ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE SISTEMAS

UNIDAD DE TITULACIÓN

TÍTULO DEL TRABAJO DE TITULACIÓN:

WEB SCRAPING PARA ANÁLISIS DE LOS DATOS DEL PERSONAL DEL MINISTERIO DE EDUCACIÓN EN EL PERIODO 2015-2021.

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL GRADO DE MAGISTER EN SISTEMAS DE INFORMACIÓN

AUTOR: Eco. Erika Carolina Bonifaz Rosero
erika.bonifaz@epn.edu.ec

DIRECTOR: Ing. Henry Paz MSc.
henry.paz@epn.edu.ec

Quito, 2023

APROBACIÓN DEL DIRECTOR

Como director del trabajo de titulación WEB SCRAPING PARA ANÁLISIS DE LOS DATOS DEL PERSONAL DEL MINISTERIO DE EDUCACIÓN EN EL PERIODO 2015-2021, desarrollado por la Sra. Erika Carolina Bonifaz Rosero, con cédula de ciudadanía 0604219204, estudiante de la Maestría en Sistemas de Información Mención Inteligencia de Negocios y Analítica de Datos Masivos, habiendo supervisado la realización de este trabajo y realizado las correcciones correspondientes, doy por aprobada la redacción final del documento escrito para que prosiga con los trámite correspondientes a la sustentación de la Defensa Oral.

HENRY PATRICIO PAZ ARIAS

Firmado digitalmente por HENRY PATRICIO PAZ ARIAS
Fecha: 2022.12.29 19:49:31 -05'00'

Ing. Henry Paz MSc.
DIRECTOR

DECLARACIÓN DE AUTORÍA

Yo, Erika Carolina Bonifaz Rosero, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen este documento.

La Escuela Politécnica Nacional puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Firmado electrónicamente por:
ERIKA CAROLINA
BONIFAZ ROSERO

Eco. Erika Carolina Bonifaz Rosero

DEDICATORIA

A Dios, a mis padres, a mi esposo, a mi hermana y a Iliak.

Los amo con toda mi vida.

Erika Carolina Bonifaz Rosero

AGRADECIMIENTO

Agradecida con Dios por guiar mi camino y bendecirme en cada segundo de mi vida, con mis padres, quienes son mí mayor inspiración y mí mejor ejemplo de lucha y superación, con mi esposo, por impulsarme a ser mi mejor versión y volar conmigo, con mi hermana por ser mí apoyo y mí bastón en todo momento y con mi gran maestro y director Ing. Henry Paz MSc., por sus enseñanzas y por su valioso tiempo.

Erika Carolina Bonifaz Rosero

CONTENIDO

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

RESUMEN

Esta tesis de Maestría tiene la intensión de extraer la información pública disponible en el sitio oficial del Ministerio de Educación, dada la obligatoriedad de cumplimiento de la Ley Orgánica de Transparencia y Acceso a la Información Pública. Para objeto de este estudio únicamente nos enfocaremos en el literal c) La remuneración mensual por puesto y todo ingreso adicional, incluso el sistema de compensación, según lo establezcan las disposiciones correspondientes. La extracción de los datos se lo realiza a través de la técnica de Web Scraping. Adicionalmente se analiza la importancia de esta herramienta y se sugiere ciertos procedimientos de implementación por medio del lenguaje Python y comandos de línea. La visualización de los hallazgos proporcionados por esta data será presentada a través de un tablero dinámico para consumo público.

Palabras clave: DSR. Web Scraping. Python. Power BI. Ministerio de Educación. Remuneración.

ABSTRACT

This Master's thesis is intended to extract the public information available on the official website of the Ministry of Education, given the obligation to comply with the Organic Law on Transparency and Access to Public Information. For the purpose of this study, we will only focus on subparagraph (c) Monthly remunerationper post and any additional income, including the compensation system, as established by the corresponding provisions. The extraction of the data is done through the Web Scraping technique. In addition, the importance of this tool is analyzed, and certain implementation procedures are suggested through the Python language and line commands. The visualization of the findings provided by this datawill be presented through a dynamic board for public consumption.

Keywords: DSR. Web scraping. Python. Power BI. Ministry of Education. Remuneration.

CAPITULO I

1. INTRODUCCIÓN

El Ministerio de Educación es la entidad con la mayor asignación presupuestaria, un alcance de aproximadamente el 80% del monto total (Ministerio de Educación, 2021). Además, según cifras de Observatorio de Política Fiscal el sector educativo alcanza el 45% en tamaño de estado, considerándose aquel con el mayor peso por institución. De acuerdo con la ejecución presupuestaria sectorial presentada por el Ministerio de Economía y Finanzas, el sistema educativo, en el periodo de 2015 a 2021, consiguió un monto total de inversión aproximado de 27 mil millones, aunque, la menor inversión fue realizada en el año 2021.

La calidad de la educación es consecuencia del presupuesto educativo y los salarios, por lo que el aumento de las desigualdades y el deterioro de la capacidad institucional de generar aprendizaje se ha evidenciado en la deficiencia de este sector. Existen múltiples métodos para la asignación de recursos a la educación, pero la restricción presupuestaria es uno de los criterios dominantes. A pesar de ello, se conoce información poco detallada sobre la relación entre remuneraciones mensuales y el número de trabajadores en el Ministerio de Educación.

Es necesario conocer la evolución de los recursos destinados al sistema educativo puesto que esta característica influye en la calidad de este y por tanto en el desarrollo del país. Existe una cantidad considerable de investigaciones que estudian el impacto de la inversión monetaria en la educación, por lo que el análisis de una parte del presupuesto educativo generará información útil para el estado, instituciones académicas, profesores y estudiantes.

En este trabajo, se pretende realizar un análisis de las remuneraciones mensuales de los trabajadores del Ministerios de Educación desde enero de 2015 hasta diciembre de 2021. El conjunto final de datos comprende ochenta y cuatro reportes mensualizados compuestos por la información del empleado público, salario e ingresos adicionales. Una alternativa para este análisis es la utilización de la técnica de Web Scraping, puesto que esta permite extraer la información de sitios web, adicionalmente, la presentación de los resultados será a través de un

tablero dinámico en el cual se visualizarán indicadores y relaciones mensualizadas; para estos dos procesos se utilizará Python y Power BI respectivamente.

1.1 Objetivo general

Analizar la información del personal del Ministerio de Educación en el periodo 2015 a 2021 a través de técnicas de Web Scraping y herramientas de visualización.

1.2 Objetivos específicos

- Realizar una revisión sistemática de la literatura, enfocada en el estudio de contribuciones relacionadas con el uso de técnicas de Web Scraping.

- Extraer datos masivos a través de la técnica de Web Scraping.

- Seleccionar e implementar técnicas estadísticas para medir el tamaño del personal que labora en el Ministerio de Educación y la relación que existe con la inversión a dicha entidad.

- Visualizar el análisis evolutivo del tamaño del personal del Ministerio de Educación mediante la herramienta Power BI.

1.3 Alcance

Debido a que no existen estudios previos referentes a la información del personal del Ministerio de Educación, este trabajo propone una solución que abarca desde la recolección de los datos hasta el análisis de resultados, todo esto gestionado mediante un Web Scraping, que es el objetivo principal del proyecto.

Algunos de los inconvenientes a superar son: obtener datos que se puedan utilizar para el desarrollo del proyecto, gestionar la descarga y el almacenamiento de los datos, desarrollar programas específicos para el tratamiento y manipulación de los datos, planificar y distribuir de manera correcta las actividades del proyecto para cumplir con el cronograma de trabajo.

Debido a estos inconvenientes, se debe limitar las tareas que se llevarán a cabo para la ejecución del proyecto, las mismas que se detallan a continuación:

- El análisis se realizará sobre el sitio web oficial del Ministerio de Educación.

- Los datos se descargarán mediante técnicas de Web Scraping, para esto se utilizará Python como lenguaje de programación, la librería Beautiful Soup para facilitar la extracción de datos a partir de las páginas Web. La biblioteca Tabula-py que permite la extracción de la tabla y convierte el archivo PDF directamente en marcos de datos utilizando el lenguaje de programación Python.

- El almacenamiento de los datos extraídos se realizará a través de archivos planos, almacenados en el disco duro. Estos elementos, tanto los de la base de datos como los archivos HTML, serán compartidos con el propósito de cumplir el objetivo de crear un nuevo conjunto de datos relacionado al ámbito educativo.

1.4 Marco Teórico

1.4.1 Antecedentes del estudio

De acuerdo a la Ley Orgánica de Transparencia y Acceso a la Información Pública en el Art.7 señala que, "por la transparencia en la gestión administrativa que están obligadas a observar todas las instituciones del Estado que conforman el sector público, difundirán a través de un portal de información o página web, así como de los medios necesarios a disposición del público, implementados en la misma institución, la información mínima actualizada, que para efectos de esta Ley, se la considera de naturaleza obligatoria". Para objeto de este estudio únicamente nos enfocaremos el literal "c) La remuneración mensual por puesto y todo ingreso adicional, incluso el sistema de compensación, según lo establezcan las disposiciones correspondientes".

Dada la obligatoriedad de cumplimiento de esta Ley, el Ministerio de Educación tiene dentro de su página web, en el segmento de transparencia, un repositorio donde se encuentra toda la información mencionada en el Art. 7 de la LOTAIP, de los literales a) al t), siendo el literal c) el objeto de estudio. Para esto, la técnica Web Scraping nos permitirá realizar la extracción de

datos únicamente de los reportes de interés para los periodos de este estudio. Para lo cual es necesario conocer que es, en qué consiste y la importancia que tiene como herramienta de investigación esta técnica de extracción de datos.

1.4.2 Web Scraping

El Web Scraping o extracción de la web es una técnica para extraer datos de la red Informática Mundial (WWW) y guardarlos en un sistema de archivos planos o en una base de datos, es utilizada mediante programas de software para extraer información de sitios web que simulan la navegación de un humano en la red informática ya sea utilizando el protocolo HTTP manualmente, o incrustando un navegador en una aplicación (Khabsa&Giles, 2014).

Es ampliamente conocida por ser una técnica efectiva al momento de recolectar información con datos heterogéneos, complejos y de gran escala, Big Data. De hecho, la gran variedad de información y los distintos escenarios para la visualización de esta, han permitido convertir sitios web en conjuntos de datos estructurados y organizados.

El web scraping está muy "relacionado con la indexación de la web, la cual indexa la información de la web utilizando un robot y es una técnica universal adoptada por la mayoría de los motores de búsqueda. Sin embargo, el web scraping se enfoca más en la transformación de datos sin estructura en la web (como el formato HTML) en datos estructurados que pueden ser almacenados y analizados en una base de datos central" (Pulido&Morales, 2021).

Es una solución intermedia entre la recolección manual de datos (marcando, copiando y pegando textos) y el acceso automatizado a los mismos con base en un protocolo predeterminado (una interfazde programación, API). Se aplica cuando tales protocolos no están disponibles y la cantidad de datos que se desea extraer es demasiado grande para que pueda ser realizada de forma manual (López, 2018). El Web Scraping es una forma de minería de datos no estructurada, que permite extraer información de páginas web, escanear su código HTML y generar patrones de extracción de datos (Murillo&Saavedra, 2017).

Para realizar Web Scraping se utiliza un programa, orquestador, que organiza y ejecuta las repeticiones al browser. Se deben tener bien definidos los elementos a buscar, y se indique el estado de la búsqueda a realizar (búsqueda exitosa, errores en la búsqueda, sin resultados). El proceso de extracción de información se divide en dos fases: la adquisición de recursos web y la extracción de la información objetivo.

Esta técnica empieza por componer una petición HTTP para adquirir recursos del sitio web, esta solicitud puede tener el formato de una URL que contenga una consulta GET o un pedazo de mensaje HTTP que contenga una consulta POST, posteriormente, una vez que la solicitud es recibida y procesada el recurso solicitado será recuperado del sitio web y luego enviado de regreso al programa web crawler; el recurso puede estar en múltiples formatos, como páginas web construidas a partir de HTML, fuentes de datos en formato XML o JSON, o datos multimedia como imágenes, archivos de audio o vídeo.

Luego, el proceso de extracción analiza, reformatea y organiza los datos descargados de forma estructurada. Posteriormente se analiza los datos para obtener información. En cuanto a las técnicas para la extracción de datos se tiene: Web bot, Spider, Crawler, Arañas y Rastreadores que inspeccionan las páginas web de internet de forma metódica y automatizada, se utilizan para rastrear la red, leen la estructura de hipertexto y accede a todos los enlaces referidos en el sitio web, son utilizadas la mayoría de las veces para poder crear una copia de todas las páginas web visitadas para que después puedan ser procesadas por un motor de búsqueda y, esto hace que se puedan indexar las páginas, proporcionando un sistema de búsquedas rápido.

Otra técnica de extracción de datos son las Plataformas de Agregación Verticales, la cuales crean y controlan numerosos robots que están destinados para mercados verticales específicos, se realiza mediante el establecimiento de la base de conocimientos destinado a la totalidad de plataformas verticales y luego a crearla automáticamente, mide las plataformas por la calidad de la información que se obtiene, lo que asegura que la robustez de las plataformas utilizadas consiga la información de calidad y no sólo fragmentos de datos inútiles. La Reorganización de Anotación Semántica es una técnica que puede realizarse para páginas web que adoptan marcas y anotaciones que pueden ser destinadas a localizar fragmentos específicos semánticos o metadatos, las anotaciones pueden ser incrustadas en las páginas y esto puede ser visto como

análisis de la representación estructurada (DOM), esto permite recuperar instrucciones de datos desde cualquier capa de páginas web (Hernández&Gómez, 2015).

Para realizar el proceso de Web Scraping se pueden encontrar herramientas como librerías, complementos del navegador, servicios en línea, aplicaciones de escritorio, frameworks, que pueden ser de código abierto, gratuitas o comerciales (Rosero, 2021). Los módulos esenciales de un programa de Web Scraping son los que componen una petición HTTP, como Urllib2, que define un conjunto de funciones como la autenticación, las redirecciones, las cookies, entre otras, o Selenium, que construye un navegador web, como Google Chrome o Internet Explorer, y permite a los usuarios automatizar el proceso de navegación por un sitio web mediante programación; y los que analizan y extraen la información del códigoHTML en bruto, como Beautiful Soup o PyQuery .

El primero está diseñado para el scraping de HTML y otros documentos XML, además, permite navegar, buscar y modificar un árbol de análisis sintáctico para descomponer archivos HTML y extraer la información, procesa y convierte una codificación en legible para el usuario. Mientras que el segundo, proporciona un conjunto de funciones similares a JQuery para analizar documentos XML, pero a diferencia de Beautiful Soup, solo soporta LXML para el procesamiento rápido de XML. Los programas de web crawler se crean para reconocer la estructura de datos de la página web de forma automática o para proporcionar una interfaz gráfica que reduzca la necesidad de escribir manualmente el código rastreador y mitigar la complejidad de extracción de información. Scrapy es un crawler escrito en Python que acelera el proceso de creación y ampliación de grandes proyectos de rastreo, además proporciona un Shell que simula el comportamiento de navegación de un usuario humano. (Zhao, 2017)

Como herramienta de investigación, el Web Scraping multiplica las posibilidades de recolectar información que, aunque está publicada en Internet, es inaccesible por no tener una estructura clara, estar dispersa o ser masiva (López, 2018). Los servicios web existentes no cubren todas las posibles demandas de datos de los usuarios, en consecuencia, el web scraping ofrece un servicio válido y valioso a una amplia gama de aplicaciones informáticas, que van desde simples robots de extracción hasta multiservidores en línea. El Web Scraping puede utilizarse para una gran variedad de escenarios, como la extracción, monitorización y comparación de cambios de

precios, la recogida de reseñas de productos, la recopilación de listados inmobiliarios, la monitorización de datos meteorológicos, dar seguimiento del presupuesto gubernamental para un análisis de calidad de la educación o la tasade desempleo, la detección de cambios en el sitio web y la integración de datos web. (Glez-Peña, Lourenço, López-Fernández, Reboiro-Jato, & Fdez-Riverola, 2014).

1.4.3 Extractor de datos

El extractor de datos es un software interactivo que obtiene la información necesaria de una página web, aprovechando la estructura HTML de la página web y utilizandolas expresiones regulares, palabras claves, elementos y atributos contenidos en la estructura propia del lenguaje de contenido para extraer la información encapsulada que existe en estos selectores (Ruiz, 2021).

Tabla 1. Procedimiento para realizar web scraping

PASOS	**DESCRIPCIÓN**
Inspeccionar código fuentes	Identificar dentro del lenguaje de contenido los elementos, atributos, expresiones regulares o palabras clave que contienen los datos que se desean.
Extracción de datos	Escribir un programa y diseñar las funciones que sean necesarias para conseguir lo datos que se necesitan.
Almacenar o procesar los datos extraídos	Trasladar los datos extraídos en una estructura útil para el uso y la visualización posterior.

Fuente: Ruiz Ronquillo, 2021

1.4.4 Usos de web scraping

El uso del web scraping va acorde a la necesidad de cada individuo, esta técnica permite aprovechar datos de la web que son cada vez más valiosos y generar aplicaciones prácticas como (Ruiz, 2021):

- Comparar precios en línea
- Descubrir cambios en la web
- Recolectar noticias, artículos, etc.
- Recopilar contactos de interés
- Aplicaciones de Inteligencia Empresarial
- Aumentar significativamente la cantidad de datos para investigación
- Análisis del competidor
- Periodismo de datos
- Análisis de sentimiento de clientes
- Monitoreo de datos meteorológicos

1.4.5 Impedimentos para realizar web scraping

Existen algunos factores que entorpecen la recuperación de datos con herramientas destinadas a realizar web scraping, llegando a complicar o impedir que estos programas cumplan con el objetivo para el cual fueron desarrollados (Ruiz, 2021). Cuanto más valor tenga una página web, más seguras serán las medidas que adoptarán los operadores. Según Imperva (Imperva, 2016), algunos ejemplos de impedimentos para realizar web scraping son:

- Sistemas de autenticación como CAPTCHA contenidos en páginas web, endonde no es posible programar un scraper totalmente automatizado para extraer información ya que, en cada solicitud requiere una validación totalmente diferente a la anterior (Ruiz, 2021).

- Páginas antiguas que no han actualizado ni modificado desde hace mucho tiempo la estructura simple del lenguaje de contenido lo que imposibilita la extracción organizada desde las etiquetas HTML (Ruiz, 2021).

- Bloqueadores de dirección IP que se activan como medida de prevención ante ataques de denegación de servicio que se puede producir al recibir múltiples peticiones de un bot en un mismo servidor (Ruiz, 2021).

- Uso de cookies para dar seguimiento a la interacción de los usuarios con el recurso web para rastrear patrones de navegación anormales y tasas de solicitudes sospechosamente agresivas lo que ayuda a identificar bots (Ruiz,2021).

1.4.6 Precauciones antes de hacer web scraping

Para buscadores como Bing, Google, DuckDuckGo, el análisis de millones de páginas y la indexación de estas son tareas cotidianas del día a día, sin embargo, para algunos sitios web el realizar estas acciones programadas y repetitivas pueden considerarse maliciosas, debido a que todos los bots usan el mismo sistema para acceder a los datos del sitio y resulta complicado distinguir entre quienes son y quienes no, por lo tanto, antes de realizar web scraping es importante tener en cuenta ciertos aspectos (Ruiz, 2021):

- Verificar las restricciones de acceso a través del archivo robots.txt que se encuentra ubicado en el directorio principal de la web (Ruiz, 2021).

- Revisar políticas de privacidad y navegación de la página web a la cual se desea scrapear (Ruiz, 2021).

- Tener en cuenta las leyes locales de protección de datos del país en donde aloja el sitio web (Ruiz, 2021).

- No realizar tareas repetitivas en corto tiempo para evitar bloqueos (Ruiz, 2021).

- Verificar que los enlaces a extraer no se hayan ocultado mediante CSS ya que es una técnica de honeypot de los sitios anti-scraping para atrapar arañasweb (Ruiz, 2021).

1.4.7 Técnicas de web scraping

La técnica de extracción de datos web scraping ha revolucionado la interrelación usuario computadora, facilitando el acceso a una gran cantidad de información para beneficio del usuario (Ruiz, 2021). Existen varias técnicas para recopilar información que se han venido desarrollando a través del tiempo como las que se describen a continuación (Saurkar, 2021):

Copiar y pegar: Es una de las soluciones más simples de raspado web, pero también uno de las más primitivos y lenta, por lo general, solo se realiza una vez (Saurkar, 2021).

Programación HTTP: Facilita el acceso y la recuperación de información contenida en páginas web estáticas y dinámicas a través de la expedición de peticiones HTTP al servidor web remoto (Saurkar, 2021).

Parser HTML: Es un analizador sintáctico que permite comprender la estructura y contenido de una página o sitio web para extraer datos de preferencia. Generalmente se construyen mediante lenguajes de programación como Python, PHP y R (Saurkar, 2021).

Web Scraping software: El uso de esta técnica de web scraping es menos compleja que la técnica anterior, con la ayuda de una interfaz gráfica permite reconocer automáticamente la estructura de una página web y simular la navegación humana, eliminando la necesidad de redactar manualmente el código de un bot para conseguir los datos deseados y transformarlos en un formato útil y descifrable (Saurkar, 2021).

Análisis DOM (Document object model): Con los objetos DOM que representan documentos HTML y XML a través de un conjunto estándar de objetos, es posible acceder, añadir y modificar el contenido estructurado del HTML y XML a través de un lenguaje de programación como JavaScript y recuperar el contenido o parte del contenido dinámico generado por los scripts del cliente web (Saurkar, 2021).

Uso de APIs: A través de una interfaz de programación de aplicación, ya sea privada o pública, también es posible acceder y recuperar datos estructurados mediante programación, algunos ejemplos de sitios web que proporcionan APIs son LinkedIn, Twitter y Facebook (Ruiz, 2021).

1.4.8 Herramientas de web scraping

Existen diversas herramientas dedicadas a realizar web scraping que facilitan la comprensión del usuario y el ahorro de tiempo cuando se desea conseguir grandes cantidades de datos entre las cuales se encuentran herramientas de programación, aplicaciones de escritorio; aplicaciones web y extensiones para navegador.

1.4.8.1 Beautiful Soup

Beautiful Soup es una biblioteca de Python para extraer datos de archivos HTML y XML. Funciona con su analizador favorito para proporcionar formas idiomáticas de navegar, buscar y modificar el árbol de análisis, puede combinarse con el soporte nativo del lenguaje para las conexiones HTTP, facilita la extracción de páginas web y reduce las líneas de código que crean los programadores. (Breuss, 2021).

Se instala e importa esta librería de la siguiente manera:

```
pip install beautifulsoup4
import requests
from bs4 import BeautifulSoup
URL = "https://realpython.github.io/fake-jobs/"
page = requests.get(URL)
soup = BeautifulSoup (page.content, “html.parser”)
```

En una página web HTML, cada elemento puede tener asignado un atributo que hace que el elemento sea identificable de forma única en la página, para ello se explora la página web (clic derecho + Inspeccionar). Por ejemplo, se puede buscar el atributo con el valor “Resultados”

```
<div id="Resultados">
  <! -- all the job listings -->
</div>
```

Se puede realizar la búsqueda a través de una clase, como:

```
job_elements = results.find_all("div", class_="card-content")
```

1.4.8.2 Scrapy

Es el marco de trabajo de Python de código abierto y multipropósito más popular en temas relacionados a crawling y scraping en la web, aplicado principalmente en minería de datos por ser rápido y concurrente, lo que resulta muy útil al momento de descargar datos de manera masiva (Ruiz, 2021).

1.4.8.3 Tabula-py

En la actualidad, mucha información se encuentra almacenada en archivos PDF, es decir, no es necesario disponer la información archivos con formatos CSV o JSON, por lo cual es necesario poder acceder a esta información, la forma más sencilla es copiar y pegar la tabla en una hoja de cálculo o en un editor de texto la data objetivo, pero es posible que la información del documento sea variada, sin estructura y extensa por lo que puede ser un trabajo tedioso e incurrir en errores. Por ello, Python proporciona una biblioteca de código abierto, también conocida como tabula-py, que permite a los usuarios extraer más de una tabla por separado.

Tabula es una herramienta basada en una aplicación de interfaz gráfica de usuario (GUI); sin embargo, tabula-java es una herramienta basada en una interfaz de usuario de línea de comandos (CUI), tabula-java proporciona los enlaces de Ruby,R y NodeJS, pero no para Python por lo cual, se introdujo el concepto de tabula-pyque proporciona la vinculación de Python.

Tabula-py es una envoltura básica de tabula-java que permite a los usuarios la extracción de la tabla y convierte el archivo PDF directamente en marcos de datos utilizando el lenguaje de programación Python. El usuario también puede extraer tablas del PDF y convertirlas en archivos con formato TSV, CSV o JSON. Adicionalmente, posibilita el desarrollo de analítica de datos avanzada y la automatización de este análisis a través de scripts de Python y la conversión a pandas DataFrame.

Tabula se utiliza para potenciar el periodismo de investigación en organizaciones de noticias de todos los tamaños, como ProPublica, The Times of London, Foreign Policy, La Nación

(Argentina), The New York Times y el St. Paul (MN) Pioneer Press, puesto que estas organizaciones utilizan este framework para para convertir los informes en formato PDF, con una gran precisión, en hojas de cálculo Excel, CSV y archivos JSON para posteriormente realizar minería de datos, ETL y análisisde las base de datos con el fin de generar información de la data obtenida, es decir facilita la adquisición de la información en documentos complejos. (Tabula Python,s.f.)

Puesto que tabula-py es una librería de código abierto de Python, utilizaremos el instalador pip para instalar la misma. Luego se importa.

```
pip install tabula-py
import tabula as tb
```

Esta librería proporciona varias funciones como la lectura de un archivo PDF, la lectura de una tabla en una página específica de un archivo PDF, la lectura de múltiples tablas en la misma página de un archivo PDF, o la conversión de archivosPDF directamente a un archivo CSV.

Por ejemplo, la lectura de los datos se puede realizar con la función read_pdf () con la siguiente estructura:

```
read_pdf(input_path, output_format=None, encoding='utf-8', java_options=None,
pandas_options=None, multiple_tables=True)

   Argumentos:
      input_path archivo PDF objetivo tareget PDF file.
      output_format: formato de retorno de la data (dataframe o un json)
      encoding: codificafro de pandas, Default: utf-8
      java_options (list, optional):
      pandas_options (dict, optional): opciones de pandas
'header': None}
```

multiple_tables (booleano): Capacidad para manejar mas de una tabla en una misma página del archivo.

Pages: pagina que se quiere leer del archivo.

Resultados:

Lista de DataFrames or diccionarios.

Se puede observar cómo ejemplo demostrativo:

```
tb.read_pdf ( ‘archivo.pdf’ , pages = 1)
```

Finalmente, una vez realizada la lectura del archivo esta se puede almacenar en un panda DataFrame para su posterior análisis.

1.4.8.4 PyPDF2

Es una librería de Python que se utiliza para realizar las principales tareas de los archivos PDF, como la extracción de la información específica del documento, la fusión de los archivos PDF, la división de las páginas de un archivo PDF, la adición de marcas de agua a un archivo, el cifrado y descifrado de los archivos PDF, etc. Esta librería proporciona metadatos sobre el documento PDF, por ejemplo, el tema, el número de hojas en el archivo, el autor, entre otras funciones que son de interés al momento de almacenar la información. (Sachdeva, 2021)

Para instalar PyPDF2, se utiliza el instalador pip y se importa la librería.

```
pip install PyPDF2
import PyPDF2
```

Las funciones de esta librería facilitan la adquisición de información de un formato de archivo que es fácil de descargar y es muy frecuentemente utilizado para transmitir información, por ejemplo, la creación de objetos y la lectura de los mismos.

```
pdfFileObject = open (pdf_path, 'rb')
pdfReader = PyPDF2.PdfFileReader(pdfFileObject)
```

1.4.9 Dashboard

El tablero dinámico o dashboard es una aplicación web que permite analizar los datos recolectados, mediante indicadores clave y gráficos que resumen los datos y proporcionan información que permite la evaluación específica de ciertos perfiles o de determinados periodos de tiempo. Los datos son examinados mediante la aplicación de filtros, los cuales generan cambios dinámicos en las visualizaciones proporcionando un servicio de consultas.

La visualización convierte lo abstracto y complejo en concreto y sencillo al amplificar la cognición humana. Few explicó que los cuadros de mando bien diseñados conducen a una comunicación eficaz, así como a una correcta toma de decisiones, al destacar la información relevante y al colocar los elementos de forma consistente facilita la rápida percepción mediante diversas tecnologías de visualización y apoya el objetivo inmediato y final de la toma de decisiones. Esta aplicación pretende presentar patrones de comportamiento en el entorno de la educación con la finalidad de generar un impacto en la calidad de esta a travésde la evaluación de los recursos destinados. (Park & Jo, 2015)

1.5 Metodología de minería de investigación

Dentro de la minería de datos existe un conjunto de metodologías que permiten ejecutar de manera sistemática y no trivial un proyecto data mining de principio a fin. Estas metodologías ayudan a comprender el proceso de descubrimiento de conocimiento y brindan orientación para planificar y materializar un proyecto en mente.

Para el desarrollo de este trabajo se utilizará la metodología de la investigación basada en la Ciencia del Diseño (Design Science Research, DSR) que está compuesta de un conjunto de seis fases y que, mediante el conocimiento y la comprensión de un determinado problema, permite diseñar una solución construyendo un artefacto (Rosero, 2021). Las 6 fases de la metodología consistenen: identificar el problema y la motivación, definir objetivos para la solución, diseño y

desarrollo, evaluación, demostración y, comunicación.

El propósito de cada una de las fases se detalla a continuación:

Fase 1: guiar en la comprensión del negocio, a la vez que se identifica el problema y la motivación del proyecto. Además, se establece los objetivos de la solución junto con un plan de ejecución.

Fase 2: generar una familiarización con los datos mediante varias actividades: la descarga inicial de datos, la exploración y descripción de los datos, la generación de gráficas informativas sobre los datos, entre otras. Adicionalmente, en esta fase se obtiene la definición de la capa de acceso a datos y el servicio de descarga de la capa de servicios del framework el cual se encarga de la orquestación de la descarga y almacenamiento de los datos. Además, se presenta el componente de ingreso de URL's que pertenece a la capa de aplicación del framework al igual que el servicio de ingreso de URL's perteneciente a la capa de servicios el framework.

Fase 3: una vez que se conoce los datos se puede trabajar con ellos mediante tareas de limpieza y transformación, esto con el fin de facilitar su uso en tareas futuras. Esta fase conduce en la obtención del servicio de procesamiento de datos de la capa de servicios del framework el cual se encarga de procesar los datos para obtener unconjunto de estos que cumplan las características requeridas para el análisis.

Fase 4: debido a que en el proyecto no se pretende desarrollar un modelo predictivo o prescriptivo, en esta fase se llevarán a cabo tareas de análisis de datos para la obtención de indicadores clave sobre estos. En esta fase se obtiene un aplicativo a manera de dashboard correspondiente a la capa de aplicación del framework el cualse encarga de la presentación de resultados del análisis de los datos mediante indicadores clave.

Fase 5: dentro de esta fase se analizará los resultados obtenidos para conocer el nivel de adopción de la tecnología del Web Scraping.

Fase 6: se relaciona con las fases de demostración y comunicación de DSR, se sustenta con la publicación del trabajo realizado para este proyecto, así como los resultados obtenidos.

1.5.1 Fase 1: Comprensión del negocio

En la fase de comprensión del negocio, a la vez que se identifica el problema y la motivación del proyecto. Además, se establece los objetivos de la solución junto con un plan de ejecución.

1.5.1.1 Objetivos del negocio

En el contexto del Web Scraping no existe un proceso automatizado que destine sus esfuerzos a un análisis de ámbito netamente educativo, entendiéndose con esto que el análisis esté orientado al sitio Web del Ministerio de Educación y que se analice específicamente al personal del Ministerio de Educación para el periodo 2015-2021. De forma aún más específica, no existe un proceso automatizado que extraiga toda la información pública disponible en cuanto a remuneración mensual unificada.

Ante esta problemática, se propuso el desarrollo de un Web Scraping que permita entender como es la evolución del personal que labora en el Ministerio de Educaciónen cuanto al número de colaboradores por año, a su remuneración y cargo que ocupa.

1.5.1.2 Análisis de la situación actual

Dentro de esta tarea se detallan aspectos importantes relacionados con los recursos, limitaciones, suposiciones y otros factores importantes para el objetivo de la mineríade datos y el plan del proyecto (Rosero, 2021). El estado de la situación actual se puede determinar respondiendo a las siguientes preguntas:

¿Cuál es el conocimiento previo disponible acerca del problema?

No se cuenta con un conjunto de datos específicos sobre los cuales realizar el análisis propuesto. A pesar de contar con la información pública disponible, no existe una base de datos que contenga la información mensual disponible acerca dela remuneración mensual unificada.

¿Se cuenta con la cantidad de datos requerida para resolver el problema?

Actualmente no existe un conjunto de datos para llevar a cabo el análisis, pero se pretende, como parte de los objetivos, crear este conjunto de datos a partir de la extracción, mediante técnicas de Web Scraping, del contenido público del sitio Web del Ministerio de Educación.

¿Cuáles son los requisitos, supuestos y restricciones del proyecto?

- Se debe contar con una conexión estable de Internet para el proceso de recopilación de datos puesto que se obtendrán del sitio Web del Ministerio de Educación.

- Se espera que el sitio Web sea estático y cuente con un sistema sencillo para la navegación.

- Se cuenta con que el Ministerio de Educación publique su información dentro de su sitio Web oficial, con formato PDF, para luego ser transformados en archivo plano.

- Se considera que al acceder a contenido público en el sitio Web del Ministerio de Educación, no se producirán restricciones de acceso para poder extraer la información requerida.

1.5.1.3 Objetivos de la minería de datos

- Comprender el formato del sitio Web del Ministerio de Trabajo, en donde publica los datos.
- Determinar qué información es la que se publica en el sitio Web del Ministerio de Educación.

- Identificar, describir y organizar los datos en el texto (HTML).

1.5.2 Fase 2: Comprensión de los datos

En esta fase es importante generar una familiarización con los datos mediante varias actividades: la descarga inicial de datos, la exploración y descripción de los datos, la generación de gráficas informativas sobre los datos, entre otras. Adicionalmente, en esta fase se obtiene la definición de la capa de acceso a datos y el servicio de descarga de la capa de servicios del framework el cual se encarga de la orquestación, de la descarga y almacenamiento de los datos. Además, se presenta el componente de ingreso de URL's que pertenece a la capa de aplicación del framework al igual que el servicio de ingreso de URL's perteneciente a la capa de servicios del framework.

Por tanto, para el presente trabajo esta fase se realizaron actividades destinadas a comprender y familiarizarse con los datos, como la recolección inicial de los datos, conversión de los datos de PDF a CSV, conteo del número de páginas por cada PDFy la obtención de todos los archivos CSV.

1.5.2.1 Recolección de datos iniciales

Los datos que se desea obtener y analizar se encuentran en el sitio web oficial del Ministerio de Educación, en formato PDF, más específicamente la remuneración mensual por puesto.

Para recolectar estos datos necesitamos aplicar la técnica de extracción de datos Web Scraping, para así obtener los archivos en formato PDF y posteriormente convertirlos en archivos planos que permitan acceder a los datos para posteriormente tratarlos, analizarlos y visualizarlos.

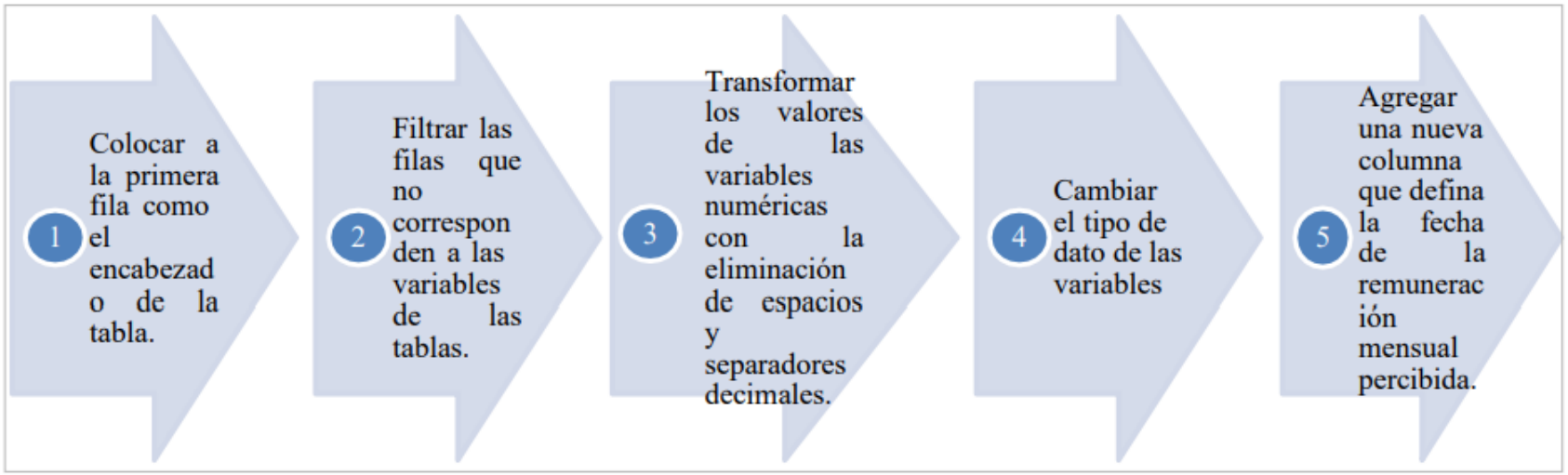
1.5.3 Fase 3: Preparación de los datos

Una vez que se conoce los datos se puede trabajar con ellos mediante tareas de limpieza y transformación, esto con el fin de facilitar su uso en tareas futuras. Esta fase conduce en la obtención del servicio de procesamiento de datos de la capa de servicios del framework el cual se encarga de procesar los datos para obtener un conjunto de estos que cumplan las características requeridas para el análisis.

Esta fase permite manipular los datos con el fin de prepararlos para el análisis principal del proyecto. La ilustración a continuación describe los pasos en que los datos fueron tratados y

depurados, para generar un conjunto de datos limpios a través de la herramienta de Power Query en Power BI.

Ilustración 1. Flujo del proceso de limpieza de datos



1.5.4 Fase 4: Modelado

Normalmente en esta fase se realiza la selección y posterior aplicación de una o varias técnicas de modelado. El problema pararesolver puede ser abordado desde diferentes perspectivas por lo que varios modelos pueden ser aplicables para su resolución, la idea es optimizar los modelos seleccionados a fin de generar buenos resultados (Rosero, 2021).

En esta fase se desarrolla un aplicativo a manera de dashboard, correspondiente a la capa de aplicación del framework, el cual se encarga de la presentación de los resultados mediante indicadores clave, como por ejemplo la relación entre el número de empleados y la remuneración anual, indicadores del número de empleados, de los ingresos adicionales, de la remuneración unificada y una ratio de la remuneración por hora. Estos nos permiten identificar ciertos criterios en base a una línea temporal.

1.5.5 Fase 5: Resultados obtenidos

Esta fase se relaciona con las etapas de demostración y comunicación de DSR, se sustenta con la publicación del trabajo realizado para este proyecto, así como los resultados obtenidos.

En esta fase se revisarán los resultados conseguidos a través de implementar la técnica de extracción de datos web scraping, como son, las variables y su tipo de datos, mientras que la visualización se realiza en Power BI.

Tabla 2. Variables resultantes

Nombre Variables	Tipo Dato
Apellidos y nombres de los servidores y servidoras	varchar
Puesto Institucional	varchar
Régimen laboral al que pertenece	varchar

Número de partida presupuestaria	varchar
Grado jerárquico o escala al que pertenece el puesto	varchar
Remuneración mensual unificada	varchar
Remuneración unificada (anual)	varchar
Décimo Tercera Remuneración	varchar
Décima Cuarta Remuneración	varchar
Horas suplementarias y extraordinarias	varchar
Encargos y subrogaciones	varchar
Total ingresos adicionales	varchar
Fecha declaración	varchar

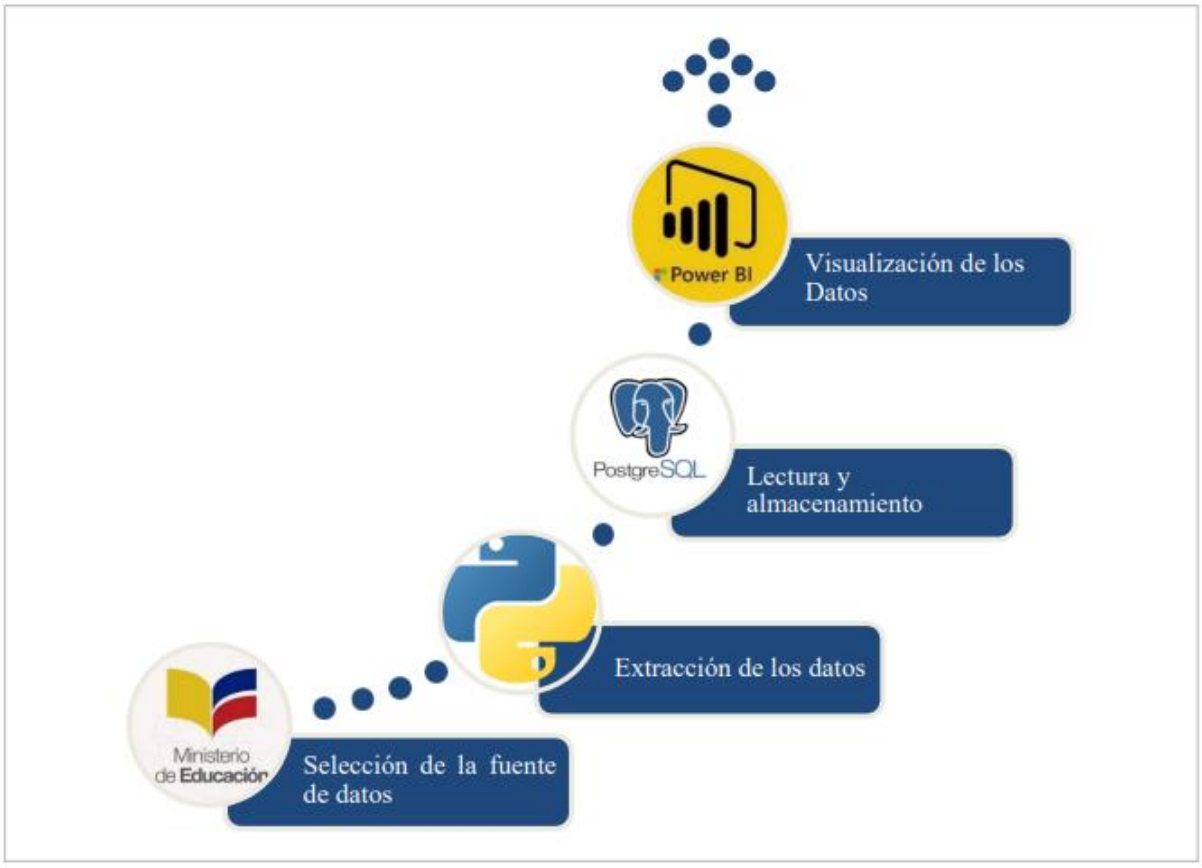
En cuanto a la visualización, el conjunto de elementos visuales que se utiliza son gráficos de líneas, gráficos circulares y cards de indicadores. Los gráficos de líneas permiten demostrar tendencias y evoluciones en una línea de tiempo, los gráficos circulares permiten visualizar porciones o secciones de la totalidad de un conjunto de datos (Ruiz, 2021) y las cards en Power BI son los elementos más recomendados para evidenciar una construcción de un indicador determinado.

CAPITULO II

2. METODOLOGÍA

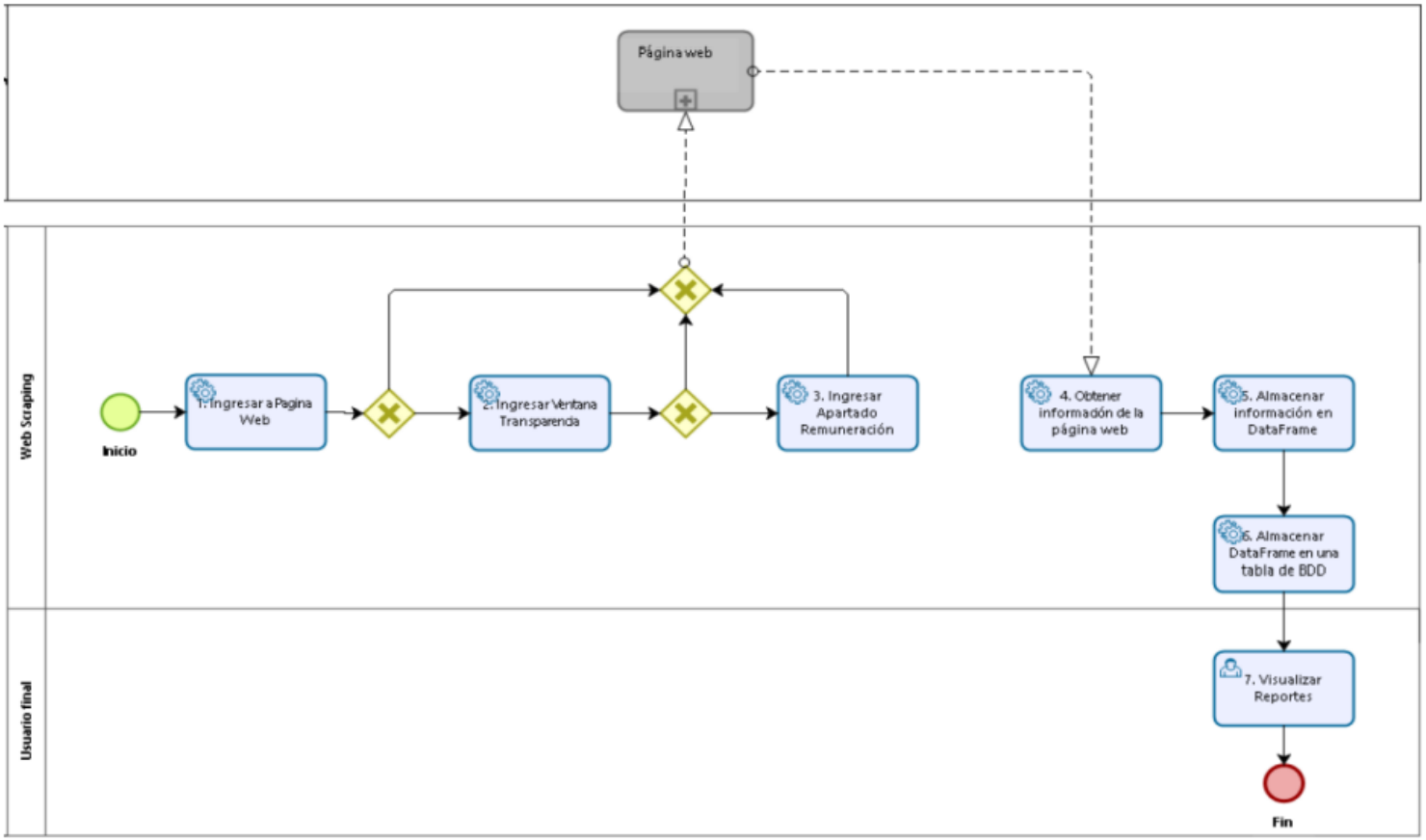
Para llevar a cabo el presente estudio se escogió un conjunto de herramientas necesarias en el desarrollo del proyecto que van desde la fase de extracción de datos hasta la visualización de los estos. El lenguaje de programación Python permitirá recopilar los datos de la página web, los archivos .CSV leídos mediante Postgres constituirán la herramienta de almacenamiento de datos y Power Bi constituirá la herramienta de visualización.

Ilustración 2. Esquema general de la metodología



El flujo de trazabilidad nos permite entender a mayor detalle la metodología de este estudio, desde el ingreso a la página web de donde se extraerán los datos, identificación del apartado de la página en donde se requiere obtener los datos, en este caso la remuneración mensual por puesto, extracción de los datos, almacenamiento, explotación y visualización de estos.

Ilustración 3. Trazabilidad Metodológica



2.1 Selección de la fuente de datos

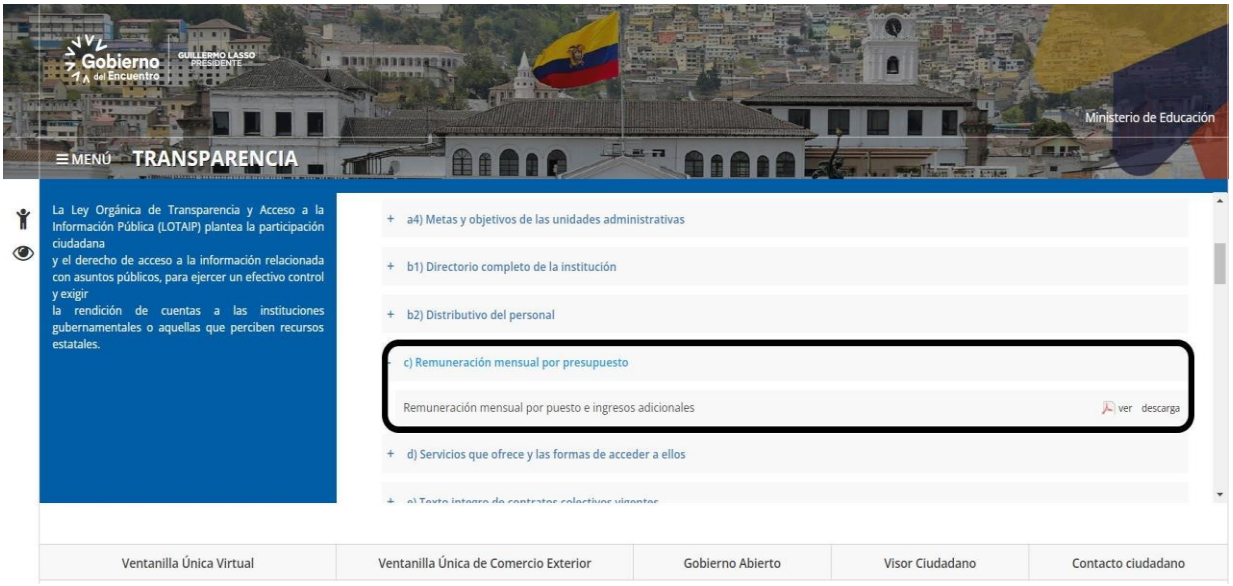
El Ministerio de Educación es la entidad con la mayor asignación presupuestaria, con un alcance de aproximadamente el 80% del monto total (Ministerio de Educación, 2021). Por lo que en este trabajo se pretende realizar un análisis de las remuneracionesmensuales de los trabajadores, información que se encuentra depositada en la página web del Ministerios de Educación.

Ilustración 4. Página Web Ministerio de Educación

Fuente: Información tomada de la página web del Ministerio de Educación.

Dada la obligatoriedad de cumplimiento de la Ley Orgánica de Transparencia y Acceso a la Información Pública en el Art.7 señala que, "por la transparencia en la gestión administrativa están obligados a observar todas las instituciones del Estado que conforman el sector público, quienes difundirán a través de un portal de información o página web, así como de los medios necesarios a disposición del público, implementados en la misma institución, la información mínima actualizada, que para efectos de esta Ley, se la considera de naturaleza obligatoria". Por tanto, el Ministerio de Educación tiene dentro de su página web, en el segmento de transparencia, un repositorio donde se encuentra toda la información mencionada en el Art. 7 de la LOTAIP, de los literales a) al t), siendo el literal c) el objeto de estudio, "c) La remuneración mensual por puesto y todo ingreso adicional, incluso el sistema de compensación, según lo establezcan las disposiciones correspondientes".

Ilustración 5. Apartado de Transparencia - Publicación de Datos



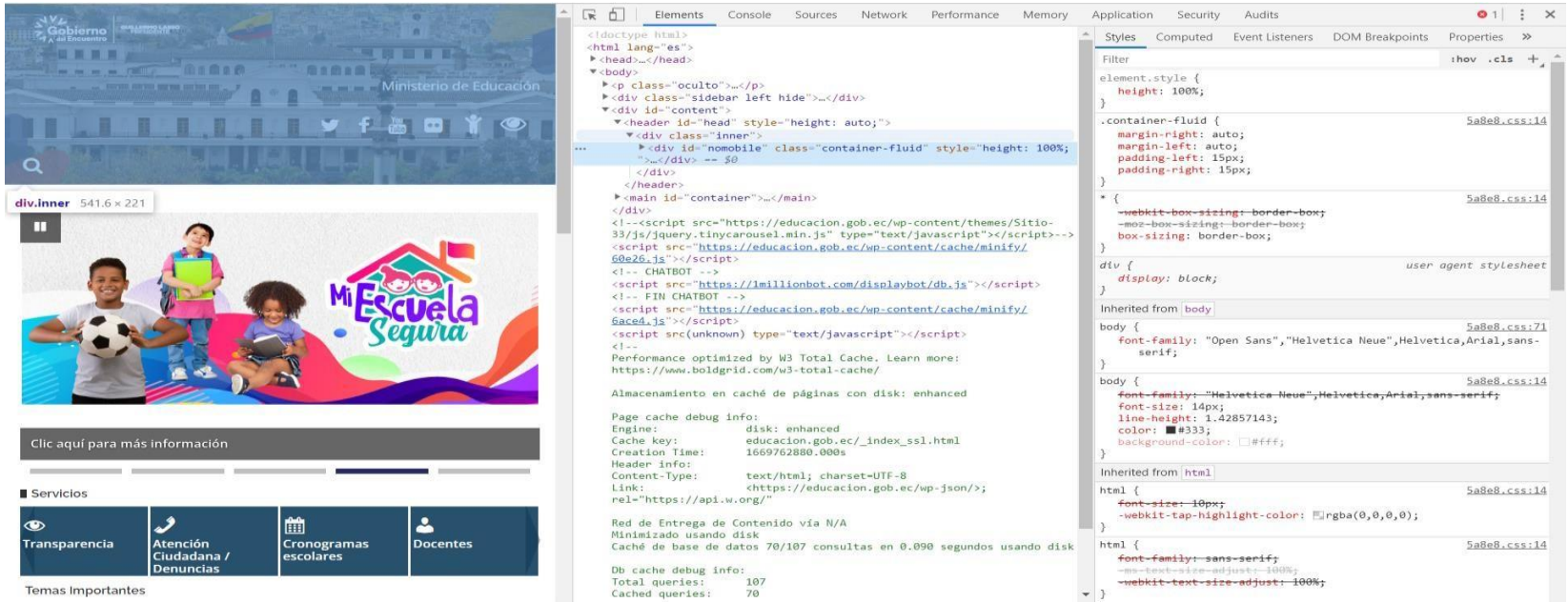
Fuente: Información tomada de la página web del Ministerio de Educación.

Para esto, la técnica Web Scraping nos permitirá realizar la extracción de datos únicamente de los reportes de interés para los periodos de este estudio.

2.2 Extracción de los datos

El primer paso radica en explorar y comprender la estructura del portal web del Ministerio de Educación para identificar los datos a extraer y encontrar patrones dentro del lenguaje de marcado. Los tipos de elementos básicos de un archivo .html son: Etiquetas; Atributos y NavigableString que sirven para encerrar diferentes partes del contenido para que se perciban y comporten de distintas maneras (Ruiz, 2021).

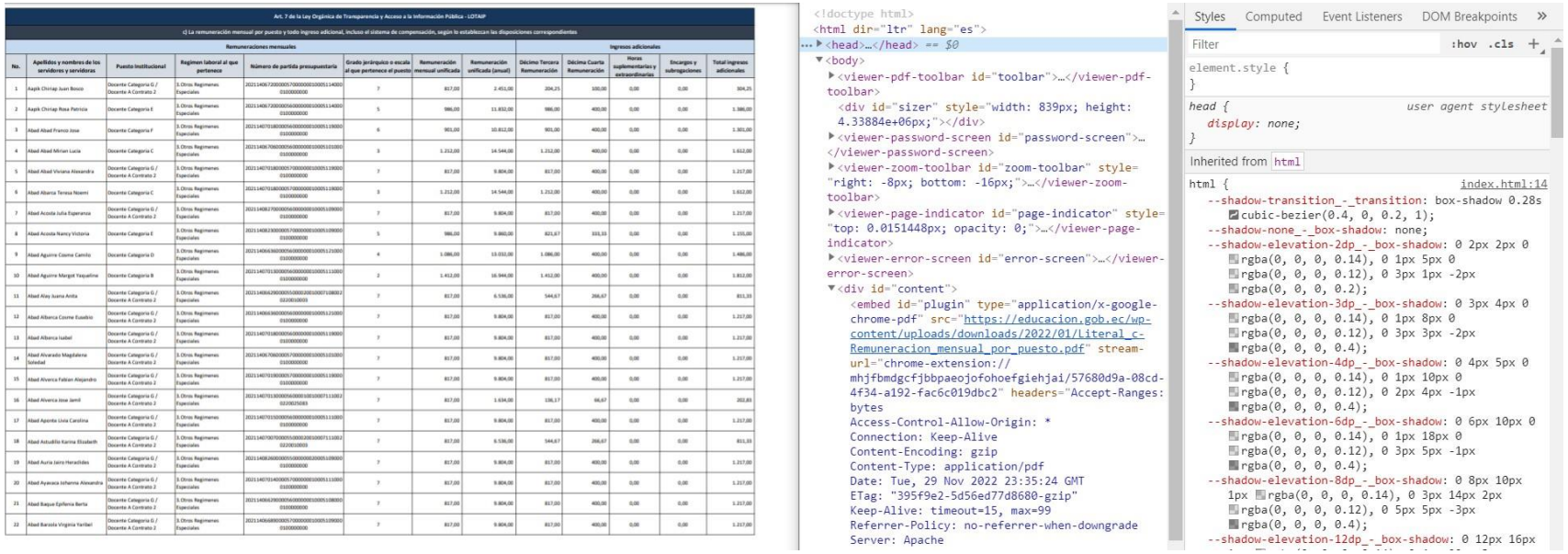
Ilustración 6. Parte del HTML de la página del Ministerio de Educación



Fuente: Información tomada de Ministerio de Educación.

Dentro del sitio web del Ministerio de Educación, en el apartado de transparencia existe información mensualizada desde el año 2015, según la disposición del Art.7 de la Ley de Transparencia y, para este caso de estudio en específico la remuneración mensual por puesto, un total de 84 pdfs, cada uno de estos presenta información de: Apellidos y nombres de los servidores y servidoras, Puesto Institucional, Régimen laboral al que pertenece, Número de partida presupuestaria, Grado jerárquico o escala al que pertenece el puesto, Remuneración mensual unificada, Remuneración unificada (anual), Décimo Tercera Remuneración, Décima Cuarta Remuneración, Horas suplementarias y extraordinarias, Encargos y subrogaciones, Total ingresos adicionales.

Ilustración 7. Parte del HTML del apartado de transparencia literal c



Fuente: Información tomada de Ministerio de Educación.

Con base en lo anterior, en esta etapa se utiliza el lenguaje de programación Python para buscar el contenido de la página web, herramienta eficaz para aplicar la técnica Web Scraping, las librerías Beautiful Soup y Request son las más recomendadas para recopilar datos eficazmente, estas y otras más como Pandas sirven para almacenar los datos extraídos, que serán las protagonistas del script a desarrollar.

Primero es necesario instalar Python, por lo que se descargó la última versión del programa desde la página oficial www.python.org/downloads/, no fue necesario instalar el gestor de librerías pip porque ya viene instalado.

Antes de proceder a desarrollar el código, se debe descargar las librerías Beautiful Soup, Requests y Pandas. En la terminal se insertaron los siguientes comandos:

- pip install beautifulsoup4
- pip install Requests
- pip install pandas

Beautifulsoup4 es una librería de Python que extrae información típicamente de archivos XML o HTML mediante el análisis del árbol de elementos con métodos como html.parser, lxml.parser y html5lib.parser.

Requests es la librería HTTP de Python que permite hacer peticiones HTTP a un servidor web a través de métodos como GET () o POST ().

Pandas es una librería de Python para el análisis de datos que proporciona una estructura flexible a los datos para que se puedan manipular eficientemente.

2.3 Pseudocódigo y Diagramas de Flujo

El programa se divide en 3 algoritmos, el primero descarga la información, el segundo lee los archivos pdf, encuentra las estructuras tipo tabla, las pasa en memoria a un DATAFRAME para posteriormente pasarlas a tablas postgres de 13 y 6 columnas, en las tablas de 6 columnas en la segunda columna se encuentra que algunos casos pueden almacenar el contenido de 7

columnas, las cuales con un tercer algoritmo son transformadas a 13 columnas según la particularidad encontrada.

Ilustración 8. Proceso Web Scraping

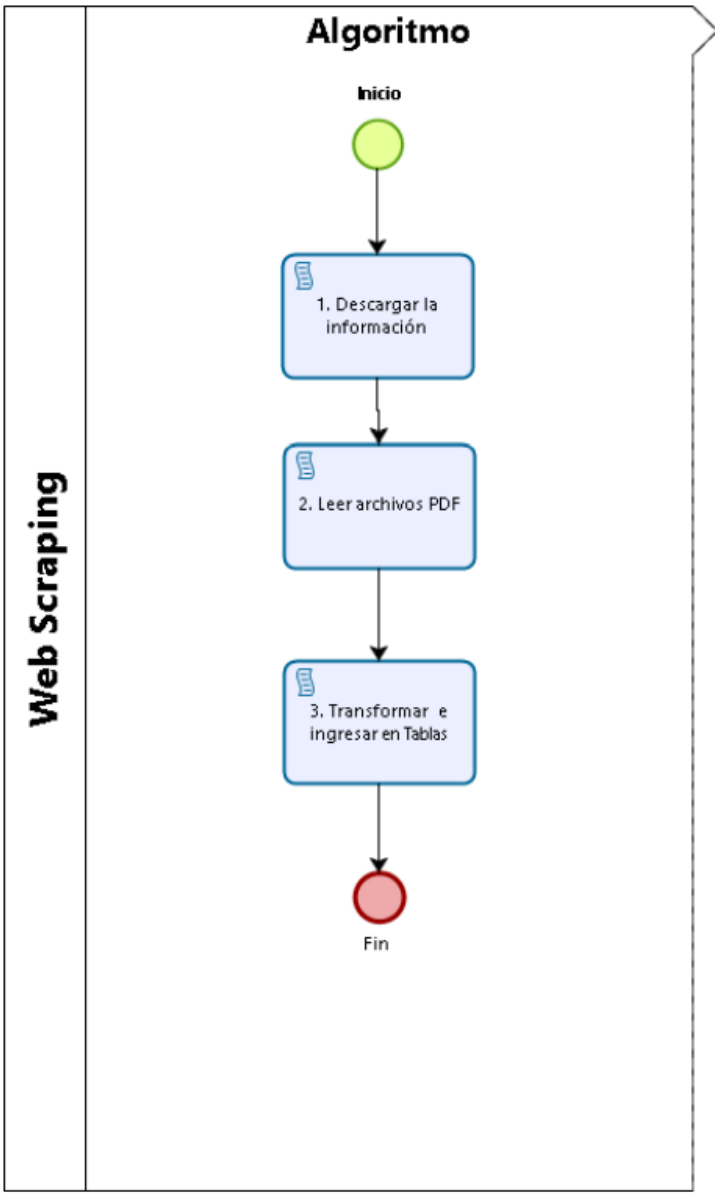
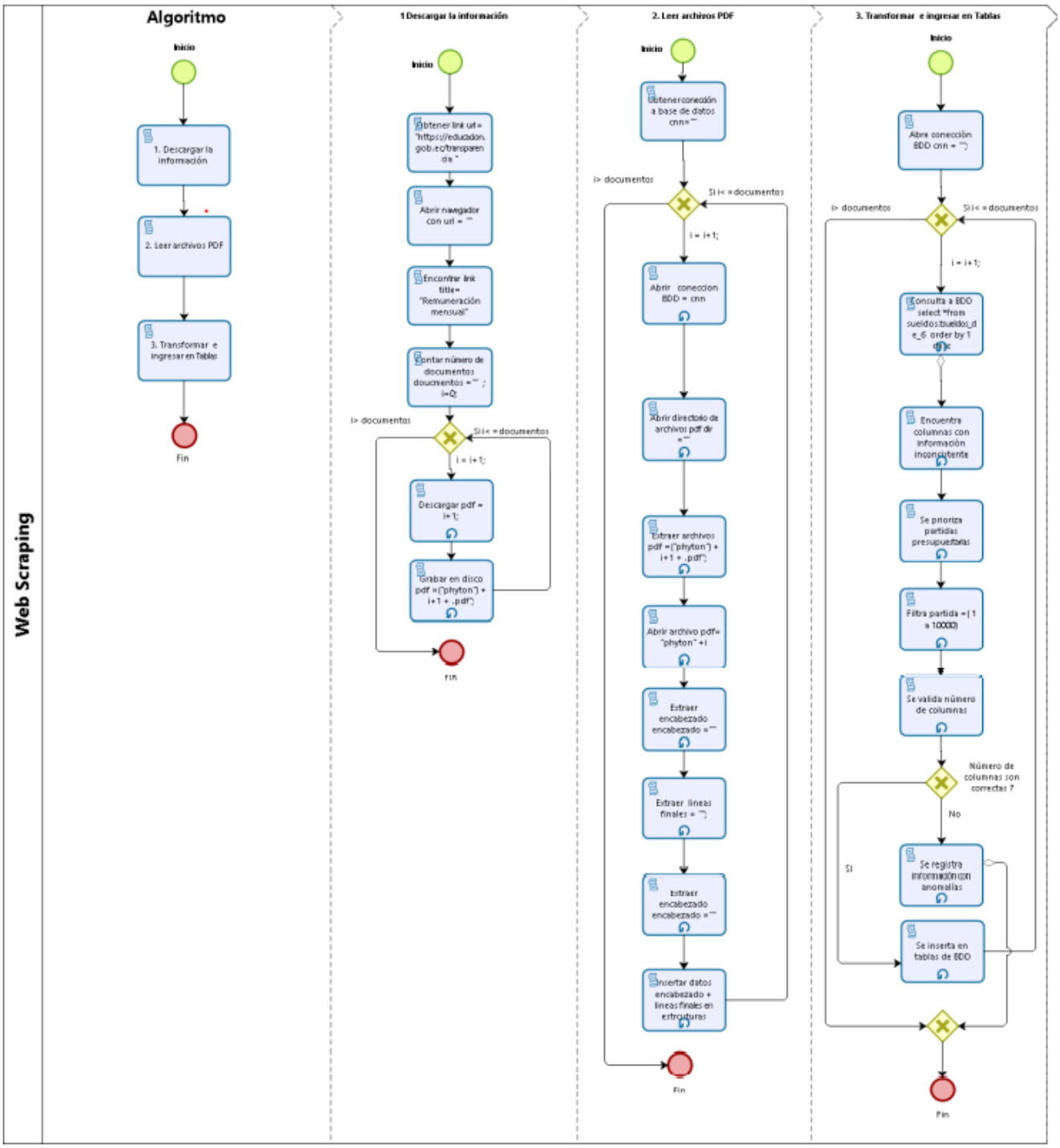


Ilustración 9. Proceso de desarrollo de los algoritmos



2.3.1 Descarga de información: Descarga de PDFs en la misma ruta en donde se ejecuta el algoritmo.

IMPORTAR LIBRERIA requests

DESDE bs4 IMPORTAR LIBRERIA BeautifulSoup direcciones = [https://educacion.gob.ec/transparencia]PARA url EN direcciones ITERAR:
pagina = PETICIONES WEB .CONSEGUIR(url)
CONTENIDOINTERNET = BeautifulSoup(page.content, 'html.parser')INICIAR prueba ES IGUAL A list()
INICIAR nombre ES IGUAL A list()
PARA enlace EN CONTENIDOINTERNET.ENCONTRAR TODO('a', class_='ver'):
a=enlace.CONSEGUIR('title')#MOSTRAR(a)
SI a.ENCONTRAR('Remuneración mensual')!=-1: prueba.AGREGAR(enlace.CONSEGUIR PARTES QUE INICIEN CON ('href'))
nombre. AGREGAR (enlace. CONSEGUIR PARTES QUE INICIEN CON('title'))
PARA i EN rango(longitud de cadena(prueba)) ITERAR:
r=PETICIONES WEB. CONSEGUIR(prueba[i], PARAMETRO stream=True)
CON ABRIR ARCIHVO (f"python{i}.pdf", "wb") COMO pdf:
PARA PARTECITA EN r.RECORRER CONTENIDO(PARAMETROTAMAÑO BLOQUES=1024):
SI PARTECITA:
pdf.ESCRIBIR (PARTECITA)

2.3.2 Procesamiento de TABLAS: Se procede a validar cada PDF, las tablas que se encuentran en las hojas del PDF se trasladan a una tabla postgres.

El presente algoritmo sirve para obtener una estructura de 13 columnas con las siguientes características:

COLUMNAS ['numero', 'apellidos_nombres', 'puesto_institucional', 'regimen_laboral', 'partida_presupuestaria', 'grado_jerarquico', 'remuneracion_mensual_unif', 'remuneracion_anual_unif', 'decimo_tercera_remuneracion', 'decimo_cuarta_remuneracion', 'horas_suplementarias', 'encargos_subrogaciones', 'total_ingresos_adicionales']

A una tabla que contiene la misma cantidad de columnas más la fecha y dirección del archivo, la tabla es en una base de datos postgres.

```
create table sueldos.tsueldos_de_13
(
numero                    varchar(320), apellidos_nombres
                          varchar(256), puesto_institucional
                          varchar(510), regimen_laboral                varchar(512),
partida_presupuestaria       varchar(256), grado_jerarquico            varchar(256),
remuneracion_mensual_unif varchar(320),remuneracion_anual_unif
                          varchar(320), decimo_tercera_remuneracion varchar(320),
decimo_cuarta_remuneracion varchar(320),horas_suplementarias
                          varchar(320), encargos_subrogaciones
                          varchar(320), total_ingresos_adicionales varchar(320), archivopdf
                           varchar(512),
fechadeclaracion             varchar(32)
);
```

```
alter table sueldos.tsueldos_de_13 owner to postgres;"""
#Aquí se importan las librerías IMPORTAR LIBRERIA tabula
IMPORTAR LIBRERIA pandas COMO pd       para manejo de matrices
DESDE sqlalchemy IMPORTAR LIBRERIA CREAR CONEXION A BASE DEDATOS
DESDE sqlalchemy.pool IMPORTAR LIBRERIA  QueuePoolDESDE io IMPORTAR
LIBRERIA StringIO
IMPORTAR LIBRERIA csv,os manejar archivos CSV IMPORTAR LIBRERIA os.path
manejo de archivos DESDE os IMPORTAR LIBRERIA path
DESDE glob IMPORTAR LIBRERIA glob
IMPORTAR LIBRERIA logging manejo de log IMPORTAR LIBRERIA PyPDF2  manejo de
archivos PDF
IMPORTAR LIBRERIA re  para manejar expresiones regulares

#patrón para encontrar fechas en el formato dd/mm/yyyy
patron=re.compile(r'\d{2}/\d{2}/\d{4}')
INICIAR logger  ES IGUAL A  logging.getLogger("PyPDF2")
logger.setLevel(logging.ERROR)
#unicamente pone el coloros.system("")
DEFINIR CLASE style():
INICIAR BLACK ES IGUAL A '\033[30m' INICIAR RED ES IGUAL A '\033[31m'
INICIAR GREEN ES IGUAL A '\033[32m' INICIAR YELLOW ES IGUAL A  '\033[33m'
INICIAR BLUE ES IGUAL A  '\033[34m'
INICIAR MAGENTA ES IGUAL A '\033[35m'INICIAR CYAN ES IGUAL A '\033[36m'
INICIAR WHITE ES IGUAL A '\033[37m' INICIAR UNDERLINE ES IGUAL A '\033[4m'
INICIAR RESET ES IGUAL A  '\033[0m'
```

```
#posibles encabezados
encabezado13= ['No.', 'Apellidos y nombres de los servidores y servidoras', 'Puesto
Institucional', 'Regimen laboral al que pertenece', 'Numero de partida
presupuestaria', 'Grado jerarquico o escala al que pertenece el puesto',
'Remuneración mensual unificada', 'Remuneración unificada (anual)', 'DécimoTercera
Remuneración', 'Décima Cuarta Remuneración', 'Horas suplementarias yextraordinarias',
'Encargos y subrogaciones', 'Total ingresos adicionales'] encabezado13a= ['numero',
                'apellidos_nombres', 'puesto_institucional','regimen_laboral',
                'partida_presupuestaria', 'grado_jerarquico',
'remuneracion_mensual_unif', 'remuneracion_anual_unif',
'decimo_tercera_remuneracion', 'decimo_cuarta_remuneracion',
'horas_suplementarias', 'encargos_subrogaciones', 'total_ingresos_adicionales']encabezado6a=
['numero', 'apellidos_nombres', 'area_depedencia', 'cargo', 'remuneracion', 'grado']
encabezado4a= ['numero', 'cargo', 'partida', 'remuneracion'] encabezado12=['Apellidos y
nombres de los servidores y servidoras', 'PuestoInstitucional', 'Regimen
                laboral al que pertenece', 'Número de partida
presupuestaria', 'Grado jerárquico o escala al que pertenece el puesto',
'Remuneración mensual unificada', 'Remuneración unificada (anual)', 'DécimoTercera
Remuneración', 'Décima Cuarta Remuneración', 'Horas suplementarias yextraordinarias',
'Encargos y subrogaciones', 'Total ingresos adicionales']

#cadenas de conexión para poder conectarse a la base de datos y guardar los datos INICIAR
motor ES IGUAL A CREAR CONEXIÓN A LA BASE DE DATOS
(f'postgresql+pg8000://postgres:jdfoxito10@localhost:5432/cell', echo=False,
poolclass=QueuePool)

#ruta donde deben estar los PDF en formato descrito en el encabezado de estealgoritmo
INICIAR directorio ES IGUAL A r"D:\SRI\bigdata\pdf"

#trae un dato único de la base de datos
DEFINIR PROCESO CONSEGUIR DATO UNICO (sql):INICIAR valor COMO VACIO
INICIAR results ES IGUAL A motor.ejecutar(sql)CAPTURAR:
SI resultados.conteodefilas > 0:
   INICIAR valor ES IGUAL A resultados.primero()[0]CASO CONTRARIO:
      INICIAR valor ES IGUAL A " EXCEPCION: EXCEPCION COMO ex:

valor="
RETORNAR valor

#inserta una fila en la tabla de la base de datosDEFINIR PROCESO get_insertar(sql):
INICIAR respuesta ES IGUAL A -1
CON motor.connect().execution_options(autocommit=True) COMO conn:INICIAR retorno ES
IGUAL A conn.execute(str(sql))
                                                SI (retorno.rowcount > 0):
INICIAR respuesta ES IGUAL A retorno.fetchone()[0]
RETORNAR respuesta
```

```
    #traer los archivos de tipo PDF
  DEFINIR PROCESO fx_traer_archivos_tipo(dir):lista= []
    CAMBIAR AL DIRECTORIO dir
    INICIAR extensiones ES IGUAL A ['pdf']PARA ext EN extensiones:
        lista+=glob(UNIR TUTA EN(dir,f"*.{ext}"))RETORNAR lista

  DEFINIR PROCESO andromeda(listaPDF):#inicializacion de arreglos
    INICIAR listaHojas13 ES IGUAL A []INICIAR listaHojas6 ES IGUAL A [] INICIAR
    listaHojas4 ES IGUAL A []
    INICIAR numeroarchivos ES IGUAL A len(listaPDF)INICIAR numarchs ES IGUAL A 0
    INICIAR encontrado ES IGUAL A  "IMPORTAR LIBRERIA timeit
    INICIAR t_0  ES IGUAL A  timeit.default_timer()INICIAR tabla13 ES IGUAL A
    'tsueldos_de_13' INICIAR tabla6 ES IGUAL A 'tsueldos_de_6' INICIAR tabla4  ES IGUAL
    A  'tsueldos_de_4' #se recorren todos los pdf dentro del directorio PARA archivopdf  EN
    listaPDF:
    numarchs+=1
    INICIAR listaHoja  ES IGUAL A  []
    MOSTRAR(f" {style.MAGENTA}  Inicio del procesamiento de {archivopdf}
    {style.RESET}")
        INICIAR  cnsulta13  ES  IGUAL  A  f"""  select  count(1)  DESDE
    sueldos.{tabla13} where archivopdf  ES IGUAL A  '{archivopdf}'; """
        INICIAR cnsulta6 ES IGUAL A f""" select count(1) DESDE sueldos.{tabla6}where
    archivopdf ES IGUAL A  '{archivopdf}'; """
        SI  get_scalar(cnsulta13)  EQUIVALENTE  A  0  and  get_scalar(cnsulta6)
    EQUIVALENTE A 0:
    INICIAR pdfFileObj ES IGUAL A open(archivopdf, 'rb')
    INICIAR  pdfReader  ES  IGUAL  A  PyPDF2.PdfFileReader(pdfFileObj)  INICIAR
    numeroPaginas ES IGUAL A pdfReader.numPages
          INICIAR  ldf  ES  IGUAL  A  tabula.read_pdf(archivopdf,
    multiple_tables=True,pages=f"{numeroPaginas}",  encoding  ES  IGUAL  A  "ISO- 8859-1",
    stream=True)
    CAPTURAR:
    INICIAR listaHoja ES IGUAL A tabula.read_pdf(archivopdf, pages=f'1-
  {numeroPaginas}', encoding ES IGUAL A "ISO-8859-1")EXCEPCION: EXCEPCION COMO ex:
    INICIAR listaHoja ES IGUAL A tabula.read_pdf(archivopdf, pages=f'1-
  {numeroPaginas}', encoding  ES IGUAL A  "ISO-8859-1",stream=True)SI len(listaHoja)>0:
    MOSTRAR  (f"  {style.CYAN}  {numarchs}  de  {numeroarchivos}
{style.RESET} archivo en curso {style.GREEN} {archivopdf} {style.RESET} ")INICIAR ihoja ES
    IGUAL A  0
  PARA dfHoja EN listaHoja:ihoja+=1
    SI ihoja ==1:
    SI len(ldf)>0:
      INICIAR df ES IGUAL A  pd.concat(ldf)PARA ix, fila EN  df.iterrows():
    INICIAR cadena ES IGUAL A  fila.to_string()
    INICIAR  sihay  ES  IGUAL  A
    re.findall(r"[\d]{1,2}/[\d]{1,2}/[\d]{4}", cadena)
```

SI len(sihay)>0:
INICIAR encontrado ES IGUAL A ".join(sihay)SI (len(dfHoja.columns.tolist()) EQUIVALENTE A 13):
#en la primera hoja se recorta el encabezadoSI ihoja ==1:
INICIAR dfHoja ES IGUAL A dfHoja.iloc[5:]#se reemplaza el encabezado
INICIAR dfHoja.columns ES IGUAL A encabezado13a#en la última página se recortan las filas finales
SI ihoja >= numeroPaginas:
INICIAR dfHoja ES IGUAL A dfHoja.iloc[:-10]
#se escogen las columnas con el tamanho necesario, evitando sefiltre tomar filas resumen, o totales
CAPTURAR:
INICIAR dfHoja ES IGUAL A dfHoja.loc[dfHoja['numero'].str.len() <= 30]
listaHojas13.AGREGAR(dfHoja) EXCEPCION: EXCEPCION COMO ex:
MOSTRAR (ex)

#misma lógica que el bloque anterior, pero para 6 columnasSI (len(dfHoja.columns.tolist()) EQUIVALENTE A 6):
SI ihoja ==1:

INICIAR dfHoja ES IGUAL A dfHoja.iloc[6:] INICIAR dfHoja.columns ES IGUAL A encabezado6a
INICIAR dfHoja ES IGUAL A dfHoja.loc[dfHoja['numero'].str.len() <= 30]
listaHojas6.AGREGAR(dfHoja)
#misma logica que el blqoue anterior, pero para 4 columnasSI (len(dfHoja.columns.tolist()) EQUIVALENTE A 4):
SI ihoja ==1:
INICIAR dfHoja ES IGUAL A dfHoja.iloc[6:] INICIAR dfHoja.columns ES IGUAL A encabezado4alistaHojas4.AGREGAR(dfHoja)

#Concatenado de segmentos dataframe en un soloINICIAR dfHojas ES IGUAL A pd.DataFrame() SI len(listaHojas13)>0:
INICIAR dfHojas ES IGUAL A pd.concat(listaHojas13)INICIAR tablaturno ES IGUAL A tabla13
SI len(listaHojas6)>0:
INICIAR dfHojas ES IGUAL A pd.concat(listaHojas6)INICIAR tablaturno ES IGUAL A tabla6
SI not dfHojas.empty:
INICIAR dfHojas["archivopdf"] ES IGUAL A archivopdf INICIAR dfHojas["fechadeclaracion"] ES IGUAL A encontrado #en el caso de requerir esete en formato excel #dfHojas.to_excel(rf"D:\SRI\bigdata\excel\excel{numarchs}.xlsx", index=False)
CON motor.connect() COMO conn, conn.begin(): dfHojas.to_sql(tablaturno, con=conn, if_exists='append',index=False,

chunksize=100000, schema='sueldos')
INICIAR t_1 ES IGUAL A timeit.default_timer() INICIAR elapsed_time ES IGUAL A round ((t_1 - t_0), 3)
MOSTRAR (f" {style.CYAN} {archivopdf} se ha procesado en : {elapsed_time} s {style.RESET}")CASO CONTRARIO:
MOSTRAR (f"Se encuentra cargado {archivopdf}")

INICIAR t_total ES IGUAL A timeit.default_timer() INICIAR elapsed_time ES IGUAL A round ((t_total - t_0), 3)
MOSTRAR (f" {style.GREEN} se ha procesado todo en : {elapsed_time} s {style.RESET}")
#se ejecuta la función principal andromeda(fx_traer_archivos_tipo(directorio))

2.3.3 Depuración de tablas postgres: Se realiza reubicación de datos que están en otras columnas, a sus columnas propias

IMPORTAR LIBRERIA pandas COMO pd

DESDE sqlalchemy IMPORTAR LIBRERIA CREAR CONEXION A BASE DEDATOS
DESDE sqlalchemy.pool IMPORTAR LIBRERIA QueuePool
INICIAR motor ES IGUAL A create_engine(f'postgresql+pg8000://postgres:jdfoxito10@localhost:5432/cell', echo=False, poolclass=QueuePool)

DEFINIR PROCESO fx_get_df(engine, consulta):
INICIAR df_oracle ES IGUAL A pd.read_sql(consulta, engine)
RETORNAR df_oracle

#inserta un valor en la base de datos DEFINIR PROCESO get_insertar(sql):
INICIAR respuesta ES IGUAL A -1
CON motor.connect().execution_options(autocommit=True) COMO conn:INICIAR retorno ES IGUAL A conn.execute(str(sql)) if(retorno.rowcount > 0):
INICIAR respuesta ES IGUAL A retorno.fetchone()[0]RETORNAR respuesta

INICIAR df23mill ES IGUAL A fx_get_df(motor, "select *from sueldos.tsueldos_de_6 order by 1 desc")
INICIAR nginx ES IGUAL A 0
INICIAR nginxTotal ES IGUAL A len(df23mill.index)PARA ix, filas EN df23mill FILAS ITERAR:
INICIAR cadena ES IGUAL A filas["area_depedencia"] INICIAR nombres ES IGUAL A filas["apellidos_nombres"] INICIAR nombres ES IGUAL A nombres. REEMPLAZAR ("","")
INICIAR cadena ES IGUAL A cadena + ' '
INICIAR elementos ES IGUAL A cadena. DIVIDIR (' ')INICIAR columnasX ES IGUAL A

```
   []
SI LONGITUD (elementos) > 15: INICIAR posicion  ES IGUAL A  0INICIAR pivot  ES
   IGUAL A  0
   INICIAR acumular ES IGUAL A " INICIAR bandera  ES IGUAL A  FalsoINICIAR pasado
   ES IGUAL A Falso INICIAR columnas ES IGUAL A [] PARA ele  EN  elementos:
   SI ele.isnumeric():
   INICIAR valor  ES IGUAL A  int(ele)
SI valor > 100  Y  valor < 9999 Y  no pasado:INICIAR pivot ES IGUAL A  valor bandera
   =VERDADERO
   INICIAR pasado  ES IGUAL A  VERDADERO
   SI bandera:
   columnas. AGREGAR (acumular. LIMPIAR VACIOS)

   SI ele== '-':
   ele='0'
SI not bandera: acumular+= ele + ' '
   CASO CONTRARIO:
      acumular= eleposicion+=1
   INICIAR num  ES IGUAL A  0
   longitud=len(columnas)acumula="
PARA nano  EN  columnas ITERAR:SI num==0:
   INICIAR position  ES IGUAL A  nano.find('-')SI position>0:
   INICIAR a ES IGUAL A  nano[0:position-2] INICIAR b  ES IGUAL A  nano[position-
   1:len(nano)]columnasX.AGREGAR(a)
         columnasX.AGREGAR(b)SI longitud ==8:
SI num < 3 and num > 0: columnasX.AGREGAR(nano)
   SI num EQUIVALENTE A 3:
   acumula= nano
   SI num EQUIVALENTE A 4:
   acumula+= nano

      columnasX.AGREGAR(acumula)SI num >4:
         columnasX.AGREGAR(nano)CASO CONTRARIO:
SI num > 0: columnasX.AGREGAR(nano)
   num+=1
SI len(columnasX) EQUIVALENTE A 8:columnasX.insert(0,nombres)
   columnasX.insert(0,filas["numero"]) INICIAR grado ES IGUAL A  "
SI filas["grado"] EQUIVALENTE A '-':INICIAR grado  ES IGUAL A  '0'
   columnasX.AGREGAR(") columnasX.AGREGAR(") columnasX.AGREGAR(grado)
   columnasX.AGREGAR(filas["archivopdf"])
   columnasX.AGREGAR(filas["fechadeclaracion"])
   CASO CONTRARIO:
   columnasX.AGREGAR(filas["numero"])columnasX.AGREGAR(nombres)
   columnasX.AGREGAR(filas["cargo"]) columnasX.AGREGAR(filas["area_depedencia"])
   columnasX.AGREGAR(") columnasX.AGREGAR(filas["grado"])
   columnasX.AGREGAR(filas["remuneracion"]) #SE AGREGAN 6 columnas vacias
```

columnasX.AGREGAR(filas["archivopdf"])
columnasX.AGREGAR(filas["fechadeclaracion"])
Se igualan las columnas a 13,
SI longitud (columnasX) > 13:nginx+=1
INICIAR consulta ES IGUAL A f"""insert into sueldos.tsueldos_de_13_n2 values('{columnasX[0]}','{columnasX[1]}','{columnasX[2]}','{columnasX[3]}','{c olumnasX[4]}','{columnasX[5]}','{columnasX[6]}',

'{columnasX[7]}','{columnasX[8]}','{columnasX[9]}','{columnasX[10]}','{colum nasX[11]}','{columnasX[12]}','{columnasX[13]}','{columnasX[14]}') RETORNARing 1; """

LLAMAR A FUNCION get_insertar(consulta)MOSTRAR(f" {nginx} de {nginxTotal} ")

Con el proceso que antecede, la cantidad de variables obtenidas fueron 13, las cuales describen la remuneración mensual percibida por cada servidor público perteneciente al Ministerio de Educación desde el mes de enero del año 2015 al mesde octubre del año 2022.

Tabla 3. Variables y Tipo de Datos

Nombre Variables	Tipo Dato
Apellidos y nombres de los servidores y servidoras	varchar
Puesto Institucional	varchar
Régimen laboral al que pertenece	varchar
Número de partida presupuestaria	varchar
Grado jerárquico o escala al que pertenece el puesto	varchar
Remuneración mensual unificada	varchar

Remuneración unificada (anual)	varchar
Décimo Tercera Remuneración	varchar
Décima Cuarta Remuneración	varchar
Horas suplementarias y extraordinarias	varchar
Encargos y subrogaciones	varchar
Total ingresos adicionales	varchar
Fecha declaración	varchar

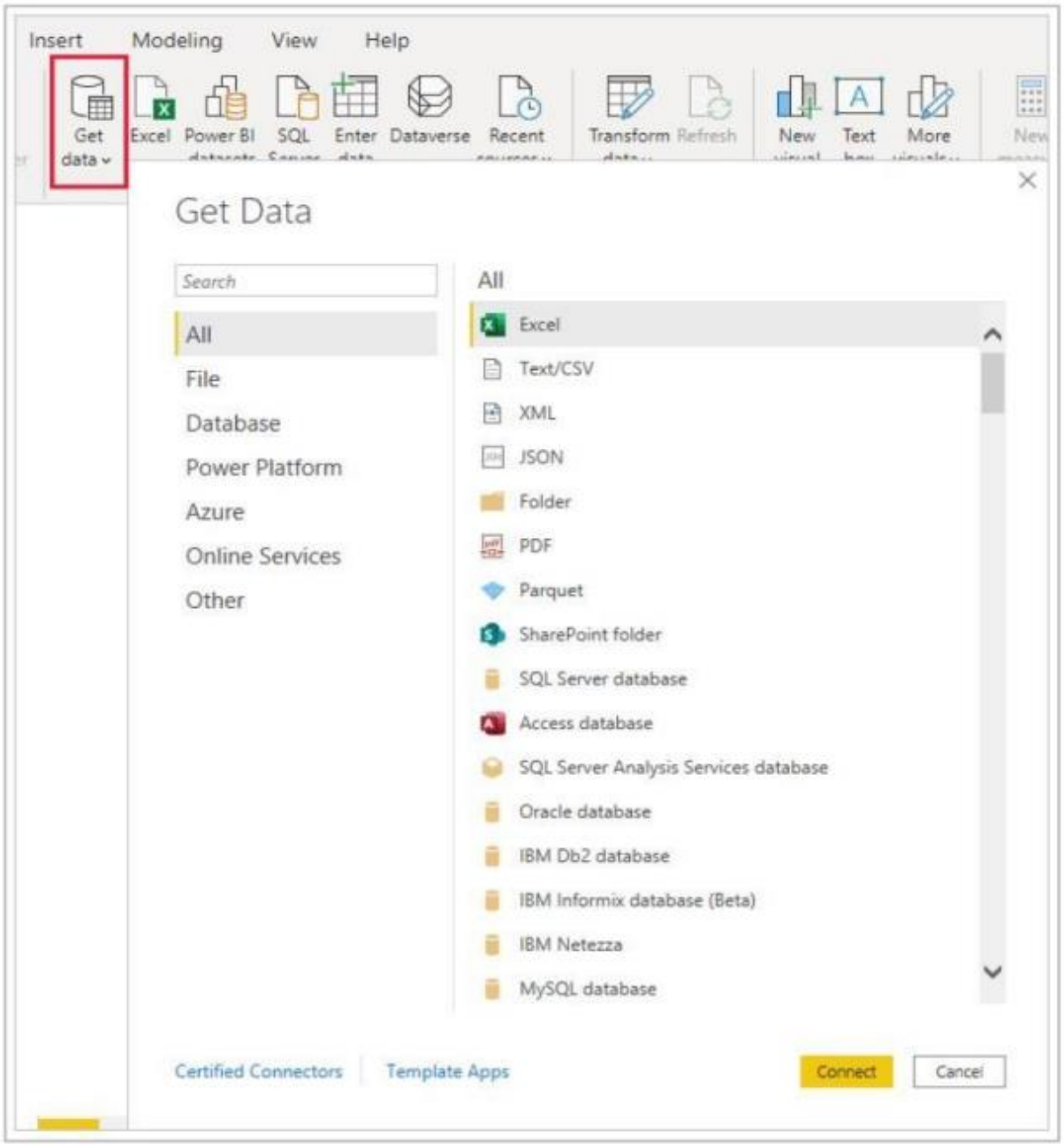
2.4 Visualización de datos

La visualización de datos es la etapa final de la metodología propuesta, para este último paso se consideró a Power BI como herramienta de visualización. Se seleccionó esta herramienta porque permite crear tableros de mando visualmente atractivos y dinámicos y también porque posee una versión gratuita lo suficientemente poderosa ya que ofrece todos los servicios Pro en un tiempo de duración de 2 meses, con el registro a través de una cuenta institucional u organizacional.

2.4.1 Conexión con origen de datos

Power BI Desktop permite conectar diferentes orígenes de datos de la Suite Office, entre ellas Base de Datos PostgreSQL, en este proceso se importan los datos extraídos del apartado de transparencia de la página web del Ministerio de Educación con el formato de tabla que permite administrar y analizar datos relacionados.

Ilustración 10. Obtener datos



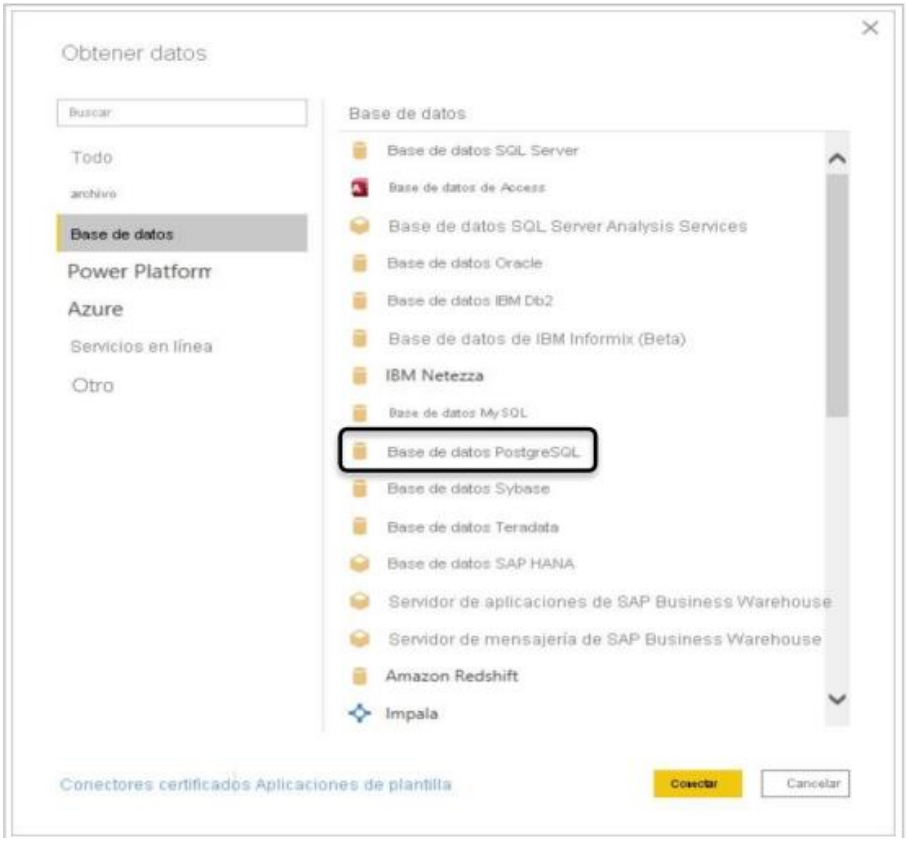
Información tomada de: Power Bi.

En el cuadro de diálogo Obtener datos se organizan los tipos de datos en las categorías siguientes:

- Todo
- Archivo
- Base de datos
- Power Platform
- Azure
- Servicios en línea
- Otros

La categoría Base de datos proporciona la conexión de datos a Base de datos PostgreSQL.

Ilustración 11. Obtener datos para Base de Datos



Información tomada de: Power Bi.

2.4.2 Diseño del dashboard

Con el conjunto de datos importado en Power BI se procede a diseñar el dashboard que permitirá explorar visualmente los datos, mediante objetos visuales creados y organizados sobre un área del lienzo de la vista Informe. Cada uno de estos objetos visuales puede contener uno o varios campos de la tabla importada mediante la acción arrastrar y soltar (Ruiz, 2021).

Ilustración 12. Pasos aplicados para la preparación de datos en Power BI

El conjunto de elementos visuales que se utiliza son gráficos de líneas, cards de indicadores y gráficos circulares. Los gráficos de líneas permiten demostrar la tendencia de las remuneraciones percibidas por funcionario, cargo y régimen en una línea de tiempo; se genera 1 gráfico de este

tipo, para visualizar la relación entre número de empleados vs la remuneración anual.

El uso más común del grafico circular es el de visualizar porciones o secciones de la totalidad de un conjunto de datos (Ruiz, 2021), dentro del dashboard se destinó 3 de estos gráficos para presentar los distintos regímenes laborales, grado jerárquico y puesto institucional.

Las cards en Power BI son los elementos más recomendados para evidenciar una construcción de un indicador determinado, dentro del dashboard se destinó 4 cards para presentar el número de empleados por mes y año, el total de ingresos adicionales por mes y año, la remuneración unificada anual de los servidores por mes y año y el promedio de remuneración anual por hora.

CAPITULO III

3. RESULTADOS

Ilustración 13. Dashboard para análisis de remuneraciones de los funcionarios del Ministerio de Educación

En el dashboard se puede observar la relación entre el número de empleados y la remuneración anual, siendo esta directamente proporcional, sin embargo, a partir del año 2019 al 2020 hay un incremento leve del número de empleados, mientras que la remuneración en este periodo presenta un pico, con lo cual se podría inferir que hubo un incremento en la remuneración de estos. A pesar de ello, se evidencia un decrecimiento tanto del número de empleados como de la remuneración, atribuyendo esto al efecto pandemia.

Adicionalmente, se presentan los indicadores del número de empleados, de los ingresos adicionales, de la remuneración unificada y una ratio de la remuneración por hora. Estos nos permiten identificar ciertos criterios en base a una línea temporal.

A manera de resumen, en los gráficos de pastel podemos observar que el mayor se encuentra concentrado en el nivel 7 y que el puesto institucional corresponde al cargo de Docente Categoría G.

En cuanto a la técnica estadística seleccionada para medir el tamaño del personal que labora en el Ministerio de Educación y la relación que existe con la inversión a dicha entidad, es la estadística descriptiva, que tiene como finalidad última resumir la información de conjuntos más o menos numerosos de datos.

El punto de partida de la estadística descriptiva es la recolección de los datos, paso ejecutado con el web scraping, como punto seguido está el tratamiento de datos, paso ejecutado con el algoritmo 2 y 3 antes detallado y la presentación de información mediante tablas y gráficos lo tenemos a continuación:

Tabla 4. Inversión del estado en el Sector de Educación

Año	Inversión Sector Educación	Variación
2015	4.188.946.027,91	
2016	4.300.349.139,71	2,66%
2017	4.697.965.582,41	9,25%
2018	4.772.233.401,08	1,58%
2019	4.769.067.215,20	-0,07%
2020	4.235.938.064,69	-11,18%
2021	4.177.238.138,14	-1,39%

Ilustración 14. Inversión Sector Educación

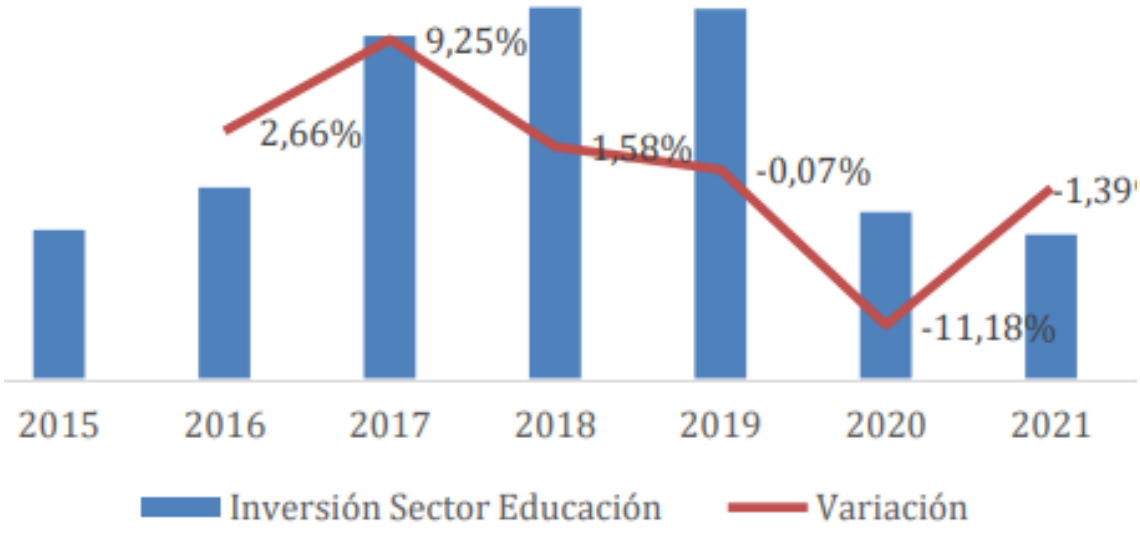


Tabla 5. Tamaño del personal Ministerio Educación

Año	No de servidores Ministerio de Educación	Variación
2015	250.880	
2016	233.540	-7,00%
2017	188.686	-19,00%
2018	181.636	-4,00%
2019	181.636	0,00%
2020	181.646	0,00%
2021	173.060	-5,00%

Ilustración 15. Tamaño del personal

Por tanto, la relación que existe entre el tamaño del personal que labora en el Ministerio de Educación y la inversión a dicha entidad es directamente proporcional. Se observa que para el año 2021 tanto la inversión como el tamaño presentan los valores más bajos dentro del periodo de estudio. En cuanto a la inversión, el año 2018 fue el de menor desembolso mientras que, en tamaño el año 2015 fue el de mayor número de servidores existió.

CAPITULO IV

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- El presente trabajo nos permite concluir que la relación entre el número de empleados del Ministerio de Educación y la remuneración anual es directamente proporcional, sin embargo, a partir del año 2019 al año 2020 hay un incremento leve del número de empleados, mientras que la remuneración en este periodo presenta un pico, con lo cual se podría inferir que hubo un incremento en la remuneración de estos.

- La técnica Web Scraping ayudó a la extracción rápida de información con estructura HTML. El algoritmo diseñado para este proyecto es efectivo y se encuentra optimizado, sin embargo, las técnicas de conversión para el almacenamiento de la información dentro de los archivos PDF no precisa un procesamiento eficaz, puesto que los archivos de los reportes de las remuneraciones son demasiado extensos y su formato no se encuentraestandarizado, lo que incurre en ciertos errores de la ejecución, que son solventados a costo del incremento del uso de los recursos.

- La relación que existe entre el tamaño del personal que labora en el Ministerio de Educación y la inversión a dicha entidad es directamente proporcional. Se observa que para el año 2021 tanto la inversión como el tamaño presentan los valores más bajos dentro del periodo de estudio. En cuanto a la inversión, el año 2018 fue el de menor desembolso, mientras que, en tamaño, el año 2015 fue el de mayor número de servidores existió.

4.2 Recomendaciones

- Utilizar técnicas de extracción de datos automatizadas que permitan el seguimiento continuo y proactivo del tamaño del personal en las entidades públicas.

- Extraer la información disponible de remuneración por puesto de todas las instituciones públicas, ya que sus páginas web mantienen la misma estructura, únicamente modificando en el web scraping desarrollado en este trabajo de investigación las direcciones url de cada entidad.

- Implementar estadística prescriptiva que permita predecir el incremento o disminución del tamaño del personal del Ministerio de Educación para los siguientes 3 años.

- Desarrollar un dashboard en donde los datos puedan ser actualizados en tiempo real para revelar en los años siguientes el comportamiento del personal del Ministerio de Educación en relación con el presupuesto destinado al sector, con el fin de controlar el tamaño del personal.

REFERENCIAS

Park, Y., & Jo, I.-H. (2015). Development of the Learning Analytics Dashboard to Support Students' Learning Performance. *Journal of Universal Computer Science*. Obtenidode https://dspace.ewha.ac.kr/bitstream/2015.oak/230480/1/001.pdf

Breuss, M. (2021). *Real Pyhton*. Recuperado el agosto de 2022, de https://realpython.com/beautiful-soup-web-scraper-python/

Glez-Peña, D., Lourenço, A., López-Fernández, H., Reboiro-Jato, M., & Fdez-Riverola, F. (2014). *Web scraping technologies in an API world* (Vol. 15). Briefings in Bioinformatics. doi:https://doi.org/10.1093/bib/bbt026

Hernández&Gómez. (2015). *Universidad Politécnica de Chiapas.* Obtenido de https://rcs.cic.ipn.mx/2015_95/Metodologias%20para%20analisis%20politico%20utilizando%20Web%20Scraping.pdf

Imperva. (13 de JULIO de 2016). *Imperva*. Recuperado el 30 de AGOSTO de 2022, de https://www.imperva.com/learn/application-security/web-scrapingattack/?redirect=Distil

JavaTPyhton. (s.f.). Recuperado el 14 de Agosto de 2022, de https://www.javatpoint.com/tabula-python

Khabsa&Giles. (2014). The number of scholarly documents on the public web. *PloS one.*

López, J. (12 de Enero de 2018). *Web scraping*. Obtenido de Academia Accelerating the world´s research: https://d1wqtxts1xzle7.cloudfront.net/55775125/web-scraping-with-cover-page-v2.pdf?Expires=1655941002&Signature=AfVAzRNQ1FYwdp7vkyStW0dnUSEicx

FdfhMbT3GuPpf~YyGjFQD7~Z8Cie0eNEwV1QzsNdfzXk6BJODQ-YSKfTfJmsQdmUxcu3ogv0xvfFZ9Qs5wjuV8GpgA4qe~MeN34zreBwslQQoR-Bg

Martines&Rodriguez. (2019). XXV Congreso Argentino de Ciencias de la Computación. *ANÁLISIS DE TÉCNICAS DE RASPADO DE DATOS*, (págs. 1-10). Argentina.

Murillo&Saavedra. (2017). Web Scraping de los Perfiles y Publicaciones de una Afiliación en Google Scholar utilizando Aplicaciones Web e implementando un Algoritmo en R. *4to Congreso Internacional AmITIC 2017, Popayán, Colombia.*, (pág. 8).Colombia.

Pulido&Morales. (17 de Diciembre de 2021). *Researchgate*. Obtenido de https://www.researchgate.net/publication/334130228_Recuperacion_de_metadatos_e_indicadores_de_impacto_para_publicaciones_cientificas_mediante_servicios_de_Google_academico

Rosero, E. (23 de Octubre de 2021). *EPN*. Obtenido de http://bibdigital.epn.edu.ec/handle/15000/21884

RUTH, R. R. (ABRIL de 2021). *UNIVERSIDAD DE GUAYAQUIL*. Recuperado el 30 de AGOSTO de 2022, de http://repositorio.ug.edu.ec/bitstream/redug/56200/1/RUIZ%20RONQUILLO%20RUTH%20ROXANA.pdf

Sachdeva, S. (Septiembre de 2021). *Analytics Vidhya*. Recuperado el Agosto de 2022, de https://www.analyticsvidhya.com/blog/2021/09/pypdf2-library-for-working-with- pdf-files-in-python/

Saurkar, A. P. (2021). An overview on web scraping techniques. *International Journal on Future Revolution in Computer*. Obtenido de http://www.ijfrcsce.org/index.php/ijfrcsce/article/view/1529/1529

Zhao, B. (Mayo de 2017). *Web Scraping.* doi:10.1007/978-3-319-32001-4_483-1

ANEXOS

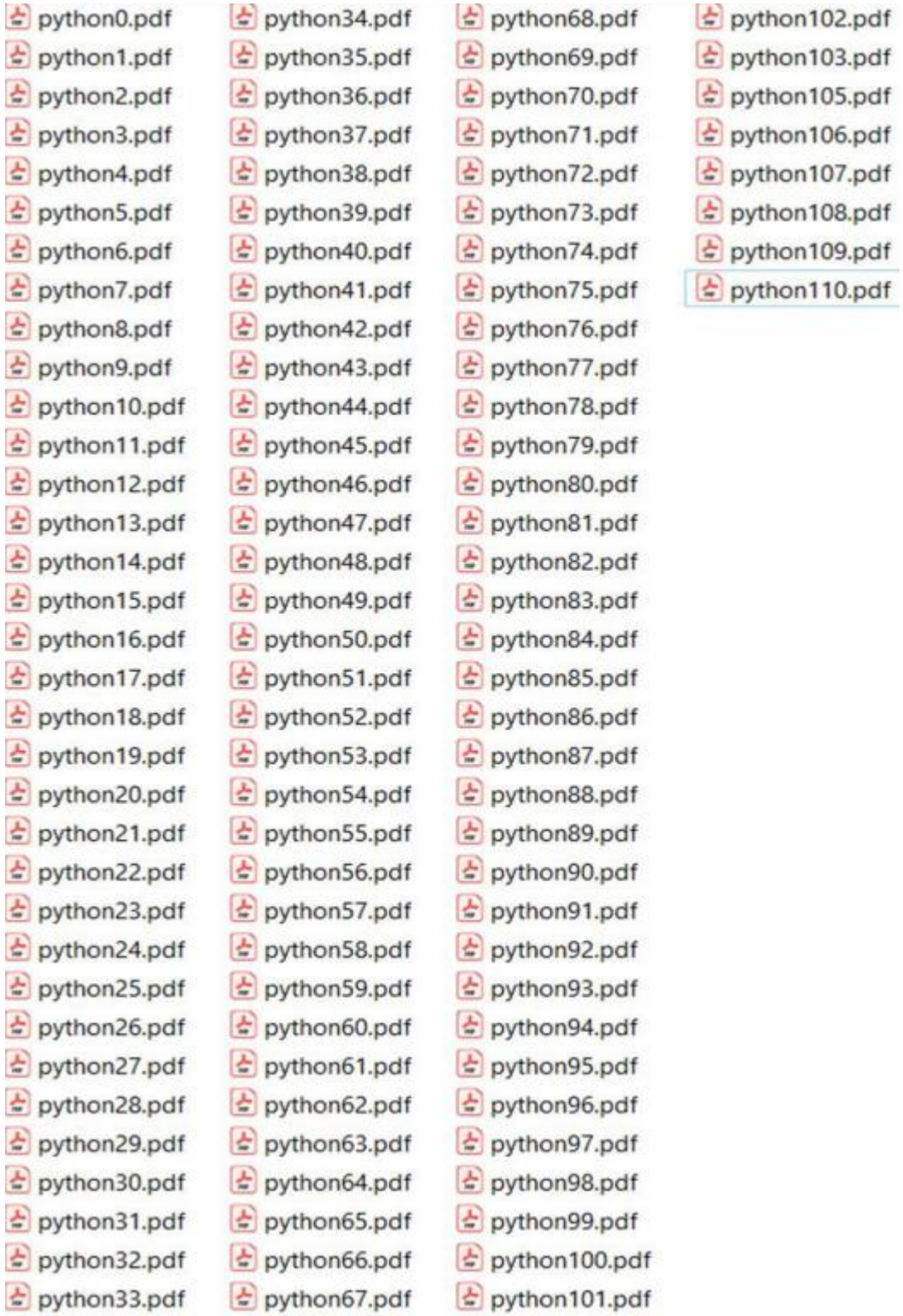
Anexo 1.Código

Algoritmo 1: Descarga de información de la página del Ministerio de Educación (https://educacion.gob.ec/transparencia), mediante las librerías antes instaladas, reques y Beautiful Soup.

```
#librerias para que funcione la descarga
import requests                         #sirve para la descarga de la informacion
from bs4 import BeautifulSoup               #




direcciones = [“https://educacion.gob.ec/transparencia”]


#barrido de las direcciones

for url in direcciones:
   page=requests.get(url)
   soup=BeautifulSoup(page.content, 'html.parser')
   prueba = list()
   nombre = list()
   for enlace in soup.find_all('a', class_='ver'):
      a=enlace.get('title')
      #print(a)
      if a.find('Remuneración mensual')!=-1:
         prueba.append(enlace.get('href'))
         nombre.append(enlace.get('title'))
   for i in range(len(prueba)):
      r=requests.get(prueba[i], stream=True)
      with open(f"python{i}.pdf", "wb") as pdf:
         for chunk in r.iter_content(chunk_size=1024):
            if chunk:
               #graba en el disco duro
               pdf.write(chunk)
```

Los archivos descargados tienen la siguiente forma y se descargan en la carpeta endonde el programa Python corra.

Ilustración 16. Archivos extraídos de la página del Ministerio de Educación



Algoritmo 2: Paso de pdfs a tabla postgres, este proceso toma alrededor de 72 horasen leer 101 archivos PDF entre 50MB y 650MB; de los cuales se han descartado la lectura de archivos de 4 columnas por no aportar con información de los nombres de las personas.

```
El presente algoritmo sirve para poder pasar una estructura de 13 columnas

que son las siguientes:

COLUMNAS ['numero',    'apellidos_nombres',    'puesto_institucional',    'regimen_laboral',
               'partida_presupuestaria',    'grado_jerarquico',    'remuneracion_mensual_unif',
               'remuneracion_anual_unif',    'decimo_tercera_remuneracion',    'decimo_cuarta_re
muneracion',
               'horas_suplementarias',     'encargos_subrogaciones',     'total_ingresos_adicion
ales']

a una tabla que contiene la misma cantidad de columnas mas la fecha y direccion del archivo


la tabla es en una base de datos postgres

create table sueldos.tsueldos_de_13
(
    numero                      varchar(320),
    apellidos_nombres           varchar(256),
    puesto_institucional        varchar(510),
    regimen_laboral             varchar(512),
    partida_presupuestaria      varchar(256),
    grado_jerarquico            varchar(256),
    remuneracion_mensual_unif   varchar(320),
    remuneracion_anual_unif     varchar(320),
    decimo_tercera_remuneracion varchar(320),
    decimo_cuarta_remuneracion  varchar(320),
    horas_suplementarias        varchar(320),
    encargos_subrogaciones      varchar(320),
    total_ingresos_adicionales  varchar(320),
    archivopdf                  varchar(512),
    fechadeclaracion            varchar(32)
);

alter table sueldos.tsueldos_de_13 owner to postgres;




"""


import tabula                           #detecta tablas en el pdf y los pasa a dataframe
import pandas as pd                     #libreria para manejar los dataframe
from sqlalchemy import create_engine    #conexion a la base de datos
from sqlalchemy.pool import QueuePool   #manejo de conexiones a la base de datos
from io import StringIO                 #librerias para manejo de cadenas
```

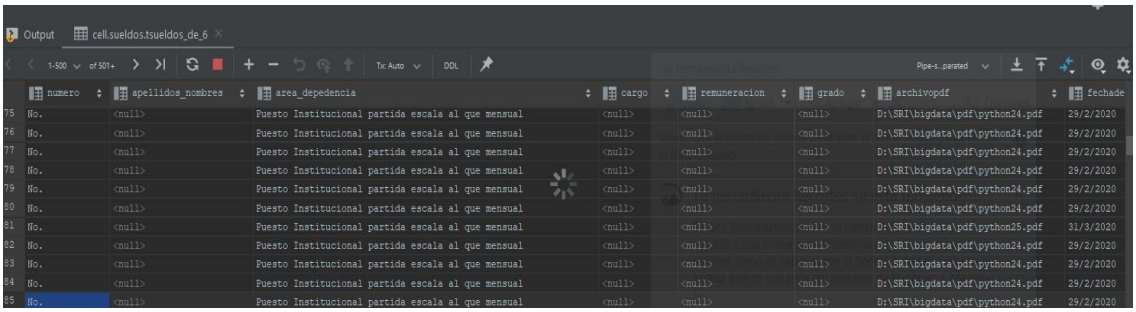
```
import csv,os                           #para manejar archivos csv
import os.path                          #librerias del sistema operativo, encontrar archivos en
una carpeta
from os import path
from glob import glob                   #encuentra archivo sen una carpeta de un tipo especifico
import logging                          #log de eventos
import PyPDF2                           #para manupulacion de los pdf,
import re                               #expresion regular de la fecha




#patro para encontrar fechas en el formato dd/mm/yyyy
patron=re.compile(r'\d{2}/\d{2}/\d{4}')
logger = logging.getLogger("PyPDF2")
logger.setLevel(logging.ERROR)



#unicamente pone el color
os.system("")
class style():
    BLACK = '\033[30m'
    RED = '\033[31m'
    GREEN = '\033[32m'
    YELLOW = '\033[33m'
    BLUE = '\033[34m'
    MAGENTA = '\033[35m'
    CYAN = '\033[36m'
    WHITE = '\033[37m'
    UNDERLINE = '\033[4m'
    RESET = '\033[0m'


# encabezados de la tabla

encabezado13a=['numero',    'apellidos_nombres',    'puesto_institucional',    'regimen_laboral',
               'partida_presupuestaria',    'grado_jerarquico',    'remuneracion_mensual_unif',
               'remuneracion_anual_unif',    'decimo_tercera_remuneracion',    'decimo_cuarta_re
muneracion',
               'horas_suplementarias',     'encargos_subrogaciones',     'total_ingresos_adicion
ales']


encabezado6a=['numero',    'apellidos_nombres',    'area_depedencia',    'cargo',  'remuneracion'
,    'grado']


#cadenas de conexion para poder conectarse a la base de datos y guardar los datos
motor = create_engine(f'postgresql+pg8000://postgres:jdfoxito10@localhost:5432/cell',
echo=False,  poolclass=QueuePool)

#ruta donde deben estar los PDF en formato descrito en el encabezado de este algoritmo
directorio = r"D:\SRI\bigdata\pdf"

#trae un dato unico de una consulta hecha hacia alguna tabla de la base de datos
```

```
def get_scalar(sql):
    valor = ''
    results = motor.execute(sql)
    try:
        if results.rowcount > 0:
            valor = results.first()[0]
        else:
            valor = ''
    except Exception as ex:
        valor=''
    return valor


#traer los archivos de tipo PDF
def fx_traer_archivos_tipo(dir):
    lista=[]
    os.chdir(dir)
    extensiones = ['pdf']
    for ext in extensiones:
        lista+=glob(path.join(dir,f"*.{ext}"))
    return lista


#función principal que realiza el paso de las tablas encontradas en los PDF hacia tablas postgres
de 13 columnas, en el caso de encontrar tablas pdf de 6 columnas se guarda en tablas de 6
columnas

def andromeda(listaPDF):
    #inicializacion de arreglos
    listaHojas13 = []
    listaHojas6 = []
    listaHojas4 = []

    numeroarchivos = len(listaPDF)
    numarchs = 0
    encontrado = ''
    import timeit
    t_0 = timeit.default_timer()
    tabla13 = 'tsueldos_de_13'
    tabla6 = 'tsueldos_de_6'
    tabla4 = 'tsueldos_de_4'
    #se recorren todos los pdf dentro del directorio
    for archivopdf in  listaPDF:
        numarchs+=1
        listaHoja = []
        print(f" {style.MAGENTA}  Inicio del procesamiento de {archivopdf}  {style.RESET}")
        cnsulta13 = f""" select count(1) from sueldos.{tabla13} where archivopdf =
'{archivopdf}'; """
        cnsulta6 = f""" select count(1) from sueldos.{tabla6} where archivopdf = '{archivopdf}';
"""
        if get_scalar(cnsulta13) == 0 and get_scalar(cnsulta6) == 0:
            pdfFileObj = open(archivopdf, 'rb')
            pdfReader = PyPDF2.PdfFileReader(pdfFileObj)
            numeroPaginas = pdfReader.numPages
            ldf = tabula.read_pdf(archivopdf, multiple_tables=True,pages=f"{numeroPaginas}",
encoding = "ISO-8859-1", stream=True)
```

```
            try:
                listaHoja = tabula.read_pdf(archivopdf, pages=f'1-{numeroPaginas}', encoding =
"ISO-8859-1")
            except Exception as ex:
                listaHoja = tabula.read_pdf(archivopdf, pages=f'1-{numeroPaginas}', encoding =
"ISO-8859-1",stream=True)

            if len(listaHoja)>0:
                print(f" {style.CYAN} {numarchs}  de {numeroarchivos} {style.RESET} archivo en
curso  {style.GREEN}  {archivopdf} {style.RESET} ")
                ihoja = 0
                for dfHoja in  listaHoja:
                    ihoja+=1

                    if ihoja ==1:

                        if len(ldf)>0:
                            df = pd.concat(ldf)
                        for ix, fila in df.iterrows():
                            cadena = fila.to_string()
                            sihay = re.findall(r"[\d]{1,2}/[\d]{1,2}/[\d]{4}", cadena)
                            if len(sihay)>0:
                                encontrado = ''.join(sihay)



                    if (len(dfHoja.columns.tolist()) == 13):
                        #enla primera hoja se recorta el encabezado
                        if ihoja ==1:
                            dfHoja = dfHoja.iloc[5:]
                        #se reemplaza el encabezado
                        dfHoja.columns = encabezado13a
                        #en la ultima pagina se recortan las filas finales
                        if ihoja >= numeroPaginas:
                            dfHoja = dfHoja.iloc[:-10]
                        #se escogen las columnas con el tamanho necesario, evitando se filtre
tomar filas resumen, o totales
                        try:
                            dfHoja = dfHoja.loc[dfHoja['numero'].str.len() <= 30]
                            listaHojas13.append(dfHoja)
                        except Exception as ex:
                            print(ex)

                    #misma logica que el blqoue anterior pero para 6 columnas
                    if (len(dfHoja.columns.tolist()) == 6):
                        if ihoja ==1:
                            dfHoja = dfHoja.iloc[6:]

                        dfHoja.columns = encabezado6a
                        dfHoja = dfHoja.loc[dfHoja['numero'].str.len() <= 30]
                        listaHojas6.append(dfHoja)

                    #misma logica que el blqoue anterior pero para 4 columnas
                    if (len(dfHoja.columns.tolist()) == 4):
                        if ihoja ==1:
                            dfHoja = dfHoja.iloc[6:]
```

```
                        dfHoja.columns = encabezado4a
                        listaHojas4.append(dfHoja)

            #Concatenado de segmentos dataframe en un solo
            dfHojas = pd.DataFrame()
            if len(listaHojas13)>0:
                dfHojas = pd.concat(listaHojas13)
                tablaturno = tabla13

            if len(listaHojas6)>0:
                dfHojas = pd.concat(listaHojas6)
                tablaturno = tabla6


            # paso del dataframe reconocido previamente hacia una tabla de la base de datos
postgres

            if not dfHojas.empty:
                dfHojas["archivopdf"] = archivopdf
                dfHojas["fechadeclaracion"] = encontrado
                #en el caso de requerir esete en formato excel
                with motor.connect() as conn, conn.begin():
                    dfHojas.to_sql(tablaturno, con=conn, if_exists='append',index=False,
chunksize=100000, schema='sueldos')
                #medicion del tiempo de demora ne procesar
                t_1 = timeit.default_timer()
                elapsed_time = round((t_1 - t_0), 3)
                print(f" {style.CYAN}  {archivopdf}  se ha procesado en :  {elapsed_time}
s  {style.RESET}")
        else:
            print(f"Se encuentra cargado  {archivopdf}")


    t_total = timeit.default_timer()
    elapsed_time = round((t_total - t_0), 3)
    print(f"  {style.GREEN} se ha procesado todo en :  {elapsed_time} s  {style.RESET}")


#ejecucion de la funcion principal de paso de PDF tabla a tabla postgres usando dataframes

andromeda(fx_traer_archivos_tipo(directorio))
```

Con el algoritmo anterior, finalmente contamos con una tabla en postgres con la siguiente estructura:



Algoritmo 3: Paso a tablas postgres de 13 y 6 columnas, en las tablas de 6 columnas en la segunda columna se encuentra que en algunos casos pueden almacenar el contenido de 7 columnas, las cuales con un tercer algoritmo son transformadas a 13 columnas según la particularidad encontrada.

```
#librerias para el procesamiento
import pandas as pd  # libreria para manipulación de dataframes
from sqlalchemy import create_engine                    # para conectarse a la base de datos
(postgres)
from sqlalchemy.pool import QueuePool                   #  reusa conexiones a la base de datos


#abriendo conexión a la base de datos

motor = create_engine(f'postgresql+pg8000://postgres:jdfoxito10@localhost:5432/cell',
echo=False,  poolclass=QueuePool)

#trae las filas de 6 columnas para su procesamiento
def fx_get_df(engine, consulta):
    df_oracle = pd.read_sql(consulta, engine)
    return df_oracle

#inserta una fila depurada en una nueva tabla
def get_insertar(sql):
    respuesta = -1
    with motor.connect().execution_options(autocommit=True) as conn:
        retorno = conn.execute(str(sql))
        if(retorno.rowcount > 0):
            respuesta = retorno.fetchone()[0]
    return respuesta


#se procesan 23 millones de filas
df23mill = fx_get_df(motor, "select *from sueldos.tsueldos_de_6  order by 1 desc")
nginx = 0
nginxTotal = len(df23mill.index)
#iteracion que encuentra las columnas que presentan anomalías para depurarlas y ubicarlas en el
lugar que correpsponden
#en este punto se identifica que la columna area_depedencia" contiene 7 columnas las cuales se
```

```
proceden a su separacion
for ix, filas in df23mill.iterrows():
    cadena = filas["area_depedencia"]
    nombres = filas["apellidos_nombres"]
    nombres = nombres.replace("'",'')
    cadena = cadena + ' '
    elementos = cadena.split(' ')
    columnasX = []
    if len(elementos) > 15:
        posicion = 0
        pivot = 0
        acumular = ''
        bandera = False
        pasado = False
        columnas = []
#se busca de forma priorizada el numerico de la partida presupuestaria
        for ele in elementos:
            if ele.isnumeric():
                valor = int(ele)
                if valor > 100 and valor < 9999 and not pasado:
                    pivot = valor
                    bandera =True
                    pasado = True

            if bandera:
                columnas.append(acumular.strip())

            if ele== '-':
                ele='0'
            if not bandera:
                acumular+= ele + ' '
            else:
                acumular= ele

            posicion+=1

        num = 0

        longitud=len(columnas)
        acumula=''
       #se ubica la partida presupuestaria que es un numero entre 1 y diez mil
        for nano in columnas:
            if num==0:
                position =  nano.find('-')
                if position>0:
                    a = nano[0:position-2]
                    b = nano[position-1:len(nano)]
                    columnasX.append(a)
                    columnasX.append(b)


            if longitud ==8:
                if num < 3 and num > 0:
                    columnasX.append(nano)

                if num == 3:
```

```
                    acumula= nano

                if num == 4:
                    acumula+= nano
                    columnasX.append(acumula)

                if num >4:
                    columnasX.append(nano)
            else:
                if num > 0:
                    columnasX.append(nano)

            num+=1
        # en el caso de que tenga 8 columnas implica que son archivos bien escaneados de 6 filas,
se añaden dos columnas de fecha y pdf cargado, adicionalmente se cargan las siguientes columnas
para poder transformas a 13 columnas
        if len(columnasX) == 8:
            columnasX.insert(0,nombres)
            columnasX.insert(0,filas["numero"])
            grado = ''
            if filas["grado"] == '-':
                grado = '0'
            columnasX.append('')
            columnasX.append('')
            columnasX.append(grado)
            columnasX.append(filas["archivopdf"])
            columnasX.append(filas["fechadeclaracion"])


    else:
        columnasX.append(filas["numero"])
        columnasX.append(nombres)
        columnasX.append(filas["cargo"])
        columnasX.append(filas["area_depedencia"])
        columnasX.append('')
        columnasX.append(filas["grado"])
        columnasX.append(filas["remuneracion"])
        columnasX.append('')
        columnasX.append('')
        columnasX.append('')
        columnasX.append('')
        columnasX.append('')
        columnasX.append('')
        columnasX.append(filas["archivopdf"])
        columnasX.append(filas["fechadeclaracion"])
    #si existen mas de 13 columnas se insertan en la nueva tabla que consolida las anomalias
    if len(columnasX) > 13:
        nginx+=1
        consulta = f"""insert into sueldos.tsueldos_de_13_n2
values('{columnasX[0]}','{columnasX[1]}','{columnasX[2]}','{columnasX[3]}','{columnasX[4]}','{col
umnasX[5]}','{columnasX[6]}',
                    '{columnasX[7]}','{columnasX[8]}','{columnasX[9]}','{columnasX[10]}','{column
asX[11]}','{columnasX[12]}','{columnasX[13]}','{columnasX[14]}') returning 1; """
        get_insertar(consulta)
        print(f" {nginx}  de  {nginxTotal} ")
```

La corrida de esta sección toma aproximadamente 5 horas. Procesa 23 millones de filas, en donde si encuentra que la columna 2 contiene columnas internas las separa, las ubica en su columna respectiva y entrega una tabla cargada con 13 columnas más la fecha y ruta del pdf y finalmente contamos con la tabla de datos lista para sobre ella trabajar en una herramienta de visualización en Power BI.