como Dashboards y Reportes
Dependencias	Deben existir estos objetos BI en el entorno de desarrollo.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Estos elementos serán informados a partir de los Data Mart de Salesforce y Garantías.

Tabla AI.21: Caso de uso 21.

C.U. 21	UAT Dashboards y Reportes
Actores	Desarrollador BI
Descripción	Validación de los desarrollos BI.
Dependencias	Desarrollo de Dashboards y Reportes
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Ninguno.

Tabla AI.22: Caso de uso 22.

C.U. 22	Automatización Dashboards y Reportes
Actores	Desarrollador BI
Descripción	Automatización de los elementos como Dashboards y Reportes subidos a producción.
Dependencias	Despliegue de elementos bi en producción.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	La automatización vendrá después de la planificación de los Data Marts de Salesforce y Garantías de Inmuebles.

Tabla AI.23: Caso de uso 23.

C.U. 23	Diseño ETLs
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Diseño y métodos de ETLs para la integración de los datos
Dependencias	Requisitos de Usuario.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Se diseña la mejor estrategia para llevar a cabo las integraciones y conseguir la información requerida por los usuarios

Tabla AI.24: Caso de uso 24.

C.U. 24	Diseño Tablas
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Diseño de tablas y campos que llevarán la información
Dependencias	De los usuarios de administración de bases de datos.
Postcondición	Creación de la tabla y campos por parte del administrador de BBDD
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Aunque el rol de Ingeniero de datos en la entidad no puede modificar la base de datos, si es el encargo de diseñarlas y pedir su creación.

Tabla AI.25: Caso de uso 25.

C.U. 25	Desarrollo ETLs
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Desarrollo de ETLs con la herramienta Powercenter
Dependencias	Diseño de ETL
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Este caso de uso abarca tanto la creación como la modificación de las ETLs de inmuebles

Tabla AI.26: Caso de uso 26.

C.U. 26	Optimización ETL
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Optimización de la integración de datos
Dependencias	Ninguna
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Aborda la modificación de integraciones para conseguir una mejor eficiencia en la ejecución de los procesos de integración.

Tabla AI.27: Caso de uso 27.

C.U. 27	Calidad del Dato
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Conjunto de procesos que aseguran la limpieza y corrección de errores de los datos
Dependencias	Ninguna
Frecuencia	Baja
Importancia	Media
Urgencia	Media
Comentarios	La diferencia con el caso de uso de pruebas es que este va dirigido a los datos y no a los procesos que los integran.

Tabla AI.28: Caso de uso 28.

C.U. 28	Desarrollo de informes
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Desarrollo de informes a partir del DW.
Dependencias	Carga del DW de inmuebles.
Frecuencia	Baja
Importancia	Media
Urgencia	Media
Comentarios	Este caso de uso abarca tanto la creación como modificación de informes que se informan del DW.

Tabla AI.29: Caso de uso 29.

C.U. 29	Implantación ETL
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Despliegue del proceso ETL en el entorno de producción.
Dependencias	Validación de proceso ETL en desarrollo
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Se despliegan los procesos de integración en producción.

Tabla AI.30: Caso de uso 30.

C.U. 30	Pruebas ETL
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Validación y pruebas de integración ETL.
Dependencias	Desarrollo del proceso de integración en desarrollo.
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Se asegura que los elementos de integración funcionen correctamente respecto a lo deseado en las especificaciones de los clientes.

Tabla AI.31: Caso de uso 31.

C.U. 31	Automatización Proceso ETL
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Se automatizan los procesos ETL de producción.
Dependencias	Despliegue del proceso ETL en producción.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Se automatizan las integraciones para que carguen el Data Warehouse en horas no laborales para tener los datos disponibles al día siguiente.

Tabla AI.32: Caso de uso 32.

C.U. 32	Migración
Actores	Ingeniero de Datos, Jefe de Proyectos Inmuebles
Descripción	Transferencia de datos entre sistemas.
Dependencias	Ninguna
Frecuencia	Baja
Importancia	Media
Urgencia	Media
Comentarios	Este caso de uso conlleva el traslado de datos de una o más fuentes a un nuevo sistema de Data Warehouse.

Tabla AI.33: Caso de uso 33.

C.U. 33	Comunicación y Reportes
Actores	Jefe de Proyectos Inmuebles
Descripción	Presentación de los avances usando reportes.
Dependencias	Ninguna
Frecuencia	Baja
Importancia	Media
Urgencia	Media
Comentarios	Se usan reportes obtenidos a partir de OLAP del Data Warehouse de inmuebles para apoyar la comunicación y presentar el progreso y resultados del proyecto de inmuebles a los usuarios finales.

Tabla AI.34: Caso de uso 34.

C.U. 34	Seguimiento
Actores	Jefe de Proyectos Inmuebles
Descripción	Seguimiento de tareas e hitos del proyecto.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Realiza seguimientos diarios para comprobar el progreso y planificar las tareas.

Tabla AI.35: Caso de uso 35.

C.U. 35	Análisis de Requisitos
Actores	Jefe de Proyectos Inmuebles
Descripción	Recopila y prioriza los requisitos de los usuarios.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Es el caso de uso más habitual en el jefe de proyecto de inmuebles, compromete acciones como reuniones con los usuarios, sintetizar los requisitos y comunicación de los mismos al ingeniero de datos.

Tabla AI.36: Caso de uso 36.

C.U. 36	Accesos
Actores	Administrador de Seguridad
Descripción	Otorga permisos de acceso al resto de usuarios.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Alta
Urgencia	Media
Comentarios	El caso de uso abarca diferentes permisos, a los propietarios de las tablas, a Powercenter, a los entornos, en especial a las herramientas BI y los Paneles y Reportes que las componen por contener datos delicados.

Tabla AI.37: Caso de uso 37.

C.U. 37	Roles y Perfiles
Actores	Asignación de roles y perfiles.
Descripción	Asignación de un conjunto de permisos a un usuario.
Dependencias	Ninguna.
Frecuencia	Diaria
Importancia	Alta
Urgencia	Media
Comentarios	En el caso de uso de acceso se han puesto como ejemplo varias situaciones en las que se necesitan permisos, la asignación de roles y perfiles, empaqueta una serie de accesos que se dan a los usuarios para no ir de uno en uno.

Tabla AI.38: Caso de uso 38.

C.U. 38	Gestión entidades
Actores	Administrador de BBDD
Descripción	Administración de las bbdd.
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Administra las bbdd de inmuebles y elementos primarios como las tablas.

Tabla AI.39: Caso de uso 39.

C.U. 39	Creación Tabla
Actores	Administrador de BBDD
Descripción	Creación de tablas en modelo de inmuebles
Dependencias	Ninguna
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	La creación de tablas abarca desde el *staging area* hasta los Data Mart pasando por el Data Warehouse, son todas las tablas relacionadas con el modelo.

Tabla AI.40: Caso de uso 40.

C.U. 40	Modificación Tabla
Actores	Administrador de BBDD
Descripción	Modificación de la estructura de las tablas.
Dependencias	Ninguna.
Frecuencia	Diaria
Importancia	Media
Urgencia	Media
Comentarios	Este caso de uso se refiere a la acción que podemos hacer sobre una tabla, modificar, crear o eliminar campos, cambiar claves principales o tipo de datos, etc.

Tabla AI.41: Caso de uso 41.

C.U. 41	Buck up
Actores	Administrador de BBDD
Descripción	Realización de copias de seguridad del modelo de inmuebles
Dependencias	Ninguna.
Frecuencia	Media
Importancia	Media
Urgencia	Media
Comentarios	Proceso de realizar copias de seguridad de los datos del almacén para que en caso de pérdida por fallos humanos o del sistema, se pueda recuperar la información.

Tabla AI.42: Caso de uso 42.

C.U. 42	Monitoreo ETLs
Actores	Operador Soporte
Descripción	Monitoreo de procesos de integración en el entorno de producción.
Dependencias	Ninguna.
Frecuencia	Diaria
Importancia	Alta
Urgencia	Alta
Comentarios	Caso de uso habitual por el Operador de Soporte que gestiona los procesos críticas en producción por ejemplo pueda abortar procesos en el entorno de producción