Automatización de Informes en Tigo

Santiago Molina Echeverri

Informe final semestre de industria para optar por el título de Ingeniero de sistemas

Asesor
Gina Paola Maestre Góngora
Doctora en Ingeniería de Sistemas y Computación

Universidad de Antioquia
Facultad de Ingeniería
Ingeniería de Sistemas
Medellín
2024

Cita	Molina Echeverri Santiago [1]
Referencia Estilo IEEE (2020)	[1] S. Molina Echeverri, “Automatización de Informes en Tigo”, Pregrado, Ingeniería de Sistemas, Universidad de Antioquia, Medellín, 2024.

Dedicatoria

A mi familia por su apoyo.

Agradecimientos

A mis tutores por su entrega.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

SIGLAS, ACRÓNIMOS Y ABREVIATURAS

IVR	Interactive Voice Response (Respuesta de voz interactiva**)**
ETL	Extract, Transform and Load (Extraer, transformar y cargar)
SSIS	SQL Server Integration Services
B2B	Business-to-business

RESUMEN

Este informe presenta el desarrollo de dos automatizaciones implementadas para la empresa Tigo. La primera automatización se centra en la creación de un informe en Excel, donde el contenido se modifica automáticamente con los resultados de diferentes consultas SQL. La segunda se dedica a la extracción, limpieza y carga de datos en una tabla SQL, la cual contiene información relacionada con las llamadas al IVR.

El desarrollo de estas automatizaciones se llevó a cabo utilizando la metodología Scrum, a lo largo de cuatro Sprint mediante el uso de herramientas como Python y SSIS. La aplicación de esta metodología permitió una gestión eficiente del proyecto, asegurando iteraciones continuas y mejoras progresivas.

Como resultado, las automatizaciones implementadas han demostrado una mejora significativa en la operación del equipo, proporcionando acceso eficiente a los datos. Esto ha permitido optimizar los procesos internos y mejorar la toma de decisiones basada en datos actualizados.

***Palabras clave* — Automatizaciones, SQL, Datos**

ABSTRACT

This report presents the implementation of two automations implemented for the company Tigo. The first automation focuses on the creation of a report in Excel, where the content is automatically modified with the results of different SQL queries. The second one is dedicated to the extraction, cleaning and loading of data into a SQL table, which contains information related to IVR calls.

The development of these automations was carried out using the Scrum methodology, over four sprints, using tools such as Python and SSIS. The application of this methodology allowed for efficient project management, ensuring continuous iterations and progressive improvements.

As a result, the implemented automations have shown a significant improvement in the team's operation, providing efficient access to data. This has allowed optimizing internal processes and improving decision making based on updated data.

Keywords **— Automations, SQL, Data**

I. INTRODUCCIÓN

En el escenario actual del sector de las telecomunicaciones, la constante evolución tecnológica demanda soluciones innovadoras y eficientes para optimizar los procesos empresariales. Tigo, uno de los líderes en el sector, se enfrenta a la necesidad de mejorar la eficiencia operativa y la gestión de datos. En este sentido, se ha identificado la oportunidad de optimizar dos áreas específicas que son fundamentales para el rendimiento integral de la organización. En este contexto, se tiene como objetivo abordar dos desafíos clave en el entorno de una empresa de telecomunicaciones.

En primer lugar, se encuentra la problemática asociada a la generación de informes a partir de consultas SQL y documentos de Excel. La elaboración manual de estos reportes implica la inversión de recursos tanto humanos como tecnológicos traducidos en tiempo y esfuerzo, lo cual puede derivar en errores. En segundo lugar, existe la necesidad de obtener información acerca de llamadas que se hacen al IVR (Interactive Voice Response) para el posterior análisis de los datos por las personas encargadas del área.

Es por esto por lo que, esta práctica tiene como objetivo abordar estos desafíos mediante dos tareas específicas. La primera orientada a la implementación de una automatización que permita la generación eficiente de informes mediante consultas SQL y la gestión de documentos en Excel. La segunda se enfoca en la automatización de las consultas mediante las cuales se obtienen los diferentes campos de información relacionados con las llamadas hacia el IVR, para posteriormente exponer esta información en una tabla de la base de datos usada por el equipo de trabajo.

Lo anterior se realizará mediante un enfoque colaborativo entre el estudiante y el equipo de profesionales de la empresa, donde se busca fomentar el aprendizaje práctico y la aplicación de conocimientos teóricos, pudiendo contribuir significativamente a la resolución de problemas reales basado en metodologías ágiles.

II. OBJETIVOS

A. Objetivo general

Implementar una automatización utilizando Python para la generación eficiente de informes en la empresa Tigo, basado en consultas SQL y la manipulación de datos en documentos de Excel, con el propósito de mejorar la eficiencia operativa y reducir los errores asociados a los procesos manuales; y generar de forma automática la obtención de datos relacionados con las llamadas hacía el IVR y su exposición en la base de datos.

B. Objetivos específicos

- Desarrollar scripts en Python que ejecuten las consultas SQL de manera automatizada, asegurando la correcta extracción y procesamiento de los datos requeridos para los informes.
- Analizar la eficiencia operativa antes y después de la implementación de la automatización, cuantificando el tiempo ahorrado, asegurando la precisión de los informes generados y comparando los resultados con los obtenidos a través de métodos manuales.
- Generar las consultas de datos SQL relacionadas con las llamadas hacia el IVR, como también la creación de las tablas y diferentes campos que sean necesarios para el posterior análisis de las personas que consumen dicha información.
- Automatizar el proceso de ejecución mediante el uso de una herramienta de ETL, donde se obtenga la información base del IVR, se ejecuten las consultas SQL y se realicen las modificaciones necesarias a la tabla final.

III. MARCO TEÓRICO

Automatización de procesos:

La automatización de procesos mediante la programación en Python se ha convertido en una práctica común para mejorar la eficiencia en la manipulación de datos y la generación de informes. Autores como McKinney [1]. destacan la versatilidad de Python en el ámbito empresarial, especialmente para tareas relacionadas con la manipulación de datos y la generación de informes automatizados Además de esto, según Kimball y Ross [2]., la generación automatizada de informes permite a las organizaciones agilizar el acceso a la información, reducir errores humanos y mejorar la toma de decisiones basada en datos

El uso de consultas SQL para extraer datos de bases de datos y la manipulación de documentos en Excel pueden integrarse eficientemente en un flujo de trabajo automatizado con Python, aprovechando bibliotecas como Pandas y openpyxl. Esta integración ofrece una solución robusta para mejorar la eficiencia en la generación de informes y reducir la propensión a errores asociada con los métodos manuales.

En este sentido, la automatización de consultas SQL se fundamenta en el uso de lenguajes de consulta estructurados (SQL) para recuperar y manipular datos almacenados en bases de datos relacionales. Date destaca la importancia de comprender los principios de diseño de bases de datos relacionales y la sintaxis de SQL para desarrollar consultas eficientes y optimizar el rendimiento del sistema [3].

IVR - Interactive Voice Response

El Sistema de Respuesta de Voz Interactiva, conocido como IVR (Interactive Voice Response), es una tecnología que permite a los usuarios interactuar con sistemas informáticos mediante el uso de entradas de voz o tonos DTMF (Dual-Tone

Multi-Frequency) generados desde un teclado telefónico. Según C. Bhat y B. S. Mithun [4]., "la tecnología de respuesta de voz interactiva (IVR) hace que el servicio al cliente esté disponible las 24 horas, los 7 días de la semana y sea rentable y ha sido utilizada por la mayoría de las empresas orientadas al cliente." Es por esto por lo que estos sistemas son esenciales en centros de atención telefónica para manejar grandes volúmenes de llamadas, lo que mejora la eficiencia y reduce los costos operativos.

Los sistemas IVR son ampliamente utilizados en diversas industrias, como telecomunicaciones, servicios financieros y atención médica, debido a su capacidad para manejar múltiples tipos de interacciones de manera efectiva. Por lo que autores como J. Anton y N. L. Petouhoff [5] destacan que el uso de este tipo de sistemas trae consigo diferentes beneficios como lo son la automatización de servicios, la reducción de costos operativos, la mejora en la experiencia del cliente, la obtención de datos para la continua retroalimentación y la facilitación de la autoayuda, entre otros.

ETL usando SSIS

El proceso de ETL, que significa Extracción, Transformación y Carga, es fundamental en la integración de datos y la creación de almacenes de datos. ETL implica tres fases principales: la extracción de datos de diversas fuentes, la transformación de estos datos en un formato adecuado para análisis y la carga en un sistema de destino, como un almacén de datos. SQL Server Integration Services (SSIS) es una herramienta poderosa de Microsoft utilizada para realizar tareas de ETL de manera eficiente.

Según T. Wu, "la utilización de SSIS para realizar la función ETL puede reducir en gran medida el costo de desarrollo de aplicaciones específicas de algunas empresas y mejorar la eficiencia del trabajo" [6]. SSIS se destaca por su capacidad para manejar grandes volúmenes de datos y su integración con otros productos de Microsoft SQL Server.

IV. METODOLOGÍA

A. Metodología:

a. Desarrollo Ágil con Enfoque en Ingeniería de Datos.

Según Schwaber y Sutherland, Scrum es un marco de trabajo ágil para el desarrollo de software que se centra en la entrega iterativa e incremental de productos (Figura 1). Los componentes principales de Scrum incluyen roles, eventos y artefactos que permiten a los equipos trabajar de manera colaborativa y flexible para alcanzar objetivos comunes y adaptarse a los cambios [7].

Roles: Product Owner: Representa las necesidades y expectativas del cliente., Scrum Master: Facilita el proceso y elimina obstáculos para el equipo., Equipo de Desarrollo: Realiza el trabajo para entregar el producto.

b. Eventos:

Sprint: Periodo de tiempo fijo (usualmente 2-4 semanas) durante el cual se entrega un incremento de producto potencialmente entregable.
Reunión de Planificación del Sprint: Define el trabajo a realizar durante el Sprint.
Revisión del Sprint: Demostración del trabajo completado al final del Sprint.
Retrospectiva del Sprint: Reflexión sobre el Sprint y mejoras para el siguiente.

c. Artefactos:

Product Backlog: Lista priorizada de características y requisitos del producto.
Sprint Backlog: Tareas seleccionadas para el Sprint a partir del Product Backlog.
Incremento: Versión mejorada y funcional del producto al final de cada Sprint.

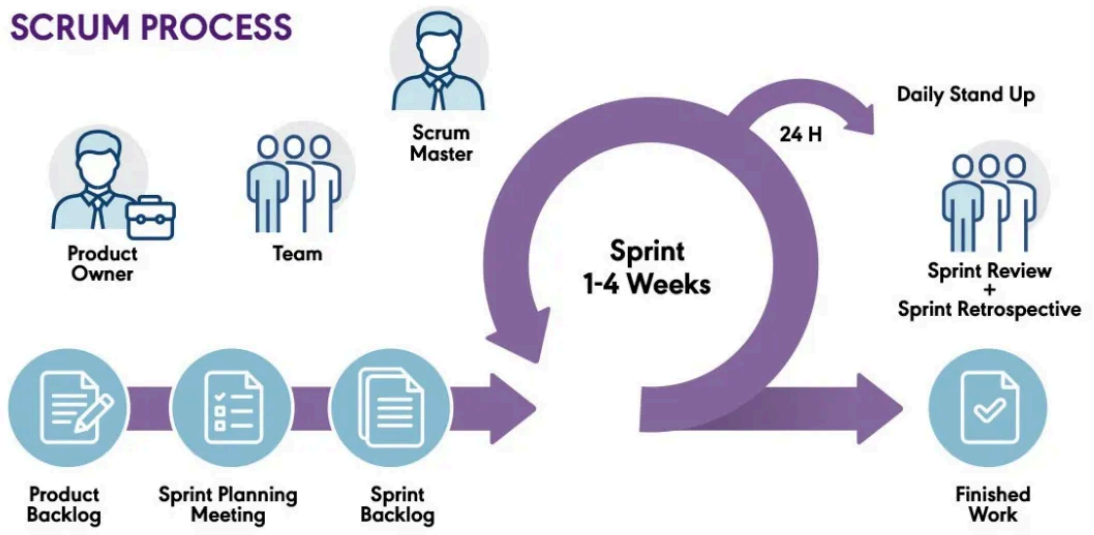

Figura 1: Proceso SCRUM

Fuente: PM Partners. (s.f.). The Agile Journey - A Scrum Overview. Recuperado de https://www.pm-partners.com.au/insights/the-agile-journey-a-scrum-overview/

La metodología se centrará en la eficiencia, calidad, procesamiento y análisis cuantitativo de datos, con un énfasis especial en la generación automatizada de informes y la manipulación de datos en Python; este enfoque permitirá medir la eficiencia operativa y los resultados numéricos.

B. Fases del Proyecto:

a. Planificación y Análisis:

- Identificación de requisitos específicos de ingeniería de datos para la generación automatizada de informes.
- Análisis detallado de la infraestructura de datos actual y las fuentes de información.
- Establecimiento de metas y objetivos específicos relacionados con la eficiencia y calidad del procesamiento de datos.

b. Iteración de desarrollo:

- Sprint 1:
  - Generación de datos mediante consultas SQL
- Sprint 2:
  - Generación de informes en Excel usando implementación en Python

- Evaluación de la eficiencia operativa y precisión de los informes generados.
- Elaboración de documentación detallada de la arquitectura de ingeniería de datos implementada.

- Sprint 3:
  - Creación de la ETL para la obtención de la información base de las llamadas al IVR
  - Creación de tablas para almacenar la información del IVR.
  - Creación de los campos requeridos por el equipo para el posterior análisis.
  - Generación de consultas SQL para la obtención de los datos.
- Sprint 4:
  - Generación de scripts en Python para la inserción de los datos en la tabla.
  - Procesamiento de datos de los meses pasados según la necesidad del equipo.
  - Añadir proceso de Python a la ETL en SSIS para la ejecución automática del proceso según la periodicidad requerida.

c. Informe Final

- Preparación de una presentación que destaque los logros, aprendizajes y mejoras obtenidas durante el proyecto.

C. Proceso de trabajo durante el sprint:

Se realizaron diariamente reuniones de entre 10 a 15 minutos con el scrum master, estas reuniones se llevaron a cabo siguiendo las prácticas y principios de Scrum, con el propósito de mantener al equipo informado sobre el progreso, los desafíos y las próximas actividades relacionadas con las automatizaciones. En estas reuniones se mantuvieron presentes los siguientes puntos claves:

- Informe del Estado del Progreso: Donde se brinda una actualización sobre el trabajo realizado desde la última reunión diaria, destacando los logros, los impedimentos encontrados y los próximos pasos.
- Identificación de Obstáculos: Se identificaron los diferentes impedimentos que estuvieran afectando el progreso del proyecto, con el fin de buscar soluciones y eliminar obstáculos.
- Planificación de Acciones: Se discuten las acciones necesarias para abordar los obstáculos identificados.

- Cierre: El Scrum Master resume las acciones acordadas

Además de estas reuniones diarias se programan adicionales dependiendo de la necesidad, para definir y aclarar diferentes aspectos de las consultas en la base de datos. También cuando es necesario se realizan actualizaciones del backlog de producto y sprint.

D. Técnicas e Instrumentos:

a. *Tecnologías usadas:*

- Para la implementación de ambos proyectos (automatización de informe de cierre y tabla de IVR) se usó Python para la realización de las consultas, la inserción en las tablas y el manejo del documento de Excel donde residirán los datos encontrados en el informe de cierre.
- Para la creación de la ETL se usa la herramienta Microsoft SQL Server Integration Services (SSIS)
- Para que las tareas se ejecuten de forma automática según la periodicidad requerida se utiliza el scheduler alojado dentro del servidor usado dentro de la compañía.

Configuración del entorno:

- Entorno virtual:

  Se debe crear el entorno virtual y para esto se utiliza en comando py -m venv venv en la terminal, para luego activarlo se utiliza el comando venv/scripts/activate-

- Librerías:

  Debido a que las dependencias necesarias se encuentran en el archivo requirements.txt, se utiliza el comando pip install -r .\requirements.txt y así se realizará la instalación de todas las librerías utilizadas dentro del proyecto

- Servidor:

Dentro del servidor se utiliza el Scheduler en donde se debe activar el entorno virtual, ingresar el comando Python main.py y desactivar el entorno virtual, ya que esto realizará todas las tareas automáticas necesarias para tanto la creación del reporte y la inserción de los datos en la tabla de IVR. Además de esto, se configura una temporalidad dentro del aplicativo para que se ejecuten estos pasos, en este caso de un mes para el reporte de cierre y un día para la tabla de IVR.

b. *Librerías y módulos:*

- openpyxl: Esta biblioteca permite la manipulación de archivos Excel en formato xlsx. Se utiliza para crear, leer y modificar archivos Excel desde Python. Se escogió debido a su amplia funcionalidad y su comunidad activa de usuarios y desarrolladores que proporcionan soporte y contribuciones constantes al proyecto.
- locale: se utiliza para manejar configuraciones regionales, como idioma, moneda, formato de fecha y hora, entre otros. En este caso, se utiliza para establecer la configuración regional del sistema, lo que puede ser útil para garantizar que los datos se muestren correctamente en el idioma y formato deseados en el archivo Excel generado.
- platform: proporciona información sobre la plataforma y el sistema operativo en el que se está ejecutando Python. Se utiliza en este contexto para determinar el sistema operativo del usuario y ajustar la configuración regional en consecuencia.
- pandas: útil para el análisis y manipulación de datos en Python. Se utiliza comúnmente para trabajar con datos tabulares, incluidos los resultados de las consultas SQL. Proporciona herramientas poderosas para filtrar, transformar y analizar datos de manera eficiente.
- pyodbc: proporciona una interfaz para conectarse a bases de datos utilizando el estándar ODBC (Open Database Connectivity). Se utiliza para ejecutar consultas SQL y recuperar datos de una base de datos relacional, como Microsoft SQL Server, MySQL, PostgreSQL, entre otros.
- configparser: La biblioteca configparser se utiliza para leer archivos de configuración en formato INI. Puede ser útil para almacenar configuraciones, como credenciales de base de datos o parámetros de conexión, de manera segura y organizada. En este caso se utiliza para la obtención de la información de acceso a la base de datos, además que proporciona una forma simple y conveniente de gestionar la configuración de la aplicación, lo que facilita la modularidad y la reutilización del código al separar la lógica de la configuración.
- os: Proporciona funciones para interactuar con el sistema operativo subyacente, como manipulación de archivos, acceso al entorno del sistema y manipulación de rutas de archivo. Se escoge ya que es estándar en Python y proporciona funcionalidades básicas pero esenciales para la manipulación de archivos y

directorios, lo que la hace útil en muchas situaciones, incluido el manejo de archivos de Excel generados en la automatización de consultas SQL.

- datetime: es una herramienta para trabajar con fechas y horas. Proporciona clases para manejar fechas, horas, deltas (diferencias entre fechas) y calendarios. Se elige debido a que esta librería permite realizar cálculos de tiempo (sumar o restar intervalos de tiempo a una fecha o hora) y permite formatear y analizar fechas y horas.
- encoders: Este módulo de la librería email proporciona funciones para codificar archivos adjuntos antes de enviarlos por correo electrónico, asegurando que los datos binarios se manejen correctamente.
- mime: Este módulo de la librería email se utiliza para crear objetos MIME (Multipurpose Internet Mail Extensions) que pueden ser utilizados como partes de un mensaje de correo electrónico (imágenes, texto, archivos adjuntos y contenido HTML)
- smtplib: Este módulo define una interfaz para enviar correos electrónicos a través del protocolo SMTP (Simple Mail Transfer Protocol). Es fundamental para enviar correos electrónicos desde Python.

V. RESULTADOS

Para resolver el objetivo de desarrollar scripts en Python que ejecuten las consultas SQL de manera automatizada, asegurando la correcta extracción y procesamiento de los datos requeridos para los informes se usa un sprint.

A. Sprint 1:

a. Elicitación de requisitos:

Para obtener los requisitos necesarios del proyecto se utilizaron los casos de uso donde se identificaron dos actores principales:

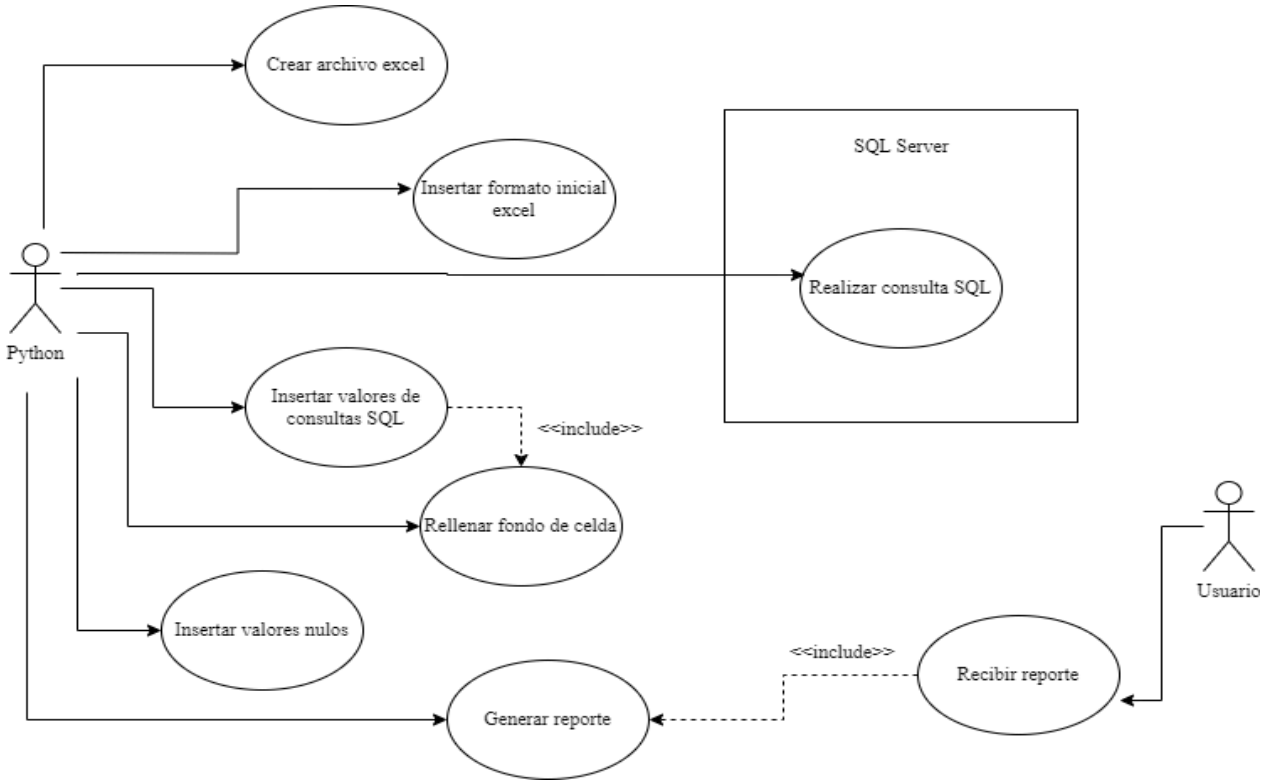

Figura 2: Diagrama caso de uso

b. Estructura Cierre Millicom:

El siguiente diagrama es una representación visual que permite modelar la estructura estática del sistema, mostrando las clases de objetos, sus atributos, métodos y las relaciones entre ellas. Su importancia radica en su capacidad para

proporcionar una comprensión clara y concisa de la arquitectura de un sistema, facilitando así su diseño, implementación y mantenimiento.

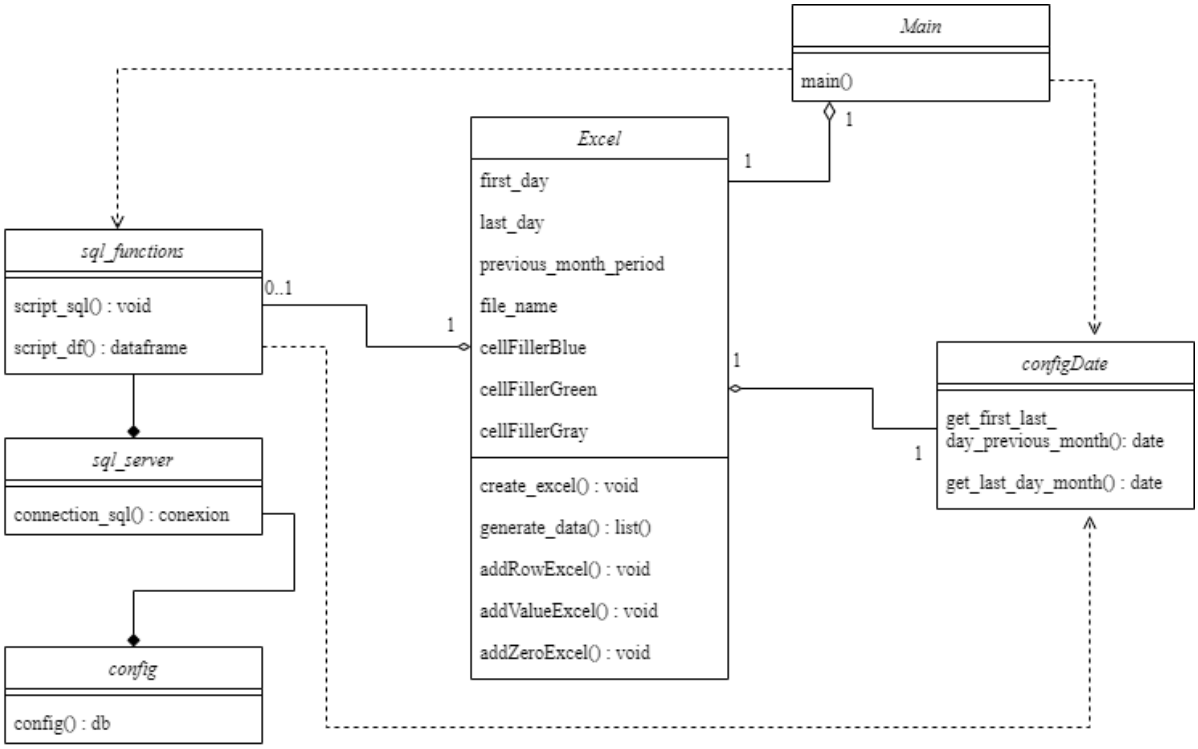

Figura 3: Diagrama de clases Cierre Millicom

c. *Estructura de los archivos:*

El archivo principal es el main.py que es donde se hace el llamado de las funciones para las consultas y el llenado del documento de Excel, el cual es manejado por el archivo Excel.py, este archivo implementa diferentes funciones para el llenado inicial del documento, la llamada de las consultas mediante el uso de las funciones del archivo sql_funtions.py y su escritura posterior en el documento de Excel, además del archivo dateconfig.py que se usa para la obtener la fecha del sistema. También el archivo sql_server.py, usado por el archivo de sql_funtions.py, se encarga de la conexión hacia la base de datos, y usa el archivo config.py que lee los datos de usuario, contraseña y servidor de la base de datos de un archivo de texto.

Finalmente, el proyecto se ejecuta en un módulo de Python para crear entornos virtuales llamado venv, el cual permite el aislamiento de dependencias al instalar paquetes y dependencias específicas para un proyecto sin afectar a otros proyectos

o al sistema base y la reproducibilidad ya que garantiza que cualquier persona que trabaje en el proyecto pueda crear el mismo entorno virtual con las mismas dependencias.

d. Archivos:

- main.py: Contiene el llamado a las funciones de consulta SQL y llenado de las celdas del Excel.
- sql: Esta carpeta almacena las consultas SQL individuales que son utilizadas en el proyecto.
- Excel.py: Contiene todo el manejo del documento de Excel desde la creación como la manipulación de sus celdas.
- sql_funtions.py: Contiene las funciones que permiten realizar consultas o modificaciones en una base de datos.
- sql_sever: contiene el manejo de la conexión hacia la base de datos.
- config.py: Realiza la lectura de los datos de acceso a la base de datos.
- configDate.py : Contiene la configuración de las fechas necesarias para las consultas que realizan en las sentencias SQL.
- venv: Contiene todos los archivos de configuración referentes al entorno virtual.
- requirements.txt: Lista de dependencias del proyecto.
- cierre.bat: Contiene los pasos para la ejecución automática del proceso.

e. Funcionamiento de la automatización:

Inicialmente se crea un archivo de Excel mediante la función ‘createExcel()’ con una base de cinco columnas estáticas que contiene información de áreas de donde se obtendrá información, fecha y periodo del reporte; como también la creación del encabezado de la tabla.

Posteriormente se realizan trece consultas para la obtención de los datos, y los datos obtenidos son escritos dentro de cada una de las celdas correspondientes, y son rellenadas de un color verde, esto mediante el uso de las funciones ‘addRowExcel()’ que se usa cuando se obtienen varios resultados de una misma consulta y la función ‘addValueExcel()’ que se usa cuando se obtiene un solo valor de la consulta.

Además, cada una de consultas son realizadas mediante la función 'script_df()' la cual obtiene la conexión de la base de datos, lee la sentencia SQL y devuelve una tabla en pandas con los resultados.

Finalmente, se añaden un conjunto de valores en cero en diferentes celdas a lo largo de la tabla y se realiza el envío del documento resultante mediante correo electrónico a la persona encargada de revisar el reporte.

Este proceso se presenta más detalladamente en el siguiente diagrama de flujo, que proporciona una guía paso a paso del flujo de trabajo, desde la creación del reporte hasta su envío, destacando las etapas y decisiones realizadas en el proceso. La comprensión de este flujo facilita la implementación, mantenimiento y optimización de la automatización del reporte.

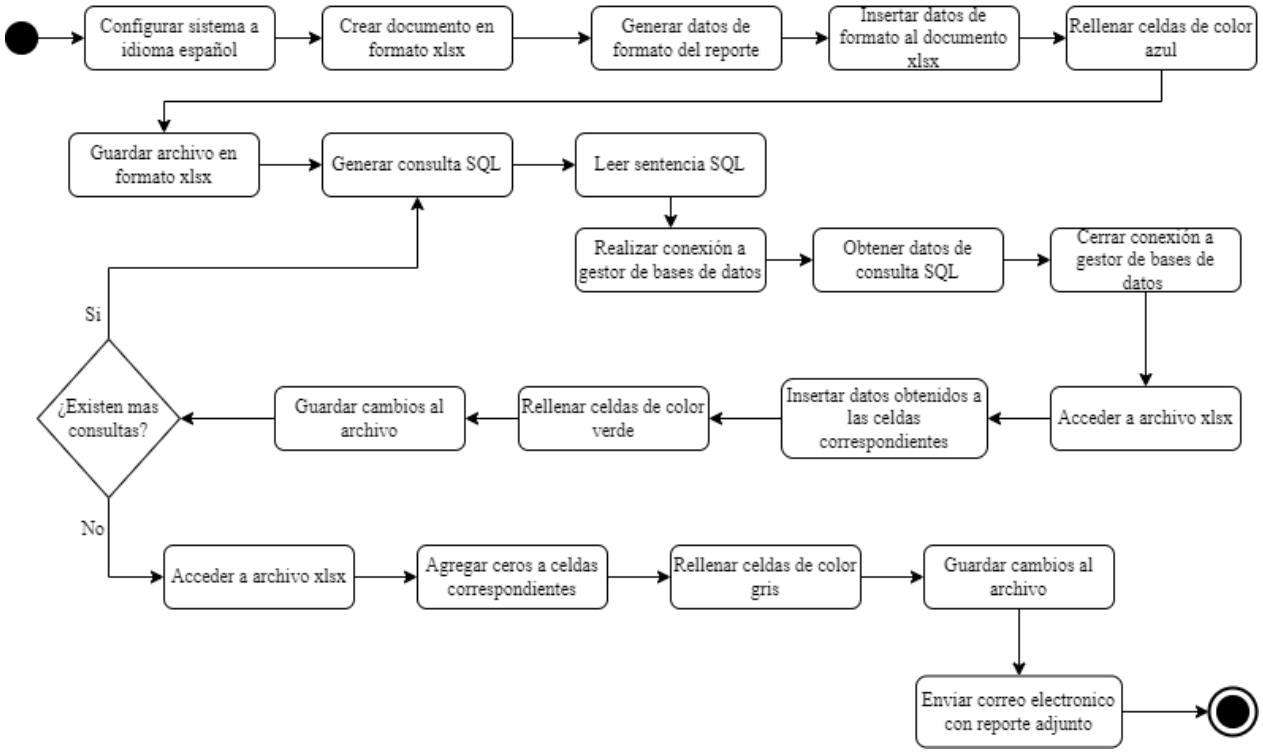

Figura 4: Diagrama de flujo del proceso

f. Funciones clave:

i. Excel.py:

- create_excel(): Esta es una función void, la cual se encarga de la creación del documento de Excel y generar el formato base en el cual se ingresarán

los datos. Usa internamente la función generate_data() para la creación de los datos del formato base y la manipulación del Excel se realiza mediante las funciones de la librería openpyxl. Debe ser la primera función a ejecutar para así tener un archivo de Excel en el cual insertar los datos.

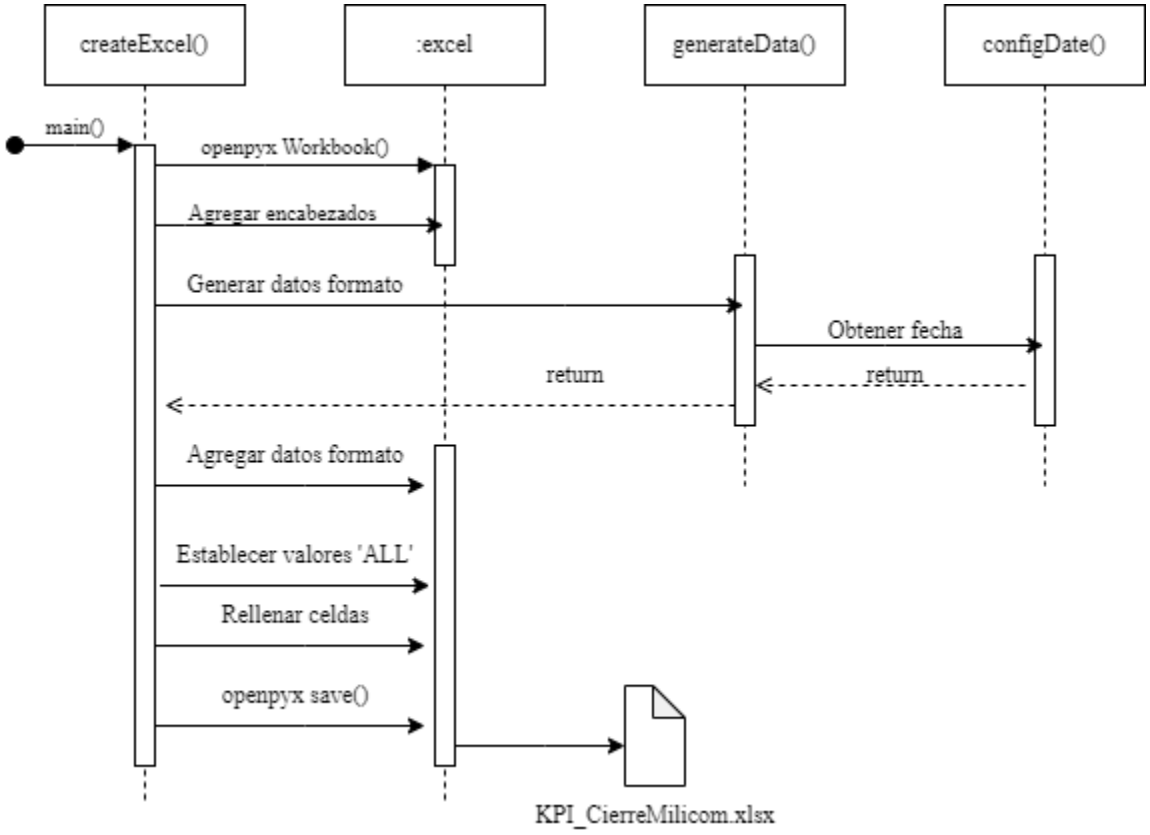

Figura 5: Diagrama de secuencia creación de Excel

- addRowExcel(query_path, number): Esta es una función void, que recibe como parámetros la ruta de la sentencia SQL (query_path) y el número de la fila donde se realizará la inserción. Inicialmente, realiza la consulta de la sentencia mediante la función script_df() y guarda este resultado en la variable data. Luego de esto, carga el libro de Excel creado mediante la función load_workbook(file_name) de openpyxl y se comienza a ingresar los datos obtenidos en cada celda de la siguiente manera sheet['LetraColumna'+number] = data['DATO''][0] en donde se ingresa la columna y el dato que se desea obtener. Posterior al ingreso del dato, se rellena la celda de color verde usando la función PatternFill() de openpyxl y se finalmente se guardan los cambios hechos en el archivo.

Esta función se debe usar cuando se obtengan los siete valores que se ingresan aquí desde la consulta SQL.

- addValueExcel(query_path, cell, columnValue, flag=False): Esta es una función void que tiene como parámetros la ruta de la sentencia SQL (query_path), la celda donde se realizará la inserción (cell), el nombre de la columna del dato que se va a obtener y una bandera que está por defecto en Falso(flag-opcional). Esta función sigue los mismos pasos que la función anterior a excepción de que se tiene una bandera que sí es enviada como True, a la función de script_df() se le enviaran dos parámetros más y si no se realizará de la forma anterior. Además de esto, esta función usa el dato completo de la celda, es decir, fila y columna para hacer el ingreso del dato sheet[cell] = data[columnValue][0].

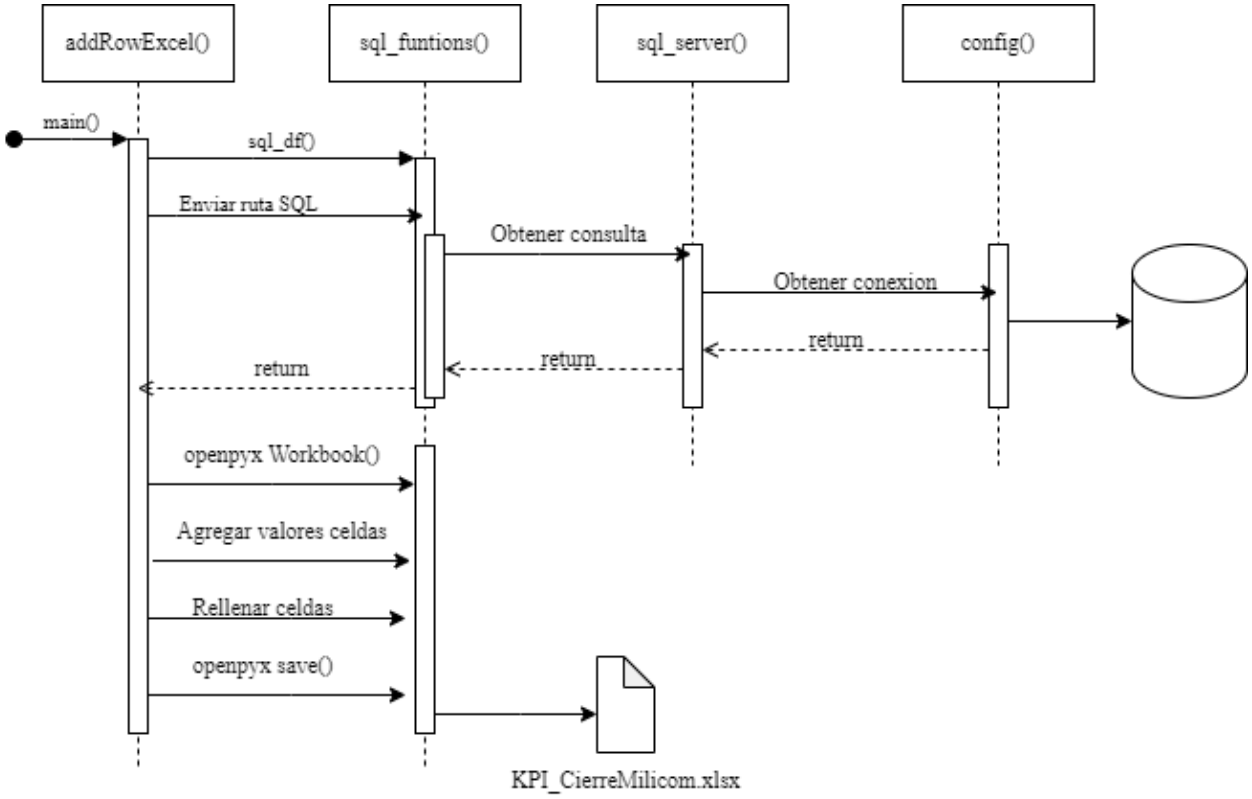

Figura 6: Diagrama de secuencia añadir valores a celdas

ii. sql_funtions.py:

- script_df(query, period, first_day='0000-00-00' ,last_day='0000-00-00'): La función se utiliza para ejecutar una consulta SQL y retorna los

resultados en un DataFrame de pandas. Tiene como parámetros la cadena que representa el nombre del archivo de consulta SQL ubicado en la carpeta "sql" (query), el período de tiempo para el que se desea ejecutar la consulta(period), la fecha de inicio del período de tiempo, que por defecto es '0000-00-00' (first_day-opcional) y la fecha de finalización del período de tiempo, que por defecto es '0000-00-00' (last_day - opcional).

Inicialmente genera la conexión a la base de datos mediante método con_sql(), qué es el encargado de obtener los datos necesarios y de gestionar dicha conexión. Luego se lee la consulta que se encuentra en la carpeta sql y realiza la inserción del periodo y fechas cuando corresponda en los campos que lo requieran dentro de la consulta. Finalmente se realiza la consulta y se retornan los datos.

- Observaciones:
  - Los parámetros de las funciones claves son todos de tipo cadenas de caracteres (strings) a excepción de la bandera que es de tipo booleano.
  - Cuando se necesiten realizar procesamientos de meses anteriores basta con modificar la cantidad de meses que se desea retroceder (tipo integer) en la función get_first_last_day_previous_month en el archivo Excel.py.

    Ejemplo:

    - Dos meses atrás:

      first_day, last_day = date.get_first_last_day_previous_month(2)

Para resolver el objetivo de analizar la eficiencia operativa antes y después de la implementación de la automatización, cuantificando el tiempo ahorrado, asegurando la precisión de los informes generados y comparando los resultados con los obtenidos a través de métodos manuales.

B. Sprint 2:

a. Optimización de consultas SQL:

Durante la migración de consultas desde Oracle SQL hacia SQL server se inició analizando las consultas individualmente, para luego unir consultas según el área del negocio de donde se obtiene la información. Además de esto, ya que en algunas de estas consultas se requiere usar subconsultas se optó por utilizar CTE- Common Table

Expressions debido a que esta herramienta permite mejorar la legibilidad, mantenibilidad y rendimiento de las consultas.

Además de esto es importante aclarar que el proceso de generación del informe de forma manual tomaba aproximadamente tres horas, debido a la realización de las consultas SQL en Oracle, la creación y el llenado de cada uno de los campos para cada una de las áreas de negocio, además de diferentes cálculos que se debían de realizar de información obtenida desde tableros de visualización, los cuales fueron sintetizados dentro de las consultas creadas para SQL Server.

b. Pruebas de eficiencia:

Para medir la eficiencia en la ejecución del proyecto, se realizan pruebas donde se mide el tiempo de ejecución de las partes claves del proyecto, es decir, las consultas por áreas del negocio (B2B – Business-to-business, Móvil y Fijo o Home), las consultas en total y la ejecución completa del proyecto, en donde se incluye la creación del archivo y el envío del correo electrónico.

Para el cumplimiento de cada una de las pruebas se definen criterios de evaluación en base al tiempo de ejecución definidos por el product owner llamados Criterio Temporal (Tabla 1), estos criterios se basan en su conocimiento y experiencia del funcionamiento de las consultas SQL realizadas y la cantidad de datos que se manejan dentro ellas.

TABLA I

CRITERIOS TEMPORALES DE RENDIMIENTO

Test	Criterio Temporal (seg)
test_excel_creation()	1
test_sql_insertion_b2b()	15
test_sql_insertion_home()	25
test_sql_insertion_movil()	25
test_sql_insertion()	65
test_complete_process()	131

Para la implementación de las pruebas se utiliza la librería pytest y en todas se sigue el siguiente formato:

```
@pytest.mark.performance
def nombre_test():
    start_time = time.time()
    # Lógica para ejecutar sección de código
    end_time = time.time()
    execution_time = end_time - start_time
    assert execution_time < limite, "Mensaje de paso de límite"
```

Se ejecutan cada una de las pruebas en conjunto (figura 6), en donde cada una de ellas obtienen resultados para pasar el criterio temporal establecido previamente, con su respectivo porcentaje de mejora (Tabla II).

Figura 7: Resultado de pruebas de rendimiento.

TABLA II

RESULTADOS PRUEBAS DE RENDIMIENTO

Test	Criterio Temporal (seg)	Resultado	Porcentaje de mejora
test_excel_creation()	1	0.1310367584228515	86.9%
test_sql_insertion_b2b()	15	5.688931465148926	62.07%
test_sql_insertion_home()	25	17.734113216400146	29.08%
test_sql_insertion_movil()	25	13.889460563659668	44.44%
test_sql_insertion()	65	35.766544580459595	44.97%
test_complete_process()	131	73.211	44.11%

El cálculo del porcentaje se realiza mediante la fórmula:

$$Porcentaje = \frac{CriterioTemporal - Resultado}{CriterioTemporal} * 100 \quad (1)$$

C. Sprint 3:

Para resolver el objetivo relacionado con la generación de estructura que tiene la tabla del IVR, inicialmente, se crea la tabla base donde se alojarán los datos obtenidos de las llamadas al IVR con el nombre TBL_MICRO_IVR, la cual contiene ocho campos base. Estos ocho campos se actualizan de forma periódica mediante el uso de la ETL de SSIS, para ello se trunca la tabla y se extrae la información del periodo determinado como se muestra en la figura 7. Se borra la información de esta tabla debido se usará como una tabla auxiliar para obtener los datos en un rango de tiempo determinado (normalmente un día), y posteriormente insertarlos en la tabla final.

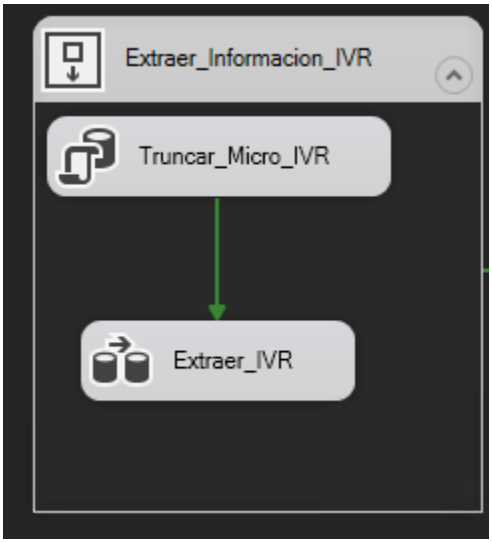

Figura 8. ETL de extracción de información IVR

Luego de esto, mediante la sentencia ALTER TABLE, se comienzan a añadir cada uno de los campos solicitados, cada uno de los campos es modificado mediante una sentencia de inserción SQL, algunos de estos datos son extraídos directamente desde los campos base que se encuentran en la tabla (ejemplo 1) y otros mediante el cruce con otras tablas ya existentes (ejemplo 2).

Ejemplo 1:

```
UPDATE A
SET A.COLUMNA =
        CONDICIONES
   END
FROM [TBL_Micro_IVR] A WITH (NOLOCK);
```

Ejemplo 2:

```
UPDATE A
SET A.REGIONAL =
        CONDICIONES
 END
FROM [TBL_Micro_IVR] A WITH(NOLOCK)
 LEFT JOIN
        TBL_EXTERNA B WITH(NOLOCK)
 ON A.COLUMNA=B.COLUMNA;
```

Finalmente se crea la tabla TBL_MICROANALISIS_IVR, la cual toma los mismos campos que la tabla anterior y es aquí donde se aloja todo el histórico de datos.

Para resolver el objetivo relacionado con la automatización del proceso de ejecución mediante el uso de una herramienta de ETL se usa igualmente un sprint.

D. Sprint 4:

a. *Funcionamiento de automatización:*

Para la automatización de las inserciones a la tabla TBL_MICRO_IVR se utiliza igualmente Python, generando la conexión a la base de datos y ejecutando las diferentes sentencias. Estas sentencias se han repartido en diferentes archivos, los cuales se ejecutan de forma secuencial.

Antes de comenzar a ejecutar las sentencias se obtiene la cantidad de días desde donde se obtendrá la información, esto en base a la fecha del sistema y enviando como parámetro dichos días a regresar.

Posteriormente, se eliminan los registros que tengan una fecha posterior a la cual se están obteniendo los datos y finalmente se insertan los datos que se encuentran en la tabla TBL_MICRO_IVR en la tabla TBL_MICROANALISIS_IVR.

Este proceso de Python es agregado a la ETL de SSIS para su ejecución automática como se muestra en la figura 9.

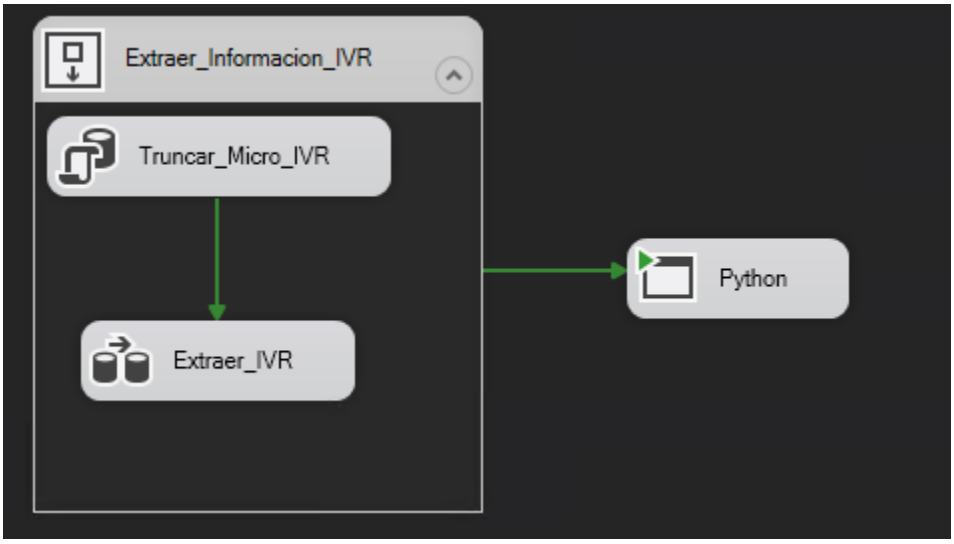

Figura 9. ETL completa de IVR

El funcionamiento de la ETL se agrega a un archivo .bat y luego es añadido al scheduler del servidor para su ejecución diaria.

b. *Procesamiento de meses anteriores:*

Se realiza el procesamiento de la información de cada uno de los meses anteriores a la creación de la tabla, para esto se usa la variable creada que obtiene la cantidad de días a retroceder y además se modifica la fecha dentro de la extracción de los datos base del IVR.

Luego de esto, se ejecutan las inserciones de un mes a la vez en la tabla TBL_MICRO_IVR, se eliminan los registros con fecha mayor a la que se está ejecutando en la tabla TBL_MICROANALISIS_IVR y se realiza la inserción de estos datos.

Este proceso se repite hasta que se encuentre actualizada la tabla TBL_MICROANALISIS_IVR y cada vez que se ejecuta se eliminan los registros de la tabla TBL_MICRO_IVR y se extraen nuevamente los del nuevo mes.

c. Estructura de los archivos:

El siguiente diagrama representa visualmente la estructura estática del sistema, mostrando las clases de objetos, sus atributos, métodos y las relaciones entre ellas.

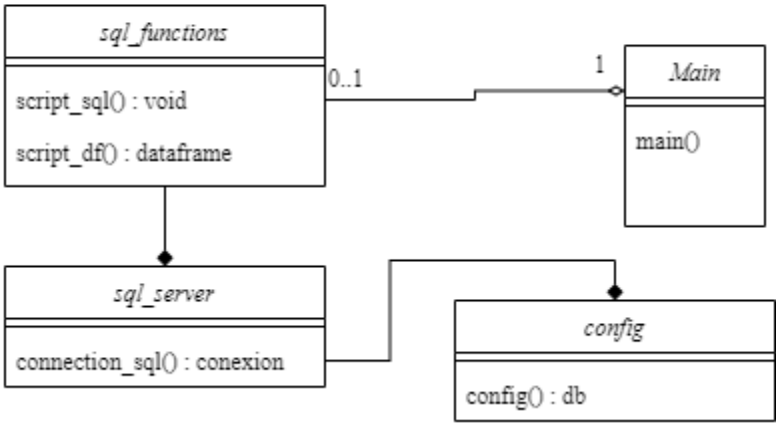

Figura 10. Diagrama de clases IVR

Además de esto, el proyecto de esta automatización contiene los siguientes archivos:

- main.py: Contiene el llamado a las funciones de consulta SQL y la obtención de la fecha requerida.
- sql: Esta carpeta almacena las consultas SQL individuales que son utilizadas en el proyecto.
- sql_funtions.py: Contiene las funciones que permiten realizar consultas o modificaciones en una base de datos.
- sql_sever: contiene el manejo de la conexión hacia la base de datos.
- config.py: Realiza la lectura de los datos de acceso a la base de datos.
- venv: Contiene todos los archivos de configuración referentes al entorno virtual.
- requirements.txt: Lista de dependencias del proyecto.
- ivr: Contiene todos los archivos necesarios para la creación de la ETL usando SSIS.
- ETL.bat : Contiene los pasos para la ejecución automática del proceso.

Adicionalmente, el funcionamiento para generar la conexión a la base de datos y las inserciones se realizan con la misma lógica que para el reporte de cierre.

VI. ANÁLISIS

La implementación de las automatizaciones de procesos ETL tuvo un impacto significativo en la eficiencia operativa dentro del equipo de trabajo. A través de cuatro Sprint, se realizaron actividades que permitieron mejorar la manipulación de datos y la generación de informes.

Durante el primer sprint, se enfocó en la elicitación de requisitos y la estructuración de la automatización para el cierre de Millicom, lo que permitió definir claramente las necesidades del proyecto y establecer una base sólida para el desarrollo subsecuente. Además, la creación de scripts en Python para la ejecución de consultas SQL y modificación de archivos de Excel demostró ser eficiente ya que redujo el tiempo necesario para generar informes, donde se obtuvo una mejoría del 99.32% debido a que se pasó de aproximadamente 3 horas manuales a aproximadamente 77.211 segundos (0.02034 horas) de forma automática.

En el segundo sprint, se implementaron las pruebas de efectividad, donde los resultados mostraron una disminución en los tiempos de ejecución esperados según los criterios establecidos. Se obtuvo una mejora del 86.9% al crear el archivo de Excel mediante la prueba test_excel_creation(), una mejora promedio del 45.2% en la consulta conjunta e inserción de datos para cada una de las áreas del negocio, promediando los resultados de las pruebas test_sql_insertion_b2b(), test_sql_insertion_home(), y test_sql_insertion_movil(). Además, hubo una mejora del 44.97% en la ejecución agrupada de las consultas para todas las áreas de negocio, según la prueba test_sql_insertion(), y una mejora del 44.11% en la ejecución completa del proceso mediante la prueba test_complete_process(). Esto indica una mejora en la eficiencia de las consultas y la modificación de los archivos de Excel.

El tercer sprint se centró en la creación de la ETL para la obtención de información de las llamadas realizadas hacia el IVR. La automatización de este proceso permitió la extracción, transformación y carga de datos en una tabla SQL específica, manteniendo la integridad de los datos cargados en la nueva tabla en base a la fuente de donde se extrajeron. Además, esto facilitó el análisis posterior por parte del equipo de trabajo debido al orden que se le dio a la estructura de los datos.

Finalmente, en el cuarto sprint se integraron los scripts en Python en la ETL de SSIS lo que permitió la ejecución automática del proceso según la periodicidad requerida y el procesamiento de los datos de todos los meses anteriores. Esta integración completa aseguró que el sistema pudiera manejar datos históricos y actuales de manera eficiente.

En general, los resultados de los Sprint indican que por medio de las automatizaciones realizadas se obtuvo una mejora en los tiempos de consulta, la generación de informes y en la estructura de los datos de las llamadas del IVR.

VII. CONCLUSIONES

- La implementación de scripts automatizados en Python y el uso de herramientas como SSIS permitieron una gestión más eficiente de los datos, la reducción del tiempo necesario para generar informes y la disminución de errores manuales, al evitar la creación manual de consultas y cálculos matemáticos y estadísticos.
- La automatización de consultas SQL y la generación de informes en Excel facilitaron el acceso a la información necesaria para la toma de decisiones, lo cual permitió reducir el uso de recursos humanos y tecnológicos, mejorando la productividad del equipo.
- La creación de una ETL para la información de las llamadas hacia el IVR y la integración de scripts en Python facilitaron la generación de una estructura que mantiene la integridad de los datos, permitiendo al equipo de trabajo realizar los análisis requeridos
- La aplicación de la metodología Scrum permitió que la gestión de los procesos fuera iterativa e incremental, asegurando mejoras continuas y adaptaciones según las necesidades del proyecto. Además, que la colaboración constante y la retroalimentación oportuna fueron clave para el éxito de las automatizaciones.

REFERENCIAS

[1] W. McKinney, Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, 2nd ed. Sebastopol, CA: O'Reilly Media, 2017.

[2] R. Kimball and M. Ross, The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd ed. Indianapolis, IN: John Wiley & Sons, 2013.

[3] C. J. Date, An Introduction to Database Systems, 8th ed. Boston, MA: Addison-Wesley, 2004.

[4] C. Bhat, B. S. Mithun, V. Saxena, V. Kulkarni and S. Kopparapu, "Deploying usable speech enabled IVR systems for mass use," 2013 International Conference on Human Computer Interactions (ICHCI), Chennai, India, 2013, pp. 1-5, doi: 10.1109/ICHCI-IEEE.2013.6887794.

[5] J. Anton and N. L. Petouhoff, Customer Relationship Management: The Bottom Line to Optimizing Your ROI. Upper Saddle River, NJ, USA: Prentice Hall, 2002.

[6] T. Wu, "ETL Function Realization of Data Warehouse System Based on SSIS Platform," 2010 2nd International Workshop on Database Technology and Applications, Wuhan, China, 2010, pp. 1-4, doi: 10.1109/DBTA.2010.5659098.

[7] K. Schwaber and J. Sutherland, The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game, 2017.