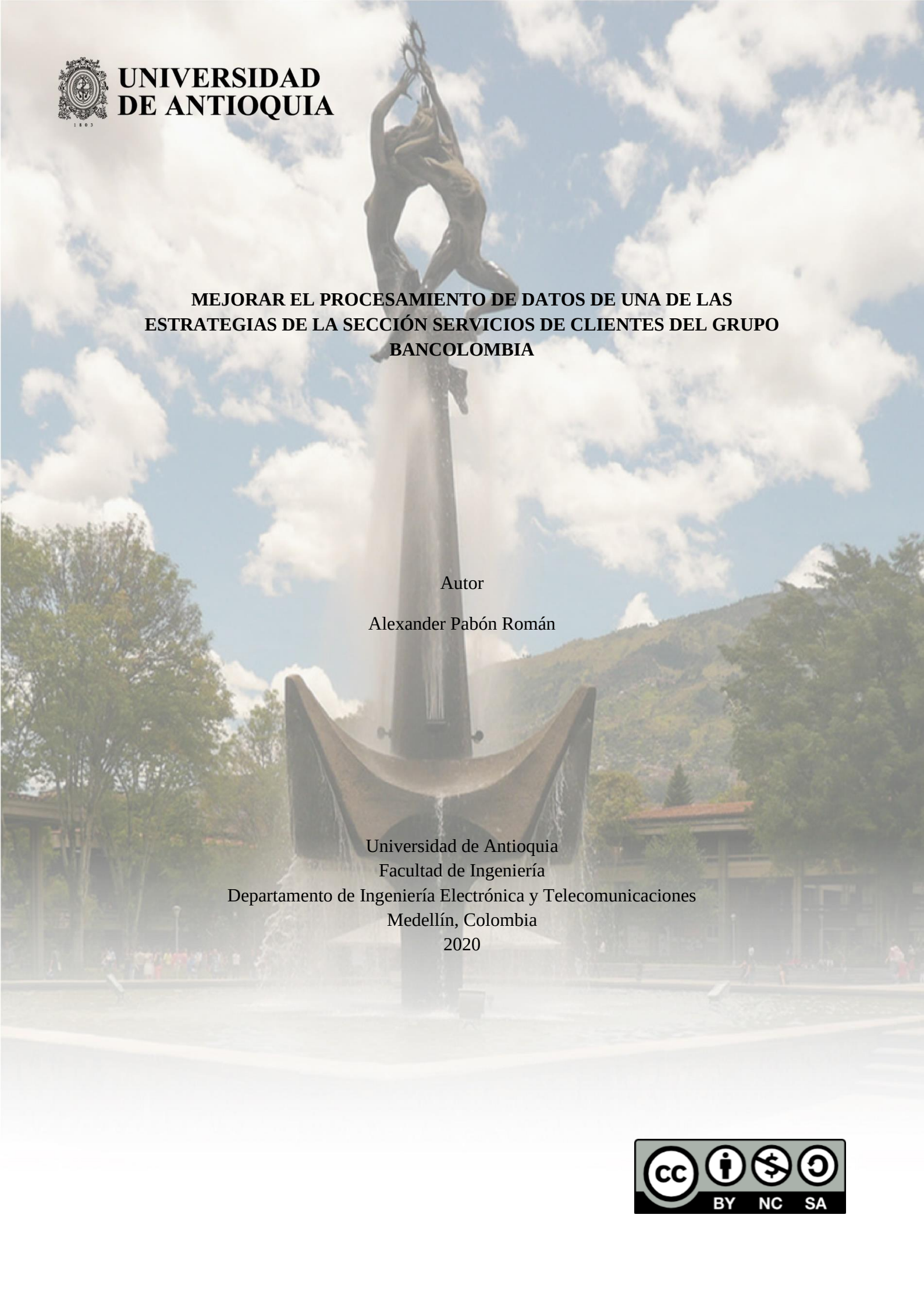


UNIVERSIDAD DE ANTIOQUIA

MEJORAR EL PROCESAMIENTO DE DATOS DE UNA DE LAS ESTRATEGIAS DE LA SECCIÓN SERVICIOS DE CLIENTES DEL GRUPO BANCOLOMBIA

Autor

Alexander Pabón Román

Universidad de Antioquia
Facultad de Ingeniería
Departamento de Ingeniería Electrónica y Telecomunicaciones
Medellín, Colombia

2020

Mejorar el procesamiento de datos de una de las estrategias de la sección servicios de clientes del Grupo Bancolombia

Alexander Pabón Román

Informe de práctica como requisito para optar al título de:
Ingeniero Electrónico.

Asesor interno
Juan Pablo Urrea
Profesor Universidad de Antioquia

Asesor externo
Juan Jose Henao Bastidas
Ingeniero Informático

Universidad de Antioquia
Facultad de Ingeniería
Departamento de Ingeniería Electrónica y Telecomunicaciones
Medellín, Colombia
2021

Tabla de contenido

Resumen

En el presente proyecto, se mejoró el procesamiento de datos de uno de los procesos que conforman la amplia sección del servicio de clientes del Grupo Bancolombia; procesos que se encargan de ejecutar políticas del banco sobre los datos de los clientes para ofrecerles productos o servicios para los que aplican cada uno. Actualmente estos procesos manejan bases de datos con alrededor de 20 millones de registros y 40 campos diferentes, cantidad que convierte el procesamiento en una realidad del Big Data.

Se tomó como punto de partida el proceso diseñado por los colaboradores del Grupo Bancolombia en el software SAS, el cual es un software estadístico para la gestión y análisis de datos; desde este se partió para evaluar su desempeño y funcionamiento, para así aplicar una estrategia diferente para crearlos nuevamente, la cual consistió en desarrollar el proceso desde el lenguaje de programación Python, acoplado con un lenguaje de consulta estructurada (SQL) y diferentes librerías que permiten a este lenguaje de programación realizar una gestión y análisis de datos similar al que presenta el software SAS.

En esta nueva versión desarrollada como código en lenguaje Python, se agregaron nuevas funciones que genera una mayor automatización, como las de enviar correos electrónicos automáticos con los resultados generados en los mismos y la publicación de resultados en rutas compartidas. Con esto, se logra aumentar la eficiencia del proceso y adicionalmente se realiza una comparación de tiempos de ejecución y la facilidad para modificar y ejecutar los mismos.

Introducción

Bancolombia es un grupo financiero multinacional colombiano cuyo negocio está distribuido en diferentes países americanos. Este grupo cuenta con más de 14 millones de clientes, de los cuales 11 millones pertenecen a Colombia, consolidando así al banco como el más grande del país por cantidad de clientes [1]. Bancolombia internamente está conformado por secciones que se encargan de diferentes trabajos o negocios propios del banco, una de estas es la sección de servicios de clientes, la cual se encarga de perfilar a los clientes del banco con el fin de saber si aplican o no para ciertos servicios o productos disponibles en el banco. Gracias a esto, en dicha sección existe un alto y constante flujo de datos, los cuales son procesados y almacenados con el fin de aplicar estrategias y actividades del banco.

Una de las estrategias de la sección servicios de clientes recibe como nombre "Preaprobar Clientes Personas y Pyme"; dentro de esta, se ejecuta el proceso llamado "Rediferidos tarjetas de crédito", el cual ha sido creado mediante el software SAS (Statistical Analysis System) que es un software de análisis de datos y está dirigido

para usuarios no técnicos en el ámbito del procesamiento y almacenamiento de datos [2].

SAS, al estar dirigido para usuarios no técnicos tiende a crear en múltiples ocasiones procesos extensos e ineficientes, creando múltiples nodos y descargando bases de datos innecesarias. Estas y otras situaciones afectan los diferentes procesos que conforman la sección de servicios de clientes; en cada proceso existen alrededor de 100 nodos como se observa en la figura número 1. Estos nodos ejecutan de a una instrucción SQL y para cada ejecución se deben modificar gran parte de estos ya que las condiciones de ejecución cambian periódicamente, situación que retrasa el proceso. La siguiente imagen muestra una pequeña parte de uno de los procesos desarrollado en SAS, donde se puede evidenciar la gran cantidad de nodos que conforman el mismo. En el anexo número 1 se puede ver completamente este proceso.

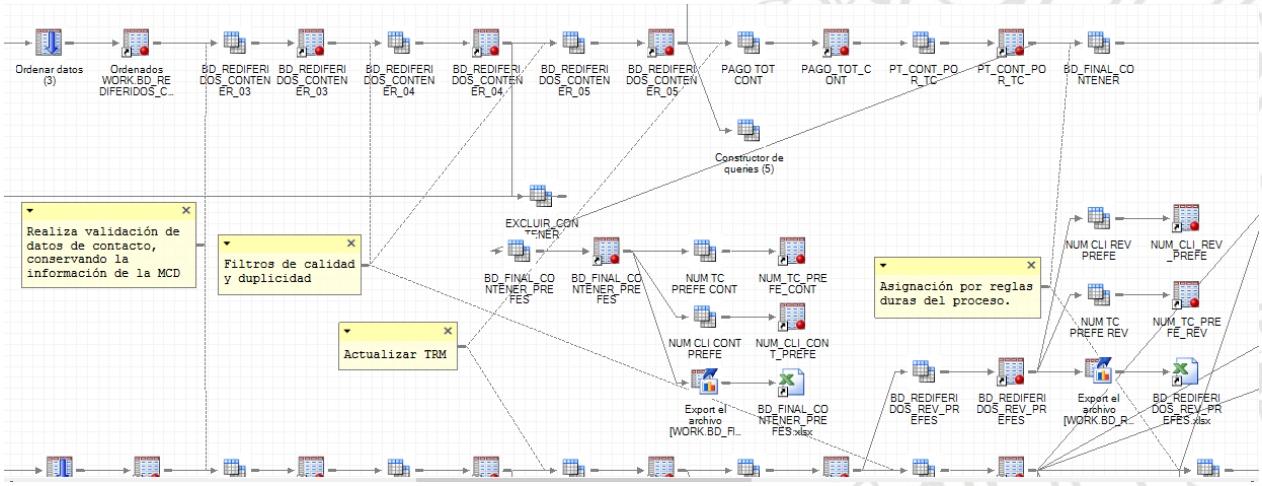


Figura 1. Porción de proceso "Rediferidos tarjetas de crédito" en SAS.

Como se evidencia en la anterior imagen, durante todo el proceso se descargan al espacio de trabajo bases de datos no necesarias para el análisis del comportamiento del proceso, este problema genera gran retraso debido a los grandes volúmenes de cada una de estas (alrededor de 20 millones de datos); todo esto hace que el proceso tarde entre un 60% y un 80% más del tiempo estimado.

Debido a la situación planteada, la sección de servicios de clientes se ve en la necesidad de reestructurar y mejorar sus procesos con el fin de aumentar su eficiencia de trabajo; para ello se planteó crear nuevamente los procesos teniendo el lenguaje de programación Python como base de estos, acoplándose con el lenguaje de consulta estructurada SQL mediante librerías como pyodbc y sparky que son propias de Python; todo esto con el fin de automatizar y mejorar la trata de datos dentro de la sección, donde el colaborador del grupo solo debe ejecutar los códigos y analizar

los resultados. En esta nueva versión se añade nuevas funciones como la de publicación de datos en las carpetas compartidas necesarias y vía correo electrónico de forma automática, con el fin de evitar este proceso al colaborador, ya que esta información se distribuye a gran cantidad de interesados de una forma estratégica.

Objetivos

Objetivo general

Automatizar el procesamiento de datos del proceso "Rediferidos tarjetas de crédito" perteneciente a la estrategia "Preaprobar Personas y Pyme" de la sección servicios de clientes del Grupo Bancolombia, mediante la implementación de este en lenguaje Python acoplado con consultas estructuradas SQL, con el fin de aumentar la eficiencia de trabajo respecto a tiempos de ejecución.

Objetivos específicos

- Evaluar las consultas SQL generadas en la versión anterior de los procesos en el software SAS, para reestructurar e integrarlas a la nueva versión en código Python.
- Desarrollar una nueva versión en código Python, aprovechando la versatilidad del lenguaje para la manipulación de bases de datos, acoplándose con consultas estructuradas SQL mediante las librerías pyodbc y sparky.
- Generar documentación e instructivos para el proceso, los cuales puedan orientar y ayudar a cualquier colaborador del Grupo Bancolombia a la hora de ejecutar cada uno.
- Evaluar y comparar el desempeño de la nueva versión del proceso respecto a su antigua versión en el software SAS.

Marco Teórico

En esta sección se dejarán claros los conceptos fundamentales para comprender el desarrollo del proyecto, desde la información de los lenguajes de programación hasta los conceptos básicos de las estrategias del Banco.

En una primera instancia se abordan los diferentes conceptos referentes al Grupo Bancolombia, los cuales se deben comprender para tener claridad del punto de partida del proyecto.

Estrategia – "Preaprobar clientes personas y pyme"

Una de las principales estrategias presentadas en la sección de servicios de clientes, recibe como nombre "Preaprobar Clientes Personas y Pyme". Esta estrategia tiene como objetivo aplicar las diferentes políticas de riesgo, segmentos y productos para los clientes que cumplan con la perfilación requerida para preaprobar cupos de productos de riesgo como: tarjetas de crédito, créditos y carteras, aumentos de cupo; o para la oferta de seguros, con el fin de apalancar la rentabilidad del grupo. Para cumplir con esto, la ejecución del proceso deberá generar la base de clientes potenciales a obtener un preaprobado mediante reglas demográficas, políticas de riesgo, segmento y producto. El proceso de perfilación genera dos bases de datos:

1. Total de Clientes Bancolombia perfilados, incluye todas las reglas de negocio.

2. Clientes potenciales para asignar un cupo y un producto. Incluye los filtros de negocio para generar la base de clientes susceptibles de preaprobados.

Así pues, el proceso de **Rediferidos tarjetas de crédito** es una aplicación de la estrategia mencionada, el cual se diferencia de los demás por las políticas aplicadas y por el producto a ofrecer.

Por otra parte, es fundamental conocer ciertos conceptos y herramientas que son necesarios para llevar a cabo el proyecto.

Términos

Comenzando por el termino ***Hadoop***, el cual se refiere a una estructura de software de código abierto para almacenar datos y ejecutar aplicaciones en clústeres de hardware comercial, además proporciona almacenamiento masivo para cualquier tipo de datos y ofrece un enorme poder de procesamiento [3]. Este se convierte en un término importante al conocer que el Grupo Bancolombia efectúa su procesamiento de datos haciendo uso de ***Cloudera Impala*** que es un motor de consultas MPP SQL de código abierto totalmente integrado y de última generación, diseñado específicamente para aprovechar la flexibilidad y escalabilidad de Hadoop [4].

Lenguajes, librerías y software

Python

Python es un lenguaje de programación interpretado de tipado dinámico cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma y disponible en varias plataformas [5]. Dicho de otro modo, Python es:

- Interpretado: Se ejecuta sin necesidad de ser procesado por el compilador y se detectan los errores en tiempo de ejecución.
- Multiparadigma: Soporta programación funcional, programación imperativa y programación orientada a objetos.
- Tipado dinámico: Las variables se comprueban en tiempo de ejecución.
- Multiplataforma: disponible para plataformas de Windows, Linux o MAC.
- Gratuito: No dispone de licencia para programar.

Python se toma como base para el desarrollo de los proyectos mencionados debido a la versatilidad que este presenta para el análisis de datos mediante las funciones descritas a continuación.

Pyodbc

Como es mencionado anteriormente, la solución para este proyecto es la construcción de un código en el lenguaje de programación Python unido a su vez con el lenguaje de consulta estructurada SQL. Esto se da gracias al acoplamiento de esto mediante el uso del estándar ODBC y el módulo PYODBC, los cuales se refieren a un estándar de acceso a las bases de datos desarrollado por SQL Access Group (SAG) en 1992, con el objetivo de hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué sistema de gestión de bases de datos almacene los mismos [6].

Spark

Apache Spark es un sistema de procesamiento distribuido de código abierto que se utiliza para cargas de trabajo de big data. Utiliza el almacenamiento en caché en memoria y la ejecución de consultas optimizadas para realizar consultas rápidas con datos de cualquier tamaño [7]. En pocas palabras, Spark es un motor rápido y general para el procesamiento de datos a gran escala.

La parte general significa que se puede usar para múltiples aplicaciones, como ejecutar SQL distribuido, crear canalizaciones de datos, ingerir datos en una base de datos, ejecutar algoritmos de aprendizaje automático, trabajar con gráficos o flujos de datos y mucho más.

Sparky-Bc ofrece la posibilidad de integrar este sistema a Python, dando la facilidad y versatilidad requerida para el procesamiento de datos desde el lenguaje de programación.

Pandas

Pandas es una biblioteca de software escrita como extensión de NumPy para manipulación y análisis de datos para el lenguaje de programación Python. En

particular, ofrece estructuras de datos y operaciones para manipular tablas numéricas y series temporales [8].

Statistical Analysis System (SAS)

SAS es un paquete de software estadístico desarrollado por SAS Institute para la gestión de datos, análisis avanzado, análisis multivariante, inteligencia empresarial, investigación criminal y análisis predictivo. SAS puede extraer, alterar, administrar y recuperar datos de una variedad de fuentes y realizar análisis estadísticos sobre ellos; proporciona una interfaz gráfica de usuario para usuarios no técnicos [9].

En este proyecto, los procesos realizados en SAS son el punto de partida, desde donde se analiza su construcción y funcionamiento, para así partir a la programación en Python.

Landing Zone

El aljibe de datos, también conocido como la LZ ("Landing Zone"), es la infraestructura de almacenamiento y análisis de grandes cantidades de información utilizada por el Grupo Bancolombia [10]. Esta infraestructura la conforman grandes componentes como las tecnologías Hadoop y HDFS mencionadas en el proyecto.

Finalmente para ilustrar todo el proceso anteriormente mencionado, se presenta la figura 2, la cual permite visualizar las diferentes interacciones del proceso, comenzando por el lenguaje de programación de Python que se tiene como el centro y controlador de todas las acciones, recibiendo los parámetros de entrada, conectándose con el motor de consultas IMPALA que a su vez es el encargado de ejecutar toda consulta SQL requerida, posteriormente Python recibe de nuevo los resultadas entregados por las bases de datos y realiza una toma de decisiones respecto a ellos, generando nuevas consultas o procesando estos resultados para luego exportarlos en archivos de formato Excel.

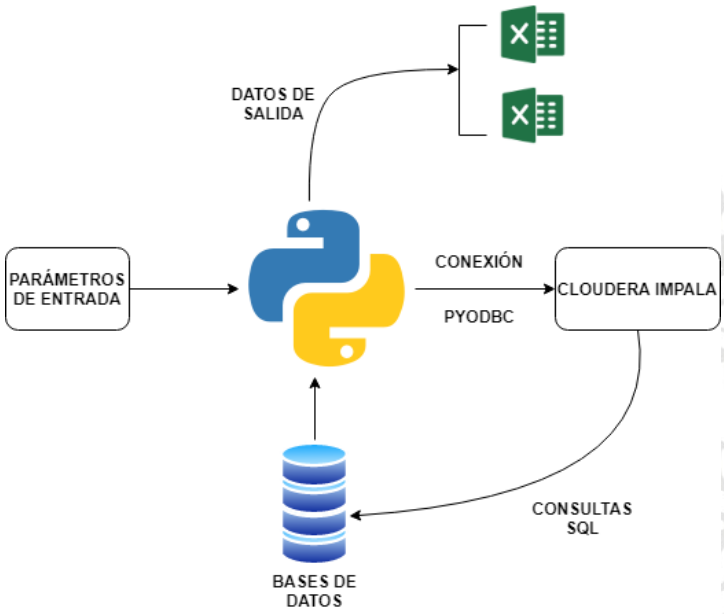


Figura 2. Estructura general de los procesos.

Metodología

Para el desarrollo del proyecto se siguió la siguiente metodología.

Fase 1: **Capacitación frente al funcionamiento de la sección de clientes**

Para dar inicio con este proyecto se deben comprender los antecedentes, el funcionamiento y la finalidad de cada proceso, para así conocer sus problemáticas y necesidad; es por esto por lo que se inicia el proceso de prácticas con la capacitación de los diferentes procesos que conforman la sección, principalmente el que se va a intervenir.

Actividad 1: Estudio de la estrategia "Preaprobar Clientes Personas y Pyme".

La capacitación comienza desde lo más general que es la estrategia "Preaprobar Clientes Personas y Pyme" la cual no es un proceso como tal, por lo contario, es una definición de políticas y productos que está conformada por múltiples procesos y es la sustentación de estos.

Actividad 2: Estudio del proceso "Rediferidos tarjetas de crédito"

Como se menciona anteriormente, el proceso intervenido en el proyecto de prácticas es "Rediferidos de tarjetas de crédito", es por esto por lo que la actividad dos se realiza con el fin de comprender desde sus fines de políticos y lucrativos hasta su funcionamiento en Statistical Analysis System (SAS).

Fase 2: Evaluar desempeño

Como su nombre lo indica, la segunda fase se centra en evaluar el desempeño de los procesos que el Grupo Bancolombia desarrolló en el Software Statistical Analysis System. En estos, se examina la necesidad de cada consulta SQL que se realiza dentro de los ellos, especialmente las consultas generadas por la opción de "Constructor de Querys" que presenta el software SAS y es mencionada anteriormente; se verifica si cada consulta es fundamental dentro del desarrollo del proceso o si, por el contrario, puede ser reemplazada o incluida dentro de una consulta ya existente.

Adicionalmente, como se evidencia en la figura 1, gran cantidad de los datos procesados son descargados de la "Landing zone" al espacio de trabajo de SAS, proceso que toma gran cantidad de tiempo al considerar el volumen de las bases de datos; es por esto que se evalúa la necesidad de la descarga de datos, eligiendo cuales salidas o tablas son fundamentales para analizar y descargar, las demás se toman para ser procesadas directamente en la Landing zone, donde no se tiene una vista directa de los resultados de cada operación.

Fase 3: Desarrollar procesos en código Python

Se pasa ahora a la fase 3, la cual es el centro del proyecto ya que consiste en un nuevo desarrollo para los procesos mencionados, esto directamente en Python, dejando atrás el uso del software SAS y aprovechando los diferentes módulos que ofrece Python para este tipo de proyectos.

Se comienza realizando la conexión entre el lenguaje de programación y el motor de consultas Impala mediante el módulo de Python PYODBC, que es mencionado en ocasiones anteriores; permitiendo el acceso a la Landing zone del Grupo Bancolombia, desde este lenguaje de programación.

Es importante tener en cuenta que para poder llevar a término el punto anterior, es necesario estar conectado a una de las VPN del banco, que dan acceso a diferentes aplicativos, archivos y procesos. Adicionalmente, el usuario a ejecutar los procesos debe contar con permisos especiales, los cuales le dan acceso a ciertos puntos o zonas dentro de la Landing zone, es por esto por lo que se requiere acceder a Impala con un usuario que tenga habilitado los insumos y las zonas a modificar dentro del proceso.

Luego de una conexión exitosa, se realiza el procesamiento de los datos mediante consultas SQL, implementando tanto consultas evaluadas en el punto anterior, como nuevas consultas que son necesarias para que el procesamiento complete correctamente los resultados requeridos.

Por último, se implementan en Python diferentes funciones y opciones que este ofrece para la estadística y el análisis de datos; adicionalmente, se agregan funciones que ayudan al usuario a realizar tareas efectivas luego del procesamiento de datos, como

lo son la publicación de resultados vía correo electrónico a las diferentes empresas o proveedores que hacen uso de estos, compartiendo los mismos en rutas compartidas de propiedad del banco, esto de forma automática claramente.

Fase 4: Documentación

En la cuarta fase se creó un instructivo para cada proceso intervenido donde se orienta al colaborador encargado sobre el funcionamiento de este. En estos instructivos se especifica lo siguiente:

- Paso a paso para ejecutar el proceso.
- Salida esperada para cada etapa del proceso.
- Lugares donde debe modificar el código en caso tal de que cambien las políticas para el proceso.

Fase 5: Desempeño

Por último, se evalúa y compara el desempeño de las nuevas versiones de procesos desarrolladas en Python, respecto a las versiones anteriores desarrolladas en SAS. Se compararon diferentes medidas como el tiempo de ejecución, la calidad y precisión de los archivos de salida, además de la facilidad y comodidad a la hora de ejecutar.

Resultados y análisis

Como se menciona durante todo el proyecto, se crea un código desarrollado en Python el cual será analizado desde diferentes puntos en las siguientes secciones.

Es importante conocer que el Grupo Bancolombia, como muchas compañías, restringen su información y sus aplicativos especialmente para los colaboradores de la compañía. Por lo tanto, para ejecutar el código entregado, se debe contar con un usuario que tenga acceso de lectura y escritura a las bases de datos.

Conexión con las bases de datos

Luego de estar conectado a la red privada del banco por medio de una VPN como requisito, el código inicia identificando el usuario que está haciendo uso del sistema y se conecta con Impala y con la Landing Zone, verificando que dicho usuario si cuente con permisos para esto. En la figura número 3 se muestra el modo de conexión haciendo uso de la librería "Pyodc". Por otra parte, la tabla número 1, despliega un dato importante y es el tiempo que tarda en realizarse la conexión en dos días diferentes, el primer dato, corresponde a un día de poco procesamiento de datos dentro del banco y el segundo corresponde a una quincena, fecha donde se ejecutan la gran mayoría de los procesos. Estos resultados muestran que en una fecha de gran afluencia la conexión y el procesamiento en general tarda un mayor tiempo;

si bien estos dos tiempos no demuestran una gran diferencia, si se hace notable luego de que se procesan millones de datos.

```
#Se realiza conexión
CONN_STR = "DSN=impala_prod"

try:
    print("INICIA CONEXIÓN CON IMPALA")
    connection = pyodbc.connect(CONN_STR, autocommit = True)
    print("Conexion exitosa")
except:
    print("Error en conexión")
    sys.exit(1)
```

Figura 3. Conexión con la Landing Zone.

Tabla 1. Tiempo de conexión en momentos diferentes

Fecha	13/12/2020	15/12/2020
Tiempo de conexión	0.78 segundos	1.72 segundos

Procesamiento

Posterior a una conexión exitosa, se continua con el procesamiento y la ejecución de las políticas del proceso presente, para este caso serían las políticas de "Rediferidos tarjetas de crédito".

La siguiente figura muestra la filosofía de la ejecución en sí, donde se leen un grupo de consultas extraídas de archivos SQL y se ejecutan todas secuencialmente. Este segmento de la ejecución es el fundamento del proceso y es donde se realiza el gran cambio respecto al proceso implementado en el software SAS adjunto en el anexo 1.

```
file_name_3='/SQL/4_Ctrl_Campos_Perf.sql'
funcs.runQueriesV2(path, file_name_3, connection)

#4.---- Estructuras Bases de datos ----
file_name_4='/SQL/5_Estructuras_BD.sql'
funcs.runQueriesV2(path, file_name_4, connection)

#5.---- Estadisticas perfilacion ----
file_name_5='/SQL/6_Estadis_Perf.sql'
funcs.runQueriesV2(path, file_name_5, connection)

#6.---- Distribucion de clientes ----
file_name_6='/SQL/7_Distribucion.sql'
archivo = open(path+file_name_6, 'r')
distribucion=archivo.read()
```

Figura 4. Segmento de código, ejecución de consultas.

La figura 4 muestra cómo se lee cada grupo de consultas y luego se ejecuta haciendo uso de "funcs", esta consta de diferentes funciones que hacen uso de las librerías *pandas* y *Sparky* con el fin de cumplir el objetivo de ejecutar cada consulta.

Por otra parte, como se muestra en la figura 1, el proceso en SAS cuenta con 110 nodos que ejecutan de a una consulta SQL, adicionalmente, se descargan 100 tablas al espacio de trabajo del software y finalmente solo se exportan 3 de estas tablas como resultados finales. Para el caso del código desarrollado en Python, se crean las consultas fundamentales para el cumplimiento del proceso y se agrupan según la función a cumplir dentro del proceso. A continuación, se explica la función principal de cada grupo de consultas.

- **Controles de insumos**

  Como su nombre lo indica, el fin es tener un control sobre los diferentes insumos que son necesarias para la ejecución del flujo. Los controles principalmente se realizan teniendo una trazabilidad de la cantidad de datos de cada insumo, es decir, se compara la cantidad de datos de cada tabla para la ejecución, con la cantidad de datos para ejecuciones anteriores y de esta forma identificar si se lleva un mismo volumen en cada ciclo y no existe un error en las entradas a este. Los volúmenes entregados para cada ejecución son guardados en un archivo de Excel con el fin de llevar la trazabilidad. Adicionalmente, se agrega la función para desplegar la cantidad de datos de forma gráfica, que el colaborador encargado ejecuta para identificar si existe algún error en los insumos. En la gráfica número 5 se puede observar como el insumo llamado "Master Customer Data" tiene un buen comportamiento en la ultimas ejecuciones al llevar volúmenes muy similares.

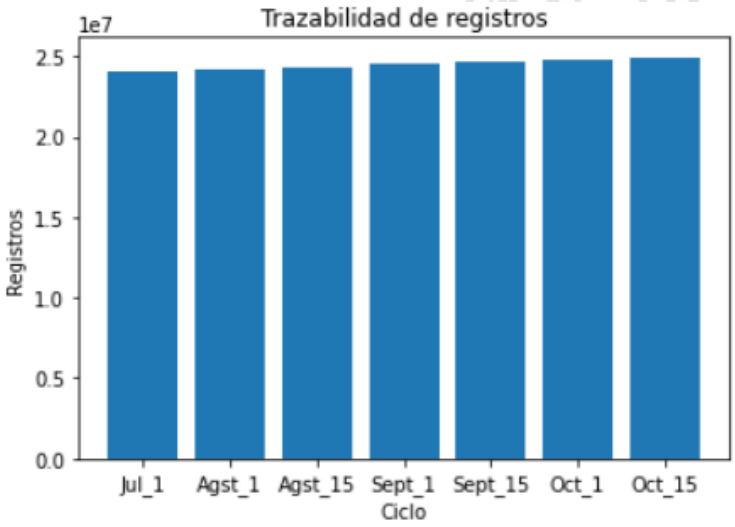


Figura 5. Trazabilidad de los insumos.

- **Perfilación**

  Este archivo de "perfilación" contiene un grupo de consultas SQL que como su nombre lo indica, tiene como fin perfilar a todos los clientes para determinar si cumple con las reglas del negocio y clasificarlo para ofrecerle el producto.

  Para comprender completamente el fin de la perfilación se debe conocer que los clientes que quedan perfilados para el producto se entregan a cuatro empresas diferentes para ser contactados; estas empresas son Emergia, Domina, Brm y Gmv. Adicionalmente, los clientes son catalogados en dos categorías las cuales son "Contener" y "Revolvente". Estas dos categorías corresponden a los perfiles de los clientes, la primera se refiere a clientes óptimos identificados en los modelos de riesgos, con características de montos y saldos de deuda, mientras que la segunda corresponde a perfiles de clientes que en el último año presentan alguna condición de riesgo y según el monto del pago mínimo se le oferta el servicio.

  Por lo tanto, el fin de la perfilación es entregar una base de datos con todos los clientes que apliquen para el producto, donde deben estar marcados con la empresa a la que lo van a entregar y catalogados como "Contener" o "Revolvente".

- **Control de campos perfilación**

  El fin de estas consultas es tener un control a la salida de la base de datos entregada por la etapa anterior. Esto se realiza haciendo un conteo y examinando diferentes campos, algunos ejemplos son:

  - Se examina los teléfonos de contacto de cada cliente verificando si son correctos estructuralmente. Si es un número fijo verificar la longitud y que no comience por el número 1. Si es un numero de celular, verificar 10 dígitos y que comience por 3.
  - Se realiza un conteo de cuantos clientes son catalogados como "Contener" o "Revolvente".
  - Se realiza un conteo de cuantos clientes son dirigidos para cada empresa (Emergia, Domina, Brm, Gmv).

  De esta forma se realiza el control sobre los diferentes campos importantes en la salida de la perfilación, esto con el fin de analizar el comportamiento del proceso y los datos durante cada ejecución.

- **Distribución**

  En cada ejecución del proceso (cada 15 días) ciertas reglas de negocio varían, una de esas reglas es el volumen de clientes que se le entregan a las

empresas Emergia, Brm y Domina. Para la empresa GMV no se especifica cuantos clientes se deben entregar en la ejecución.

Se da un ejemplo para comprender el funcionamiento. Se supone que para una ejecución puntual se tienen las cantidades de 90.000, 80.000, 65.000 que corresponden a clientes a entregar para las empresas Emergia, Brm y Domina respectivamente. El proceso se debe encargar de extraer clientes de la base de "GMV" con el fin de completar las cifras necesarias para las otras tres empresas.

Es importante mencionar que, en las versiones anteriores, este proceso se realizaba de forma manual. Luego de tener las bases para cada empresa se calculaban los volúmenes y si no estaban completos los clientes para las 3 compañías principales, se extraía de la base de Gmv para ingresar a las demás.

- **Generar salidas**

  Finalmente, las últimas consultas tienen el fin de generar las salidas distribuidas según la empresa y la categoría. Es decir, para cada empresa se genera una base de datos con clientes "Contener" y una con clientes "Revolvente", para un total de 8 salidas que son compartidas a cada empresa.

Publicación de resultados

Una de las tareas del Colaborador luego de ejecutar el proceso es distribuir la información a las empresas y a las personas interesadas en esta. Al ser una tarea repetitiva, se destina un segmento de código para realizar esta distribución de forma automática, tanto vía correo electrónico como en rutas compartidas, como se logra identificar en la figura 7. Esto se realiza con el fin de automatizar y mejorar los tiempos en todo lo que respecta con "Rediferidos tarjetas de crédito".

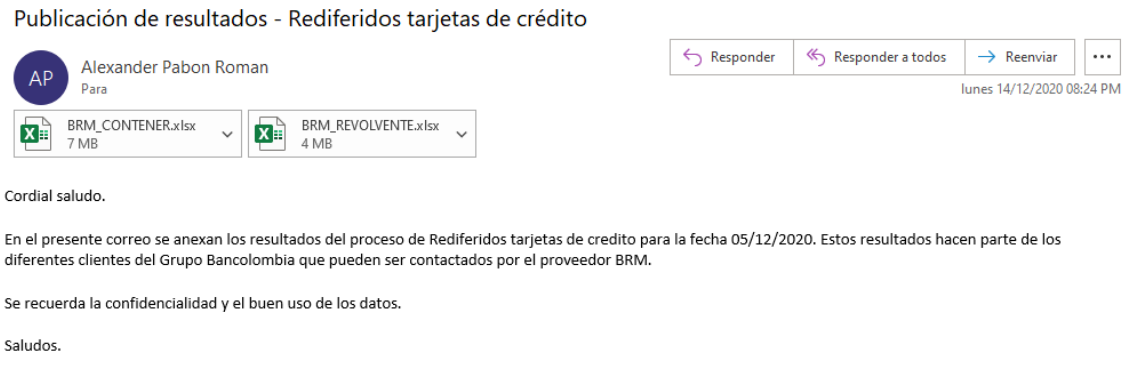


Figura 7. Publicación de información vía correo electrónico.

Comparación de versiones del proceso

Como se evidencia durante el proyecto, el código desarrollado en Python es una mejora desde diferentes aspectos a un proceso desarrollado inicialmente en SAS; es por esto por lo que se comparan desde diferentes puntos.

- **Posibilidad de cambios**

  Uno de los principales problemas del desarrollo en el software SAS es el uso del "Constructor de consultas" que guía al colaborador por medio de una interfaz para crea la consulta necesaria. Esto genera un gran conflicto cuando el proceso no es estático en cada ejecución, como es para este caso. El proceso de Rediferidos tarjetas de crédito tiene cambios en sus políticas para cada ejecución (cada 15 días), algunos ejemplos de esos cambios pueden ser el monto de dinero dentro de las tarjetas de crédito del cliente, o a qué tipo de persona estará dirigida la ejecución (persona Jurídica, persona natural); gracias a esto, las consultas que se ejecutan deben ser modificadas para cada ejecución y ahí es donde se presenta problema con el constructor de consultas, ya que este, una vez creado, tarda aproximadamente 40 segundos solo para abrir y permitir modificarse. Este problema genera un retraso de aproximadamente de 30 minutos, teniendo en cuenta que cada ejecución se modifica alrededor de 40 consultas.

  Gracias a que existe una gran cantidad de consultas SQL que se ejecutan dentro del proceso, para el desarrollo en Python, se decide agruparlas según su finalidad, con el fin de manipular y modificar dichas consultas de una forma cómoda y ordenada. Es claro que para este caso si se requiere modificar una consulta, solo se debe localizar en el archivo SQL correspondiente, no es necesario esperar un tiempo adicional como en el caso anterior.

  Se debe tener en cuenta que para el desarrollo en Python se redacta un instructivo donde se explica la finalidad de cada grupo de consultas, con el fin de especificar donde se debe modificar el código según la necesidad del cambio.

- **Tiempos del proceso**

  Con el fin de realizar una comparación entre ambas versiones, se presentan las tablas 2 y 3 que consolidan los diferentes tiempos que son empleados para todo el proceso de Rediferidos tarjetas de crédito en las dos versiones que se tienen para este.

Tabla 2. Tiempos del proceso versión SAS.

Versión en SAS	
Concepto	**Tiempo**
Modificación del proceso	90 minutos
Ejecución	120 minutos
Análisis de resultados	90 minutos
Publicación de resultados	30 minutos
Total	330 minutos

Tabla 3. Tiempos del proceso versión Python.

Versión en Python	
Concepto	**Tiempo**
Modificación del proceso	45 minutos
Ejecución	15 minutos
Análisis de resultados	45 minutos
Publicación de resultados	10 minutos
Total	115 minutos

El tiempo total de ejecución del proceso es divido en varios conceptos o etapas que cumplen una tarea específica dentro del este, cada una de estas etapas es explicada detalladamente a continuación.

- **Modificación del proceso**

La etapa de "Modificación del proceso" consiste en adecuar el proceso para la ejecución actual, como se explica en momentos anteriores, el proceso es ejecutado cada 15 días y para cada ejecución cambian algunas reglas del negocio, lo cual impacta gran parte de la consulta SQL que se realizan dentro de este. En las tablas anteriores se nota una diferencia de 45 minutos (50%) en este concepto y esto es dado por dos razones fundamentales; la primera se refiere a la estructura del proceso, como se evidencia tanto en la figura 1 como

en el anexo 1, la versión para SAS no tiene una estructura definida, simplemente son 110 consultas individuales distribuidas durante todo el proyecto, lo cual dificulta el comprender que lugares o consultas se deben modificar para cumplir un objetivo, obligando al colaborador a realizar una búsqueda exhaustiva para encontrar las consultas a modificar.

La segunda razón y la que hace la gran diferencia respecto a tiempos, se refiere a una debilidad del software SAS al utilizar el asistente para la construcción de consultas. Para poder modificar una consulta creada de esta forma, se debe de abrir, este proceso de apertura toma aproximadamente 40 segundos para mostrar el diseño de la consulta y permitir realizar modificaciones, problema que impacta de gran medida a esta versión en SAS ya que si se tiene en cuenta que está creada por 110 consultas de las cuales se deben modificar aproximadamente 45 para cada ejecución. Es por esto por lo que los tiempos de modificación en esta versión se incrementan.

En conclusión, este tiempo se logra disminuir para la versión en Python por las mismas dos razones. Primero se plantea una estructura consolidada para el proceso, donde se agrupan las consultas por su finalidad y por sus antecedentes, es decir que para modificar algo en específico se puede encontrar una forma más ágil. Como segundo punto se tiene la ventaja que para este caso no se deben "abrir" nodos para modificarlos, simplemente contar con el script requerido para su modificación.

- **Ejecución**

Esta etapa se refiere a los tiempos de procesamiento de los datos, tiempo que se toma desde que se ejecuta el proceso. En esta etapa se logra identificar la mayor diferencia en cuestión de tiempos y esto es gracias a las razones que se explican a continuación.

La primera razón y las más importante es la descarga de datos al espacio de trabajo. Cuando se utiliza el constructor de consultas se crea un nodo similar al que se muestra en la figura 8, estas consultas solo se pueden realizar a tablas cuyos datos están descargados en el espacio de trabajo del software, es decir que, obliga al colaborador a descargar todas las tablas temporales que solo utiliza para llegar a un resultado final. Esta situación lleva a descargar aproximadamente 100 tablas temporales, situación que se convierte en problemática al saber que cada tabla contiene alrededor de 40 columnas y 450.000 filas, tardando un promedio de 60 segundos para su descarga.

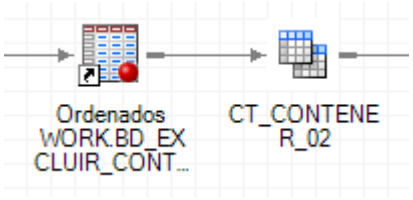


Figura 8. Ejemplo de constructor de consultas.

La segunda razón se refiere al número de consultas que se ejecutan para completar el proceso. En la versión de Python se utilizan únicamente consultas necesarias para llegar a un resultado final, es por esto por lo que el número de estructuras SQL para esta versión, disminuye considerablemente. Como se menciona en momentos anteriores, la versión en SAS contiene un total de 110 consultas, mientras que la versión de Python contiene un total de 70.

La nueva versión del proceso se enfoca entonces en el procesamiento de los datos directamente en la Landing Zone con el fin de que no exista la necesidad de descargar datos que son temporales. Dentro de la Landing Zone se crean estas tablas temporales que no son descargadas, como se evidencia en la figura 9, la cual muestra la lógica de ejecución de las estructuras SQL, en esta se puede evidenciar que se crean 3 tablas temporales que al final se unen para llegar a un resultado concreto. De esta forma se evita el procesamiento local y se permite el procesamiento en la Landing Zone.

```
drop table if exists proceso_consumidores.clientes_aliado purge;
create table proceso_consumidores.clientes_aliado stored as parquet as /*Tabla final*/
    conteo_domina as( /*Tabla temporal*/
        select "DOMINA" as Aliado,
        count(DISTINCT identificacion) as Registros
        from domina
    ),
    conteo_emergia as( /*Tabla temporal*/
        select "EMERGIA" as Aliado,
        count(DISTINCT identificacion) as Registros
        from emergia
    ),
    conteo_brm as( /*Tabla temporal*/
        select "BRM" as Aliado,
        count(DISTINCT identificacion) as Registros
        from brm
    )
    select * from conteo_gmv UNION ALL
    select * from conteo_domina UNION ALL
    select * from conteo_emergia UNION ALL
    select * from conteo_brm;
```

Figura 9. Ejemplo de procesamiento en Landing Zone.

A continuación, la tabla 3 muestra los resultados en tiempo de ejecución de los diferentes grupos de consultas que se realizaron. Estos grupos se conforman

de diferentes estructuras que conllevan a un resultado común para la etapa correspondiente.

Tabla 3. Rendimiento proceso en Python.

Segmento	Tiempo en segundos
Ctrl_Insumos	16.67
Perfilación	185.13
Crtl_Campos_Perf	20.18
Estructuras_BD	101.19
Estadis_Perf	37.6
Distribución	58.6
Resumen_Aliado	20.3
Generar_Layouts	190
Cascadas	39.5

Estos tiempos de ejecución disminuyen enormemente por las razones ya mencionadas, básicamente por el procesamiento directo en la LZ. Adicionalmente estos tiempos demuestran el gran potencial de diferentes herramientas de ambiente de almacenamiento que se da en el banco; comenzando por el procesamiento masivo en paralelo que ofrece el motor de búsqueda Cloudera Impala, el cual a su vez hace uso de Apache Hadoop para poder explotar la ideología de clúster de computadoras. Es decir que el procesamiento realizado para llevar a cabo las políticas de "Rediferidos tarjetas de crédito" no se procesa en una máquina local, por el contrario, se ejecuta mediante múltiples computadoras que activan nodos a trabajar en una misma tarea, cumpliendo el entorno del clúster de computadoras y la tecnología Hadoop.

Es importante reconocer que SAS también tiene la oportunidad de conectarse y realizar su procesamiento directamente en el ambiente del banco llamado Landing Zone, es por esto por lo que a modo de prueba se crea una tercera versión del proceso, esta vez enfocada en este software. Esta versión fue creada solo para ejecutar las consultas SQL agrupadas y mencionadas en momentos anteriores, como se muestra en la figura 10, el desarrollo está conformado por diferentes programas que contiene cada grupo de consultas.

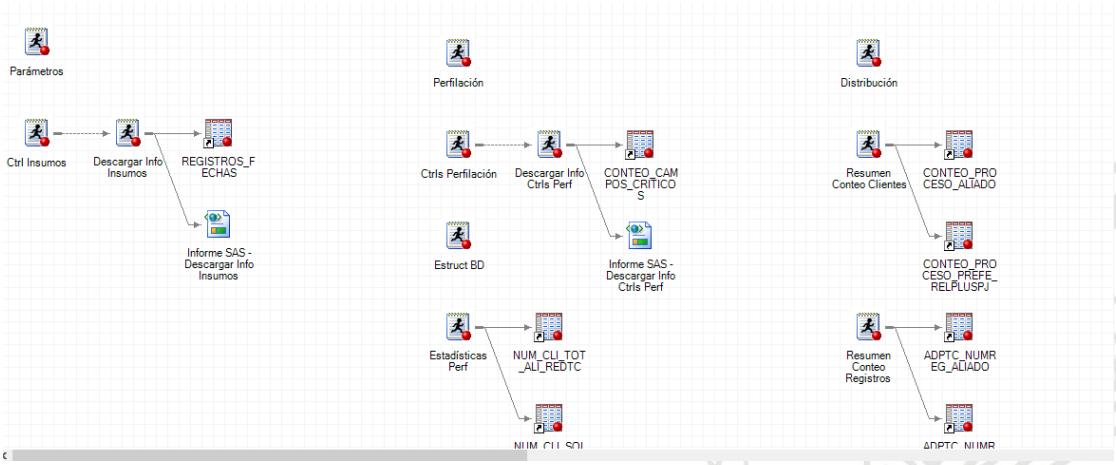


Figura 10. Estructura de nueva versión en software SAS.

Al ejecutar esta tercera versión se obtienen resultados similares a los obtenidos por el procesamiento en la versión de Python, la siguiente tabla, donde se tiene una comparación entre ambas versiones, lleva a concluir que el gran problema inicial fue el tipo de procesamiento que se utiliza, donde se comienza con un procesamiento local y se lleva a mejoras haciendo uso de los conceptos y desarrollos conocidos como Hadoop y clúster de computadoras, los cuales están intrínsecos en el ambiente de almacenamiento del banco generalmente llamado como Landing Zone.

Tabla 4. Comparación de desempeño.

Segmento	**Tiempos en Python (Segundos)**	**Tiempos en SAS (Segundos)**
Ctrl_Insumos	16.67	17
Perfilación	185.13	197
Crtl_Campos_Perf	20.18	21
Estructuras_BD	101.19	71
Estadis_Perf	37.6	11
Distribución	58.6	74
Resumen_Aliado	20.3	20
Generar_Layouts	190	296
Cascadas	39.5	10

A pesar del buen resultado entregado por SAS, se continua el desarrollo en Python ya que permite realizar diferentes acciones de una forma más ágil, permitiendo automatizar completamente el proceso. Adicionalmente el banco generalmente realiza mantenimientos constantes en las conexiones entre la LZ y SAS, generando indisponibilidad de este software constantemente.

Finalmente, para cerrar la etapa de ejecución, se realiza una prueba final con la versión del proyecto desarrollada en Python. La prueba consistió en ejecutar el código en dos fechas y momentos diferentes. La primera fecha (15/12/2020 15:20) corresponde a un martes de quincena para el mes de diciembre, fecha y hora donde se ejecutan la gran mayoría de procesos dentro de la sección de servicios de clientes. La segunda fecha (13/12/2020 10:00) corresponde a domingo de diciembre donde generalmente ningún proceso se ejecuta dentro de la misma.

Tabla 5. Ejecución en tiempos diferentes.

Segmento	**15/12/2020 15:20**	**13/12/2020 10:00**
Ctrl_Insumos	16.67	15.21
Perfilación	185.13	140.51
Crtl_Campos_Perf	20.18	20
Estructuras_BD	101.19	96.34
Estadis_Perf	37.6	35.11
Distribución	58.6	58
Resumen_Aliado	20.3	19.4
Generar_Layouts	190	160.5
Cascadas	39.5	15

Se obtiene un resultado que muestra la ejecución de la ideología Hadoop, la mejoría de tiempo cuando se tienen todos los cluster de computadoras trabajando en un mismo proceso, dando la posibilidad de que trabajen para un mismo nodo y las consultas se ejecuten con mayor velocidad. Claramente cuando los cluster están repartidos para múltiples tareas, las consultas tardan mayor tiempo para su ejecución.

- **Análisis de resultados**

Como tercera etapa en el proceso, se tiene el análisis de resultados, momento donde el colaborador debe confirmar la veracidad y confiablidad de los resultados. Para llevar a cabo esta tarea en la primera versión, el colaborador debía realizar diferentes opciones de Microsoft Excel, como lo son los filtros, el conteo de registros, la agrupación por categorías, entre otros. Esta tarea le tomaba un promedio de 90 minutos para analizar las 8 bases resultantes del proceso.

Para la versión de Python, se mitiga este tiempo de análisis al agregarle al procesamiento dos detalles principales. El primero es la construcción de controles y estadísticas que se ven reflejados en consultas SQL que s ejecutan en grupos como "Ctrl_Insumos", "Crtl_Campos_Perf" y "Estadis_Perf", las cuales generan tablas que finalmente se consolidan en un formato Excel que es exportado para el análisis del colaborador. La estructura de este formato puede evidenciarse en la figura 11.

PROCESO	CAMPO	CATALOGO	NUM REGISTROS
20200819	asignado_a	GMV	251959
20200819	asignado_a	PREFE	73504
20200819	asignado_a	PJ	12561
20200819	asignado_a	REL PLUS	10813
20200819	asignado_a	ALIADO	126553
20200819	asignado_a	COMUNES	29170
20200903	asignado_a	ALIADO	123842
20200903	asignado_a	PJ	6100
20200903	asignado_a	COMUNES	26137
20200903	asignado_a	GMV	244945
20200903	asignado_a	REL PLUS	10847
20200903	asignado_a	PREFE	72982
20200916	asignado_a	ALIADO	140840
20200916	asignado_a	PJ	6948
20200916	asignado_a	GMV	275971
20200916	asignado_a	PREFE	80012
20200916	asignado_a	REL PLUS	12030
20200916	asignado_a	COMUNES	31138
20201005	asignado_a	REL PLUS	11066
20201005	asignado_a	PREFE	77610

Suma de NUM REGISTROS	Etiquetas de columna			
I	1,692	1,317	2,502	2,139
(en blanco)			1	
cupo_global	**584,696**	**543,206**	**938,899**	**857,816**
LLENO	584,696	543,206	938,899	857,816
Debito	**584,696**	**543,206**	**938,899**	**857,816**
N	274,241	250,246	458,493	399,962
Tipo_ID_amparado	**584,696**	**543,206**	**938,899**	**857,816**
1	10,268	9,055	12,002	10,352
2	247	231	300	247
3	1,649	1,497	1,900	1,384
4	14	9	11	12
5	17	18	20	20

Figura 11. Ejemplo de estadísticas de salida.

El segundo se refiere a las gráficas entregadas por el proceso, con el fin de que el colaborador pueda notar con mayor facilidad cuando exista una inconsistencia o error en los datos, este tipo de graficas son presentadas en las figuras 5 y 12.

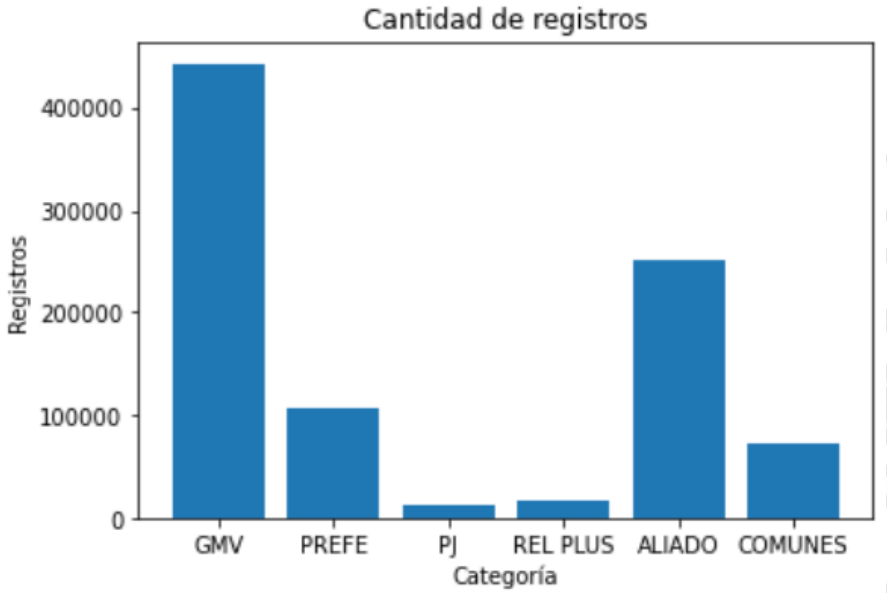


Figura 12. Ejemplo de grafico de estadísticas.

Conclusiones

Se logra aumentar la eficiencia del proceso Rediferidos tarjetas de crédito en un 65% respecto a tiempos de ejecución, gracias al uso apropiado del motor de búsquedas de Cloudera Impala y la arquitectura Hadoop, eliminando así el problema inicial del procesamiento local.

Se automatiza el proceso mediante el lenguaje de programación Python, incluyendo funciones adicionales para graficar, calcular estadísticas y compartir resultados, permitiendo al Colaborador una fácil y rápida ejecución, contando con herramientas para analizar y confirmar resultados.

Al unir el software SAS con el procesamiento en Cloudera Impala, se obtienen resultados similares respecto a los tiempos de ejecución obtenidos con el código en Python. Sin embargo, este último lenguaje da la posibilidad de desarrollar funciones adicionales que contribuyen a la automatización y el aumento de eficiencia.

Referencias Bibliográficas

[1] Acerca de nosotros. {En línea]. {02 de diciembre de 2020}. Disponible en: https://www.grupobancolombia.com/wps/portal/acerca-de

[2] Corral, J. Introducción a la estadística experimental. {En línea}. {02 de diciembre de 2020}. Disponible en: https://dicaf.ujed.mx/archivos/20170703114858IntroduccionEstadisticaExperimental.pdf

[3] Hadoop, ¿Qué es por qué es importante?. {En línea}. {02 de diciembre de 2020}. Disponible en: https://www.sas.com/es_co/insights/big-data/hadoop.html

[4] Russell John. Cloudera Impala. California: O'REILLY, 2013.

[5] ¿Qué es Python?. {En línea}. {02 de diciembre de 2020}. Disponible en: https://www.programoergosum.com/cursos-online/raspberry-pi/244-iniciacion-a-python-en-raspberry-pi/que-es-python

[6] Python SQL Driver - pyodbc. {En línea}. {02 de diciembre de 2020}. Disponible en: https://docs.microsoft.com/en-us/sql/connect/python/pyodbc/python-sql-driver-pyodbc?view=sql-server-ver15

[7] What is Spark?. {En línea}. {02 de diciembre de 2020}. Disponible en: https://chartio.com/learn/data-analytics/what-is-spark/

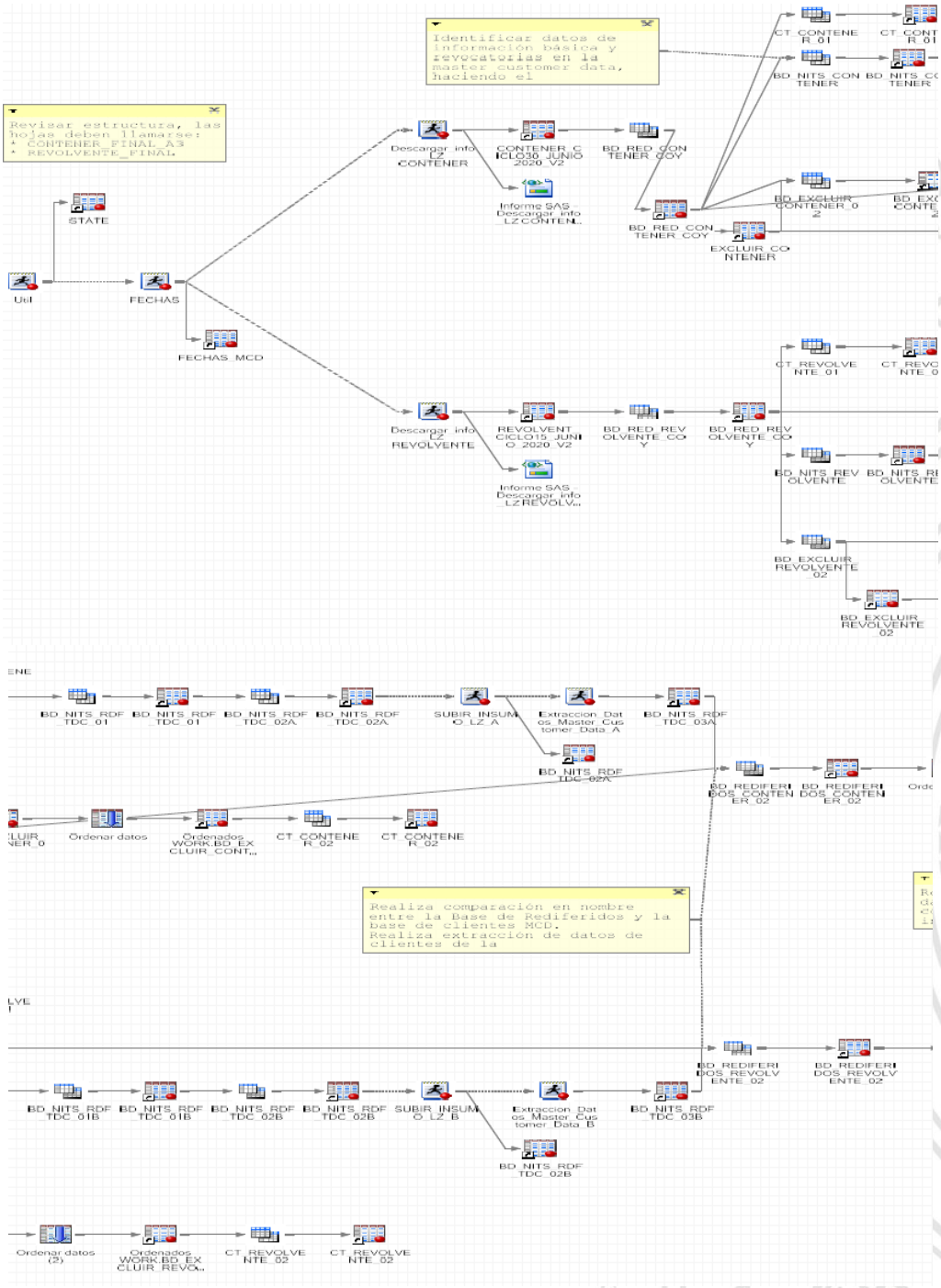
[8] Pandas, Package overview. {En línea}. {02 de diciembre de 2020}. Disponible en: https://pandas.pydata.org/pandas-docs/stable/getting_started/overview.html

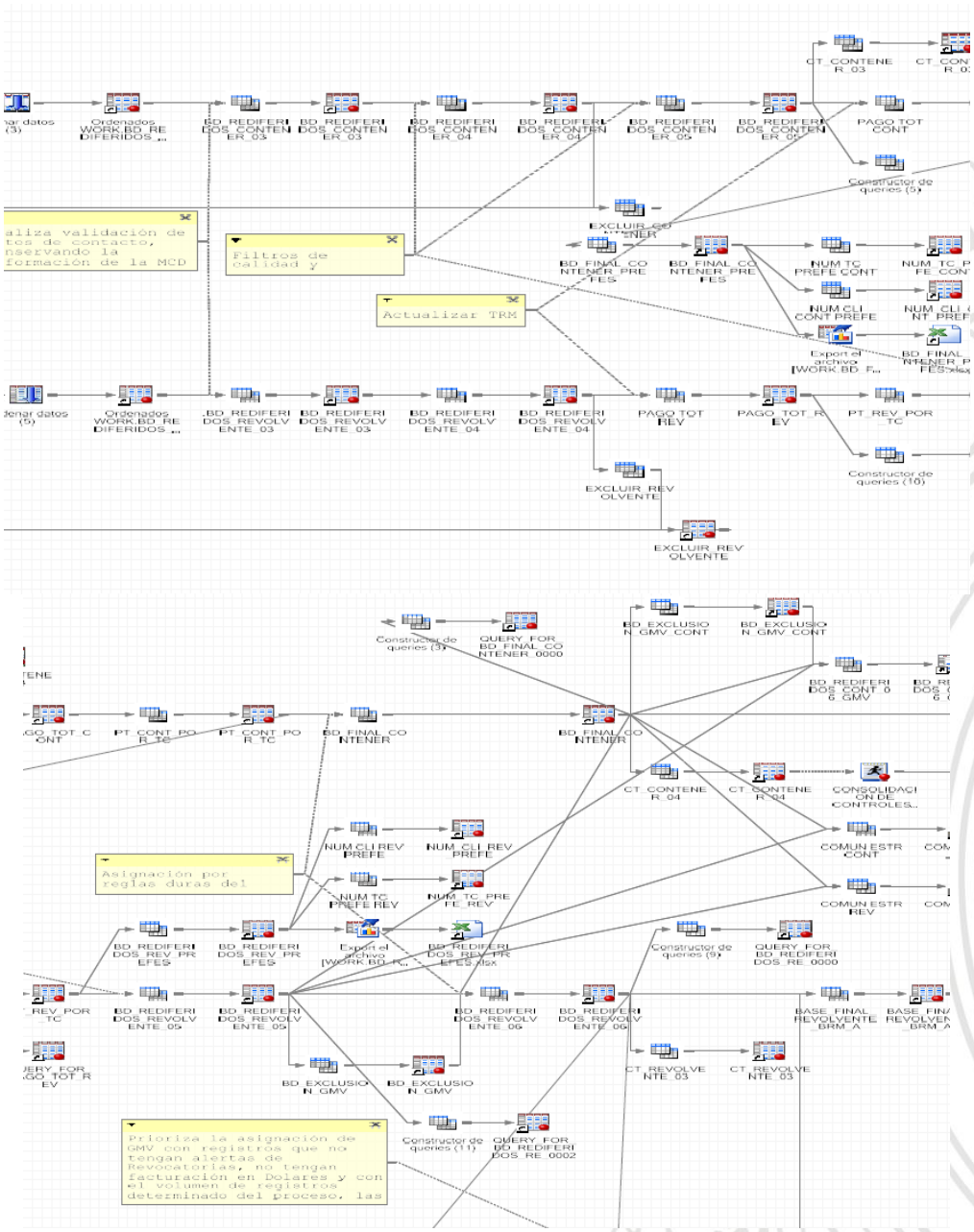
[9] About SAS. {En línea}. {02 de diciembre de 2020}. Disponible en: https://www.sas.com/en_us/company-information.html#history

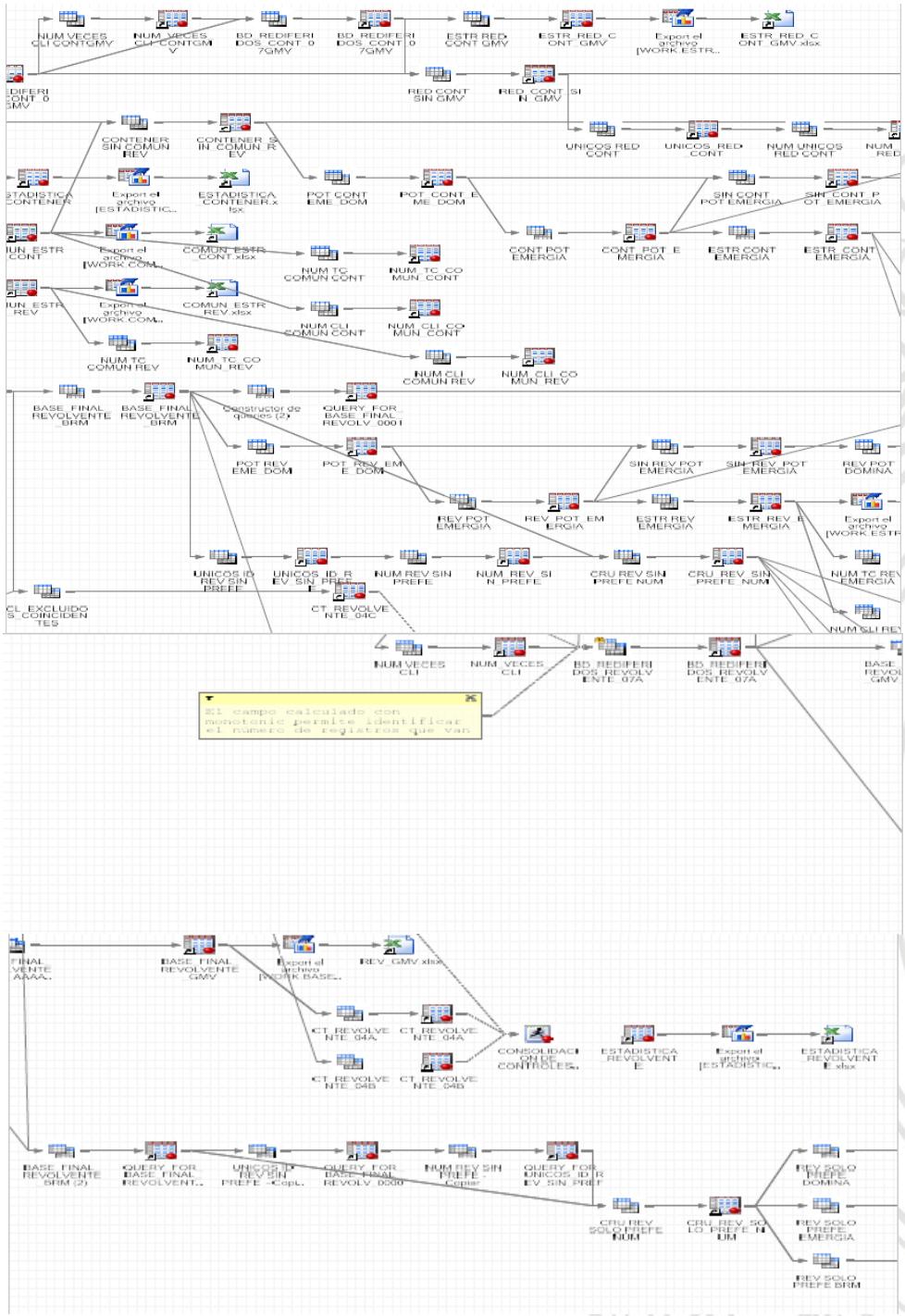
[10] Landing zone. {En línea}. {02 de diciembre de 2020}. [Documento reservado para personal Grupo Bancolombia]. Recuperado de: https://bancolombia.sharepoint.com/:w:/r/teams/ComunidadAnalticaBAM2/_layouts/15/Doc.aspx?sourcedoc=%7B8CDB2DFB-B322-4812-8E4C-58C148D220B9%7D&file=Modelo%20de%20Gobierno%20Landing%20Zone%20y%20SAS.docx&action=default&mobileredirect=true&DefaultItemOpen=1

Anexos

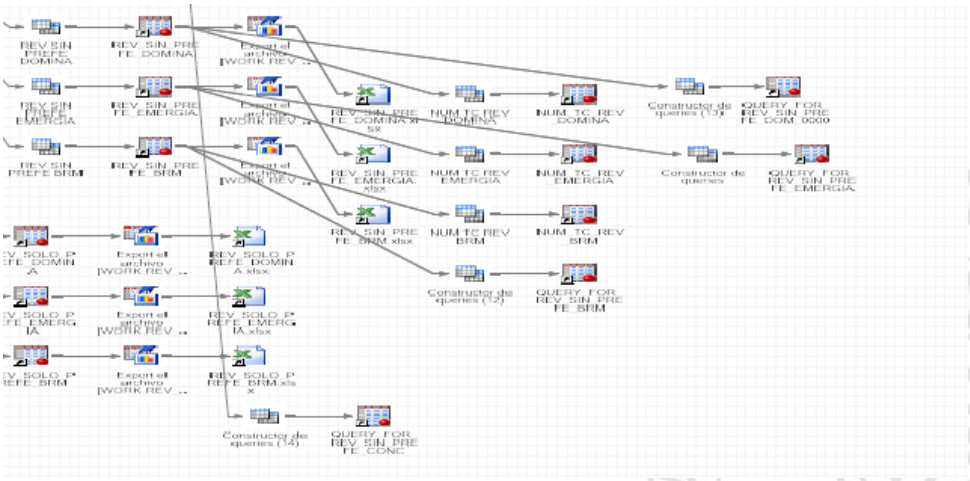
[1] Proceso Rediferidos tarjetas de crédito.


Identificar datos de información básica y reevaluarlas en la master customer data, haciendo el
Revisar estructura, las hojas deben llamarse:
* CONTENER_FINAL_A3
* REVOLVENTE_FINAL
STATE
Usl
FECHAS
FECHAS_MCD
Descargar info LZ CONTENER
CONTENER_CICLO36_JUNIO_2020_V2
Informe SAS Descargar info LZ CONTEN...
BD_RED_CONTENER_COY
BD_RED_CONTENER_COY
EXCLUIR_CONTENER
CT_CONTENER_01
BD_NITS_CONTENER
BD_EXCLUIR_CONTENER_02
Descargar info LZ REVOLVENTE
REVOLVENTE_CICLO15_JUNIO_2020_V2
Informe SAS Descargar info LZ REVOLV...
BD_RED_REVOLVENTE_COY
BD_RED_REVOLVENTE_COY
CT_REVOLVENTE_01
BD_NITS_REVOLVENTE
BD_EXCLUIR_REVOLVENTE_02
BD_NITS_RDF_TDC_01
BD_NITS_RDF_TDC_01
BD_NITS_RDF_TDC_02A
BD_NITS_RDF_TDC_02A
SUBIR_INSUMO_LZ_A
Extraccion_Datos_Master_Customer_Data_A
BD_NITS_RDF_TDC_03A
BD_NITS_RDF_TDC_02A
BD_REDIFERIDOS_CONTENER_02
BD_REDIFERIDOS_CONTENER_02
Ordenar datos
Ordenados WORK.BD_EXCLUIR_CONT...
CT_CONTENER_02
CT_CONTENER_02
Realiza comparación en nombre entre la Base de Rediferidos y la base de clientes MCD. Realiza extraccion de datos de clientes de la
BD_REDIFERIDOS_REVOLVENTE_02
BD_REDIFERIDOS_REVOLVENTE_02
BD_NITS_RDF_TDC_01B
BD_NITS_RDF_TDC_01B
BD_NITS_RDF_TDC_02B
BD_NITS_RDF_TDC_02B
SUBIR_INSUMO_LZ_B
Extraccion_Datos_Master_Customer_Data_B
BD_NITS_RDF_TDC_03B
BD_NITS_RDF_TDC_02B
Ordenar datos (2)
Ordenados WORK.BD_EXCLUIR_REVO...
CT_REVOLVENTE_02
CT_REVOLVENTE_02


Ordenados WORK.BD_RE DIFERIDOS_...
BD_REDIFERI DOS_CONTEN ER_03
BD_REDIFERI DOS_CONTEN ER_03
BD_REDIFERI DOS_CONTEN ER_04
BD_REDIFERI DOS_CONTEN ER_04
BD_REDIFERI DOS_CONTEN ER_05
BD_REDIFERI DOS_CONTEN ER_05
CT_CONTENE R_03
PAGO_TOT CONT
Constructor de queries (5)
EXCLUIR_CO NTENER
BD_FINAL_CO NTENER_PRE FES
BD_FINAL_CO NTENER_PRE FES
NUM_TC PREFE_CONT
NUM_CLI CONT_PREFE
Export el archivo [WORK.BD_F...
Filtros de calidad y
Actualizar TRM
Ordenados WORK.BD_RE DIFERIDOS_...
BD_REDIFERI DOS_REVOLV ENTE_03
BD_REDIFERI DOS_REVOLV ENTE_03
BD_REDIFERI DOS_REVOLV ENTE_04
BD_REDIFERI DOS_REVOLV ENTE_04
PAGO_TOT REV
PAGO_TOT_R EV
PT_REV_POR TC
Constructor de queries (10)
EXCLUIR_REV OLVENTE
EXCLUIR_REV OLVENTE
Constructor de queries (3)
QUERY_FOR_ BD_FINAL_CO NTENER_0000
BD_EXCLUSIO N_GMV_CONT
BD_EXCLUSIO N_GMV_CONT
BD_REDIFERI DOS_CONT_0 6_GMV
PT_CONT_PO R_TC
PT_CONT_PO R_TC
BD_FINAL_CO NTENER
BD_FINAL_CO NTENER
CT_CONTENE R_04
CT_CONTENE R_04
CONSOLIDACI ON_DE_ CONTROLES...
NUM_CLI_REV PREFE
NUM_CLI_REV PREFE
COMUN_ESTR _CONT
Asignación por reglas duras del
NUM_TC PREFE_REV
NUM_TC_PRE FE_REV
COMUN_ESTR _REV
BD_REDIFERI DOS_REV_PR EFES
BD_REDIFERI DOS_REV_PR EFES
Export el archivo [WORK.BD_R...
BD_REDIFERI DOS_REV_PR EFES.xlsx
Constructor de queries (9)
QUERY_FOR_ BD_REDIFERI DOS_RE_0000
BD_REDIFERI DOS_REVOLV ENTE_05
BD_REDIFERI DOS_REVOLV ENTE_05
BD_REDIFERI DOS_REVOLV ENTE_06
BD_REDIFERI DOS_REVOLV ENTE_06
BASE_FINAL_ REVOLVENTE _BRM_A
BD_EXCLUSIO N_GMV
BD_EXCLUSIO N_GMV
CT_REVOLVE NTE_03
CT_REVOLVE NTE_03
Priorizar la asignación de GMV con registros que no tengan alertas de Revocatorias, no tengan facturación en Dolares y con el volumen de registros determinado del proceso, las
Constructor de queries (11)
QUERY_FOR_ BD_REDIFERI DOS_RE_0002


NUM_VECES_CLI_CONTGMV
NUM_VECES_CLI_CONTGMV
BD_REDIFERIDOS_CONT_07GMV
BD_REDIFERIDOS_CONT_07GMV
ESTR_RED_CONT_GMV
ESTR_RED_CONT_GMV
Export el archivo [WORK.ESTR..
ESTR_RED_CONT_GMV.xlsx
RED_CONT_SIN_GMV
RED_CONT_SIN_GMV
CONTENER_SIN_COMUN_REV
CONTENER_SIN_COMUN_REV
UNICOS_RED_CONT
UNICOS_RED_CONT
NUM_UNICOS_RED_CONT
Export el archivo [ESTADISTIC..
ESTADISTICA_CONTENER.xlsx
POT_CONT_EME_DOM
POT_CONT_EME_DOM
SIN_CONT_POT_EMERGIA
SIN_CONT_POT_EMERGIA
Export el archivo [WORK.COM..
COMUN_ESTR_CONT.xlsx
NUM_TC_COMUN_CONT
NUM_TC_COMUN_CONT
CONT_POT_EMERGIA
CONT_POT_EMERGIA
ESTR_CONT_EMERGIA
ESTR_CONT_EMERGIA
Export el archivo [WORK.COM..
COMUN_ESTR_REV.xlsx
NUM_CLI_COMUN_CONT
NUM_CLI_COMUN_CONT
NUM_TC_COMUN_REV
NUM_TC_COMUN_REV
NUM_CLI_COMUN_REV
NUM_CLI_COMUN_REV
BASE_FINAL_REVOLVENTE_BRM
BASE_FINAL_REVOLVENTE_BRM
Constructor de gráficos (2)
QUERY_FOR_BASE_FINAL_REVOLV_0001
POT_REV_EME_DOM
POT_REV_EME_DOM
SIN_REV_POT_EMERGIA
SIN_REV_POT_EMERGIA
REV_POT_DOMINA
REV_POT_EMERGIA
REV_POT_EMERGIA
ESTR_REV_EMERGIA
ESTR_REV_EMERGIA
Export el archivo [WORK.ESTR..
UNICOS_ID_REV_SIN_PREFE
UNICOS_ID_REV_SIN_PREFE
NUM_REV_SIN_PREFE
NUM_REV_SIN_PREFE
CRU_REV_SIN_PREFE_NUM
CRU_REV_SIN_PREFE_NUM
NUM_TC_REV_EMERGIA
NUM_CLI_REV_EMERGIA
CL_EXCLUIDOS_COINCIDENTES
CT_REVOLVENTE_04C
NUM_VECES_CLI
NUM_VECES_CLI
BD_REDIFERIDOS_REVOLVENTE_07A
BD_REDIFERIDOS_REVOLVENTE_07A
El campo calculado con monotonic permite identificar el número de registros que van

FINAL_VENTE_AAAA..
BASE_FINAL_REVOLVENTE_GMV
Export el archivo [WORK.BASE..
REV_GMV.xlsx
CT_REVOLVENTE_04A
CT_REVOLVENTE_04A
CT_REVOLVENTE_04B
CT_REVOLVENTE_04B
CONSOLIDACION DE CONTROLES..
ESTADISTICA_REVOLVENTE
Export el archivo [ESTADISTIC..
ESTADISTICA_REVOLVENTE.xlsx
BASE_FINAL_REVOLVENTE_BRM (2)
QUERY_FOR_BASE_FINAL_REVOLVENT..
UNICOS_ID_REV_SIN_PREFE - Copi..
QUERY_FOR_BASE_FINAL_REVOLV_0030
NUM_REV_SIN_PREFE - Copier
QUERY_FOR_UNICOS_ID_REV_SIN_PREF
CRU_REV_SOLO_PREFE_NUM
CRU_REV_SOLO_PREFE_NUM
REV_SOLO_PREFE_DOMINA
REV_SOLO_PREFE_EMERGIA
REV_SOLO_PREFE_BRM


REV SIN PREFE DOMINA
REV_SIN_PREFE_DOMINA
Export el archivo [WORK REV ...
REV SIN PREFE EMERGIA
REV_SIN_PREFE_EMERGIA
Export el archivo [WORK REV ...
REV SIN PREFE BRM
REV_SIN_PREFE_BRM
Export el archivo [WORK REV ...
REV_SIN_PREFE_DOMINA.xlsx
NUM TC REV DOMINA
NUM_TC_REV DOMINA
Constructor de queries (13)
QUERY FOR REV_SIN_PREFE_DOM_0000
REV_SIN_PREFE_EMERGIA.xlsx
NUM TC REV EMERGIA
NUM_TC_REV EMERGIA
Constructor de queries
QUERY FOR REV_SIN_PREFE_EMERGIA
REV_SIN_PREFE_BRM.xlsx
NUM TC REV BRM
NUM_TC_REV BRM
Constructor de queries (12)
QUERY FOR REV_SIN_PREFE_BRM
V_SOLO_PREFE_DOMINA
Export el archivo [WORK REV ...
REV_SOLO_PREFE_DOMINA.xlsx
V_SOLO_PREFE_EMERGIA
Export el archivo [WORK REV ...
REV_SOLO_PREFE_EMERGIA.xlsx
V_SOLO_PREFE_BRM
Export el archivo [WORK REV ...
REV_SOLO_PREFE_BRM.xlsx
Constructor de queries (14)
QUERY FOR REV_SIN_PREFE_CONC