UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

NUEVO MODELO DE DATOS E IMPLEMENTACIÓN DE SUBFLUJOS EN PROCESO ETL PARA AUMENTAR LA EFICIENCIA OPERATIVA EN NEOSOFT

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERA CIVIL INDUSTRIAL

ALEXANDRA IGNACIA LÓPEZ ESPINOZA

PROFESORA GUÍA:
ROCÍO RUIZ MORENO

MIEMBROS DE LA COMISIÓN:
ANDRÉS GORMAZ CANAVE
FELIPE VILDOSO CASTILLO

SANTIAGO DE CHILE
2024

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERA CIVIL INDUSTRIAL
POR: ALEXANDRA IGNACIA LÓPEZ ESPINOZA
FECHA: 2024
PROF. GUÍA: ROCÍO RUIZ MORENO

NUEVO MODELO DE DATOS E IMPLEMENTACIÓN DE SUBFLUJOS EN PROCESO ETL PARA AUMENTAR LA EFICIENCIA OPERATIVA EN NEOSOFT

El proyecto desarrollado se enmarca en la empresa tecnológica chilena NeoSoft, especializada en la generación y validación de reportes regulatorios para instituciones financieras. Uno de los servicios ofrecidos por esta empresa es GenSIID, que se encarga de la generación del reporte SIID. Este corresponde a un repositorio de transacciones de derivados (como forward, cross, opciones, entre otros) que las entidades financieras deben informar. La exigencia de este reporte proviene del Banco Central, una entidad autónoma responsable de dirigir la política monetaria del país.

En este contexto, se han identificado deficiencias en el modelo de datos y en el proceso ETL (Extract, Transform, Load) utilizados para generar los reportes de los clientes. Específicamente, el modelo de datos presenta tablas repetidas y sin uso, mientras que el proceso ETL muestra un alto grado de redundancia. Estas deficiencias resultan en un tiempo elevado de implementación de nuevos clientes y en dificultades para mantener el proceso de generación. De esta forma, la oportunidad detectada corresponde a la falta de eficiencia operativa dentro de GenSIID. Así, el objetivo del proyecto es crear un nuevo modelo de datos que organice mejor la información e implementar subflujos en el proceso ETL para optimizar su rendimiento.

El enfoque metodológico incluye, en primer lugar, la recopilación de información sobre el proceso actual. A continuación, se formulan propuestas para solucionar los problemas identificados, tanto en el modelo de datos como en el proceso ETL. Finalmente, una vez aprobadas las propuestas por el equipo, se implementan en un ambiente controlado para asegurar la calidad antes de su implementación en un entorno productivo.

Respecto al modelo de datos, se logra eliminar la redundancia de información, se simplifica la estructura de la base de datos y se obtiene un modelo más generalizado. Incluso, el número total de tablas se reduce en un 42 %, disminuyendo también el uso de la memoria en un 28 %. Respecto al proceso ETL, se logra reducir el número de procesadores en un 60 %.

Con esto se concluye que las soluciones implementadas en el ambiente de desarrollo mejoraron significativamente la eficiencia operativa de NeoSoft, dado que facilita el mantenimiento y la incorporación de nuevos clientes al sistema. A su vez, esto permite aumentar la escalabilidad de GenSIID. También, el tiempo dedicado a la generación de reportes disminuye, lo que permite que el equipo se enfoque en tareas más valiosas. Finalmente, el tiempo necesario para integrar a un cliente disminuye de 5 a 3 meses, lo que permite brindar un servicio más inmediato y oportuno, mejorando la experiencia del cliente y aumentando la presencia en el mercado de NeoSoft.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

1. Antecedentes generales

NeoSoft es una empresa informática chilena fundada el año 1997, que actualmente cuenta aproximadamente con 50 empleados y presencia en cinco países latinoamericanos. Combina de manera integral el conocimiento normativo y técnico, dedicándose a la generación y validación de informes regulatorios específicamente diseñados para instituciones financieras. De esta manera, NeoSoft cuenta con una amplia gama de clientes, sumando un total de 30 empresas (Neosoft, 2021). La competencia en el mercado de esta empresa corresponde a cualquier otra consultora tecnológica que esté dispuesta a entregar los mismos servicios de asesoría que entrega NeoSoft. Además, las mismas entidades financieras pueden generar sus propios reportes, por lo que los clientes también pueden ser vistos como competencia. Debido a que la consultoría es un área muy explotada, la presencia en el mercado de NeoSoft no es muy importante (menor al 1 %), sin embargo, se posiciona como una empresa líder gracias a su enfoque especializado en un área compleja, donde la comprensión de los conceptos financieros es fundamental. Cuenta con equipos especializados en este tipo de normativas, lo que se materializa en una ventaja respecto a las otras empresas tecnológicas.

El Banco Central de Chile, como una entidad autónoma y constitucionalmente establecida, tiene la responsabilidad de dirigir la política monetaria del país y garantizar la estabilidad y eficiencia del sistema financiero. En el ejercicio de su autoridad, ha implementado el Sistema Integrado de Información sobre Derivados (SIID o SIID-TR), que funciona como un repositorio de transacciones de derivados[1], siguiendo estándares internacionales. Las instituciones financieras residentes en Chile, incluyendo bancos y otras entidades supervisadas por la Comisión para el Mercado Financiero (CMF) tienen la obligación de reportar dichas transacciones a esta plataforma (Banco Central de Chile, s.f.).

Dentro de este contexto, NeoSoft da origen al proyecto GenSIID, un servicio dirigido a las entidades financieras chilenas para generar los reportes que deben entregar al SIID del Banco Central. A continuación se describen los reportes que se deben emitir de forma diaria (DIR, DFX y DFI) y mensual (MIR, MFX y MFI).

- Diario Tasa (DIR) / Mensual Tasa (MIR): se reportan los contratos de derivados sobre tasas de interés, es decir, instrumentos financieros cuyo valor depende de cómo cambian las tasas de interés en el mercado.
- Diario Moneda (DFX) / Mensual Moneda (MFX): se reportan los contratos de derivados sobre monedas o tipos de cambio (incluyendo los CLP/UF).
- Diario Renta Fija (DFI) / Mensual Renta Fija (MFI): se reportan los contratos de derivados sobre instrumentos de renta fija, es decir, herramientas de inversión con pagos de intereses y retornos de capital establecidos.

Es importante mencionar que los tipos de reportes que las entidades financieras deben emitir varían según sus actividades, por lo tanto, no todas emiten los mismos reportes.

[1] Instrumento financiero cuyo valor se deriva del valor de otro activo subyacente. Algunos ejemplos son los futuros, swaps y opciones.

2. Descripción de la oportunidad

NeoSoft, dentro de sus años de trayectoria, ha tenido un desempeño que ha logrado atraer y fidelizar a varios clientes. Sin embargo, recientemente han identificado algunas oportunidades de mejora en ciertos proyectos. Es importante para la empresa abordar y resolver estos desafíos con el fin de alcanzar la eficiencia, es decir, generar los productos para los clientes minimizando tanto el esfuerzo como el tiempo requerido.

Actualmente, el proyecto GenSIID de NeoSoft cuenta con cinco clientes: Banco BCI, BCI BAM, BCI Corredora, Banco Ripley y Coopeuch. La información necesaria para generar los reportes se almacena es una base de datos, una herramienta que recopila y organiza grandes volúmenes de datos (Microsoft, s.f.). Esta se encuentra en MariaDB, una plataforma que organiza los registros en grandes estructuras llamadas esquemas, los que a su vez contienen tablas donde se almacenan los datos (MariaDB, s.f.). En GenSIID se usan los siguientes siete esquemas.

- *bc*: el nombre del esquema hace referencia al Banco Central. Contiene información sobre el funcionamiento de GenSIID.
- *db_tabgral*: su nombre proviene de "tablas generales". Esta contiene información sobre el formato que deben cumplir los reportes generados.
- *banco_bci*, *bci_bam*, *bci_corredora*, *banco_ripley* y *coopeuch*.

Los últimos cinco contienen información sobre cada cliente y están compuestos por tablas auxiliares que contienen tanto datos crudos como transformados, tablas con información antigua sobre el formato de los reportes, tablas diccionario con la definición de variables y tablas que contienen información sobre los reportes diarios y mensuales. Los nombres de estos esquemas hacen referencia a cada uno de los clientes.

Al realizar una primera exploración es posible notar que ninguno de estos esquemas tiene sus tablas relacionadas, es decir, todas son independientes entre sí. Además, se logra observar que hay varias tablas que no se usan, hay algunas que se encuentran duplicadas con el esquema *db_tabgral* y hay otras que son comunes entre los esquemas de los clientes, es decir, se repiten en *banco_bci, bci_bam, bci_corredora, banco_ripley y coopeuch*. Para evidenciar esto de forma más concreta, en la Tabla 1 se muestran los porcentajes de tablas repetidas y sin usar en cada esquema.

Todos estos hallazgos corresponden a una oportunidad de mejora para el modelo de datos.

Tabla 1: Porcentaje de tablas repetidas y sin usar por esquema.

Esquema	**Tablas Repetidas (%)**	**Tablas Sin Usar (%)**
bc	0	40
db_tabgral	0	0
banco_bci	49.6	5.7
bci_bam	89.7	0
bci_corredora	67.5	0
banco_ripley	71.3	5.7
coopeuch	61.3	10.4

Adicionalmente, se realiza un proceso ETL (Extract, Transform, Load) para trabajar esta información y luego generar los reportes. Este proceso es fundamental en la gestión de datos, donde se realiza la extracción de la data desde una o varias fuentes, se transforman según las necesidades requeridas, y finalmente, se carga el resultado a una base para su posterior análisis y uso (Amazon Web Services, 2023). Este proceso se realiza a través de Apache NiFi, un software que permite una buena gestión y procesamiento de flujos de datos. Es especialmente útil en entornos de Big Data donde se requiere una solución robusta para el movimiento y procesamiento de grandes volúmenes de información en tiempo real. La componente básica de esta plataforma son los procesadores, que realizan tareas específicas en el flujo de datos, como la ingestión, transformación o enriquecimiento de datos (Fernandez, s.f.). Para simplificar, pueden ser vistos como funciones, donde ingresan datos crudos, se hace algún cambio y luego salen datos transformados. Además, se conectan entre sí para formar un flujo de trabajo que efectúa cambios sucesivos, comenzando desde la información cruda proporcionada por los clientes y culminando en los datos necesarios para generar los reportes.

A continuación se muestra una secuencia de cuatro diagramas del actual flujo de Banco BCI. Se utiliza este cliente como ejemplo, siendo los flujos del resto de los clientes completamente análogos.

Figura 1: Diagrama 1 del flujo actual.
Fuente: Elaboración propia.

En la figura anterior se muestra el grupo de procesadores destinados a Banco BCI. En su interior se encuentran todos los procesadores que se muestran en la Figura 2.

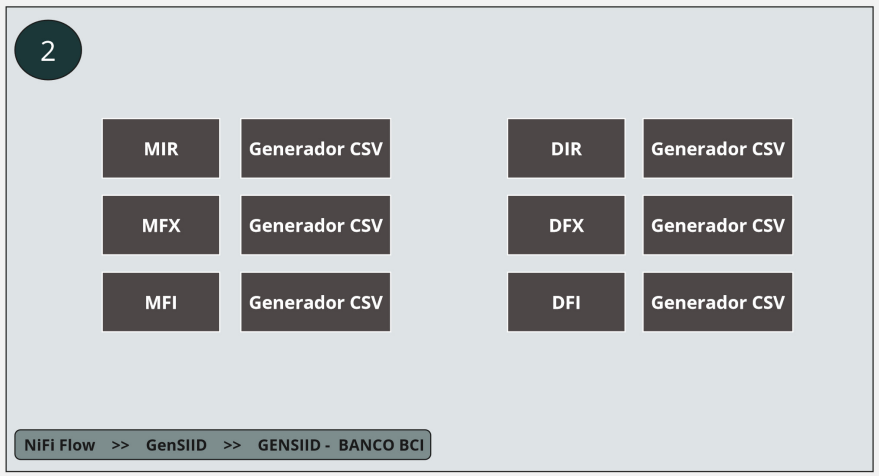


Figura 2: Diagrama 2 del flujo actual.
Fuente: Elaboración propia.

En la Figura 2 se puede observar que existen seis grupos de procesadores, uno para cada tipo de reporte (MIR, MFX, MFI, DIR, DFX y DFI). Dentro de cada uno de ellos se desarrollan todos los procesos necesarios para generar los respectivos reportes (en la siguiente figura se detallan estos procesos). Además, el grupo de procesadores para descargar los reportes en formato CSV se encuentra repetido seis veces (uno para cada reporte).

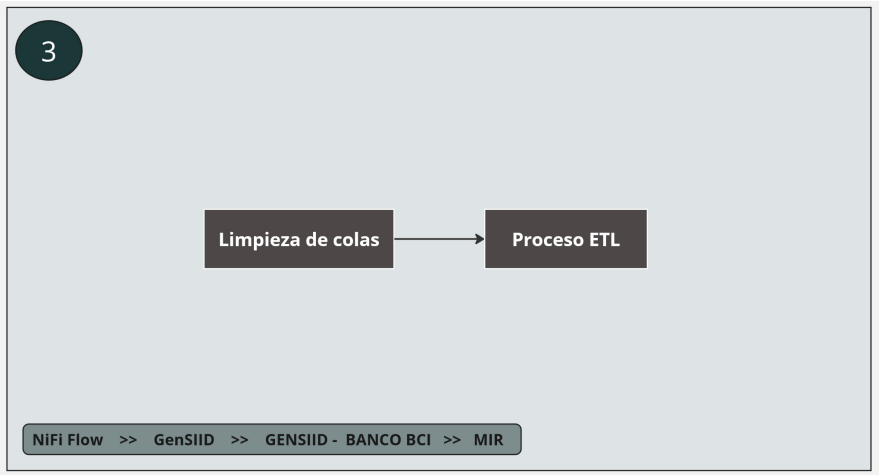


Figura 3: Diagrama 3 del flujo actual.
Fuente: Elaboración propia.

En Apache NiFi, las colas son estructuras utilizadas para almacenar temporalmente los flujos de datos que se procesan a través del sistema, es decir, los datos entrantes y salientes de los procesadores. Es importante limpiarlas regularmente para evitar la acumulación de datos innecesarios y garantizar que solo la data relevante se procese y almacene en el sistema.

En la Figura 3 se muestra el interior del grupo “MIR”, donde se observa que existe un grupo de procesadores destinado a la limpieza de colas dentro del flujo y otro grupo para

realizar el proceso ETL (en la siguiente figura se detallan estos procesos). Los grupos de procesadores del resto de los reportes son completamente análogos, por lo tanto, el diagrama mostrado se encuentra seis veces repetido.

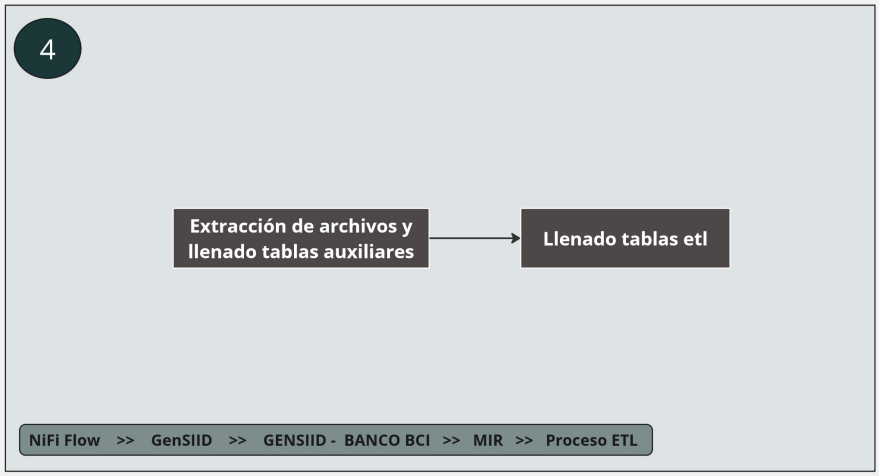


Figura 4: Diagrama 4 del flujo actual.
Fuente: Elaboración propia.

La Figura 4 muestra los grupos de procesadores que se encuentran al interior de "Proceso ETL". Se observa que se realizan dos procesos de llenado, uno para las tablas auxiliares, donde se almacenan los datos crudos, y otro para las tablas etl, donde se guardan los datos transformados. Nuevamente, estos grupos de procesadores se encuentran repetidos para todos los tipos de reportes, es decir, se repiten seis veces.

Al analizar este flujo, es posible notar que presenta redundancias, con procesadores repetidos para cada cliente, llegando al punto extremo de modificar hasta 35 procesadores distintos para implementar un solo cambio, lo que además incrementa la probabilidad de cometer un error humano. Además, las configuraciones de estos procesadores se definen directamente en cada una de estas componentes, es decir, se realizan de forma manual.

Lo anterior implica la existencia de dos grandes problemas:

1. Difícil mantención del proceso ETL: ante la necesidad de realizar algún cambio en la generación de alguno de los reportes, debe replicarse tantas veces como clientes haya. Del mismo modo, cualquier corrección para un cliente en particular implica intervenir en los seis reportes.

2. Alto tiempo de implementación de un nuevo cliente: debido a que existen muchos procesadores repetidos para cada reporte y que además estos se configuran de forma manual, se necesitan realizar modificaciones manuales en múltiples lugares del flujo para lograr un proceso ETL que logre transformar los datos de un nuevo cliente exitosamente, impactando negativamente en el tiempo de implementación.

Una de las razones detrás de esto es la falta de experiencia dentro del equipo de NeoSoft, lo que resulta en la creación de sistemas poco óptimos. Sin embargo, actualmente esta problemática está siendo abordada mediante capacitaciones y cursos.

Adicionalmente, en GenSIID existe la oportunidad de automatizar los pasos productivos y las tareas sencillas, lo que puede ahorrar una cantidad importante de tiempo para realizar tareas mecánicas. Esta situación está siendo abordada en sus etapas iniciales mediante la implementación de Jenkins, una herramienta de código abierto que se emplea para automatizar la integración y entrega continua en el desarrollo de aplicaciones. Permite a los equipos de desarrollo automatizar tareas como la compilación, las pruebas y el despliegue de software de manera eficiente.

Considerando lo anterior, la oportunidad detectada corresponde a la falta de eficiencia operativa dentro de GenSIID. Es importante recordar que eficiencia se entiende como la capacidad de generar productos para los clientes minimizando tanto el esfuerzo como el tiempo requerido. En la siguiente figura se resumen las causas ya descritas.

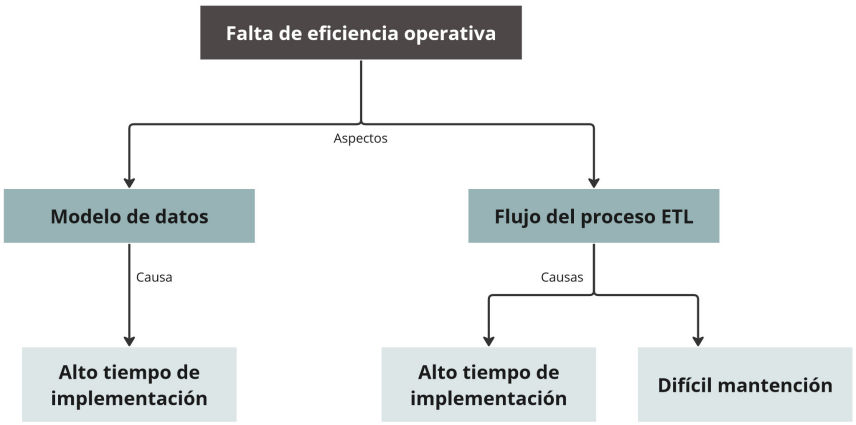


Figura 5: Causas de la oportunidad detectada.
Fuente: Elaboración propia.

Uno de los dolores más directos que se origina de este problema corresponde a los costos adicionales en términos de tiempo y recursos humanos, dado que debe existir una dedicación constante para corregir errores, mantener los modelos de datos y resolver cualquier tipo de problema que pueda surgir dentro de estos procesos. Adicionalmente, esto se encuentra altamente relacionado con el desperdicio de recursos en actividades que podrían haberse evitado al contar con procesos más eficientes y optimizados, eliminando la oportunidad de desarrollar nuevas actividades que conlleven resultados positivos para NeoSoft.

Otra consecuencia o dolor importante para la empresa es la dificultad para escalar GenSIID. Con el actual modelo de datos y flujo del proceso ETL, la integración de un nuevo cliente puede tomar hasta cinco meses en implementar, lo que resulta en la entrega de un servicio poco inmediato. Esto puede dañar la reputación de la empresa en el mercado, lo que dificulta la adquisición de nuevos clientes. Un ahorro de tiempo en este aspecto permitiría integrar a dos clientes en el mismo período que actualmente se necesita para integrar a uno solo. Además, el equipo podría enfocarse en otras actividades esenciales como la mejora continua de los servicios entregados por NeoSoft, la atención personalizada a los clientes actuales y el desarrollo de nuevas funcionalidades.

3. Descripción y justificación del proyecto

Considerando la información expuesta en las secciones previas, el proyecto a realizar se enfoca en la causa principal que origina este problema. Así, el proyecto consiste en la creación de un nuevo modelo de datos y en la implementación de subflujos en el proceso ETL que se lleva a cabo en Apache NiFi. Esta decisión se tomó luego de discutir con el resto del equipo, identificando que estos son los cambios más significativos y necesarios para aprovechar esta oportunidad.

Es importante mencionar que el proyecto se realiza en un ambiente de desarrollo de NeoSoft. Este permite realizar todas las pruebas necesarias y así comprobar el correcto funcionamiento del modelo de datos y de los subflujos en el proceso ETL que se están implementando. Estas pruebas son fundamentales para garantizar la calidad antes de llevar el proyecto a un ambiente de producción, es decir, a un entorno real donde tanto el modelo de datos como los subflujos en el proceso ETL son implementados para los clientes, utilizando datos reales y actualizados.

3.1. Nuevo modelo de datos

La creación de un nuevo modelo brinda la oportunidad de organizar de mejor forma los datos en distintos esquemas, agrupando las tablas apropiadamente de acuerdo a la información que proporcionan. A su vez, al realizar esto es posible relacionar los datos, permitiendo realizar consultas complejas y obtener información valiosa.

También se puede realizar un filtro e incluir solo los datos que efectivamente se usan para generar los reportes, dejando de lado las tablas que ya no se usan y los datos repetidos. De acuerdo a la Tabla 1, se puede evidenciar que la mayoría de los esquemas de los clientes tienen más de la mitad de sus tablas repetidas, llegando hasta el 89.7 % . Además, en la misma tabla se observa que cuatro de los esquemas cuentan con tablas sin usar, alcanzando el 40 % en el caso de *bc*. Abordar estos cambios permite reducir la magnitud de la base de datos y así optimizar el uso de la memoria.

Actualmente, hay tablas que se encuentran repetidas en todos los esquemas de los clientes debido a su uso generalizado. Esto se debe a que la mayoría son tablas diccionario, es decir, almacenan la definición de ciertas variables y conceptos. Fueron creadas con el propósito de guardar información universal respecto a la generación de los reportes, como los formatos que deben seguir. En un nuevo modelo se puede plantear un único esquema que contenga una sola vez esta información que se encuentra repetida, lo que simplifica la estructura de la base de datos y mejora la eficiencia.

Esta solución resulta beneficiosa ya que al integrar un nuevo cliente a GenSIID, se elimina la necesidad de crear todas las tablas desde cero, lo que reduce significativamente el tiempo de implementación. De esta manera, se pueden dirigir los esfuerzos únicamente a la creación de las tablas que almacenan los datos crudos enviados por los clientes y que contienen información específica de ellos. Esto también permite aumentar la escalabilidad de GenSIID, ya que se puede obtener un modelo de datos más genérico y menos enfocado a cada uno de los clientes.

Para realizar esto, es necesario recopilar información de la situación actual, especificando qué tablas están repetidas y, entre ellas, cuáles son las que realmente se utilizan. Además, se debe analizar el contenido de cada tabla para agruparlas según su temática e identificar aquellas que no son usadas, con el fin de eliminarlas del modelo. Con esto claro, se puede plantear una propuesta que aborde todos los aspectos mencionados anteriormente.

3.2. Subflujos en el proceso ETL

Uno de los principales obstáculos para lograr una eficiencia operativa es el flujo actual de Apache NiFi que realiza el proceso ETL. Esto se debe a dos razones: en primer lugar, la complejidad de mantener dicho flujo, y en segundo lugar, el tiempo considerable que lleva implementarlo cuando se integra un nuevo cliente.

Mantención

El hecho de que existan múltiples procesadores repetidos dificulta la tarea de mantener actualizado el flujo ante cualquier cambio dictado por el Banco Central sobre la generación de los reportes, la corrección de eventuales errores en el flujo o cualquier tipo de cambio que se desee realizar, ya que se deben modificar muchos procesadores para aplicar un solo cambio.

Un subflujo es un grupo de procesadores y conexiones que se pueden crear y configurar como una unidad dentro de un flujo principal más grande, aplicando cambios sucesivos en la data cruda. En palabras simples, es un flujo dentro de otro. Se usan principalmente para organizar y encapsular componentes con el fin de reutilizarlos, facilitando la gestión y el mantenimiento de flujos complejos.

Considerando lo anterior, implementar subflujos en el proceso ETL de GenSIID permite alcanzar tres importantes aspectos que conllevan a facilitar la mantención del flujo.

Primero, se puede organizar de mejor forma el proceso ETL que se está llevando a cabo, ya que se pueden agrupar los procedimientos que son comunes para cada reporte. Por ejemplo, en la Figura 2, se puede encapsular el grupo de procesadores “Generador CSV”, ya que es exactamente el mismo proceso para los seis reportes. Esto permitiría reducir este grupo de procesadores de seis a uno, ya que el subflujo común se estaría reutilizando por todos los reportes.

Segundo, a partir de lo anterior también se reduce el número de procesadores. Esto simplifica la tarea de realizar modificaciones en el proceso ETL, puesto que en lugar de tener que ajustar seis procesadores para implementar un solo cambio, bastaría con configurar un único procesador para llevar a cabo la misma tarea.

Tercero, se simplifica la arquitectura del proceso que está ocurriendo, obteniendo un sistema mucho más ordenado y entendible. Además, esto facilita la manipulación del flujo para cualquier miembro del equipo GenSIID.

Tiempo de implementación

Desarrollar un nuevo flujo de datos para un nuevo cliente que se esté integrando a GenSIID toma un tiempo de cinco meses aproximadamente, lo que se traduce en la entrega de un servicio poco inmediato. La causante principal de esto es que las configuraciones de los procesadores se definen directamente en cada una de estas componentes, obligando a tener que modificar la mayoría de los procesadores para poder ingresar los datos de un nuevo cliente.

Para abordar esta situación es crucial parametrizar el flujo, es decir, definir variables que almacenen esta información al inicio del procesamiento. Luego, durante el desarrollo del proceso ETL, estas variables se utilizan para acceder a la información guardada. Así, en caso de que surja la necesidad de modificar algún dato, basta con actualizar una vez la variable correspondiente, eliminando la necesidad de configurar individualmente los procesadores donde se utiliza dicho dato. Esto simplifica significativamente el proceso de ajuste y mantenimiento del flujo.

Con esta propuesta, se obtiene un flujo de datos escalable a otros clientes, dado que permite la aplicación rápida y sencilla de los cambios necesarios para integrar nuevas instituciones. También disminuye la probabilidad de cometer errores, ya que el cambio se realiza una sola vez en la definición de la variable. Del mismo modo, evita la existencia de inconsistencias, ya que las configuraciones comunes estarán sincronizadas en todos los procesadores.

Para realizar esto, es necesario recopilar información de la situación actual, distinguiendo las partes del proceso que son comunes para todos los reportes y que se pueden consolidar en un único grupo de procesadores. También es importante identificar cuál es la información que se puede parametrizar. Una vez definido esto, se puede plantear una propuesta de subflujos que logre solucionar los problemas ya descritos.

4. Objetivo general

Para abordar esta oportunidad se plantea el siguiente objetivo general: “Reestructurar el modelo de datos e implementar subflujos en el proceso ETL para aumentar la eficiencia operativa del Proyecto GenSIID de NeoSoft”.

5. Objetivos específicos

Para lograr el objetivo general, se plantean los siguientes objetivos específicos.

1. Analizar la estructura actual del modelo de datos y del flujo del proceso ETL de GenSIID, identificando áreas de redundancia, ineficiencia y oportunidades de mejora.
2. Diseñar un nuevo modelo de datos y desarrollar subflujos en Apache NiFi para optimizar los procesos dentro de GenSIID y facilitar la integración de nuevos clientes.
3. Implementar el nuevo modelo de datos y los subflujos en un entorno controlado, asegurando su funcionalidad y compatibilidad con los requisitos de GenSIID.
4. Desarrollar un plan de implementación que logre gestionar el conocimiento dentro del equipo.

6. Alcances

El proyecto busca establecer un sistema genérico y adaptable ante cualquier cambio futuro, simplificando considerablemente la mantención del flujo y permitiendo que el equipo GenSIID se enfoque en otras tareas y proyectos. A continuación se mencionan algunos impactos.

- Se reduce el recurso necesario de las máquinas virtuales, ahorrando el valor asociado a estas suscripciones.
- Disminuye el tiempo de implementación de un nuevo cliente. Anteriormente, esto se lograba entre 4 y 6 meses, pero ahora se puede reducir a 3 meses.
- Se elimina la necesidad de contratar a un profesional para tareas de mantenimiento y de implementación, ahorrando el valor asociado a un salario.

Respecto al plan de implementación, este se llevará a cabo en un ambiente de desarrollo, es decir, un espacio controlado para realizar pruebas y garantizar la calidad. Sin embargo, se espera que en un futuro cercano se desarrolle en un ambiente productivo de NeoSoft. Este corresponde al espacio que es realmente usado por los clientes, por lo que debe funcionar de manera estable, segura y eficiente. En este sentido, la meta o indicador de éxito radica en lograr la completa implementación del modelo de datos y de los subflujos en el proceso ETL, lo que permitirá integrar a cualquier cliente que se sume a GenSIID de manera eficiente.

De esta forma, el proyecto no se centra en la calidad de los informes generados, es decir, se espera que los reportes no cambien en su contenido y formato, y que mantengan su calidad actual. Sin embargo, se busca mejorar la eficiencia en su generación mediante la optimización del modelo de datos y del proceso ETL. Otro aspecto que el proyecto no abarca es el análisis de los reportes generados y la posterior toma de decisiones en base a los resultados. El alcance del proyecto finaliza una vez que el reporte generado se guarda en el directorio de NeoSoft. Posteriormente, una vez que el informe esté listo, se envía a los clientes, quienes son responsables de utilizar esta información para llevar a cabo el análisis y la toma de decisiones pertinentes en sus respectivas organizaciones.

Respecto a las decisiones estratégicas que se pueden tomar a partir del proyecto, se decide que, por el momento, la solución desarrollada no será implementada en otros servicios fuera de GenSIID. Esto se debe a que es necesario realizar un análisis exhaustivo sobre los requerimientos específicos de los otros servicios proporcionados por NeoSoft y evaluar si la implementación de esta solución realmente aportaría beneficios a dichos servicios.

A pesar de las limitaciones mencionadas, el proyecto aborda aspectos importantes relacionados con los recursos humanos y tecnológicos. Se escoge esta solución por sobre otras por dos grandes razones. La primera de ellas corresponde a la flexibilidad y adaptabilidad que le brinda a GenSIID, permitiendo que cambios futuros se manejen con facilidad. Dado esto, es posible afirmar que esta solución no solo se hace cargo de un problema actual, sino que también favorece la tarea de generar reportes en el futuro. La segunda gran razón por la que se escoge esta solución es porque no tiene costos asociados, puesto que el proyecto se desarrolla a partir de los recursos con los que GenSIID ya cuenta. Adicionalmente, la solución propuesta fue discutida con el resto del equipo, reuniendo varios puntos de vista y llegando al consenso de que el mejor enfoque para abordar el problema es este.

7. Marco conceptual

Para la realización de este proyecto se utilizan las siguientes herramientas.

7.1. Base de datos en MariaDB

Una base de datos es un sistema que está hecho para organizar y recopilar información de forma rápida y segura, permitiendo realizar consultas complejas. Pueden almacenar información variada, como por ejemplo sobre personas, clientes, productos u otras cosas (Microsoft, s.f.). El uso de esta herramienta es fundamental para las organizaciones en la actualidad, ya que los datos desempeñan un papel crucial, permitiendo aprender y descubrir aspectos clave del negocio. Así, estas estructuras sustentan las operaciones internas de las empresas, al almacenar tanto información administrativa como datos más específicos, lo que asegura la integridad y protección de los datos (Amazon Web Services, 2023). Inlcuso, Garrido et al. (2020) considera que las bases de datos son una componente vital en todo sistema en la actualidad.

MariaDB es un sistema que permite crear, gestionar y acceder a estas bases usando un lenguaje de programación llamado SQL. Los datos se organizan en esquemas que contienen tablas con todos los registros que se desean almacenar, presentados en forma de filas y columnas (MariaDB, s.f.). Una de sus desventajas es la baja compatibilidad con MySQL, puesto que cuenta con algunas diferencias que dificultan la migración entre ambas bases de datos. Sin embargo, esto no significa un problema para el proyecto, ya que no se realizan migraciones.

Se decide usar este sistema porque cuenta con variados beneficios. Uno de estos es que es de código abierto, es decir, se puede usar de forma gratuita. También cuenta con constantes mejoras de rendimiento y seguridad para proteger los datos confidenciales. Además, es el sistema que se usa actualmente en NeoSoft, lo que es una ventaja, puesto que garantiza la congruencia del proyecto con las aplicaciones que ya están en uso, facilitando la integración y compatibilidad entre las diferentes componentes del sistema.

7.2. Proceso ETL (Extract, Transform, Load)

El proceso ETL corresponde a la extracción de archivos desde diferentes fuentes de origen, a la transformación de los datos que están contenidos en ellos y a la carga del resultado en una base de datos o repositorio. Su objetivo principal es facilitar el movimiento de los datos y la transformación de los mismos, siendo muy útil cuando se requiere modificar un gran volumen de datos de forma automatizada y rápida para su posterior análisis (PowerData, s.f.).

A pesar de lo anterior, su principal desventaja es la dificultad para manejar y mantener el proceso en entornos de datos grandes. Ante esto, se puede dividir el proceso ETL en módulos o subprocesos más pequeños, con el fin de hacerlos más sencillos y manejables. Otra consecuencia corresponde al efecto negativo en el rendimiento del sistema, ya que las cargas de datos son lentas. Una forma de abordar esto es implementar técnicas de procesamiento paralelo para distribuir la carga.

Este proceso ofrece una serie de beneficios significativos. Primero, permite la consolidación de datos provenientes de múltiples fuentes, lo que facilita una visión integral y coherente de la información. Además, al limpiar y transformar la data, se mejora su calidad y confiabilidad, lo que conduce a una toma de decisiones más informada y precisa. También, una vez que los registros están alojados en su ubicación definitiva, fácilmente pueden ser utilizados en otros sistemas operacionales, lo que potencia y optimiza los procesos de negocio (PowerData, s.f.). Además, tal como indica Conde (2022), un ETL reporta una gran cantidad de beneficios, incluyendo la gobernabilidad de los datos y la generación de documentación apta para la toma de decisiones.

Actualmente, en GenSIID ya se realiza un proceso ETL para generar los reportes de los clientes y corresponde a un aspecto fundamental, por lo tanto, es algo que se debe mantener, mejorar y potenciar. Incluso, Seenivasan (2023) señala que "Sin ETL, los datos permanecerían en silos, dispersos en diferentes sistemas y en diferentes formatos, lo que dificultaría obtener conocimientos y tomar decisiones basadas en datos" (traducción propia).

7.3. Subflujos

Apache Nifi es un software open source (gratis) que fue creado para extraer, transformar y cargar datos, es decir, desarrollar procesos ETL. La unidad básica de este programa son los procesadores, que se encargan de realizar tareas específicas en los datos, como los procesos de extracción, transformación y carga. Una ventaja de este software es que cuenta con un gran número de procesadores, abarcando también un gran número de tareas. De esta forma, se encuentran variadas alternativas para desarrollar distintos procesos. Otra de sus ventajas es que permite crear flujos de datos arrastrando y conectando procesadores, eliminando la necesidad de contar con conocimientos de programación especializados, sino que basta con entender el funcionamiento de estas componentes básicas para poder configurarlas correctamente (Fernandez, s.f.).

Su desventaja principal es la dificultad para entender la interacción entre los subflujos. Para abordarlo, se pueden utilizar nombres descriptivos para los procesadores, facilitando la comprensión de su propósito y funcionamiento. Además, se pueden añadir comentarios y anotaciones dentro de NiFi para explicar la lógica detrás del flujo.

De esta forma, un flujo es lo que se obtiene al conectar distintos procesadores. Dado que cada componente realiza un cambio en los datos, un flujo puede ser visto como varios cambios sucesivos. A modo de ejemplo, en el Anexo A se muestra un flujo que contiene cuatro procesadores. El primero de ellos inicia el proceso, el segundo extrae un archivo Excel, el tercero lo transforma a formato CSV y el cuarto descarga el archivo final. Se puede evidenciar que cada procesador realiza una acción y al unirlos se obtiene una serie de alteraciones que llevan a un resultado en particular, que en este caso es la descarga del archivo.

Un subflujo corresponde a un conjunto de procesadores que se encapsulan en una sola estructura para luego ser insertados en un flujo más grande. Esto posibilita la organización eficiente de los procesos que se están llevando a cabo y su reutilización para diferentes conjuntos de datos, lo que conlleva a una reducción significativa de la redundancia. Además, simplifica la gestión de flujos complejos al dividirlos en unidades más pequeñas y manejables, facilitando tanto el diseño como el mantenimiento de flujos de datos. Para ilustrar esto, en

el Anexo B se muestra una situación A, donde existen los flujos I y II. Se observa que los procesadores de inicio del flujo y de extracción de archivo están repetidos en ambos flujos y, por lo tanto, son redundantes. En el Anexo C se muestra una situación B, donde se consolida este proceso común como un subflujo. En el Anexo D se muestra el contenido de este último. Es posible notar que se elimina la redundancia al tener solo una instancia de los procesadores en común dentro del subflujo, lo que conduce a una situación más optimizada.

En el caso de GenSIID, el proceso ETL que se realiza cuenta con varios procesos en común que se repiten para generar cada uno de los seis reportes. En esta situación resulta beneficioso implementar subflujos, dado que logran consolidar estos procesos de forma única, para luego ser reutilizados por los seis tipos de reportes que se generan, evitando redundancias, simplificando la estructura del proceso y optimizando la mantención de este mismo.

7.4. Modelo Wiig

La gestión del conocimiento o Knowledge Management (KM) corresponde al proceso de identificar, organizar, almacenar y difundir información dentro de una organización. Se definen un conjunto de actividades para favorecer el intercambio de información y experiencias entre el personal de una empresa. De esta forma, se construye una base de conocimiento colectivo que busca la eficiencia operativa y potencia la innovación. Así, las organizaciones que cuentan con estrategias de gestión del conocimiento alcanzan resultados comerciales rápidamente, ya que la toma de decisiones se ve facilitada por el aprendizaje organizacional y por la colaboración entre los miembros del equipo (IBM, s.f.). Cuenta con múltiples beneficios, como la identificación de áreas de mejora, la toma de decisiones más informadas, la eficiencia operativa, una mayor colaboración y comunicación, entre otros.

Dalkir (2011) describe el Modelo Wiig para crear y usar el conocimiento, fundado en 1993. Este se basa en el principio de que el conocimiento debe estar organizado para ser útil y valioso. Destaca que debe organizarse de manera diferente según su uso, definiendo las siguientes tres formas de conocimiento: público, compartido y personal. También define los siguientes cuatro tipos de conocimiento: factual, conceptual, expectacional y metodológico.

Para NeoSoft resulta altamente beneficioso implementar el Modelo Wiig, puesto que esta metodología de gestión del conocimiento permite compartir con todo el equipo los hallazgos alcanzados en este proyecto de forma organizada. Esto incluye aspectos relacionados con la implementación de la solución, las ventajas que conlleva y cómo puede resultar ventajoso en otros proyectos.

8. Metodología

Para solucionar problemas dentro de una empresa, Michael Hammer y James Champy (2001) proponen un enfoque integral de reingeniería que implica desarrollar las siguientes etapas.

1. Comprender el proceso: los equipos deben entender el proceso actual identificando qué hace, qué tan bien funciona y los problemas críticos que afectan su rendimiento.

2. Rediseño centrado en el cliente: crear un diseño que esté basado en los hallazgos encontrados, obteniendo uno superior al actual.

3. Implementación de la propuesta en proyecto piloto y escalado: comenzar con proyecto piloto para probar los nuevos procesos y hacer ajustes basados en la retroalimentación del mundo real antes de escalar a toda la organización.

De acuerdo a lo planteado por estos autores, para poder alcanzar los objetivos definidos y poder desarrollar el proyecto de la mejor forma, se proponen las siguientes etapas a realizar.

1. **Levantamiento de información sobre el modelo**
   Se reúne información sobre el modelo de datos de GenSIID. Se realizan detalladas descripciones de los esquemas, incluyendo qué función cumplen en la generación de los reportes y qué tablas contienen. A su vez, se describe la información que entrega cada una de estas tablas. La finalidad de esta etapa es alcanzar un entendimiento completo sobre la actual situación del modelo de datos.

2. **Propuesta de un nuevo modelo**
   Al tener conocimiento de la actual base de datos, es posible identificar oportunidades de mejora y trabajarlas para obtener un sistema más optimizado. Así, se consideran todos estos aspectos para redactar una propuesta formal de los cambios a realizar.

3. **Implementación del nuevo modelo**
   Una vez aprobada la propuesta por el resto del equipo, debe ser implementada. De acuerdo a los aspectos mencionados en la sección anterior, se decide desarrollar el nuevo modelo de la base de datos en MariaDB. Para lograr esto, se deben crear los nuevos esquemas y las tablas vacías que contiene cada uno de estos. Posteriormente, estas tablas se deben poblar con los datos existentes en el antiguo modelo, por lo tanto, los datos se consideran como un insumo para la implementación de esta solución.

4. **Levantamiento de información sobre el flujo del proceso ETL**
   Se recopila información sobre el actual flujo de datos, especificando los procesadores que se usan y qué función cumplen dentro del procesamiento de los datos del ETL. Con esto se busca entender a cabalidad los procesos que actualmente ocurren, incluyendo la extracción de los archivos fuente, la transformación de los datos y la carga de los reportes en formato CSV.

5. **Propuesta de implementación de subflujos**
   Una vez identificados todos los puntos a mejorar, se proponen cambios que logren solucionar estos problemas. Específicamente, se propone la implementación de subflujos para optimizar el procesamiento de datos entre reportes, evitando así la existencia de procesadores repetidos.

6. **Implementación de subflujos en el proceso ETL**
   Teniendo la propuesta aprobada por el resto del equipo, debe ser implementada en Apache NiFi. Para lograr esto, se usa el flujo antiguo como punto de partida, por lo que se considera como un insumo vital para el desarrollo de esta solución, y se realizan los cambios pertinentes para evitar redundancias y procesadores repetidos dentro del proceso ETL.

7. **Gestión del conocimiento**
   Una vez implementados los cambios, es necesario comunicar las mejoras y los conocimientos alcanzados al resto del equipo, con el fin de fomentar un ambiente de aprendizaje y colaboración. De acuerdo al Modelo Wiig, el conocimiento adquirido durante la realización de este proyecto corresponde al tipo metodológico compartido, por lo tanto, se debe realizar una serie de actividades acorde a este tipo de conocimiento, es decir, difundir los aprendizajes alcanzados a través del lenguaje y representaciones especializadas. Específicamente, se realizan las siguientes tareas.

   - Crear un manual que describa el funcionamiento del modelo de datos y del proceso ETL resultante.
   - Realizar un taller de capacitación con el fin de compartir los aprendizajes alcanzados.

9. Desarrollo y resultados

A continuación se describe el desarrollo de las etapas de levantamiento de información, la formulación de las propuestas de las soluciones y la implementación de estas, tanto del modelo de datos como de los subflujos del proceso ETL. Finalmente, se detallan las actividades relacionadas con la gestión del conocimiento dentro del equipo.

9.1. Nuevo modelo de datos

En primera instancia, se levanta información del modelo de datos actual a través de un detallado análisis para evidenciar todos los problemas que se puedan mejorar. En el Anexo E se encuentra esta información. De esto, se obtienen los siguientes hallazgos para los dos esquemas base.

- *bc*: contiene información sobre el funcionamiento de GenSIID y datos que se obtienen de la página web. Hay 31 tablas en total, donde 12 de ellas no se usan.
- *db_tabgral*: cuenta con datos sobre el formato que deben cumplir los reportes generados. Todas las tablas de este esquema se usan y son vitales para la creación de estos.

Para los esquemas de los clientes se resume la siguiente información.

Tabla 2: Información levantada para cada esquema cliente.

Esquema	**Total Tablas**	**Tablas Auxiliares**	**Tablas Sin Usar**	**Tablas Repetidas con Otros Clientes**	**Tablas Repetidas con db_tabgral**
banco_bci	123	47	7	39	22
bci_bam	39	4	0	15	20
bci_corredora	77	26	0	31	21
banco_ripley	87	19	5	39	23
coopeuch	106	29	11	42	23

Es importante mencionar que las tablas que no se usan existen porque fueron creadas en un momento dado cuando se necesitaban. Algunas fueron creadas para guardar información sobre procesos que ya no se realizan, mientras que otras fueron diseñadas para almacenar datos que no varían en el tiempo. Sin embargo, actualmente cuentan con información obsoleta que ya no es requerida para la generación de los reportes. Además, la Tabla 3 muestra la distribución de la memoria usada por los esquemas.

Para formular una propuesta, es importante considerar que un modelo de datos debe cumplir con algunos criterios. Uno de ellos es considerar los objetivos del negocio al momento de agrupar la data, puesto que es importante la forma en que los datos se relacionan con el propósito de la empresa. Segundo, diseñar un modelo que sea ajustable en el tiempo. Este aspecto es esencial para integrar los cambios relacionados al crecimiento del volumen de datos, facilitando la escalabilidad de los sistemas. Tercero, es crucial escoger la técnica de modelado adecuada para los datos disponibles, dado que la data debe ser representada de manera precisa para capturar todos los aspectos importantes. Algunos ejemplos son el Modelo Entidad-Relación (ER), Modelo Jerárquico, Modelo Relacional, entre otros (Mahler, 2023).

Tabla 3: Tamaño de los esquemas originales.

Esquema	**Tamaño (MB)**
bc	9.30
db_tabgral	0.63
banco_bci	4432.99
bci_bam	6.00
bci_corredora	6.45
banco_ripley	4.71
coopeuch	424.30
Total	**4884.38**

Considerando lo anterior, se propone un modelo de datos relacional que se centra en dos aspectos fundamentales. En primer lugar, se busca eliminar las tablas que no están en uso y reducir la redundancia de aquellas tablas que se repiten, consolidándolas en una única instancia. El objetivo de esto es lograr un modelo optimizado que utilice de manera eficiente los recursos de almacenamiento disponibles. En segundo lugar, se busca organizar de mejor forma los datos disponibles, con un enfoque hacia un modelo más generalizado que no esté centrado en cada cliente. Este enfoque tiene como fin principal la creación de un modelo de datos más flexible y escalable, que pueda adaptarse fácilmente a un nuevo cliente. Una vez aceptada esta propuesta por el resto del equipo, es implementada en un ambiente de desarrollo, es decir, un espacio controlado para realizar pruebas.

El modelo desarrollado cuenta con seis esquemas que funcionan de forma genérica. A continuación se describen los esquemas base.

- gensiid_gs: contiene toda la información relacionada con el funcionamiento de GenSIID. Se eliminan todas aquellas tablas que no se usan en *bc* y se cambia el nombre del esquema a uno más apropiado. La información contenida en este esquema permite comprender cuáles son los procesos asociados a cada institución financiera dentro del flujo. A partir de aquí, es posible extraer el estado de estos procesos y a qué procesadores están asociados, lo que lo convierte en información muy útil. Por ejemplo, si surge un error dentro del proceso ETL, se puede identificar en qué parte del flujo se está ocasionando el error, facilitando así su pronta solución. En la siguiente figura se muestra un diagrama de este esquema.

Figura 6: Diagrama del esquema gs.
Fuente: Elaboración propia.

- db_tabgral: contiene información sobre los formatos que deben cumplir los reportes. Esta información se usa para validar que los reportes generados cumplan con los estándares establecidos por el Banco Central. Este esquema no sufre ningún cambio, por lo que se mantiene idéntico al original. Cuenta con 20 tablas y dado que son aspectos variados y que no guardan relación entre sí, es que las tablas de este esquema no se relacionan.

- gensiid_web: contiene toda la información relacionada con el funcionamiento de la página web. Las tablas que conforman este esquema se extraen de *bc*. En la siguiente figura se muestra un diagrama de este esquema.

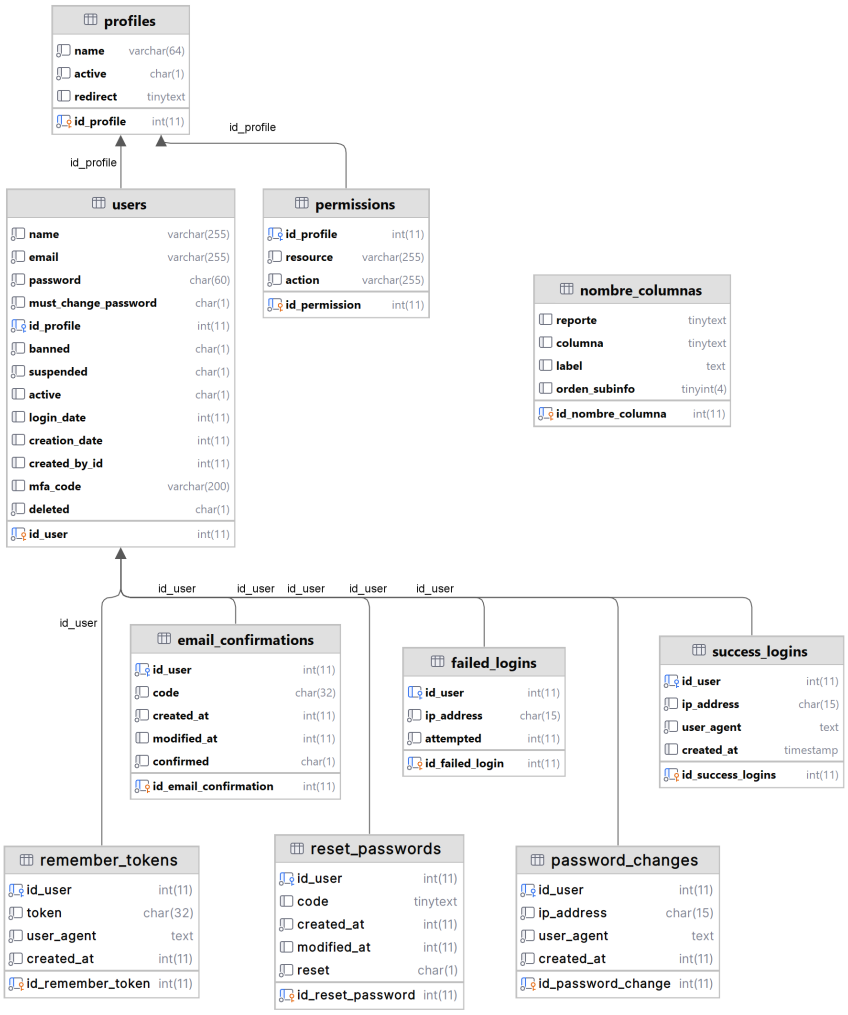


Figura 7: Diagrama del esquema web.
Fuente: Elaboración propia.

Se observa que los datos se organizan en torno a los usuarios, abarcando detalles sobre su perfil y los permisos asociados, así como registros que contienen información sobre sus contraseñas y actividades en la página web.

Tras dialogar con los demás miembros del equipo, se pudo evidenciar que es necesario disolver los esquemas de clientes (*banco_bci, bci_bam, bci_corredora, banco_ripley y coopeuch*) para lograr un modelado más genérico de los datos. Así, se crean tres nuevos esquemas que organizan toda la información en conjunto. Estos se describen a continuación.

- gensiid_dsa: compuesto por 118 tablas, este conjunto abarca todas las tablas auxiliares, así como aquellas exclusivas para cada cliente. El nombre del esquema hace referencia a un Data Staging Area (DSA), un espacio intermedio de almacenamiento donde se recopilan y transforman los datos antes de ser cargados en el almacén de datos final. Esto se debe a que se usa principalmente dentro del flujo del proceso ETL, donde se guardan los datos crudos enviados por los clientes y los datos transformados para insertar en los reportes. En este esquema, dado que se mezclan datos de distintos clientes, se cambia la nomenclatura del nombre de las tablas, donde cada una comienza con una

abreviación del nombre del cliente y se agrega “src” a aquellas tablas que contienen datos fuentes, es decir, datos crudos que no se han transformado. El objetivo de esto es poder identificar y diferenciar la información de cada institución.

- gensiid_etl: contiene 42 tablas que incluyen datos transformados con información de los reportes diarios (DFI, DFX y DIR) y mensuales (MFI, MFX y MIR). Debido a que los datos entre los reportes no se relacionan, no es necesario conectar estas tablas. Al igual que el esquema anterior, este también se usa principalmente dentro del flujo del proceso ETL, almacenando los registros necesarios para generar los reportes.

- gensiid_val: incluye las tablas que entregan información sobre la validación de los reportes generados, es decir, los procesos de revisión a los que son sometidos dichos reportes. Además, incluye algunas tablas diccionario que son usadas por todos los clientes y que guardan la definición de algunas variables o conceptos. A continuación se muestra un diagrama de este esquema.

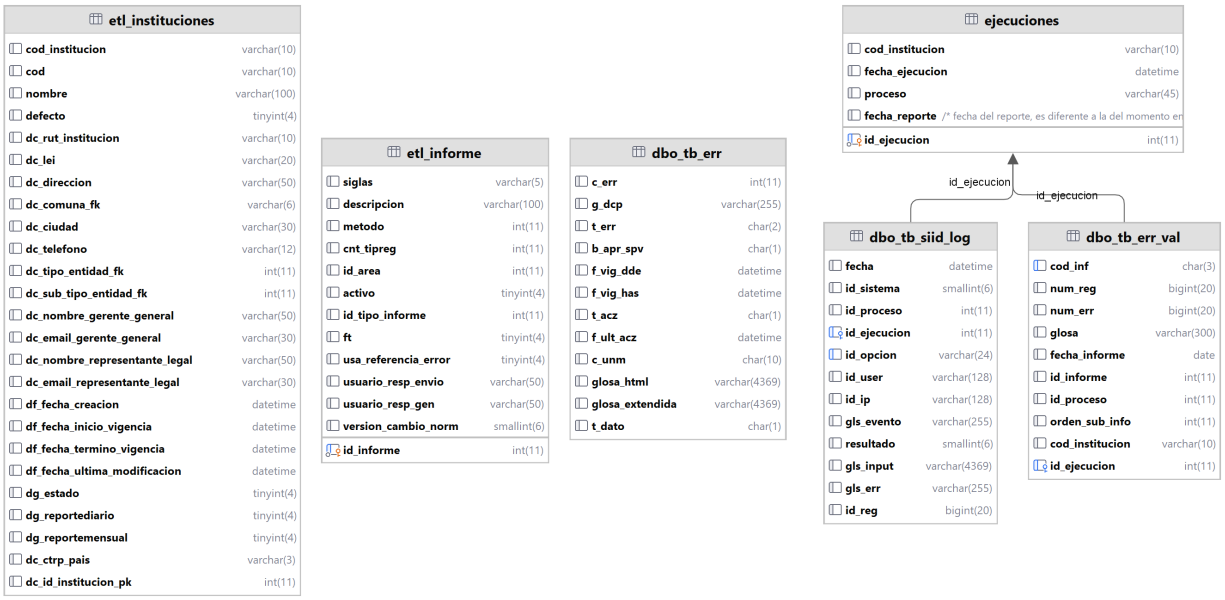


Figura 8: Diagrama del esquema val.
Fuente: Elaboración propia.

Las tres primeras tablas corresponden a diccionarios, es decir, contienen información que no varía en el tiempo, como la definición de algunos parámetros y variables. Las tres tablas que se encuentran a la derecha del diagrama se organizan por ejecuciones.

La siguiente figura compara el modelo original con el nuevo. Se observa que anteriormente los datos se agrupaban por cliente, pero con esta mejora se organizan de acuerdo a su uso. Además, es posible apreciar que las tablas específicas de cada cliente se encuentran todas juntas en el esquema gensiid_dsa y las comunes, que antes se encontraban repetidas, se unificaron en el esquema gensiid_etl.

Figura 9: Comparación entre el modelo original y el nuevo.
Fuente: Elaboración propia.

Aplicar este nuevo modelo involucra importantes beneficios, ya que optimiza el uso y la distribución de la información entregada por los clientes. En primer lugar, el modelo se hace cargo de la alta duplicidad presente en los datos. Esta surge ante la existencia de múltiples tablas idénticas en esquemas diferentes. Al consolidar estas tablas repetidas, se elimina la redundancia de información y se simplifica la estructura de la base de datos. Esto no solo mejora la eficiencia de almacenamiento, sino que también facilita el mantenimiento y la actualización de los datos.

En segundo lugar, una característica clave del nuevo modelo es su mayor nivel de generalización. Anteriormente, se mantenía un esquema separado para cada cliente, lo que podía generar complejidades de mantenimiento a medida que aumenta el número de clientes. La propuesta simplifica este enfoque al consolidar todos los registros de todos los clientes en solo tres esquemas. Esto no solo reduce la complejidad estructural, sino que también elimina la necesidad de crear y gestionar esquemas individuales para cada cliente, facilitando la administración y escalabilidad del sistema.

En tercer lugar, este modelo incorpora una estructura que establece conexiones claras entre las tablas. Esto significa que se capturan y documentan las relaciones entre diferentes conjuntos de datos. Al comprender cómo las tablas se relacionan entre sí, se pueden realizar consultas más complejas y así extraer información más significativa. Este aspecto facilita la comprensión y el análisis de la base de datos, mejorando la capacidad para responder preguntas específicas sobre los datos y permitiendo una toma de decisiones más informada para NeoSoft.

La Figura 10 ilustra el impacto del proyecto en la cantidad de tablas. Se observa una reducción significativa del número total, disminuyendo de 483 a 203. En cuanto a las tablas comunes, se ha logrado una consolidación notable, pasando de 166 a 42.

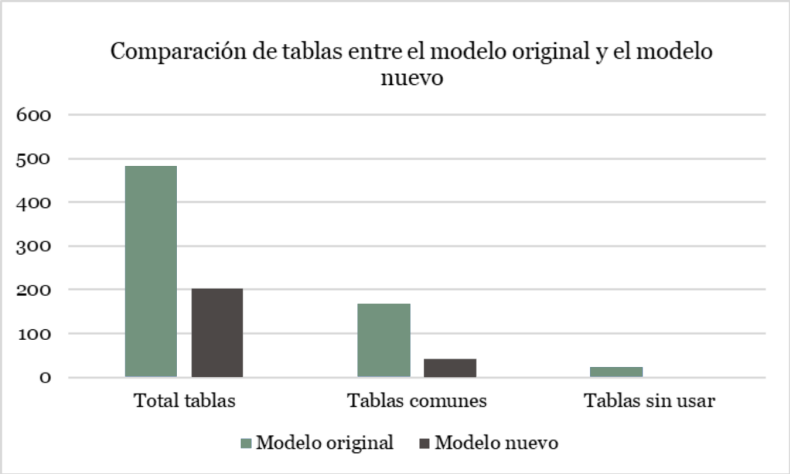


Figura 10: Comparación del número de tablas.
Fuente: Elaboración propia.

Además, la Tabla 4 muestra el uso de la memoria de este nuevo modelo de datos. Es posible notar que en comparación con el modelo antiguo, la memoria se reduce en un 28 %.

Tabla 4: Tamaño de los nuevos esquemas.

Esquema	**Tamaño (MB)**
db_tabgral	0.63
gensiid_gs	7.49
gensiid_web	0.36
gensiid_dsa	1486.62
gensiid_etl	2021.74
gensiid_val	0.25
Total	**3517.09**

Debido a que este modelo es implementado en un ambiente de desarrollo, todos los beneficios anteriormente mencionados aplican solo a este espacio. En el futuro, cuando se realice el paso a producción, será necesario aislar los datos en ambientes distintos para cada cliente, es decir, cada cliente debe contar con su propio ambiente. El fin de esto es garantizar que cada uno pueda ver únicamente sus propios datos. La Figura 11 ilustra los ambientes de desarrollo y producción antes de implementar el nuevo modelo de datos. La Figura 12 muestra estos mismos ambientes, pero después de haber implementado el modelo.

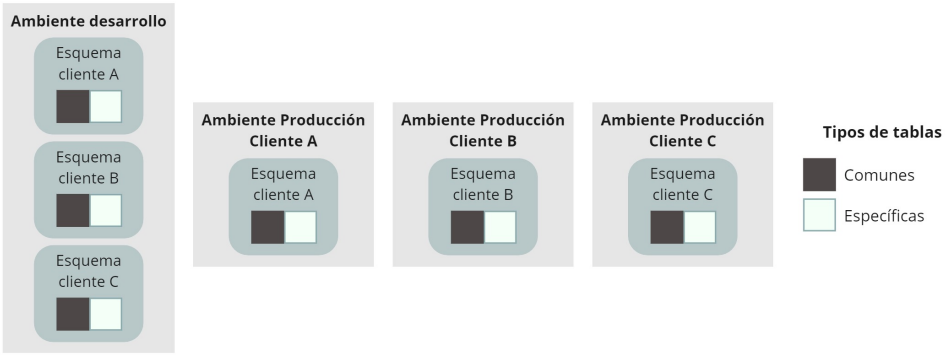


Figura 11: Ambientes de desarrollo y producción antes del proyecto.

Fuente: Elaboración propia.

Se puede observar que antes de implementar el nuevo modelo de datos, en el ambiente de desarrollo cada cliente tiene su propio esquema que contiene tanto tablas comunes como tablas específicas. En producción, se puede apreciar que además cada uno de estos esquemas está en un ambiente distinto.

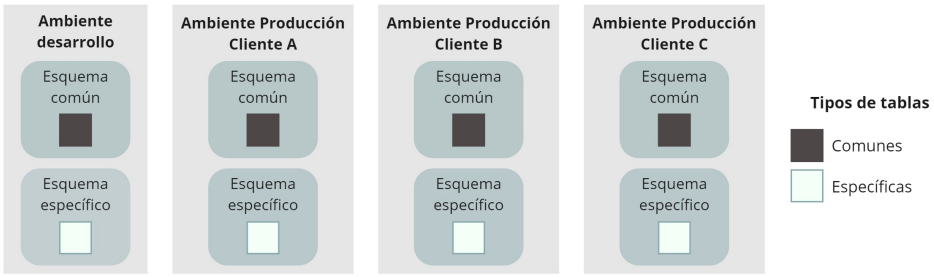


Figura 12: Ambientes de desarrollo y producción después del proyecto.

Fuente: Elaboración propia.

Al implementar el nuevo modelo de datos se observa que en el ambiente de desarrollo se unifican las tablas comunes y específicas en solo dos esquemas. Esto se replica para los ambientes de producción de cada cliente, donde cada uno cuenta con estos mismos dos esquemas.

Es posible notar que la situación en producción antes y después de haber implementado el modelo de datos no es muy distinta, ya que la única diferencia entre ambos casos es que las tablas comunes y específicas se encuentran organizadas en esquemas distintos. A pesar de lo anterior, es beneficioso tener el ambiente de desarrollo optimizado. Una de las razones es porque permite a los desarrolladores de GenSIID trabajar más rápido y con mayor eficiencia, reduciendo el tiempo de espera y los retrasos para futuras pruebas. De la misma manera, permite detectar y corregir errores de forma más rápida en la generación de los reportes. Otra razón es que favorece el proceso de mejora continua, ya que facilita la creación y prueba de nuevas características en el modelo de datos. Finalmente, un entorno optimizado también mejora la productividad del equipo de GenSIID, permitiéndoles concentrarse en la creación de valor a partir del desarrollo de otras tareas.

9.2. Subflujos en el proceso ETL

Primeramente, se realiza un levantamiento de información sobre el actual flujo del proceso ETL. En este, se describe la función que cumple cada uno de los procesadores utilizados y además se evidencia cuáles son las componentes específicas que están repetidas. En el Anexo F se resume esta información.

Durante las reuniones con el equipo fue posible notar que los reportes son personalizados para cada cliente, por lo tanto, es necesario que los subflujos a implementar en Apache NiFi se organicen de acuerdo a cada uno de ellos, es decir, que cada cliente cuente con su propio flujo y grupos de procesadores, funcionando como un sistema aislado. A partir de eso, se propone implementar subflujos en ciertas etapas del proceso ETL para eliminar la redundancia. Además, se sugiere la creación de variables que logren parametrizar el flujo para guardar información de forma más eficiente. Una vez aceptada la propuesta por el equipo, se implementan todos los cambios sugeridos. A continuación, se describe el flujo de Banco Bci a modo de ejemplo, siendo la implementación para el resto de los clientes completamente análoga.

1. Se crea un nuevo grupo de procesadores para el cliente (Diagrama 1). En su interior se encuentran siete grupos de procesadores que se explican en el siguiente punto.

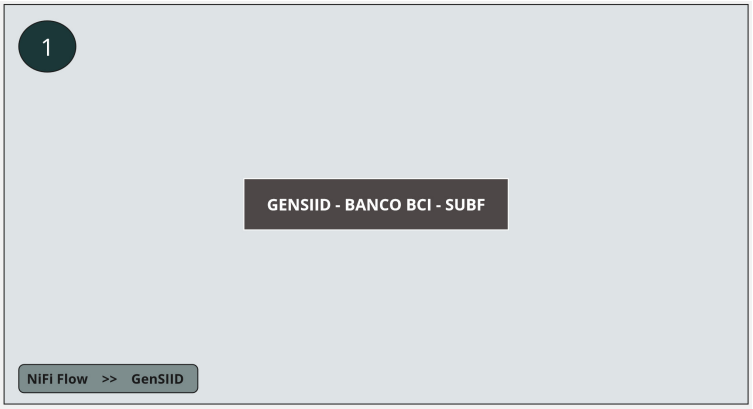


Figura 13: Diagrama 1 del nuevo flujo.
Fuente: Elaboración propia.

2. El flujo se organiza en torno a siete grandes grupos de procesadores:

   - subflujos: se realizan los procesos que son comunes a todos los reportes. Dado que hay varias partes del proceso ETL que son idénticas para cada reporte, se han consolidado en este único grupo de procesadores. Esto implica que los flujos de datos de todos los reportes deben pasar por esta etapa de procesamiento.
   - MIR, MFX, MFI, DIR, DFX, DFI: cada uno de estos grupos de procesadores realiza las transformaciones necesarias a los datos crudos, buscando obtener los índices que se deben incluir en cada reporte.

   A continuación se muestra el diagrama correspondiente.

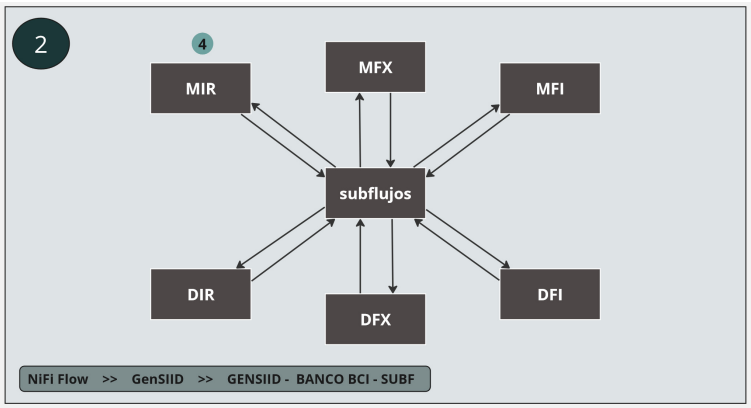


Figura 14: Diagrama 2 del nuevo flujo.
Fuente: Elaboración propia.

Se observa que los grupos de procesadores de cada reporte están conectados con el grupo central "subflujos". En el siguiente punto se describe con más detalle los procesos que ocurren al interior de este último.

3. En "subflujos" se realizan las siguientes tareas en este orden (en las Figuras 14 y 15 también se señala el orden de estas tareas).

- (1) Inicio del flujo de datos
  Como se puede apreciar en el Diagrama 3, cada reporte cuenta con su propio punto de inicio, lo que permite que la generación de cada informe se lleve a cabo de manera individual. La etiqueta "Inicio automático diario" corresponde al inicio conjunto de los tres reportes diarios. Se decide unirlos porque DIR, DFX y DFI se ejecutan todos los días al mismo tiempo, por lo tanto, resulta práctico tener un único punto de inicio que abarque los tres reportes simultáneamente.

- (2) Limpieza de colas
  Se limpian las colas que hayan podido quedar de ejecuciones anteriores, eliminando la acumulación de datos innecesarios en el sistema. Esto garantiza que no se cometan errores al procesar datos irrelevantes.

- (3) Extracción de archivos y llenado de tablas dsa
  Se extraen los archivos de los clientes desde las fuentes. Estos datos se insertan en las tablas del modelo creado anteriormente. Específicamente, se insertan en tablas que se encuentran en el esquema gensiid_dsa. Esto permite almacenar la data original en las bases de datos de NeoSoft.

- (4) MIR, MFX, MFI, DIR, DFX y DFI
  Se realizan las transformaciones de los datos de acuerdo a lo requerido por cada reporte.

- (5) Llenado tablas etl
  Se insertan los datos transformados en tablas del esquema gensiid_etl del nuevo modelo, almacenando la información que se debe incluir en los reportes.

- (6) Generación CSV
  Para formar cada reporte se seleccionan las columnas de las tablas etl anteriormente pobladas que deben ser incluidas y se guardan en formato CSV. Luego, este último es cargado en el directorio de NeoSoft, dando finalización al proceso ETL.

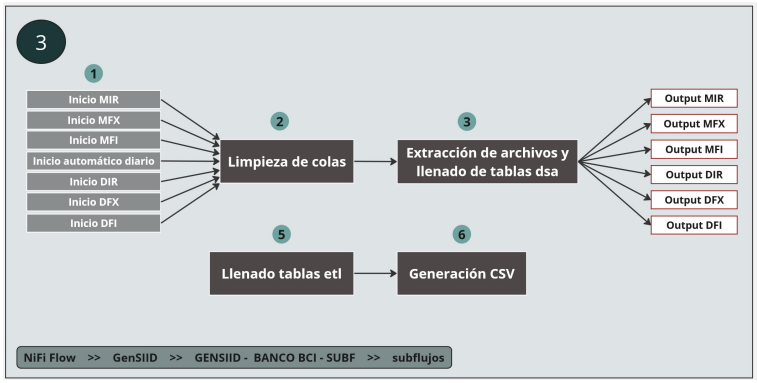


Figura 15: Diagrama 3 del nuevo flujo.
Fuente: Elaboración propia.

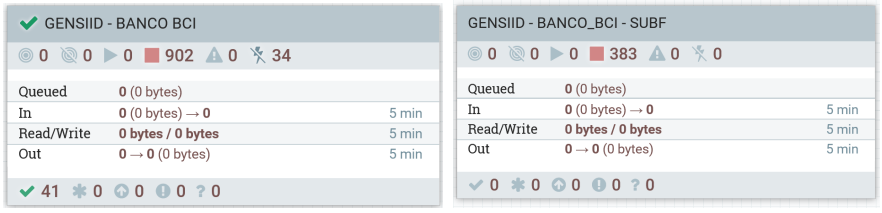
El resultado de este proceso es un reporte en formato CSV. A modo de ejemplo, la Figura 16 muestra un reporte MFX con datos ficticios y solo para fines referenciales. La primera columna indica la sección a la que pertenece el registro. En este caso, el archivo cuenta con las secciones 1, 2, 3, 6, 7 y 8. Las siguientes cuatro columnas indican el RUT del cliente, el identificador del contrato, la fecha de envío y el identificador de operación estructurada. Se observa que estos campos son comunes para todas las secciones. El resto de las columnas muestra información distinta para cada sección.

600000004MFX20201130

1	610000002	Id_00012679	2020-10-01T	0	Y	MVI		500000007	213800FQ9Y	ABCD	CHL
1	620000000	Id_00012647	2020-10-02T	0	Y	MVI		500000007	213800FQ9Y	ABCD	CHL
1	630000009	Id_00012754	2020-10-05T	0	Y	MVI		500000007	213800FQ9Y	ABCD	CHL
2	610000002	Id_00012679	2020-10-01T	0	FWD				CO	USD	02-10-2020
2	620000000	Id_00012647	2020-10-02T	0	FWD				CO	USD	03-10-2020
2	630000009	Id_00012754	2020-10-05T	0	FWD				CO	USD	06-10-2020
3	610000002	Id_00012679	2020-10-01T	0		USD	2000000	CLP	1516000000		
3	620000000	Id_00012647	2020-10-02T	0		USD	1000000	CLP	747320000		
3	630000009	Id_00012754	2020-10-05T	0		USD	4000000	CLP	2930680000		
6	630000009	Id_00012754	2020-10-05T	0	E	CLP	Portafolio_Id	500000	##########		
7	630000009	Id_00012754	2020-10-05T	0	E	Portafolio_Id	BBCH	55			
7	630000009	Id_00012754	2020-10-05T	0	E	Portafolio_Id	BTGR	45			
8	610000002	Id_00012679	2020-10-01T	0	M	USD	##########				
8	620000000	Id_00012647	2020-10-02T	0	M	USD	67.894.382				
8	630000009	Id_00012754	2020-10-05T	0	M	USD	##########				

Figura 16: Reporte MFX.
Fuente: Elaboración propia.

La Figura 17 muestra el grupo de procesadores para Banco BCI antes y después de integrar subflujos. Es posible notar que se redujo considerablemente el número de procesadores a utilizar, pasando de 936 a 383, y se simplifica la estructura del proceso ETL que se lleva a cabo, puesto que se reutilizan los procesadores que realizan tareas comunes a todos los reportes.



(a) Sin subflujos (b) Con subflujos

Figura 17: Grupo de procesadores de Banco BCI.
Fuente: Elaboración propia.

Paralelamente, se crean variables para almacenar valores que pueden ser utilizados en diferentes procesadores. Estas variables pueden ser definidas en varios lugares dentro del flujo, y luego referenciadas en diversas partes de este mismo. Esto evita que las configuraciones de algunos procesadores se definan de forma manual, es decir, editando directamente cada una de estas componentes. De esta manera, si se desea realizar algún cambio, solo se debe modificar una vez la variable creada y no hay que modificar cada uno de los procesadores donde se usa esta información, ahorrando tiempo y evitando errores humanos. Además, promueve un flujo más genérico y adaptable. Algunos ejemplos de variables son el rut del cliente, las direcciones de los directorios a los que se debe acceder para extraer los archivos fuentes y para cargar el reporte final, el nombre de usuario y contraseña de la base de datos, entre otros.

Para integrar a un nuevo cliente, primero se debe duplicar un flujo existente de otro cliente. En otras palabras, el flujo para el nuevo cliente se construye sobre la base de un flujo que ya existe. Posteriormente, una vez duplicado el flujo, se deben llevar a cabo las modificaciones necesarias para incorporar la información específica de la nueva institución. Así, la integración de variables facilita la implementación de nuevos clientes, dado que ya no es necesario modificar la configuración de todos los procesadores, sino que solo se deben actualizar las variables con los nuevos valores.

Para evidenciar el cambio generado, en la Figura 18 se expone la situación original, es decir, el proceso ETL sin subflujos para cada uno de los reportes, y en la Figura 19 se muestra el proceso ETL con subflujos implementados.

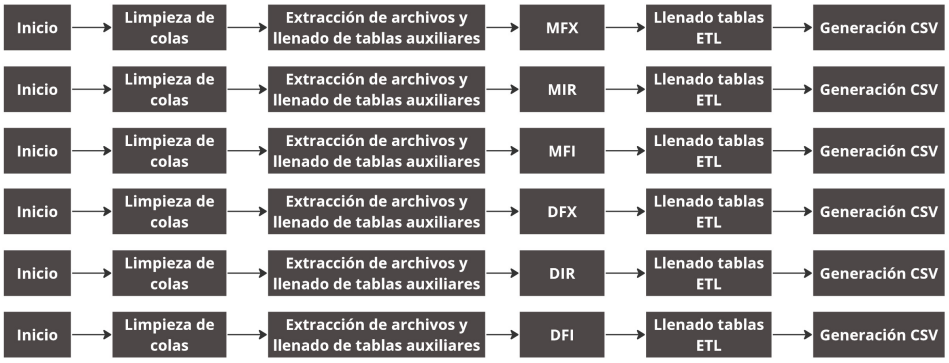


Figura 18: Proceso ETL antes de la implementación de subflujos.
Fuente: Elaboración propia.

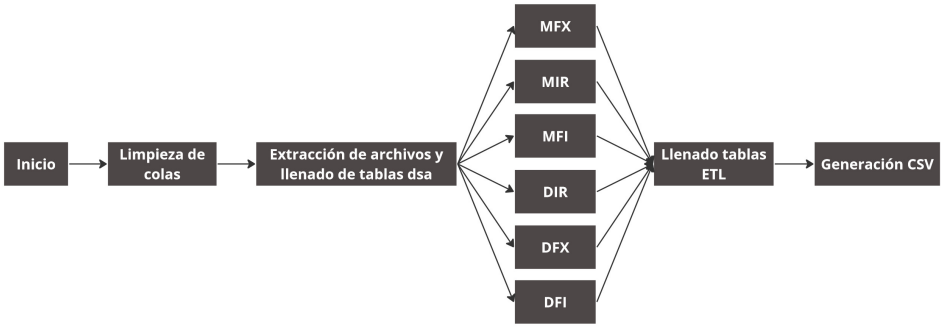


Figura 19: Proceso ETL después de la implementación de subflujos.
Fuente: Elaboración propia.

Se observa que los subflujos son usados por todos los reportes y que solo la parte de la transformación es diferenciada, logrando simplificar el proceso de forma considerable y requiriendo pocas acciones para implementar cambios importantes. Una consecuencia de esto es la reducción de tiempo destinado a la mantención del flujo, pasando de 3 horas a solo 1 hora por cada cambio realizado a cada cliente. También se debe considerar que cada cambio pasa por un proceso de revisión. El tiempo destinado a esta tarea se logra reducir de 2 a 1.3 horas (valores estimados en base a entrevistas realizadas al equipo). Este ahorro de tiempo se puede valorar tomando como referencia el sueldo promedio de un ingeniero de datos. La tabla 5 muestra estos valores.

Tabla 5: Tiempos y ahorro en CLP para un cambio.

	Tiempo flujo original (h)	**Tiempo subflujos (h)**	**Ahorro (CLP)**
Mantención	3	1	$ 14444
Revisión	2	1.3	$ 5055
Total	5	2.3	$ 19500

Se observa que el total de horas reducido tiene un impacto de casi 20 mil pesos para cada cambio realizado al flujo. Es decir, si se realizan 2 cambios a la semana durante un mes, se ahorran $156.000. Durante un año, este monto asciende a $1.872.000. Además, hay que considerar que cada uno de los cinco clientes cuenta con un proceso ETL distinto (se mantienen separados unos de otros porque involucran data diferente) y que cada uno de ellos pasa por procesos de mantención, por lo tanto, suponiendo que los flujos de los clientes son similares, el monto final de ahorro asciende a $ 9.360.000 anualmente.

Dado lo anterior, es posible afirmar que el proyecto cuenta con un impacto económico positivo que beneficia a NeoSoft. Sin embargo, este monto no significa un gran cambio para la empresa. En este sentido, el valor principal del proyecto reside en la capacidad de NeoSoft para contar con procesos optimizados que puedan reducir el tiempo y esfuerzo dedicados a tareas rutinarias. Además, esto permite hacer un mejor uso de los recursos humanos, redirigiendo los esfuerzos del equipo hacia actividades que generen mayores beneficios para la empresa. De esta manera, el verdadero impacto del proyecto radica en el aumento de la eficiencia y la maximización del potencial del equipo.

9.3. Gestión del conocimiento

Una vez implementadas las soluciones, es necesario compartir los cambios realizados y hallazgos encontrados con el resto del equipo de GenSIID. Esto se desarrolla de acuerdo al Modelo Wiig, que plantea que el conocimiento debe estar organizado para ser útil y valioso. Considerando lo anterior, se define una serie de actividades para velar que las soluciones implementadas se usen de forma correcta en el futuro.

En primer lugar, se crea un manual que muestra la situación original y luego describe el contenido del nuevo modelo de datos y del nuevo flujo del proceso ETL. Esto permite visualizar los cambios realizados y ayuda a que el equipo de NeoSoft se adapte al nuevo sistema. Además, menciona qué función cumple cada una de estas partes y, en el caso del flujo, cómo ejecutarlas. Para finalizar, se describen los beneficios de ambas soluciones. De esta forma, el manual aporta a la gestión del conocimiento asegurando que la información se preserve en

el tiempo, se comparta efectivamente y se utilice para la mejora continua. También facilita la comprensión y el uso del sistema por parte de nuevos empleados y de aquellos que no participaron en la creación de las soluciones.

En segundo lugar, se imparte un taller de capacitación, donde se muestra el sistema con ambas soluciones implementadas y se enseña cómo usarlo. El aporte de esta actividad a la gestión del conocimiento es que facilita la transición al nuevo sistema y crea un espacio para que los miembros del equipo compartan ideas y experiencias, promoviendo la colaboración. En cuanto al modelo de datos, se detallan los seis esquemas que lo conforman, indicando los tipos de datos que incluye cada uno y su utilidad dentro del modelo. Esta información no solo permite que el equipo se familiarice con el nuevo modelo, sino que también les ayuda a comprender la lógica detrás de esta nueva organización de los datos. Luego, se listan las buenas prácticas utilizadas durante la implementación de esta solución, lo que permite que el equipo pueda replicar estos actos en otros proyectos. Finalmente, se explican los beneficios de esta solución, lo que ayuda al equipo a comprender por qué se decidió implementarla y por qué es superior al modelo anterior. Respecto al proceso ETL, en primera instancia se muestra el flujo original, evidenciando cuáles son los problemas de este. Posteriormente, se presenta el nuevo flujo, detallando cada una de sus partes y explicando qué rol cumplen dentro del proceso ETL. Además, desde la misma interfaz de Apache NiFi, se enseña cómo ejecutarlo para generar cada uno de los reportes. Finalmente, se indican los beneficios de esta solución.

En tercer lugar, es esencial almacenar toda la documentación relacionada con el proyecto en el repositorio compartido de GenSIID. Esto garantiza que el acceso a estos archivos sea libre y equitativo para todos los miembros del equipo, eliminando cualquier barrera que pueda dificultar la utilización de estos recursos. De esta forma, al centralizar la documentación en un repositorio común, se facilita la colaboración y se garantiza que todos los involucrados puedan acceder rápidamente a la información necesaria para tomar decisiones informadas y realizar sus tareas de manera eficiente.

Es importante mencionar que la información mostrada en el manual y en la capacitación debe mantenerse actualizada en el tiempo, dado que su contenido eventualmente quedará obsoleto ante los futuros cambios que surjan. Es por esta razón, que en cuarto lugar se establece que la documentación debe ser actualizada cada seis meses. Esta práctica asegura que todos los cambios importantes, que puedan afectar tanto el funcionamiento del modelo de datos como del proceso ETL, sean capturados y reflejados adecuadamente. Así, se garantiza que el personal de NeoSoft siempre cuente con información precisa y relevante, facilitando una mejor adaptación y optimización de los procesos.

En último lugar, se designa a un encargado que se encargue de difundir la documentación y solicite la actualización de esta en los plazos que corresponda. Esta persona debe contar con los conocimientos y facultades necesarias para desempeñar su rol de forma satisfactoria, por lo que se sugiere que cuente con el cargo de desarrollador senior.

Estas actividades son esenciales para gestionar los conocimientos alcanzados durante el desarrollo de este proyecto. Además, fortalecen la capacidad de NeoSoft para enfrentar desafíos similares a este en el futuro, puesto que evitan la repetición de errores y fomentan la réplica de prácticas exitosas.

10. Discusiones

Los objetivos planteados al inicio del documento apuntan hacia la mejora del modelo de datos y de la eficiencia del proceso ETL mediante la implementación de un nuevo modelo y de subflujos en el ETL. Aunque se cumplieron los objetivos específicos, es crucial cuestionar si estos fueron suficientemente ambiciosos y alineados con las necesidades a largo plazo de NeoSoft. En ese sentido, hubiese sido positivo considerar como un objetivo la implementación del proyecto en un ambiente de producción, puesto que esto hubiese permitido probar las soluciones en tiempo real con los clientes actuales. Sin embargo, esto no fue considerado debido al tiempo acotado del proyecto.

El alcance tuvo un enfoque limitado al no considerar la integración de tecnologías emergentes en la implementación de subflujos. Esto se debe a que se decidió priorizar el programa que ya se utilizaba en GenSIID, ya que permitía ahorrar tiempo aprovechando una solución preexistente. Migrar a un programa diferente implicaba comenzar desde cero, lo que habría consumido más tiempo. A pesar de lo anterior, se pudo haber evaluado la elección de una tecnología más nueva y eficiente que Apache NiFi.

Otra limitante fue no considerar la calidad de los reportes dentro del proyecto. En su lugar, se priorizó conservar el contenido de los informes que actualmente se ejecutan para poder enfocarse completamente en la arquitectura detrás de la generación de estos documentos. En este caso, hubiese sido beneficioso optimizar las consultas SQL que se usan para transformar los datos crudos que envían los clientes, con el fin de obtener una transformación metódica y organizada, evitando cualquier tipo de errores. Incluso, este cambio podría reducir el tiempo de ejecución de cada consulta SQL, contribuyendo a un proceso ETL más rápido y eficiente. Haber considerado este aspecto dentro del proyecto hubiese significado una importante mejora.

La metodología usada permitió en primera instancia notar las deficiencias y problemas específicos que existían a través de una exploración del modelo de datos y del proceso ETL. Luego, en base a estos hallazgos se redactó una propuesta que fue discutida con el resto del equipo. Una vez aprobada, fue implementada. Si bien esta metodología permitió alcanzar los objetivos establecidos, es importante considerar si otras metodologías podrían haber ofrecido beneficios adicionales. En particular, habría sido beneficioso realizar entrevistas al equipo de GenSIID para obtener sus opiniones sobre la situación original. Esto habría proporcionado una variedad de perspectivas sobre la problemática en cuestión. Con estas entrevistas, se habría logrado una visión holística y completa del sistema original, lo que podría haber conducido a un mayor entendimiento de las oportunidades y desafíos. En consecuencia, esto habría permitido desarrollar una solución más adecuada y alineada con las necesidades y expectativas del equipo.

Los resultados obtenidos sugieren mejoras significativas en la eficiencia y organización del modelo de datos, así como en el proceso ETL. No obstante, es crucial evaluar si estos resultados podrían haberse mejorado aún más. Para maximizar los beneficios, habría sido beneficioso implementar un sistema de evaluación continua del desempeño de las soluciones implementadas. Este sistema permitiría monitorear en tiempo real la efectividad del modelo de datos y del proceso ETL, identificando rápidamente cualquier área que necesite ajustes o

mejoras. Con un sistema de evaluación continua, se podría reaccionar de manera proactiva a posibles problemas, realizar ajustes necesarios de manera oportuna y asegurar que las mejoras implementadas se mantengan efectivas a largo plazo. Además, este enfoque facilitaría la detección de nuevas oportunidades para optimizar aún más los procesos y adaptarse a las necesidades cambiantes del entorno donde se encuentra NeoSoft.

Dadas las limitantes encontradas, para maximizar los beneficios derivados de este proyecto es fundamental que se implemente en un entorno de producción para todos los clientes de GenSIID en un futuro cercano. Esta implementación permitirá a todos los clientes aprovechar las mejoras en eficiencia y organización que ofrece el nuevo modelo de datos y los subflujos. Además, sería positivo extender el uso de estas mejoras a otros proyectos de NeoSoft que también utilicen procesos ETL. Esto no solo facilitará la mejora continua en toda la empresa, sino que también estandarizará las prácticas óptimas y aumentará la cohesión y eficiencia entre diferentes proyectos y equipos dentro de NeoSoft. Al hacerlo, la empresa podrá consolidar su posición competitiva en el mercado y ofrecer un valor añadido significativo a sus clientes.

11. Conclusiones

Una vez finalizado el proyecto, es posible dar cuenta de los beneficios que este trajo. Principalmente, se ha logrado un ahorro de tiempo significativo en tareas innecesarias. En cuanto al modelo de datos, ya no se requiere crear tablas redundantes y, en relación al proceso ETL, se evita la duplicación de procesadores existentes, por lo que basta con añadir las transformaciones específicas de cada cliente. Así, se logró obtener un sistema de generación de reportes mucho más simple y organizado, lo que facilita tanto su propio mantenimiento como la integración de nuevos clientes. De este modo, es posible afirmar que el proyecto permitió aumentar la eficiencia operativa de NeoSoft, cumpliendo con el objetivo general establecido.

Durante la ejecución de este proyecto, el enfoque del rol fue abordar todas las etapas requeridas para su completa realización, desempeñando un papel vital en la materialización de la solución previamente planteada. Sin una intervención activa y diligente, el sistema seguiría operando con redundancias y sería poco óptimo en términos de eficiencia y escalabilidad.

En primer lugar, fue necesario comprender los datos que alimentan el proyecto para poder agrupar correctamente la información en el nuevo modelo de datos y así sacar provecho de esta. Del mismo modo, también fue imprescindible entender la función que cada procesador cumple dentro del flujo y qué cambio realiza a los datos. Estos aspectos contribuyeron a la identificación de redundancia, ineficiencia y oportunidades de mejora. Por todo lo anterior, se considera que el primer objetivo específico sí se cumple.

Después de identificar claramente todos los problemas presentes en el modelo de datos y en el flujo del proceso ETL, se redactaron propuestas destinadas a abordar dichos asuntos. Durante este proceso se buscaron diversas alternativas para abordar las oportunidades encontradas, las que fueron presentadas al resto del equipo, manteniéndose un continuo intercambio de ideas, hasta que se llegó al diseño final de las soluciones. Se puede concluir que un buen modelo de datos es aquel que organiza la información en función del uso que se le otorga, establece relaciones entre los datos, es ajustable en el tiempo para permitir la implementación de mejoras de manera simple y es adaptable a grandes volúmenes de información, facilitando su escalabilidad. Además, un proceso ETL eficiente es aquel que evita repetir procedimientos comunes. Teniendo en cuenta estos aspectos, se puede afirmar que el segundo objetivo específico se ha logrado satisfactoriamente.

Posteriormente, se implementaron las soluciones en ambientes controlados. Esto permitió realizar múltiples pruebas que aseguraron el correcto funcionamiento de ambos productos. Se destaca la importancia de esta etapa, ya que fue durante el proceso de prueba y error donde se obtuvieron los aprendizajes más significativos. En este período, se desarrollaron diversas estrategias para abordar los diferentes problemas que surgieron. Por lo anterior, es que el tercer objetivo específico también se considera logrado.

Finalmente, las actividades desarrolladas para difundir los hallazgos y aprendizajes alcanzados dentro de NeoSoft permitió lograr una exitosa gestión del conocimiento en base al Modelo Wiig, por lo que el cuarto y último objetivo también se logra con éxito.

Ahora bien, la aplicación de este nuevo modelo de datos y la respectiva implementación de subflujos en el proceso ETL tiene impactos concretos en el operar de NeoSoft. Uno de los impactos directos corresponde a la reducción de recursos tecnológicos necesarios. Específicamente, disminuye el uso de memoria en la nube y la capacidad de las máquinas virtuales[2] usadas. Concretamente, en el caso del modelo de datos, se logra reducir el uso de la memoria en un 28 %. Esto se traduce en la reducción de gastos en las suscripciones de Microsoft Azure[3].

El impacto más significativo, sin lugar a dudas, radica en la notable reducción del tiempo necesario para incorporar a un nuevo cliente. Anteriormente, este proceso consumía entre cinco y seis meses, aproximadamente, pero con las mejoras aplicadas, ahora puede completarse en un periodo de tres meses. Esta eficiencia se alcanza gracias al ahorro de tiempo en tareas repetitivas que se logra con el sistema genérico que se ha implementado. Como consecuencia directa, los tiempos de entrega también se acortan, proporcionando un servicio mucho más ágil y oportuno. Este logro repercute positivamente en la percepción de NeoSoft en el mercado, fortaleciendo su posición como proveedor de soluciones eficientes y rápidas.

Otro impacto positivo del proyecto radica en la disminución del tiempo dedicado a las tareas relacionadas con el modelo de datos y con el flujo del proceso ETL. Específicamente, es posible mantener la generación de los reportes en menos tiempo. Un hecho concreto es la reducción de la cantidad de procesadores en un 60 %, pasando de 936 a 383. Una consecuencia directa es que facilita las futuras modificaciones al proceso ETL, ya que no es necesario configurar tantos procesadores para poder implementar un solo cambio. Esto también reduce la probabilidad de cometer un error humano al manipular el flujo e implica que disminuya el nivel de especialización de GenSIID, ya que al estar todo más generalizado, se facilita que alguien nuevo se inserte en el proyecto. De esta manera, el tiempo destinado a la mantención y revisión se logra reducir en un 46 %, permitiendo, bajo ciertos supuestos, que se ahorre un monto de $ 9.360.000 de forma anual. Paralelamente, el número total de tablas del modelo de datos disminuye en un 42 %. Esta optimización no solo agiliza las operaciones diarias, sino que también libera tiempo valioso para el equipo de NeoSoft, permitiéndoles enfocarse tanto en nuevas iniciativas y proyectos, así como en mejorar la calidad de los servicios existentes. Lo anterior permite solucionar uno de los dolores de NeoSoft, correspondiente a los costos adicionales en tiempo y recursos humanos que involucra el sistema original.

Otro dolor actual es la dificultad para escalar GenSIID. Respecto a esto, el proyecto brinda un modelo de datos simple y organizado, y un proceso ETL sin redundancias, permitiendo implementar a un cliente nuevo en menos tiempo, dado que ya no es necesario crear tantas tablas ni configurar tantos procesadores para lograrlo. Concretamente, de las 166 tablas comunes que existían, únicamente se conservaron 42, y para un nuevo cliente solo es necesario crear las tablas requeridas para guardar su propia data. Así, se ha creado un sistema escalable que permite integrar a nuevos clientes en poco tiempo e, incluso, contando con la organización adecuada, se podrían integrar de forma paralela.

[2] Representación virtual o emulación de un ordenar, permitiendo ejecutar sistemas operativos y aplicaciones como si fuera una computadora física.

[3] Plataforma en la nube que ofrece diversos servicios y herramientas para almacenar datos, ejecutar aplicaciones y gestionar recursos informáticos a través de internet.

Lo anterior se encuentra estrechamente relacionado con la entrega de un servicio poco inmediato. Este dolor también se puede abordar mediante el sistema optimizado que se ha implementado para integrar a los clientes de manera más rápida. A su vez, esto mejora la experiencia del cliente, lo cual no solo incrementa la retención de clientes existentes, sino que también puede atraer a nuevos, aumentando así la presencia de NeoSoft en el mercado.

Incluso, sin ir más allá, si la empresa decide optimizar todos sus procesos en todos los servicios que ofrece, fácilmente esta mejora podría convertirse en una propuesta de valor sólida. Al desarrollarla y perfeccionarla en el tiempo, se vuelve difícil de imitar por los competidores, consolidándose como una competencia central. Esta última permite a NeoSoft ofrecer un valor superior a sus clientes, lo que a su vez puede traducirse en un mejor desempeño en el mercado. Esta mejora tiene el potencial de convertirse en una ventaja competitiva significativa, diferenciando a NeoSoft de sus competidores y posicionándola como una empresa líder en su área.

Finalmente, el trabajo futuro del proyecto incluye tres aspectos clave. Primero, es esencial extender la implementación de la solución a todos los clientes de GenSIID. Al hacerlo, no solo se logrará un servicio uniforme y estandarizado para todos los usuarios, sino que también se garantiza que cada cliente pueda aprovechar al máximo las mejoras y optimizaciones implementadas.

Segundo, es de suma importancia llevar el nuevo modelo de datos y los subflujos del proceso ETL desde el entorno de desarrollo a un ambiente de producción. Esta transición permitirá que los avances y beneficios obtenidos durante la realización del proyecto se reflejen en la operación diaria y en tiempo real con los clientes actuales.

El tercer y último punto a abordar en el futuro, es la calidad de los reportes generados. Esto se puede lograr a través de la optimización de las consultas SQL que se ejecutan en la etapa de transformación de los datos. Una optimización constante del código no solo mejorará el rendimiento de las consultas, sino que también reducirá la posibilidad de errores, facilitando un manejo más eficiente y preciso de los datos. Mantener el código ordenado y eficiente es esencial para que el equipo de GenSIID pueda manejar las consultas de manera más efectiva, minimizando la posibilidad de problemas futuros.

Adicionalmente, es relevante considerar que todo el proyecto debe estar sometido a un proceso de mejora continua. Esta práctica permitirá que las tecnologías y metodologías utilizadas no queden obsoletas con el tiempo, asegurando que la generación de reportes y el rendimiento del sistema sean siempre los mejores posibles. En definitiva, siempre existirá margen para seguir mejorando y adaptándose a las nuevas necesidades, avances tecnológicos y desafíos que presente el mercado.

12. Bibliografía

1. Amazon Web Services. (2023). *¿Qué es ETL?* Recuperado el 1 de mayo de 2024 de https://aws.amazon.com/es/what-is/etl/

2. Amazon Web Services. (2023). *¿Qué es una base de datos?* Recuperado el 1 de mayo de 2024 de https://aws.amazon.com/es/what-is/database/#:~:text=Una%20base%20de%20datos%20es,almacenar%2C%20recuperar%20y%20editar%20datos

3. Banco Central de Chile. (s.f.). *Manual de Participantes.* Recuperado el 22 de marzo de 2024 de https://www.siid.cl/documents/2446522/2454286/Manual+del+Usuario+-+SIID+v1.0++Participante.pdf/a860a362-29e4-b5f1-3487-98c8e64fbbdf?t=1603805386747

4. Conde, D. (2022). *Diseño de un framework de análisis de datos abiertos mediante un proceso ETL.* Recuperado el 1 de mayo de 2024 de https://oa.upm.es/72875/1/TFG_DANIEL_CONDE_RAMIREZ.pdf

5. Dalkir, K. (2011). *Knowledge Management in Theory and Practice.* Recuperado el 15 de mayo de 2024 de https://nibmehub.com/opac-service/pdf/read/Knowledge%20Management%20in%20Theory%20and%20Practice%20by%20Kimiz%20Dalkir-%20Jay%20Liebowitz.pdf

6. Fernandez, O. (s.f.). *Apache NiFi: Introducción.* Recuperado el 1 de mayo de 2024 de https://aprenderbigdata.com/introduccion-apache-nifi/

7. Garrido, A., López, Y., & Gutiérrez, G. (2020). Rendimiento de MariaDB y PostgreSQL. *Revista Científica y Tecnológica UPSE (RCTU)*, 7(2), 9-16. https://doi.org/10.26423/rctu.v7i2.538

8. Hammer, Michael, y James Champy. (2001). *Reengineering the Corporation: A Manifesto for Business Revolution.* HarperBusiness.

9. IBM. (s.f.). *Gestión de conocimientos.* Recuperado el 10 de mayo de 2024 de https://www.ibm.com/es-es/topics/knowledge-management

10. Mahler, Charles. (2023). *7 Data Modeling Best Practices.* The New Stack. Recuperado el 19 de julio de 2024 de https://thenewstack.io/7-data-modeling-best-practices/

11. MariaDB. (s.f.). *MariaDB Server.* Recuperado el 22 de marzo de 2024 de https://mariadb.org/es/#entry-header

12. Microsoft. (s.f.). *Conceptos básicos sobre bases de datos.* Recuperado el 1 de mayo de 2024 de https://support.microsoft.com/es-es/topic/conceptos-b%C3%A1sicos-sobre-bases-de-datos-a849ac16-07c7-4a31-9948-3c8c94a7c204

13. Neosoft. (2021). Recuperado el 22 de marzo de 2024 de https://neosoft.cl

14. PowerData. (s.f.). *Procesos ETL: Definición, características, beneficios y retos.* Recuperado el 10 de mayo de 2024 de https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/312584/procesos-etl-definici-n-caracter-sticas-beneficios-y-retos

15. PowerData. (s.f.). *¿Qué son los procesos ETL?* Recuperado el 10 de mayo de 2024 de https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/qu-son-los-procesos-etl

16. Seenivasan, D. (2023). ETL (extract, transform, load) best practices. *International Journal of Computer Trends and Technology, 71*(1), 40-44. https://doi.org/10.14445/22312803/IJCTT-V71I1P106

13. Anexos

Anexo A

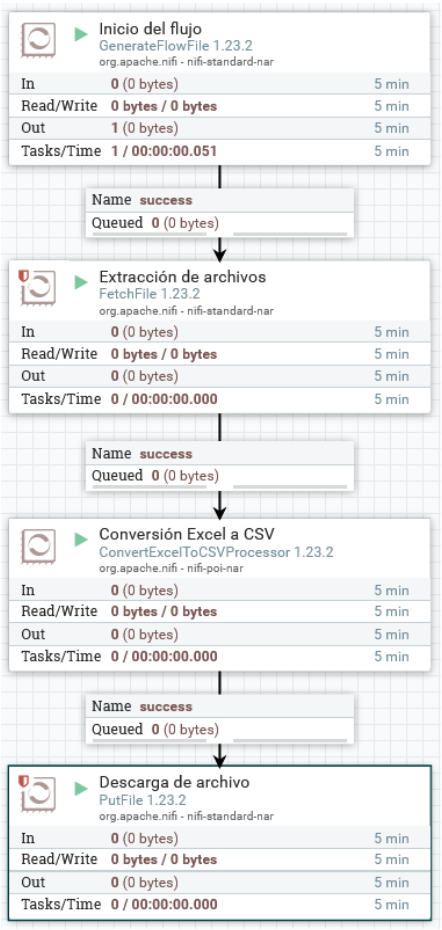


Figura 20: Flujo de ejemplo.
Fuente: Elaboración propia.

Anexo B

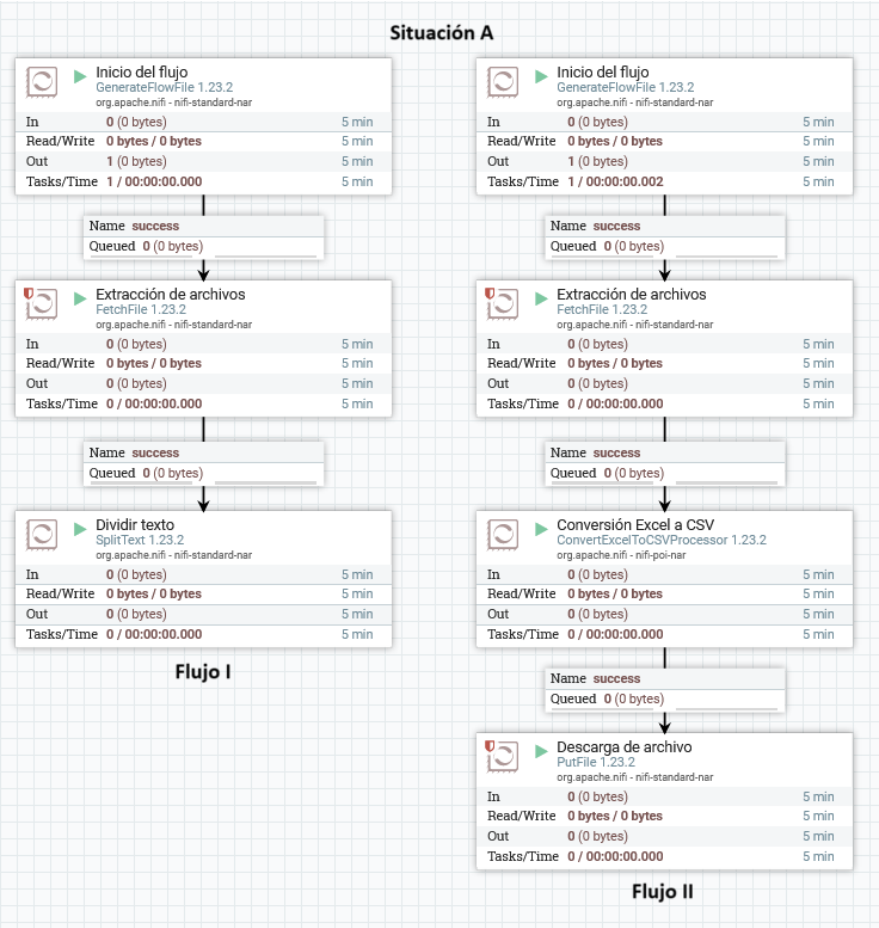


Figura 21: Situación A.
Fuente: Elaboración propia.

Anexo C

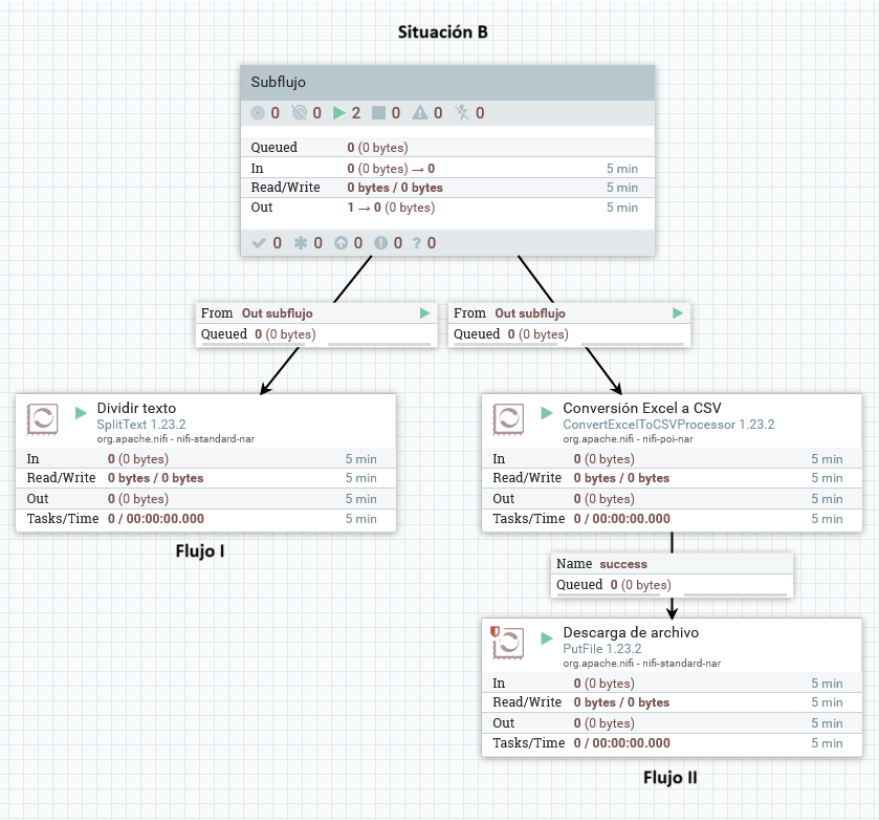


Figura 22: Situación B.
Fuente: Elaboración propia.

Anexo D

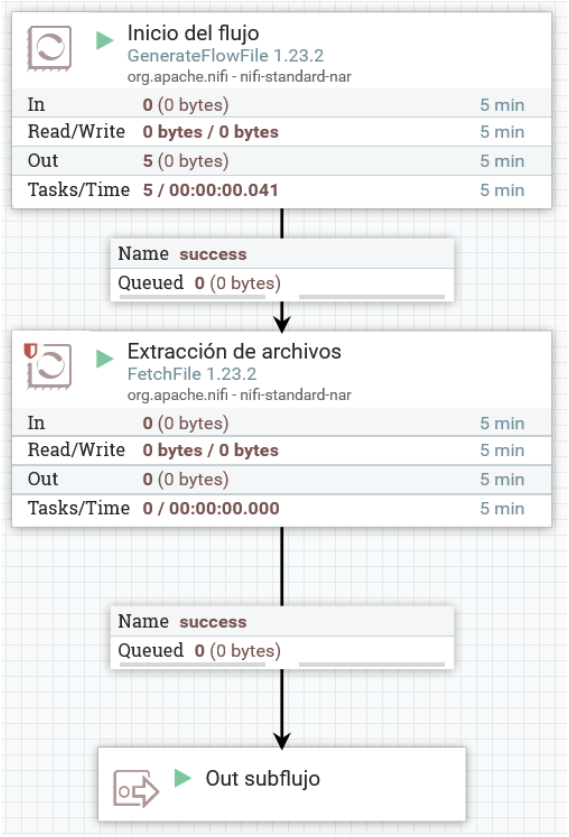


Figura 23: Interior del subflujo en situación B.
Fuente: Elaboración propia.

Anexo E

El modelo de datos de GenSIID cuenta con dos esquemas base que contienen información sobre el funcionamiento (*bc*) y el formato de los reportes (*db_tabgral*). Además, cada cliente cuenta con su propio esquema. Es importante mencionar que ninguno de estos tiene sus tablas relacionadas, es decir, todas las tablas son independientes entre sí.

A continuación, se describen las tablas contenidas en cada uno de los esquemas.

bc

- aux_tb_err: Errores de validación de los reportes. Están dentro de la aplicación web y se muestran a los clientes.
- Calendario_fechas_habiles: Tabla con fechas desde el 1 de enero del 2020 hasta el 14 de mayo del 2021, donde se indica si cada día es hábil o no.
- destinatarios_alertatemprana: Lista de destinatarios clientes definidos para notificar que falta un archivo.
- destinatarios_email: Lista de destinatarios de email.
- email_confirmations: Incluye códigos de confirmación de usuarios.
- esperando_validacion: No se usa.
- failed_logins: Registro de logins fallidos de la aplicación web.
- fuentes_cliente: Indica qué tipo de documento es cada fuente (archivos que entregan los clientes para generar los reportes). Se usan en Apache NiFi.
- grupo_procesadores: Indica el nombre de los procesadores del diagrama de Apache NiFi y a qué subproceso pertenecen.
- id_ejecuciones: Identificación de las ejecuciones de los reportes. Incluye fechas de reporte y de ejecución.
- Libros_SIID: No se usa.
- log_auditoria: No se usa.
- log_etl_general: Guarda todo lo que ocurre en Apache NiFi.
- log_general_proceso: Registra el envío exitoso de un correo. Esta tabla solo es usada por Banco Ripley.
- log_gensiid: No se usa.
- nombre_columnas: Diccionario que indica el nombre de las columnas del reporte final y una pequeña descripción de cada una de ellas.
- password_changes: Registro de cambios en las contraseñas de la aplicación web.
- permissions: Lista de permisos de cada perfil.

- phinxlog: No se usa.
- profiles: Lista con los tipos de perfiles.
- Prueba_almacenar_datos: Tabla de prueba. No se usa.
- prueba_group: Tabla de prueba. No se usa.
- prueba_input: Tabla de prueba. No se usa.
- remember_tokens: Se usa para recordar los tokens de inicio de sesión de la aplicación web.
- reset_passwords: Registro de contraseñas.
- success_logins: Registro de los logins exitosos.
- test_dfx_1: No se usa.
- users: Lista de usuarios de la aplicación web.
- uuid_etl: No se usa.
- uuid_generador: No se usa.

db_tabgral

Todas las tablas de este schema se relacionan con el formato que deben cumplir los reportes financieros.

- TU_SIID_CLASE_OPCION
- TU_SIID_CLAUSULA_TERMINO
- TU_SIID_COMPRESION_CARTERA
- TU_SIID_CONVENIO_MARCO
- TU_SIID_EVENTO_REPORTE
- TU_SIID_GARANTIAS
- TU_SIID_INSTRUMENTOS
- TU_SIID_METODO_VALORIZACION
- TU_SIID_MODALIDAD_LIQUIDACION
- TU_SIID_MONEDAS
- TU_SIID_NATURALEZA_INFORMACION
- TU_SIID_OBJETIVO_OPERACION
- TU_SIID_PAISES

- TU_SIID_PLATAFORMA_NEGOCIACION
- TU_SIID_POSICION_OPERACION
- TU_SIID_RECOUPONING
- TU_SIID_TIPO_ACTIVO_GARANTIA
- TU_SIID_TIPO_COD_ACTIVO_SUBY
- TU_SIID_TIPO_PAGO_EFECTUADO
- TU_SIID_TIPO_TASA

banco_bci

Este schema cuenta con 47 tablas auxiliares. El uso de estas tablas se destina a los procesos que se llevan a cabo en Apache NiFi.

- colaterales_clientesIM: Contiene el rut de clientes del banco.
- dbo_tb_err: Listado de todos los errores que existen sobre los reportes.
- dbo_tb_err_val: Errores que ocurrieron durante la validación.
- dbo_TB_SIID_LOG: Guarda información de la validación de los reportes con respecto a lo que pide el Banco Central.
- dbo_TU_SIID_DFI_X: Tipo de reporte diario. Tiene 3 secciones ($\texttt{X} \in \{1, 2, 3\}$).
- dbo_TU_SIID_DFI_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_DFX_X: Tipo de reporte diario. Tiene 4 secciones ($\texttt{X} \in \{1, 2, 3, 4\}$).
- dbo_TU_SIID_DFX_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_DIR_X: Tipo de reporte diario. Tiene 4 secciones ($\texttt{X} \in \{1, 2, 3, 4\}$).
- dbo_TU_SIID_DIR_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_INSTITUCIONES_RELACIONADAS: Indica cuando un cliente reporta en nombre de otra empresa.
- dbo_TU_SIID_MFI_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MFX_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MIR_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_VAL_LEI: Diccionario sobre los valores LEI.
- diccionario_tasas: Diccionario sobre las tasas.

- ETL_Informe: Diccionario sobre los tipos de reporte.
- ETL_Instituciones: Información sobre el banco.
- ExternalCounterParty: Contiene datos de la contraparte del banco.
- ExternalCounterParty1: No se usa.
- ExtrCounterparty_Comder: Contiene datos de otro tipo de contraparte del banco.
- fechas_pagos_procesadas: Tabla de fechas.
- mapeo_garantias: Garantías que ofrece el banco. Se incluyen en los reportes.
- TU_SIID_X: Conjunto de tablas que entrega información sobre el formato de los reportes ($\texttt{X} \in$ *db_tabgral*). Estas tablas se encuentran duplicadas con las de *db_tabgral*.

bci_bam

Este schema cuenta con 4 tablas auxiliares. El uso de estas tablas se destina a los procesos que se llevan a cabo en Apache NiFi.

- dbo_tb_err: Listado de todos los errores que existen sobre los reportes.
- dbo_tb_err_val: Errores que ocurrieron durante la validación.
- dbo_TB_SIID_LOG: Guarda información de la validación de los reportes con respecto a lo que pide el Banco Central.
- dbo_TU_SIID_INSTITUCIONES_RELACIONADAS: Indica cuando un cliente reporta en nombre de otra empresa.
- dbo_TU_SIID_MFX_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_VAL_LEI: Diccionario sobre los valores LEI.
- ETL_Informe: Diccionario sobre los tipos de reporte.
- ETL_Instituciones: Información sobre el banco.
- TU_SIID_X: Conjunto de tablas que entrega información sobre el formato de los reportes ($\texttt{X} \in$ *db_tabgral*). Estas tablas se encuentran duplicadas con las de *db_tabgral*.

bci_corredora

Este schema cuenta con 26 tablas auxiliares. El uso de estas tablas se destina a los procesos que se llevan a cabo en Apache NiFi.

- dbo_tb_err: Listado de todos los errores que existen sobre los reportes.
- dbo_tb_err_val: Errores que ocurrieron durante la validación.
- dbo_TB_SIID_LOG: Guarda información de la validación de los reportes con respecto a lo que pide el Banco Central.

- dbo_TU_SIID_INSTITUCIONES_RELACIONADAS: Indica cuando un cliente reporta en nombre de otra empresa.
- dbo_TU_SIID_MFI_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MFX_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MIR_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_VAL_LEI: Diccionario sobre los valores LEI.
- ETL_Informe: Diccionario sobre los tipos de reporte.
- ETL_Instituciones: Información sobre el banco.
- TU_SIID_X: Conjunto de tablas que entrega información sobre el formato de los reportes ($\texttt{X} \in$ *db_tabgral*). Estas tablas se encuentran duplicadas con las de *db_tabgral*.

banco_ripley

Este schema cuenta con 19 tablas auxiliares. El uso de estas tablas se destina a los procesos que se llevan a cabo en Apache NiFi.

- archivo_avro_secc8: No se usa.
- condicionesgenerales: Convenios de la contraparte.
- dbo_tb_err: Listado de todos los errores que existen sobre los reportes.
- dbo_tb_err_val: Errores que ocurrieron durante la validación.
- dbo_TB_SIID_LOG: Guarda información de la validación de los reportes con respecto a lo que pide el Banco Central.
- dbo_TU_SIID_DFX_X: Tipo de reporte diario. Tiene 4 secciones ($\texttt{X} \in \{1, 2, 3, 4\}$).
- dbo_TU_SIID_DFX_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_DIR_X: Tipo de reporte diario. Tiene 4 secciones ($\texttt{X} \in \{1, 2, 3, 4\}$).
- dbo_TU_SIID_DIR_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_INSTITUCIONES_RELACIONADAS: Indica cuando un cliente reporta en nombre de otra empresa.
- dbo_TU_SIID_MFI_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MFX_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MIR_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_VAL_LEI: Diccionario sobre los valores LEI.

- ETL_Informe: Diccionario sobre los tipos de reporte.
- ETL_Instituciones: Información sobre el banco.
- mapeo_garantias: Garantías que ofrece el banco. Se incluyen en el reporte.
- TU_SIID_X: Conjunto de tablas que entrega información sobre el formato de los reportes ($\texttt{X} \in$ *db_tabgral*). Estas tablas se encuentran duplicadas con las de *db_tabgral*.

coopeuch

Este schema cuenta con 29 tablas auxiliares. El uso de estas tablas se destina a los procesos que se llevan a cabo en Apache NiFi.

- condicionesgenerales: Convenio de la contraparte.
- dbo_tb_err: Listado de todos los errores que existen sobre los reportes.
- dbo_tb_err_val: Errores que ocurrieron durante la validación.
- dbo_TB_SIID_LOG: Guarda información de la validación de los reportes con respecto a lo que pide el Banco Central.
- dbo_TU_SIID_DFI_X: Tipo de reporte diario. Tiene 3 secciones ($\texttt{X} \in \{1, 2, 3\}$).
- dbo_TU_SIID_DFI_X_STOCK: Tipo de reporte diario stock. Tiene 7 secciones ($\texttt{X} \in \{1, 2, 3, 5, 6, 7, 8\}$). No se usa.
- dbo_TU_SIID_DFX_X: Tipo de reporte diario. Tiene 4 secciones ($\texttt{X} \in \{1, 2, 3, 4\}$).
- dbo_TU_SIID_DFX_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_DIR_X: Tipo de reporte diario. Tiene 4 secciones ($\texttt{X} \in \{1, 2, 3, 4\}$).
- dbo_TU_SIID_DIR_X_STOCK: Tipo de reporte diario stock. Tiene 2 secciones ($\texttt{X} \in \{1, 2\}$). No se usa.
- dbo_TU_SIID_INSTITUCIONES_RELACIONADAS: Indica cuando un cliente reporta en nombre de otra empresa.
- dbo_TU_SIID_MFI_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MFX_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_MIR_X: Tipo de reporte mensual. Tiene 8 secciones ($\texttt{X} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$).
- dbo_TU_SIID_VAL_LEI: Diccionario sobre los valores LEI.
- ETL_Informe: Diccionario sobre los tipos de reporte.
- ETL_Instituciones: Información sobre el banco.
- temporary_flujos_ordenados: Tabla "temporal" para ordenar los flujos. Funciona como una tabla auxiliar.
- TU_SIID_X: Conjunto de tablas que entrega información sobre el formato de los reportes ($\texttt{X} \in$ *db_tabgral*). Estas tablas se encuentran duplicadas con las de *db_tabgral*.

Anexo F

A grandes rasgos, el funcionamiento del flujo es el siguiente.

1. Cada institución cuenta con su propio grupo de procesadores. Actualmente existen cinco instituciones clientes.

2. Dentro de cada uno de ellos se encuentran dos grupos de procesadores por cada reporte que se genera; uno tiene el fin de generar el reporte y el otro de generar un csv.

3. En los procesadores para generar el reporte se incluye la limpieza de colas y el proceso ETL. Este último cuenta con procesadores para llenar tablas auxiliares y para llenar el modelo SIID.

Banco BCI cuenta con grupos de procesadores para generar los reportes MIR, MFX, MFI, DIR, DFX y DFI, y además generar sus respectivos archivos csv. La estructura del flujo de Banco BCI es la siguiente.

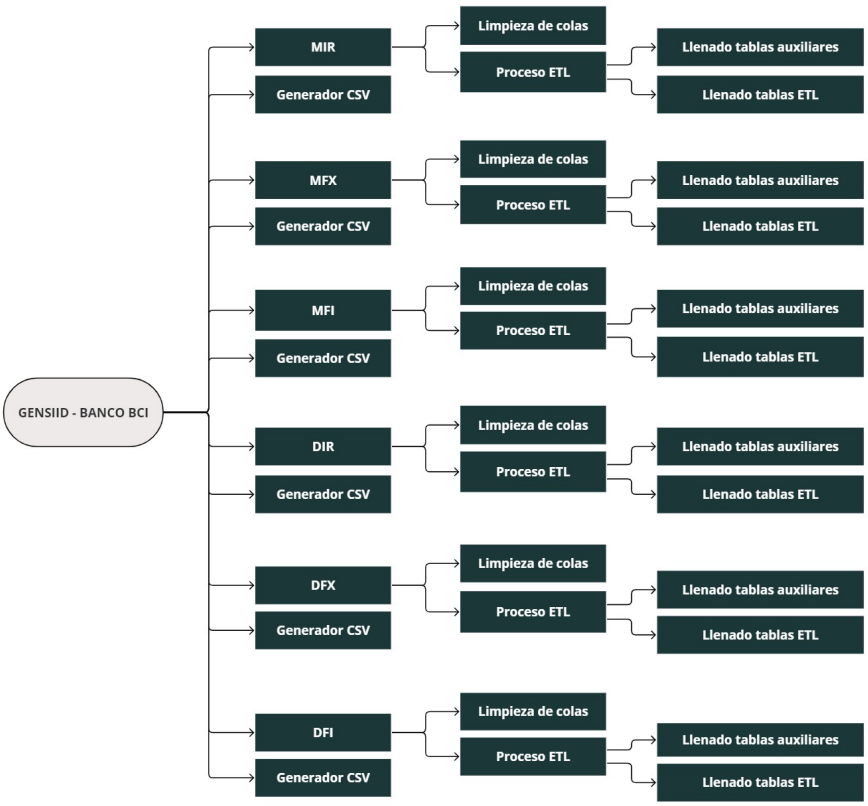


Figura 24: Grupos de procesadores de Banco BCI.

Fuente: Elaboración propia.

A continuación, se detallan los procesadores usados por el cliente Banco Bci.

Limpieza_colas_MFX

- GenerateFlowFile: genera el flujo de datos.
- FetchDistributedMapCache: obtiene la fecha de la última ejecución del reporte desde un caché distribuido de Redis.
- ExecuteGroovyScript: define el año, mes y día a partir de la fecha obtenida en el procesador anterior.
- ReplaceText: obtiene el username y password de la sesión actual de NiFi.
- GetToken (InvokeHTTP): se extrae un token de autorización de la API de NiFi.
- Token in atribute (ExtractText): guarda el token anterior como atributo en el ecosistema NiFi.
- UpdateAttribute: guarda el token anterior en un atributo "Authorization" como un Bearer Token ("Bearer" + token anterior).
- ExecuteSQLRecord: se insertan datos notificando el inicio del proceso de limpieza en la tabla log_etl_general del esquema bc.
- Funnel: distribuye el flujo de datos.
- InvokeHTTP: se conecta a la API de NiFi para obtener los Process Group asociados a todos los flujos del proyecto (viene en formato JSON).
- EvaluateJsonPath: extrae id's de los JSON Process Groups obtenidos en los procesadores anteriores.
- ExtractText: se extraen los id's del procesador anterior como texto plano.
- Get Connection id's in process group (InvokeHTTP): se llama a la API de NiFi con los id's anteriores para obtener las conexiones a todos los Process Groups.
- SplitJson: se extraen las id's de todos los procesos internos asociados a todos los Process Groups.
- Extract id, keep as attribute (EvaluateJsonPath): guarda id's de los procesos como atributos en el ecosistema NiFi.
- Drop Request (InvokeHTTP): se llama a la API de NiFi para que vacíe todas las colas de las conexiones a los procesos obtenidos anteriormente.
- MergeContent: se unen los flujos de datos en 3 requerimientos.
- MergeContent: se unen los 3 requerimientos en uno solo.
- ExecuteSQLRecord: inserta datos de Log en la tabla log_etl_general del esquema bc anunciando el fin del proceso de limpieza de colas exitoso.

Proceso_ETL_MFX_BAM

- Llenado tabla auxiliar:
  - GenerateFlowFile: genera el flujo de datos.
  - FetchDistributedMapCache: lee la fecha almacenada en el caché y lo guarda como atributo.
  - ExecuteGroovyScript: se eliminan dos atributos del caché para que los contadores partan de cero.
  - ExecuteSQLRecord: se definen fechas.
  - EvaluateJsonPath: guarda las fechas como propiedad.
  - ExecuteSQLRecord: se insertan valores en id_ejecuciones y log_etl_general del esquema bc.
  - UpdateAttribute: guardan el nombre y extensión de los archivos. Además, les asigna una prioridad.
  - Funnel: se juntan los flujos de datos.
  - FetchFile: recupera los archivos csv del servidor de Neosoft con los datos guardados anteriormente en el flujo.
  - Notify: se emiten notificaciones bajo el identificador MFX_BAM (aumenta su contador).
  - Wait: se pausa el flujo de datos → ReplaceText (se cambian algunos caracteres y se hace un split). MergeContent (se unen los datos). Log (genera nueva entrada en etl_log_general).
  - SplitText: separa archivo en partes de 100 líneas para no colapsar la memoria.
  - ConvertRecord: convierte a formato avro.
  - RouteOnAttribute: dirige los datos a procesadores específicos (fwd, movimientos y saldos).
  - PutDatabaseRecord (aux_fwd, aux_movimientos, aux_saldos): se insertan datos en las tablas auxiliares correspondientes.
  - MergeContent: se unen los requerimientos.
  - ReplaceText: se reemplazan algunos caracteres y se realiza un Split.
  - MergeContent: se unen los datos del Split anterior.
  - ExecuteSQLRecord (transformación_datos_fwd, transformación_datos_movimientos): se transforman algunos datos.
  - Notify: se emiten notificaciones bajo el identificador MFX_AUX_BAM (aumenta su contador).
  - Wait: se pausa el flujo de datos.
  - PutDatabaseRecord (Insertaren_aux_SIID): se insertan datos en una tabla auxiliar.
  - ReplaceText: se reemplazan algunos caracteres y se realiza un Split.
  - MergeContent: se unen los datos del Split anterior.

- Llenado modelo SIID
  - ExecuteSQLRecord: selecciona columnas necesarias para tabla TU_SIID correspondiente.
  - PutDatabaseRecord: llenado de tabla TU_SIID correspondiente.
  - Funnel: une el flujo de datos.
  - ReplaceText: se reemplazan algunos caracteres y se realiza un Split.
  - MergeContent: junta los requerimientos en uno solo.
  - ExecuteSQLRecord (log): genera nueva entrada en etl_log_general.

Generación CSV

- GenerateFlowFile: genera el flujo de datos.
- FetchDistributedMapCache: toma la fecha de un atributo en el flujo de datos y la almacena en un caché distribuido utilizando Redis.
- ExecuteSQLRecord: inserta valores en log_etl_general.
- Funnel: distribuye el flujo de datos en varios procesadores.
- ExecuteSQLRecord: selecciona datos.
- UpdateAttribute: entrega prioridad a los distintos flujos de datos.
- Funnel: une los flujos de datos.
- MergeContent: junta los requerimientos en uno solo.
- ReplaceText: reemplaza rut y fecha.
- UpdateAttribute: entrega un nombre al archivo.
- PutFile: define el directorio y guarda el archivo en él.
- ExecuteSQLRecord: inserta valores en log_etl_general.