

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

Máster Universitario en Big Data

Análisis y Optimización de un Flujo de Trabajo End-to-End

Autor
Gonzalo Barderas Adarraga

Dirigido por
José Luis Gahete Díaz

Madrid
Enero 2025

Gonzalo Barderas Adarraga, declara bajo su responsabilidad, que el Proyecto con título "**Análisis y Optimización de un Flujo de Trabajo End-to-End**" presentado en la ETS de Ingeniería (ICAI) de la Universidad Pontificia Comillas en el curso académico 2024/25 es de su autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.



Fdo.: ..... Fecha: 09 / 01 / 2025

Autoriza la entrega:

El Director del Proyecto

José Luis Gahete Díaz

GAHETE DIAZ JOSE LUIS - 11802871F

Firmado digitalmente por GAHETE DIAZ JOSE LUIS - 11802871F
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-11802871F, givenName=JOSE LUIS, sn=GAHETE DIAZ, cn=GAHETE DIAZ JOSE LUIS - 11802871F
Fecha: 2025.01.10 08:07:31 +01'00'

Fdo.: ..... Fecha: ..... / ..... / .....

V. B. del Coordinador de Proyectos

Carlos Morrás Ruiz-Falcó

Fdo.: ..... Fecha: ..... / ..... / .....

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D.Gonzalo Barderas Adarraga **DECLARA** ser el titular de los derechos de propiedad intelectual de la obra: Análisis y Optimización de un Flujo de Trabajo End-to-End, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, los derechos de digitalización, de archivo, de reproducción, de distribución y de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

(a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.

(b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.

(c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.

(d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.

(e) Asignar por defecto a estos trabajos una licencia Creative Commons.

(f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente*).

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

(a) Que la Universidad identifique claramente su nombre como autor de la misma

(b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.

(c) Solicitar la retirada de la obra del repositorio por causa justificada.

(d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

(a) El autor se compromete a:

(b) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.

(c) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.

(d) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.

(e) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.

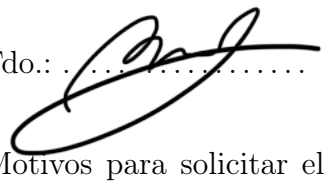
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.

- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.

- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 09 de enero de 2025

ACEPTA

Fdo.: 

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

Máster Universitario en Big Data

Análisis y Optimización de un Flujo de Trabajo End-to-End

Autor
Gonzalo Barderas Adarraga

Dirigido por
José Luis Gahete Díaz

Madrid
Enero 2025

Resumen

Los objetivos principales de este trabajo de fin de máster son analizar en profundidad y optimizar un flujo de trabajo end-to-end. En este contexto, el flujo de trabajo es un proceso en el que se transforma y carga un conjunto de datos pertenecientes a un cliente de SDG Group España. De hecho, el cliente es una empresa de gran consumo, y los datos que intervienen tienen relación con la participación física de sus productos en el canal de alimentación.

Este proyecto surge de la necesidad de mejorar un proceso que, desde un principio, no se había diseñado e implementado de la manera más óptima. Consecuentemente, el tiempo de ejecución del proceso era bastante elevado, además de que la forma del diseño del flujo de trabajo resultaba poco eficiente. Es por ello que la optimización del proyecto se basa en reducir el tiempo de ejecución y simplificar la estructura del flujo de trabajo.

A través del uso de Common Table Expressions (CTEs), y entendiendo bien la lógica de la tabla final que se desea conseguir, se han logrado alcanzar los objetivos establecidos. De este modo, se ha conseguido reducir el tiempo de ejecución del flujo de trabajo en un $80,34\,\%$. Aparte, se ha conseguido reducir significativamente la complejidad del flujo de tarbajo, ofreciendo un $68,97\,\%$ menos de consultas SQL en el flujo de trabajo optimizado, comparado con el flujo de trabajo definido originalmente. Es más, todo lo anterior se ha conseguido obteniendo una tabla final consistente, bien estructurada y con el formato que desea el cliente.

Abstract

The primary objectives of this master's thesis are to thoroughly analyze and optimize an end-to-end workflow. In this context, the workflow involves transforming and loading a dataset belonging to a client of SDG Group Spain. Specifically, the client is a large consumer goods company, and the data pertains to the physical presence of its products in the food channel.

This project arose from the need to improve a process that, from the beginning, had not been designed and implemented in the most optimal way. Consequently, the process execution time was quite high, and the workflow design was inefficient. Therefore, the optimization of the project focuses on reducing execution time and simplifying the workflow structure.

By using Common Table Expressions (CTEs) and thoroughly understanding the logic of the desired final table, the set objectives were achieved. The workflow execution time was reduced by 80.34 %, and the complexity of the workflow was significantly decreased, with 68.97 % fewer SQL queries in the optimized workflow compared to the original one. Moreover, all these improvements were achieved while delivering a consistent, well-structured final table in the format desired by the client.

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a todas las personas que han hecho posible la realización de este trabajo de fin de máster.

En primer lugar, agradezco a la empresa SDG Group por brindarme la oportunidad de realizar las prácticas en un entorno profesional tan enriquecedor. La experiencia ha sido fundamental para mi desarrollo académico y profesional.

Especialmente, me gustaría agradecer a Gloria Cabrero Posch, compañera de equipo, por su constante apoyo y guía a lo largo de este proyecto. Su disposición para resolver mis dudas, aportar ideas y escuchar mis avances ha sido invaluable. Gloria, tu paciencia y conocimiento han sido esenciales para llevar este proyecto a buen puerto.

También quiero expresar mi gratitud a Mario Quiñones González, director del equipo, quien me presentó el proyecto y estuvo dispuesto a escucharme en el proceso de selección del mismo. Su confianza en mis capacidades y su orientación fueron determinantes para elegir y desarrollar este trabajo.

A todos ustedes, muchas gracias por su tiempo, dedicación y apoyo. Sin su colaboración, este proyecto no habría sido posible.

Índice general

1. Introducción **1**
1.1. Contextualización del proyecto . . . 1
1.1.1. Importancia del proyecto en el ámbito tecnológico y empresarial . . 1
1.1.2. Breve descripción de proceso ETL completo . . . 2
1.2. Objetivos del proyecto . . . 3
1.2.1. Justificación de los Objetivos . . . 3
1.3. Motivación . . . 4

2. Marco Teórico **5**
2.1. Conceptos clave . . . 5
2.2. Importancia y Beneficios de los Procesos ETL en la Integración y Análisis de Datos . . . 9
2.3. Métodos de Optimización de procesos ETL . . . 10
2.3.1. Indexación y Optimización de Consultas . . . 10
2.3.2. Particionamiento . . . 10
2.3.3. Common Table Expressions (CTEs) . . . 11

3. Metodología **13**
3.1. Descripción del Caso de Estudio . . . 13
3.1.1. Descripción del contexto de los productos en el canal de alimentación 13
3.1.2. Detalle del Caso de Estudio y su Relevancia . . . 14
3.2. Herramientas y Tecnologías Utilizadas . . . 14
3.2.1. Stratio Rocket . . . 14
3.2.2. Stratio Discovery . . . 15
3.2.3. PostgreSQL . . . 16
3.2.4. Tableau . . . 17
3.3. Implementación de las Herramientas y Tecnologías Utilizadas . . . 17

4. Descripción y Funcionamiento del Flujo de Trabajo Original **19**
4.1. Entradas al Flujo de Trabajo . . . 19
4.2. Continuación del Flujo de trabajo . . . 20
4.3. Tabla Final Generada por el Flujo Original . . . 24

5. Optimización del Flujo de Trabajo **25**
5.1. Cambios Realizados . . . 25
5.1.1. Entradas al nuevo flujo de trabajo . . . 25
5.1.2. Transformaciones tipo *Trigger* . . . 27
5.1.3. Nuevo flujo de trabajo . . . 30
5.2. Tabla final generada por el nuevo flujo . . . 30

6. Limitaciones del Proyecto **33**
6.1. Técnicas de optimización identificadas y valoradas 33
6.2. Limitaciones encontradas 33
6.3. Impacto de las limitaciones 34

7. Resultados y Conclusión **35**
7.1. Resultados 35
7.1.1. Técnicas de optimización utilizadas 35
7.1.2. Mejoras en el rendimiento del flujo de trabajo 35
7.1.3. Discusión de los resultados obtenidos 36
7.2. Conclusiones 37

Apéndice **38**

A. Códigos anonimizados presentes en el flujo de trabajo optimizado **39**
A.1. Código de `PRINCIPAL_DL` 39
A.2. Código de `Trigger` 41

Apéndice **45**

B. Códigos anonimizados presentes en el flujo de trabajo original **47**
B.1. Consultas del flujo de trabajo original que equivalen a la entrada `PRINCIPAL_DL` en el flujo nuevo. 47
B.2. Código de `TRIGGER_DL` del flujo de trabajo original 49

Bibliografía **55**

Índice de figuras

2.1. Diagrama de un proceso ETL. Fuente: Bartley, 2023 7

3.1. Flujo de Trabajo entre las Herramientas 18

4.1. Flujo de Trabajo Original 21

5.1. Entradas Nuevas al Flujo de trabajo 26
5.2. Entradas Originales al Flujo de trabajo 26
5.3. Registro de la tabla final vieja 30
5.4. Mismo registro en la tabla final nueva 30
5.5. Flujo de Trabajo Nuevo 31

Índice de cuadros

4.1. Entradas al Flujo de Trabajo 22

5.1. Operaciones clave de `Trigger` 29

7.1. Comparación de tiempos de ejecución entre procesos 36
7.2. Métricas de evaluación de la forma de los flujos de trabajo 36

Listados

A.1. Código de `PRINCIPAL_DL` de la Figura 5.5 39
A.2. Código de `Trigger` de la Figura 5.5 41
B.1. Código de `ANIO_MAX` de la Figura 4.1 47
B.2. Código de `AUDIT_DL` de la Figura 4.1 47
B.3. Código de `AUDIT_DL_FILTRO` de la Figura 4.1 48
B.4. Código de `PRINCIPAL_DL` de la Figura 4.1 48
B.5. Código de `TRIGGER_DL` de la Figura 4.1 49

Acrónimos y Siglas

ICAI	Insitituto Católico de Artes e Industrias
BBDD	Bases de Datos
ETL	Extract, Transform y Load
SQL	Structured Query Language
CTE	Common Table Expression
BI	Business Intelligence
ACID	Atomicidad, Consistencia, Aislamiento y Durabilidad

Capítulo 1

Introducción

1.1. Contextualización del proyecto

En el actual entorno empresarial, la capacidad de gestionar y analizar datos de manera eficiente es fundamental para obtener una ventaja frente a los competidores. Las empresas que operan en sectores competitivos, como el de alimentación, necesitan manejar una inmensa cantidad de datos eficazmente. La gestión eficiente de los datos es crucial ya que les permite optimizar operaciones, mejorar la toma de decisiones, personalizar la experiencia del cliente y mantenerse relevantes en un mercado en constante cambio.

El trabajo de fin de máster se enmarca dentro de esta realidad. Se trata de un proyecto realizado con SDG Group España, en el cual se va a comprender a fondo, analizar y optimizar un flujo de trabajo end-to-end basado en lenguaje SQL (Structured Query Language), con el fin de mejorar un proceso interno esencial para un cliente.

El flujo de trabajo objeto de estudio está definido y en producción en Stratio. En concreto, en este trabajo se emplean dos funcionalidades que proporciona el servicio: Stratio Rocket y Stratio Discovery. Stratio Rocket permite construir el flujo completo de trabajo; se generan unas entradas de las que se obtiene la información, se transforman adecuadamente los datos según los requisitos que se necesitan y, posteriormente, se almacenan en base de datos. En cambio, en Stratio Discovery se presenta una *technical view* de las tablas originales y de la tabla final. Es decir, la herramienta se utiliza como una ayuda de visualización de los datos que se ven involucrados en el flujo de trabajo. Esta misma herramienta también tiene implantada una funcionalidad que permite realizar consultas SQL. De este modo, el uso de Stratio Discovery ha sido fundamental para poder visualizar resultados intermedios del flujo de trabajo.

1.1.1. Importancia del proyecto en el ámbito tecnológico y empresarial

La optimización del flujo de trabajo end-to-end tiene implicaciones significativas tanto desde un punto de vista tecnológico como empresarial. Tecnológicamente el uso de Stratio Rocket para crear y ejecutar el flujo de trabajo, junto con Stratio Discovery, permite manejar grandes volúmenes de información eficientemente. Estas herramientas brindan la flexibilidad y solidez necesarias para realizar análisis complejos y obtener conocimientos valiosos a partir de los datos.

Empresarialmente, optimizar estos procesos internos es crucial para mejorar la eficiencia operativa y la calidad del servicio al cliente. En esta instancia, la empresa cliente, que opera en el sector de gran consumo, necesita monitorizar y analizar datos detallados sobre los metros lineales de productos, tanto dentro como fuera de los lineales en supermercados e hipermercados. Mejorar la precisión del análisis de estos datos posibilita al cliente optimizar la disposición de sus productos, maximizar las ventas y tomar decisiones estratégicas basadas en información precisa y actualizada.

Para entender la importancia que tiene el proyecto para el cliente, primero hace falta definir el concepto de Foto de Éxito (FE). En el ambiente empresarial, la Foto de Éxito hace referencia a la visión clara del estado futuro de la empresa. En este caso, y por lo tanto para la empresa cliente, se trata de una representación conceptual de los objetivos a alcanzar. Para ser más preciso, la Foto de Éxito establece un escenario ideal al que la empresa quiere llegar todos los años. Cada año se establecen unos nuevos baremos para la Foto de Éxito. Estos objetivos se definen conociendo los metros lineales de productos que tiene la empresa en cada una de las tiendas. Se entiende por Dentro de Lineal (DL) el espacio que ocupan los productos en estanterías convencionales, y por Fuera de Lineal (FL) el resto del espacio que ocupan los productos en los supermercados, como stands de promoción y neveras.

1.1.2. Breve descripción de proceso ETL completo

El flujo de trabajo end-to-end forma parte de un proceso ETL (Extract, Transform, Load). Se ha incluido esta sección para explicar el papel que toma el flujo de trabajo end-to-end dentro del proceso total. La ETL completa se estructura en varias etapas para asegurar una gestión eficiente de los datos. Las principales fases del proceso son las siguientes:

- **Recolección de Datos:** Los datos son proporcionados por el cliente y abarcan diversas métricas relevantes para la gestión de productos en supermercados e hipermercados. Estos datos se almacenan en varias bases de datos.
- **Transformación de Datos:** Usando Stratio Rocket, se diseña el flujo de trabajo con el que limpiar, transformar y preparar los datos para su análisis. El flujo de trabajo original se divide en dos grandes secciones: dentro de lineal y fuera de lineal, cada una subdividida en cuatro partes específicas: datos del año actual, datos del año anterior, datos de Foto de Éxito y datos del escenario de mínimos (MIN).
- **Almacenamiento de Datos:** Los datos transformados se almacenan en una única tabla en Postgres. Almacenar una sola tabla facilita consultar los datos eficientemente usando SQL y permite realizar análisis detallados. A su vez, estudiar e inspeccionar la tabla final proporciona al cliente *insights* valiosos sobre la disposición de sus productos en los puntos de venta.

De estas etapas, el trabajo está enfocado en la transformación de los datos. Es ahi donde se enmarca el flujo de trabajo end-to-end; el flujo de trabajo es un proceso de transformación de los datos. En cambio, no se tiene acceso a la etapa de recolección de datos. La sección de carga de datos se define en Stratio Rocket, en la ultima parte del flujo. Aun así, no se tiene acceso a la base de datos donde se almacena la tabla que se genera. Tampoco se puede

inferir en el formato de almacenamiento de la tabla final, que el cliente ha establecido que se guarde en formato parquet.

1.2. Objetivos del proyecto

Como se instaura en el título del trabajo, el principal objetivo del proyecto es comprender y estudiar el flujo de trabajo original. Tras estudiar el proceso, también se quiere mejorar su eficiencia en términos de tiempo de ejecución y forma.

En cuanto a la optimización temporal, el objetivo es reducir el tiempo total que tarda en ejecutarse el proceso de transformación de datos. Esto hará que los datos estén disponibles más rápido para su análisis, y mejorará la capacidad de respuesta y eficiencia al tomar decisiones con esos datos.

Respecto a la optimización en términos de forma, se busca rediseñar partes del flujo de trabajo para hacerlo más sencillo y claro, sin comprometer la precisión y la integridad de los datos finales. Stratio Rocket permite generar el flujo completo de forma esquemática y organizada. Debido a esta característica, y a la distribución original del flujo de trabajo, es posible crear un diseño más eficiente, donde el flujo simplificado resulte en una tabla final con las caracteríticas y estructura que establece el cliente.

1.2.1. Justificación de los Objetivos

La selección de los objetivos de este proyecto responde a la necesidad de mejorar y optimizar un flujo de trabajo integral generado por SDG Group España. La gestión de los datos que intervienen en el flujo cobra relevancia dada la importancia que tienen para el cliente. El análisis detallado del flujo original es el paso inicial para identificar y corregir ineficiencias. Esta fase proporciona una base sólida para construir mejoras y asegurar que cualquier cambio esté bien fundamentado y alineado con los objetivos del cliente.

El objetivo de optimizar el tiempo es sumamente importante en un contexto donde la rapidez en la disponibilidad de datos puede marcar una gran diferencia en la respuesta y toma de decisiones estratégicas. Por lo tanto, reducir el tiempo de ejecución del proceso permite obtener una mayor agilidad en el análisis de los datos que se tratan. Esto es esencial en entornos dinámicos y competitivos, como el sector alimenticio. Mejorar la eficiencia temporal no solo facilita respuestas más rápidas, también puede traducirse en ahorros de costos operativos y mejor uso de los recursos disponibles.

Por otro lado la optimización en términos de forma aborda la complejidad inherente a muchos flujos de trabajo; pueden ser redundantes y difíciles de gestionar visualmente. Simplificar el flujo para hacerlo más claro ayuda a los usuarios a entenderlo y manejarlo de mejor manera. Este ajuste no compromete la precisión ni la integridad de los datos finales; mantener la fiabilidad del proceso es fundamental. El uso de herramientas como Stratio Rocket es beneficioso ya que permite visualizar el flujo de forma esquemática y eliminar redundancias o pasos innecesarios.

Al simplificar el diseño del flujo de trabajo se obtiene una tabla final con los datos deseados, pero con una estructura más clara y manejable. Al mismo tiempo, mejora la eficiencia

operativa, reduciendo errores y facilitando la formación y transferencia de conocimientos dentro de la organización.

1.3. Motivación

La motivación de este trabajo de fin de máster se basa en varios factores clave. Sobresalen la importancia del estudio y el valor añadido que aporta tanto personal como profesionalmente.

Participar en un proyecto real ofrece una experiencia valiosa ya que se enfrentan desafíos relacionados con el mundo empresarial. Del mismo modo, también permite entender la dinámica corporativa. Esta oportunidad me concede aplicar conocimientos teóricos estudiados durante el máster a situaciones prácticas reales. Así, la experiencia ayuda a confirmar y fortalecer mi formación académica. Además, el proyecto va a servir para gestionar soluciones según las necesidades específicas de una organización.

La capacidad de identificar problemas y proponer soluciones originales es muy valorada en cualquier ámbito profesional. Este proyecto me ofrece la oportunidad de desarrollar un enfoque crítico y analítico esencial para comprender las complejidades de un flujo de trabajo y aumentar su eficiencia. Poder innovar y probar distintas estrategias estimula tanto la creatividad como el pensamiento lateral, habilidades clave para la resolución de problemas.

Contribuir a la eficiencia operativa de una empresa es un gran motivador. Al optimizar el flujo de trabajo se busca reducir tiempos, de la misma manera que mejorar la calidad y la precisión de los datos procesados. Esta mejora en la eficiencia puede afectar directamente a la competitividad y capacidad del cliente para enfrentar desafíos del mercado. Es motivador saber que mi trabajo puede impactar de manera tangible y positiva al cliente.

El uso de herramientas avanzadas como Stratio para implementar y optimizar el flujo de trabajo es una motivación adicional. Adquirir y mejorar habilidades en tecnologías modernas refuerza mi perfil profesional y me posiciona mejor en el mercado laboral. Estas competencias técnicas son esenciales en un mundo cada vez más digitalizado y basado en datos.

Capítulo 2

Marco Teórico

Para completar este informe, es relevante mencionar y explicar un conjunto de conceptos que resultan esenciales para la comprensión y entendimiento del trabajo. A su vez, se van a introducir diversos métodos a tener en cuenta a la hora de optimizar el flujo de trabajo.

2.1. Conceptos clave

En esta sección se presentan una serie de conceptos fundamentales que son esenciales para entender la implementación del proyecto, los objetivos, las metodologías y los resultados. Comprender estos términos aclara los procesos y técnicas realizadas. Para explicar adecuadamente los conceptos clave, se va a ofrecer una definición precisa para cada término y se explicará su relevancia en el proyecto.

ETL

ETL es el proceso de extracción de datos de diversas fuentes, su transformación según los requisitos de análisis y su carga en una base de datos destino. La función fundamental de las ETLs es preparan los datos para un análisis posterior (Surge, 2024).

La transformación es un paso crucial; consiste en eliminar inconsistencias, ensamblar los datos a un formato consistente, añadir campos que faltan, resumir datos detallados y derivar nuevos campos para almacenar datos calculados. (Surge, 2024).

Varias herramientas se ven involucradas para diseñar y ejecutar el proceso ETL completo. De estas herramientas, la que cobra más importancia es Stratio Rocket. La relevancia de esta herramienta reside en que con ella se diseñan e implementan las transformaciones pertinentes del proceso, y ahí está el foco del trabajo.

Data Warehouse

Un almacén de datos (Data Warehouse) es un sistema centralizado que agrega y almacena datos provenientes de diversas fuentes para apoyar en el análisis y la toma de decisiones empresariales (IBM, 2024a). Este sistema se caracteriza por su capacidad de integrar grandes volúmenes de datos históricos y actuales, organizados de manera estructurada, lo cual es esencial para las actividades de inteligencia empresarial (BI) y análisis

de datos (Surge, 2024).

La centralización de los datos es un concepto que hace referencia a la idea de que los Data Warehouse recopilan datos de múltiples fuentes, incluyendo sistemas transaccionales, bases de datos relacionales, y otros repositorios de datos. Esta centralización permite tener una "fuente única de verdad", donde todos los datos relevantes están disponibles para el análisis y reporte. Aun así, la mayoría están construidas alrededor de un sistema de base de datos relacional (Starburst, 2024).

A diferencia de las bases de datos transaccionales que están diseñadas para manejar un gran volumen de transacciones en tiempo real, los Data Warehouses están optimizados para consultas complejas y análisis de grandes conjuntos de datos (IBM, 2024a).

En la ETL, PostgreSQL actúa como el almacén de datos, siendo la base de datos destino donde se almacenan los datos tratados y modificados. Esta herramienta actúa como el repositorio central de datos transformados y preparados para el análisis.

Base de Datos Relacional

Una base de datos relacional es un sistema de almacenamiento de datos que organiza la información en tablas interrelacionadas, lo que permite gestionar grandes volúmenes de datos de manera eficiente y flexible. Las tablas son el fundamento de una base de datos relacional. Cada tabla representa cierta información del mundo real, como por ejemplo información relativa a clientes, productos o pedidos (IBM, 2024b).

Una tabla esta compuesta por filas y columnas. Las columnas definen los diferentes campos de datos que cada fila puede tener. Las filas contienen registros individuales, y cada fila en una tabla representa un registro único de datos. Las columnas definen los tipos de datos para cada registro en la tabla, y cada columna tiene un nombre y un tipo de datos asociado (IBM, 2024b).

En la ETL, los datos relevantes que se quieren transformar están almacenados en varias bases de datos. Simultáneamente, Postgres es una base de datos relacional que actúa como data warehouse. Se trata de una herramienta cuyas características la permiten tomar ese papel.

Atendiendo a la Figura 2.1, se entiende de mejor forma cómo se involucran las bases de datos y los Data Warehouse dentro de un proceso ETL. Siguiendo el esquema presentado en la figura, la información proveniente de una base de datos pasa por una “staging area“, dónde se alteran los datos según unos requisitos. La salida del proceso de transformación de datos llega a un Data Warehouse para ser almacenada.

En este caso, PostgreSQL hace de Data Warehouse, y Stratio Rocket es la herramienta que ocupa el “staging area“. El rol de base de datos de origen lo desempeñan varias bases de datos distintas. Stratio es capaz de leer las tablas procedentes de diferentes bases de datos. La herramienta de visualización Tableau ocupa el tramo de “analytics“ que aparece en la Figura 2.1.

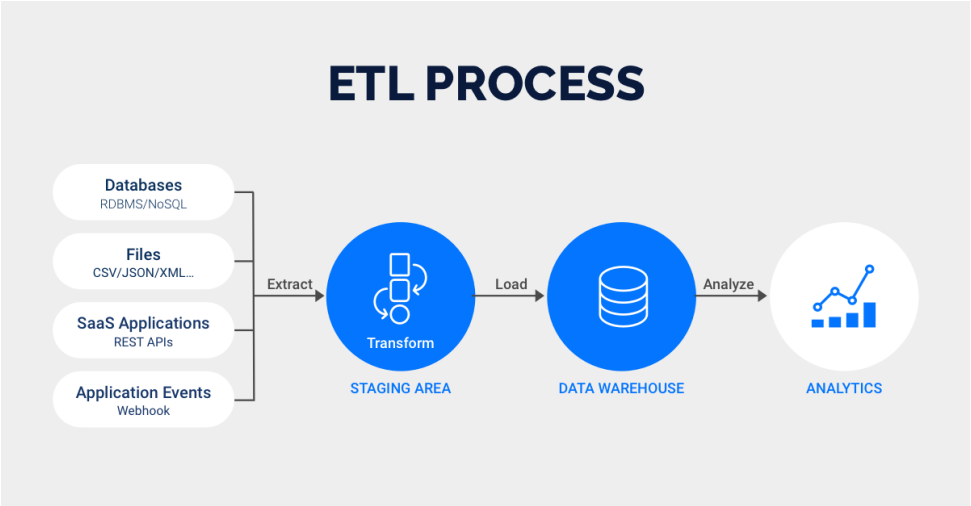


Figura 2.1: Diagrama de un proceso ETL. Fuente: Bartley, 2023

ACID

Son las siglas de las propiedades que poseen las transacciones. Las transacciones son comandos de interacción con una base de datos. Un ejemplo común sería una transacción bancaria, donde una cantidad definida de dinero se transfiere de una cuenta a otra. Las propiedades de las transacciones son las siguientes:

- **Atomicidad (Atomicity)**: La atomicidad garantiza que todas las operaciones dentro de una transacción se realicen como una unidad indivisible. Esto significa que, o todas las operaciones se completan con éxito, o ninguna de ellas se aplica (IBM, 2024b).
- **Consistencia (Consistency)**: La consistencia asegura que una transacción lleva la base de datos de un estado válido a otro estado válido. Esto significa que cualquier transacción completada no viola las reglas de integridad de la base de datos (IBM, 2024b).
- **Aislamiento (Isolation)**: El aislamiento asegura que las transacciones concurrentes se ejecutan como si fueran secuenciales. Es decir, el estado intermedio de una transacción no es visible para otras transacciones hasta que la primera se complete (IBM, 2024b).
- **Durabilidad (Durability)**: La durabilidad garantiza que una vez que una transacción se ha completado, los cambios realizados son permanentes y se mantienen incluso en caso de fallos del sistema (IBM, 2024b).

SQL

SQL es un lenguaje de programación utilizado para gestionar y manipular bases de datos relacionales. Facilita la ejecución de operaciones complejas de administración y análisis de datos, y es fundamental para el desarrollo y mantenimiento de aplicaciones que dependen de bases de datos. Comunmente, a las instrucciones escritas en lenguaje SQL se las conoce como consultas (IBM, 2024a).

En este trabajo se usa SQL en Stratio Discovery para obtener información valiosa acerca de los datos de origen y de la tabla final que genera el flujo de trabajo. También se utiliza SQL para definir y llevar a cabo las etapas de transformación del proceso ETL, dentro de Stratio Rocket.

CTE

Una Common Table Expression (CTE) es una expresión temporal en SQL que se define dentro de una consulta, y se utiliza para estructurar y organizar mejor las consultas complejas. Pueden ser referenciadas varias veces en una consulta. Esto mejora la legibilidad y mantenibilidad de un código SQL, ya que evita repetir subconsultas y permite un diseño más modular (Bisso, 2022).

En este proyecto, se usan CTEs para mejorar la transformación de datos. Con ellas se logra estructurar las consultas de forma más clara y eficiente, facilitando la implementación del proceso de transformación de datos. Esto mejora la capacidad de manejar grandes volúmenes de datos y ejecutar análisis complejos de manera más efectiva.

Flexibilidad

En este contexto, la flexibilidad hace referencia a la capacidad de un sistema para adaptarse y responder a cambios en los requisitos de datos, estructuras de datos, y volúmenes de datos sin una necesidad significativa de rediseñar o reconfigurar el sistema. Se trata de una cualidad crucial que permite a las organizaciones adaptarse a nuevas necesidades comerciales. También permite integrar nuevas fuentes de datos y cambiar los procesos de transformación de datos fácilmente (Bilsdon, 2021).

Escalabilidad

La escalabilidad se refiere a la aptitud de un sistema para manejar un aumento en la carga de trabajo, volumen de datos o número de usuarios, sin comprometer el rendimiento. Un sistema escalable puede crecer para soportar más datos y mayor demanda de procesamiento (Stonefly, 2024).

Flujo de Trabajo End-to-End

Un flujo de trabajo end-to-end es un proceso integral que abarca todas las etapas necesarias para completar una tarea o proyecto, desde su inicio hasta su finalización. En él, se incluyen fases intermedias y se asegura que cada paso cumpla los objetivos establecidos. En este informe, cobra relevancia el concepto de flujo de trabajo end-to-end puesto que hace alusión al diagrama definido en Stratio Rocket, en donde se realizan las transformaciones de los datos.

Optimización de Procesos

En este marco, la optimización de procesos implica mejorar un flujo de trabajo para hacerlo más eficiente en términos de tiempo, recursos y resultados. El objetivo del proyecto es mejorar el flujo de trabajo para aumentar la eficiencia y la capacidad de análisis.

Metros Lineales

Se trata de la unidad de medida utilizada para cuantificar el espacio ocupado por los productos en los lineales de supermercados e hipermercados. Es la métrica principal utilizada por el cliente para analizar y optimizar la distribución de productos en tiendas.

Dentro de Lineal (DL)

Hace referencia al espacio que ocupan los productos en las estanterías convencionales dentro de un supermercado. Constituye una de las categorías clave analizadas para mejorar la disposición de productos.

Fuera de Lineal (FL)

Es el espacio que ocupan los productos en ubicaciones distintas a las estanterías convencionales, como stands de promoción y neveras. Resulta otra categoría importante para el análisis del rendimiento del cliente.

Foto de Éxito (FE)

En el contexto empresarial, la Foto de Éxito es una representación conceptual de los objetivos a alcanzar por una empresa en un periodo específico. En este proyecto la Foto de Éxito se mide en metros lineales de productos en supermercados. Es un indicador clave con el que se marca el estado idóneo para el cliente.

Datos Maestros

Son los datos esenciales y estables que describen las entidades principales de una organización, como clientes, productos, empleados, proveedores, y activos. Estos datos suelen cambiar con poca frecuencia y son fundamentales para las operaciones del negocio.

2.2. Importancia y Beneficios de los Procesos ETL en la Integración y Análisis de Datos

La integración de datos es el proceso en el que se combinan datos que provienen de diferentes fuentes para crear una visión unificada. En el panorama digital actual, las organizaciones recopilan una enorme cantidad de datos provenientes de muchos sistemas. Los datos pueden estar en varios formatos y estructuras, y las ETLs sirven para normalizarlos y estandarizarlos según unos requisitos (TapData, 2023).

Asimismo, el desarrollo de procesos ETL permite a las organizaciones realizar análisis significativos sobre los conjuntos de datos integrados. Transformando los datos extraídos a un formato consistente, adecuado para el análisis, las empresas son capaces de destapar conocimientos valiosos que impulsen procesos de toma de decisiones (TapData, 2023).

Existen herramientas especializadas en el desarrollo de ETLs que permiten racionalizar el proceso y hacerlo más eficiente. Estas herramientas tienen integradas funcionalidades como la extracción automática de datos de varias fuentes, la transformación de los datos y la carga en bases de datos destino.

Las herramientas de desarrollo ETL ahorran tiempo y esfuerzo comparado con el trabajo que resulta extraer y transformar datos de forma manual. Muchas herramientas, además de ser capaces de manejar grandes volúmenes de datos eficientemente, automatizan tareas repetitivas y ofrecen interfaces visuales que facilitan diseñar e implementar flujos de trabajo ETL.

Bajo el caso de estudio que se está tratando, las diferentes secciones de la ETL están divididas en distintas herramientas especializadas. Esto significa que, para cada etapa del proceso ETL, se utiliza una herramienta adecuada para esa tarea específica. Esta división no solo optimiza el flujo de trabajo, sino que también aprovecha las fortalezas de cada herramienta, resultando en un proceso más eficiente y efectivo en la integración de datos y en la generación de análisis significativos.

2.3. Métodos de Optimización de procesos ETL

Existen multitud de enfoques con los que mejorar la eficiencia y velocidad con la que los datos son extraidos, transformados y cargados en un base de datos destino. Dado que uno de los focos de este estudio va ligado con la optimización del proceso de transformación involucrado en una ETL, en este apartado se explican las principales técnicas y prácticas para mejorar la eficiencia de una ETL. Se presentan métodos específicos para optimizar las transformaciones de datos, ya que estas suelen constituir la mayor parte del tiempo de ejecución en los procesos ETL. Además, se abordarán estrategias para reducir los tiempos de ejecución, enfocándose en cómo las transformaciones eficientes pueden acelerar el proceso global de ETL.

2.3.1. Indexación y Optimización de Consultas

Una indexación adecuada de tablas puede mejorar considerablemente el rendimiento de las consultas SQL, reduciendo el tiempo necesario para recuperar datos. La indexación crea estructuras adicionales que ayudan al sistema de bases de datos a encontrar información más rápido. Es como un índice en un libro, que permite localizar rápidamente un tema específico. Se realiza sobre las columnas de las tablas que se consultan con mayor frecuencia, y sobre las usadas en las condiciones de unión entre tablas. Esto genera rutas de acceso más eficientes a los datos y minimiza la cantidad de datos que el sistema necesita escanear para encontrar los resultados deseados (TapData, 2023).

2.3.2. Particionamiento

El particionamiento es una técnica en la que se dividen tablas grandes en otras más pequeñas y manejables, llamadas particiones. Esta técnica mejora la escalabilidad y eficiencia de los procesos ETL. Dividir una tabla grande en particiones permite procesar diferentes partes simultáneamente. Esto mejora la eficiencia de las consultas al eliminar la necesidad de escanear toda la tabla; en su lugar, el sistema puede escanear solo las particiones relevantes para una consulta específica (TapData, 2023).

2.3.3. Common Table Expressions (CTEs)

Las Common Table Expressions (CTE) son una herramienta poderosa en SQL que mejora la legibilidad y mantenibilidad de las consultas complejas, lo cual es crucial para la optimización de procesos de transformación de datos. Las CTEs crean resultados temporales que se pueden usar en la misma consulta. Así se evita repetir subconsultas, logrando un diseño más modular y claro de las consultas SQL. Al usar CTEs, también es posible juntar varios pasos de transformación en una sola consulta. Esto reduce las etapas en el flujo de trabajo y minimiza tanto el tiempo de procesamiento como la complejidad del código. Las CTEs también son útiles para manejar datos temporales necesarios solo durante el proceso de transformación, lo que reduce la necesidad de crear tablas temporales adicionales y simplifica el proceso de limpieza y gestión de datos (WELCH, 2024).

Capítulo 3

Metodología

Esta sección presenta uno de los pilares de este trabajo, el análisis completo del flujo de trabajo end-to-end. Para ello, se detalla y describe el caso de estudio que estamos tratando, además de todas las herramientas involucradas en el proceso ETL.

3.1. Descripción del Caso de Estudio

Resulta muy relevante el caso de estudio del proyecto puesto que, comprenderlo a fondo, facilita el esquematizar, diseñar y entender conceptualmente los procesos de transformación de datos. En este apartado se exploran los detalles específicos del caso y el contexto general del canal de alimentación. Igualmente, la relevancia de los datos tratados se destaca a través del impacto que tienen en las estrategias comerciales del cliente. Así se ofrece una visión completa de cómo los datos auditados informan decisiones clave.

3.1.1. Descripción del contexto de los productos en el canal de alimentación

Durante las últimas décadas se ha observado un notable incremento en la diversidad y cantidad de productos disponibles para los consumidores. Esto influye en sus decisiones de compra y hábitos alimenticios. Los productores y distribuidores necesitan evolucionar constantemente sus estrategias para satisfacer a un mercado más exigente e informado. El canal de alimentación es un sector dinámico y competitivo. Los supermercados como punto de venta principal son cruciales en la cadena de distribución e influyen en las decisiones de compra de los consumidores. La disposición de productos en estantes y otros espacios es esencial para captar la atención del consumidor y fomentar la compra (Hustleescape, 2019).

Este canal se caracteriza por tener una alta rotación de productos y constantes novedades. Los supermercados optan por maximizar el uso del espacio disponible para mostrar muchos productos. Esto causa una competencia intensa por los metros lineales y el espacio que se ocupa en las tiendas. La ubicación de los productos en el lineal y su accesibilidad son factores cruciales que influyen directamente en las ventas.

El comportamiento del consumidor en los supermercados es crucial para entender la importancia de la ubicación de los productos. Muchas decisiones de compra ocurren en el lugar de venta y la disposición de los productos puede afectar en gran medida. Las áreas

fuera de lineal también son importantes para promover productos específicos como aquellos en oferta o recién lanzados.

Para optimizar la ocupación de metros lineales las empresas desarrollan estrategias de distribución basadas en datos auditados y análisis de mercado. Estas buscan equilibrar la presencia en estanterías, conjunto a la colocación en stands de promoción y neveras, para maximizar la visibilidad del producto y potenciar las ventas. La auditoría permite a las empresas verificar si se cumplen los acuerdos comerciales con los supermercados y ajustar sus estrategias.

3.1.2. Detalle del Caso de Estudio y su Relevancia

Los datos tratados en este proceso son de extrema relevancia; reflejan la presencia y distribución de los productos que ofrece el cliente dentro de una enorme cantidad de supermercados e hipermercados. Se consideran los metros lineales ocupados dentro y fuera de las estanterías. Personal especializado verifica estos datos con auditorías in situ para asegurar la precisión de las mediciones reportadas.

Además de los datos auditados actuales, son relevantes los del año anterior. Compararlos ayuda a identificar tendencias y variaciones importantes en la ocupación de metros lineales y ofrece una visión clara del desempeño del cliente en el mercado. También se consideran los datos relacionados con la Foto de Éxito, que muestran la cantidad de metros lineales ideal que el cliente quiere alcanzar. El escenario de mínimos es otra métrica clave en este análisis. Establece el umbral mínimo de metros lineales que se quiere mantener. Alcanzar este umbral es esencial para asegurar un desempeño competitivo en el mercado.

La integración y el análisis de estos datos permiten al cliente evaluar su rendimiento pasado y presente, proyectar escenarios futuros y definir estrategias para optimizar la ocupación del espacio en los supermercados. Así el cliente puede asegurarse de que se toman las decisiones correctas para mejorar su presencia en el mercado y aumentar su rentabilidad.

3.2. Herramientas y Tecnologías Utilizadas

En el presente trabajo se utiliza Stratio para construir, analizar y optimizar el flujo de trabajo. En concreto se utilizan sus funcionalidades Rocket y Discovery. En esta sección,, se muestra una descripción detallada de estas herramientas y sus características junto con ventajas y desventajas. Asímismo, también se va a discutir y explicar el funcionamiento del resto de herramientas involucradas en la ETL completa; Postgres y Tableau. Igualmente, se discute su relevancia en la gestión de datos masivos.

3.2.1. Stratio Rocket

Stratio Rocket es una plataforma que facilita la colaboración en proyectos de Big Data. Es útil para usuarios con distintos niveles de conocimientos técnicos: científicos de datos, ingenieros, usuarios empresariales y gerentes de proyectos.

Una de las principales características de la plataforma es su herramienta visual, que permite a los usuarios crear aplicaciones de Big Data sin necesidad de escribir código. Esto es especialmente útil para usuarios sin conocimientos técnicos avanzados (Stratio, 2024).

Adicionalmente, con esta herramienta, los usuarios pueden personalizar los componentes de un flujo de trabajo según las necesidades del proyecto, lo que permite más flexibilidad y ofrece una mayor capacidad de adaptación (Stratio, 2024). En este trabajo, se aprovechó la cualidad de personalización para ajustar el flujo de trabajo al lenguaje SQL y a las particularidades del proceso ETL.

Stratio es compatible con una variedad de tecnologías y motores distribuidos, incluyendo Flink, Kafka Streams, Elasticsearch, MongoDB, Redis, y otros conectores SQL y NoSQL. La herramienta no solo posee una gran flexibilidad y adaptabilidad para la transformación de datos, sino que también demuestra ser capaz de adaptarse a diferentes entornos de datos (Stratio, 2024).

Asimismo, Stratio Rocket incluye funcionalidades avanzadas de seguridad y gobernanza de datos, asegurando que los datos sean gestionados de manera segura y conforme a las normativas aplicables (Stratio, 2024).

Esta herramienta utiliza Apache Spark como motor principal para el procesamiento de datos en tiempo real y por lotes. La plataforma soporta la definición y despliegue de flujos de trabajo tanto en streaming (datos en tiempo real, nada más lleguen) como en batch (datos por lotes). De esta manera, se facilita la creación de canalizaciones de datos con ambos tipos de procesamientos (Stratio, 2024).

Ventajas

Las importantes ventajas que ofrece esta herramienta son un fácil uso, flexibilidad, escalabilidad y la gestión integral de los datos. La interfaz visual intuitiva de la que dispone esta herramienta hace que sea fácil de utilizar. Con esta característica se reduce la barrera de entrada y acelera el desarrollo de aplicaciones en el análisis de datos. La capacidad de personalización y la compatibilidad con varias tecnologías permiten adaptar la plataforma a diferentes necesidades y entornos de datos, proporcionando flexibilidad y escalabilidad. Además, dadas las capacidades de seguridad y gobernanza de datos que ofrece Rocket, se proporciona una gestión integral y segura de los datos.

Desventajas

Como desventaja está la dependencia de las tecnologías subyacentes. Stratio necesita el rendimiento y estabilidad de Apache Spark y otros motores distribuidos. Problemas en estos pueden afectar su rendimiento. Por otra parte, aunque la interfaz visual es intuitiva, usar algunas herramientas avanzadas requiere más conocimiento, especialmente para personalizar componentes e integrar con otras tecnologías.

3.2.2. Stratio Discovery

Stratio Discovery es otra herramienta clave dentro de la plataforma de Stratio; está diseñada para ofrecer capacidades avanzadas de análisis de datos. El programa se conecta

a diversas fuentes de datos, estructurados y no estructurados, y permite a los usuarios explorar y analizar datos de varias fuentes fácilmente. De esta forma, proporciona una vista coherente de la información, ayudando a identificar patrones y obtener información valiosa. La herramienta también incluye capacidades avanzadas de visualización para crear *dashboards* interactivos y reportes detallados (Stratio, 2024).

Ventajas

Las ventajas destacables de Stratio Discovery incluyen la habilidad de conectar y unificar datos de varias fuentes, y mostrar estos datos de forma clara y comprensible con visualizaciones avanzadas. Estas funciones mejoran mucho la calidad del análisis y hacen más fácil interpretar los resultados. Al mismo tiempo, mejorando el análisis de datos y facilitando la extracción de información valiosa, Stratio Discovery se convierte en un aliado interesante para las empresas que buscan optimizar su toma de decisiones.

Desventajas

Stratio Discovery no presenta desventajas claras, dado que se trata de una funcionalidad añadida que enriquece significativamente el ecosistema de Stratio.

3.2.3. PostgreSQL

PostgreSQL, o Postgres, es un sistema de bases de datos relacional de código abierto perfeccionado durante más de 35 años. La arquitectura confiable de la que dispone, la integridad de los datos que gestiona, además de sus características robustas, hacen de PostgreSQL una de las bases de datos con mejor reputación (PostgreSQL, 2024). Es por ello que se trata de uno de los software de bases de datos que más se utiliza, estando presente en una gran cantidad de sectores y de aplicaciones críticas.

En el contexto del proyecto, PostgreSQL se utiliza como Data Warehouse. Esta herramienta es ideal porque maneja grandes volúmenes de datos y es compatible con SQL, simplificando la integración de procesos ETL. PostgreSQL soporta muchos tipos de datos y operaciones, permitiendo una gestión flexible y eficiente de los datos (PostgreSQL, 2024).

Ventajas

Entre las ventajas de PostgreSQL están su naturaleza de código abierto, por lo que es una aplicación gratuita. Tiene una comunidad activa que contribuye continuamente a su desarrollo y mejora. Es posible definir tipos de datos personalizados y funciones sin recompilar la base de datos. Además, se puede programar en diversos lenguajes. PostgreSQL cumple con los principios ACID, lo que asegura transacciones confiables y seguras. También es altamente escalable, capaz de manejar grandes volúmenes de datos y muchos usuarios concurrentes (PostgreSQL, 2024). Esto la hace ideal para aplicaciones críticas que necesitan alta disponibilidad y rendimiento.

Desventajas

La única desventaja remarcable de esta herramienta es que puede no ser eficiente en entornos de escritorio o en aplicaciones con requisitos de bajo rendimiento, ya que su uso

se ha focalizado para grandes aplicaciones y entornos de servidor.

3.2.4. Tableau

Tableau es una herramienta líder en el ámbito de la visualización de datos y la inteligencia empresarial, utilizada ampliamente para convertir datos en gráficos interactivos y comprensibles. Gracias a su interfaz intuitiva, la herramienta es capaz de manejar grandes volúmenes de datos y crear visualizaciones detalladas y sofisticadas rápidamente. Esto facilita a los usuarios sin conocimientos técnicos profundos crear y explorar datos de manera efectiva (Tableau, 2024).

Ventajas

Tableau es fácil de utilizar, permitiendo a usuarios sin experiencia en programación aprender rápidamente a utilizar la herramienta. De la misma manera, su capacidad para conectarse a múltiples fuentes de datos mejora la calidad del análisis y permite la creación de dashboards unificados y completos. Adicionalmente, Tableau permite la integración con lenguajes como Python y R para realizar cálculos complejos y mejorar el rendimiento mediante el uso de paquetes externos (Tableau, 2024).

Desventajas

Una de las principales limitaciones es su alto costo, que puede ser prohibitivo para pequeñas y medianas empresas. También carece de ciertas funcionalidades avanzadas de preparación de datos, lo que a menudo requiere que los usuarios realicen estas tareas en herramientas adicionales antes de importar los datos a Tableau. Además, la herramienta tiene restricciones en la personalización de visualizaciones y programación de informes, lo que puede ser un inconveniente para usuarios avanzados que requieren un alto grado de control sobre sus visualizaciones.

3.3. Implementación de las Herramientas y Tecnologías Utilizadas

En este segmento del capítulo se discuten los roles principales de las tecnologías del proyecto y su integración. Para comenzar, las tablas que proporciona el cliente, y que alimentan el flujo de trabajo, están almacenadas en diversas bases de datos. Son tablas que se generan apuntando a esas bases de datos. Las tablas llegan a Stratio y, con la funcionalidad de Discovery, es posible visualizarlas. Estas tablas muestran información sobre los metros lineales ocupados por los productos que ofrece el cliente en los puntos de venta. La función de Discovery es esencial porque permite ver el contenido de cada tabla y analizarlas a fondo. Esto ayuda a diseñar una estrategia óptima para transformar, cruzar los datos y ofrecer una visión unificada de ellos.

Una vez comprendida la información que va a intervenir en el proceso, Stratio Rocket permite diseñar el flujo de trabajo completo. Este diseño se realiza utilizando lenguaje SQL. Rocket proporciona una interfaz visual para crear un esquema del flujo de trabajo con secciones claras: entradas al flujo, transformaciones y almacenamiento. La tabla resultante se almacena en PostgreSQL. El cliente, posteriormente, conecta esta tabla final

a Tableau, donde podrá crear cuadros de mando y realizar análisis para mejorar su inteligencia empresarial.

La combinación de Discovery y Rocket permite un manejo eficiente de los datos desde su visualización y análisis hasta su transformación y almacenamiento final. Este enfoque asegura que el cliente obtiene una visión completa y coherente de los datos. Entonces, el flujo de trabajo es un paso crucial para optimizar la gestión de los metros lineales de sus productos en los puntos de venta. La Figura 3.1 muestra esquemáticamente el diseño del proceso ETL, mostrando la circulación de datos entre las diferentes herramientas involucradas.

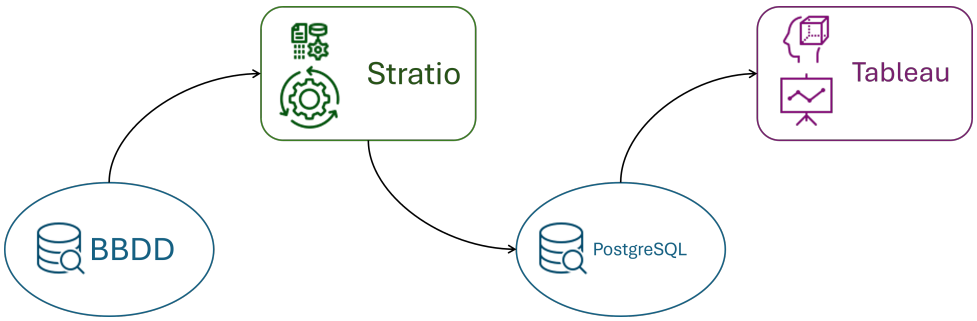


Figura 3.1: Flujo de Trabajo entre las Herramientas

Capítulo 4

Descripción y Funcionamiento del Flujo de Trabajo Original

En esta sección se discute el diseño del flujo de trabajo original para dar una visión general de las transformaciones que se realizan sobre los datos. Asímismo, se efectúa un análisis profundo sobre este proceso definido en Stratio Rocket, incluyendo todos los pasos involucrados, y se discute la idoneidad de cada etapa. La Figura 4.1 muestra el flujo de trabajo original. Me voy a apoyar en esta imagen para explicar el flujo y la lógica detrás de él.

4.1. Entradas al Flujo de Trabajo

Como se puede observar en la Figura 4.1, el proceso cuenta con 12 entradas. Las entradas se dividen en dos secciones: la información de dentro y la de fuera de lineal. A su vez, las dos secciones están fraccionadas en 3 partes diseñadas de la misma manera. Estas partes son las entradas relacionadas con los datos auditados de metros lineales, las entradas que tienen que ver con los datos de Foto de Éxito, y las entradas en las que intervienen los datos del escenario de mínimos.

En esta sección del informe se adjunta una tabla que ayuda a clarificar la estructura de las entradas al flujo de trabajo original. La Tabla 4.1 está dividida según se fracciona el flujo de trabajo. De este modo, cada fila muestra un par de entradas al flujo de trabajo, los datos que se obtienen según esas entradas, y las variables más importantes que se consiguen con esas entradas.

La lógica detrás de la implementación de las entradas es la siguiente:

- **Datos auditados dentro de lineal:** Con `ANIO_MAX` se obtiene el año máximo para el que hay dato de metros lineales de productos dentro del lineal, filtrado por una ubicación. El filtro se aplica dados los requerimientos del cliente. A través del Listado B.1, que incluye el código anonimizado de `ANIO_MAX`, se aporta una visión de cómo se consigue el año máximo. Mediante el uso de `AUDIT_DL` se consiguen todos los datos que hay acerca de los productos que vende el cliente dentro del lineal. El código de `AUDIT_DL` viene anonimizado en el Listado B.2.
- **Datos de Foto de Éxito dentro de lineal:** `ANIO_MAX_FE_DL` consigue el año máximo para el que hay datos de los metros lineales de productos en el escenario de

Foto de Éxito y en el espacio dentro del lineal. `FOTO_EXITO_DL` adquiere, de forma directa, toda la información involucrada del escenario ideal de los metros lineales para los productos dentro de lineal. Son códigos diseñados de manera muy similar a las anteriores entradas.

- **Datos de escenario de mínimos dentro de lineal:** `ANIO_MAX_FE_MIN_DL` obtiene el último año para el que existen datos del mínimo de metros lineales al que se tiene que llegar en el espacio de dentro de lineal. `FOTO_EXITO_MIN_FL` proporciona directamente toda la información acerca del escenario de mínimos para los productos dentro de lineal. De nuevo, estas entradas se asemejan mucho en diseño a las entradas mencionadas anteriormente.

La sección de fuera de lineal sigue exactamente la misma lógica y estructura conceptual. Las únicas diferencias son que, claro está, intervienen los datos de fuera de lineal en vez de los de dentro de lineal. También, que en `ANIO_MAX_FL` se filtran los productos según unos determinados elementos que ocupan en el espacio de fuera de lineal y por unas ubicaciones diferentes a la filtrada en `ANIO_MAX_DL`.

Desde un primer instánte, me ha parecido remarcable el numero de entradas al flujo y sentía que estas se podían reducir, para así poder tener un flujo de trabajo más claro y eficiente. Sobretodo, me desconcertaban las entradas que aportaban el año máximo para el que hay datos. Uno de los primeros pasos a la hora de optimizar el flujo de trabajo fue reducir esa complejidad añadida en las entradas, ya bien agrupando los años máximos en una sola etapa (que debería ser el mismo en todas las tablas de entrada al flujo), o juntando varias etapas del principio del flujo entre sí.

4.2. Continuación del Flujo de trabajo

La siguiente sección del flujo es una etapa de filtrado de las entradas, donde los pares de entradas se combinan y cruzan. De este modo, se obtiene toda la información relevante al año máximo para el que hay dato. A su vez, el diseño del filtro permite obtener también toda la información para el año anterior al máximo. A modo de ejemplo, `ANIO_MAX_FL` y `AUDIT_FL` entran a `AUDIT_FL_FILTRO`, y la salida del filtro es toda la información auditada de los productos fuera del lineal, para el último año que hay dato y para el año anterior. El funcionamiento de los filtros es igual para cada par de entradas. De nuevo, es remarcable destacar que se podría reducir facilmente la complejidad de esta etapa, ya que se utiliza exactamente el mismo código para cruzar cada par de entradas.

Posteriormente, la salida de los filtros alimenta una fase en la que se quiere conseguir un mayor detalle de los datos. La granularidad se obtiene cruzando las salidas de los filtros con unos ficheros de datos maestros; maestros de productos y puntos de venta que proporciona el cliente. Para las transformaciones `PRINCIPAL_DL` y `DETALLE_AA_DL`, se filtran los datos por una determinada ubicación. De la misma manera, para `PRINCIPAL_FE_FL` y `DETALLE_AA_FL` se depuran los datos para solo obtener los involucrados en ciertos lugares de España y ubicaciones dentro de las tiendas. El tramo de detalle es coherente y su diseño es correcto. A partir de esta sección de detalle, los datos poseen la misma estructura que la tabla final. Es decir, las salidas de esta etapa tienen exatamente las mismas columnas que la tabla final.

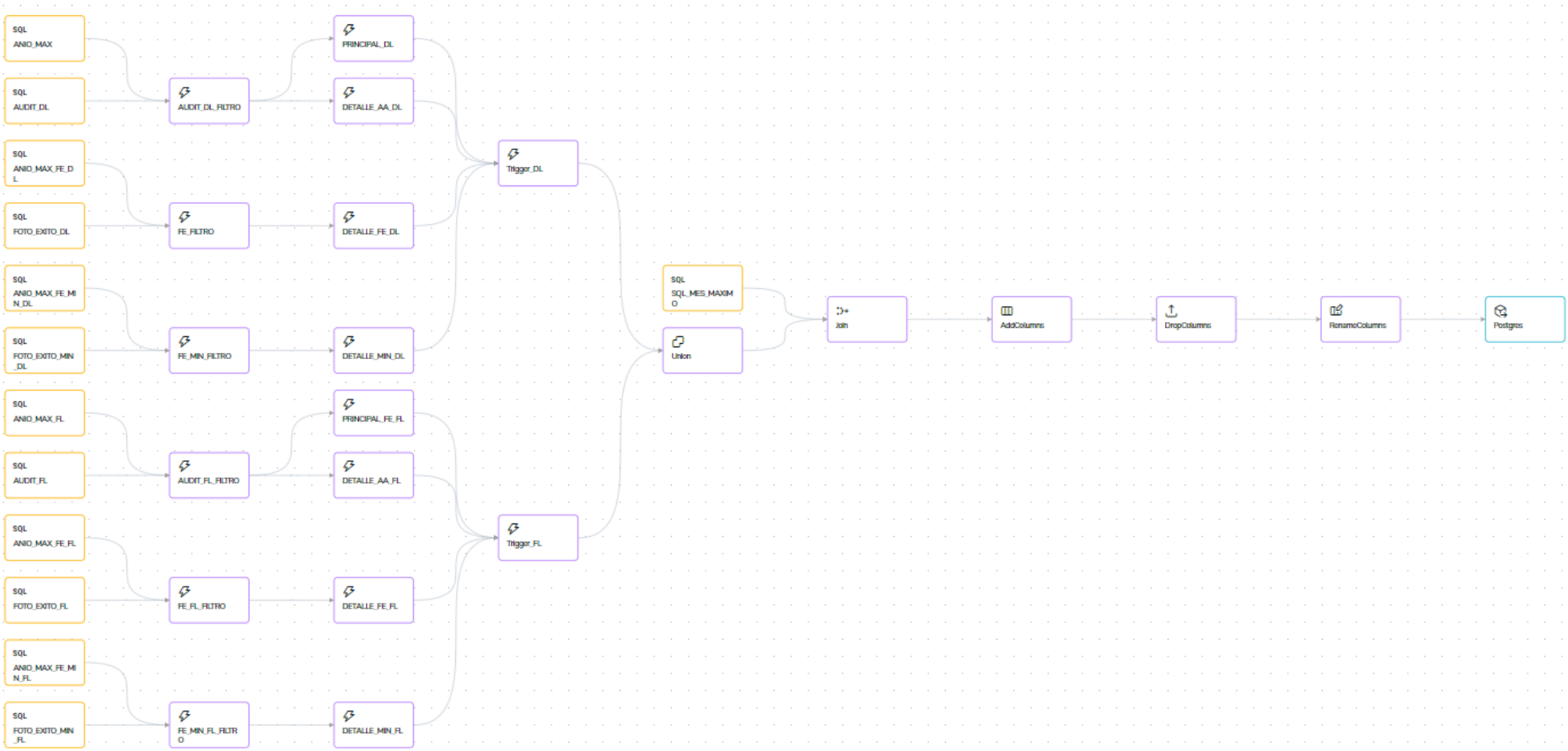


Figura 4.1: Flujo de Trabajo Original

Entradas	Datos	Variables Principales
ANIO_MAX AUDIT_DL	Datos auditados de productos dentro de lineal en los puntos de venta	Metros lineales de productos
ANIO_MAX_FE_DL FOTO_EXITO_DL	Datos del escenario ideal de los productos dentro de lineal en los puntos de venta	Foto de éxito de los metros lineales de productos Retorno de los productos de Foto de éxito Metros lineales de diciembre del año anterior
ANIO_MAX_FE_MIN_DL FOTO_EXITO_MIN_DL	Datos del espacio mínimo a alcanzar por los productos dentro de lineal en los puntos de venta	Metros lineales mínimos de los productos Retorno mínimo de los productos
ANIO_MAX_FL AUDIT_FL	Datos auditados de productos fuera de lineal en los puntos de venta	Metros lineales de productos
ANIO_MAX_FE_FL FOTO_EXITO_FL	Datos del escenario ideal de los productos fuera de lineal en los puntos de venta	Foto de éxito de los metros lineales de productos Metros lineales de diciembre del año anterior
ANIO_MAX_FE_MIN_FL FOTO_EXITO_MIN_FL	Datos del espacio mínimo a alcanzar por los productos fuera de lineal en los puntos de venta	Metros lineales mínimos de los productos Retorno mínimo de los productos

Cuadro 4.1: Entradas al Flujo de Trabajo

La estructura de la tabla final es fija puesto que no hay acceso a donde se almacena la tabla final en PostgreSQL, y la manera en la que se cargan nuevos datos es sobrescribiendo la tabla final cada vez que se ejecuta el proceso. Tener los datos con la estructura de la tabla final facilita en buena medida las transformaciones siguientes. Para asegurar que las salidas de las etapas de detalle tienen esa estructura, se inicializan a 0 las columnas que faltan.

Una vez pasada la fase de detalle, se combinan todas las partes correspondientes a DL y a FL mediante el uso de `TRIGGER_DL` y `TRIGGER_FL`. El objetivo de estos códigos es consolidar los diferentes datos en un solo conjunto de resultados. En el caso de `TRIGGER_DL`, se combinan las tablas `PRINCIPAL_DL`, `DETALLE_FE_DL`, `DETALLE_MIN_DL` y `DETALLE_AA_DL`. Esta consolidación se realiza a través de una serie de consultas `SELECT` y cruces `LEFT JOIN`, seguidas de operaciones `UNION` para unir los resultados de múltiples consultas. Las uniones y cruces se utilizan para garantizar que todas las posibles combinaciones de datos relevantes se incluyen en el conjunto de resultados finales. Hay un ejemplo anonimizado de este código presente en el Listado B.5, con el que se aporta una noción de la estructura del código. `TRIGGER_FL` sigue exactamente la misma lógica.

Se va a estudiar en gran medida la optimización de esta etapa, ya que es la que más tiempo tarda en ejecutarse debido a un gran número de `LEFT JOIN` y `UNION` presentes en los códigos. De nuevo, desde el momento que se tuvo acceso al código, y se visualizó esta fase del flujo de trabajo, parecía altamente optimizable con comandos como `CROSS JOIN` o `FULL JOIN`.

De nuevo, todas las entradas a esta sección tienen la misma estructura, las mismas 31 columnas, que es la estructura que posee la tabla final. Sin embargo, cada tabla entrante tiene diferentes columnas compuestas con datos específicos. Algunas columnas están integradas en todas las tablas con la misma información, como por ejemplo el año, el mes o la marca del producto. Esto garantiza que existan los mismos registros en todas las tablas que alimentan esta fase.

Hay otras columnas que no tienen el mismo dato en todas las tablas entrantes a la etapa de *Triggers*. Estas columnas son las que poseen las métricas de metros lineales de los productos, que incluyen las variables principales presentes en el Cuadro 4.1. Por ejemplo, una observación de la salida de `PRINCIPAL_DL` tendrá un valor en la columna `ML_DL`, que indica los metros lineales ocupados por los productos dentro del lineal, mientras que el resto de las métricas y columnas relacionadas con el espacio estarán inicializadas a 0. En cambio, una observación de `DETALLE_AA_DL` tendrá un valor en la columna `ML_DL_AA` (metros lineales ocupados por los productos en el lineal del año anterior), y el resto de las métricas estarán a 0. Por supuesto, la salida de `TRIGGER_DL` tendrá todas las columnas que tengan que ver con métricas de FL a 0, y viceversa.

Al combinar estas diferentes tablas, se pretende ofrecer una visión integral del espacio que ocupa cada producto en cada tienda. La estabilización de los datos permite obtener una imagen completa y detallada de cómo se distribuyen los productos en el espacio disponible.

Una vez llegados a este punto, `TRIGGER_DL` y `TRIGGER_FL` se unen directamente para

conseguir la tabla completa con toda la información. Asímismo, se realizan una serie de transformaciones para manejar determinados casos especiales. Destaca la generación de una consulta, nombrada en el flujo de trabajo como `SQL_MES_MAXIMO`, con la que se obtiene el mes máximo, del año máximo, para el que hay datos en las tablas de origen. Esto se implementa para limitar el número de meses de datos del año anterior, ya que debería haber la misma cantidad de meses que para el año en curso. Si para este año hay datos hasta junio (junio incluido), se garantiza que hayan datos del año anterior solo para los 6 primeros meses. Basicamente, la anterior transformación facilita la integración de los datos del año anterior en Tableau. Permite realizar visualizaciones comparativas coherentes entre datos del año en curso y el anterior. Las demás transformaciones sirven para manejar valores nulos e incorporar, eliminar y renombrar unas columnas en concreto. Después de realizar las transformaciones finales, el flujo de trabajo llega a su fin, y la tabla final es almacenada en PostgreSQL.

4.3. Tabla Final Generada por el Flujo Original

Para garantizar que se tratan los datos de manera acorde, y que se consiguen los resultados que se deberían, realicé varias consultas sobre la tabla final que crea este flujo de trabajo.

En un primer instánte, se quiso cerciorar que los resultados totales de las métricas que ofrece esta tabla final coinciden con los resultados finales obtenidos de las tablas de origen. Para ello, se visualizó la suma total de cada una de las métricas involucradas; la suma total de las métricas obtenida de los ficheros de origen y de la tabla final. Sorprendentemente, resulta que la cantidad total de cuatro de las métricas de la tabla final no tienen el valor que se debería conseguir. Por ende, me comprometí a asegurar que la tabla final, del flujo de trabajo optimizado, ofreciese visiones totales acumuladas precisas, tomando el valor que se debería obtener.

Pude sacar más reseñas a la tabla final. Los registros no se unifican correctamente. Buscando el contenido de determinados registros pude verificar que, con este diseño de flujo de trabajo, se obtienen dos observaciones, cuando en realidad solo debería aparecer una. Una de las dos observaciones que se conseguen tiene los datos de DL, y otra los de FL. De nuevo, pretendo corregir esta irregularidad y ofrecer resultados unificados correctamente. Es por ello que, en este punto, se estableció como un nuevo objetivo alcanezar una tabla final donde cada registro tenga una sola observación, con todos los datos referentes.

Capítulo 5

Optimización del Flujo de Trabajo

En está sección se discuten los cambios implantados en el flujo de trabajo original, y el impacto que tienen en la eficiencia y ejecución del proceso completo. Como se ha discutido en el anterior capítulo, los cambios no se deben quedar solo en optimizar el tiempo de ejecución ni la forma del flujo de trabajo, sino que se debe de asegurar que la estructura y los resultados de la tabla final son correctos y consistentes.

5.1. Cambios Realizados

5.1.1. Entradas al nuevo flujo de trabajo

Desde un primer instante, me parecía que el flujo de trabajo contaba con muchas etapas iniciales. Es decir, tenía la impresión de que podían combinarse para así reducir el número de cajas involucradas en el diagrama del proceso. De esta manera, se atendería al objetivo de optimizar el flujo de trabajo en términos de forma, reduciendo el número de etapas iniciales. Esto se logra combinando la extracción de datos con el posterior filtro, y con la etapa de detalle presente en el flujo original. De esta manera, la salida de las entradas del nuevo flujo tendría la estructura de la tabla final. Esta idea aporta un diseño más apropiado y fácil de entender conceptualmente.

Para conseguir la nueva estructura simplificada, se ha hecho uso de CTEs en cada una de las nuevas entradas. Mediante las CTEs se obtiene el año máximo para el que hay dato, seguido por todos los datos auditados filtrados por el año máximo y el año anterior al máximo. En el Listado A.1 hay un ejemplo anonimizado del código presente en una de las entradas, `PRINCIPAL_DL`. El código es muy parecido al que hay en el resto de entradas. Dependiendo de la entrada, se ha añadido una CTE para calcular el mes máximo, del año máximo, para el que hay dato. Es el caso, por ejemplo, de la entrada `DETALLE_AA_DL`. En los datos oiginales están los datos para todos los meses del año anterior. Si se cogen solo los datos hasta el mes máximo del último año, la etapa `SQL_MES_MÁXIMO` se vuelve prescindible, y se consigue simplificar en mayor medida el tamaño del diagrama del flujo.

Una vez conseguidos todos los datos iniciales, se realiza una consulta sobre las CTEs para aplicar la etapa de detalle que hay en el flujo original. Atendiendo a las Figuras 5.1 y 5.2 se puede observar de forma clara el cambio realizado sobre el flujo. Se ha reducido en gran medida la complejidad en términos de forma; toda la estructura presente en las etapas iniciales del flujo original, que incluye las fases contenidas en la Figura 5.2, se

reduce a lo que se puede observar en la Figura 5.1.

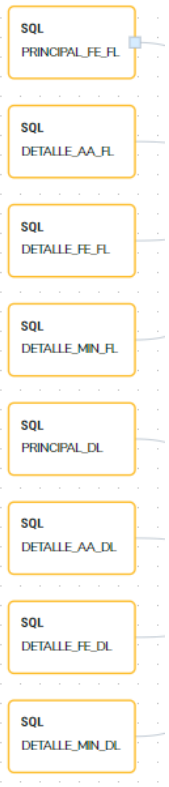


Figura 5.1: Entradas Nuevas al Flujo de trabajo

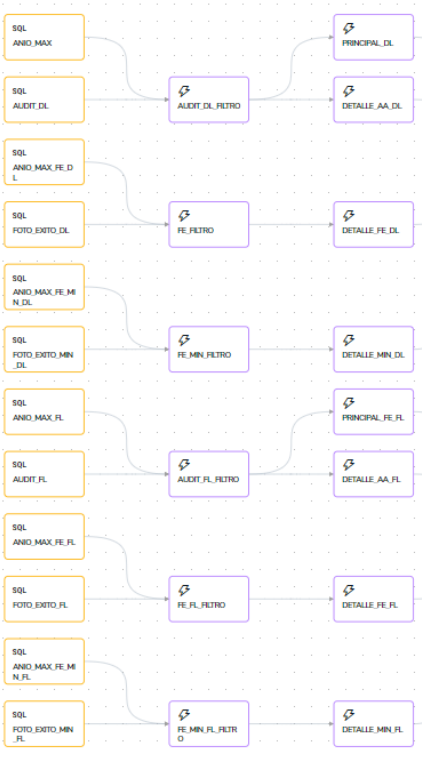


Figura 5.2: Entradas Originales al Flujo de trabajo

Para asegurar que las salidas de las nuevas entradas son consistentes con los datos de origen, se realizaron varias consultas SQL en Discovery. Para ello, se calculó la suma total de las métricas involucradas en el flujo de trabajo, tanto a partir de las tablas que alimentan al flujo, como a partir de las nuevas entradas del flujo. De esta forma, se logró comparar los resultados de las salidas de todas las entradas con los resultados obtenidos de los datos de origen. Así, se pudo validar y verificar que las salidas de las entradas ofrecen datos correctos y coherentes. Por aclarar, la suma de los metros lineales que hay a la salida de, por ejemplo `PRINCIPAL_DL`, es igual que la conseguida a partir de la tabla de origen pertinente.

Beneficios del cambio

A continuación, se comentan los principales beneficios de haber aplicado los cambios en las entradas al flujo.

- **Reducción de la Complejidad:** Al combinar las CTEs, se reduce la cantidad de operaciones intermedias, lo que simplifica la lógica de las consultas de entrada.
- **Eficiencia mejorada:** Con el nuevo diseño aportado a las entradas, se aplican filtros de manera temprana y se combinan operaciones, permitiendo que el motor de la base de datos optimice mejor la ejecución y reduciendo el tiempo de procesamiento.
- **Mantenimiento de la integridad de los datos:** Las consultas de entrada siguen pasos lógicos parecidos a los originales, asegurando que los resultados obtenidos sean los mismos que con la consulta original. Los resultados a la salida de la Figura 5.2 son correctos, al igual que los de la Figura 5.1.

5.1.2. Transformaciones tipo *Trigger*

Las transformaciones de tipo *Trigger* son donde se unifican y cruzan todos los datos referentes a DL y a FL, por separado, en el flujo de trabajo original. De este modo, ocupan las cajas nombradas como `TRIGGER_DL` y `TRIGGER_FL`, respectivamente. Las consultas utilizadas para realizar estas transformaciones resultan muy complejas, confusas y desordenadas. Es por ello que se optó por rediseñar estas etapas.

Visión general de `TRIGGER_DL` y `TRIGGER_FL`

Las consultas originales son muy complejas e involucran varias operaciones `UNION` junto a multitud de declaraciones `SELECT`. Intentan juntar y filtrar datos provenientes de múltiples tablas intermedias (`PRINCIPAL_DL`, `DETALLE_FE_DL`, `DETALLE_MIN_DL`, `DETALLE_AA_DL`, `PRINCIPAL_FE_FL`, `DETALLE_FE_FL`, `DETALLE_MIN_FL` y `DETALLE_AA_FL`). Las consultas utilizan `LEFT JOIN` y `LTRIM(RTRIM(...))` para cruzar los datos y controlar los espacios iniciales y finales en las columnas por las que se realiza el cruce.

Problemas con la consulta original

Seguidamente, se van a discutir los principales problemas que introducen `TRIGGER_DL` y `TRIGGER_FL`.

- **Complejidad y rendimiento:** La consulta resulta ser muy compleja debido a la numerosa cantidad de operaciones `UNION` y `LEFT JOIN` que se introducen. La gran cantidad de operaciones `UNION` hace que las tablas involucradas se escaneen varias veces, produciendo una consulta ineficiente.
- **Redundancia:** Las operaciones repetitivas de `LTRIM(RTRIM(...))` indican que pueden haber problemas subyacentes en relación a la calidad de los datos, como el uso inconsistente de los espacios. Si bien pueden ser necesarios, aumentan la complejdad de las consultas.
- **Errores potenciales:** La lógica compleja que se ha empleado, junto a las multiples operaciones de `UNION`, aumentan las posibilidades de generar erroes lógicos. Adicionalmente, deficultan el mantenimiento y la depuración de las consultas.

Visión general de `Trigger`

La nueva consulta simplifica el enfoque original utilizando `FULL OUTER JOIN` y `COALESCE`. En el Listado A.2, se muestra una visión general, anonimizada, del código dentro de la etapa de `Trigger`. Seguidamente, se detalla un desglose paso a paso de cómo la nueva consulta es mejor que la original.

- **Uso de `FULL OUTER JOIN`:** `Trigger` emplea `FULL OUTER JOIN` para combinar todas las tablas relevantes en una sola operación. Esto asegura la inclusión de todas las filas de las tablas que se cruzan, tanto si tienen filas coincidentes como si no las tienen.

  Al utilizar `COALESCE`, la consulta certifica que si el valor en una columna para un registro es `NULL` en una de las tablas, se coge el valor de otra tabla, fusionando efectivamente los datos.
- **Simplificación y legibilidad:** Reduciendo el número de operaciones `UNION` que se aplican, la consulta se vuelve más legible y mantenible. Adicionalmente, usar `COALESCE` para manejar valores `NULL`, y `FULL OUTER JOIN` para incluir todas las filas relevantes, simplifica enormemente la lógica.
- **Mejora del rendimiento:** La consulta realiza menos escaneos sobre las tablas de entrada. Esto se logra a partir de consolidar la lógica en un único conjunto de uniones, en lugar de manejar multiples operaciones `UNION`. Así pues, se logra una mejora significativa en el rendimiento de la consulta, especialmente si se toma en cuenta la gran cantidad de datos que se tratan.
- **Consistencia de los datos:** El uso de `COALESCE` asegura resultados consistentes. El comando coge el primer valor no `NULL` de entre todas las tablas intermedias para una columna numérica, ayudando a fusionar los datos entre las diferentes tablas.

Componentes clave de `Trigger`

A modo de resumen, se incluye la Tabla 5.1 para presentar y explicar los componentes que hacen de `Trigger` una mejor implementación que `TRIGGER_DL` y `TRIGGER_FL`.

Componente	**¿Qué hace?**	**Utilidad**
`COALESCE`	Utilizado para manejar valores `NULL`. Toma el primer valor no `NULL` entre las columnas de diferentes tablas.	Particularmente útil para combinar datos de diferentes tablas.
`FULL OUTER JOIN`	Asegura que todas las filas de las tablas cruzadas se incluyan en el resultado, tanto si tienen columnas coincidentes, como si no.	Resulta crucial para combinar conjuntos de datos con posibles discrepancias en las claves. En una base de datos, las claves (keys) son columnas que se utilizan para identificar de manera única las filas de una tabla.
Agregación con `SUM` y `MAX`	Realiza una agregación por varias columnas, consiguiendo la suma total de las métricas para cada registro, y el valor máximo de otra de las columnas.	Las columnas numéricas son agregadas de manera correcta a lo largo de las tablas unidas. Además, se selecciona la última de las fechas de una columna de tipo *TIMESTAMP*.

Cuadro 5.1: Operaciones clave de `Trigger`

5.1.3. Nuevo flujo de trabajo

La Figura 5.5 muestra la disposición del nuevo flujo de trabajo. Indudablemente, resalta la reducción de la complejidad del diagrama, que incluye únicamente 10 cajas, en vez de las 35 que involucra el flujo de trabajo original. Dado que la etapa `Trigger` del nuevo flujo maneja adecuadamente los valores nulos, y que se filtra el mes máximo del año máximo en las entradas que correponden, no hace falta incluir las últimas fases del flujo original en el nuevo (a partir del `Join` de la Figura 4.1).

Gracias a un mejor manejo de las entradas aplicando CTEs, a combinar la unión de los regstros empleando una única etapa con comandos adecuados, y a aplicar una lógica más depurada y estructurada, se logra atender al objetivo de optimizar la forma del flujo de trabajo.

5.2. Tabla final generada por el nuevo flujo

Al contrario que para el flujo de trabajo original, los resultados totales de las columnas numéricas de la tabla final nueva coinciden con los que se obtienen de las tablas de origen que proporciona el cliente.

Atendiendo a otro de los problemas que surge con la tabla final que genera el flujo de trabajo original, con el nuevo flujo, los registros se unifican correctamente. Para todos los registros, se juntan adecuadamente todas las columnas numéricas para las que hay dato. De esta forma, se obtiene una tabla final consistente y coherente, tal y como la quiere el cliente.

Observando la Figura 5.3, se puede entender claramente el problema que provoca el flujo de trabajo original. Ambas filas que aparecen en la figura son del mismo registro; una fila posee los datos de DL y otra los de FL. En cambio, la Figura 5.4 muestra el mismo registro en la tabla nueva. Se puede visualizar de forma clara la mejora en la estructura de la tabla final, además de que a la tabla final original le faltan datos.

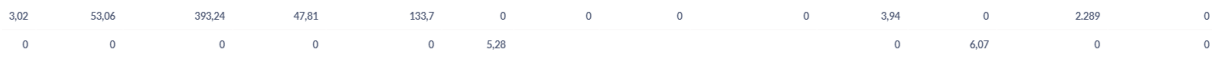


Figura 5.3: Registro de la tabla final vieja

Figura 5.4: Mismo registro en la tabla final nueva

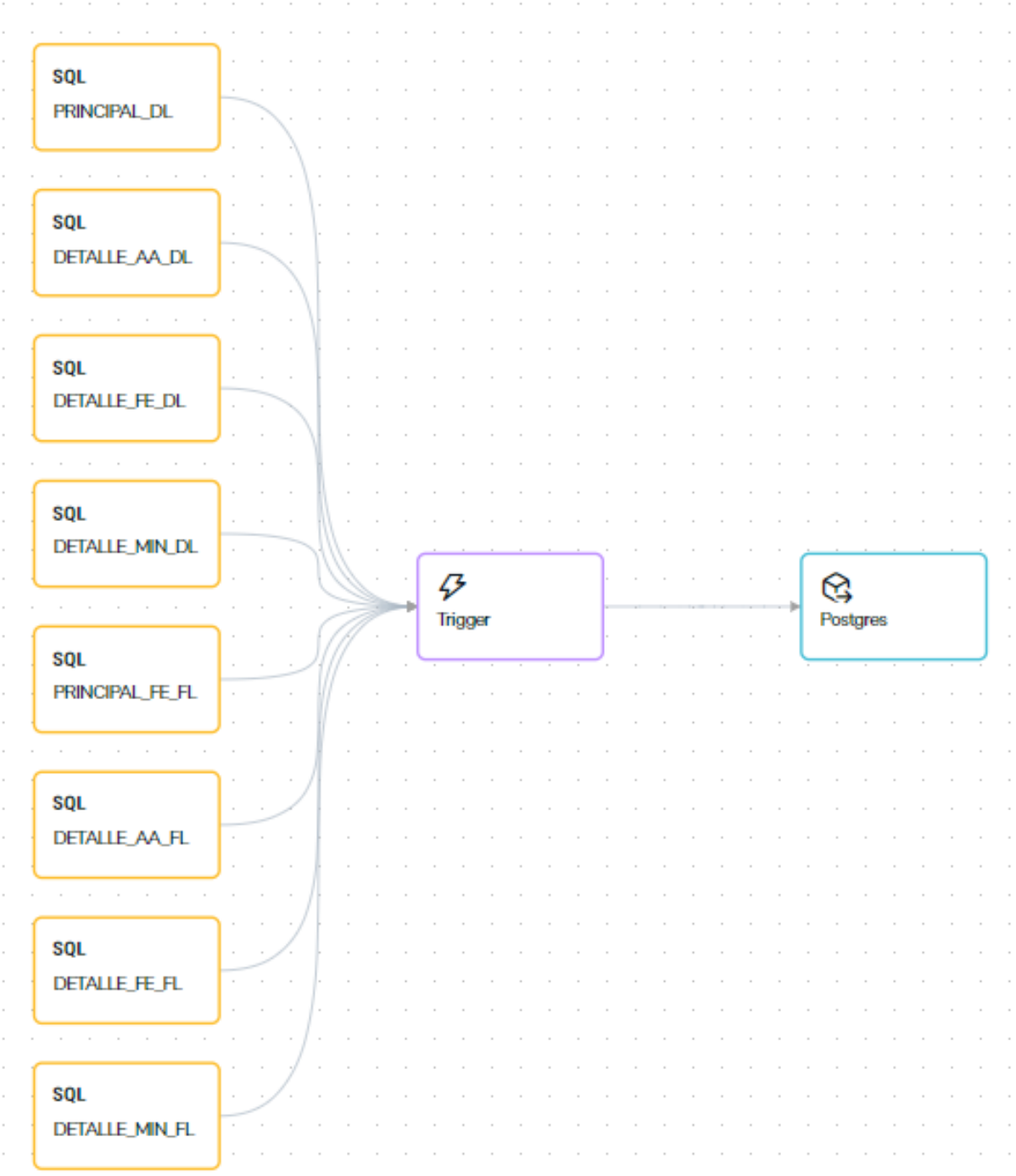


Figura 5.5: Flujo de Trabajo Nuevo

Capítulo 6

Limitaciones del Proyecto

En este capítulo, se discuten las diversas limitaciones que han surgido a la hora de optimizar el flujo de trabajo. A pesar de haber encontrado diferentes formas de mejorar el rendimiento de la ETL en su totalidad, como es la indexación de tablas y el particionamiento, ciertas restricciones restringen su implementación. Este capítulo proporciona una visión general de las limitaciones y su impacto en el proyecto.

6.1. Técnicas de optimización identificadas y valoradas

Antes de profundizar en las limitaciones, es esencial resumir las técnicas de optimización consideradas para mejorar el proceso ETL en su conjunto, con el objetivo principal de reducir el tiempo de ejecución del flujo de trabajo.

- **Indexación:** Generar tablas con campos indexados para mejorar la velocidad a la que se recuperan los datos.
- **Particionamiento:** Dividir grandes tablas en otras más pequeñas y manejables, mejorando el rendimiento de las consultas.

6.2. Limitaciones encontradas

Acto seguido, se argumenta porqué no se han podido implementar, o adaptar, las técnicas aportadas en el anterior apartado.

Falta de acceso a las bases de datos

Una de las limitaones principales es la falta de acceso a las bases de datos donde se almacenan las tablas que entran al flujo de trabajo. Esta restricción tiene una gran implicacion. Al no tener acceso a las bases de datos, no se pueden crear índices en las tablas de origen. Así pues, se elimina la posibilidad de incrementar la velocidad de recuperación de datos empleando esta técnica.

Restricciones en la creación de nuevas tablas

La incapacidad de crear nuevas tablas, incluyendo tablas temporales, plantea grandes desafíos. El particionamiento requiere la creación adicional de tablas o la modificación de

las tablas existentes. Es por ello que no fue posible aplicar el particionamiento de tablas para mejorar la velocidad de ejecución del proceso.

6.3. Impacto de las limitaciones

Las limitaciones han restringido significativamente la optimización efectiva del flujo de trabajo. La falta de acceso a recursos y entornos críticos han condicionado la aplicación de varias técnicas de optimización identificadas. La incapacidad de poder implementar las técnicas significa que el flujo de trabajo no se ha podido perfeccionar al máximo.

A pesar de las restricciones enfrentadas, el proyecto ha proporcionado conocimientos valiosos acerca de areas potenciales para optimizar la ejecución de un proceso de transformación dentro de una ETL. Entender las limitaciones enfrentadas hace destacar la importancia de tener los accesos y recursos adecuados para una efectiva transformación de los datos. Trabajos futuros deberían enfocarse en abordar estas limitaciones.

No obstante, se ha logrado optimizar el flujo de trabajo sin importar las limitaciones argumentadas. De hecho, se ha optimizado en gran medida, mejorando enormemente el tiempo de ejecución y la forma del flujo de trabajo. La optimización se ha tenido que centrar en el código SQL aplicado en cada una de las partes del flujo.

Capítulo 7

Resultados y Conclusión

El objetivo primordial de este proyecto es analizar y optimizar un flujo de trabajo empleado para realizar transformaciones de datos dentro de un proceso ETL. El foco de la optimización de la etapa de transformación ha sido disminuir el tiempo de ejecución y simplificar la estructura del flujo de trabajo. Mediante la implementación de CTEs en las diferentes etapas del flujo, a parte de entender bien la lógica de la tabla final que deseaba el cliente, se han podido observar mejoras significativas.

7.1. Resultados

El flujo de trabajo original resulta elevadamente complejo y tarda mucho tiempo en ejecutarse. El proceso involucra una desproporcionada cantidad de pasos y transformaciones.

7.1.1. Técnicas de optimización utilizadas

Gran parte de la optimización del proceso se ha conseguido gracias al uso de CTEs. Las CTEs han hecho posible simplificar y refactorizar las consultas SQL empleadas para la transformación de los datos. Así es como se ha reducido el número de pasos intermedios y se ha mejorado la interpretabilidad del flujo de trabajo. Este cambio se ejemplifica comparando los Listados B.1, B.2, B.3 y B.4 con el Listado A.1, cuyos códigos anonimizados son equivalentes. El Listado A.1 tiene un formato mucho más adecuado, organizado y legible.

7.1.2. Mejoras en el rendimiento del flujo de trabajo

De forma seguida, se argumenta el impacto de las mejoras introducidas en el tiempo de ejecución y la forma del flujo de trabajo.

Tiempo de ejecución

Las optimizaciones aplicadas han significado una importante reducción en los tiempos de ejecución del flujo de trabajo. Esto viene ejemplificado por la Tabla 7.1, donde se

presentan los tiempos que han tardado en ejecutarse tanto el flujo original como el optimizado. El tiempo medio de ejecución entre el proceso antiguo y el nuevo se ha reducido de 2 horas, 5 minutos y 33 segundos, a 24 minutos y 41 segundos. Este descenso en el tiempo de ejecución representa una mejora del $80,34\,\%$.

	Proceso Original	**Proceso Nuevo**
Tiempo medio de ejecución (minutos)	$125,55$	$24,68$

Cuadro 7.1: Comparación de tiempos de ejecución entre procesos

Forma del flujo de trabajo

La forma del flujo de trabajo ha mejorado enormemente, siendo el nuevo proceso más intuitivo y fácil de mantener. Los cambios realizados en la forma del flujo se pueden apreciar comparando la Figura 4.1 con la Figura 5.5. Se ha conseguido reducir la complejidad del flujo considerablemente, haciéndolo más accesible para futuras modificaciones. En la Tabla 7.2, se presentan diversas métricas que evidencian la reducción en la complejidad entre el flujo antiguo y el nuevo.

	Proceso Original	**Proceso Nuevo**
Numero total de lineas SQL	855	707
Numero total de consultas SQL	29	9
Numero de consultas anidadas	20	34

Cuadro 7.2: Métricas de evaluación de la forma de los flujos de trabajo

El número de lineas de código entre el nuevo flujo de trabajo y el antiguo se ve reducido en un $17,3\,\%$, y el número total de consultas decrece en un $68,97\,\%$. A decir verdad, el porcentaje de la reducción en el número de líneas y de consultas SQL debería ser superior. Esto se debe a que el flujo de trabajo original utiliza funciones que proporciona Stratio Rocket para generar operaciones sin escribir código.

Por el contrario, el número total de consultas anidadas ha aumentado en un $70\,\%$. Este incremento es lógico y esperado, considerando el número de CTEs implicadas en el flujo de trabajo optimizado. Puesto que las CTEs son una herramienta que ayuda a estructurar y organizar consultas complejas, su uso conlleva un mayor número de subconsultas individuales. Aunque haya habido un incremento significativo, las CTEs contribuyen a mejorar la legibilidad y el mantenimiento del código SQL, facilitando su comprensión y modificación.

7.1.3. Discusión de los resultados obtenidos

Los resultados compartidos en la anterior sección indican, indudablemente, que el flujo de trabajo se ha optimizado de manera efectiva. La forma de evaluar los tiempos de

ejecución entre ambos flujos es directa, y es una medida que da Rocket una vez se ejecuta el proceso. No cabe duda de que, en términos de tiempo de ejecución, el nuevo flujo de trabajo es significativamente más rápido.

De la misma manera, se ha conseguido reducir la complejidad del flujo de trabajo enormemete. Esto no solo viene evidenciado por las Figuras 4.1 y 5.5, sino que también se ofrece una serie de medidas, en la Tabla 5.1, que demuestran el cumplimiento de este objetivo.

7.2. Conclusiones

Atendiendo a los resultados obtenidos, se han conseguido alcanzar los objetivos establecidos a la hora de empezar el proyecto. Por lo tanto, se ha logrado reducir la complejidad de la forma del flujo de trabajo y el tiempo de ejecución.

Dado que la optimización ha sido muy satisfactoria, se deduce que también se ha logrado otro de los objetivos clave, que es analizar y comprender a fondo el flujo de trabajo y los datos involucrados. Sin este entendimiento de los pasos involucrados en el flujo original, de los datos que intervienen en el flujo de trabajo, y de la forma deseada para la tabla final, no se hubiera logrado optimizar adecuadamente el flujo de trabajo. Además, se han conseguido arreglar las imperfecciones que generaba el proceso original, resultando en una tabla final con una estructura precisa y consistente.

Las limitaciones de accesos y permisos no han presentado un gran problema. De todas formas, me hubiera gustado poder inferir en el particionamiento de las tablas y en la indexación de campos. Esto se debe a que, así, podría estudiar el grado en el que se puede disminuir en mayor medida el tiempo de ejecución del flujo de trabajo.

Apéndice A

Códigos anonimizados presentes en el flujo de trabajo optimizado

A.1. Código de `PRINCIPAL_DL`

El siguiente fragmento de código corresponde al procedimiento `PRINCIPAL_DL` presente en la Figura 5.5, dentro del flujo de trabajo optimizado. Este código realiza una serie de operaciones SQL para filtrar y agrupar datos auditados de este año. Con esta entrada al flujo se obtienen los metros lineales de los productos dentro del lineal. El código se encuentra anonimizado en el Listado A.1. Solo se ha decidido incluir `PRINCIPAL_DL` para dar una noción de los cambios realizados respecto al flujo de trabajo original.

Listing A.1: Código de `PRINCIPAL_DL` de la Figura 5.5

```
WITH ANIO_MAX AS (
    SELECT MAX(YEAR(B.Column2)) AS ANIO_MAX
    FROM anonymous_schema.Table1 A
    LEFT JOIN anonymous_schema.Table2 B
    ON LTRIM(RTRIM(A.Column3)) = LTRIM(RTRIM(B.Column3))
    WHERE A.Column4 IN ('13')
    AND A.Column5 <> 0
),
AUDIT_DL_FILTRO AS (
    SELECT
        YEAR(B.Column2) AS ANIO,
        MONTH(B.Column2) AS MES,
        A.*,
        B.Column2
    FROM anonymous_schema.Table1 A
    LEFT JOIN anonymous_schema.Table2 B
    ON LTRIM(RTRIM(A.Column3)) = LTRIM(RTRIM(B.Column3))
    JOIN ANIO_MAX C
    ON YEAR(B.Column2) = C.ANIO_MAX
    OR YEAR(B.Column2) = C.ANIO_MAX - 1
)
```

```
SELECT
    A.ANIO,
    A.MES,
    A.Column6 AS ID_PdV,
    CASE WHEN C.Column7 = '5'
    THEN 'CANARIAS' ELSE 'PENINSULA' END AS ZONA_ESPANA,
    E.Column8 AS ID_DIRECTOR_KAM,
    E.Column9 AS NOMBRE_DIRECTOR_KAM,
    E.Column10 AS ID_KAM,
    E.Column11 AS NOMBRE_KAM,
    E.Column12 AS CLIENTE20,
    E.Column13 AS DESCRIP_CLIENTE20,
    F.Column14 AS ID_MegaMarca,
    F.Column15 AS MegaMarca,
    F.Column16 AS Fabricante,
    F.Column17 AS Marca_MsM,
    F.Column18 AS ID_Marca_MSM,
    'T_003' AS COD_ORIGEN,
    'Detalle_FE_Espacio' AS TABLA_ORIGEN,
    CURRENT_TIMESTAMP AS STRATIO_CREATED_DATE,
    SUM(A.Column5) AS ML_DL,
    0 AS mL_FE_DL,
    0 AS Retorno_FE_DL,
    0 AS ML_DL_AA_DIC,
    0 AS ML_DL_AA,
    0 AS mL_MIN_DL,
    0 AS Retorno_MIN_DL,
    0 AS ML_FL,
    0 AS mL_FE_FL,
    0 AS mL_MIN_FL,
    0 AS ML_Min_FL_AA,
    0 AS ML_FL_AA,
    0 AS ML_FL_AA_DIC
FROM AUDIT_DL_FILTRO A
LEFT JOIN anonymous_schema.Table3 C
ON LTRIM(RTRIM(A.ID_PdV)) = LTRIM(RTRIM(C.Column6))
LEFT JOIN anonymous_schema.Table4 E
ON LTRIM(RTRIM(E.Column19))
= LTRIM(RTRIM(C.Column20))
LEFT JOIN (
    SELECT DISTINCT LTRIM('0',
    (LTRIM(RTRIM(Column21)))) AS Column21,
    CAST(Column18 AS bigint) AS Column18,
    Column17,
    Column14,
    Column15,
    Column16,
    Column22
```

```
    FROM anonymous_schema.Table5
) F
ON LTRIM('0', (LTRIM(RTRIM(F.Column21)))) =
LTRIM('0', (LTRIM(RTRIM(A.Column21))))
WHERE
    A.ANIO = (SELECT MAX(ANIO) FROM AUDIT_DL_FILTRO)
    AND A.Column4 IN ('13')
    AND F.Column22 IN ('CERVEZA')
GROUP BY
    A.ANIO, A.MES, A.ID_PdV,
    C.Column7,
    E.Column8, E.Column9, E.Column10, E.Column11,
    E.Column12, E.Column13,
    F.Column14, F.Column15, F.Column16, F.Column17, F.Column18
```

A.2. Código de `Trigger`

En el Listado A.2, se presenta el código anonimizado de la etapa de `Trigger` del flujo de trabajo optimizado (Figura 5.5). Este código se emplea para combinar todos los datos de los productos dentro y fuera del lineal.

Listing A.2: Código de `Trigger` de la Figura 5.5

```
SELECT
    COALESCE(A.ANIO, G.ANIO, Z.ANIO, U.ANIO,
    F.ANIO, H.ANIO, J.ANIO, K.ANIO) AS ANIO,
    COALESCE(A.MES, G.MES, Z.MES, U.MES,
    F.MES, H.MES, J.MES, K.MES) AS MES,
    COALESCE(A.Column1, G.Column1, Z.Column1,
    U.Column1, F.Column1, H.Column1, J.Column1,
    K.Column1) AS ID_PdV,
    '0' AS EAN,
    COALESCE(A.Column2, G.Column2,
    Z.Column2, U.Column2, F.Column2,
    H.Column2, J.Column2, K.Column2)
    AS ZONA_ESPANA,
    COALESCE(A.Column3, G.Column3,
    Z.Column3, U.Column3,
    F.Column3, H.Column3,
    J.Column3, K.Column3)
    AS ID_DIRECTOR_KAM,
    COALESCE(A.Column4,
    G.Column4, Z.Column4,
    U.Column4, F.Column4,
    H.Column4, J.Column4,
    K.Column4) AS NOMBRE_DIRECTOR_KAM,
    COALESCE(A.Column5, G.Column5, Z.Column5,
```

```
U.Column5, F.Column5, H.Column5, J.Column5,
K.Column5) AS ID_KAM,
COALESCE(A.Column6, G.Column6,
Z.Column6, U.Column6, F.Column6,
H.Column6, J.Column6, K.Column6)
AS NOMBRE_KAM,
COALESCE(A.Column7, G.Column7,
Z.Column7, U.Column7, F.Column7,
H.Column7, J.Column7, K.Column7)
AS CLIENTE20,
COALESCE(A.Column8,
G.Column8, Z.Column8,
U.Column8, F.Column8,
H.Column8, J.Column8,
K.Column8) AS DESCRIP_CLIENTE20,
COALESCE(A.Column9, G.Column9,
Z.Column9, U.Column9, F.Column9,
H.Column9, J.Column9, K.Column9)
AS ID_MegaMarca,
COALESCE(A.Column10, G.Column10, Z.Column10,
U.Column10, F.Column10, H.Column10,
J.Column10, K.Column10) AS MegaMarca,
COALESCE(A.Column11, G.Column11,
Z.Column11, U.Column11, F.Column11,
H.Column11, J.Column11, K.Column11)
AS Fabricante,
COALESCE(A.Column12, G.Column12, Z.Column12,
U.Column12, F.Column12, H.Column12,
J.Column12, K.Column12) AS Marca_MsM,
COALESCE(A.Column13, G.Column13,
Z.Column13, U.Column13, F.Column13,
H.Column13, J.Column13, K.Column13)
AS ID_Marca_MSM,
'0' AS ID_Visita,
COALESCE(A.Column14, G.Column14, Z.Column14,
U.Column14, F.Column14, H.Column14,
J.Column14, K.Column14) AS COD_ORIGEN,
COALESCE(A.Column15, G.Column15,
Z.Column15, U.Column15, F.Column15,
H.Column15, J.Column15, K.Column15)
AS TABLA_ORIGEN,
MAX(COALESCE(A.Column16,
G.Column16, Z.Column16,
U.Column16, F.Column16,
H.Column16, J.Column16,
K.Column16)) AS STRATIO_CREATED_DATE,
SUM(COALESCE(A.Column17, 0)) AS ML_DL,
SUM(COALESCE(G.Column18, 0)) AS mL_FE_DL,
```

```
    SUM(COALESCE(G.Column19, 0)) AS Retorno_FE_DL,
    SUM(COALESCE(Z.Column20, 0)) AS mL_Min_DL,
    SUM(COALESCE(Z.Column21, 0)) AS Retorno_Min_DL,
    SUM(COALESCE(U.Column22, 0)) AS ML_DL_AA,
    SUM(COALESCE(G.Column23, 0)) AS ML_DL_AA_DIC,
    SUM(COALESCE(F.Column24, 0)) AS ML_FL,
    SUM(COALESCE(H.Column25, 0)) AS mL_FE_FL,
    SUM(COALESCE(H.Column26, 0)) AS ML_FL_AA_DIC,
    SUM(COALESCE(J.Column27, 0)) AS mL_Min_FL,
    SUM(COALESCE(J.Column28, 0)) AS mL_MinDen_FL_AA,
    SUM(COALESCE(K.Column29, 0)) AS ML_FL_AA
FROM anonymous_schema.PRINCIPAL_DL A
FULL OUTER JOIN anonymous_schema.DETALLE_FE_DL G
    ON LTRIM(RTRIM(G.ANIO)) = LTRIM(RTRIM(A.ANIO))
    AND LTRIM(RTRIM(G.MES)) = LTRIM(RTRIM(CAST(A.MES AS STRING)))
    AND LTRIM(RTRIM(A.Column1)) = LTRIM(RTRIM(G.Column1))
    AND LTRIM('0', LTRIM(RTRIM(A.Column13))) =
    LTRIM(RTRIM(G.Column13))
FULL OUTER JOIN anonymous_schema.DETALLE_MIN_DL Z
    ON LTRIM(RTRIM(Z.ANIO)) = LTRIM(RTRIM(A.ANIO))
    AND LTRIM(RTRIM(Z.MES)) = LTRIM(RTRIM(CAST(A.MES AS STRING)))
    AND LTRIM(RTRIM(A.Column1)) = LTRIM(RTRIM(Z.Column1))
    AND LTRIM('0', LTRIM(RTRIM(A.Column13))) =
    LTRIM(RTRIM(Z.Column13))
FULL OUTER JOIN anonymous_schema.DETALLE_AA_DL U
    ON LTRIM(RTRIM(U.Column1)) = LTRIM(RTRIM(A.Column1))
    AND LTRIM('0', LTRIM(RTRIM(U.Column13))) =
    LTRIM('0', LTRIM(RTRIM(A.Column13)))
    AND LTRIM(RTRIM(CAST(U.MES AS STRING))) =
    LTRIM(RTRIM(A.MES))
FULL OUTER JOIN anonymous_schema.PRINCIPAL_FE_FL F
    ON LTRIM(RTRIM(F.ANIO)) = LTRIM(RTRIM(A.ANIO))
    AND LTRIM(RTRIM(F.MES)) = LTRIM(RTRIM(CAST(A.MES AS STRING)))
    AND LTRIM(RTRIM(A.Column1)) = LTRIM(RTRIM(F.Column1))
    AND LTRIM('0', LTRIM(RTRIM(A.Column13))) =
    LTRIM(RTRIM(F.Column13))
FULL OUTER JOIN anonymous_schema.DETALLE_FE_FL H
    ON LTRIM(RTRIM(H.ANIO)) = LTRIM(RTRIM(A.ANIO))
    AND LTRIM(RTRIM(H.MES)) = LTRIM(RTRIM(CAST(A.MES AS STRING)))
    AND LTRIM(RTRIM(A.Column1)) = LTRIM(RTRIM(H.Column1))
    AND LTRIM('0', LTRIM(RTRIM(A.Column13))) =
    LTRIM(RTRIM(H.Column13))
FULL OUTER JOIN anonymous_schema.DETALLE_MIN_FL J
    ON LTRIM(RTRIM(J.ANIO)) = LTRIM(RTRIM(A.ANIO))
    AND LTRIM(RTRIM(J.MES)) = LTRIM(RTRIM(CAST(A.MES AS STRING)))
    AND LTRIM(RTRIM(A.Column1)) = LTRIM(RTRIM(J.Column1))
    AND LTRIM('0', LTRIM(RTRIM(A.Column13))) =
    LTRIM(RTRIM(J.Column13))
```

```
FULL OUTER JOIN anonymous_schema.DETALLE_AA_FL K
    ON LTRIM(RTRIM(K.Column1)) = LTRIM(RTRIM(A.Column1))
    AND LTRIM('0', LTRIM(RTRIM(K.Column13))) =
    LTRIM('0', LTRIM(RTRIM(A.Column13)))
    AND LTRIM(RTRIM(CAST(K.MES AS STRING))) =
    LTRIM(RTRIM(A.MES))
GROUP BY
    COALESCE(A.ANIO, G.ANIO, Z.ANIO, U.ANIO,
    F.ANIO, H.ANIO, J.ANIO, K.ANIO),
    COALESCE(A.MES, G.MES, Z.MES, U.MES, F.MES,
    H.MES, J.MES, K.MES),
    COALESCE(A.Column1, G.Column1, Z.Column1,
    U.Column1, F.Column1, H.Column1, J.Column1,
    K.Column1),
    COALESCE(A.Column2, G.Column2,
    Z.Column2, U.Column2, F.Column2,
    H.Column2, J.Column2, K.Column2),
    COALESCE(A.Column3, G.Column3,
    Z.Column3, U.Column3,
    F.Column3, H.Column3,
    J.Column3, K.Column3),
    COALESCE(A.Column4, G.Column4,
    Z.Column4, U.Column4,
    F.Column4, H.Column4,
    J.Column4, K.Column4),
    COALESCE(A.Column5, G.Column5, Z.Column5, U.Column5,
    F.Column5, H.Column5, J.Column5, K.Column5),
    COALESCE(A.Column6, G.Column6, Z.Column6,
    U.Column6, F.Column6, H.Column6, J.Column6,
    K.Column6),
    COALESCE(A.Column7, G.Column7, Z.Column7,
    U.Column7, F.Column7, H.Column7, J.Column7,
    K.Column7),
    COALESCE(A.Column8, G.Column8,
    Z.Column8, U.Column8,
    F.Column8, H.Column8,
    J.Column8, K.Column8),
    COALESCE(A.Column9, G.Column9, Z.Column9,
    U.Column9, F.Column9, H.Column9,
    J.Column9, K.Column9),
    COALESCE(A.Column10, G.Column10, Z.Column10,
    U.Column10, F.Column10, H.Column10, J.Column10,
    K.Column10),
    COALESCE(A.Column11, G.Column11, Z.Column11,
    U.Column11, F.Column11, H.Column11, J.Column11,
    K.Column11),
    COALESCE(A.Column12, G.Column12, Z.Column12,
    U.Column12, F.Column12, H.Column12, J.Column12,
```

```
K.Column12),
COALESCE(A.Column13, G.Column13, Z.Column13,
U.Column13, F.Column13, H.Column13,
J.Column13, K.Column13),
COALESCE(A.Column14, G.Column14, Z.Column14,
U.Column14, F.Column14, H.Column14, J.Column14,
K.Column14),
COALESCE(A.Column15, G.Column15, Z.Column15,
U.Column15, F.Column15, H.Column15,
J.Column15, K.Column15)
```

Apéndice B

Códigos anonimizados presentes en el flujo de trabajo original

B.1. Consultas del flujo de trabajo original que equivalen a la entrada PRINCIPAL_DL en el flujo nuevo.

En este apartado, se incluye un conjunto de listados donde intervienen varias consultas SQL anonimizadas. Estas consultas combinadas, que pertenecen al flujo antiguo, realizan la misma función que PRINCIPAL_DL en el flujo nuevo, es decir, equivalen al Listado A.1. El conjunto de consultas están presentes en el Listado B.1, Listado B.2, Listado B.3 y Listado B.4.

Listing B.1: Código de ANIO_MAX de la Figura 4.1

```
SELECT MAX(ANIO) AS ANIO_MAX FROM (
SELECT
    YEAR(B.Column1) AS ANIO,
    Column2 AS ML_DL
FROM anonymous_schema.Table1 A
LEFT JOIN (SELECT Column1, Column3
FROM
anonymous_schema.Table2) B
ON LTRIM(RTRIM(A.Column3)) = LTRIM(RTRIM(B.Column3))
WHERE Column4 IN ('13')
) C
WHERE ML_DL <> 0
```

Listing B.2: Código de AUDIT_DL de la Figura 4.1

```
SELECT
    YEAR(B.Column1) AS ANIO,
    MONTH(B.Column1) AS MES,
    *
FROM anonymous_schema.Table1 A
LEFT JOIN (SELECT Column1, Column2
FROM
```

```
anonymous_schema.Table2) B
ON LTRIM(RTRIM(A.Column2)) = LTRIM(RTRIM(B.Column2))
```

Listing B.3: Código de `AUDIT_DL_FILTRO` de la Figura 4.1

```
SELECT *
FROM anonymous_schema.Table1 A
JOIN anonymous_schema.Table2 B
ON B.Column1 = A.Column2 OR A.Column2 = B.Column1 - 1
```

Listing B.4: Código de `PRINCIPAL_DL` de la Figura 4.1

```
SELECT
    A.Column1 AS ANIO,
    A.Column2 AS MES,
    A.Column3 AS ID_PdV,
    (CASE WHEN C.Column4 = '5' THEN 'CANARIAS'
    ELSE 'PENINSULA' END) AS ZONA_ESPANA,
    E.Column5 AS ID_DIRECTOR_KAM,
    E.Column6 AS NOMBRE_DIRECTOR_KAM,
    E.Column7 AS ID_KAM,
    E.Column8 AS NOMBRE_KAM,
    E.Column9 AS CLIENTE20,
    E.Column10 AS DESCRIP_CLIENTE20,
    F.Column11 AS ID_MegaMarca,
    F.Column12 AS MegaMarca,
    F.Column13 AS Fabricante,
    F.Column14 AS Marca_MsM,
    F.Column15 AS ID_Marca_MSM,
    'T_003' AS COD_ORIGEN,
    'Detalle_FE_Espacio' AS TABLA_ORIGEN,
    CURRENT_TIMESTAMP AS STRATIO_CREATED_DATE,
    SUM(A.Column16) AS ML_DL,
    0 AS mL_FE_DL,
    0 AS Retorno_FE_DL,
    0 AS ML_DL_AA_DIC,
    0 AS ML_DL_AA,
    0 AS mL_MIN_DL,
    0 AS Retorno_MIN_DL,
    0 AS ML_FL,
    0 AS mL_FE_FL,
    0 AS mL_MIN_FL,
    0 AS ML_Min_FL_AA,
    0 AS ML_FL_AA,
    0 AS ML_FL_AA_DIC
FROM anonymous_schema.Table1 A
LEFT JOIN anonymous_schema.Table2 C
ON LTRIM(RTRIM(A.Column3)) = LTRIM(RTRIM(C.Column3))
LEFT JOIN anonymous_schema.Table3 E
ON LTRIM(RTRIM(E.Column17)) = LTRIM(RTRIM(C.Column18))
```

```
LEFT JOIN (
    SELECT DISTINCT LTRIM('0', (LTRIM(RTRIM(Column19)))) AS Column19,
    Column15 AS ID_Marca_MSM,
    Column14 AS Marca_MsM,
    Column11 AS ID_MegaMarca,
    Column12 AS MegaMarca,
    Column13 AS Fabricante,
    Column20 AS Tipo_Producto
    FROM anonymous_schema.Table4
) F
ON LTRIM('0', (LTRIM(RTRIM(F.Column19)))) =
LTRIM('0', (LTRIM(RTRIM(A.Column21))))
WHERE
    A.Column1 = (SELECT MAX(Column1) FROM anonymous_schema.Table1)
    AND A.Column22 IN ('13') AND F.Column20 IN ('CERVEZA')
GROUP BY
    A.Column1, A.Column2, A.Column3,
    C.Column4,
    E.Column5, E.Column6, E.Column7, E.Column8, E.Column9,
    E.Column10,
    F.Column11, F.Column12, F.Column13, F.Column14, F.Column15
```

B.2. Código de `TRIGGER_DL` del flujo de trabajo original

A continuación, en el Listado B.5, se presenta el código anonimizado de la etapa `TRIGGER_DL` del flujo de trabajo original. Este código se emplea para combinar todos los datos de los productos dentro del lineal. Se ha decidido incluirlo para comparar de manera directa su legibilidad frente a la del código de `Trigger` (Listado A.2), que pertenece al flujo de trabajo optimizado.

Listing B.5: Código de `TRIGGER_DL` de la Figura 4.1

```
SELECT
    A.Column1 AS ANIO,
    A.Column2 AS MES,
    A.Column3 AS ID_PdV,
    '0' AS EAN,
    A.Column4 AS ZONA_ESPANA,
    A.Column5 AS ID_DIRECTOR_KAM,
    A.Column6 AS NOMBRE_DIRECTOR_KAM,
    A.Column7 AS ID_KAM,
    A.Column8 AS NOMBRE_KAM,
    A.Column9 AS CLIENTE20,
    A.Column10 AS DESCRIP_CLIENTE20,
    A.Column11 AS ID_MegaMarca,
    A.Column12 AS MegaMarca,
    A.Column13 AS Fabricante,
    A.Column14 AS Marca_MsM,
    A.Column15 AS ID_Marca_MSM,
```

```
      '0' AS ID_Visita,
      A.Column16 AS COD_ORIGEN,
      A.Column17 AS TABLA_ORIGEN,
      A.Column18 AS STRATIO_CREATED_DATE,
      A.Column19 AS ML_DL,
      G.Column20 AS mL_FE_DL,
      G.Column21 AS Retorno_FE_DL,
      Z.Column22 AS mL_Min_DL,
      Z.Column23 AS Retorno_Min_DL,
      U.Column24 AS ML_DL_AA,
      G.Column25 AS ML_DL_AA_DIC,
      A.Column26 AS ML_FL,
      A.Column27 AS mL_FE_FL,
      A.Column28 AS mL_Min_FL,
      A.Column29 AS ML_MinDen_FL_AA,
      A.Column30 AS ML_FL_AA,
      A.Column31 AS ML_FL_AA_DIC
FROM anonymous_schema.Table1 A
LEFT JOIN anonymous_schema.Table2 G
      ON LTRIM(RTRIM(G.Column1)) = LTRIM(RTRIM(A.Column1))
      AND LTRIM(RTRIM(G.Column2)) =
      LTRIM(RTRIM(CAST(A.Column2 AS STRING)))
      AND LTRIM(RTRIM(A.Column3)) = LTRIM(RTRIM(G.Column3))
      AND LTRIM('0', (LTRIM(RTRIM(A.Column15)))) =
      LTRIM (RTRIM(G.Column15))
LEFT JOIN anonymous_schema.Table3 Z
      ON LTRIM(RTRIM(Z.Column1)) = LTRIM(RTRIM(A.Column1))
      AND LTRIM(RTRIM(Z.Column2)) =
      LTRIM(RTRIM(CAST(A.Column2 AS STRING)))
      AND LTRIM(RTRIM(A.Column3)) = LTRIM(RTRIM(Z.Column3))
      AND LTRIM('0', (LTRIM(RTRIM(A.Column15)))) =
      LTRIM (RTRIM(Z.Column15))
LEFT JOIN anonymous_schema.Table4 U
      ON LTRIM(RTRIM(U.Column3)) = LTRIM(RTRIM(A.Column3))
      AND LTRIM('0', (LTRIM(RTRIM(U.Column15)))) =
      LTRIM('0', (LTRIM(RTRIM(A.Column15))))
      AND LTRIM(RTRIM(CAST(U.Column2 AS STRING)))
      = LTRIM(RTRIM(CAST(A.Column2 AS STRING)))

UNION

SELECT * FROM (
SELECT
      X.Column1 AS ANIO,
      X.Column2 AS MES,
      X.Column3 AS ID_PdV,
      '0' AS EAN,
      X.Column4 AS ZONA_ESPANA,
```

```
    X.Column5 AS ID_DIRECTOR_KAM,
    X.Column6 AS NOMBRE_DIRECTOR_KAM,
    X.Column7 AS ID_KAM,
    X.Column8 AS NOMBRE_KAM,
    X.Column9 AS CLIENTE20,
    X.Column10 AS DESCRIP_CLIENTE20,
    X.Column11 AS ID_MegaMarca,
    X.Column12 AS MegaMarca,
    X.Column13 AS Fabricante,
    X.Column14 AS Marca_MsM,
    X.Column15 AS ID_Marca_MSM,
    '0' AS ID_Visita,
    X.Column16 AS COD_ORIGEN,
    X.Column17 AS TABLA_ORIGEN,
    X.Column18 AS STRATIO_CREATED_DATE,
    G.Column19 AS ML_DL,
    X.Column20 AS mL_FE_DL,
    X.Column21 AS Retorno_FE_DL,
    Z.Column22 AS mL_Min_DL,
    Z.Column23 AS Retorno_Min_DL,
    U.Column24 AS ML_DL_AA,
    X.Column25 AS ML_DL_AA_DIC,
    X.Column26 AS ML_FL,
    X.Column27 AS mL_FE_FL,
    X.Column28 AS mL_Min_FL,
    X.Column29 AS ML_MinDen_FL_AA,
    X.Column30 AS ML_FL_AA,
    X.Column31 AS ML_FL_AA_DIC
FROM anonymous_schema.Table2 X
LEFT JOIN anonymous_schema.Table1 G
    ON LTRIM(RTRIM(X.Column1)) = LTRIM(RTRIM(G.Column1))
    AND LTRIM(RTRIM(X.Column2)) =
    LTRIM(RTRIM(CAST(G.Column2 AS STRING)))
    AND LTRIM(RTRIM(G.Column3)) = LTRIM(RTRIM(X.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(G.Column15)))) =
    LTRIM (RTRIM(X.Column15))
LEFT JOIN anonymous_schema.Table3 Z
    ON LTRIM(RTRIM(Z.Column1)) = LTRIM(RTRIM(X.Column1))
    AND LTRIM(RTRIM(Z.Column2)) = LTRIM(RTRIM(X.Column2))
    AND LTRIM(RTRIM(X.Column3)) = LTRIM(RTRIM(Z.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(X.Column15)))) =
    LTRIM (RTRIM(Z.Column15))
LEFT JOIN anonymous_schema.Table4 U
    ON LTRIM(RTRIM(U.Column3)) = LTRIM(RTRIM(X.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(U.Column15)))) =
    LTRIM('0', (LTRIM(RTRIM(X.Column15))))
    AND LTRIM(RTRIM(CAST(U.Column2 AS STRING))) =
    LTRIM(RTRIM(X.Column2))
```

```
)
WHERE ML_DL IS NULL

UNION

SELECT * FROM (
SELECT
    Y.Column1 AS ANIO,
    Y.Column2 AS MES,
    Y.Column3 AS ID_PdV,
    '0' AS EAN,
    Y.Column4 AS ZONA_ESPANA,
    Y.Column5 AS ID_DIRECTOR_KAM,
    Y.Column6 AS NOMBRE_DIRECTOR_KAM,
    Y.Column7 AS ID_KAM,
    Y.Column8 AS NOMBRE_KAM,
    Y.Column9 AS CLIENTE20,
    Y.Column10 AS DESCRIP_CLIENTE20,
    Y.Column11 AS ID_MegaMarca,
    Y.Column12 AS MegaMarca,
    Y.Column13 AS Fabricante,
    Y.Column14 AS Marca_MsM,
    Y.Column15 AS ID_Marca_MSM,
    '0' AS ID_Visita,
    Y.Column16 AS COD_ORIGEN,
    Y.Column17 AS TABLA_ORIGEN,
    Y.Column18 AS STRATIO_CREATED_DATE,
    G.Column19 AS ML_DL,
    Z.Column20 AS mL_FE_DL,
    Z.Column21 AS Retorno_FE_DL,
    Y.Column22 AS mL_Min_DL,
    Y.Column23 AS Retorno_Min_DL,
    U.Column24 AS ML_DL_AA,
    Z.Column25 AS ML_DL_AA_DIC,
    Y.Column26 AS ML_FL,
    Y.Column27 AS mL_FE_FL,
    Y.Column28 AS mL_Min_FL,
    Y.Column29 AS ML_MinDen_FL_AA,
    Y.Column30 AS ML_FL_AA,
    Y.Column31 AS ML_FL_AA_DIC
FROM anonymous_schema.Table3 Y
LEFT JOIN anonymous_schema.Table1 G
    ON LTRIM(RTRIM(Y.Column1)) = LTRIM(RTRIM(G.Column1))
    AND LTRIM(RTRIM(Y.Column2)) =
    LTRIM(RTRIM(CAST(G.Column2 AS STRING)))
    AND LTRIM(RTRIM(G.Column3)) = LTRIM(RTRIM(Y.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(G.Column15)))) =
    LTRIM (RTRIM(Y.Column15))
```

```
LEFT JOIN anonymous_schema.Table2 Z
    ON LTRIM(RTRIM(Z.Column1)) = LTRIM(RTRIM(Y.Column1))
    AND LTRIM(RTRIM(Z.Column2)) = LTRIM(RTRIM(Y.Column2))
    AND LTRIM(RTRIM(Y.Column3)) = LTRIM(RTRIM(Z.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(Y.Column15)))) =
    LTRIM (RTRIM(Z.Column15))
LEFT JOIN anonymous_schema.Table4 U
    ON LTRIM(RTRIM(U.Column3)) = LTRIM(RTRIM(Y.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(U.Column15)))) =
    LTRIM('0', (LTRIM(RTRIM(Y.Column15))))
    AND LTRIM(RTRIM(CAST(U.Column2 AS STRING))) =
    LTRIM(RTRIM(Y.Column2))
)
WHERE ML_DL IS NULL AND mL_FE_DL IS NULL

UNION

SELECT * FROM (
SELECT
    V.Column1 AS ANIO,
    V.Column2 AS MES,
    V.Column3 AS ID_PdV,
    '0' AS EAN,
    V.Column4 AS ZONA_ESPANA,
    V.Column5 AS ID_DIRECTOR_KAM,
    V.Column6 AS NOMBRE_DIRECTOR_KAM,
    V.Column7 AS ID_KAM,
    V.Column8 AS NOMBRE_KAM,
    V.Column9 AS CLIENTE20,
    V.Column10 AS DESCRIP_CLIENTE20,
    V.Column11 AS ID_MegaMarca,
    V.Column12 AS MegaMarca,
    V.Column13 AS Fabricante,
    V.Column14 AS Marca_MsM,
    V.Column15 AS ID_Marca_MSM,
    '0' AS ID_Visita,
    V.Column16 AS COD_ORIGEN,
    V.Column17 AS TABLA_ORIGEN,
    V.Column18 AS STRATIO_CREATED_DATE,
    G.Column19 AS ML_DL,
    Z.Column20 AS mL_FE_DL,
    Z.Column21 AS Retorno_FE_DL,
    U.Column22 AS mL_Min_DL,
    U.Column23 AS Retorno_Min_DL,
    V.Column24 AS ML_DL_AA,
    Z.Column25 AS ML_DL_AA_DIC,
    V.Column26 AS ML_FL,
    V.Column27 AS mL_FE_FL,
```

```
    V.Column28 AS mL_Min_FL,
    V.Column29 AS ML_MinDen_FL_AA,
    V.Column30 AS ML_FL_AA,
    V.Column31 AS ML_FL_AA_DIC
FROM anonymous_schema.Table4 V
LEFT JOIN anonymous_schema.Table1 G
    ON LTRIM(RTRIM(V.Column1)) = LTRIM(RTRIM(G.Column1))
    AND LTRIM(RTRIM(CAST(V.Column2 AS STRING))) =
    LTRIM(RTRIM(CAST(G.Column2 AS STRING)))
    AND LTRIM(RTRIM(G.Column3)) = LTRIM(RTRIM(V.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(G.Column15)))) =
    LTRIM (RTRIM(V.Column15))
LEFT JOIN anonymous_schema.Table2 Z
    ON LTRIM(RTRIM(Z.Column1)) = LTRIM(RTRIM(V.Column1))
    AND LTRIM(RTRIM(Z.Column2)) =
    LTRIM(RTRIM(CAST(V.Column2 AS STRING)))
    AND LTRIM(RTRIM(V.Column3)) = LTRIM(RTRIM(Z.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(V.Column15)))) =
    LTRIM (RTRIM(Z.Column15))
LEFT JOIN anonymous_schema.Table3 U
    ON LTRIM(RTRIM(U.Column3)) = LTRIM(RTRIM(V.Column3))
    AND LTRIM('0', (LTRIM(RTRIM(U.Column15)))) =
    LTRIM('0', (LTRIM(RTRIM(V.Column15))))
    AND LTRIM(RTRIM(U.Column2)) =
    LTRIM(RTRIM(CAST(V.Column2 AS STRING)))
WHERE V.Column2 <= G.Column2
)
WHERE ML_DL IS NULL AND mL_FE_DL IS NULL AND mL_Min_DL IS NULL
```

Bibliografía

Bartley, K. (2023). What is the Difference Between ETL and ELT? [Accedido: 28 de mayo de 2024]. https://rivery.io/blog/etl-vs-elt/

Bilsdon, A. (2021). Data Flexibility [Accedido: 5 de junio de 2024]. https://www.datinal.com.au/2021/02/06/data-flexibility/

Bisso, I. L. (2022). What is a Common Table Expression? [Accedido: 23 de mayo de 2024]. https://learnsql.com/blog/what-is-common-table-expression/

Hustleescape. (2019). Supermarket Psychology: How They Use Science to Make Us Spend More [Accedido: 29 de mayo de 2024]. https://www.hustleescape.com/supermarket-spending-psychology/

IBM. (2024a). What is a data warehouse? [Accedido: 2 de junio de 2024]. https://www.ibm.com/topics/data-warehouse

IBM. (2024b). What is a relational database? [Accedido: 2 de junio de 2024]. https://www.ibm.com/topics/relational-databases

PostgreSQL. (2024). PostgreSQL: About [Accedido: 30 de mayo de 2024]. https://www.postgresql.org/about/#:~:text=PostgreSQL%20is%20a%20powerful%2C%20open%20source%20object-relational%20database,store%20and%20scale%20the%20most%20complicated%20data%20workloads.

Starburst. (2024). Centralized Data [Accedido: 4 de junio de 2024]. https://www.starburst.io/data-glossary/centralized-data/#:~:text=Centralized%20data%20is%20the%20long,company's%20single%20source%20of%20truth.

Stonefly. (2024). Understanding Scalability in Data Storage [Accedido: 5 de junio de 2024]. https://stonefly.com/blog/understanding-scalability-in-data-storage/#:~:text=Scalability%20ensures%20that%20your%20acquired,worry%20about%20additional%20capacity%20requirements.

Stratio. (2024). Stratio: Home [Accedido: 30 de mayo de 2024]. https://www.stratio.com/home

Surge, Q. (2024). ETL Testing [Accedido: 25 de mayo de 2024]. https://www.querysurge.com/solutions/etl-testing?utm_source=QS-Site&utm_medium=web-page&utm_campaign=ETL-Testing-Everything-you-need-to-know&msclkid=13eafd3244f017869d6a9b

Tableau. (2024). Tableau [Accedido: 30 de mayo de 2024]. https://www.tableau.com/es-es

TapData. (2023). Mastering ETL Processes with SQL Server: A Comprehensive Guide [Accedido: 28 de mayo de 2024]. https://tapdata.io/articles/mastering-etl-processes-with-sql-server-a-comprehensive-guide/

WELCH, A. (2024). Common Table Expressions: When and How to Use Them [Accedido: 25 de mayo de 2024]. https://www.atlassian.com/data/sql/using-common-table-expressions