Trabajo Fin de Grado

Grado en Ingeniería de las Tecnologías de Telecomunicación

Fundamentos y caso práctico de diseño de una solución B.I para explotar datos abiertos

Autor: Mohamed Lahlou

Tutor: Teresa Ariza Gómez

Dep. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017

Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

Fundamentos y caso práctico de diseño de una solución B.I para explotar datos abiertos

Autor:
Mohamed Lahlou

Tutor:
Teresa Ariza Gómez
Profesor titular

Dep. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2017

Trabajo Fin de Grado: Fundamentos y caso práctico de diseño de una solución B.I para explotar datos abiertos

Autor: Mohamed Lahlou

Tutor: Teresa Ariza Gómez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2017

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

A mi padre que no ha dejado ni una milésima de segundo de creer en mí y en darme todo el apoyo que necesitaba, a mi madre que ha sufrido mucho mi ausencia de casa durante toda mi estancia de estudios y que no ha parado ni un momento de rezar por mí, a mis hermanos, sobrinos y amigos que siempre me han apoyado, a mis compañeros de trabajo que con su ayuda he podido adquirir los conocimientos suficientes como para llevar este proyecto a cabo y finalmente a España y especialmente a la ciudad de Sevilla por darme la oportunidad de realizar mis estudios en su Universidad.

Mohamed Lahlou

Sevilla, 2017

Resumen

Hoy en día vivimos en un mundo que se está haciendo cada vez más tecnológico y en el que cada día se genera una cantidad enorme de información que de alguna forma habrá que tratarla y sacar de ella algún activo que nos ayude a mejorar.

Este proyecto tiene como objetivo introducir los conceptos fundamentales de Business Intelligence que se usa para transformar los datos en información y la información en conocimiento y el conocimiento en acción. Aprovechando los datos abiertos facilitados por el gobierno de España.

Abstract

Nowadays we live in an increasingly technological world in which every day a huge amount of information is being generated. This sum of information needs to then be analyzed so it can help make improvements.

This project aims to introduce the fundamental concepts of Business Intelligence that is used to transform data into information and information into knowledge and therefore knowledge into action taking advantage of the open data provided by the Government of Spain.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

1 INTRODUCCION

"Information is the oil of the 21st century."

– Peter Sondergaard, Gartner Research

.

1.1 Motivación

Hoy en día vivimos en el mundo donde la información es poder y más en el sector empresarial en el que la información es uno de sus activos fundamentales. Esto nos obliga a tener mecanismos y procesos para tratar los datos y sacar de ellos el mayor provecho para aumentar la eficiencia y conseguir ventaja a los competidores y ofrecer servicios personalizados a los clientes.

Con este objetivo los gobiernos ponen al servicio de las empresas y de forma pública y gratuita sin restricciones de derechos de autor conjuntos de datos de todos los ámbitos (transporte, salud, economía, educación, deporte…) para ser explotados y crear a partir de ellos nuevas líneas de negocio. España posee el liderazgo en gestión de datos abiertos en servicios públicos digitales (open data) según CNMC (Comisión Nacional de los Mercados y la Competencia).

Business Intelligence (B.I) o inteligencia de negocios en español hace referencia al conjunto de herramientas y servicios que permiten a los usuarios acceder y analizar de manera rápida y sencilla el conjunto de información para una posterior toma de decisiones de negocio a nivel operativo, táctico y estratégico, transformando los datos en información y la información en conocimiento.

Desde un punto de vista tecnológico podemos definir Business Intelligence como el conjunto de metodologías, aplicaciones y tecnologías que permiten reunir, depurar y transformar datos de los sistemas transaccionales e información desestructurada (interna y externa a la compañía) en información estructurada, para su explotación directa (reporting, análisis, alertas...) o para su análisis y conversión en conocimiento, dando así soporte a la toma de decisiones sobre el negocio.

La inteligencia de negocio actúa como un factor estratégico para una empresa u organización, generando una ventaja competitiva, que no es otra que proporcionar información privilegiada para responder a los problemas de negocio como facilitar la entrada a nuevos mercados y ayudar a realizar promociones en productos o hacer posible el análisis de perfiles de clientes, etc. De allí la importancia de este proyecto que sirve como introducción al mundo de la inteligencia de negocio.

1.2 Objetivos

Este proyecto tiene como objetivo introducir los conceptos necesarios para diseñar e implementar una solución B.I para una empresa de seguros de coches que quiere estudiar el comportamiento de las incidencias de tráfico para dar respuesta a la pregunta de cómo distribuir su flota de guras de forma óptima en la comunidad de Euskadi. Para ello vamos a explotar los datos abiertos que ofrece el gobierno de Euskadi sobre las incidencias de tráfico en dicha comunidad para crear una base de datos Data Warehouse / Data Mart que después será objetivo de nuestro análisis, como se puede ver en la figura 1.

Para conseguir este objetivo se ha dividido el mismo en los siguientes sub-objetivos:

- Introducir el Concepto de datos abiertos (Open Data) que hoy en día está en auge y que se usa para mejorar la vida ciudadana y en desarrollo de ciudades inteligentes.
- Introducir los fundamentos de inteligencia de negocios (Business Intelligence) que permitan afrontar el diseño de una solución B.I.
- Diseñar almacén de datos Data Warehouse / Data Mart de incidencias de tráfico en PostgreSQL, aprovechando la base de datos Open Data del gobierno de Euskadi.
- Desarrollo de procesos automatizados para extraer, transformar y cargar los datos (ETL) mediante las herramientas Talend y Rundeck.
- Diseño de cuadros de mando para facilitar la explotación de los datos una vez almacenados en Data Warehouse / Data Mart usando la herramienta (OLAP) Tableau.
- Finalmente responder a la pregunta de negocio sobre cómo se deben distribuir las flotas de forma óptima y rentable en la comunidad de Euskadi basándose en el análisis hecho sobre el Data Warehouse / Data Mart que hemos creado en PostgreSQL.

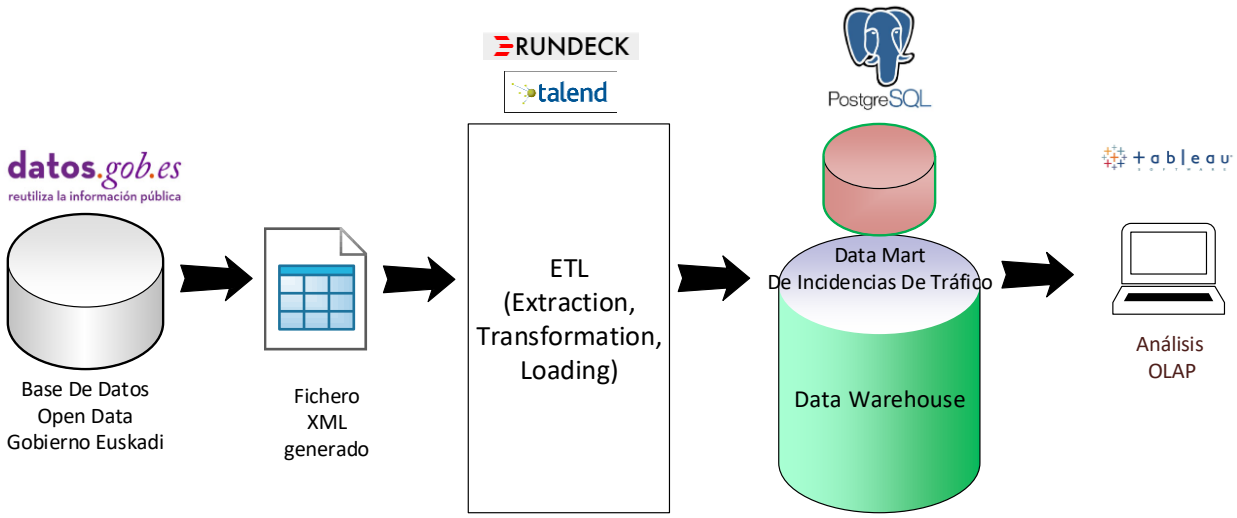


Figura 1. Objetivos del proyecto.

1.3 Estructura de la Memoria

- En el capítulo 1 además de la motivación y el objetivo de este proyecto se hará una presentación de la empresa en la cual se han adquirido los conceptos y la experiencia necesarios y en la que vamos a implementar el proyecto.
- En el capítulo 2 se abordarán los conceptos teóricos básicos de Open data, Business Intelligence, Datawarehouse y OLAP.
- En el capítulo 3 se van a ver las diferentes tecnologías a usar en este proyecto haciendo hincapié en sus principales características.
- En el capítulo 4 se va a analizar y describir el contexto general del proyecto que queremos llevar a cabo
- En el capítulo 5 se realizará el diseño y la implementación del caso práctico además se llevará a cabo el diseño del cuadro de mando dando respuestas a las preguntas planteadas por el negocio.
- En el capítulo 6 se verán las conclusiones y líneas futuras.

1.4 Presentación de la empresa EUI Global Service

1.4.1 Admiral group en la actualidad

El grupo Admiral es una de las compañías más solventes y rentables del Reino Unido. Cotiza en la Bolsa de Londres desde el año 2004 y Cuenta con más de 3 millones de clientes entre sus sedes nacionales de Cardiff y Swansea y las sedes internacionales.

En el año 2006 Admiral inicia su presencia más allá de las fronteras del Reino Unido, eligiendo España como sede para su primera marca internacional, **Balumba**. Tras el mercado español, Admiral ha desembarcado en Italia con **Conte.it**, en Virginia con **Elephant Auto Insurance** y en Francia con **Lelynx.fr** y **L'olivier.**

Actualmente en España Admiral tiene registradas 4 marcas de seguros que son **Balumba**, **Qualitas Auto**, **Wiyou auto** además del comparador de precios **Rastreador**.

EUI IT Global Services es la filial tecnológica del grupo Admiral. La división tecnológica EUI GS tiene en Sevilla más de 100 empleados que dan soporte a la actividad aseguradora de 3 países: Francia, Italia y España;

Admiral España ha conseguido a lo largo de su historia una serie de reconocimientos y premios gracias a la profesionalidad que le caracteriza. ha sido reconocida varias veces como una de las 50 mejores empresas para trabajar a nivel europeo "Great Place to Work" quedando en segundo puesto a nivel nacional en el año 2017.

1.4.2 Cifras de negocio

Admiral Group ofrece la posibilidad de consultar su facturación, sus números generales y sus beneficios anuales

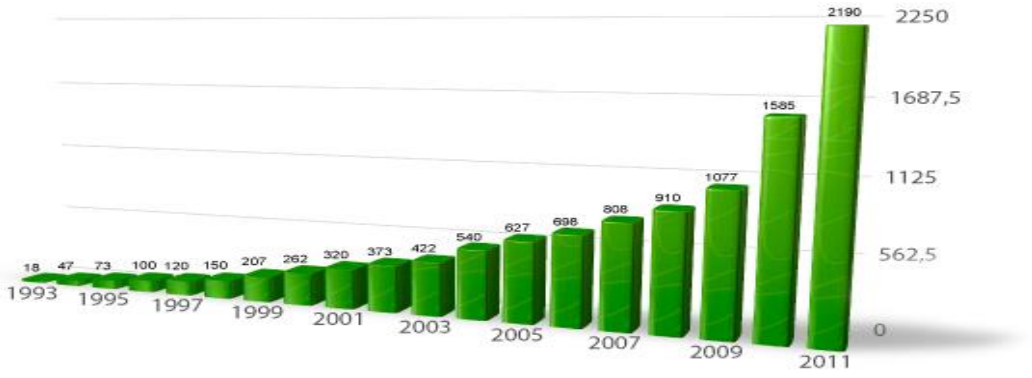


Figura 2. Evolución de la facturación (1993 – 2011) en (Millones de Libras).

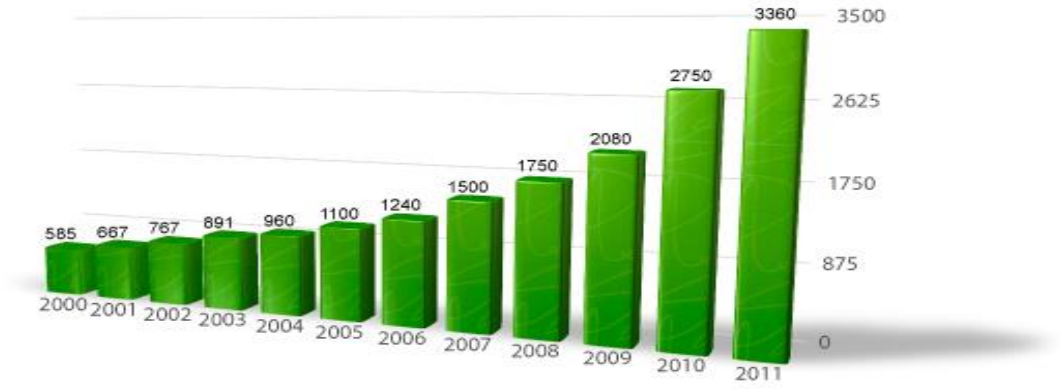


Figura 3. Evolución de número de clientes (Miles) entre año (2000- 2011).

1.4.3 Marcas nacionales e internacionales registradas del grupo Admiral

- Admiral, Home and Car Insurance (http://www.admiral.com).
- Balumba, Car Insurance (http://www.balumba.es/).
- Bell, Car Insurance (http://www.bell.co.uk/).
- Compare.com, Price Comparison (http://www.compare.com).
- Confused.com, Price Comparison (http://www.confused.com/).
- ConTe.it, Car Insurance (http://www.conte.it/).
- Diamond, Car Insurance (http://www.diamond.co.uk/).
- Elephant, Car Insurance (http://www.elephant.co.uk/).
- Elephant Auto, Car Insurance (http://www.elephant.com/).
- Gladiator, Van Insurance (http://www.gladiator.co.uk).
- L'olivier – assurance auto, Car Insurance (http://www.lolivier.fr/).
- LeLynx.fr, Price Comparison (http://www.lelynx.fr/).
- Qualitas Auto, Car Insurance (http://www.qualitasauto.com).
- Rastreator, Price Comparison (http://www.rastreator.com/).
- Wiyou auto (https://www.wiyouseguros.com/).

2 FUNDAMENTOS

"In god we all trust, all others must bring data"

--W. Edwards Deming

Las empresas siempre han estado en busca de nuevas estrategias que les permita coger las riendas de su sector , posicionarse y tener ventaja sobre la competencia. Para ello se han visto obligadas a aprovechar el gran flujo de datos del que disponen de forma organizada o no para sacar de ello la información que les permita adaptar nuevas estrategias, validar sus inversiones y conquistar nuevos mercados.

En este capítulo se realizará una introducción a los conceptos básicos de Open data, Business Intelligence, Data Warehouse y otros más que permitirán tener una idea sobre el mundo de inteligencia de negocio, metodologías y tecnologías usadas por las organizaciones para implementar soluciones B.I. Como permitirá también más adelante el entendimiento y la realización del caso práctico.

2.1 Open data

2.1.1 ¿Qué es?

Actualmente muchas de las aplicaciones que forman parte de nuestro día a día y que han cambiado nuestros hábitos cotidianos tienen como fuente de datos los datos abiertos facilitados por organismos públicos. Por lo cual uno de los objetivos de este proyecto es acerca de cómo liberar y aprovechar el potencial de la información oficial para dar vida a nuevos servicios y tener un impacto positivo en la sociedad.

Los datos abiertos o precisamente datos abiertos de gobierno es la información facilitada por los organismos públicos de cada país y que cualquiera es libre de acceder, reutilizar y redistribuir para cualquier propósito de forma gratuita y transparente. Datos como estadísticas, cartas geográficas, horarios, datos económicos y financieros, de transporte etc. Un cuadro jurídico legal y estricto define que tipo de datos públicos pueden ser abiertos y cuáles no, los datos sensibles de seguridad nacional y de carácter personal suelen estar excluidos.

2.1.2 ¿Para qué se usa?

Desde el año 2007 muchos gobiernos han puesto en marcha iniciativas gubernamentales y plataformas hacia la apertura de la información pública para crear y aportar valor añadido en áreas como:

- Transparencia y control democrático.
- Innovación.
- Administración.
- Sociedad.
- Salud.
- Cultura y educación.
- Etc.

Existen diversos ejemplos que demuestran como los datos abiertos están creando valor añadido en la sociedad tanto económico como tecnológico y social. Y con una pequeña búsqueda en la App store o el Google play podemos encontrar aplicaciones que ofrecen horarios de autobuses, localización de paradas de metros, hospitales y gasolineras.

- Administración

En el sector administrativo (gubernamental) se ha conseguido una mejor transparencia y control democrático conforme se ha aumentado la participación ciudadana en el control y la vigilancia de sus gobiernos. Ejemplos vivos podemos encontrarlos en Canadá donde los datos abiertos han permitido ahorrar más de 3.2 mil millones de dólares en el fraude fiscal. También en Finlandia encontramos "**Tax free"** o el "**Where does my money go**" en Britania que permite al ciudadano ver donde se gasta el dinero de sus impuestos.

- Sociedad

Los datos abiertos también nos ayudan a tomar decisiones diarias como a qué hora tenemos que salir de casa para no perder el bus. Aplicaciones que nos avisan de que la calidad del aire en el sitio donde nos encontramos esta baja "**vervuilingsalarm.nl",** o de los servicios públicos más cercanos de la posición en la que nos encontramos "**findtoilet.dk"**.

- Economía

Varios estudios afirman que el valor económico de los nuevos servicios, productos y puestos de trabajos creados a partir de la open data es de varios miles de millones de euros.

Todos estos proyectos y muchos más que están funcionando hoy en día, no habrían sido posibles sin los datos abiertos. Y esto no ha hecho nada más que empezar.

2.1.3 ¿Cómo abrir datos?

Abrir los datos se ha convertido en una cuestión inevitable en empresas y organizaciones que incluyen los datos abiertos en sus procesos de innovación. Y Para que estos se consideren abiertos deben cumplir los siguientes criterios que forman la esencia del open data y que garantizan la interoperabilidad que es la capacidad de que diferentes empresas o sistemas puedan trabajar juntos.

- Accesibles

Los datos deben estar totalmente accesibles y con un coste de reproducción bajo, deben estar disponibles en internet para ser descargados de forma fácil e intuitiva y siempre que sea posible ordenados bajo catálogos que facilitan la identificación de los mismos.

- Reutilizables

Los datos deben ser ofrecidos con condiciones o bajo licencias que permitan su redistribución, modificación y explotación.

- No discriminatorios

Todo el mundo debe tener acceso a los datos sin discriminación contra el uso que se les va a dar, o discriminación por pertenecer a un grupo de personas, empresa o país. Los datos deben ser libres de derechos y no propietarios.

Abrir sus datos puede ser un desafío y aquí van algunos consejos.

1) Elegir la temática de los datos a abrir, ciencia, economía, transporte, etc.
2) Dar un contexto a los datos proporcionando información que ayudaría al usuario a localizar de forma fácil el contenido de los datos (metadatos).
    a. Título.
    b. Descripción.
    c. Palabras clave.
    d. Información legal.
    e. Fecha de publicación, y la última modificación.
    f. Frecuencia de actualización.
3) Estructurar los datos en formatos tecnológicos que los usuarios puedan usar XML, CSV, JSON… y usar API's si es posible.
4) Elegir una licencia abierta, este paso es obligatorio para determinar los derechos de propiedad intelectual de los datos. Hay varias licencias abiertas a elegir entre ellas y aquí nombramos algunas de mayor uso.
    a. licencia abierta Etalab.
    b. Public Domain Dedication and License.
    c. Open Data Commons Attribution License.
5) Publicar los datos en tu propia página web o en un portal de open data como el DataHub.
6) Promocionar a los datos. Una vez estos están disponibles se deben promocionar en las redes sociales, foros, y todos los medios de comunicación posibles para darles por conocer.
7) Mantener en la medida de lo posible los datos actualizados y frescos.

2.1.4 Plataformas de publicación de datos abiertos

Las plataformas de publicación de datos abiertos son un elemento clave en la estrategia de apertura de datos.

Ofrecen al usuario "productor" de los datos las herramientas y la asistencia necesaria para publicar sus datos como ofrecen al usuario "consumidor" de los datos una forma coherente y fácil para acceder a los datos.

En este capítulo veremos qué herramientas deben ofrecer estas plataformas para dar suporte tecnológico, sus principales características, así como cuáles son las plataformas más conocidas en España y en el resto del mundo.

2.1.4.1 Características principales

- Catálogo de datos: Es el elemento principal de cualquier plataforma de publicación de datos porque hace que los datos estén ordenados y centralizados de una manera que al usuario le sea fácil identificar y filtrar los datos que le interesan y para ello deben ofrecer soporte de metadatos. Además de ser compatibles con las normas de intercambios de datos.
- Acceso a los datos: Las plataformas no solo deben ofrecer acceso a los datos sino también deben garantizar varias formas de acceso a estos adaptándose a las necesidades de usuario garantizando cargas directas en formatos como XML, JSON, ZIP o mediante API's. Además, deben de ofrecer varios filtros de búsqueda (simples, combinados, avanzados).
- Actualización y visualización: Las plataformas deben garantizar mecanismos de actualización de datos asegurándose de que los datos no pierden sus referencias en ningún momento, además sería muy recomendable que las plataformas inviertan en la forma de visualización de los datos para hacer que la tarea del usuario final sea más amena.
- Aplicaciones desarrolladas: Una sección o galería con las aplicaciones que han sido desarrolladas a partir de los datos disponibles en la plataforma es muy útil porque sirve para enriquecer el contenido y sirve de inspiración a nuevos usuarios.
- Herramientas de comunidad: Hace falta promocionar los conjuntos de datos para obtener el mayor número posible de colaboradores incluyendo secciones para feedback en las plataformas como comentarios, opiniones, peticiones y valoraciones. También es muy recomendable la integración de las redes sociales en las plataformas, eso ayudara en compartir y a dar por conocer a los conjuntos de datos entre las comunidades de usuarios.

2.1.4.2 Plataformas en el mercado

A la hora de publicar sus datos existen dos tipos de opciones, la primera es crear su propia plataforma de publicación de datos, opción que requiere una gran inversión y esfuerzo y la segunda opción es recurrir a plataformas ya disponibles creadas por terceros, esta opción requiere menos inversión, pero tiene como inconveniente la pérdida de personalización de la plataforma según las necesidades.

- CKAN (https://ckan.org/): Es la plataforma líder mundial de gestión y publicación de conjuntos de datos. Es de software abierto desarrollada por "**the Open Knowledge Foundation**" desde 2006 y usada por varios gobiernos, organizaciones nacionales e internacionales como es el caso del gobierno australiano que tiene más de 35,000 conjuntos de datos publicados, gobierno de estados unidos 180,000 y el portal de datos de la unión europea que tiene más de 600,000 conjuntos de datos y muchas otras más instituciones.

  La plataforma dispone de todas las características vistas anteriormente como soporte de métodos, herramientas de comunidad, actualización y visualización de los datos de sección de aplicaciones desarrolladas etc.

- DKAN (http://getdkan.com): Es una plataforma de software abierto de publicación de datos abiertos desarrollada en dupral. Ofrece un gran número de herramientas para catalogar, publicar los datos además de incorporar herramientas de visualización de datos de forma fácil pudiendo montar cuadros de mando con solo arrastrar y soltar.
- Junar (http://www.junar.com/): Desde sus oficinas en Silicon Valley y Latinoamérica ofrece una plataforma de datos abiertos basada en la nube que permite acceso rápido a los datos. Además, a través de esta plataforma se puede elegir qué datos están destinados al uso público y cuáles son para uso interno. Permite también elegir el tiempo y la forma en la que van a ser visualizados los datos y puestos a disposición del público.
- Socrata (https://socrata.com/): Es una plataforma de publicación de datos en la nube que ofrece varias soluciones exclusivas de open data para ciudades y gobiernos como **Open Data & Citizen Engagement**, **financial tranparency**, **federal government** permitiendo que los datos sean descubiertos, utilizables y escalables tanto para los gobiernos como para los ciudadanos.
- Datos.gob.es (http://datos.gob.es/): Es la plataforma promovida por el ministerio de Energía, Turismo y Agenda digital. Para la iniciativa de datos abiertos en España está destinada a los ciudadanos, profesionales como a organizaciones públicas. Tiene una gran variedad de conjuntos de datos gubernamentales que cubren todos los sectores.
- OpenDataSoft (https://www.opendatasoft.fr): Es una plataforma de publicación de datos en la nube y nos ofrece una lista de los portales de open data en todo el mundo con una lista detallada de más de 2600 portales de todos los países del mundo con un mapa interactiva que ayuda en la búsqueda de los portales según la localización del país en el mapa.

Figura 4. Mapa demostrativo de los portales open data que ofrece la lista de OpendataSoft.

En definitiva, no hay una solución única a la hora de seleccionar la plataforma de publicación de datos abiertos, ya que la mayoría cumple con las principales características vistas anteriormente. Por lo tanto, será el coste, nivel de seguridad y capacidad de adaptarse a los requisitos concretos requeridos por su iniciativa los que harán que nos decantemos por una solución u otra.

2.2 Business Intelligence

2.2.1 Introducción al Business Intelligence

Conscientes de que los datos son el activo principal de las empresas, hace que los dirigentes de estas se encuentren frente a un gran problema a la hora de querer analizar una gran cantidad de datos no estructurados que provienen de varios sistemas no homogéneos para tomar decisiones importantes que afectan al rumbo de sus empresas.

De allí la necesidad del Business Intelligence, que nos ofrece las tecnologías y las herramientas necesarias para estructurar y transformar los datos en información y la información en conocimiento para luego tomar mejores decisiones en tiempo razonable, ofreciendo a los dirigentes un cuadro de mando que les orienta en su tarea de pilotar la empresa.

2.2.2 ¿Qué es Business Intelligence?

B.I representa el puesto de pilotaje de las empresas en la sociedad de la información, facilitando métodos y mecanismos de extracción y transformación de datos dentro de la organización. En primera aproximación se puede entender como una evolución de los sistemas de soporte a las decisiones (DSS, Decissions Support System). Pero una definición más correcta sería el conjunto de tecnologías, aplicaciones y prácticas usadas para ayudar a los usuarios de una organización a adquirir una mejor visión del negocio transformando los datos en información y la información en conocimiento y el conocimiento en acción.

B.I ayuda al negocio a recoger, integrar, difundir y presentar los datos como también les ayuda a observar, comprender, decidir y predecir, contestando a preguntas tal como:

- ¿Qué ha pasado?
- ¿Cómo y por qué ha pasado?
- ¿Qué pasará en el futuro?
- ¿Qué decisiones se deben tomar?

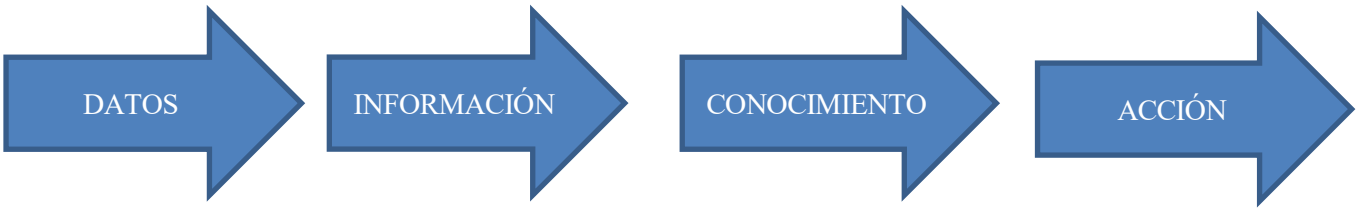


Figura 5. Utilidad de B.I.

Algunas tecnologías que forman parte del Business Intelligence son:

- Data Warehouse.
- Data marts.
- Integración de datos (herramientas ETL).
- Herramientas OLAP.
- Minería de datos.
- Reporting.
- Reglas de negocio.
- Análisis predictivo.

2.2.3 Necesidad y beneficios de Business Intelligence.

En este apartado intentaremos aclarar los beneficios que aporta una solución B.I a las empresas y por qué es necesario implementarla.

- Una solución B.I permite entregar la información correcta, en el momento correcto, a las personas correctas.
- Ofrece un acceso rápido a la información.
- Permite presentar de manera coherente y estructurada los datos.
- Permite aliñar la empresa alrededor de un conjunto coherente de indicadores claves de rendimiento (KPI-Key Performance Indicator).
- Facilita la toma de decisiones haciendo que este proceso sea más intuitivo.
- Permite Anticipar y prever las tendencias de los clientes / mercado.
- Facilita la campaña de marketing.
- Permite cruzar información de distintos departamentos.

2.2.4 Fases de desarrollo de un proyecto B.I

1) Estudiar el campo de negocio

Antes de empezar a programar cualquier cosa es muy importante realizar un estudio del campo de negocio de la empresa por la que se quiere implementar la solución B.I, también hay que familiarizarse con los términos usados dentro de esta última, podría serle también de gran utilidad familiarse con los programas y aplicaciones usadas dentro de la misma empresa.

2) Entrevistar a los actores principales

Otro elemento clave a la hora de desarrollar una solución B.I es conocer el organismo y los actores principales de este y que funciones desarrollan dentro del mismo, esto facilitará la tarea de formular las preguntas correctas a las personas correctas, como te ayudará a identificar la información requerida según a quién irá dirigida.

3) Dividir las necesidades del cliente en temas / categorías

Después de las reuniones se debe tener las ideas bastante claras como para poder definir los procesos a implementar y dividirlos en temas / categorías según el modelo de negocio que se quiere construir. A modo de ejemplo, si el proyecto es para una aseguradora de coches y el actor entrevistado es el responsable del área de accidentes pues pedirá el desarrollo de herramientas que le permiten conocer el número de accidentes, pérdidas...etc. esto significa que se va a tener que definir el modelo de negocio para los siniestros…(ClaimCenter). Pero si la persona entrevistada es la encargada de la parte de gestión de pólizas pues te pedirá informes sobre el número de pólizas vendidas, activas, canceladas …etc. (PolicyCenter). Por lo cual estaríamos hablando de dividir el proyecto en dos categorías y que son el centro de siniestros y el centro de pólizas.

4) Definir los procesos de negocio

Esta etapa es muy importante y en ella se debe agrupar los temas / categorías que pertenecen a los mismos procesos de negocio, para evitar duplicidad en estos. Siguiendo con el ejemplo anterior de la empresa aseguradora de coche, para definir los procesos de siniestros (ClaimCenter) y el de pólizas (PolicyCenter) hay que darse cuenta de que las pólizas deben estar creadas ya en el centro de pólizas y que no hace falta volver a crearlas en el centro de siniestros sino solo tener una referencia a estas así se evita la duplicidad en los procesos.

5) Modelado dimensional

Una vez identificados los procesos de negocio se accede al diseño conceptual del modelo dimensional identificando las tablas de hecho y las tablas de dimensiones. Volveremos sobre este punto más adelante en la sección de diseño de Data Warehouse para explicar qué son las tablas de hecho y las de dimensiones.

2.2.5 Modelo de madurez de un proyecto B.I (B.I.Maturity Model)

En esta sección lo que trataremos es de dar unas descripciones que permiten clasificar y saber el grado de madurez de la solución B.I implementada dentro de una empresa

- Nivel 1: Los datos se encuentran disponibles en hojas de cálculo como Excel u otro tipo de archivo similar y sus métricas son generalmente desarrolladas y usadas por usuarios individuales dentro de la misma empresa. Tienen un gran grado de inexactitud y poca seguridad lo que conlleva una toma de decisiones basada en datos no consistentes.
- Nivel 2: Operational Report. Los datos están disponibles en los sistemas de procesamiento de transacciones en línea (OLTP, Online Transaction Processing) de modo que los informes generados se enfocan en dar detalles de los datos transaccionales de un área específica del negocio, mientras que una solución B.I va mucho más allá permitiendo dar una visión global del negocio y cruzando información de todos los departamentos.
- Nivel 3: Implementación de un repositorio de datos Data Warehouse sobre el que se basa la generación de informes diarios. Además de formar un equipo que se dedica al mantenimiento de este repositorio de datos.
- Nivel 4: Aprovechar el repositorio de datos y despliegue de herramientas OLAP permitiendo generar información valiosa para tomar decisiones.
- Nivel 5: Data Mining es el proceso que permite analizar un gran conjunto de datos para sacar de ellos patrones de comportamiento de negocio, clientes, etc. Y así ayudar a la empresa a predecir el futuro y tomar ventaja.

2.2.6 Componentes de Business Intelligence.

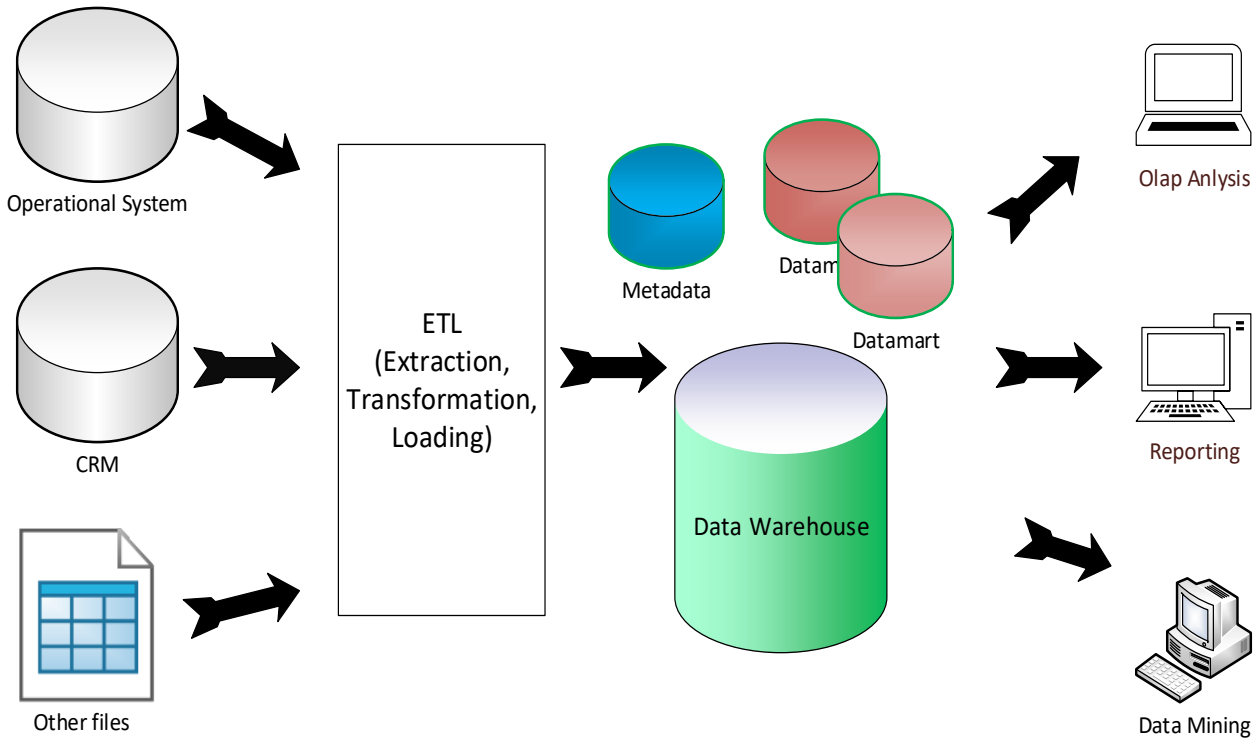


Figura 6. Esquema representativo de los componentes de la arquitectura Business Intelligence.

2.3 Diseño de Data Warehouse

2.3.1 Introducción

El Data Warehouse o el repositorio de datos es uno de los pilares más importantes del Business Intelligence. Antes lo que se hacía para hacer informes era montar queries complejas y lanzarlas directamente contra el sistema operacional (OLTP Online Transaction Processing), las queries tardaban debido al enorme cantidad de datos a analizar y la complejidad de las mismas además de las cargas de trabajo en paralelo que debían soportar estos sistemas, para finalmente generar informes en forma de hoja de cálculo.

Este enfoque de generar informes no ha durado mucho tiempo debido a su coste, poca eficiencia y también por ralentizar el sistema operacional. De allí surgió la necesidad de crear un nuevo sistema separado que recoja e integre los datos más importantes de las distintas fuentes de datos y almacenarlos ofreciendo fácil acceso a los mismos. Estamos hablando de un Data Warehouse.

2.3.2 ¿Qué es Data Warehouse?

El repositorio de datos Data Warehouse nació para aportar una solución a la problemática del continuo crecimiento de datos. Las empresas ya no estaban satisfechas con el comportamiento de los sistemas tradicionales de bases de datos por lo cual se hacía necesario un nuevo concepto de base de datos.

Hoy en día podemos encontrar empresas grandes como pequeñas que hacen uso de esta tecnología en todos los ámbitos (bancas, seguros, automóvil, medicina, etc.), para hacer que los procesos de toma de decisiones sean más eficientes.

El concepto de Data Warehouse fue formalizado por Bill Inmon en 1990, se trataba de construir una base de datos no volátil orientada a integrar los datos, homogeneizarlos y historificarlos con el objetivo de ayudar en el proceso de toma de decisiones en las empresas.

Según el propio Bill Inmon las características de un Data Warehouse son:

- Temático: El Data Warehouse debe estar organizado de forma que contenga la información que más valor va a aportar en el proceso de toma de decisiones, además los datos deben estar estructurados de forma temática contrariamente a los datos de los sistemas de operación que están estructurados de forma funcional. Asi si hay datos comunes entre varios temas estos no van a estar duplicados en el Data Warehouse.
- Integrado: El Data Warehouse es un proyecto que integra datos de varias fuentes en la empresa, por lo cual integrar datos es uno de los pilares del Data Warehouse, como es un proceso que determina la cantidad y la calidad de datos a entregar al usuario final. En este proceso hace falta también tener en cuenta las peculiaridades de las distintas fuentes de datos a integrar y eliminar las posibles inconsistencias que se pueden dar entre las distintas fuentes de datos a integrar.
- No volátil: Con el fin de tener una traza de la información de cómo ha ido cambiando a lo largo del tiempo el Data Warehouse no puede ser volátil, eso significa que la información contenida en él nunca puede ser borrada sino se inserta un nuevo registro por cada información actualizada en el sistema origen.
- Histórico: La históricificacion es una de las funcionalidades del Data Warehouse, permite el análisis de los indicadores y sus tendencias a lo largo del tiempo, cosa que los sistemas operacionales no pueden ofrecer porque en estos los datos reflejan el estado de los datos en el momento actual. Para conseguir este objetivo de históricificacion de datos una referencia al tiempo debe estar asignada a los datos en el Data Warehouse como por ejemplo la fecha de carga, o la fecha en la que se generó el dato en el sistema operacional.

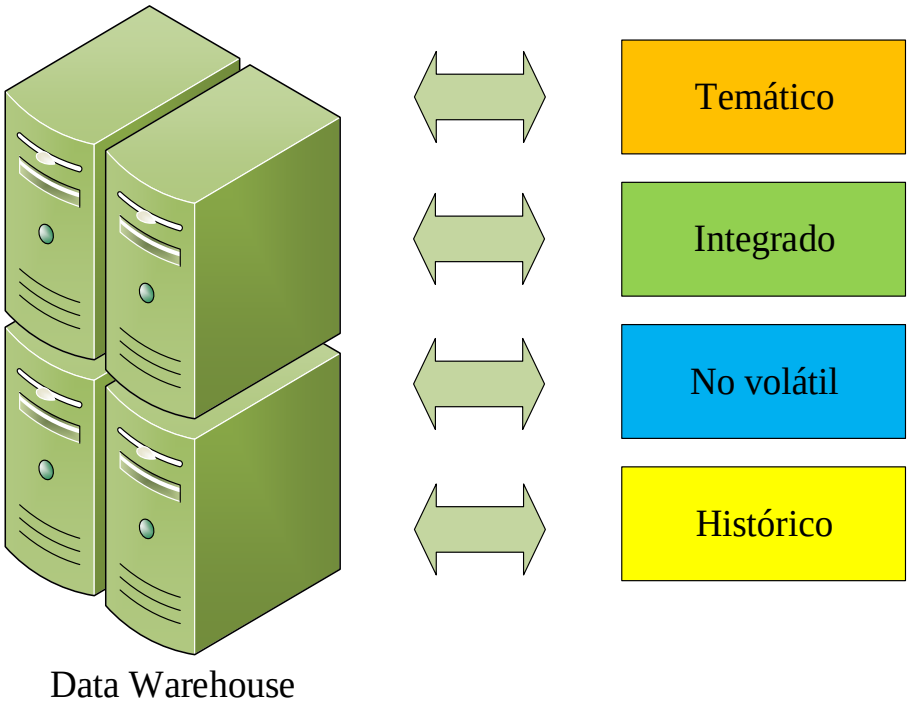


Figura 7. Características de Data Warehouse.

2.3.3 Modelado de datos

- Modelado temático

La técnica de modelado usada en el Data Warehouse es el modelado por temas, es una técnica que tiene como objetivo organizar y clasificar los datos de los sistemas operacionales en temas funcionales. Es una técnica basada en el modelo "entidad/relación" y es previa al modelado dimensional. Cada tabla en el Data Warehouse corresponde a un tema funcional dentro de la empresa.

- Modelado dimensional

El modelado dimensional o también el llamado modelado OLAP es una alternativa al modelo relacional, está técnica que tiene el objetivo de representar los datos de forma intuitiva y lógica además de representarlos en forma de cubos de dimensión N > 3. Esto hace que los datos ya no se representan en tablas sino en cubos o híper cubos, estos cubos van dedicados y concentrados a actividades de la empresa.

- Tablas de hecho

Las tablas de hecho representan los procesos de negocio en el seno de la empresa y que son objeto de análisis. Por ejemplo (ventas, pólizas de un seguro, accidentes de pólizas, reclamaciones, pagos, devoluciones, etc.). Las tablas de hecho guardan generalmente dos tipos de atributos, las medidas de un proceso y las claves subrogadas a las tablas de dimensiones relacionadas con el proceso.

H_POLI	TABLA DE HECHOS DE POLIZAS
ID_H_POLI	Índice interno de la tabla (PK)
CODE_POLI	Código de póliza
ID_SRCE	Clave foránea en el sistema fuente (FK)
DATE_SRCE	Fecha de modificación del registro en el sistema operacional
DATE_LOAD	Fecha de carga del registro en el Data Warehouse
ID_D_SRCE_SYST	Clave foránea a la tabla de dimensión que contiene los identificadores de las posibles fuentes de datos (FK)
ID_D_STAT_POLI	Clave foránea a la definición del estado de la póliza (opend, closed, etc.) (FK)
ID_D_TYPE_POLI	Clave foránea a la definición del tipo de póliza (renewal, cancellation, etc.) (FK)
ID_D_ZONE_NIGHT	Clave foránea a la definición de zona donde aparcar el coche por la noche (FK)
ID_D_ZONE_WORK	Clave foránea a la definición de zona donde aparca el conductor al ir a trabajar (FK)
ID_D_VEHI_OWNER	Clave foránea a la definición del propietario del coche (FK)

Tabla 1. Modelo conceptual de la tabla de hechos de pólizas de seguro de coches.

Para definir una tabla de hecho se debe entender muy bien la información que guardará esta, localizando la información a cargar en la fuente de datos origen, además se debe definir el nivel de granularidad de la tabla y las dimensiones a asignar a esta y que definen su nivel de detalle. Por ejemplo, en la tabla 1 encontramos el hecho que es la póliza en sí y encontramos también claves foráneas a la información a nivel de estado de la póliza, fuente de datos de donde se ha cargado...en las tablas de dimensiones.

- Tablas de dimensiones.

Las tablas de dimensiones son tablas que definen el nivel de granularidad de la tabla de hecho y recogen los diferentes puntos de análisis de un hecho, por ejemplo, una póliza se puede analizar dependiendo del tipo de esta (submission, policy change, renewal, draft) o de la fuente de donde esta ha sido cargada (ClaimCenter, PolicyCenter, BillingCenter) o por donde aparca el coche de esta póliza (d_zone_night, d_zone_work) etc.

D_TYPE_POLI	TABLA DE DIMENSIÓN DE TIPO DE POLIZA
ID_D_TYPE_POLI (PK)	Clave primaria de la tabla de dimensión.
ID_D_SRCE_SYST (UK	Clave foránea a la tabla de dimensión de las posibles fuentes de datos.
ID_SRCE (UK)	Clave foránea al dato en la fuente origen.
DATE_LOAD	Fecha de carga de dato en el Data Warehouse.
CODE_TYPE_POLI	Código que describe tipo de la póliza.
DESC_TYPE_POLI	Descripción del tipo de póliza.
SWIT_RETI	Descripción de si el registro está retirado o no del sistema origen.

Tabla 2. Modelo conceptual de la tabla de dimensión que define los tipos pólizas.

D_STAT_POLI	TABLA DE DIMENSIÓN QUE DESCRIBE PROPIETARIO DE VEHICULO
ID_D _STAT_POLI	Clave primaria de la tabla de dimensión.
ID_D_SRCE_SYST	Clave foránea a la tabla de dimensión de las posibles fuentes de datos.
ID_SRCE	Clave foránea al dato en la fuente origen.
DATE_LOAD	Fecha de carga de dato en el Data Warehouse.
CODE_ STAT_POLI	Código que describe el estado de la póliza.
DESC_ STAT_POLI	Descripción del estado de la póliza.
SWIT_RETI	Descripción de si el registro está retirado o no del sistema origen.

Tabla 3. Modelo conceptual de la tabla de dimensión que define los estados de póliza.

D_ZONE_NIGHT	TABLA DE DIMENSION ZONE NIGHT
ID_ D_ZONE_NIGHT	Clave primaria de la tabla de dimensión.
ID_D_SRCE_SYST	Clave foránea a la tabla de dimensión de las posibles fuentes de datos.
ID_SRCE	Clave foránea al dato en la fuente origen.
DATE_LOAD	Fecha de carga de dato en el Data Warehouse.
CODE_TYPE_ZONE_NIGHT	Código que describe la zona de aparcamiento de noche.
DESC_ TYPE_ZONE_NIGHT	Descripción de la zona de aparcamiento de noche.

SWIT_RETI	Descripción de si el registro está retirado o no del sistema origen.

Tabla 4. Modelo conceptual de la tabla de dimensión que define los estados de póliza.

- Estructura de la base de datos.

En el Data Warehouse se usan generalmente dos tipos de esquemas, el primero llamado esquema en estrella y el segundo copo de nieve.

⇨ ***Esquema en estrella***

En el esquema en estrella la tabla de hechos que contiene los hechos a analizar es el elemento central de este esquema y contiene referencias (claves foráneas) a las tablas de dimensiones que la describen en detalle. De modo que cada dimensión está descrita por una única tabla. Siguiendo con el ejemplo visto anteriormente y usando las mismas tablas definidas (solo usaremos las 3 tablas de dimensiones definidas) un esquema de la base de datos sería:

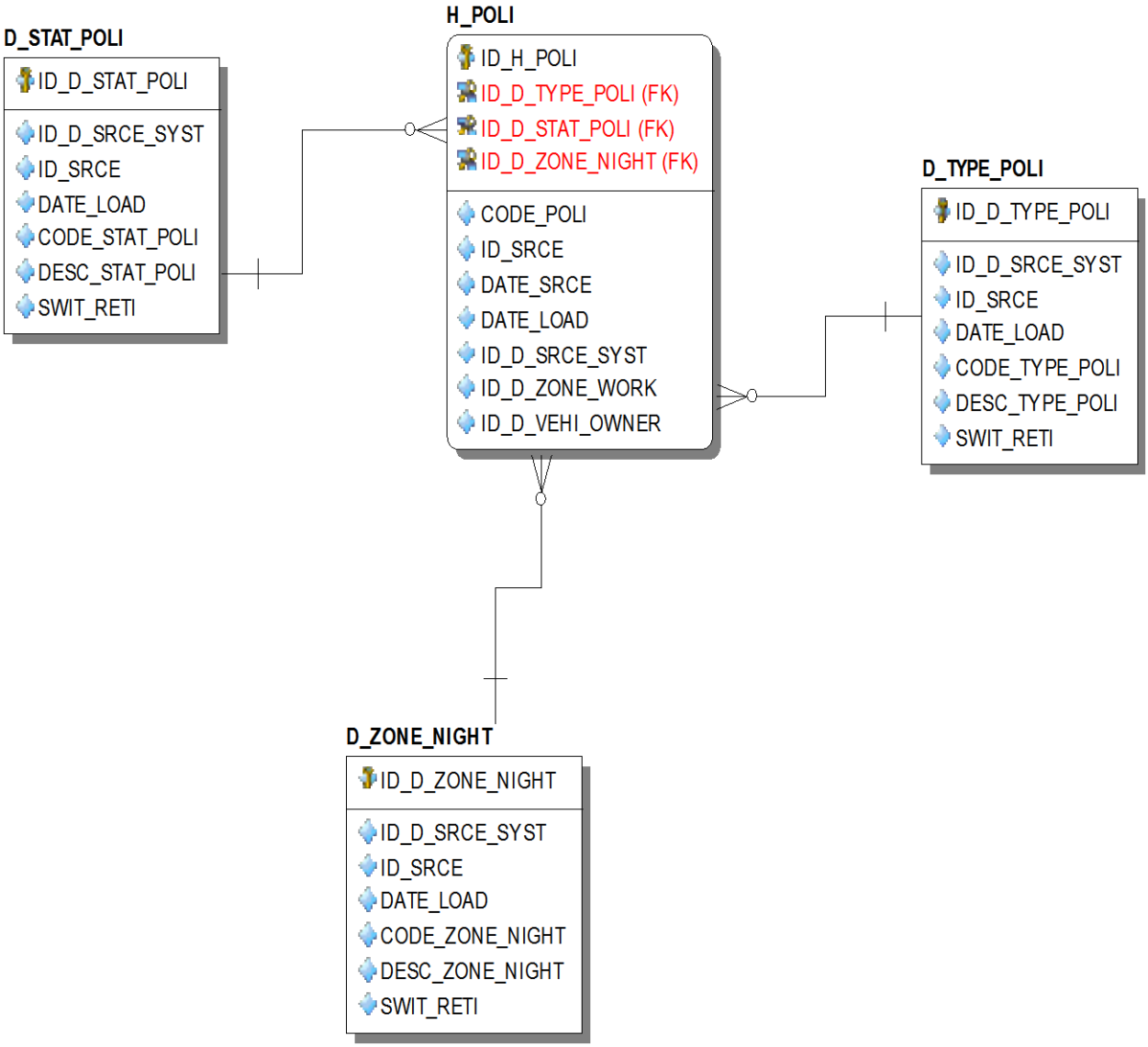


Figura 8. Ejemplo de esquema en estrella.

⇨ ***Esquema de copo de nieve***

Este esquema es el menos usado por su complejidad y es una variedad del esquema en estrella, se trata de que además de que haya un elemento central que es la tabla de hecho y que esta contenga claves foráneas a otras tablas de dimensiones que la describen, pues que las tablas de dimensiones también contengan claves foráneas a otras tablas que las describen también, así se consigue que no haya redundancia de información

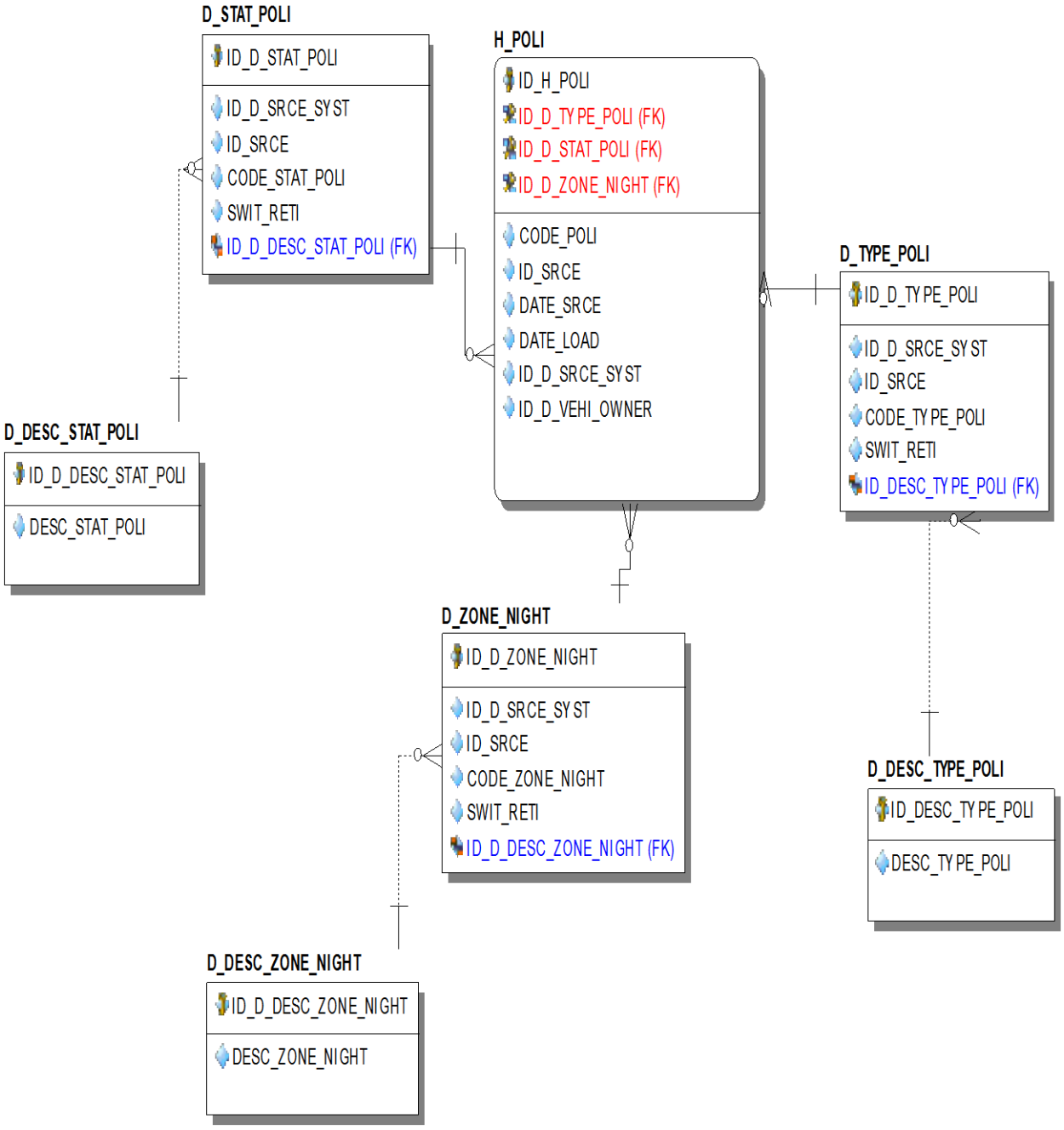


Figura 9. Ejemplo de esquema copo de nieve.

2.3.4 Arquitectura de Data Warehouse

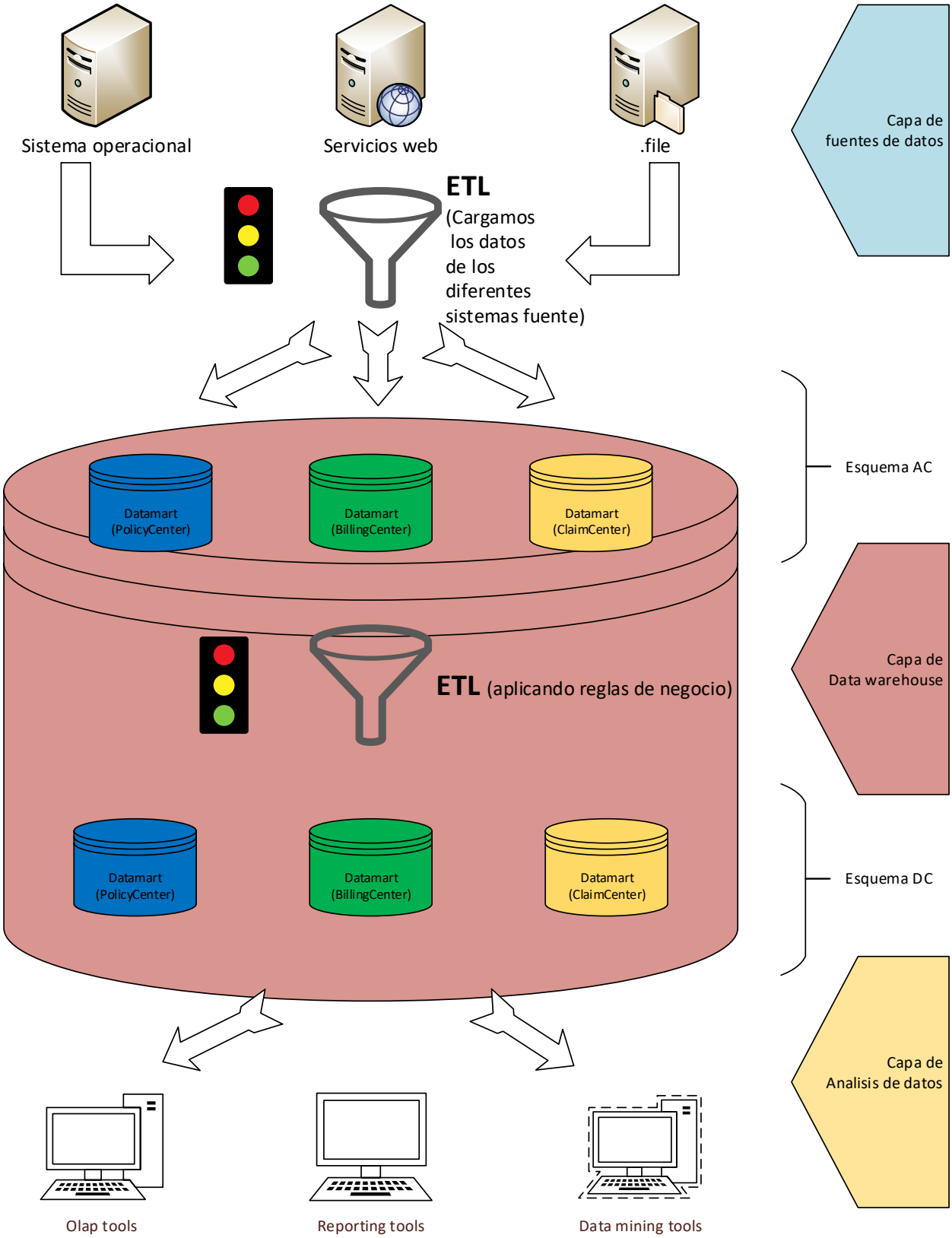


Figura 10. Esquema de arquitectura de Data Warehouse implementada para una empresa de seguros.

En la figura se puede apreciar que primero se cargan los datos desde el sistema origen programando procesos ETL's específicos, para luego meter estos datos en sus Data Marts correspondientes. Una vez cargados los datos brutos en sus correspondientes Data Marts (esquema AC) se procede a consolidar estos datos aplicando las reglas de negocio que dan sentido a los datos para luego volcar los datos en el esquema DC. Una vez los datos están consolidados en el esquema DC estos estarán ya listos para ser explotados por los usuarios finales.

2.3.5 Data Warehouse vs OLTP

Los datos en los sistemas operacionales se actualizan constantemente pisando los datos antiguos de modo que no guardan un histórico de los mismos, estos datos representan únicamente el momento presente en el sistema operacional. Mientras que los datos en el Data Warehouse representan copias de los datos en el sistema operacional y como estos han ido cambiando a lo largo del tiempo.

Básicamente los datos en el Data Warehouse nunca serán borrados. Esto permite realizar análisis del estado de la empresa desde que empezó el proyecto B.I hasta el momento presente.

Otra diferencia es que el Data Warehouse opera en modo de lectura por lo cual las exigencias de DBMS (Data Base Management System) son diferentes de las de sistemas operacionales cuyas aplicaciones exigen la implementación de técnicas avanzadas de gestión de transacciones.

Las queries en los OLTP ejecutan transacciones que generalmente leen/escriben un número pequeño de registros desde/hacia un número de tablas relacionadas con relaciones simples, mientras que las queries de OLAP necesitan realizar análisis multidimensional que exige procesar un gran número de datos en unas tablas multidimensionales.

En el siguiente punto haremos una comparativa resumiendo las principales diferencias entre un OLTP y Data Warehouse.

Características	OLTP	Data Warehouse
Usuarios.	Miles.	Centenares.
Acceso.	Cientos de registros en modo escritura y lectura.	A millones de registros en modo lectura.
Objetivo.	Soportar actividades transaccionales diarias.	Soportar la toma de decisiones.
Datos.	Detallados.	Resumidos.
Tiempo de cobertura de datos.	Solo datos actuales.	Datos actuales más históricos.
Modelo.	Normalizado.	Desnormalizado, multidimensional.
Orientación.	Aplicación.	Negocio.

Tabla 5. Comparativa entre base de datos OLTP y Data Warehouse.

2.4 Diseño de procesos ETL

Una vez hecho el diseño del Data Warehouse la siguiente etapa es diseñar los procesos ETL que se encargan de integrar los datos que provienen de las diferentes bases de datos fuente, para ofrecer una visión global de los datos de la empresa.

Las herramientas ETL ayudan a programar procesos cuyas principales características son:

- Extracción de datos.
- Transformación y limpieza de datos.
- Carga de datos.

La programación de estos procesos consume el mayor tiempo de un proyecto de B.I que puede llegar hasta el 80%, de allí su gran importancia en el proyecto.

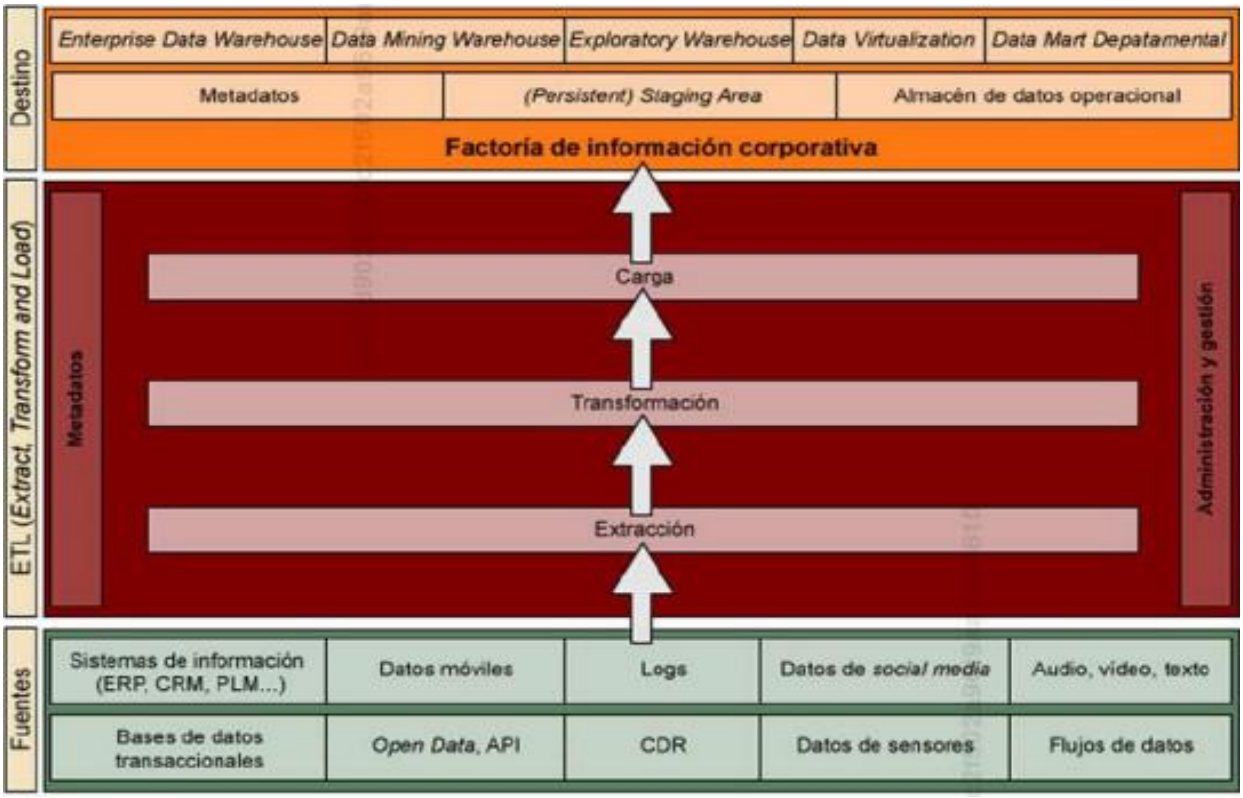


Figura 11. Descripción de la funcionalidad de las herramientas ETL. Fuente [8].

2.4.1 Extracción de datos

En esta fase se extraen los datos relevantes de los sistemas origen. se pueden programar procesos estáticos para alimentar por primera vez el Data Warehouse. Pero una vez realizada la primera extracción se programan procesos dinámicos que se encargan de actualizar el Data Warehouse regularmente y con una frecuencia fijada previamente que depende de la cantidad de datos a extraer y tiempo que tarda la extracción.

En cada ejecución de los procesos ETL dinámicos solo se extraen los datos que han cambiado en el sistema origen, desde la última extracción que se hizo. Es lo que llamamos extracción incremental.

La extracción incremental se basa a menudo en el log mantenido por la base de datos operacional donde se asigna una marca de tiempo a cada registro que ayuda a identificar cuando este fue modificado por última vez.

Se aprovecha esta marca de tiempo para hacer que los procesos de extracción sean más óptimos y eficientes.

2.4.2 Transformación y limpieza de datos

La transformación y la limpieza de datos es una fase muy importante en Data Warehouse porque es la que define la calidad de los datos y evita la información errónea o no veraz.

Algunos de los errores que hacen que el proceso de limpieza de datos sea imprescindible son.

- Duplicidad de datos.
- Inconsistencia de datos que están lógicamente asociados.
- Datos que faltan.
- Uso indebido de un campo.
- Valores erróneos o imposibles de algún campo.
- Valores inconsistentes debido al uso de prácticas diferentes en bases origen diferentes.

Esta fase aporta mayor consistencia, fiabilidad y homogeneizacióna los datos mientras que la transformación convierte los datos de su formato en la base de datos origen al nuevo asignado en el Data Warehouse.

La fase de transformación exige un mapping previo hecho a la hora del diseño conceptual del modelo del Data Warehouse donde se ha tenido que definir los nombres de los campos, longitud, formato, origen, etc.

Las principales características de la fase de transformación son:

- Hacer el cálculo de nuevos valores.
- Generación de nuevos campos.
- Matching o asociación de campos equivalentes en diferentes sistemas origen.
- Unión de datos de múltiples fuentes.

Los procesos de limpieza y transformación están estrechamente relacionados en las herramientas ETL.

2.4.3 Carga

Cargar los datos una vez extraídos, limpiados y transformados en el Data Warehouse, es la última a etapa a llevar a cabo en los procesos ETL y se puede hacer de 2 formas:

- Refrescar Los datos en el Data Warehouse rescribiéndolos por completo. Esto significa que los datos antiguos se reemplazan. Esta técnica es usada normalmente en combinación con la extracción estática para rellenar inicialmente el Data Warehouse.
- Actualización de los datos. La actualización se realiza sin borrar o modificar datos preexistentes. Esta técnica se utiliza en combinación con la extracción incremental para actualizar los almacenes de datos regularmente.

2.5 Herramientas OLAP

2.5.1 Introducción

Las herramientas OLAP (OnLine Analytical Processing) permiten la explotación del Data Warehouse y dan a los usuarios finales cuyas necesidades no pueden ser satisfechas con simples informes de Reporting, la oportunidad de realizar varios tipos de análisis que no son fáciles de definir de antemano, además de analizar y explotar datos de forma interactiva.

2.5.2 Características

Las características principales de los productos OLAP y que estos deben de ofrecer son:

- Rapidez: Tiempo de respuesta a las solicitudes de los usuarios entre 1 y 20 segundos (se utilizan pre-cálculos en productos OLAP para reducir la duración de las consultas).
- Análisis: Ofrecer a los usuarios las herramientas oportunas para poder construir sus cálculos y análisis y hacer frente a todas las estadísticas y exigencias del negocio.
- Compartido: El sistema debe crear el contexto de confidencialidad y preservarlo, además de gestionar los permisos de lectura y escritura a asignar a los diferentes usuarios.
- Multidimensional: los productos OLAP deben de ofrecer vistas multidimensionales además de soportar la jerarquía de dimensiones.
- Información: debe soportar una gran cantidad de datos e información.

2.5.3 Tipos de sistemas OLAP

Existen principalmente tres formas de implementar una herramienta de análisis OLAP y en el siguiente punto trataremos de ver la arquitectura de cada una de estas soluciones, sus ventajas y sus inconvenientes

- ***ROLAP (Relational OnLine Analytical Processing)***

Es un sistema que usa una base de datos relacional con adaptaciones específicas para OLAP, de modo que la base de datos relacional está preparada para reaccionar como si fuese una base OLAP.

En la siguiente figura podremos apreciar más la arquitectura de este sistema:

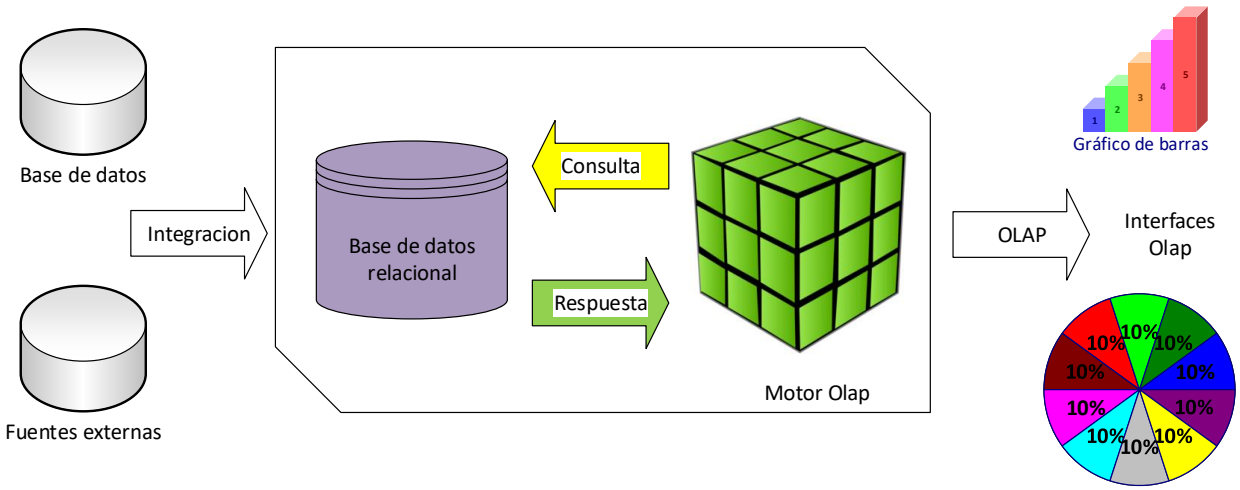


Figura 12. Sistema ROLAP.

En la figura 12 se puede apreciar

- El uso de un SGBD relacional para almacenar los datos del Data Warehouse.
- El motor OLAP es complementario y ofrece una vista multidimensional del Data Warehouse, además de ayudar a hacer cálculos y agregaciones a varios niveles.
- El motor OLAP genera queries SQL adaptadas al esquema relacional de la base de datos de Data Warehouse, transformando queries multidimensionales M en queries relacionales R.

- ***MOLAP (Multidimensional OnLine Analytical Processing)***

Es un sistema que usa una base de datos multidimensional propia SGDBM que almacena y gestiona datos multidimensionales (son la aplicación física del concepto OLAP). Estos sistemas tienen un buen rendimiento debido a los pre-cálculos que efectúan a todos los niveles de jerarquía del modelo generando grandes cantidades de información.

En la siguiente figura podremos apreciar más la arquitectura de este sistema:

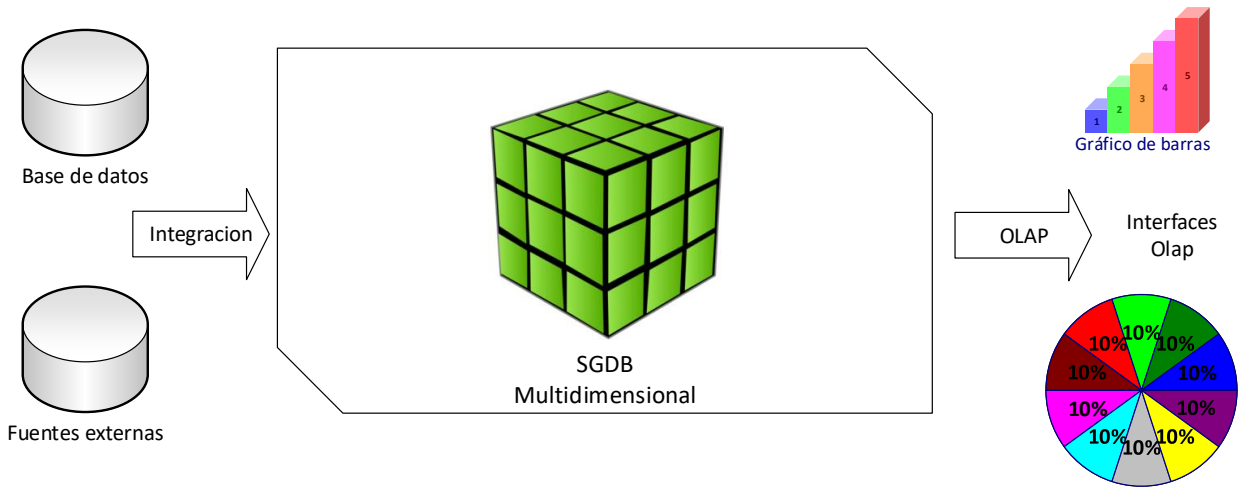


Figura 13. Sistema MOLAP.

En la figura 13 se puede apreciar:

- Los datos se cargan en la base de datos multidimensional MOLAP, una vez estos cargados se efectúan una serie de pre-cálculos y agregaciones en todos los niveles de jerarquía de dimensiones.
- Una vez rellanada la base de datos multidimensional (SGDBM) se generan índices y algoritmos para mejorar los tiempos de acceso de las consultas.
- Una vez hecho el proceso de compilación el usuario final podrá usar la base de datos multidimensional (SGDBM).
- Estos sistemas tienen la ventaja de tener un tiempo de consulta muy optimizado a costa de que requieren unos pre-cálculos intensivos de compilación cada vez que el modelo multidimensional cambia, este proceso se vuelve más costoso aun cuando el volumen de datos es muy grande.
- Generalmente los sistemas MOLAP se usan para pequeñas Data Warehouse y cuyo modelo multidimensional no cambia mucho.
- Algunos comerciales de esta tecnología son: MDDB de SAS Institute, Oracle Express-server de Oracle, DB2 OLAP Server de IBM, Cognos PowerCubes.

- ***HOLAP (Hybrid OnLine Analytical Processing)***

Hybrid-OLAP es una combinación entre las dos arquitecturas descritas anteriormente de forma que el HOLAP intenta quedarse con lo bueno de cada una de las soluciones descritas.

En la siguiente figura podremos apreciar más la arquitectura de este sistema:

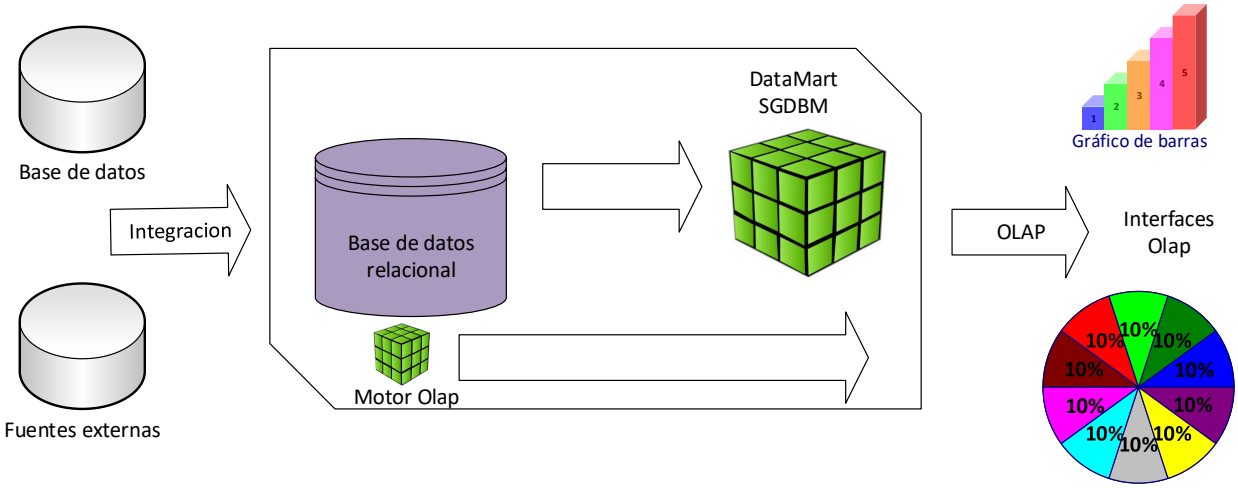


Figura 14. Sistema HOLAP.

Varios sistemas comerciales usan la solución HOLAP para:

- Manipular los datos del Data Warehouse (SGBD Relacional) con la ayuda de un motor OLAP
- Manipular los datos de Data Marts con uso de SGDB Multidimensional

Lo que permite disponer de un repositorio de datos de un tamaño grande, pero con tiempo de respuesta satisfactorio. DB2 Olap server, Olap express Server son algunos productos HOLAP.

3 TECNOLOGÍAS

La tecnología no es nada. lo importante es que tengas fe en la gente, que sean básicamente buenas e inteligentes, y si les das herramientas, harán cosas maravillosas con ellas.

- Steve Jobs -

En este capítulo introduciremos las tecnologías que vamos a usar para desarrollar nuestra solución B.I haciendo hincapié sobre sus principales características y los requerimientos necesarios para su instalación, con el objetivo de facilitar el entendimiento del caso práctico que llevaremos a cabo más adelante y que servirá de ejemplo para ver como todas estas tecnologías entran en juego y en qué orden.

3.1 PostgreSQL

3.1.1 ¿Qué es PostgreSQL?

PostgreSQL es un avanzado sistema de gestión de base de datos relacional orientado a objetos, robusto y potente capaz de manipular grandes volúmenes de datos con mucha fiabilidad incluso en situaciones críticas.

Es un software libre desarrollado por la comunidad internacional con la ayuda de miles de desarrolladores y varias empresas, está publicado bajo la licencia BSD.

PostgreSQL funciona en los principales sistemas operativos: Windows, Linux, UNIX (HP-UX, SGI IRIX, BSD, Mac OS X, Solaris, Tru64, AIX).

PostgreSQL es compatible con la mayoría de estándares (estándar SQL) y es compatible con una docena de lenguajes de programación tal como Java, Perl, Python, Ruby, C / C ++.

Figura 15. Logo PostgreSQL.

3.1.2 Características

- Es una base de datos 100% ACID (**A**tomicity, **C**onsistency, **I**solation and **D**urability, en español **A**tomicidad, Consistencia, **A**islamiento y **D**urabilidad).
- Soporta distintos tipos de datos además de que permite la creación de tipos propios.
- Incluye herencia entre tablas.
- Online/hot backups (permite copias de seguridad en caliente).
- Juegos de caracteres internacionales.
- Regionalización por columna.
- Multi-Version Concurrency Control (MVCC) significa Acceso concurrente multi-versión en español y quiere decir que PostgreSQL no bloquea al usuario que está leyendo de una tabla mientras que otros escriban en ella. De modo que cada usuario obtiene una visión consistente de la tabla una vez realizado el commit sobre ella.
- Multiples métodos de autentificación.
- Acceso encriptado via SSL.
- Actualización in-situ integrada.
- Completa documentación.
- Alta concurrencia.

3.1.3 Requerimientos

Los requerimientos mínimos para instalar PostgreSQL son:

- 8 megabytes de memoria RAM.
- 40 megabytes de espacio en el disco duro para el código fuente, ejecutables, y bases de datos básicas.

3.1.4 Ventajas

- Importante ahorro en los costes.
- Su legendaria fiablidad y estabilidad, ninguna empresa ha llegado a informar nunca de que su PostgreSQL haya dejado de funcionar ni una vez.
- Extensible, con código fuente disponible a todo el mundo.
- Multiplataforma.
- Diseñado para entornos de alto volumen.
- Ofrece herramientas gráficas de diseño y administración de base de datos.

3.2 Talend open studio for data integration

Figura 16. Logo Talend Open Studio for Data Integration.

3.2.1 ¿Qué es Talend Open Studio for Data Integration?

Talend es un líder de los sistemas de integración de datos que ofrece varias soluciones de (Data quality, Big data, data Integration, etc.) unas gratuitas y otras comerciales.

Talend Open Studio for Data Integration es una de las soluciones de software libre que nos brinda este gigante para resolver los retos de integración de datos en las organizaciones que manejan gran volumen de datos.

Este software además de ser libre tiene una curva de aprendizaje reducida ya que Talend pone a disposición de sus usuarios documentación en línea, tutoriales, fórum, blogs, bugtracker, etc. Además de que existe una gran comunidad que ayuda a resolver las dudas y problemas que se pueden encontrar.

3.2.2 Características

- Dispone de más de 900 componentes para
  - La gestión de archivos: abrir, mover, comprimir, descomprimir, etc.
  - El control de flujos de datos y su integración con las tareas principales.

- Dispone de conectores a RDBMS como Oracle, Teradata, Microsoft SQL server, PostgreSQL, etc.,

  a Packetes app como SAP y Microsoft Dynamics, además de disponer de conectores a tecnologías como Dropbox, SMTP, FTP/SFTP, etc.
- Permite una Arquitectura escalable.
- Dispone de una interface muy intuitiva.
- Entorno IDE basado en eclipse.
- Desarrollado en java (se puede ejecutar en Linux/Windows/Mac).
- Genera un .JAR que lanzaremos desde un planificador de tareas programadas (Rundeck).
- Puede ser parametrizado para distintas conexiones.
- Muchas compañías grandes figuran como sus usuarios (Panasonic, DHL, L'oreal, Pioneer, Ups, etc.).

3.2.3 Requerimientos

Tabla 6. Requerimientos de instalación de Talend.

Sistema operativo	**Versión**	**Procesador**	**Java JDK/JRE**	**Tipo de suporte**
Linux Ubuntu	12.04	64 bits	Oracle Java 7	Recomendado
Linux Ubuntu	12.04	32/64 bits	Oracle Java 6	Soportado
Linux Ubuntu	10.04/13.04	32/64 bits	Oracle Java 6/7	Soportado
Redhat Linux Enterprise Server Edition/CentOS	5.3 à 5.6	32/64 bits	Oracle Java 6	Soportado
Redhat Linux Enterprise Server Edition/CentOS	6.X (>=6.1)	64 bits	Oracle Java 7	Soportado
SUSE SLES	10/11	32/64 bits	Oracle Java 6/7	Soportado
Microsoft Windows	8	64 bits	Oracle Java 7	Recomendado
Microsoft Windows	7	64 bits	Oracle Java 7	Recomendado
Microsoft Windows	8.1	64 bits	Oracle Java 7	Soportado
Microsoft Windows	Vista SP1	32/64 bits	Oracle Java 6/7	Soportado
Microsoft Windows	7	32 bits	Oracle Java 6/7	Soportado
Microsoft Windows	XP SP3	32/64 bits	Oracle Java 6	NO RECOMENDADO

3.3 Tableau

Figura 17. Logo Tableau.

3.3.1 ¿Qué es Tableau?

Tableau es un software de Business Intelligence que proporciona herramientas de gran capacidad de visualización, representación y análisis de datos que luego hacen la tarea de toma de decisión más intuitiva y simple.

Tableau ofrece los siguientes productos que se pueden elegir según la necesidad de cada empresa:

- Tableau Desktop: Es una solución que se puede usar con solo instalar el ejecutable que facilita Tableau en su ordenador local
- Tableau Server: Es una solución hospedada en los servidores de la empresa cliente (In-situ)
- Tableau Online: Es una solución hospedada en los servidores de Tableau por lo cual la empresa cliente no tiene que preocuparse por la compra de hardware ni las actualizaciones de software, etc.

Tableau ha sido colocado como líder en el cuadrante mágico de Gartner de 2017 para plataformas de Inteligencia de negocios y Análisis por quinto año consecutivo. Como se puede ver en la siguiente figura.

Figura 18. Cuadrante mágico de gartner para plataformas de Inteligencia de negocios (febrero 2017).

3.3.2 Características

- Exploración y análisis de datos.
- Rápido e intuitivo, basta con arrastrar y soltar elementos para montar su propio cuadro de mando.
- Permite la combinación de varias fuentes de datos para su análisis (Excel, bases de datos, etc.).
- Dispone de conexiones a todo tipo de bases de SQL, NoSql, Big Data, Excel, etc.
- Sus conexiones son directas eliminando los pasos previos de creación de cubos.
- Representación gráfica de los datos facilitando plantillas y consejos que pueden ser de gran ayuda.
- Creación de informes y cuadro de mandos.
- Cartografía, gracias a su sistema de geo codificación hace posible representar datos en mapas de forma. sencilla e intuitiva.
- Difusión e intercambio de informes y cuadro de mandos gracias a URL de intercambios.

3.3.3 Requerimientos

Haremos hincapié sobre los requerimientos de Tableau Desktop Visto que es el producto de Tableau que vamos a usar en nuestro proyecto, pero para obtener más detalles de los requerimientos de Tableau server se puede consultar el siguiente enlace del proveedor.

- https://www.tableau.com/es-es/products/desktop/download#system-requirements

Windows

- Windows 10, 8, 7, Vista o XP SP3, Windows Server 2012, 2008, 2003.
- Versiones de Windows de 32 o 64 bits.
- Procesador Intel Pentium 4 o AMD Opteron.
- Espacio libre en disco de 250 MB.
- Se recomienda una profundidad de color de 32 bits.

Mac

- iMac >= 2007.
- MacBook aluminio >= 2009.
- MacBook Pro >= 2007.
- MacBook Air >= 2009.
- Mac mini >= 2009.
- OS X 10.8.1 o posterior (se recomienda 10.8.6 o más reciente).
- Memoria de 2 GB.
- Espacio libre en disco de 500 MB.

3.4 Rundeck

Figura 19. Logo Rundeck.

3.4.1 Que es Rundeck

Rundeck es un planificador de tareas, una herramienta open source de automatización de procesos (Jobs). Dispone de una consola, línea de comandos y una WebAPI que pemiten ejecutar procesos de forma automática y en varios conjuntos de nodos.

Rundeck hace el role de jefe de orquestra permitiendo ejecutar comandos y lanzar procesos en servidores o en local de forma automática y programada.

3.4.2 Características

- Dispone de Web API.
- Dispone de una interfaz web de programación y creación de Jobs intuitiva.
- Permite la ejecución de comandos distribuidos.
- Permite definir varios proyectos y dentro de cada proyecto definir sus propios Jobs.
- Permite programar Jobs de forma escalable.
- Dispone de notificación en caso de fallo de ejecución de algún Job y también en caso de ejecución correcta.
- Dispone de historial de ejecución de todos los Jobs.

3.4.3 Requerimientos

- Linux: distribuciones más recientes.
- Windows: >= XP, Server.
- Mac >= OS X 10.4.
- Versión más reciente de Java 1.7.

3.5 Embarcadero ER/Studio Data Architect 10.0

Figura 20. Logo ER/Studio Data Architect.

3.6 ¿Qué es ER/Studio Data Architect?

ER/Studio es una herramienta que ayuda a los modeladores y arquitectos de bases de datos a crear y documentar bases de datos en entornos empresariales dinámicos y complejos.

ER/Studio data Architect fue desarrollado por la empresa Embarcadero technogies que ha sido luego comprada por IDRA.

3.7 Características

- Dispone de una interfaz intuitiva y fácil de aprender.
- Mejora la calidad de los datos.
- Facilita compartir información y el modelo de la base de datos de la empresa entre I.T y el negocio.
- Permite identificar el origen de los datos y así la integración de los mismos.
- Permite documentar la estructura de la base de datos y por tanto se hace más fácil entender la base de datos para poder operar con ella con eficacia.

3.8 Requerimientos

- 670MB espacio libre en disco.
- 2GB RAM.
- Windows >= XP (32 y 64) bits.
- Resolución 1024x768.

4 CASO PRÁCTICO: ANÁLISIS

"Without data, you are just another person with an opinion"

- W. Edwards Deming -

.

Este capitulo tiene como objetivo analizar y describir el contexto general del proyecto que queremos llevar a cabo en el seno de la empresa EUIGS y que consiste en el diseño y la implementación de un sistema de toma de decisiones para ayudar a contestar las preguntas del negocio ofreciéndole la información y las herramientas necesarias en que podrá apoyarse en la toma de decisiones.

4.1 Contexto

En un Mercado donde la competencia está siendo cada vez más grande y feroz las compañías de seguros se encuentran con la necesidad de reinventarse para reducir costes, fidelizar sus clientes y seducir a clientes nuevos, para ello adoptan nuevos sistemas de toma de decisiones que les puedan ayudar a cumplir con estos objetivos.

La problemática que afrontamos hoy es cómo podemos distribuir la flota de guras de la que dispone la compañía de forma óptima para:

- Reducir costes de desplazamiento gasolina y desgaste de la grúa.
- Reducir el tiempo de respuesta de la grúa de forma que esta pueda llegar al lugar de la incidencia en el mínimo tiempo posible.
- Satisfacer una de las necesidades más importante del cliente de cualquier seguro de coches y que consiste en ser atendido en el menor tiempo posible en caso de que lo necesite.

Para dar solución a estas necesidades y otras mas preocupaciones vamos a intentar aprovechar el potencial que nos ofrece el Open Data que será nuestra fuente principal de datos, además de aprovechar las herramientas que nos brinda el Business Intelligence para diseñar e implementar un sistema decisional (Data Mart) que en un futuro lo integraremos dentro de la base de datos Data Warehouse de la empresa.

Para ello deberíamos.

- Buscar el Dataset (conjunto de datos) que nos hace falta dentro de los miles disponibles. Esta tarea es muy laboriosa debido al gran volumen de datos disponibles y muy importante al mismo tiempo ya que una mala elección hará que el proyecto fracase.
- Estudiar la fuente de datos (Dateset), y los distintos formatos en los que esta fuente esta disponible. Además de si es posible tecnológicamente su explotación.
- Diseñar el repositorio de datos (Data Mart) en el que iremos guardando los datos una vez estos han sido tratados adecuadamente definiendo las tablas que lo componen (hecho, dimensiones, etc.).
- Diseñar los procesos ETL que tendrán la funcionalidad de extraer los datos de la fuente, transformarlos y luego cargarlos en el Data Mart.
- Llegado a este punto se hace necesario la elección y el uso de las herramientas oportunas para automatizar las ejecuciones de los procesos ETL anteriormente diseñados.

Una vez diseñado y cargado el Data Mart podemos proceder a explotarlo con la herramienta OLAP oportuna y que mejor se adapte a nuestras necesidades, sacando la información que nos será útil de cara a la de toma de decisiones sobre cómo debemos distribuir nuestra flota basándonos en el comportamiento de las incidencias de tráfico.

4.2 Escenario

Debido a la innovación del proyecto y antes de proceder a implementarlo directamente dedicando recursos que no sabemos si después seremos capaces de rentabilizar habrá antes que realizar una prueba de concepto (POC, Proof Of Concept) y que consiste en desarrollar el proyecto con recursos mínimos (sin compra de servidores, ni licencias, ni equipos dedicados…), para demostrar a los jefes de negocio (Business Analist) la viabilidad de esta solución tecnológicamente.

Por lo cual al tratarse de un POC concentraremos nuestro esfuerzo en localizar y tratar los datos de la comunidad de Euskadi para que una vez la viabilidad está demostrada y el proyecto aceptado, pues se podrá adoptar la solución y escalarla a nivel nacional a todas las comunidades de España y también en un futuro a nivel internacional a los diferentes países en los que nuestra empresa de seguros está presente (Francia, Italia, Reino unido, Estados unidos).

Además, partimos de que ya disponemos de un Data Warehouse desarrollado con sus correspondientes DataMarts y que tiene la funcionalidad de historificar la información del sistema operativo que usa la empresa "Guidewire" (a estos datos no vamos a poder acceder debido a las estrictas normas de confidencialidad de la empresa).

Entonces en este proyecto centraremos nuestro esfuerzo en localizar fuente de información (Open Data) que nos ofrezca datos sobre las incidencias de tráfico en la comunidad de Euskadi con el objetivo de tratar estos datos, guardarlos y historificarlos construyendo un nuevo Data Mart para luego poder analizarlo y dar respuestas a las preguntas que nos plantea el negocio ofreciéndole las herramientas oportunas que les puedan orientar en toma de las decisiones correctas.

4.3 Vida útil del Proyecto (B.D.L)

The *Business Dimensional Lifecycle* (B.D.L) o ciclo de vida dimensional representa las etapas necesarias que hay que llevar a cabo a la hora de desarrollar una solución DWH / Data Mart.

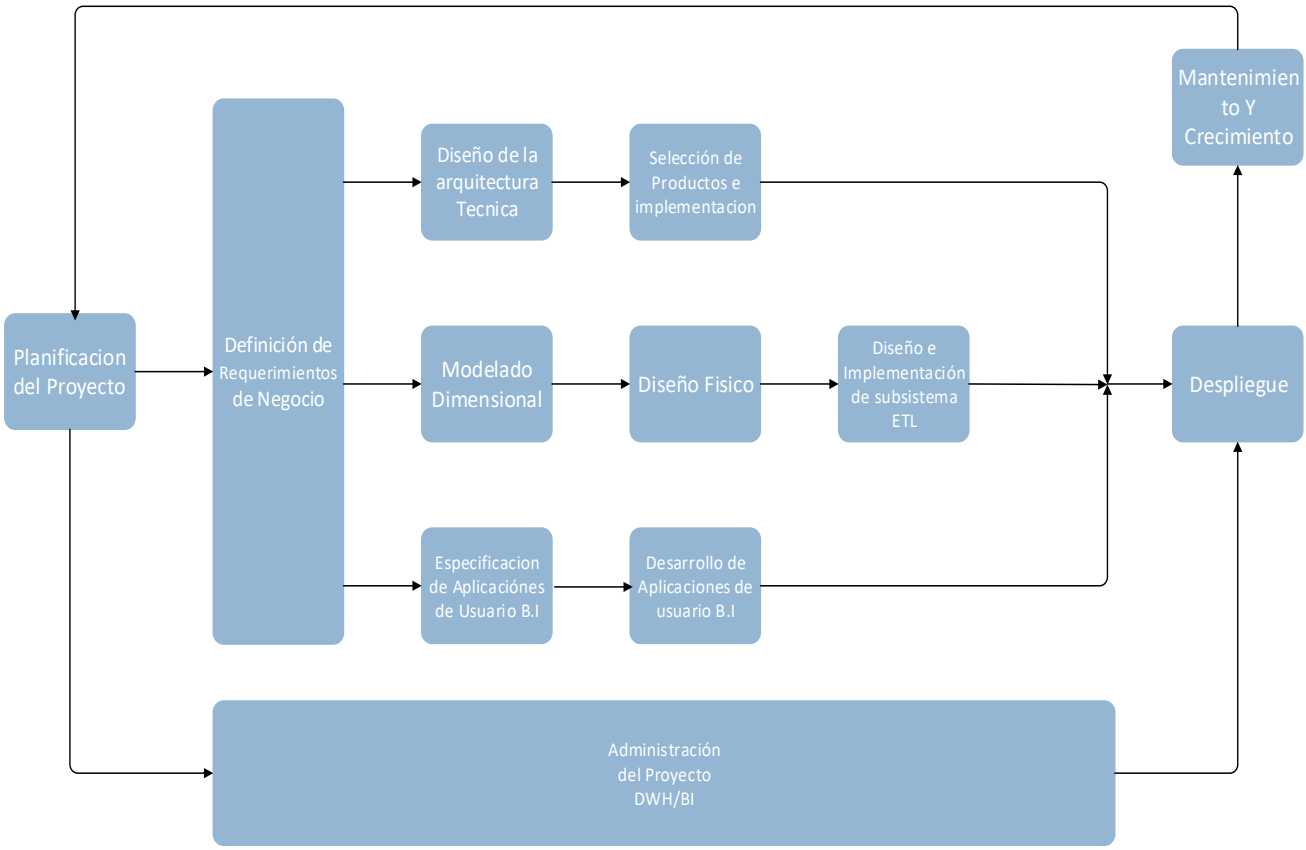


Figura 21. Ciclo de Vida Dimensional (B.D.L, Business Dimensional Lifecycle).

En la figura 20 podemos apreciar la aproximación global de todas las etapas del proyecto y que consisten en:

- Planificación del Proyecto.

Significa hacer una planificación del proyecto definiendo su alcance y las correspondientes tareas y objetivos que lo componen.

- Definición de requerimientos de negocio

Se trata de saber la necesidad del negocio para poder introducirla al proyecto e implementarla. Esta etapa es la más importante y complicada ya que requiere experiencia o una formación aparte en el área del negocio.

De esta etapa depende una gran parte del éxito del proyecto además de que constituye el punto de partida de las tres ramas paralelas que son Tecnología, Datos e Interfaz de usuario.

- Modelado dimensional

Se empieza construyendo una matriz que representa los procesos de negocio claves y su dimensionalidad y a partir de eso un modelo dimensional debe estar desarrollado. Este modelo identifica la granularidad de las tablas de hechos y las dimensiones asociadas.

- Diseño físico

Define las estructuras físicas necesarias para la implementación la base de datos lógica. Este proceso se inicia mediante la determinación de reglas de nomenclatura y las particiones y luego configurando el entorno de base de datos.

- Diseño e implementación de subsistema ETL

Se trata de diseñar los procesos de Extracción, Transformación y Carga de datos.

- Diseño de arquitectura técnica

Hay tres factores que deben tomarse en consideración y son las necesidades del negocio, el entorno técnico actual existente y por ultimo las principales técnicas estratégicas futuras previstas.

- Selección de productos e implementación

Se hace una evaluación de componentes específicos, tales como la plataforma de hardware, el SGDB y las herramientas de preparación y acceso a los datos. Una vez estos componentes evaluados y seleccionados, se procede a su instalación.

- Especificación de aplicación de usuario B.I

Se definen las especificaciones exigidas a la aplicación B.I según los roles de los usuarios finales.

- Desarrollo de aplicación de usuario B.I

Se trata de construcción de tableros, reportes y aplicaciones necesarias para explotar el Data Warehouse.

- Despliegue

Es el punto de convergencia de datos, la tecnología y la aplicación de usuario.

- Mantenimiento y crecimiento

El Data Warehouse es una base de datos que está siempre bajo nuevos desarrollos y que acompaña a la evolución de la empresa por lo cual siempre se van a necesitar incluir nuevas dimensiones, hechos, etc. Por lo cual un servicio de mantenimiento es necesario.

4.4 Planificación del proyecto

Antes de empezar la realización de cualquier proyecto una buena planificación definirá una gran parte del éxito de este. La planificación del proyecto consiste esencialmente en organizar las tareas que van a permitir alcanzar los objetivos deseados. Por lo cual vamos primero a dividir el proyecto en varios sub objetivos asignando a cada uno un conjunto de tareas a realizar y después vamos a estimar la duración que ocupará cada tarea para su realización.

El diagrama de Gantt es el mejor adaptado para estructurar nuestras tareas organizándolas de forma cronológica de modo que las tareas y los tiempos asignados a cada una, están presentados en el diagrama de Gantt como se puede ver en la siguiente figura.

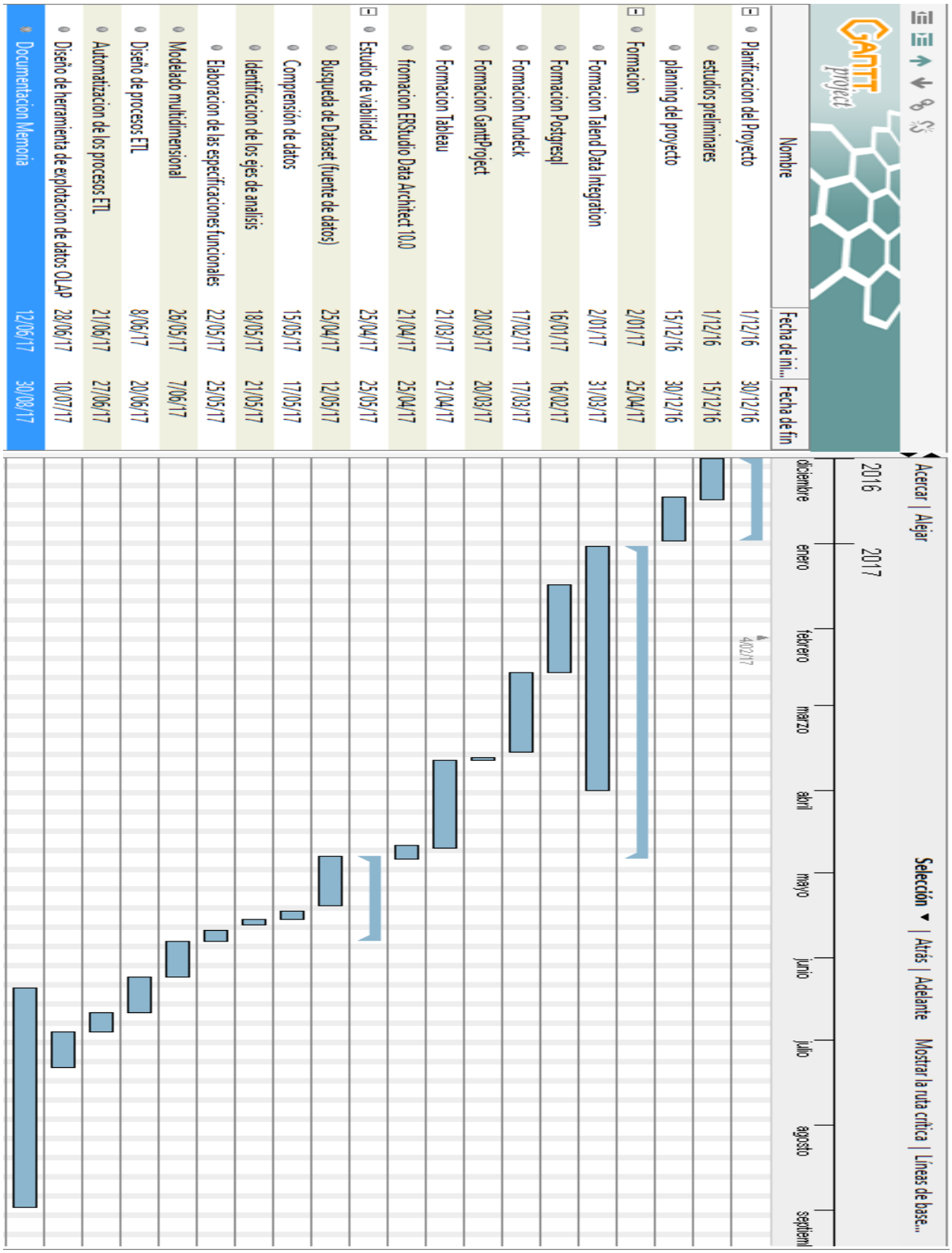


Figura 22. Diagrama de Gantt que describe la planificación que se ha seguido en el proyecto

5 CASO PRÁCTICO: DISEÑO E IMPLEMENTACIÓN

"La forma de empezar es dejar de hablar y empezar"

-Walt Disney-

A Continuación en este capítulo procederemos a desarrollar nuestra solución Business Intelligence presentando primero nuestra fuente de datos Open Data y sus características y analizando los datos que nos ofrece para luego pasar a diseñar nuestro modelo conceptual de base de datos (Data Mart) de incidencias de tráfico. Después nos dedicaremos a programar los procesos ETL necesarios que realimentarán nuestra Data Mart y automatizarlos con el planificador de tareas anteriormente presentado Rundeck. Una vez construido el Data Mart procederemos a explotar los datos y a dar respuestas a las preguntas planteadas.

5.1 Fuente de datos Open Data

Nuestro primer objetivo en este Proyecto es buscar una fuente de datos Open Data que cumpla los siguientes requisitos

- Ofrezca datos de incidencias de tráfico en la comunidad autónoma de Euskadi.
- Base de datos abierta (de acceso libre).
- Que se actualice de forma periódica manteniendo los datos frescos.
- Que los datos estén disponibles en formatos que permitan su explotación tecnológica.

Después de varios días de búsqueda intensiva entre los miles de catálogos disponibles y bases de datos hemos podido encontrar una base de datos que cumple los requisitos antes mencionados y que estudiaremos a continuación.

5.1.1 Información de la base de datos: Incidencias de tráfico de la comunidad de Euskadi

Nombre de base de datos.	Incidencias de tráfico de la comunidad de Euskadi.
Publicador.	Comunidad Autónoma del País Vasco.
Nivel de administración.	Administración Autonómica.
Licencia.	https://creativecommons.org/licenses/by/4.0/deed.es_ES
Catálogo.	http://datos.gob.es/es/catalogo/a16003011-incidencias-del-trafico-en-tiempo-real-en-euskadi
Formatos Disponibles.	Zip, XML.
Descripción.	Facilita los detalles de las incidencias en las carreteras de la Comunidad Autónoma de Euskadi de forma actualizada y en tiempo real.
Tiempo de actualización.	Cada 1h.
Idioma.	Español, euskera.
Cobertura geográfica.	País Vasco.
Fecha de creación.	19/07/2010.
Acceso directo.	https://www.trafikoa.eus/servicios/IncidenciasTDT/IncidenciasTrafikoTDTGeo
Uso.	Público.

Tabla 7. Información de la base de datos Open Data a explotar

Una vez encontrada la fuente se dedicará un tiempo a estudiar los datos que esta ofrece, para saber cómo se comportan y prevenir los posibles casos que pueden dar fallos a nivel de programación de los procesos o a nivel de incoherencia de datos.

5.1.2 Estructura de la información proporcionada por el servicio

La estructura del fichero XML proporcionado es la siguiente:

- **Tipo**
  - Meteorológica
  - Accidente
  - Retención
  - Seguridad vial
  - Otras incidencias
  - Puertos de montaña
  - Vialidad invernal tramos
  - Pruebas deportivas

- **Autonomía**
  - Euskadi

- **Provincia**
  - Alava-Araba
  - Bizkaia
  - Gipuzkoa

- **Matrícula**
  - BI
  - VI
  - SS
- **Causa**
  - En caso de ser de Tipo **Metereológica**
    - Agua
    - Viento
    - Nieve / Hielo
    - Niebla
  - En caso de ser de Tipo **Accidente**
    - Alcance
    - Atropello
    - Salida
    - Tijera camión
    - Vuelco
  - En caso de ser de Tipo **Retención**
    - Fiestas
    - Prueba deportiva
  - En caso de ser de Tipo **Seguridad Vial**
    - Aceite
    - Avería
    - Caída objetos
    - Desprendimiento
    - Gasoil
    - Incendio
    - Socavón
  - En caso de ser de Tipo **Puertos de montaña**
    - Agua nieve
    - Hielo
    - Nevando
    - Niebla
    - Nieve
    - Nieve / Hielo

- Desconocida
- Obras
- Otros

o En caso de ser de Tipo **Vialidad invernal tramos**

- Agua nieve
- Hielo
- Nevando
- Niebla
- Nieve
- Nieve / Hielo
- Desconocida
- Obras
- Otros

o En caso de ser de Tipo **Obras**

- Obra
- Otra actividad

o En caso de ser Tipo Pruebas deportivas

- Automovilismo
- Ciclismo
- Ciclocross

o Cross

- Maratón
- Biatlón
- Triatlón
- Pentatlón
- Motociclismo
- MotoCross
- Marcha ciclista
- Mixta
- Atletismo

- **Población**
- **Fecha hora inicio**
- **Nivel**
  - o Verde (Normal)
  - o Blanco (Fluido)
  - o Amarillo (Lento)
  - o Rojo (Muy lento)
  - o Negro (Parado)
  - o En el caso de Puertos de montaña se concatenan los valores del estado del puerto para Turismo **(T),** Camión **(C)** y Articulados **(A)**. Estos valores del estado son:
    - Cerrado
    - Abierto
    - Cadenas
    - Precaución
- **Carretera**
- **Punto Kilométrico inicial**
- **Punto kilométrico final**
- **Sentido**
- **Nombre**
- **Longitud**
- **Latitud**

5.1.3 Ejemplo

```xml
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<raiz>
    <incidenciaGeolocalizada>
        <tipo>Accidente</tipo>
        <autonomia>Euskadi</autonomia>
        <provincia>GIPUZKOA</provincia>
        <matricula>SS</matricula>
        <causa>Alcance</causa>
        <poblacion>Irun</poblacion>
        <fechahora_ini>2017-09-08 20:36:59</fechahora_ini>
        <nivel>Blanco</nivel>
        <carretera>N-121A</carretera>
        <pk_inicial>74</pk_inicial>
        <pk_final>74</pk_final>
        <sentido>BEHOBIA</sentido>
        <longitud>-1.741693</longitud>
        <latitud>43.32182</latitud>
    </incidenciaGeolocalizada>
    <incidenciaGeolocalizada>
        <tipo>Obras</tipo>
        <autonomia>Euskadi</autonomia>
        <provincia>GIPUZKOA</provincia>
        <matricula>SS</matricula>
        <causa>Obras</causa>
        <poblacion>Aretxabaleta</poblacion>
        <fechahora_ini>2017-09-08 16:25:49</fechahora_ini>
        <nivel>Blanco</nivel>
        <carretera>AP-1</carretera>
        <pk_inicial>126</pk_inicial>
        <pk_final>124</pk_final>
        <sentido>Madrid</sentido>
        <longitud>-2.499656</longitud>
        <latitud>43.04597</latitud>
    </incidenciaGeolocalizada>
    <incidenciaGeolocalizada>
```

Figura 23. Ejemplo de fichero XML.

5.2 Modelado del Datamart

En este apartado modelaremos nuestra nueva Data Mart de incidencias de tráfico de la comunidad de Euskadi. Partiremos desde el fichero XML que descargaremos cada 1h de la base de datos y de la información vista anteriormente. Para crear nuestro modelo estrella (tablas de dimensiones y hecho). Crearemos también dos esquemas que las llamaremos AC y es donde cargaremos los datos a lo bruto y el esquema DC que es donde guardaremos los datos consolidados para ser explotados.

5.2.1 Tablas de Dimensiones

- d_autonomia

id_d_autonomia	Índice interno de la tabla (PK).
id_d_srce_syst	Origen de la fuente de donde se cargan los datos.
date_load	Fecha de carga del dato.
desc_autonomia	Descripción de la autonomía.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 8. Dimensión d_autonomia.

- d_causa

id_d_causa	Índice interno de la tabla (PK).
id_d_srce_syst	Origen de la fuente de donde se cargan los datos.
date_load	Fecha de carga del dato.
desc_causa	Descripción de la causa de incidencia.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 9. Dimensión d_causa.

- d_matricula

id_ d_matricula	Índice interno de la tabla (PK).
id_d_srce_syst	Origen de la fuente de donde se cargan los datos.
date_load	Fecha de carga del dato.
desc_matricula	Descripción de la matrícula.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 10. Dimensión d_matricula.

- d_nivel

id_d_nivel	Índice interno de la tabla (PK).
id_d_srce_syst	Origen de la fuente de donde se cargan los datos.
date_load	Fecha de carga del dato.
desc_nivel	Descripción de nivel.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 11. Dimensión d_nivel.

- d_provincia

id_d_provincia	Índice interno de la tabla (PK).
id_d_srce_syst	Origen de la fuente de donde se cargan los datos.
date_load	Fecha de carga del dato.
desc_provincia	Descripción de la provincia.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 12. Dimensión d_provincia.

- d_srce_syst

id_ d_srce_syst	Índice interno de la tabla (PK).
date_load	Fecha de carga del dato.
desc_srce_syst	Descripción del sistema fuente de datos.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 13. Dimensión d_srce_syst.

- d_tipo

id_d_tipo	Índice interno de la tabla (PK).
id_d_srce_syst	Origen de la fuente de donde se cargan los datos.
date_load	Fecha de carga del dato.
desc_tipo	Descripción del tipo de incidencia.
swit_reti	Indica si el registro de la dimensión ha sido retirado.

Tabla 14. Dimensión d_tipo.

5.2.2 Tabla de hechos

- H_INCI

id_h_inci	Índice interno de la tabla (PK).
Id_d_srce_syst	FK a la tabla de dimensión d_srce_syst.
id_d_tipo	FK a la tabla de dimensión d_tipo.
id_d_autonomia	FK a la tabla de dimensión d_autonomia.
id_d_provincia	FK a la tabla de dimensión d_provincia.
id_d_matricula	FK a la tabla de dimensión d_matricula.
id_d_causa	FK a la tabla de dimensión d_causa.
poblacion	Población del incidente.
fechahora_ini	Fecha y hora de comienzo del incidente.
id_d_nivel	FK a la tabla de dimensión d_nivel.
carretera	Carretera donde tuvo lugar el incidente.
pk_inicial	Punto kilométrico inicial del incidente.
pk_final	Punto kilométrico final del incidente.
sentido	Sentido del incidente.
longitud	Longitud del punto del incidente.
latitud	Latitud del punto del incidente.
date_load	Fecha de carga del dato en el DWH.
date_cons	Fecha de consolidación del dato.
nombre	Nombre de incidente.

Tabla 15. De hechos h_inci.

5.2.3 Modelo en estrella

Nuestro modelo en estrella queda como muestra la siguiente figura:

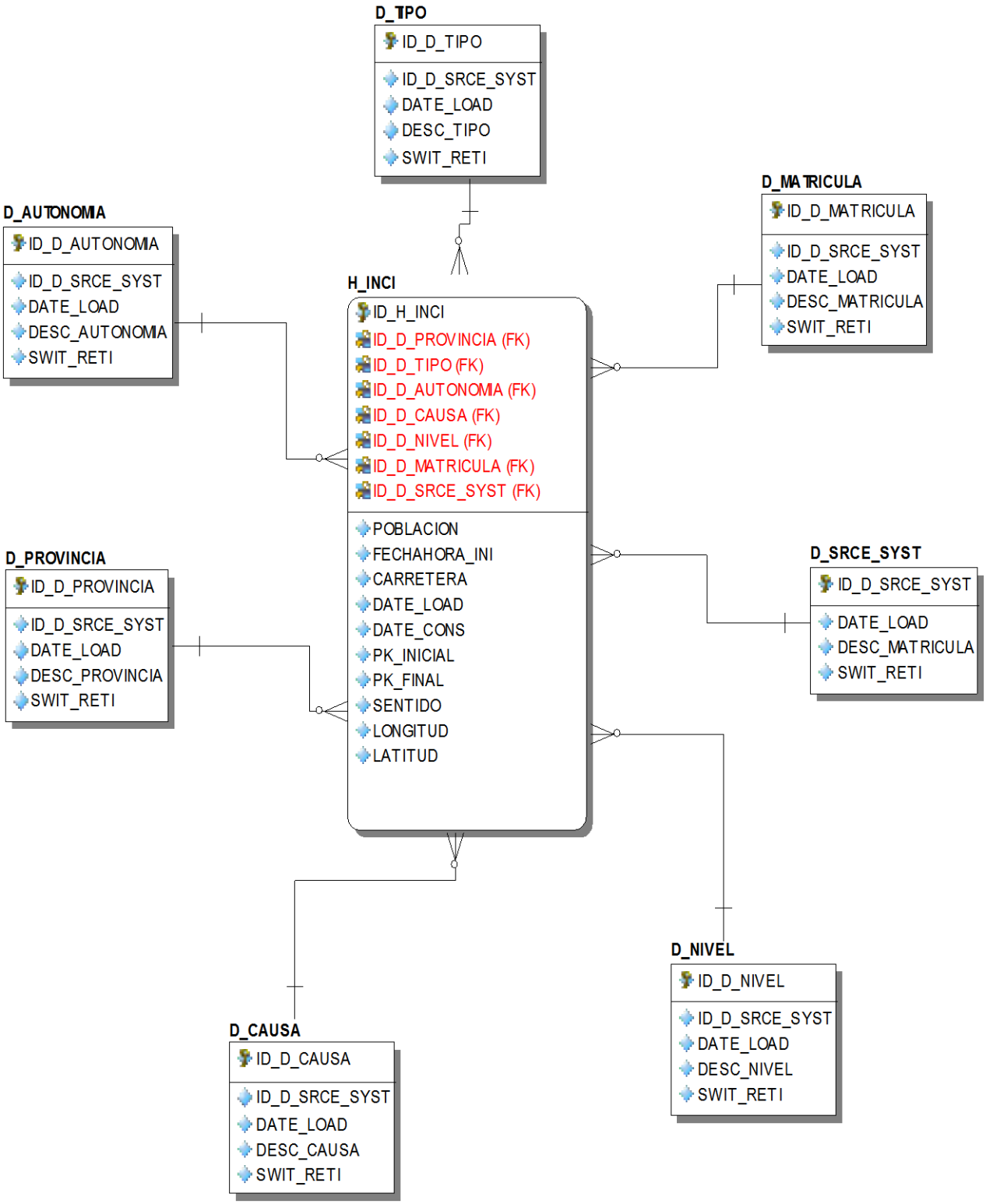


Figura 24. Modelo en estrella desarrollado.

5.3 Concepción de la base de datos PostgreSQL

Después de instalar la herramienta pgAdmin3 procedemos a la creación de nuestra base de datos relacional de incidencias de tráfico de la comunidad de Euskadi relacional.

- Creación de base de datos DWH con la siguiente configuración.

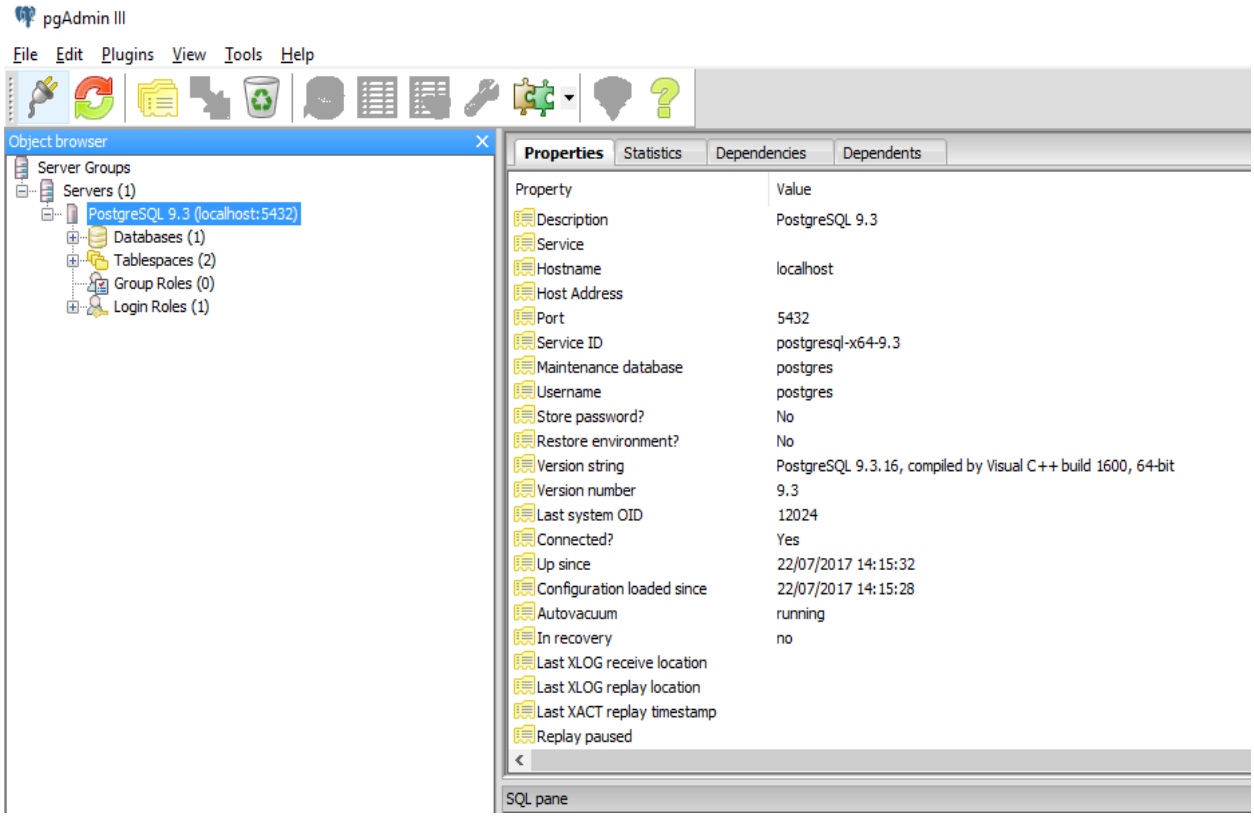


Figura 25. Configuración de la base de datos DWH.

- Después de la creación de base de datos creamos los esquemas AC, DC y TEMP.

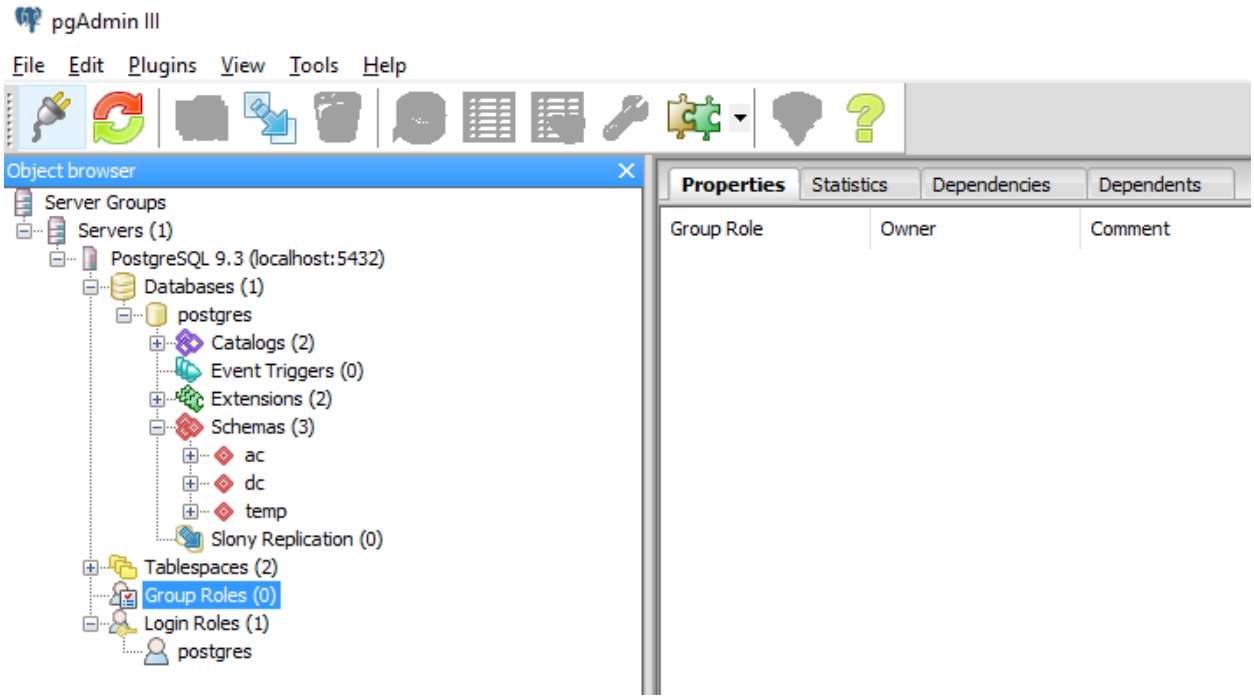


Figura 26. Creación de los esquemas AC, DC y TEMP.

Usaremos los tres esquemas para:

- **AC:** Este esquema se usará para cargar los datos desde los ficheros XML facilitados por la base de datos de servicios de incidencias de tráfico de la comunidad autónoma de Euskadi.
- **DC:** Este esquema guardará los datos cargados en el AC después de que estos estén tratados y transformados haciendo las comprobaciones oportunas como el control de duplicidad. Este esquema guardará los datos listos para ser explotados por el usuario final (Reporting) o por las herramientas OLAP.
- **TEMP:** en este esquema guardaremos las tablas temporales que necesitaremos en un futuro para resolver posibles incidencias o algo similar, las tablas de este esquema serán borradas con cierta frecuencia para ahorrar espacio en el disco duro de nuestra base de datos.

Una vez creados los esquemas procederemos a crear nuestras tablas de dimensiones y hecho anteriormente vistas en los correspondientes esquemas.

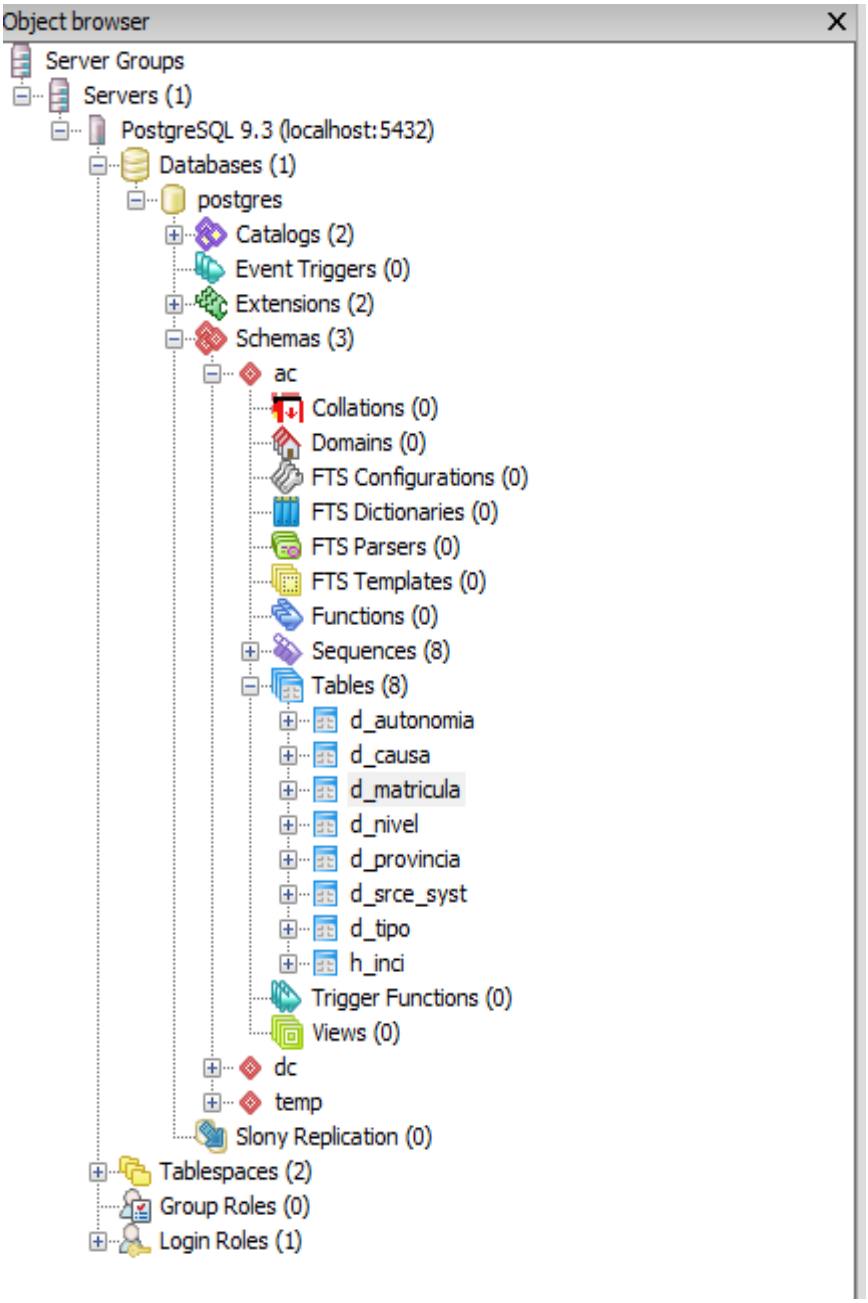


Figura 27. Tablas creadas en el esquema AC.

5.4 Diseño de procesos ETL

En este apartado explicaremos los procesos (Jobs) ETL diseñados para la extracción, carga y transformación de los datos a partir de la fuente anteriormente vista.

5.4.1 LoadXmlFile job

5.4.1.1 Funcionamiento

- Encargado de la extracción de los ficheros XML de base de datos Open Data figura 28.
- Este proceso ETL se encarga de conectarse a la base de datos anteriormente descrita mediante web service para extraer los ficheros XML correspondientes y guardarlos en un directorio local firgura 29.
- Los ficheros XML extraídos de la base de datos se irán extrayendo con frecuencia de 1extracción / 1hora por lo cual para que no haya perdida de información se guardarán en local con el nombre = "**file_YYYY_DD_HH_MM_SS.xml**)" siendo Y: year (años) D: Day (Dia), H:Hour (hora), M:minute (minutos), S:second(segundos) figura 30.

5.4.1.2 Componentes, funciones y variables de Talend usados

- Componente **tPrejob** (lanza la ejecución del job).
- Componente **tPostjob** (lanza la ejecución de un post job).
- Componente **tjava** (permite incrustar código java personalizado).
- Componente **tFileFetch (**recupera un fichero a partir de un protocolo en este caso el protocolo **Https)**.
- Componente **tFixedFlowInput (**permite gestionar datos fijos a partir de variabes internas).
- Componente **tBufferOutput** (mete en buffer datos para que estos puedan ser recuperados más tarde).
- Variable de contexto **file_name.**
- Función **TalendDate.formatDate().**
- Función **TalendDate.getCurrentDate().**

5.4.1.3 Diseño

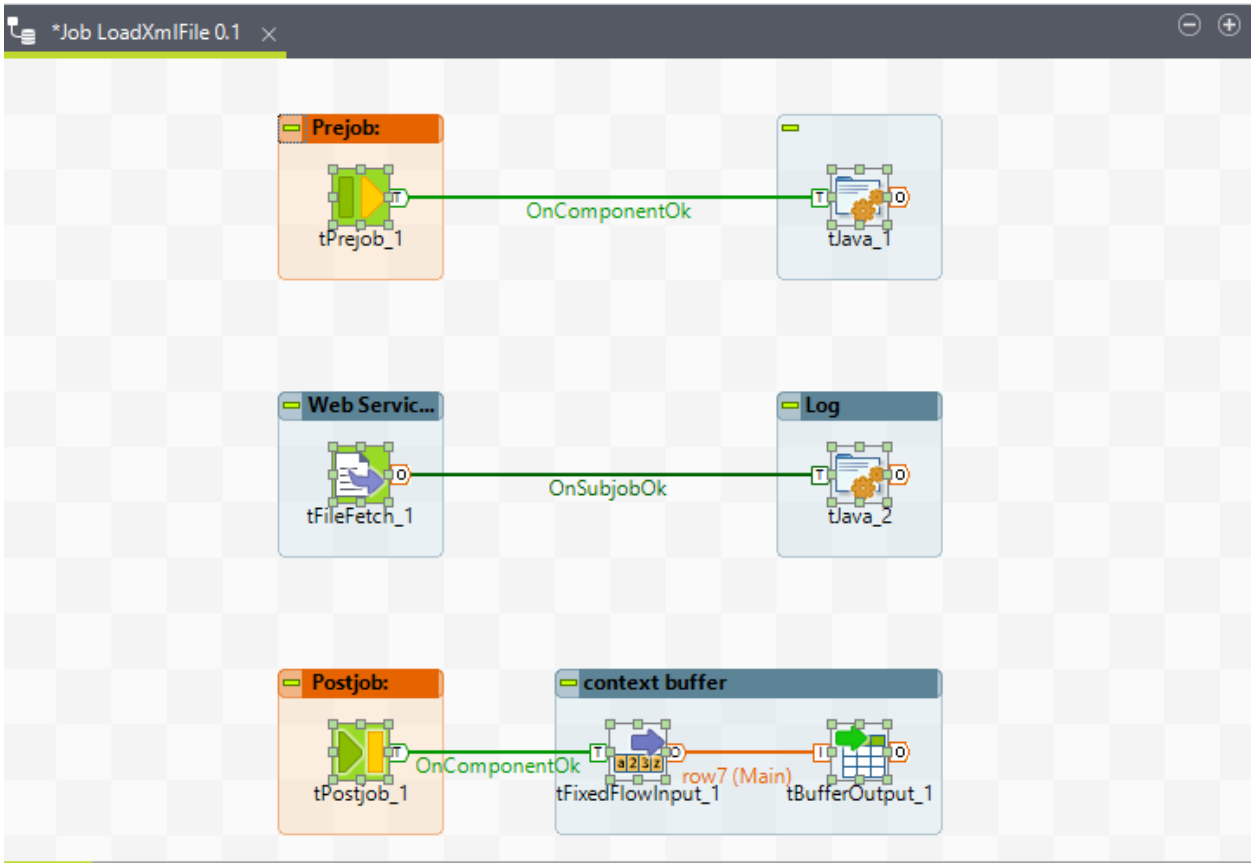


Figura 28. LoadXmlFile job diseñado en Talend.

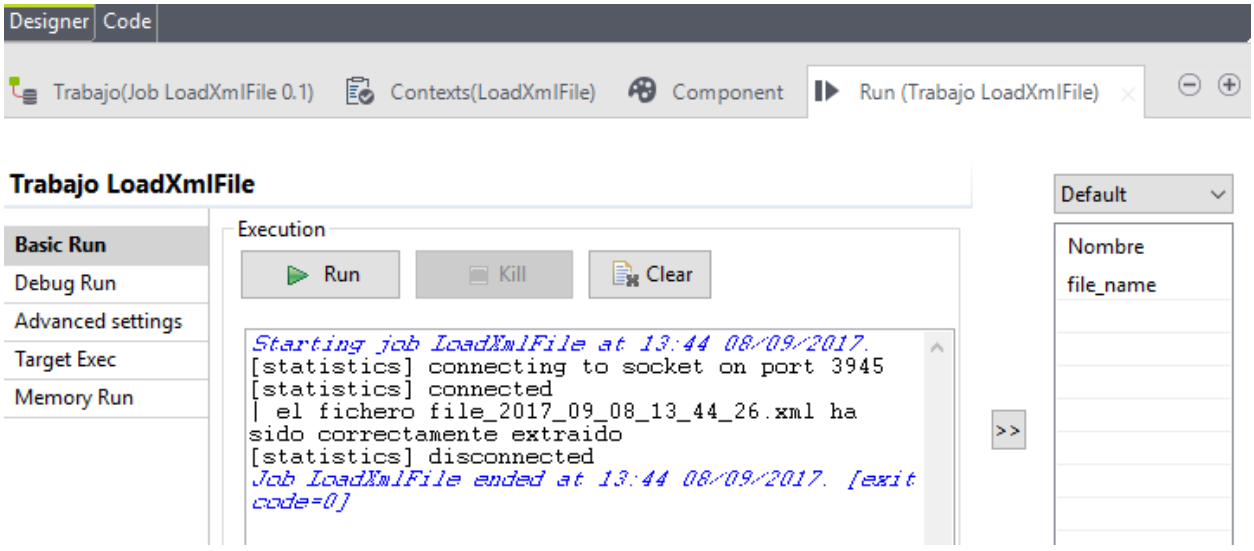


Figura 29. Resultado de ejecución del Job LoadXmlFile en local.

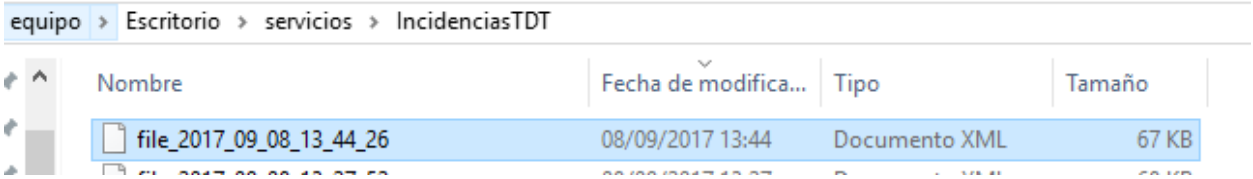


Figura 30. Fichero XML generado al ejecutar el job LoadXmlFile.

5.4.2 EUSKA_FACT_AC job

5.4.2.1 Funcionamiento

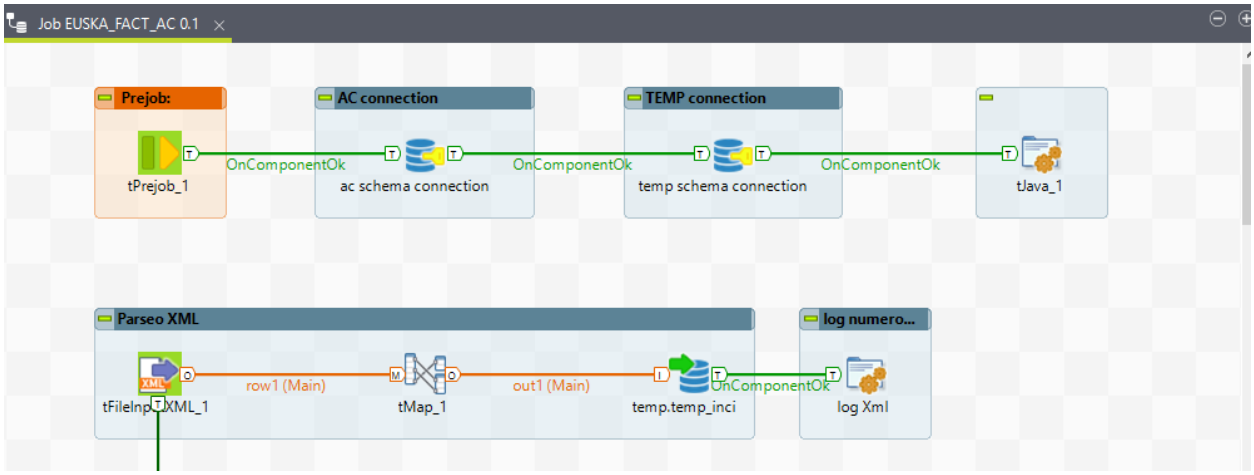
- El Job primero establece las conexiones con nuestra base de datos PostgreSQL tanto para el esquema AC como para el esquema TEMP figura 31.
- Luego se encarga de extraer los datos de los documentos XML (cargados anteriormente con el Job LoadXmlFile) parsearlos, mapearlos y cargarlos en nuestra tabla temp_inci del esquema TEMP en nuestra base de datos PostgreSQL tabla 16.
- Luego procede a transformar los datos guardados en la tabla temp_inci en el esquema TEMP mapeando las dimensiones correspondientes de los datos para finalmente guardarlos en la tabla h_inci del esquema AC tabla 17.
- Y por último hace el commit de todos los cambios aportados a la base de datos dejando las conexiones cerradas e imprimiendo un log con el resumen de numero de registros procesados.

5.4.2.2 Componentes, funciones y variables de Talend usados

- Componente **tPrejob.**
- Componente **tPostgreSqlConection** (establece la conexión a la base de datos PostgreSQL).
- Componente **tjava.**
- Componente **tFileInputXM** (permite recuprar un fichero XML y parsearlo).
- Componente **tPostgreSqlInput** (permite recuperar datos con consultas SQL de la base PostgreSQL).
- Componente **tPostgreSqlOutput** (permite guardar datos en la base de datos PostgreSQL).
- Componente **tPostgreSqlcommit** (realiza el commit de todas las transacciones y cierra la conexión).
- Componente **tMap** (dirige y transforma los datos a partir de varias fuentes de datos).
- Variable de contexto **file_name.**
- Variable de contexto **date_load.**
- Función **TalendDate.formatDate().**
- Función **TalendDate.getCurrentDate().**

5.4.2.3 Diseño

Ejecutado el Job para procesar los datos del XML extraído anteriormente con el Job LoadXmlFile obtendremos el siguiente resultado de ejecución.



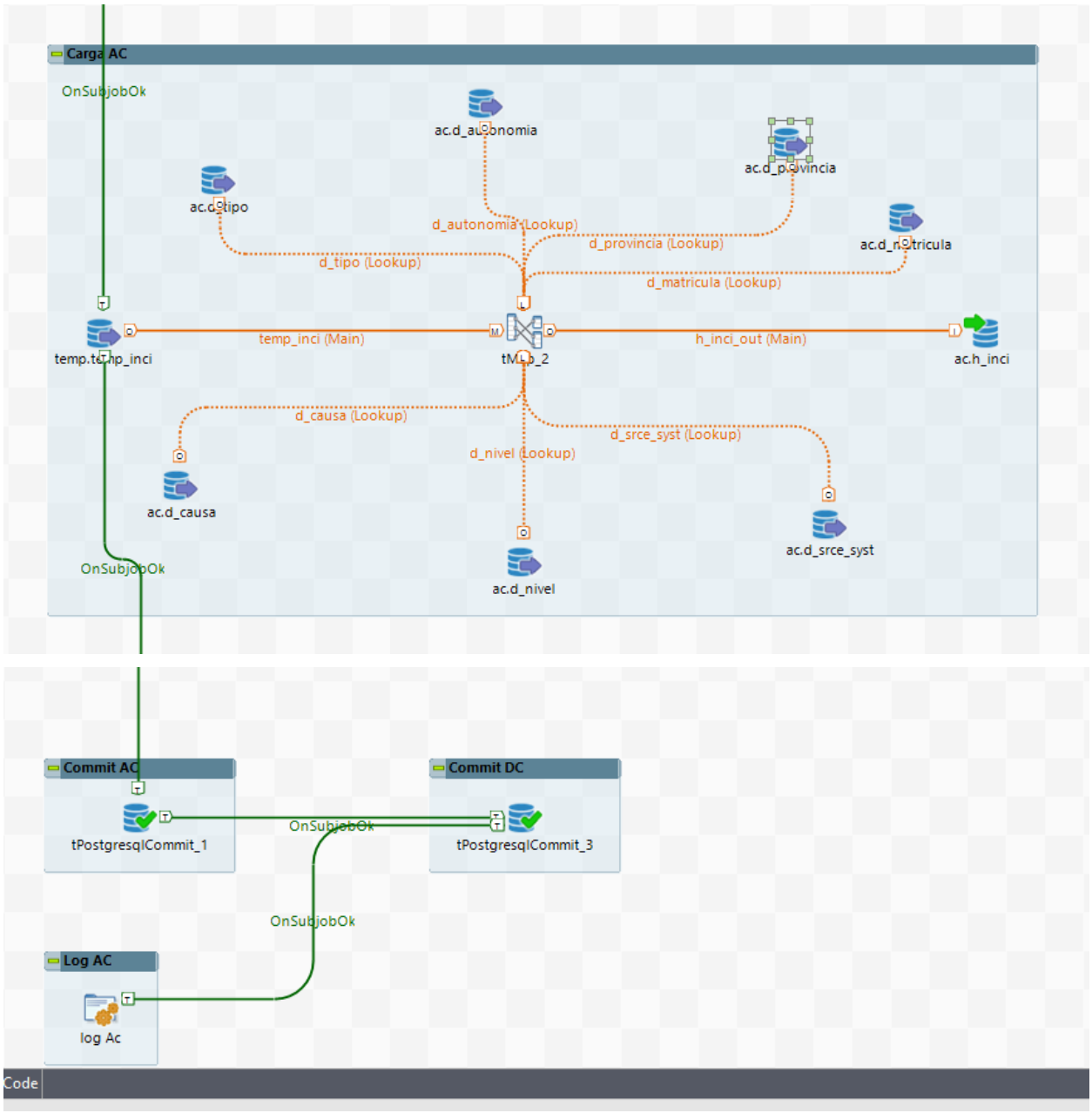


Figura 31. EUSKA_FACT_AC job diseñado en Talend.

Como se puede apreciar en las siguientes figuras el resultado de haber ejecutado el job en las tablas del esquema AC.

SQL Editor | Graphical Query Builder

Previous queries

```
select * from temp.temp_inci ;
```

Output pane

Data Output | Explain | Messages | History

	tipo character varying	autonomia character varying	provincia character varying	matricula character varying	causa character varying	poblacion character varying	fechahora_i character va
1	Seguridad vial	Euskadi	ARABA	VI	AverÃ-a	Zigoitia	2017-09-08
2	Seguridad vial	Euskadi	BIZKAIA	BI	AverÃ-a	Orozko	2017-09-08
3	Obras	Euskadi	GIPUZKOA	SS	Obras	Eibar	2017-09-08
4	Obras	Euskadi	BIZKAIA	BI	Obras	Bilbao	2017-09-08
5	Obras	Euskadi	ARABA	VI	Obras	Zambrana	2017-09-08

Tabla 16. Resultado de ejecutar el Job EUSKA_FACT_AC, tabla temp_inci.

Previous queries

```
select * from ac.h_inci
```

Output pane

Data Output | Explain | Messages | History

	id_h_inci integer	id_d_tipo integer	id_d_autonomia integer	id_d_provincia integer	id_d_matricula integer	id_d_causa integer	poblacion character varying(60)	fechahora_ir character va
1	5706	0	1	0	3	24	Eibar	2017-09-08
2	5707	5	1	0	1	25	Gordexola	2017-09-08
3	5708	2	1	0	3	5	Tolosa	2017-09-08
4	5709	4	1	0	2	0	Zigoitia	2017-09-08

Tabla 17. Resultado de ejecutar el Job EUSKA_FACT_AC, tabla h_inci.

5.4.3 EUSKA_FACT_DC Job

5.4.3.1 Funcionamiento

- El Job después de establecer las conexiones con la base de datos se encarga de hacer un control de duplicidad de los datos que se han cargado en el AC así evitamos redundancia en los datos además se encarga de borrar los datos duplicados en el AC una vez detectados figura 32.
- El Job también se encarga de mapear los datos cargados del AC con las dimensiones correspondientes en el DC figura 32.
- Después realiza el commit e imprime un resumen de los registros insertados y duplicados.

5.4.3.2 Componentes, funciones y variables de Talend usados.

- Componente **tPrejob.**
- Componente **tPostgreSqlConection.**
- Componente **tjava.**
- Componente **tPostgreSqlInput.**
- Componente **tPostgreSqlOutput.**
- Componente **tPostgreSqlcommit.**
- Componente **tMap.**
- Variable de contexto **date_load.**
- Varibale **tPostgresqlInput _NB_LINE**, **tPostgresqlOutput_NB_LINE_INSERTED.**
- Funcion **(Integer)globalMap.get().**
- Función **TalendDate.formatDate().**
- Función **TalendDate.getCurrentDate().**

5.4.3.3 Diseño

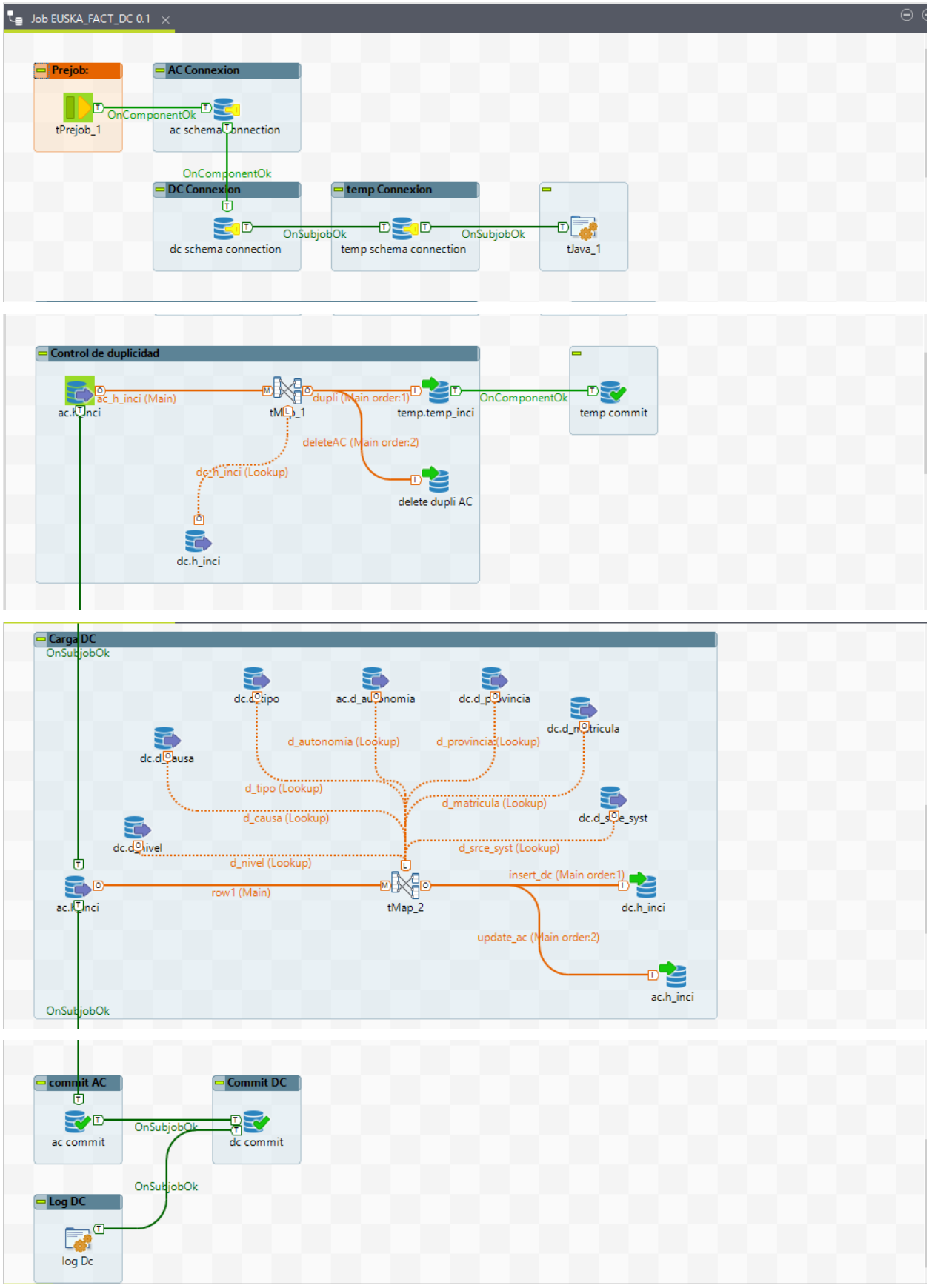
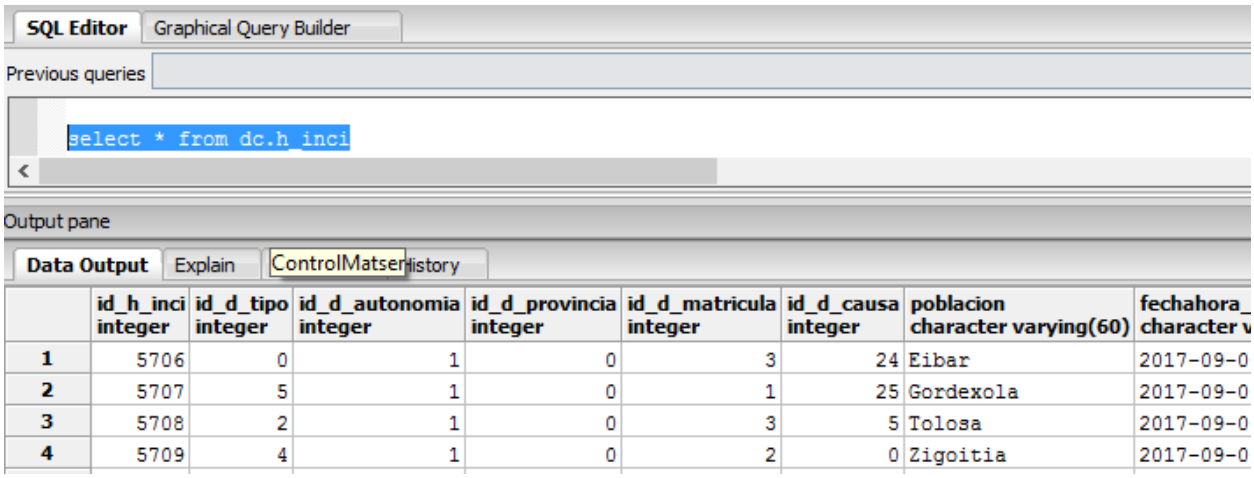


Figura 32. EUSKA_FACT_DC job diseñado en Talend.



	id_h_inci integer	id_d_tipo integer	id_d_autonomia integer	id_d_provincia integer	id_d_matricula integer	id_d_causa integer	poblacion character varying(60)	fechahora_ character v
1	5706	0	1	0	3	24	Eibar	2017-09-0
2	5707	5	1	0	1	25	Gordexola	2017-09-0
3	5708	2	1	0	3	5	Tolosa	2017-09-0
4	5709	4	1	0	2	0	Zigoitia	2017-09-0

Tabla 18. Resultado de ejecutar el Job EUSKA_FACT_DC, tabla h_inci.

5.4.4 ControlMaster Job

5.4.4.1 Funcionamiento

- Este Job es el que se encarga de orquestar el funcionamiento de los otros Job. Es el Job padre y los demás son Jobs hijos figura 33.
- Este Job también se encarga de pasar las variables de contexto a sus Jobs hijos y cargarlos de nuevo con la ayuda del componente **tContextLoad** figura 33.
- Es el job que estará compilado y ejecutado.
- Imprime un log con un resumen de todos los registros procesados en los job hijos figura 34.

5.4.4.2 Componentes, funciones y variables de talend usados

- Componente **tPrejob.**
- Componente **tPostgreSqlConection.**
- Componente **tjava.**
- Componente **tContextLoad** (modifica dinámicamente los valores del contexto activo)
- Componente **LoadXmlFile.**
- Componente **EUSKA_FACT_AC.**
- Componente **EUSKA_FACT_DC.**

5.4.4.3 Diseño

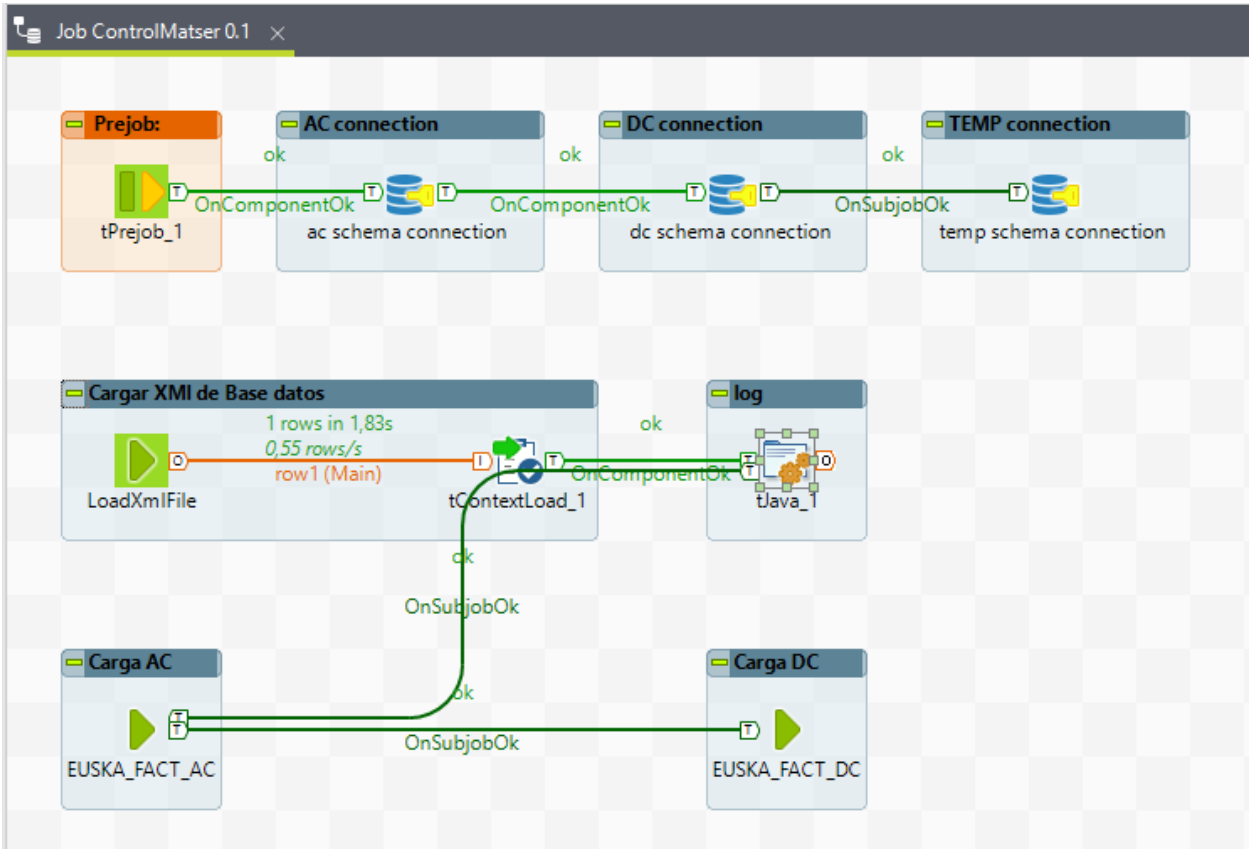


Figura 33. ControlMaster job diseñado en Talend.

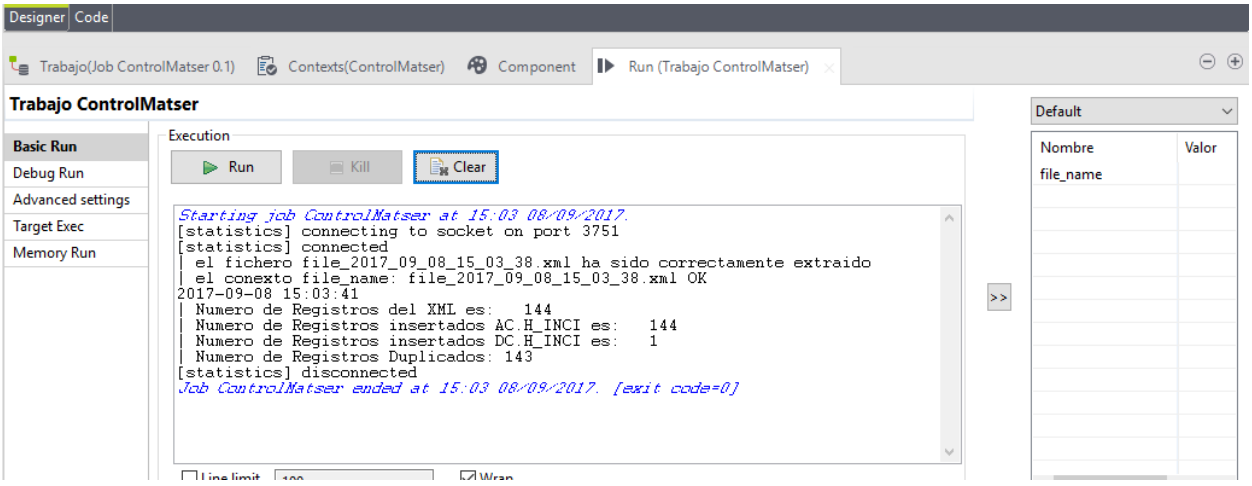


Figura 34. Resultado de ejecutar el Job ControlMatser en local.

5.4.5 EUSKA_PRO

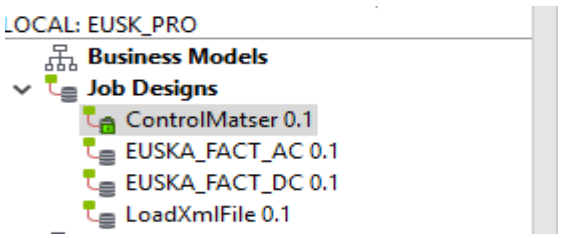


Figura 35. EUSK_PRO Proyecto contenedor de todos los Job anteriormente descritos.

5.5 Automatización de los procesos

Una vez compilado el proyecto necesitaremos una herramienta con la que podremos automatizar la ejecución de los Jobs para que estos se lancen 1vez/hora. De allí el uso de Rundeck

En este apartado veremos cómo crear un proyecto en Rundeck y como configurarlo para que lanze nuestros Jobs anteriormente diseñados. Instalaremos Rundeck en windows10 que es el sistema operativo que estamos usando para desarrollar este proyecto.

5.5.1 Pasos para crear un proyecto en Rundeck

Una vez instalado y configurado y lanzado Rundeck, se accede a la aplicación web que ofrece este último mediante la url: http://localhost:4440.

Una vez en la aplicación web, aparecerá un cuadro de autentificación.

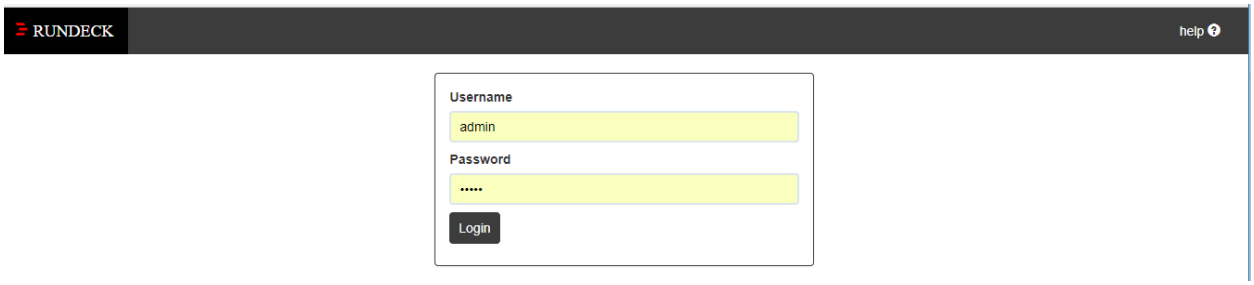


Figura 36. Cuadro de autentificación de Rundeck.

Rundeck permite configurar varios roles dando a cada uno ciertos privilegios de gestión dentro del proyecto. Por defecto y a la hora de instalarlo Rundeck permite la autentificación mediante las siguientes credenciales.

- Username: admin.
- Password: admin.

Una vez dentro procederemos a crear el proyecto EUSK_PRO.

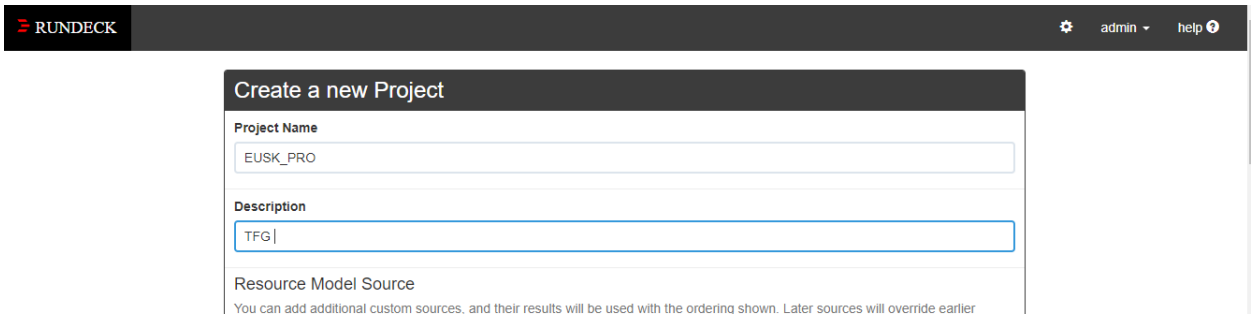


Figura 37. Cuadro de creación de un nuevo proyecto en Rundeck.

Una vez dentro del proyecto EUSK_PRO procederemos a crear el Job que se encargará de lanzar nuestros Jobs compilados.

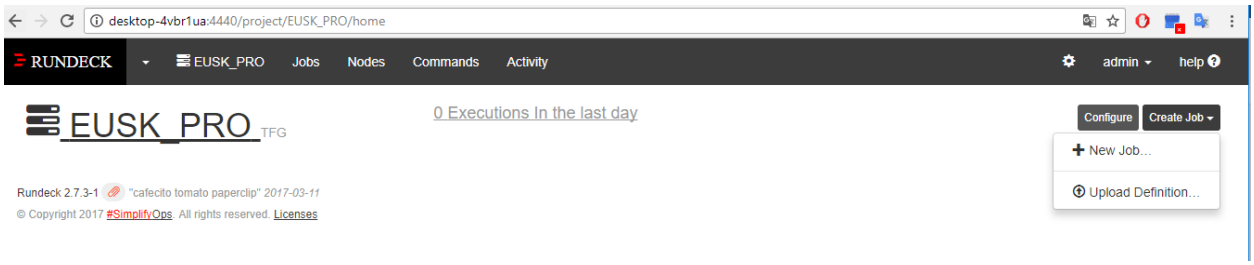


Figura 38. Cuadro para acceder a crear un nuevo Job en Rundeck.

Una vez dentro del cuadro de configuración, primero le pondremos un nombre al Job y lo llamaremos ControlMaster en referencia a Job padre compilado de Talend. En este proyecto solo tendremos un Job, pero en proyectos más grandes suele haber muchos más y si el nombre no hace referencia al compilado que lanza será muy fácil equivocarse y lanzar Job de forma errónea.

Figura 39. Cuadro de configuración de Job en Rundeck 1.

Después elegimos la forma en la que queremos que nuestro compilado sea lanzado, donde será ejecutado (en local o en servidor) y con qué frecuencia (fecha y hora de ejecución).

Figura 40. Cuadro de configuración de script en Rundeck.

- **C:\Users\simo\Desktop\ControlMatser_0.1\ControlMatser\ControlMatser_run.bat** es el directorio donde hemos colocado nuestro script que estará lanzado por Rundeck.
- El Job será ejecutado en local lanzando el comando anterior.
- Configuraremos también la frecuencia de ejecución para que sea cada 1h.

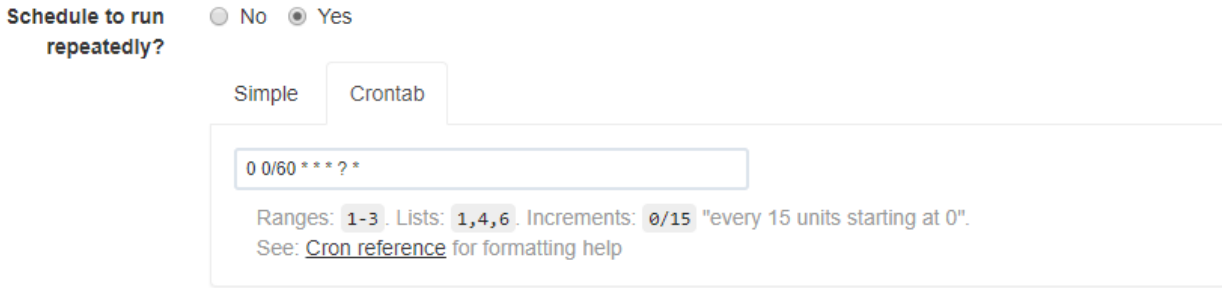


Figura 41. Cuadro de configuración de hora de ejecución de Job ControlMatser en Rundeck.

Rundeck también permite configuración de notificaciones en caso de fallo, Tiemout de ejecución y más opciones

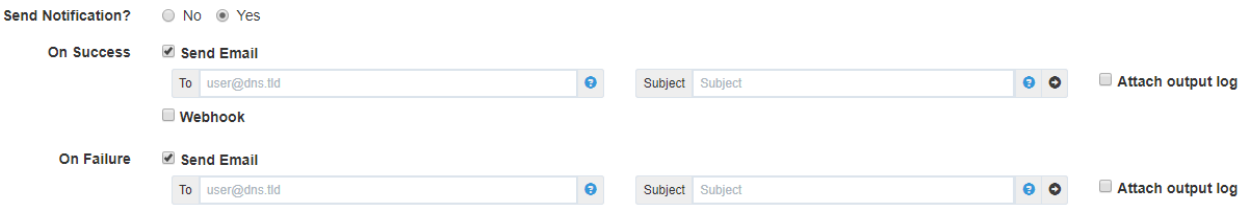


Figura 42. Cuadro de configuración de Rundeck.

Una vez creado el Job y configurado procederemos a lanzarlo manualmente (también podremos esperar a que se ejecute automáticamente en este caso serían dentro de 13minutos como aparece en la figura).

Figura 43. Cuadro de ejecución de Job en Rundeck.

Una vez ejecutado el Job podremos acceder a los logs para ver el resultado de ejecución.

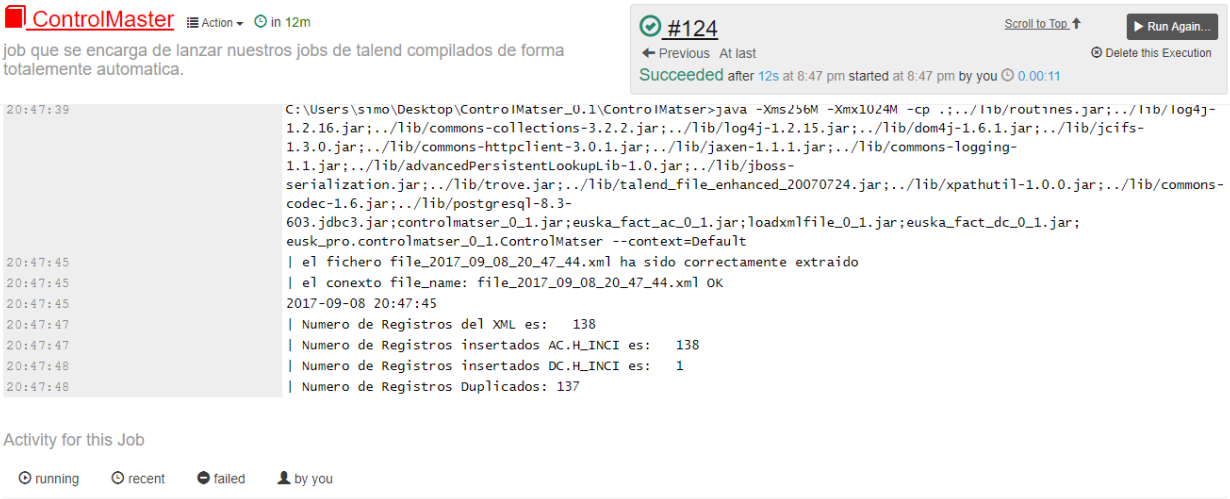


Figura 44. Log de ejecución de Job de ControlMaster en Rundeck.

En la figura 44 podemos ver que el Job se ha ejecutado correctamente, además podemos ver en el log que se ha extraído el fichero XML de la base de datos con éxito y que este XML disponía de 138 líneas a procesar 137 ya estaban cargadas en nuestra base de datos PostgreSQL (tabla dc.h_inci) cuando hacíamos pruebas en Talend y 1 línea es nueva y se ha insertado en dc.h_inci.

Rundeck también ofrece un cuadro con todas las últimaa ejecuciones de los Jobs así se puede rastrear los fallos cuando han empezado y su por qué. Este cuadro suele ser de gran utilidad para analizar el comportamiento de nuestros Jobs programados en Talend.

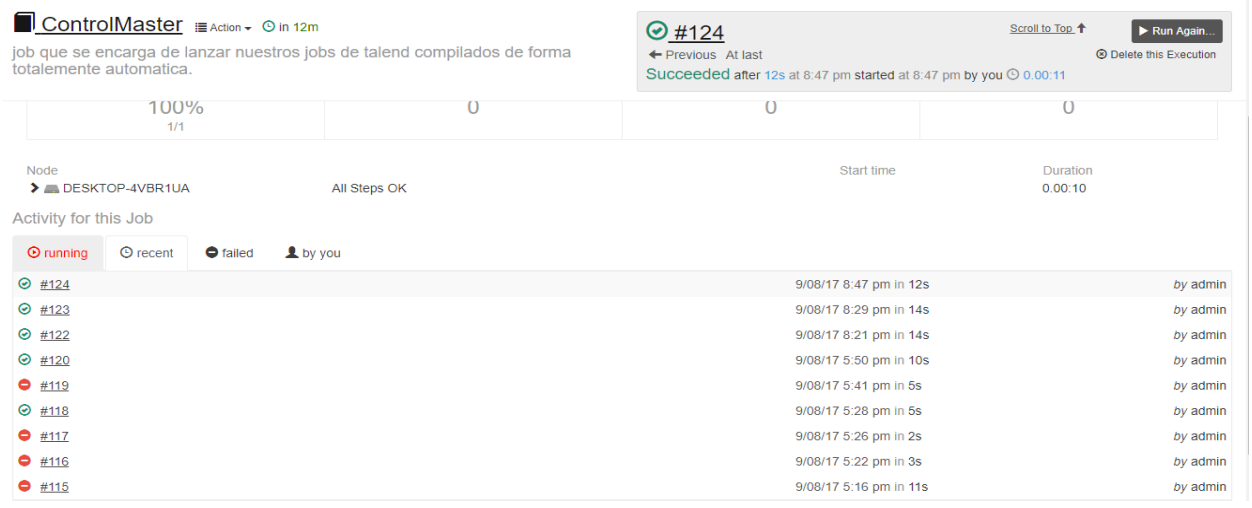


Figura 45. Cuadro de Rundeck que demuestra el historial de ejecución de un Job.

5.6 Explotación de los datos

Una vez definidos y automatizados nuestros procesos ETL, iremos cargando datos en nuestra base de datos PostgreSQL. Estos datos los analizaremos con la ayuda de la herramienta Tableau que a continuación veremos cómo funciona, para que finalmente saquemos conclusiones y demos respuestas a las preguntas que nos hemos planteado al principio de este proyecto.

Una vez instalada la herramienta Tableau en nuestro ordenador procedemos a conectarnos con la misma a la base de datos PostgreSQL (DWH). Para ello elegimos la conexión correspondiente a la base de datos PostgreSQL como se puede ver en la siguiente figura.

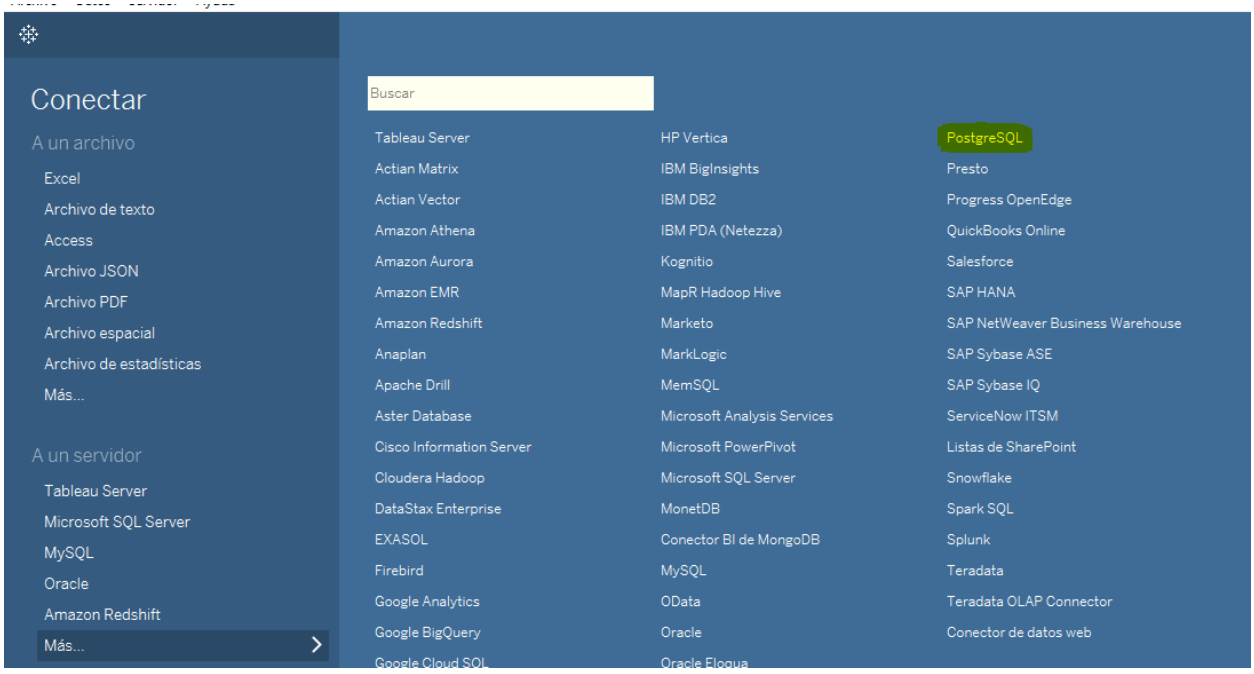


Figura 46. Conectores a bases de datos disponibles en la herramienta Tableau.

Luego introducimos los datos de nuestra base de datos PostgreSQL (DWH).

Figura 47. Cuadro con la configuración del conector de Tableau a la base PostgreSQL.

Una vez conectado a la base de datos podemos visualizar todas las tablas de las que esta dispone.

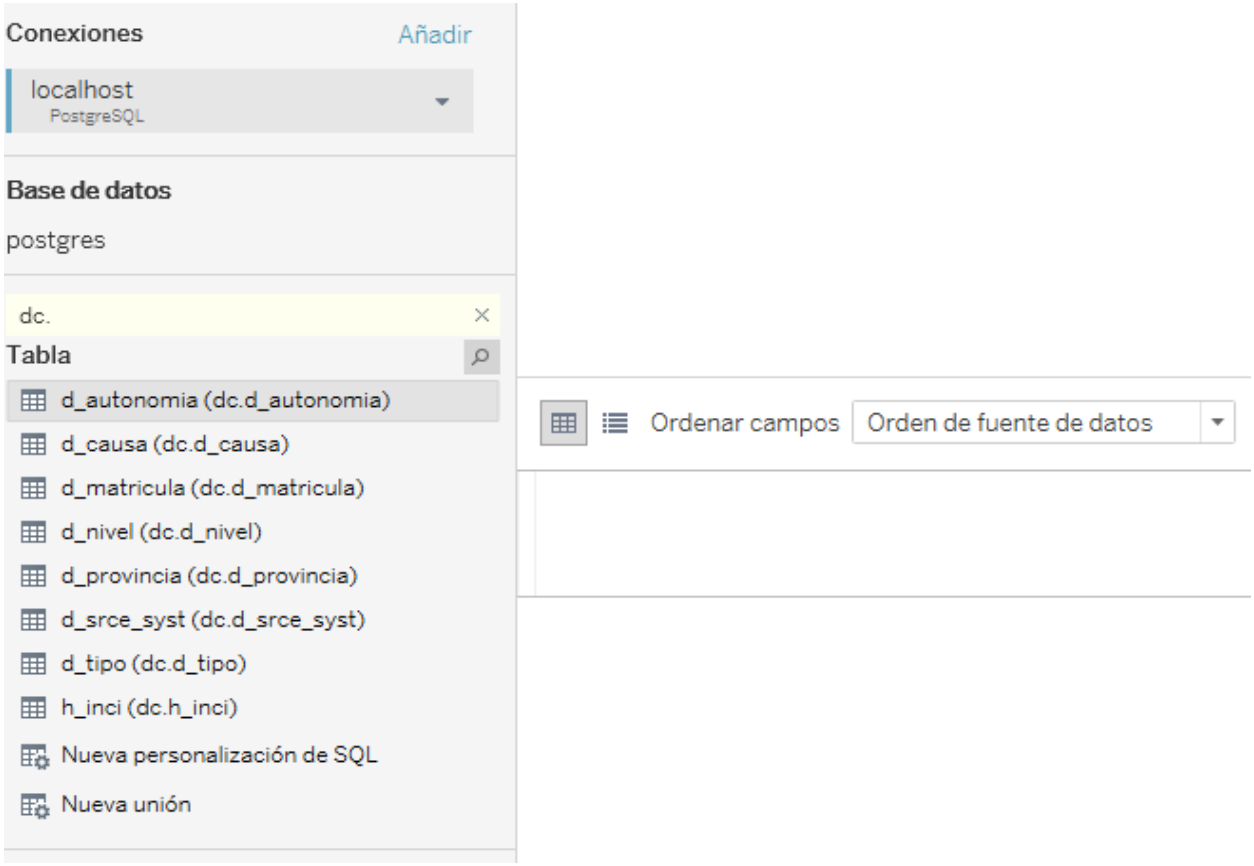


Figura 48. Tablas del esquema DC visualizadas en Tableau.

Montamos nuestro modelo con la herramienta de arrastrar y soltar que ofrece Tableau eligiendo el tipo de join según convenga.

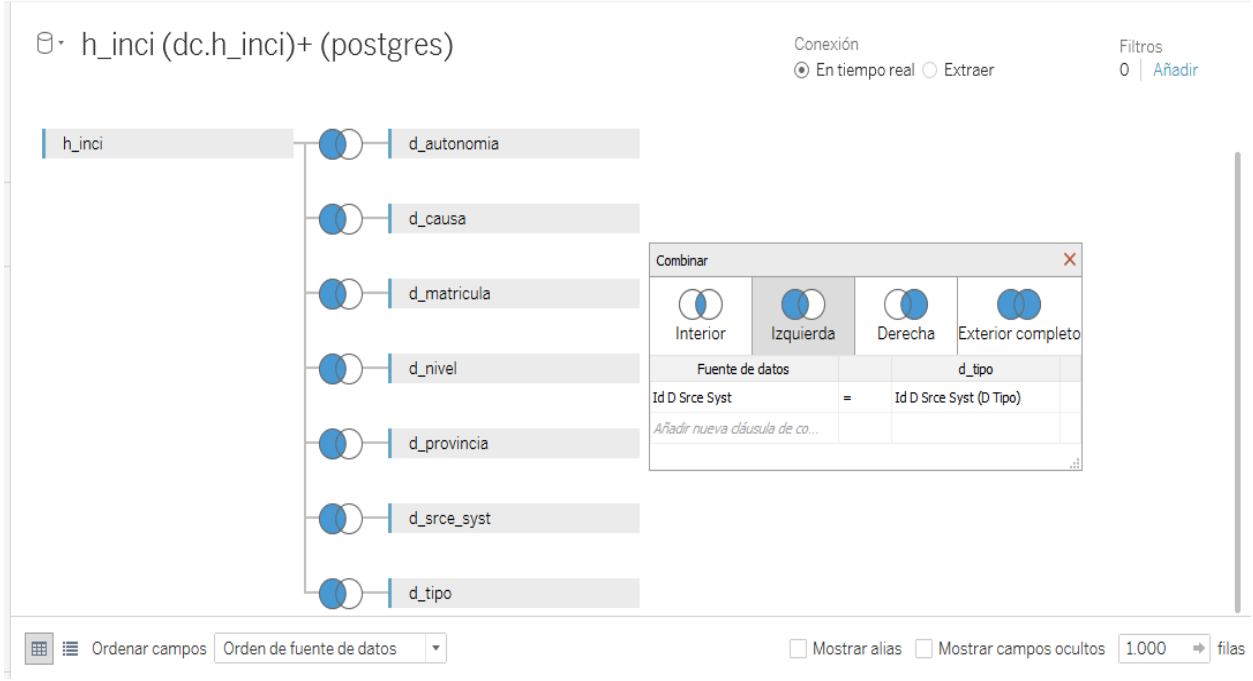


Figura 49. Modelo DWH montado en Tableau.

Una vez definido el modelo accedemos a diseñar los cuadros de mando que nos interesan. Pero debido a la limitación económica (y que el proyecto como se ha mencionado antes es una prueba de concepto) no se ha

podido comprar un servidor para alojar Rundeck y nuestra base de datos PostgreSQL (DWH) para dejar que cargue todo el tiempo la información de la base de datos de Open Data. Por lo cual para superar esta limitación y seguir adelante con el proyecto hemos dejado el ordenador que aloja el servidor Rundeck y la base de datos PostgreSQL encendido los 48h para cargar la información de dos días completos en nuestra base de datos PostgreSQL. Así que trabajaremos sobre esta información disponible teniendo en cuenta que los cuadros de mandos a desarrollar serán perfectamente válidos para el futuro también.

La tarea de análisis de datos requiere mucha destreza y capacidad analítica además de un conocimiento profundo de las reglas y términos de negocio de la empresa por la cual se está desarrollando el proyecto B.I. A esta tarea normalmente se suelen dedicar perfiles del tipo Business Analyst (analista de negocio), mientras que a las tareas de diseño ETL, y mantenimiento de base de datos se suelen dedicar perfiles de Business Intelligence developer (desarrollador de inteligencia de negocio).

A la hora de diseñar cuadros de mandos hay infinitas opciones e infinitas preguntas a las que se puede dar respuestas, pero debido al carácter del proyecto formularemos las siguientes preguntas que nos van a servir de ejemplo de cómo se diseñan cuadros de mando y que nos permitirán hacer un análisis inicial de los datos de los que disponemos.

- ¿Cuál es la distribución del número de incidencias por población en 48h?
- ¿Qué población tiene el mayor número de incidencias en 48h?
- ¿Cuál es la distribución del número de incidencias por provincia en 48h?
- ¿Qué tipo de incidencias es el más frecuente en 48h?
- ¿Cuál es la causa que provoca el mayor número de incidencias en 48h?

A continuación, presentamos los Cuadros de mando desarrollados.

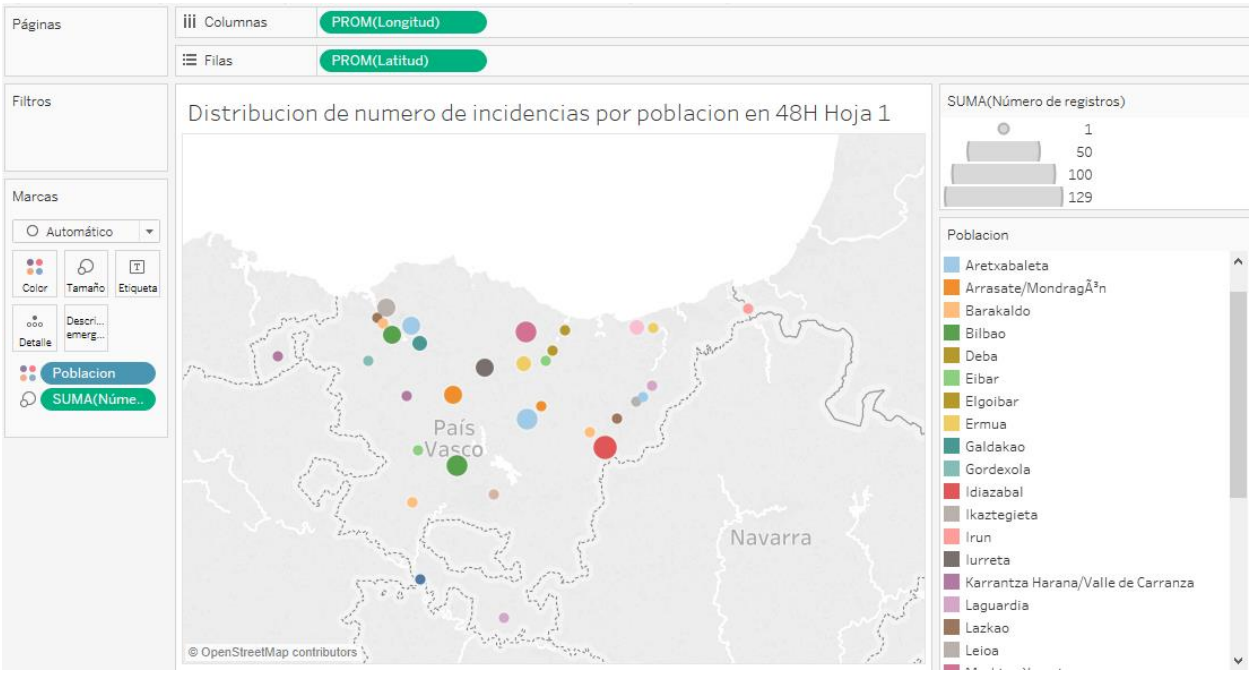


Figura 50. Distribución del número de incidencias por población en 48h.

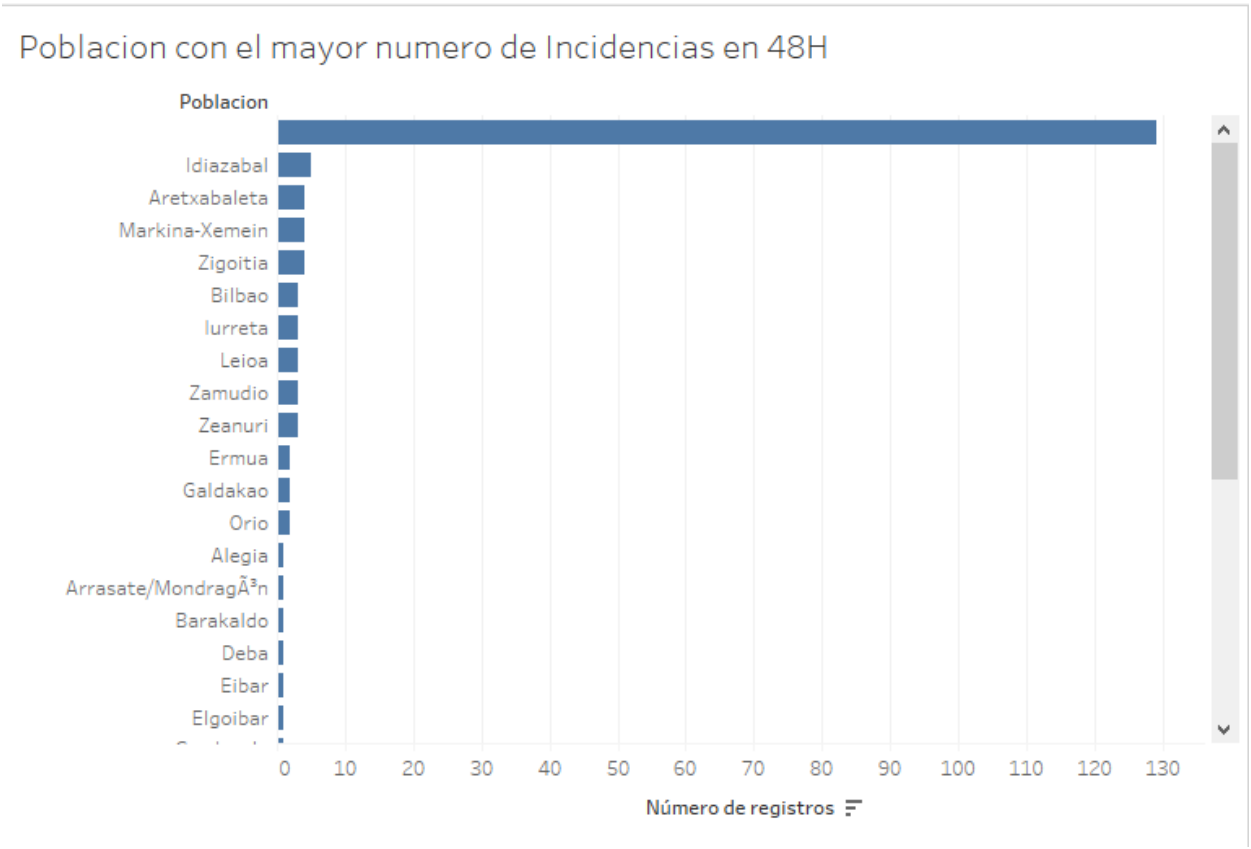


Figura 51. Población con mayor número de incidencias en 48h.

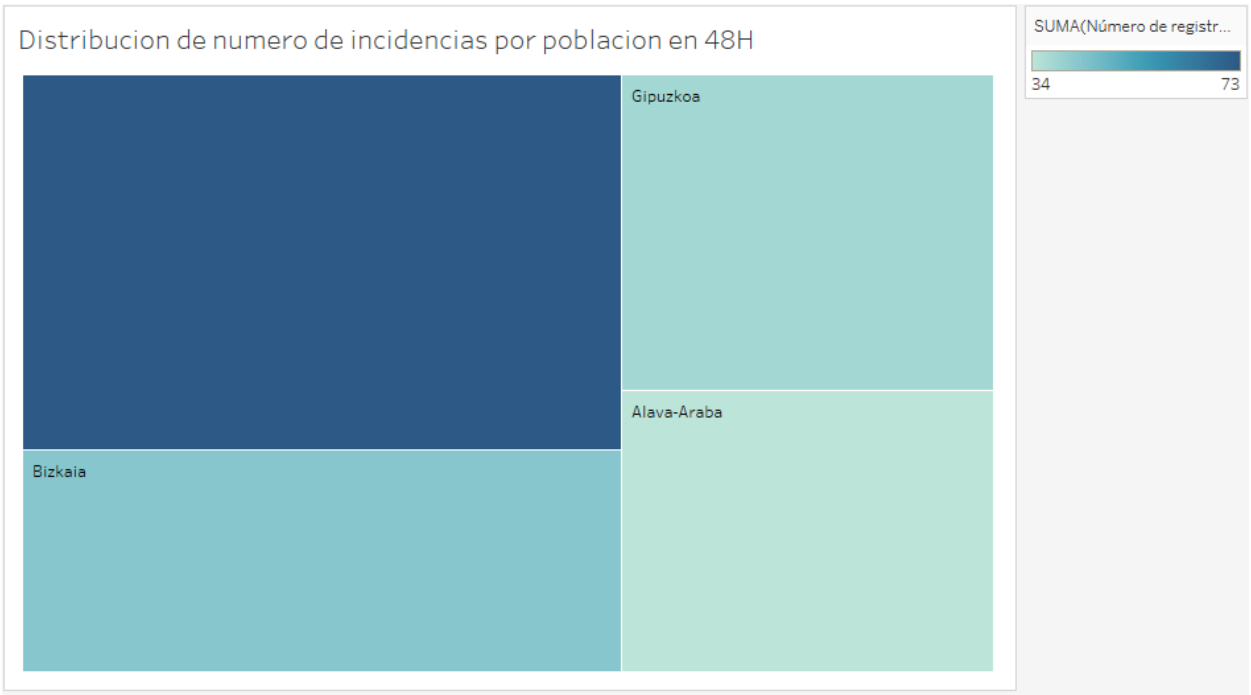


Figura 52. Distribución de número de incidencias por provincia en 48h.

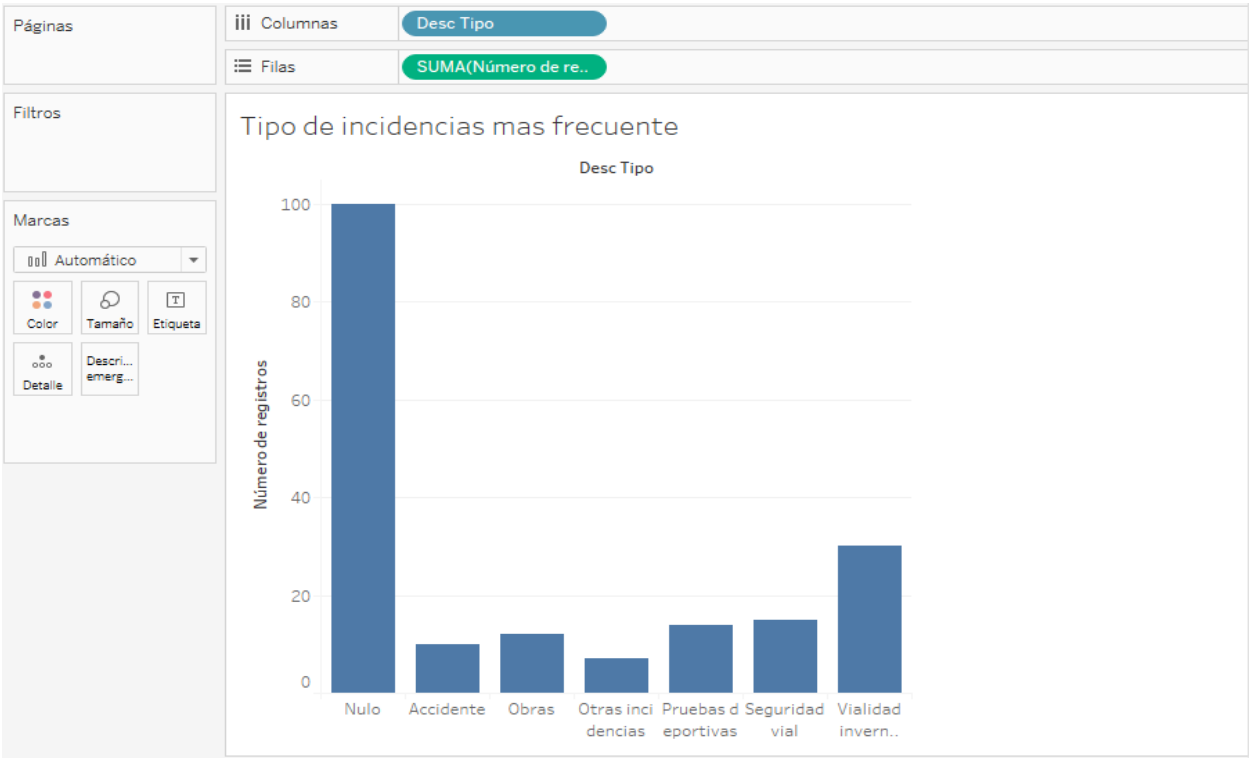


Figura 53. Tipo de incidencia más frecuente en 48h.

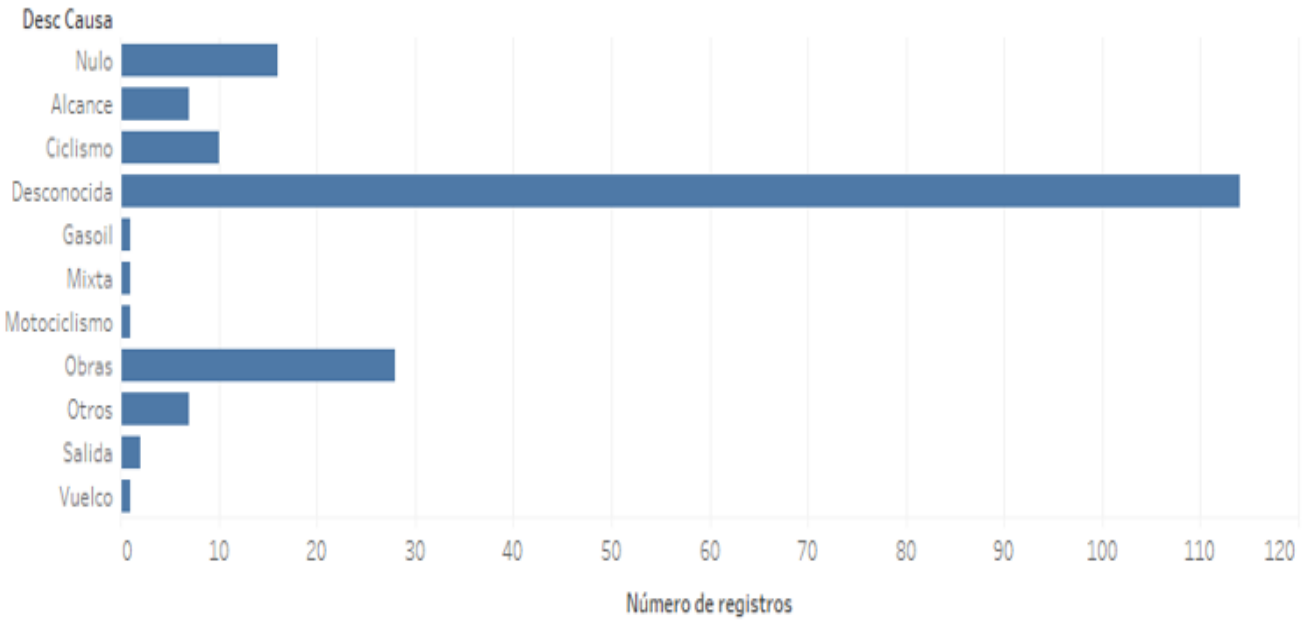


Figura 54. Distribución de causa de incidencias en 48h.

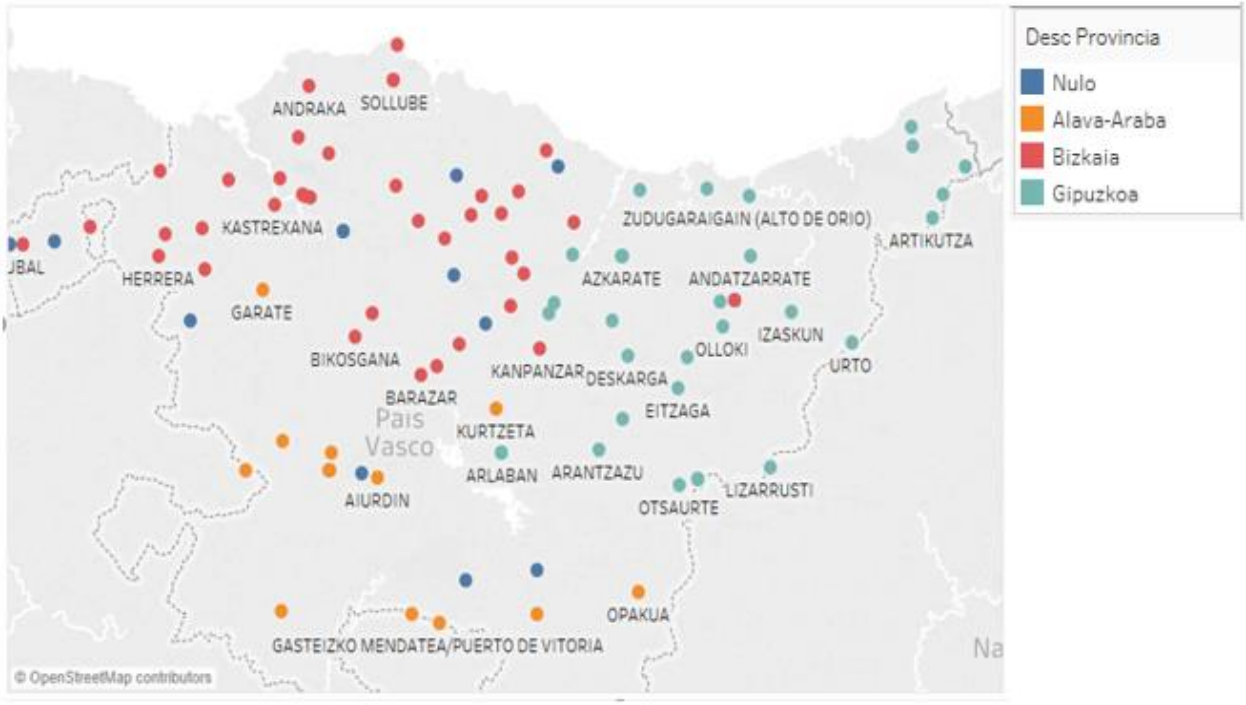


Figura 55. Distribución de todas las incidencias registradas en función de provincia en 48h.

5.7 Resultados

Las figuras anteriores responden a las preguntas que nos hemos planteado. De modo que la población con mayor número de incidencias en 48h es Idiazabal seguida de Aretxabaleta y Markina_xemein respectivamente como se puede ver en la figura 51 además el mayor tipo de incidencias es "vialidad invernal tramos" seguida de "seguridad vial" y "pruebas deportivas "como se puede ver en la figura 53.

Y analizando las causas principales de las incidencias en los 48h podemos identificar de la figura 54 que las "obras" seguidas de "ciclismo" y "alcance" representan las causas principales de incidencias en la comunidad de Euskadi. También podemos ver que hay un gran número de incidencias que vienen con causa no identificada o desconocida.

Y a nivel de provincia queda claro de la figura 52 que el mayor número de incidencias se ha registrado en "Bizkaia" con 43 incidencias seguida de "Gipuzkua" con 38 incidencias y por último "Alava-araba" con 34 incidencias y 73 incidencias sin identificar.

Como también vemos hay un gran número de incidencias que vienen de la base de datos Open Data con información incompleta, en estos casos se puede poner en contacto con el servicio técnico que facilita estos datos para comunicarles las incidencias de datos y ver si se puede llegar a alguna solución para evitar que estos problemas se repitan en el futuro.

Y finalmente y a partir de la tabla 19 y de la figura 56 podemos dar respuesta a la pregunta principal del proyecto como distribuir la flota de grúas de la empresa basándose únicamente sobre la información obtenida del análisis de nuestro Data Mart de incidencias y teniendo en cuenta que solo disponemos de información de los últimos 48h.

Provincia	Número de incidencias	Porcentaje de flotas a distribuir
Bizkaia	43	37,39%
Gipuzkua	38	33,04%
Alava-araba	34	29.56%

Tabla 19. Distribución de flotas por provincia y en porcentaje.

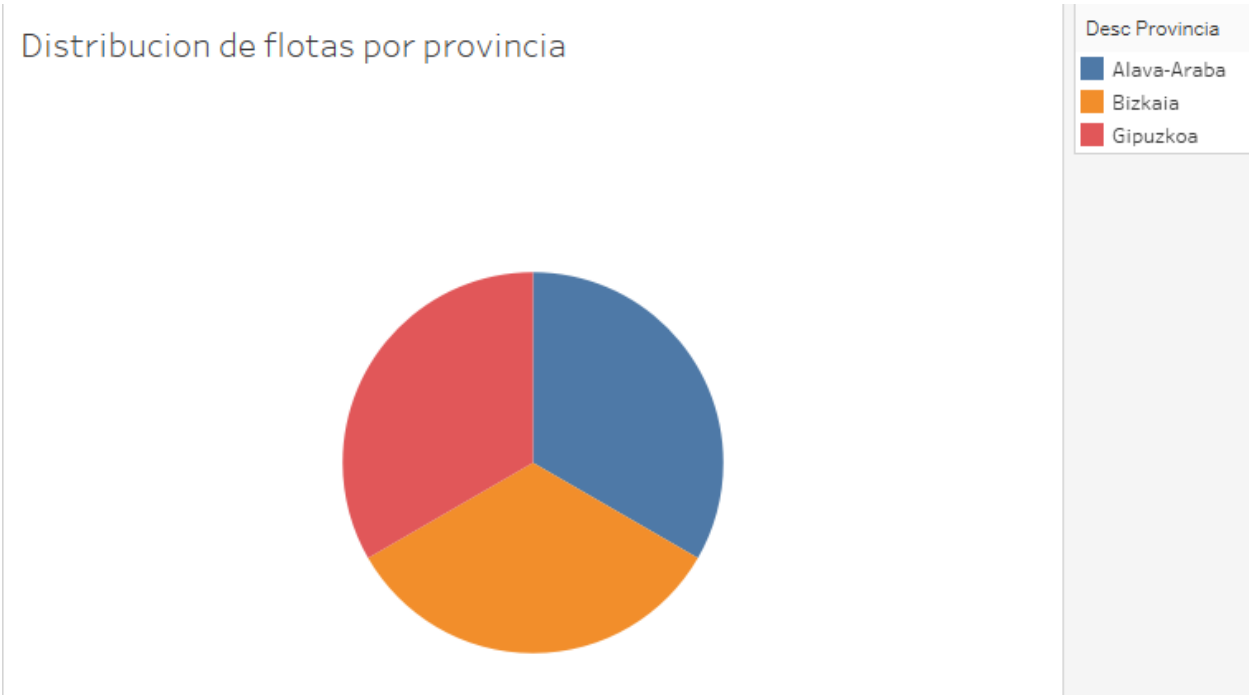


Figura 56. Distribución de flotas por provincia y en porcentaje.

6 CONCLUSIONES Y LÍNEAS FUTURAS

"La perfección se alcanza, no cuando no hay nada más que añadir, sino cuando ya no queda nada más que quitar."

- Antoine de Saint-Exupéry -

6.1 Conclusiones

En este Proyecto he intentado poner en práctica todo lo que he aprendido en mis prácticas en la empresa EUIGS a lo largo de más de 8 meses en el departamento de Business Intelligence en el puesto de Business Intelligence Developer junior.

Empecé a descubrir en aquel tiempo el todo novedoso mundo de los datos y de la información y su importancia para llevar a cualquier empresa a tomar buenas decisiones guiándola en el buen camino hacia el crecimiento y el éxito. Por lo cual tomé la decisión de desarrollar mi carrera profesional en este campo apasionante y por tanto aprovechar la oportunidad de realizar este proyecto para profundizar y adquirir nuevos conocimientos y destrezas en campos específicos como:

- Open Data.
- Bases de datos.
- ETL.
- Herramientas OLAP.
- Business Intelligence.

Por lo cual este trabajo representa una guía que además de introducir los fundamentos y conceptos básicos de Business Intelligence ofrece también un caso práctico con las herramientas principales usadas hoy en día en el mercado por los profesionales del campo.

6.2 Líneas futuras

En este proyecto han quedado fuera de su alcance algunos temas como los que mencionaremos a continuación

- Investigar y buscar nuevas bases de datos que pertenezcan a otras comunidades autónomas e integrarlas

con nuestro Data Mart de incidencias de tráfico ya desarrollado en este proyecto, construyendo una base de datos DWH nacional de incidencias de tráfico en toda España.

- Sería también muy interesante investigar cómo se podría aprovechar las bases de datos NoSql como MongoDB para alojar nuestra DWH.
- También sería de gran interés investigar las herramientas de Big Data y ver qué posibilidades y oportunidades ofrecen y cómo se pueden integrar con las herramientas de las que ya disponemos de Business Intelligence (ejemplo Talend for Big Data).
- Y por último investigar cómo se puede integrar una DWH en el que está montado un sistema de gestión de base de datos relacional por ejemplo PostgreSQL con un nuevo DWH montado en un sistema de gestión de base de datos no relacional por ejemplo MongoDB.

REFERENCIAS

[1] «Open Data Handbook,» [En línea]:
http://opendatahandbook.org/.

[2] «Open Data Soft,» [En línea]:
https://www.opendatasoft.fr/ressource-liste-portails-open-data-dans-le-monde/.

[3] «Postgresql,» [En línea]:
https://www.postgresql.org/docs/.

[4] «Rundeck,» [En línea]:
http://rundeck.org/docs/manual/index.html.

[5] «Tableau,» [En línea]:
https://www.tableau.com/es-es.

[6] «Talend,» [En línea]:
https://www.talend.com/download/data-integration-get-started/.

[7] «Developpez.com,» [En línea]:
http://taslimanka.developpez.com/tutoriels/projetbi/.

[8] J. C. Diaz, «Introducción al business intelligence».

[9] W. T. J. M. B. B. Margy Ross, «The Kimball Group Reader: Relentlessly Practical Tools for Data Warehousing and Business Intelligence».

[10] Datos.gob.es, «PLATAFORMAS DE PUBLICACIÓN DE DATOS ABIERTOS,» [En línea]: http://datos.gob.es/sites/default/files/informe-herramientas-publicacion.pdf.

GLOSARIO

B.I: Business Intelligence.

DWH: Data Warehouse.

ETL: Extract, Transform and Load.

FK: Forrign Key.

HOLAP: Hybrid OnLine Analytical Processing

MOLAP: OLAP Multidimensional

OLTP: OnLine Transaction Processing

OLAP: OnLine Analytical Processing.

POC: Proof of concept

PK: Primary key

ROLAP: Relational-OLAP.

SGDB: Sistema de Gestion de Base de Datos.

SGDBM: Sistema de Gestion de Base de Datos multidimensional.