octane_gas_analysis

July 4, 2024

0.1 Análisis Calidad Gasolina - Repúlica Dominicana

Fecha: Julio 2024

Preparado por: Rafael J. Mateo C.

Contacto: rmateo@squalitas.com

[ ]:
```
from pypdf import PdfReader, PdfWriter
from pypdf.errors import PdfStreamError
from ocrmypdf.exceptions import SubprocessOutputError
import os
import ocrmypdf
from tqdm import tqdm
from IPython.display import clear_output
import json
import pandas as pd
from typing import Callable, Generator
from datetime import datetime
from azure.core.credentials import AzureKeyCredential
from azure.core.exceptions import HttpResponseError
from azure.ai.formrecognizer import DocumentAnalysisClient
from io import BufferedReader
from dotenv import load_dotenv
from pathlib import Path
from dateutil.parser import parse, ParserError
import re
import traceback
from collections import defaultdict
from unidecode import unidecode
import numpy as np
import matplotlib.pyplot as plt
```

[ ]:
```
skip_files = []
files_with_error = []
processed_files = []

raw_dir = './files/raw'
tmp_dir = './files/tmp'
```

```
output_dir = './files/ocr'
json_dir = './files/json'
sample_dir = './files/sample'
log_dir = './var'
```

[ ]:
```
for dirname in [raw_dir, tmp_dir, output_dir, json_dir, log_dir]:
    if not os.path.exists(dirname):
        os.makedirs(dirname)
```

0.1.1 1. Preprocesamiento

1.1. Eliminando Páginas que no contienen los datos numéricos de las pruebas El siguiente código elimina los arhivos que tengan cuatro páginas o menos, ya que estos son los reportes de las estaciones que no autorizaron la realización de los ensayos. También elimina las últimas dos páginas, las cuales solo contienen las imágenes de los ensayos y las firmas de los que realizaron el reporte.

[ ]:
```
for name in os.listdir(raw_dir):
    src = os.path.join(raw_dir, name)
    reader = PdfReader(src)

    if len(reader.pages) <= 4:
        skip_files.append(name)
        continue

    dest = os.path.join(tmp_dir, name)

    #Eliminar las últimas dos páginas, ya que son fotos y firmas
    for page_num in [reader.get_num_pages()-1,reader.get_num_pages()-2,0]:
        reader.remove_page(page_num)

    writer = PdfWriter(clone_from=reader)
    writer.write(dest)
```

```
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
```

```
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
/Prev=0 in the trailer - assuming there is no previous xref table
```

```
/Prev=0 in the trailer - assuming there is no previous xref table
```

Un total de 130 archivos contienen cuatro páginas o menos, por lo que fueron omitidos para este análisis.

[ ]:
```
len(skip_files)
```

[ ]:
```
130
```

1.2. Realizando OCR En la siguiente celda realizaremos un OCR a aquellos archivos que contienen más de 4 páginas. Los archivos PDFs con OCR son colocados en el directorio 'files/ocr' por si requieren ser consultados posteriormente.

[ ]:
```
import warnings

error_files = []
warning_files = []

for name in tqdm(os.listdir(tmp_dir), unit='file', desc='Files processed: '):

    #Omite los archivos que tienen cuatro páginas o menos
    if name in skip_files or os.path.exists(os.path.join(output_dir, name)):
        clear_output(wait=True)
        continue

    try:
        with warnings.catch_warnings(record = True) as w:

            warnings.simplefilter("always")
            #Realiza OCR
            ocrmypdf.ocr(os.path.join(tmp_dir, name),
                    os.path.join(output_dir, name),skip_text=True,␣
 ↪language='spa', deskew=True)
            if w:
                raise w[-1].category(w[-1].message)

    except SubprocessOutputError as error:
        error_files.append({'file': name, 'error': error})

    except Warning as e:
        warning_files.append({'file': name, 'warning': e})

    clear_output(wait=True)
```

```
Files processed: 100%|      | 810/810 [1:33:18<00:00,  6.91s/file]
```

Solo un archivo procesado generó un error. Este se refiere al reporte "2023 03 047.pdf". Revisando el archivo manualmente, la razón del error se debe a que las primeras páginas del reporte ya están en texto y no escaneado. Como es solo un archivo, manualmente pueden eliminarse las primeras

cuantro páginas para realizar el OCR a las tablas de los datos solamente, ya que estas si están escaneadas.

[ ]:
```
error_files
```

[ ]:
```
[{'file': '2023 03 047.pdf',
  'error': ocrmypdf.exceptions.SubprocessOutputError('Ghostscript PDF/A
rendering failed')}]
```

1.3. Eliminando Páginas Ahora procedemos a eliminar las páginas que no contengan tablas de datos. Estas páginas suelen contener portadas, resúmenes u otras informaciones no relevantes para este análisis. La forma de filtrar las páginas que nos interesan es identificando aquellas que tengan el texto "Test Report".

[ ]:
```
def remove_pages(compare_func: Callable, input_dir: str, output_dir: str)␣
↪->dict[str, dict|list]:

    results = {'deleted_pages':defaultdict(list), 'errors': {} }

    for name in tqdm(os.listdir(input_dir), unit='file', desc='Files processed:␣
↪'):

        try:
            if not name.endswith('.pdf'):
                continue

            reader = PdfReader(os.path.join(input_dir, name))
            pages = list(reader.pages)

        except PdfStreamError as e:
            results['errors'][name] = e
            continue

        for i,page in enumerate(pages):
            text = page.extract_text()

            if not compare_func(text):

                reader.remove_page(page, True)
                results['deleted_pages'][name].append(i+1)

        writer = PdfWriter(clone_from=reader)
        writer.write(os.path.join(output_dir, name))

    return results
```

[ ]:
```
def page_count(dirname: str) -> int:
    num_pages = 0
    for name in tqdm(os.listdir(dirname), unit='file', desc='Files processed:␣
 ↪'):

        try:
            if not name.endswith('.pdf'):
                continue

            reader = PdfReader(os.path.join(dirname, name))
            num_pages += len(reader.pages)

        except PdfStreamError as e:
            continue
    return num_pages
```

[ ]:
```
results = remove_pages(lambda text: text.find("Test Report") >= 0, output_dir,␣
 ↪tmp_dir)
```

```
Files processed: 100%|      | 810/810 [03:08<00:00,  4.30file/s]
```

A continuación se muestran las páginas eliminadas para cada archivo

[ ]:
```
results
```

[ ]:
```
{'deleted_pages': defaultdict(list,
             {'2023 03 140.pdf': [1, 2, 3, 4],
              '2023 03 034.pdf': [2],
              '2023 05 065.pdf': [1, 2, 3, 4],
              '2023 05 059.pdf': [1],
              '2023 02 030.pdf': [2],
              '2023 08 047.pdf': [1],
              '2023 03 021.pdf': [4],
              '2023 07 001.pdf': [1, 2, 3],
              '2023 05 112.pdf': [3, 4],
              '2023 03 037.pdf': [3],
              '2023 05 072.pdf': [1],
              '2023 08 051.pdf': [1, 2],
              '2023 04 088.pdf': [1, 3],
              '2023 04 089.pdf': [1, 2, 3, 4],
              '2023 03 036.pdf': [4],
              '2023 03 022.pdf': [3],
              '2023 05 107.pdf': [1, 2, 3, 4],
              '2023 06 007.pdf': [1],
              '2023 01 080.pdf': [3],
              '2023 06 003.pdf': [2],
              '2023 06 017.pdf': [1, 2, 3, 4],
              '2023 04 066.pdf': [1],
```

```
'2023 05 076.pdf': [1],
'2023 06 016.pdf': [1, 2, 3, 4],
'2023 07 013.pdf': [1],
'2023 07 011.pdf': [1, 2, 3, 4],
'2023 05 128.pdf': [1, 2, 3, 4],
'2023 05 100.pdf': [1],
'2023 03 031.pdf': [1],
'2023 05 060.pdf': [3],
'2023 04 058.pdf': [1],
'2023 03 030.pdf': [4],
'2023 06 029.pdf': [1],
'2023 01 032.pdf': [2],
'2023 06 099.pdf': [1, 2, 3, 4],
'2023 06 106.pdf': [1],
'2023 03 043.pdf': [5],
'2023 07 102.pdf': [1, 2, 3, 4],
'2023 05 013.pdf': [2],
'2023 05 007.pdf': [1],
'2023 07 062.pdf': [1, 2, 3, 4],
'2023 07 074.pdf': [1],
'2023 01 019.pdf': [6],
'2023 06 105.pdf': [1, 2, 3, 4],
'2023 08 026.pdf': [1, 2],
'2023 05 005.pdf': [3],
'2023 07 100.pdf': [1, 2, 3, 4],
'2023 04 028.pdf': [3],
'2023 06 070.pdf': [1],
'2023 07 049.pdf': [3],
'2023 07 065.pdf': [1],
'2023 06 060.pdf': [1],
'2023 07 110.pdf': [1],
'2023 05 015.pdf': [1, 2, 3, 4],
'2023 06 049.pdf': [1, 2, 3, 4],
'2023 06 077.pdf': [1, 2],
'2023 06 117.pdf': [1],
'2023 08 021.pdf': [1],
'2023 06 076.pdf': [1, 2],
'2023 07 095.pdf': [1, 2, 3, 4],
'2023 05 027.pdf': [2],
'2023 07 122.pdf': [1],
'2023 03 063.pdf': [1, 2, 3, 4, 5, 6, 7, 8, 9],
'2023 05 030.pdf': [1, 2, 3, 4],
'2023 04 034.pdf': [2],
'2023 07 108.pdf': [1],
'2023 05 031.pdf': [1],
'2023 03 064.pdf': [1, 3],
'2023 08 003.pdf': [1],
```

```
                '2023 03 139.pdf': [1, 2, 3, 4],
                '2023 02 100.pdf': [5],
                '2023 06 122.pdf': [1],
                '2023 08 029.pdf': [1],
                '2023 06 080.pdf': [1],
                '2023 03 015.pdf': [3],
                '2023 03 029.pdf': [2, 4],
                '2023 05 044.pdf': [1],
                '2023 04 096.pdf': [1],
                '2023 02 004.pdf': [5],
                '203 05 139.pdf': [1],
                '2023 05 119.pdf': [1],
                '2023 05 131.pdf': [4],
                '2023 03 101. 102, 103, 104.pdf': [1, 2, 3, 4, 5, 6, 7, 8, 9],
                '2023 03 016.pdf': [3],
                '2023 04 042.pdf': [1, 2, 3, 4],
                '2023 01 076.pdf': [1],
                '2023 06 036.pdf': [1],
                '2023 05 123.pdf': [5],
                '2023 07 032.pdf': [1],
                '2023 01 075.pdf': [1],
                '2023 05 069.pdf': [5],
                '2023 02 029.pdf': [1],
                '2023 07 144.pdf': [1, 2],
                '2023 02 015.pdf': [1],
                '2023 04 044.pdf': [6],
                '2023 07 145.pdf': [1, 2],
                '2023 07 031.pdf': [1]}),
 'errors': {}}
```

Debido a que usaremos un servicio de pago de Azure, limitaremos el análisis únicamente a las Gasolinas. Para esto, filtramos solamente las páginas que contienen la palabra "GASOLINA". En caso de requerirse los datos para el Gasoil, simplemente sería sustituir la palabra "GASOLINA" por "GASOIL".

[ ]:
```
results2 = remove_pages(lambda text: text.find("GASOLINA") >= 0, tmp_dir,'./
 ↪files/sample')
```

```
Files processed: 100%|      | 810/810 [03:09<00:00,  4.28file/s]
```

A continuación se presentan las páginas que fueron eliminadas para cada archivo

[ ]:
```
results2
```

[ ]:
```
{'deleted_pages': defaultdict(list,
             {'2023 07 014.pdf': [4, 5],
              '2023 07 028.pdf': [4, 5],
              '2023 04 100.pdf': [4, 5],
              '2023 01 051.pdf': [3, 4],
```

```
'2023 01 045.pdf': [3, 4],
'2023 01 079.pdf': [2, 5],
'2023 06 011.pdf': [4, 5],
'2023 06 005.pdf': [4, 5],
'2023 06 039.pdf': [4, 5],
'2023 03 168.pdf': [4, 5],
'2023 05 111.pdf': [4, 5],
'2023 05 105.pdf': [4, 5],
'2023 03 034.pdf': [1, 3, 4],
'2023 03 008.pdf': [4],
'2023 08 052.pdf': [4],
'2023 05 065.pdf': [2],
'2023 05 071.pdf': [4, 5],
'2023 08 046.pdf': [4, 5],
'2023 05 059.pdf': [3],
'2023 02 025.pdf': [4, 5],
'2023 02 031.pdf': [3, 4, 5],
'2023 07 148.pdf': [4, 5],
'2023 02 019.pdf': [4, 5],
'2023 04 074.pdf': [4, 5],
'2023 04 060.pdf': [4, 5],
'2023 04 048.pdf': [5, 6],
'2023 04 049.pdf': [3, 4],
'2023 04 061.pdf': [4, 5],
'2023 04 075.pdf': [4, 5],
'2023 02 030.pdf': [3, 4],
'2023 02 024.pdf': [4, 5],
'2023 05 070.pdf': [4, 5],
'2023 08 053.pdf': [4, 5],
'2023 05 064.pdf': [4, 5],
'2023 03 009.pdf': [1, 2, 4],
'2023 03 021.pdf': [4],
'2023 05 138.pdf': [4, 5],
'2023 05 104.pdf': [4, 5],
'2023 05 110.pdf': [4, 5],
'2023 06 038.pdf': [4, 5],
'2023 06 004.pdf': [4, 5],
'2023 03 155.pdf': [4, 5],
'2023 06 010.pdf': [4, 5],
'2023 01 078.pdf': [2, 4, 5],
'2023 01 044.pdf': [4],
'2023 04 101.pdf': [4, 5],
'2023 01 050.pdf': [3, 4],
'2023 07 029.pdf': [4],
'2023 07 015.pdf': [4, 5],
'2023 07 017.pdf': [4, 5],
'2023 07 003.pdf': [4, 5, 6],
```

```
'2023 01 052.pdf': [3],
'2023 03 157.pdf': [3, 4],
'2023 06 006.pdf': [4, 5],
'2023 06 012.pdf': [4, 5],
'2023 03 143.pdf': [4],
'2023 05 106.pdf': [3],
'2023 05 112.pdf': [3],
'2023 03 023.pdf': [4, 5],
'2023 03 037.pdf': [2, 3, 4],
'2023 05 099.pdf': [4],
'2023 08 045.pdf': [4, 5],
'2023 05 072.pdf': [2],
'2023 05 066.pdf': [4],
'2023 02 026.pdf': [4, 5],
'2023 04 088.pdf': [2],
'2023 04 063.pdf': [5, 6],
'2023 04 077.pdf': [4, 5],
'2023 04 076.pdf': [4, 5],
'2023 04 062.pdf': [4, 5],
'2023 02 027.pdf': [4, 5],
'2023 02 033.pdf': [4, 5],
'2023 05 067.pdf': [4, 5],
'2023 08 050.pdf': [4, 5],
'2023 05 073.pdf': [4, 5],
'2023 03 036.pdf': [4],
'2023 03 022.pdf': [1, 3, 4],
'2023  03 169.pdf': [4, 5],
'2023 06 013.pdf': [4, 5],
'2023 03 142.pdf': [4, 5],
'2023 03 156.pdf': [4, 5],
'2023 01 053.pdf': [4, 5],
'2023 04 102.pdf': [4, 5],
'2023 01 047.pdf': [4],
'2023 07 002.pdf': [4],
'2023 07 016.pdf': [4, 5],
'2023 01 080.pdf': [3, 4],
'2023 07 012.pdf': [5, 6],
'2023 07 006.pdf': [4, 5],
'2023 01 043.pdf': [3],
'2023 01 057.pdf': [3],
'2023 06 003.pdf': [3, 4],
'2023 03 152.pdf': [2, 4, 5],
'2023 06 017.pdf': [3],
'2023 05 103.pdf': [4, 5],
'2023 05 117.pdf': [4, 5],
'2023 05 088.pdf': [4, 5],
'2023 03 026.pdf': [4],
```

```
'2023 03 032.pdf': [4, 5],
'2023 05 077.pdf': [4],
'2023 08 040.pdf': [4, 5],
'2023 05 063.pdf': [4, 5, 6],
'2023 04 099.pdf': [4, 5],
'2023 02 037.pdf': [4, 5],
'2023 02 023.pdf': [4, 5],
'2023 04 072.pdf': [4, 5],
'2023 04 073.pdf': [4, 5],
'2023 04 098.pdf': [4],
'2023 05 062.pdf': [4, 5],
'2023 05 076.pdf': [1],
'2023 08 041.pdf': [4, 5],
'2023 03 033.pdf': [5, 6],
'2023 03 027.pdf': [1, 4, 5],
'2023 05 089.pdf': [5],
'2023 05 116.pdf': [4, 5],
'2023 05 102.pdf': [3],
'2023 03 147.pdf': [4, 5],
'2023 06 016.pdf': [1],
'2023 06 002.pdf': [4, 5],
'2023 03 153.pdf': [4, 5],
'2023 01 056.pdf': [3, 4],
'2023 01 042.pdf': [3, 4],
'2023 07 007.pdf': [4, 5],
'2023 01 081.pdf': [1, 4, 5],
'2023 07 013.pdf': [5, 6],
'2023 07 039.pdf': [4, 5],
'2023 07 005.pdf': [4, 5],
'2023 07 011.pdf': [1],
'2023 01 068.pdf': [4, 5],
'2023 01 054.pdf': [1, 2],
'2023 04 111.pdf': [4],
'2023 01 040.pdf': [3],
'2023 03 179.pdf': [4, 5],
'2023 06 014.pdf': [4, 5],
'2023 05 114.pdf': [4, 5],
'2023 03 019.pdf': [4, 5],
'2023 03 031.pdf': [3, 4],
'2023 03 025.pdf': [4, 5],
'2023 05 048.pdf': [4, 5],
'2023 05 060.pdf': [3],
'2023 08 043.pdf': [4],
'2023 05 074.pdf': [4],
'2023 02 008.pdf': [1, 3, 4, 5],
'2023 02 034.pdf': [4, 5],
'2023 04 059.pdf': [4],
```

```
'2023 04 071.pdf': [4, 5],
'2023 04 064.pdf': [4, 5],
'2023 04 070.pdf': [6, 7],
'2023 04 058.pdf': [3, 4],
'2023 02 035.pdf': [4, 5],
'2023 02 021.pdf': [4, 5],
'2023 02 009.pdf': [4, 5],
'2023 08 042.pdf': [6, 7],
'2023 05 075.pdf': [4, 5],
'2023 05 061.pdf': [4, 5],
'2023 05 049.pdf': [4, 5],
'2023 03 024.pdf': [3, 4, 5],
'2023 03 030.pdf': [4],
'2023 03 018.pdf': [1, 4, 5],
'2023 05 101.pdf': [4, 5],
'2023 05 115.pdf': [4, 5],
'2023 05 129.pdf': [4, 5],
'2023 06 001.pdf': [4, 5],
'2023 06 015.pdf': [4, 5],
'2023 03 144.pdf': [4, 5],
'2023 03 178.pdf': [4],
'2023 04 110.pdf': [4, 5],
'2023 01 055.pdf': [3, 4],
'2023 01 069.pdf': [5, 6],
'2023 07 010.pdf': [1, 2],
'2023 01 082.pdf': [4, 5],
'2023 07 004.pdf': [4, 5],
'2023 07 038.pdf': [4, 5],
'2023 07 063.pdf': [4, 5],
'2023 07 077.pdf': [3],
'2023 01 032.pdf': [1],
'2023 01 026.pdf': [1, 2],
'2023 07 088.pdf': [4, 5],
'2023 06 072.pdf': [4, 5],
'2023 03 123.pdf': [3, 4],
'2023 06 066.pdf': [5, 6, 7],
'2023 06 099.pdf': [2],
'2023 03 057.pdf': [4, 5],
'2023 03 043.pdf': [4],
'2023 06 112.pdf': [5, 6],
'2023 05 006.pdf': [4, 5],
'2023 03 094.pdf': [4, 5],
'2023 08 031.pdf': [4, 5],
'2023 08 025.pdf': [4, 5],
'2023 05 012.pdf': [4, 5],
'2023 08 019.pdf': [4, 5],
'2023 02 046.pdf': [4, 5],
```

```
'2023 07 117.pdf': [4],
'2023 07 103.pdf': [4],
'2023 02 052.pdf': [1, 3, 4, 5],
'2023 02 085.pdf': [2, 4, 5],
'2023 04 017.pdf': [5, 6],
'2023 04 003.pdf': [4, 5],
'2023 04 002.pdf': [4, 5],
'2023 02 090.pdf': [1, 2, 4, 5],
'2023 02 084.pdf': [2, 3, 4, 5],
'2023 04 016.pdf': [5, 6],
'2023 02 053.pdf': [1, 2, 3, 4],
'2023 02 047.pdf': [4, 5],
'2023 07 116.pdf': [4, 5],
'2023 08 018.pdf': [4],
'2023 08 024.pdf': [4, 5],
'2023 08 030.pdf': [4, 5],
'2023 03 095.pdf': [4, 5],
'2023 06 113.pdf': [4, 5],
'2023 06 107.pdf': [4, 5],
'2023 06 098.pdf': [5, 6],
'2023 03 136.pdf': [4, 5],
'2023 06 073.pdf': [4, 5],
'2023 07 089.pdf': [3, 4],
'2023 01 027.pdf': [2, 3],
'2023 01 033.pdf': [3, 4],
'2023 07 076.pdf': [3],
'2023 07 062.pdf': [3],
'2023 07 060.pdf': [4, 5],
'2023 07 048.pdf': [4, 5],
'2023 01 025.pdf': [4, 5],
'2023 01 031.pdf': [2, 3],
'2023 01 019.pdf': [3, 4, 8, 9],
'2023 06 065.pdf': [4, 5],
'2023 06 071.pdf': [4, 5],
'2023 06 059.pdf': [4, 5],
'2023 03 108.pdf': [4],
'2023 06 111.pdf': [3, 4],
'2023 06 105.pdf': [3],
'2023 03 068.pdf': [5, 6],
'2023 05 011.pdf': [4, 5],
'2023 08 026.pdf': [3],
'2023 08 032.pdf': [4, 5],
'2023 05 005.pdf': [3, 4],
'2023 05 039.pdf': [4, 5],
'2023 07 100.pdf': [3],
'2023 07 114.pdf': [4],
'2023 07 128.pdf': [4, 5],
```

```
'2023 02 079.pdf': [4, 5],
'2023 02 092.pdf': [3, 4],
'2023 04 014.pdf': [4, 5],
'2023 02 086.pdf': [4, 5],
'2023 04 028.pdf': [3, 4],
'2023 04 029.pdf': [4, 5],
'2023 04 015.pdf': [4, 5],
'2023 02 087.pdf': [4, 5],
'2023 04 001.pdf': [4, 5],
'2023 07 129.pdf': [4],
'2023 02 078.pdf': [4, 5],
'2023 07 115.pdf': [5, 6],
'2023 05 038.pdf': [4, 5],
'2023 08 033.pdf': [4],
'2023 05 004.pdf': [4, 5],
'2023 05 010.pdf': [4, 5],
'2023 08 027.pdf': [4, 5],
'2023 06 104.pdf': [4, 5],
'2023 06 110.pdf': [4, 5, 7],
'2023 03 041.pdf': [4],
'2023 06 058.pdf': [4, 5],
'2023 03 109.pdf': [4, 5],
'2023 03 121.pdf': [4, 5],
'2022 12 028.pdf': [4],
'2023 06 070.pdf': [1],
'2023 06 064.pdf': [4, 5],
'2023 03 135.pdf': [3, 4],
'2023 01 018.pdf': [3, 4],
'2023 01 030.pdf': [3],
'2023 01 024.pdf': [4, 5],
'2023 07 061.pdf': [4, 5],
'2023 07 075.pdf': [4, 5],
'2023 07 059.pdf': [4, 5],
'2023 07 071.pdf': [4, 5],
'2023 07 065.pdf': [1],
'2023 01 008.pdf': [4, 5],
'2023 01 034.pdf': [2, 3],
'2023 06 048.pdf': [4, 5],
'2023 03 119.pdf': [4, 5],
'2023 03 131.pdf': [4, 5],
'2023 06 074.pdf': [4, 5],
'2023 03 125.pdf': [4, 5],
'2023 03 079.pdf': [4, 5],
'2023 03 045.pdf': [4, 5],
'2023 06 114.pdf': [4, 5],
'2023 06 100.pdf': [4, 5],
'2023 03 051.pdf': [1, 2, 4],
```

```
'2023 05 028.pdf': [4, 5],
'2023 08 023.pdf': [4],
'2023 05 014.pdf': [4, 5],
'2023 08 037.pdf': [4, 5],
'2023 07 139.pdf': [4, 5],
'2023 07 105.pdf': [4, 5],
'2023 02 054.pdf': [2, 3, 4, 5],
'2023 07 111.pdf': [4, 5],
'2023 04 039.pdf': [4, 5],
'2023 04 005.pdf': [4, 5],
'2023 02 097.pdf': [4, 5],
'2023 04 004.pdf': [4, 5],
'2023 02 096.pdf': [4, 5],
'2023 04 038.pdf': [4, 5],
'2023 07 110.pdf': [1, 2, 3],
'2023 07 104.pdf': [3],
'2023 02 055.pdf': [3, 4, 5],
'2023 07 138.pdf': [5, 6, 7, 8, 9, 10],
'2023 02 069.pdf': [4, 5],
'2023 05 001.pdf': [4, 5],
'2023 08 036.pdf': [4, 5],
'2023 08 022.pdf': [4, 5],
'2023 05 029.pdf': [4, 5],
'2023 06 101.pdf': [1, 2],
'2023 03 044.pdf': [2, 4, 5],
'2023 06 115.pdf': [4, 5],
'2023 06 075.pdf': [4, 5],
'2023 06 061.pdf': [4, 5],
'2023 03 118.pdf': [5, 6],
'2023 01 035.pdf': [2, 3],
'2023 01 021.pdf': [4, 5],
'2023 01 009.pdf': [3, 4],
'2023 07 064.pdf': [4, 5],
'2023 07 070.pdf': [4, 5],
'2023 02 109.pdf': [4, 5],
'2023 07 058.pdf': [4, 5],
'2023 07 066.pdf': [4, 5],
'2023 07 072.pdf': [4],
'2023 01 037.pdf': [4, 5],
'2023 01 023.pdf': [4, 5],
'2023 06 077.pdf': [1],
'2023 06 063.pdf': [1, 3, 4],
'2023 06 088.pdf': [4, 5],
'2023 03 052.pdf': [4, 5],
'2023 06 103.pdf': [4, 5],
'2023 03 046.pdf': [4, 5],
'2023 08 008.pdf': [4, 5],
```

```
'2023 03 091.pdf': [4],
'2023 08 034.pdf': [4],
'2023 05 003.pdf': [4, 5],
'2023 03 085.pdf': [4, 5],
'2023 07 112.pdf': [4, 5],
'2023 02 043.pdf': [3, 4, 5],
'2023 07 106.pdf': [4, 5],
'2023 04 012.pdf': [4, 5],
'2023 04 007.pdf': [4],
'2023 04 013.pdf': [4, 5],
'2023 02 081.pdf': [3, 4, 5],
'2023 07 107.pdf': [4, 5],
'2023 07 113.pdf': [4, 5],
'2023 02 042.pdf': [3, 4, 5],
'2023 05 016.pdf': [4, 5, 6],
'2023 08 021.pdf': [4, 5],
'2023 08 035.pdf': [3],
'2023 05 002.pdf': [3, 4],
'2023 08 009.pdf': [4, 5],
'2023 06 116.pdf': [4, 5],
'2023 03 047.pdf': [5, 6],
'2023 03 053.pdf': [4, 5],
'2023 06 102.pdf': [5, 6],
'2023 06 089.pdf': [4, 5],
'2023 06 062.pdf': [4, 5],
'2023 01 036.pdf': [4, 5],
'2023 07 098.pdf': [4, 5],
'2023 07 073.pdf': [4, 5],
'2023 07 067.pdf': [4, 5],
'2023 07 042.pdf': [4, 5],
'2023 07 056.pdf': [4, 5],
'2023 02 107.pdf': [4],
'2023 01 013.pdf': [4, 5],
'2023 07 081.pdf': [4, 5],
'2023 01 007.pdf': [4, 5],
'2023 06 053.pdf': [4, 5],
'2023 03 116.pdf': [4, 5],
'2023 06 047.pdf': [4, 5],
'2023 06 090.pdf': [4, 5, 6],
'2023 06 084.pdf': [4],
'2023 06 127.pdf': [4, 5],
'2023 05 027.pdf': [3, 4],
'2023 08 010.pdf': [4, 5],
'2023 08 004.pdf': [4, 5],
'2023 05 033.pdf': [4, 5],
'2023 08 038.pdf': [3, 4],
'2023 02 067.pdf': [2, 4, 5],
```

```
'2023 07 136.pdf': [4, 5],
'2023 04 036.pdf': [4, 5],
'2023 04 022.pdf': [4, 5],
'2023  04 009.pdf': [4, 5],
'2023 02 099.pdf': [4],
'2023 04 023.pdf': [4, 5],
'2023 04 037.pdf': [4, 5],
'2023 02 072.pdf': [1, 4, 5],
'2023 02 066.pdf': [1, 2, 3, 4, 5],
'2023 07 137.pdf': [4, 5],
'2023 08 039.pdf': [4, 5],
'2023 03 088.pdf': [4],
'2023 08 005.pdf': [4, 5],
'2023 05 032.pdf': [4, 5],
'2023 05 026.pdf': [4, 5],
'2023 08 011.pdf': [4, 5],
'2023 06 126.pdf': [4, 5],
'2023 06 085.pdf': [4, 5],
'2023 05 146.pdf': [4, 5],
'2023 06 091.pdf': [6, 7],
'2023 03 117.pdf': [4, 5],
'2023 06 046.pdf': [4, 5],
'2023 06 052.pdf': [4, 5],
'2023 07 094.pdf': [4, 5],
'2023 01 006.pdf': [3],
'2023 01 012.pdf': [3, 4],
'2023 07 080.pdf': [1, 2],
'2023 07 057.pdf': [5, 6],
'2023 02 106.pdf': [3, 4, 5],
'2023 02 112.pdf': [4, 5],
'2023 07 043.pdf': [4, 5],
'2023 02 104.pdf': [1, 4, 5],
'2023 07 055.pdf': [4, 5],
'2023 07 041.pdf': [4, 5],
'2023 02 110.pdf': [4, 5],
'2023 07 069.pdf': [4, 5],
'2023 07 096.pdf': [4, 5],
'2023 07 082.pdf': [4, 5],
'2023 01 010.pdf': [4, 5],
'2023 01 038.pdf': [4, 5],
'2023 06 044.pdf': [4, 5],
'2023 03 115.pdf': [4, 5],
'2023 06 050.pdf': [4, 5],
'2023 06 078.pdf': [4, 5],
'2023 03 129.pdf': [4, 5],
'2023 05 144.pdf': [5, 6],
'2023 06 087.pdf': [4, 5],
```

```
'2023 06 093.pdf': [5, 6],
'2023 03 061.pdf': [4, 5],
'2023 06 124.pdf': [4, 5],
'2023 06 118.pdf': [4, 5],
'2023 05 030.pdf': [3],
'2023 08 007.pdf': [4, 5],
'2023 05 024.pdf': [4, 5],
'2023 05 018.pdf': [4, 5],
'2023 02 070.pdf': [4, 5],
'2023 07 121.pdf': [4, 5],
'2023 07 135.pdf': [4, 5],
'2023 02 064.pdf': [4, 5],
'2023 02 058.pdf': [4, 5],
'2023 04 021.pdf': [4],
'2023 04 035.pdf': [4, 5],
'2023 04 034.pdf': [3, 4],
'2023 04 020.pdf': [5, 6],
'2023 07 108.pdf': [1],
'2023 07 134.pdf': [4, 5],
'2023 02 065.pdf': [4, 5],
'2023 07 120.pdf': [4, 5],
'2023 05 025.pdf': [5, 6],
'2023 08 006.pdf': [4, 5],
'2023 03 048.pdf': [4, 5],
'2023 06 119.pdf': [4, 5],
'2023 06 125.pdf': [4, 5],
'2023 06 092.pdf': [5, 6],
'2023 06 086.pdf': [4, 5],
'2023 05 145.pdf': [4, 5],
'2023 06 079.pdf': [4, 5],
'2023 03 128.pdf': [4],
'2023 06 051.pdf': [4, 5],
'2023 06 045.pdf': [4, 5],
'2023 03 114.pdf': [4, 5],
'2023 01 039.pdf': [3, 4],
'2023 07 083.pdf': [4, 5],
'2023 01 011.pdf': [3, 4],
'2023 01 005.pdf': [3],
'2023 07 097.pdf': [5, 6],
'2023 07 068.pdf': [4, 5],
'2023 07 040.pdf': [4, 5],
'2023 02 111.pdf': [4, 5, 6],
'2023 02 105.pdf': [4, 5],
'2023 07 054.pdf': [4, 5],
'2023 07 078.pdf': [4, 5],
'2023 07 050.pdf': [4, 5],
'2023 02 101.pdf': [1, 4, 5],
```

```
'2023 02 115.pdf': [2, 3, 4],
'2023 07 044.pdf': [4],
'2023 07 093.pdf': [4, 5],
'2023 01 001.pdf': [4],
'2023 01 015.pdf': [1],
'2023 07 087.pdf': [4, 5],
'2023 06 069.pdf': [4, 5],
'2023 06 041.pdf': [4, 5],
'2023 06 055.pdf': [4, 5],
'2023 06 082.pdf': [4, 5],
'2023 05 141.pdf': [3],
'2023 06 096.pdf': [6, 7],
'2023 06 109.pdf': [4, 5],
'2023 03 064.pdf': [2, 3],
'2023 06 121.pdf': [4, 5],
'2023 03 070.pdf': [4],
'2023 05 009.pdf': [4, 5],
'2023 05 021.pdf': [4],
'2023 08 016.pdf': [4, 5],
'2023 07 118.pdf': [4, 5],
'2023 02 049.pdf': [4, 5],
'2023 07 124.pdf': [4, 5],
'2023 02 075.pdf': [1, 2, 3, 4, 5],
'2023 02 061.pdf': [4, 5],
'2023 07 130.pdf': [4, 5],
'2023 04 018.pdf': [5, 6],
'2023 04 024.pdf': [3, 4],
'2023 04 030.pdf': [4, 5],
'2023 04 031.pdf': [4, 5],
'2023 04 025.pdf': [3],
'2023 04 019.pdf': [5, 6],
'2023 02 060.pdf': [4, 5],
'2023 07 131.pdf': [4, 5],
'2023 07 125.pdf': [4, 5],
'2023 02 074.pdf': [4],
'2023 07 119.pdf': [4, 5],
'2023 02 048.pdf': [4, 5],
'2023 05 020.pdf': [4],
'2023 08 017.pdf': [4, 5],
'2023 08 003.pdf': [1, 2],
'2023 05 034.pdf': [4, 5],
'2023 05 008.pdf': [4, 5],
'2023 06 120.pdf': [4, 5],
'2023 03 071.pdf': [4],
'2023 06 108.pdf': [4, 5],
'2023 06 097.pdf': [4, 5],
'2023 05 140.pdf': [4],
```

```
'2023 06 083.pdf': [4, 5],
'2023 06 054.pdf': [4, 5],
'2023 06 040.pdf': [4, 5],
'2023 06 068.pdf': [4, 5],
'2023 03 139.pdf': [2],
'2022 12 024.pdf': [1, 2],
'2023 01 014.pdf': [4],
'2023 07 086.pdf': [4, 5],
'2023 07 092.pdf': [4, 5, 6],
'2023 02 114.pdf': [4, 5],
'2023 07 045.pdf': [4, 5],
'2023 07 051.pdf': [4, 5],
'2023 02 100.pdf': [3, 4],
'2023 07 079.pdf': [4, 5],
'2023 07 047.pdf': [4, 5],
'2023 02 116.pdf': [4, 5],
'2023 02 102.pdf': [4, 5],
'2023 07 053.pdf': [4],
'2023 07 084.pdf': [4, 5],
'2023 01 016.pdf': [4, 5],
'2023 01 002.pdf': [4, 5],
'2023 07 090.pdf': [5, 6],
'2022 12 026.pdf': [3],
'2023 06 056.pdf': [4, 5],
'2023 06 042.pdf': [4],
'2023 06 095.pdf': [5, 6],
'2023 06 081.pdf': [4, 5],
'2023 03 073.pdf': [2, 4, 5],
'2023 03 067.pdf': [5, 6],
'2023 08 015.pdf': [4, 5],
'2023 05 022.pdf': [4, 5],
'2023 05 036.pdf': [4, 5],
'2023 08 001.pdf': [4, 5],
'2023 07 133.pdf': [4],
'2023 02 062.pdf': [1, 3, 4, 5],
'2023 02 076.pdf': [4, 5],
'2023 07 127.pdf': [2],
'2023 04 033.pdf': [4, 5],
'2023 04 027.pdf': [4, 5],
'2023 04 026.pdf': [4, 5],
'2023 04 032.pdf': [4, 5],
'2023 02 088.pdf': [4, 5],
'2023 02 077.pdf': [3, 4],
'2023 07 126.pdf': [4, 5],
'2023 07 132.pdf': [1, 2],
'2023 02 063.pdf': [2, 4, 5],
'2023 05 037.pdf': [4, 5],
```

```
'2023 08 014.pdf': [4, 5],
'2023 08 028.pdf': [4, 5],
'2023 06 123.pdf': [6, 7],
'2023 05 143.pdf': [4, 5],
'2023 06 094.pdf': [5, 6, 10, 11, 14, 16],
'2023 06 043.pdf': [4, 5],
'2023 06 057.pdf': [4, 5],
'2022 12 027.pdf': [4, 5],
'2023 01 003.pdf': [3, 4],
'2023 07 091.pdf': [4, 5],
'2023 07 085.pdf': [4, 5],
'2023 01 017.pdf': [3],
'2023 02 103.pdf': [2, 3, 4, 5],
'2023 07 052.pdf': [4, 5],
'2023 07 046.pdf': [4, 5],
'2023 07 021.pdf': [4, 5],
'2023 07 035.pdf': [4],
'2023 07 009.pdf': [4, 5],
'2023 01 064.pdf': [3, 4, 5],
'2023 03 161.pdf': [4, 5],
'2023 03 175.pdf': [4, 5],
'2023 06 018.pdf': [7, 8, 9, 10],
'2023 05 130.pdf': [4, 5],
'2023 05 124.pdf': [4, 5],
'2023 05 118.pdf': [4, 5],
'2023 03 015.pdf': [3, 4],
'2023 05 087.pdf': [4, 5],
'2023 05 093.pdf': [4, 5],
'2023 03 001.pdf': [4, 5],
'2023 03 029.pdf': [2],
'2023 05 044.pdf': [3],
'2023 05 050.pdf': [4, 5],
'2023 05 078.pdf': [4, 5],
'2023 04 096.pdf': [3, 4],
'2023 02 004.pdf': [4],
'2023 02 010.pdf': [4, 5],
'2023 07 141.pdf': [4, 5],
'2023 04 082.pdf': [4, 5],
'2023 02 038.pdf': [4, 5],
'2023 04 055.pdf': [4, 5, 6],
'2023 04 041.pdf': [4, 5],
'2023 04 069.pdf': [4, 5],
'2023 04 068.pdf': [4, 5],
'2023 04 054.pdf': [4, 5],
'2023 02 039.pdf': [4, 5],
'2023 02 011.pdf': [4, 5],
'2023 04 083.pdf': [4, 5],
```

```
'2023 07 140.pdf': [4],
'2023 04 097.pdf': [4],
'2023 02 005.pdf': [4, 5],
'2023 05 079.pdf': [5],
'2023 05 051.pdf': [4, 5],
'2023 05 045.pdf': [4, 5],
'2023 03 028.pdf': [4, 5],
'2023 03 014.pdf': [5, 6],
'2023 05 086.pdf': [4, 5, 6],
'2023 05 119.pdf': [3],
'2023 05 125.pdf': [4, 5],
'2023 05 131.pdf': [4],
'2023 06 019.pdf': [4, 5],
'2023 03 148.pdf': [4, 5],
'2023 03 101. 102, 103, 104.pdf': [3],
'2023 06 025.pdf': [4, 5],
'2023 03 174.pdf': [4, 5],
'2023 03 160.pdf': [4],
'2023 01 059.pdf': [3, 4],
'2023 01 065.pdf': [4, 5],
'2023 01 071.pdf': [4, 5],
'2023 07 008.pdf': [4, 5],
'2023 07 034.pdf': [4, 5],
'2023 07 020.pdf': [4, 5],
'2023 07 036.pdf': [3],
'2023 07 022.pdf': [4, 5],
'2023 01 067.pdf': [5, 6],
'2023 03 176.pdf': [4, 5],
'2023 06 027.pdf': [4, 5, 6],
'2023 06 033.pdf': [4, 5],
'2023 03 162.pdf': [4, 5],
'2023 05 127.pdf': [4, 5],
'2023 05 133.pdf': [4, 5],
'2023 05 090.pdf': [5, 6],
'2023 03 016.pdf': [2, 3, 4],
'2023 05 053.pdf': [4, 5],
'2023 05 047.pdf': [4, 5],
'2023 07 142.pdf': [4, 5],
'2023 04 081.pdf': [3],
'2023 02 013.pdf': [1, 2, 4, 5],
'2023 04 095.pdf': [5, 6],
'2023 04 042.pdf': [3],
'2023 04 056.pdf': [4, 5],
'2023 04 057.pdf': [4, 5],
'2023 04 043.pdf': [4, 5],
'2023 02 006.pdf': [1, 3, 4, 5],
'2023 04 094.pdf': [4, 5],
```

```
'2023 04 080.pdf': [4],
'2023 07 143.pdf': [4, 5],
'2023 02 012.pdf': [1, 4],
'2023 05 046.pdf': [4, 5],
'2023 05 052.pdf': [5, 6],
'2023 05 085.pdf': [4, 5],
'2023 03 017.pdf': [2, 6, 8],
'2023 03 003.pdf': [2, 3, 4],
'2023 05 091.pdf': [4, 5],
'2023 05 132.pdf': [4],
'2023 05 126.pdf': [4, 5],
'2023 03 177.pdf': [4, 5, 6],
'2023 06 026.pdf': [4],
'2023 01 072.pdf': [3, 4, 5],
'2023 01 066.pdf': [1, 4, 5],
'2023 07 023.pdf': [4, 5],
'2023 07 037.pdf': [4],
'2023 07 033.pdf': [4],
'2023 07 027.pdf': [4, 5],
'2023 01 062.pdf': [4, 5],
'2023 01 076.pdf': [4],
'2023 03 167.pdf': [4, 5],
'2023 06 036.pdf': [1],
'2023 05 122.pdf': [4, 5],
'2023 05 136.pdf': [3, 4],
'2023 05 095.pdf': [4],
'2023 03 013.pdf': [4, 5],
'2023 05 081.pdf': [4, 5],
'2023 08 049.pdf': [4, 6],
'2023 05 042.pdf': [4, 5],
'2023 04 084.pdf': [4],
'2023 07 147.pdf': [4, 5],
'2023 04 090.pdf': [4, 5],
'2023 02 002.pdf': [4, 5],
'2023 04 047.pdf': [4, 5],
'2023 04 053.pdf': [4, 5],
'2023 04 052.pdf': [4, 5],
'2023 04 046.pdf': [4, 5],
'2023 04 091.pdf': [4, 5],
'2023 02 003.pdf': [2, 3, 4],
'2023 02 017.pdf': [4, 5],
'2023 04 085.pdf': [4, 5],
'2023 05 043.pdf': [4, 5],
'2023 05 057.pdf': [5, 6],
'2023 08 048.pdf': [4, 5],
'2023 05 080.pdf': [4, 5],
'2023 03 006.pdf': [4, 5],
```

```
'2023 05 137.pdf': [5, 6],
'2023 05 123.pdf': [5],
'2023 06 037.pdf': [4, 5],
'2023 06 023.pdf': [7, 8],
'2023 03 172.pdf': [4, 5],
'2023 01 077.pdf': [1, 4, 5],
'2023 01 063.pdf': [4, 5],
'2023 07 026.pdf': [4, 5],
'2023 07 032.pdf': [1],
'2023 07 018.pdf': [4, 5],
'2023 07 024.pdf': [4, 5],
'2023 07 030.pdf': [4, 5],
'2023 01 049.pdf': [3],
'2023 01 075.pdf': [1, 3, 4],
'2023 01 061.pdf': [4],
'2023 06 009.pdf': [4],
'2023 03 158.pdf': [4],
'2023 06 035.pdf': [4, 5],
'2023 03 164.pdf': [2, 4, 5],
'2023 03 170.pdf': [3, 4],
'2023 05 109.pdf': [4, 5],
'2023 05 135.pdf': [4, 5, 6, 7],
'2023 05 121.pdf': [4, 5],
'2023 03 004.pdf': [3, 4],
'2023 05 096.pdf': [5, 6],
'2023 05 069.pdf': [4],
'2023 05 041.pdf': [4, 5],
'2023 05 055.pdf': [4, 5],
'2023 02 029.pdf': [3, 4],
'2023 04 093.pdf': [4, 5],
'2023 07 144.pdf': [1],
'2023 02 015.pdf': [4, 5],
'2023 04 078.pdf': [4, 5],
'2023 04 050.pdf': [4, 5],
'2023 04 044.pdf': [5],
'2023 04 045.pdf': [4, 5],
'2023 04 051.pdf': [4, 5],
'2023 04 079.pdf': [4, 5],
'2023 04 086.pdf': [4, 5],
'2023 04 092.pdf': [4, 5],
'2023 02 028.pdf': [4, 5],
'2023 05 054.pdf': [5, 6],
'2023 05 040.pdf': [4, 5],
'2023 05 068.pdf': [4],
'2023 03 005.pdf': [4, 5],
'2023 05 097.pdf': [4, 5],
'2023 05 083.pdf': [4, 5],
```

```
                '2023 03 011.pdf': [4],
                '2023 03 039.pdf': [1, 2, 4, 5],
                '2023 05 120.pdf': [4, 5],
                '2023 05 134.pdf': [4, 5],
                '2023 05 108.pdf': [4, 5],
                '2023 03 171.pdf': [4, 5],
                '2023 06 020.pdf': [4, 5],
                '2023 06 008.pdf': [6, 7],
                '2023 03 159.pdf': [4, 5],
                '2023 01 060.pdf': [4, 5],
                '2023 01 074.pdf': [1, 4, 5],
                '2023 07 031.pdf': [1, 2],
                '2023 07 025.pdf': [4, 5],
                '2023 07 019.pdf': [4]}),
 'errors': {}}
```

Finalmente, contamos la cantidad de páginas de nuestra muestra que serán analizadas en Azure, las cuales totalizan 2,284

[ ]:
```
page_count('./files/sample')
```

```
Files processed: 100%|      | 810/810 [00:00<00:00, 1109.38file/s]
```

[ ]:
```
2284
```

0.1.2 2. Análisis de los Documentos

2.1. Análisis y Extracción En esta sección se estará enviando cada uno de los documentos al servicio de Azure y luego se procederá a extraer la información de estos.

[ ]:
```
class DataExtractor:
    """
    Esta clase tiene como propósito extraer los datos del
    json devuelto por el servicio de Azure.
    """

    page_num = None
    key_value_pairs: list[dict] = []
    tables: list[dict] = []

    def __init__(self, data) -> None:
        self.key_value_pairs = data['key_value_pairs']
        self.tables = data['tables']

    def __filter_by_content_fn (self,page_num: int, keyword: str) -> Callable:

        return lambda item: re.search(keyword, item['key']['content'].lower())␣
↪is not None \
            and item['key']['bounding_regions'][0]['page_number'] == page_num
```

```
  def __filter_table(self, page_number: int) -> dict|None:
      results = list(filter(
          lambda table: table['bounding_regions'][0]['page_number'] ==␣
↪page_number,
          self.tables))

      if len(results) == 0:
          return None

      return results[0]

  def __extract_headers(self, cells: list[dict]) -> list[str]:
      filtered_cells =  list(filter(lambda item: 'kind' in item and␣
↪item['kind'] == 'columnHeader', cells ))
      headers = []

      for header in filtered_cells:
          # No nos interesan las columnas combinadas
          if 'column_span' in header and header['column_span'] >= 2:
              continue

          headers.insert(header['column_index'], header['content'])

      return headers

  def __extract_rows(self, cells: list[dict], columnCount: int) ->␣
↪list[str|float|int]:
      filtered_cells = list(filter(lambda item: 'kind' in item and␣
↪item['kind'] == 'content', cells))
      rows = []
      row = []

      for index,cell in enumerate(filtered_cells):

          if index >0  and (index % columnCount) == 0:

              rows.append(row)
              row = []

          row.insert(cell['column_index'], cell['content'])

      return rows

  def extract_client_name (self, page_number: int) -> str:
```

```
        results = list(filter(self.__filter_by_content_fn(page_number,␣
↪r"cliente[\s|\/\\n]+client"), self.key_value_pairs))

        if len(results) == 0:
            return ''

        return results[0]['value']['content']

    def extract_test_date (self, page_number: int) -> datetime|str:
        results = list(filter(
            self.__filter_by_content_fn(page_number, r"date-time"), self.
↪key_value_pairs))

        if len(results) == 0:
            return ''
        date = re.match(
            r"(\d{1,2}/\d{1,2}/\d{4}\s\d{1,2}:\d{2}(?:\s[ap]\.m)?)",
            results[0]['value']['content'])[0]
        try:
            return parse(date)
        except ParserError:
            return parse(re.sub(r'p\.m|a\.m', '',date))

    def extract_product_name(self, page_number: int) -> str:

        results = list(filter(self.__filter_by_content_fn(page_number,␣
↪'producto'), self.key_value_pairs))

        if len(results) == 0:
            return ''

        return results[0]['value']['content']

    def extract_table(self, page_num:int) -> pd.DataFrame:
        table_items =  self.__filter_table(page_num)
        if table_items is None:
            return pd.DataFrame

        headers = self.__extract_headers(table_items['cells'])
        values = self.__extract_rows(
            table_items['cells'], len(headers))

        df = pd.DataFrame(values, columns=headers)

        return df.assign(
            CLIENTE = self.extract_client_name(page_num),
            PRODUCTO = self.extract_product_name(page_num),
```

```
            FECHA = self.extract_test_date(page_num))

    def extract_tables(self) -> pd.DataFrame:
        df = pd.DataFrame()

        for table in self.tables:
            df1 = self.
↪extract_table(table['bounding_regions'][0]['page_number'])
            df1['NUM PAGINA'] = table['bounding_regions'][0]['page_number']
            df = pd.concat([df, df1], ignore_index=True)

        return df
```

[ ]:
```
class DocumentAnalyzer:
    """
    Esta clase se encarga de leer cada uno de los archivos
    en el directorio especificado y enviarlos al servicio de
    Azure
    """

    document_analysis_client: DocumentAnalysisClient|None = None
    processed_files: list[str] = []
    errors: list[dict[str,str|BaseException]] = []

    def __init__(self) -> None:

        load_dotenv(override=True)
        endpoint = os.environ["AZURE_FORM_RECOGNIZER_ENDPOINT"]
        key = os.environ["AZURE_FORM_RECOGNIZER_KEY"]

        self.document_analysis_client = DocumentAnalysisClient(
        endpoint=endpoint, credential=AzureKeyCredential(key)
    )

    def __file_reader(self,dirname: str) -> Generator[BufferedReader, None,␣
↪None]:

        for name in tqdm(os.listdir(dirname), unit='file', desc='Files␣
↪processed: '):
            with open(os.path.join(dirname, name), "rb") as f:
                yield f

    def __get_analysis(self, file: BufferedReader) -> dict[str, str]:

        poller = self.document_analysis_client.begin_analyze_document(
                    "prebuilt-document", document=file
```

```
                )
        return poller.result().to_dict()

    def __write_results(self,filename: str, results: dict[str, str], output_dir:
↪ str, log_dir = './var') -> None:

        with open (os.path.join(output_dir, filename + '.json'), 'w') as␣
↪json_file, \
             open(os.path.join(log_dir, 'checkpoint.txt'), 'w') as log_file:

             json_file.write(json.dumps(results))
             log_file.write(json.dumps(processed_files))

    def analyze(self,input_dir: str, output_dir: str, log_dir = './var') -> pd.
↪DataFrame:

        df = pd.DataFrame()
        self.processed_files = []
        self.errors = []

        for file in self.__file_reader(input_dir):

            try:
                filename = Path(output_dir, file.name).stem

                result =  self.__get_analysis(file)

                extractor = DataExtractor(result)
                extracted_df = extractor.extract_tables().
↪assign(ARCHIVO=filename + '.pdf')

                df = pd.concat([df,extracted_df], ignore_index = True)

                self.processed_files.append(file.name)

            except (HttpResponseError, Exception) as e:
                error = {
                    'file': file.name,
                    'error': str(e),
                    'traceback': traceback.extract_tb(e.__traceback__).format()}

                if e is HttpResponseError:
                    error['comment'] = 'Archivo no procesado'
                    result = None

                self.errors.append(error)
```

```
            finally:
                if result is not None:
                    self.__write_results(filename, result, output_dir, log_dir)
        return df
```

Procedemos a realizar el análisis de los documentos y extraer la información. En total se procesarán 809 archivos.

[ ]:
```
analyzer = DocumentAnalyzer()
df = analyzer.analyze(sample_dir, json_dir)
```

```
Files processed: 100%|      | 809/809 [3:35:36<00:00, 15.99s/file]
```

Hacemos una copia del dataframe original para fines de respaldo.

[ ]:
```
df_copy = df.copy()
```

En total se tiene más de 31,000 filas de información

[ ]:
```
len(df)
```

[ ]:
```
31470
```

Como se puede ver a continuación, algunas columnas están duplicadas, o bien, parte del texto del archivo fue leído como una columna.

[ ]:
```
df.columns
```

[ ]:
```
Index(['ANÁLISIS / ANALYSES', 'UNIDADES / UNIT', 'MIN', 'MAX',
       'MÉTODO / METHOD', 'RESULTADOS DE CALIDAD / QUALITY RESULTS', 'CLIENTE',
       'PRODUCTO', 'FECHA', 'NUM PAGINA', 'ARCHIVO', '',
       'INCERTIDUMBRE/ UNCERTAINTY %', 'MEIN', '  ', 'METODO / METHOD',
       '.MAX', '-MAX', 'DICERTIDUMBRE7. UNCERTAINTY %.', 'MEN', ': MAX',
       'BIIN', 'PRODUCTO / PRODUCT', 'ISLA #2', ': GASOLINA REGULAR', 'N/A',
       '. MAX', 'MĒTODO / METHOD', 'METODO / METIIOD', '.. ISLA #2',
       'Producto(s) / Product(s)', 'GASOLINA REGULAR GASOIL REGULAR',
       'VESSEL NAME (Nombre del Buque)', 'NAVIG8 STRENGTH',
       'VESSEL REPORT / (REPORTE\nPRODUCT', 'DEL BUQUE)\n(PRODUCTO)',
       'GASOLINA REGULAR', 'GASOIL REGULAR',
       'OVERALL ANALYSIS (ANÁLISIS TOTAL)',
       'PUERTO DE CARGA) (CIFRAS EN EL LOAD PORT B/L',
       '(CIFRAS EN EL PUERTO DE DESCARGA) VESSEL',
       '(GANANCIA / PERDIDA) GAIN/LOSS', ': N/A', 'BIEN',
       ': GASOLINA RACIAL.AR', 'MÁX', 'UNCERTAINTY %', ''MAX',
       'METODO / METITOD', '·MAX', 'AFIN', 'METODO / METTIOD', 'UNIT',
       'UNDDADES / UNIT', 'ISLA #4', 'MĒTODO / METHOD', 'UNDDADES UNIT'],
      dtype='object')
```

Para limpiar el dataframe, se define un mapa con palabras claves de las columnas repetidas e indicando el nombre correcto de la columna.

[ ]:
```
column_map = {
    'UNIDADES': 'UNIDADES / UNIT',
    'UNIT': 'UNIDADES / UNIT',
    'CERTIDUMBRE': 'INCERTIDUMBRE/ UNCERTAINTY %',
    'RESULTADOS': 'RESULTADOS DE CALIDAD / QUALITY RESULTS',
    'MÉTODO': 'MÉTODO / METHOD',
    'METODO': 'MÉTODO / METHOD',
    'MI': 'MIN',
    'MA': 'MAX',
    'ANALYSES': 'ANÁLISIS / ANALYSES'
}
```

Usando el método "combine_first" de pandas, movemos los datos de las columnas duplicadas a la columna que le corresponde. Posteriormente se procede a eliminar la columna duplicada.

[ ]:
```
for keyword,column_name in column_map.items():

    try:
        target_cols = list(filter(lambda col: col.find(keyword) >= 0 or \
            unidecode(col).find(keyword) >= 0, df.columns.to_list()))

        target_cols.remove(column_name)

        for col in target_cols:
            df[column_name].combine_first(df[col])
            df.drop(col, axis = 1, inplace=True)
    except ValueError as e:
        print(f'{column_name} not in list')
```

De las 176 columnas que existían, ahora quedan cerca de 40. Como se observa, las columnas 11 y de la 13 en adelante pueden eliminarse, pues la mayoría de las filas de estas columnas contienen valores nulos.

Por otro lado, la columna 'Fecha' está siendo detectada como tipo objeto, cuando debería ser datetime.

[ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31470 entries, 0 to 31469
Data columns (total 40 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   ANÁLISIS / ANALYSES                     30799 non-null  object
 1   UNIDADES / UNIT                         29354 non-null  object
 2   MIN                                     30832 non-null  object
 3   MAX                                     31079 non-null  object
 4   MÉTODO / METHOD                         30023 non-null  object
 5   RESULTADOS DE CALIDAD / QUALITY RESULTS 30255 non-null  object
```

```
 6   CLIENTE                                          31470 non-null  object
 7   PRODUCTO                                         31470 non-null  object
 8   FECHA                                            31470 non-null  object
 9   NUM PAGINA                                       31470 non-null  int64
 10  ARCHIVO                                          31470 non-null  object
 11                                                   173 non-null    object
 12  INCERTIDUMBRE/ UNCERTAINTY %                     856 non-null    object
 13  MEIN                                             2 non-null      object
 14                                                  22 non-null     object
 15  MEN                                              91 non-null     object
 16  BIIN                                             22 non-null     object
 17  PRODUCTO / PRODUCT                               140 non-null    object
 18  ISLA #2                                          20 non-null     object
 19  : GASOLINA REGULAR                               70 non-null     object
 20  N/A                                              70 non-null     object
 21  .. ISLA #2                                       10 non-null     object
 22  Producto(s) / Product(s)                         20 non-null     object
 23  GASOLINA REGULAR GASOIL REGULAR                  20 non-null     object
 24  VESSEL NAME (Nombre del Buque)                   10 non-null     object
 25  NAVIG8 STRENGTH                                  10 non-null     object
 26  VESSEL REPORT / (REPORTE
PRODUCT              21 non-null     object
 27  DEL BUQUE)
(PRODUCTO)                          21 non-null     object
 28  GASOLINA REGULAR                                 41 non-null     object
 29  GASOIL REGULAR                                   21 non-null     object
 30  OVERALL ANALYSIS (ANÁLISIS TOTAL)                20 non-null     object
 31  PUERTO DE CARGA) (CIFRAS EN EL LOAD PORT B/L     20 non-null     object
 32  (CIFRAS EN EL PUERTO DE DESCARGA) VESSEL         20 non-null     object
 33  (GANANCIA / PERDIDA) GAIN/LOSS                   20 non-null     object
 34  : N/A                                            10 non-null     object
 35  BIEN                                             23 non-null     object
 36  : GASOLINA RACIAL.AR                             10 non-null     object
 37  UNCERTAINTY %                                    22 non-null     object
 38  AFIN                                             2 non-null      object
 39  ISLA #4                                          20 non-null     object
dtypes: int64(1), object(39)
memory usage: 9.6+ MB
```

Convertimos la columna fecha al tipo datetime

[ ]:
```
df['FECHA'] = pd.to_datetime(df['FECHA'])
df['ANÁLISIS / ANALYSES'] = df['ANÁLISIS / ANALYSES'].astype(str)
```

Eliminamos las columnas 11 y 13 en adelante

[ ]:
```
df.drop(df.columns[np.r_[11, 13:len(df.columns)]], axis=1, inplace=True)
```

A continuación, se presenta la tabla final.

[ ]:
```
df.head(10)
```

[ ]:

	ANÁLISIS / ANALYSES	UNIDADES / UNIT	MIN	MAX
0	NUMERO DE OCTANO METODO RESEARCH (RON)	-	95	-
1	NUMERO DE OCTANO METODO MOTOR (MON)	.	82	-
2	CONTENIDO DE PLOMO	G/gal	1	0.02
3	PRESION A VAPOR REID (RVP) @ 100 º F	PSI	1	10.0
4	RVP + 0.1 E 70 ° C	1	Reportar	
5	DESTILACION 10% VOL. RECUPERADO	0	,	75
6	DESTILACION 50% VOL. RECUPERADO	·℃		121
7	DESTILACION 90% VOL. RECUPERADO	°℃	1	190
8	PUNTO FINAL	℃		225
9	RESIDUO	% Vol.		2,0

	MÉTODO / METHOD	RESULTADOS DE CALIDAD / QUALITY RESULTS
0	ASTM D-2699	96.0
1	ASTM D-2700	87.4
2	ASTM D-3237	N/D
3	ASTM D-323	8.02
4	Calculo	9.74
5	ASTM D86	62
6	ASTM D86	110
7	ASTM D86	159
8	ASTM D86	202
9	ASTM D86	0.5

	CLIENTE	PRODUCTO	FECHA
0	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
1	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
2	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
3	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
4	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
5	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
6	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
7	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
8	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00
9	ATLANTIC BONAO\n(ANIANA)	GASOLINA PREMIUM	2023-04-07 15:12:00

	NUM PAGINA	ARCHIVO	INCERTIDUMBRE/ UNCERTAINTY %
0	1	2023 07 014.pdf	NaN
1	1	2023 07 014.pdf	NaN
2	1	2023 07 014.pdf	NaN
3	1	2023 07 014.pdf	NaN
4	1	2023 07 014.pdf	NaN
5	1	2023 07 014.pdf	NaN
6	1	2023 07 014.pdf	NaN
7	1	2023 07 014.pdf	NaN

```
8           1  2023 07 014.pdf                          NaN
9           1  2023 07 014.pdf                          NaN
```

Guardamos una copia de la tabla en formato parquet y csv.

```
[ ]: df.to_parquet('./gas_station_dataset.parquet')
```

```
[ ]: df.to_csv('./gas_station_dataset.csv')
```

2.2. Análisis de los Errores Un total de 99 archivos no pudieron ser procesados. Se procederá a revisar el log de los errores encontrados.

```
[ ]: len(analyzer.errors)
```

```
[ ]: 99
```

```
[ ]: errors = []

for item in analyzer.errors:
    errors.append({
        'file': item['file'],
        'error': str(item['error']),
        'traceback': traceback.extract_tb(item['error'].__traceback__).
  ↪format()})
```

Revisando los errores, al parecer algunos datos de la columna fecha no fueron extraidos correctamente. Es posible que esto se deba a un problema en el formato de fecha como se encuentra en el reporte.

```
[ ]: print(json.dumps(errors[:5], indent=4))
```

```
[
    {
        "file": "./files/sample/2023 01 079.pdf",
        "error": "'NoneType' object is not subscriptable",
        "traceback": [
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/3819070703.py\", line 58, in analyze\n    extracted_df =
extractor.extract_tables().assign(ARCHIVO=filename + '.pdf')\n
^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 108, in extract_tables\n    df1 =
self.extract_table(table['bounding_regions'][0]['page_number'])\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 102, in extract_table\n    FECHA =
self.extract_test_date(page_num))\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
```

```
_58275/1914603399.py\", line 71, in extract_test_date\n    date = re.match(\n
^^^^^^^^^\n"
        ]
    },
    {
        "file": "./files/sample/2023 02 024.pdf",
        "error": "'NoneType' object is not subscriptable",
        "traceback": [
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/3819070703.py\", line 58, in analyze\n    extracted_df =
extractor.extract_tables().assign(ARCHIVO=filename + '.pdf')\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 108, in extract_tables\n    df1 =
self.extract_table(table['bounding_regions'][0]['page_number'])\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 102, in extract_table\n    FECHA =
self.extract_test_date(page_num))\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 71, in extract_test_date\n    date = re.match(\n
^^^^^^^^^\n"
        ]
    },
    {
        "file": "./files/sample/2023 08 053.pdf",
        "error": "'NoneType' object is not subscriptable",
        "traceback": [
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/3819070703.py\", line 58, in analyze\n    extracted_df =
extractor.extract_tables().assign(ARCHIVO=filename + '.pdf')\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 108, in extract_tables\n    df1 =
self.extract_table(table['bounding_regions'][0]['page_number'])\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 102, in extract_table\n    FECHA =
self.extract_test_date(page_num))\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 71, in extract_test_date\n    date = re.match(\n
^^^^^^^^^\n"
        ]
    },
    {
        "file": "./files/sample/2023 05 110.pdf",
```

```
        "error": "'NoneType' object is not subscriptable",
        "traceback": [
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/3819070703.py\", line 58, in analyze\n    extracted_df =
extractor.extract_tables().assign(ARCHIVO=filename + '.pdf')\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 108, in extract_tables\n    df1 =
self.extract_table(table['bounding_regions'][0]['page_number'])\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 102, in extract_table\n    FECHA =
self.extract_test_date(page_num))\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 71, in extract_test_date\n    date = re.match(\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n"
        ]
    },
    {
        "file": "./files/sample/2023 04 101.pdf",
        "error": "'NoneType' object is not subscriptable",
        "traceback": [
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/3819070703.py\", line 58, in analyze\n    extracted_df =
extractor.extract_tables().assign(ARCHIVO=filename + '.pdf')\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 108, in extract_tables\n    df1 =
self.extract_table(table['bounding_regions'][0]['page_number'])\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 102, in extract_table\n    FECHA =
self.extract_test_date(page_num))\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
            "  File \"/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel
_58275/1914603399.py\", line 71, in extract_test_date\n    date = re.match(\n
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n"
        ]
    }
]
```


Procedemos a guardar el reporte de los errores en un archivo json para poder analizarlos más adelante.

[ ]:
```python
with open('./var/errors_log.json', 'w') as f:
   json.dump(errors, f,indent=4, sort_keys=True)
```

2.3. Verificación de Calidad En esta sección tomaremos una muestra aleatoria para revisar manualmente los datos extraidos con los archivos originales, y así asegurarnos que los datos sean correctos. Para determinar el tamaño de muestra, así como los criterios de aceptación o rechazo, se procedió a usar la tabla ANSI Z1.4 con un nivel de inspección general I, un tamaño de lote de 700 archivos y un AQL de 0.40.

Por limitaciones de recursos, la verificación de la muestra contra los archivos originales se limitará a los valores RON y MON solamente.

[ ]:
```
sample = df['ARCHIVO'].sample(n = 32, random_state=42)
```

[ ]:
```
sample_df = df[df['ARCHIVO'].isin(sample)][df['ANÁLISIS / ANALYSES'].str.
↪contains('RON') | df['ANÁLISIS / ANALYSES'].str.contains('MON')]
```

```
/var/folders/gv/t0xf0frn51g8g_xd29ccdwzc0000gn/T/ipykernel_58275/831347874.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  sample_df = df[df['ARCHIVO'].isin(sample)][df['ANÁLISIS /
ANALYSES'].str.contains('RON') | df['ANÁLISIS / ANALYSES'].str.contains('MON')]
```

Resultados de la verificación: PENDIENTE

[ ]:
```
sample_df[20:54]
```

[ ]:

	ANÁLISIS / ANALYSES	UNIDADES / UNIT	MIN	MAX	\
6620	NUMERO DE OCTANO METODO RESEARCH (RON)		89	.	
6621	NUMERO DE OCTANO METODO MOTOR (MON)	.	76		
6642	NUMERO DE OCTANO METODO RESEARCH (RON)		89		
6643	NUMERO DE OCTANO METODO MOTOR (MON)		76	-	
7793	NUMERO DE OCTANO METODO RESEARCH (RON)		95		
7794	NUMERO DE OCTANO METODO MOTOR (MON)		82	.	
7815	NUMERO DE OCTANO METODO RESEARCH (RON)		89	.	
7816	NUMERO DE OCTANO METODO MOTOR (MON)	1	76		
7837	NUMERO DE OCTANO METODO RESEARCH (RON)		89	.	
7838	NUMERO DE OCTANO METODO MOTOR (MON)	-	76	-	
8407	NUMERO DE OCTANO METODO RESEARCH (RON)		95		
8408	NUMERO DE OCTANO METODO MOTOR (MON)	,	82		
8429	NUMERO DE OCTANO METODO RESEARCH (RON)		95.0		
8430	NUMERO DE OCTANO METODO MOTOR (MON)		82.0		
8431	NUMERO DE OCTANO METODO RESEARCH (RON)		89		
8432	NUMERO DE OCTANO METODO MOTOR (MON)		76	.	
8935	NUMERO DE OCTANO METODO RESEARCH (RON)	.	95		
8936	NUMERO DE OCTANO METODO MOTOR (MON)		82	.	
8957	NUMERO DE OCTANO METODO RESEARCH (RON)	1	89	.	
8958	NUMERO DE OCTANO METODO MOTOR (MON)	.	76		
8979	NUMERO DE OCTANO METODO RESEARCH (RON)	.	89	1	
8980	NUMERO DE OCTANO METODO MOTOR (MON)		76	-	
9170	NUMERO DE OCTANO METODO RESEARCH (RON)		95	.	
9171	NUMERO DE OCTANO METODO MOTOR (MON)		82		
9192	NUMERO DE OCTANO METODO RESEARCH (RON)		89		

9193	NUMERO DE OCTANO METODO MOTOR (MON)		76	-
9214	NUMERO DE OCTANO METODO RESEARCH (RON)		89	
9215	NUMERO DE OCTANO METODO MOTOR (MON)	-	76	-
9379	NUMERO DE OCTANO METODO RESEARCH (RON)	.	95	.
9380	NUMERO DE OCTANO METODO MOTOR (MON)	.	82	.
9401	NUMERO DE OCTANO METODO RESEARCH (RON)		89	
9402	NUMERO DE OCTANO METODO MOTOR (MON)		76	
9423	NUMERO DE OCTANO METODO RESEARCH (RON)		89	1
9424	NUMERO DE OCTANO METODO MOTOR (MON)		76	-

	MÉTODO / METHOD	RESULTADOS DE CALIDAD / QUALITY RESULTS	\
6620	ASTM D-2699	92.8	
6621	ASTM D-2700	81.9	
6642	ASTM D-2699	93.2	
6643	ASTM D-2700	84.2	
7793	ASTM D-2699	93.6	
7794	ASTM D-2700	84.9	
7815	ASTM D-2699	91.5	
7816	ASTM D-2700	81.2	
7837	ASTM D-2699	91.5	
7838	ASTM D-2700	81.4	
8407	ASTM D-2699	95.2	
8408	ASTM D-2700	86.1	
8429	ASTM D-2699	95.4	
8430	ASTM D-2700	86.5	
8431	ASTM D-2699	91.3	
8432	ASTM D-2700	82.4	
8935	ASTM D-2699	96.0	
8936	ASTM D-2700	87.6	
8957	ASTM D-2699	91.6	
8958	ASTM D-2700	82.3	
8979	ASTM D-2699	92.1	
8980	ASTM D-2700	82.4	
9170	ASTM D-2699	94.4	
9171	ASTM D-2700	85.0	
9192	ASTM D-2699	90.8	
9193	ASTM D-2700	81.8	
9214	ASTM D-2699	90.6	
9215	ASTM D-2700	81.6	
9379	ASTM D-2699	95.0	
9380	ASTM D-2700	86.2	
9401	ASTM D-2699	91.7	
9402	ASTM D-2700	82.2	
9423	ASTM D-2699	91.5	
9424	ASTM D-2700	82.1	

	CLIENTE	PRODUCTO	\

```
6620                                NEXT LICEY  GASOLINA REGULAR
6621                                NEXT LICEY  GASOLINA REGULAR
6642                                NEXT LICEY  GASOLINA REGULAR
6643                                NEXT LICEY  GASOLINA REGULAR
7793      TOTAL VISTA DE JIMENOA\nDÍA ANÁLISIS  GASOLINA PREMIUM
7794      TOTAL VISTA DE JIMENOA\nDÍA ANÁLISIS  GASOLINA PREMIUM
7815                   TOTAL VISTA DE JIMENOA  GASOLINA REGULAR
7816                   TOTAL VISTA DE JIMENOA  GASOLINA REGULAR
7837                   TOTAL VISTA DE JIMENOA  GASOLINA REGULAR
7838                   TOTAL VISTA DE JIMENOA  GASOLINA REGULAR
8407                        ECOPETROLEO GUASA  GASOLINA PREMIUM
8408                        ECOPETROLEO GUASA  GASOLINA PREMIUM
8429                        ECOPETROLEO GUASA  GASOLINA PREMIUM
8430                        ECOPETROLEO GUASA  GASOLINA PREMIUM
8431                        ECOPETROLEO GUASA  GASOLINA REGULAR
8432                        ECOPETROLEO GUASA  GASOLINA REGULAR
8935                   ECOPETROLEO DOÑA HILDA  GASOLINA PREMIUM
8936                   ECOPETROLEO DOÑA HILDA  GASOLINA PREMIUM
8957        : ECOPETROLEO DOÑA HILDA\nANÁLISIS  GASOLINA REGULAR
8958        : ECOPETROLEO DOÑA HILDA\nANÁLISIS  GASOLINA REGULAR
8979      ECOPETROLEO DOÑA HILDA\nDÍA ANÁLISIS  GASOLINA REGULAR
8980      ECOPETROLEO DOÑA HILDA\nDÍA ANÁLISIS  GASOLINA REGULAR
9170                             SHELL ABENSA  GASOLINA PREMIUM
9171                             SHELL ABENSA  GASOLINA PREMIUM
9192                             SHELL ABENSA  GASOLINA REGULAR
9193                             SHELL ABENSA  GASOLINA REGULAR
9214                             SHELL ABENSA  GASOLINA REGULAR
9215                             SHELL ABENSA  GASOLINA REGULAR
9379   ECOPETROLEO HNOS CONTRERAS DÍA ANÁLISIS  GASOLINA PREMIUM
9380   ECOPETROLEO HNOS CONTRERAS DÍA ANÁLISIS  GASOLINA PREMIUM
9401   ECOPETROLEO HNS CONTRERAS\nDÍA ANÁLISIS  GASOLINA REGULAR
9402   ECOPETROLEO HNS CONTRERAS\nDÍA ANÁLISIS  GASOLINA REGULAR
9423  ECOPETROLEO HNOS. CONTRERAS DÍA ANÁLISIS  GASOLINA REGULAR
9424  ECOPETROLEO HNOS. CONTRERAS DÍA ANÁLISIS  GASOLINA REGULAR

                    FECHA  NUM PAGINA          ARCHIVO  \
6620 2023-03-27 18:50:00           2  2023 03 144.pdf
6621 2023-03-27 18:50:00           2  2023 03 144.pdf
6642 2023-03-27 18:50:00           3  2023 03 144.pdf
6643 2023-03-27 18:50:00           3  2023 03 144.pdf
7793 2023-03-04 07:28:00           1  2023 04 003.pdf
7794 2023-03-04 07:28:00           1  2023 04 003.pdf
7815 2023-03-04 07:28:00           2  2023 04 003.pdf
7816 2023-03-04 07:28:00           2  2023 04 003.pdf
7837 2023-03-04 07:28:00           3  2023 04 003.pdf
7838 2023-03-04 07:28:00           3  2023 04 003.pdf
8407 2023-06-26 19:05:00           1  2023 06 107.pdf
```

```
8408 2023-06-26 19:05:00           1  2023 06 107.pdf
8429 2023-06-26 19:05:00           2  2023 06 107.pdf
8430 2023-06-26 19:05:00           2  2023 06 107.pdf
8431 2023-06-26 19:05:00           3  2023 06 107.pdf
8432 2023-06-26 19:05:00           3  2023 06 107.pdf
8935 2023-11-07 18:56:00           1  2023 07 048.pdf
8936 2023-11-07 18:56:00           1  2023 07 048.pdf
8957 2023-11-07 18:56:00           2  2023 07 048.pdf
8958 2023-11-07 18:56:00           2  2023 07 048.pdf
8979 2023-11-07 18:56:00           3  2023 07 048.pdf
8980 2023-11-07 18:56:00           3  2023 07 048.pdf
9170 2023-06-16 16:11:00           1  2023 06 071.pdf
9171 2023-06-16 16:11:00           1  2023 06 071.pdf
9192 2023-06-16 16:11:00           2  2023 06 071.pdf
9193 2023-06-16 16:11:00           2  2023 06 071.pdf
9214 2023-06-16 16:11:00           3  2023 06 071.pdf
9215 2023-06-16 16:11:00           3  2023 06 071.pdf
9379 2023-03-05 19:09:00           1  2023 05 011.pdf
9380 2023-03-05 19:09:00           1  2023 05 011.pdf
9401 2023-03-05 19:07:00           2  2023 05 011.pdf
9402 2023-03-05 19:07:00           2  2023 05 011.pdf
9423 2023-03-05 19:09:00           3  2023 05 011.pdf
9424 2023-03-05 19:09:00           3  2023 05 011.pdf

     INCERTIDUMBRE/ UNCERTAINTY %
6620                          NaN
6621                          NaN
6642                          NaN
6643                          NaN
7793                          NaN
7794                          NaN
7815                          NaN
7816                          NaN
7837                          NaN
7838                          NaN
8407                          NaN
8408                          NaN
8429                          NaN
8430                          NaN
8431                          NaN
8432                          NaN
8935                          NaN
8936                          NaN
8957                          NaN
8958                          NaN
8979                          NaN
8980                          NaN
```

```
9170                          NaN
9171                          NaN
9192                          NaN
9193                          NaN
9214                          NaN
9215                          NaN
9379                          NaN
9380                          NaN
9401                          NaN
9402                          NaN
9423                          NaN
9424                          NaN
```

0.1.3 3. Análisis de los Resultados

3.1. Preparación del Análisis En esta sección nos limitaremos a analizar los datos del RON y MON para las gasolinas del tipo regular y premium. En primer lugar, haremos una copia del dataframe conteniendo solo los datos del RON y MON.

Limitaciones: Es importante destacar que este análisis está basado en los resultados de 700 reportes, ya que cerca de 100 presentaron error al procesarse.

[ ]:
```
df_ron_mon = df[df['ANÁLISIS / ANALYSES'].str.contains(r'\(RON\)') |␣
↪df['ANÁLISIS / ANALYSES'].str.contains(r'\(MON\)')].copy()
```

[ ]:
```
df_ron_mon.head(5)
```

[ ]:
```
                       ANÁLISIS / ANALYSES UNIDADES / UNIT   MIN MAX  \
0   NUMERO DE OCTANO METODO RESEARCH (RON)               -    95   -
1      NUMERO DE OCTANO METODO MOTOR (MON)               .    82   -
22  NUMERO DE OCTANO METODO RESEARCH (RON)                  95.0
23     NUMERO DE OCTANO METODO MOTOR (MON)                  82.0   1
24  NUMERO DE OCTANO METODO RESEARCH (RON)               -    89

   MÉTODO / METHOD RESULTADOS DE CALIDAD / QUALITY RESULTS  \
0      ASTM D-2699                                    96.0
1      ASTM D-2700                                    87.4
22     ASTM D-2699                                    96.3
23     ASTM D-2700                                    87.7
24     ASTM D-2699                                    91.8

                      CLIENTE          PRODUCTO               FECHA  \
0    ATLANTIC BONAO\n(ANIANA)  GASOLINA PREMIUM 2023-04-07 15:12:00
1    ATLANTIC BONAO\n(ANIANA)  GASOLINA PREMIUM 2023-04-07 15:12:00
22  ATLANTIC BONAO I (ANIANA)  GASOLINA PREMIUM 2023-04-07 15:12:00
23  ATLANTIC BONAO I (ANIANA)  GASOLINA PREMIUM 2023-04-07 15:12:00
24  ATLANTIC BONAO I (ANIANA)  GASOLINA REGULAR 2023-04-07 15:12:00
```

	NUM PAGINA	ARCHIVO INCERTIDUMBRE/ UNCERTAINTY %
0	1	2023 07 014.pdf NaN
1	1	2023 07 014.pdf NaN
22	2	2023 07 014.pdf NaN
23	2	2023 07 014.pdf NaN
24	3	2023 07 014.pdf NaN

Ahora nos aseguramos de tener valores en formato número para poder realizar un análisis descriptivo de estos.

[ ]:
```
df_ron_mon['RESULTADOS DE CALIDAD / QUALITY RESULTS'] = df_ron_mon['RESULTADOS␣
↪DE CALIDAD / QUALITY RESULTS'].str.replace(',', '.').replace('N/A', None).
↪astype(float)
```

[ ]:
```
df_ron_mon['RESULTADOS DE CALIDAD / QUALITY RESULTS'].describe()
```

[ ]:
```
count    3778.000000
mean       88.685416
std         5.088703
min        79.500000
25%        84.100000
50%        88.200000
75%        93.800000
max        98.600000
Name: RESULTADOS DE CALIDAD / QUALITY RESULTS, dtype: float64
```

Por otro lado, vamos a proceder a crear una columna que nos permita categorizar cada resultado con la marca comercial que representa la estación.

[ ]:
```
def get_gas_station_name(row: pd.Series) -> str:

    stations = [
        'ATLANTIC', 'TEXACO', 'SHELL', 'ECOPETROLEO', 'ESSO', 'NEXT',
       'SUNIX', 'PETROMOVIL', 'PETRONAN', 'TOTAL', 'AXXON', 'TDC', 'UNITED',
       'SOL', 'NATIVA', 'SIGMA', 'SITRAS', 'EXCOM'
    ]
   # print(row['CLIENTE'].contains('A'))
    for station in stations:
        if station in row['CLIENTE']:
            break

    return station
```

[ ]:
```
df_ron_mon['NOMBRE ESTACION'] = None
df_ron_mon['NOMBRE ESTACION'] = df_ron_mon.apply(lambda row:␣
↪get_gas_station_name(row), axis=1)
```

[ ]:
```
print(df_ron_mon['NOMBRE ESTACION'].unique())
```

```
['ATLANTIC' 'TEXACO' 'SHELL' 'ECOPETROLEO' 'ESSO' 'NEXT' 'SUNIX' 'NATIVA'
 'EXCOM' 'PETROMOVIL' 'PETRONAN' 'TOTAL' 'SOL' 'SITRAS' 'SIGMA' 'AXXON'
 'TDC' 'UNITED']
```

3.2. Análisis Gasolina Premium Comencemos el análisis con la gasolina tipo premium. Primero, vamos mostrar las estadísticas descriptivas tanto para el RON como el MON

[ ]:
```
premium_df = df_ron_mon[df_ron_mon['PRODUCTO'].str.contains('PREMIUM')]
```

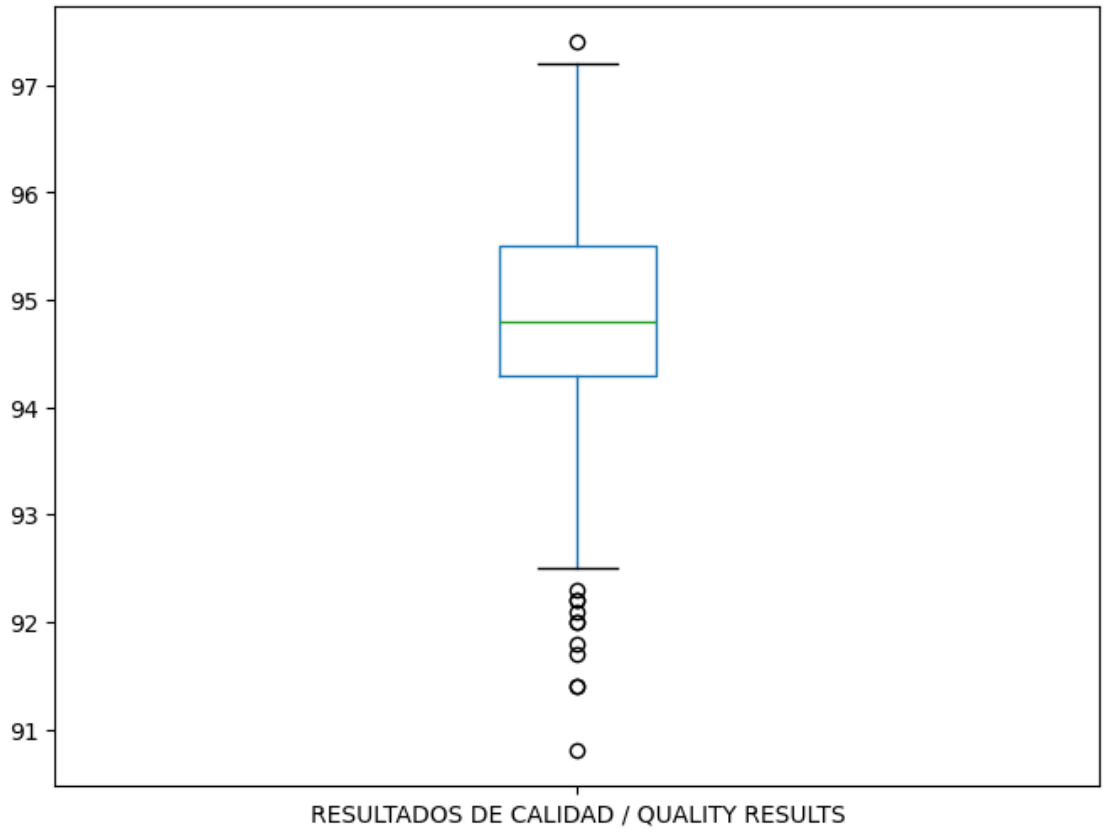
El resumen estadístico del RON muestra que el rango de valores va de 90 a 97.4 para un rango de 7.4. La desviación estándar es de 0.97 y tanto la media como la mediana tienen valores muy similares, lo que significa que existe poca variación en los datos.

Otro punto que puede estacarse es que el 50% de la estaciones tiene un valor de 94.8 hacia abajo, mientras que apenas un 25% tiene un valor por encima de 95.5.

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('RON')]['RESULTADOS␣
↪DE CALIDAD / QUALITY RESULTS'].describe()
```

[ ]:
```
count    1003.000000
mean       94.813460
std         0.970096
min        90.800000
25%        94.300000
50%        94.800000
75%        95.500000
max        97.400000
Name: RESULTADOS DE CALIDAD / QUALITY RESULTS, dtype: float64
```

[ ]:
```
boxplot = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('RON')].
↪boxplot(column = 'RESULTADOS DE CALIDAD / QUALITY RESULTS', figsize=(8,6),␣
↪grid=False)
plt.show()
```



A continuación se muestran los valores atípicos identificados en el diagrama de caja anterior

[ ]:
```
Q1 = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('RON')]['RESULTADOS DE CALIDAD / QUALITY RESULTS'].quantile(0.25)
Q3 = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('RON')]['RESULTADOS DE CALIDAD / QUALITY RESULTS'].quantile(0.75)
IQR = Q3 - Q1
threshold = 1.5

outliers = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('RON') &␣
 ↪((premium_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] < Q1 - threshold *␣
 ↪IQR) |(premium_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] > Q3 +␣
 ↪threshold * IQR) )]
```

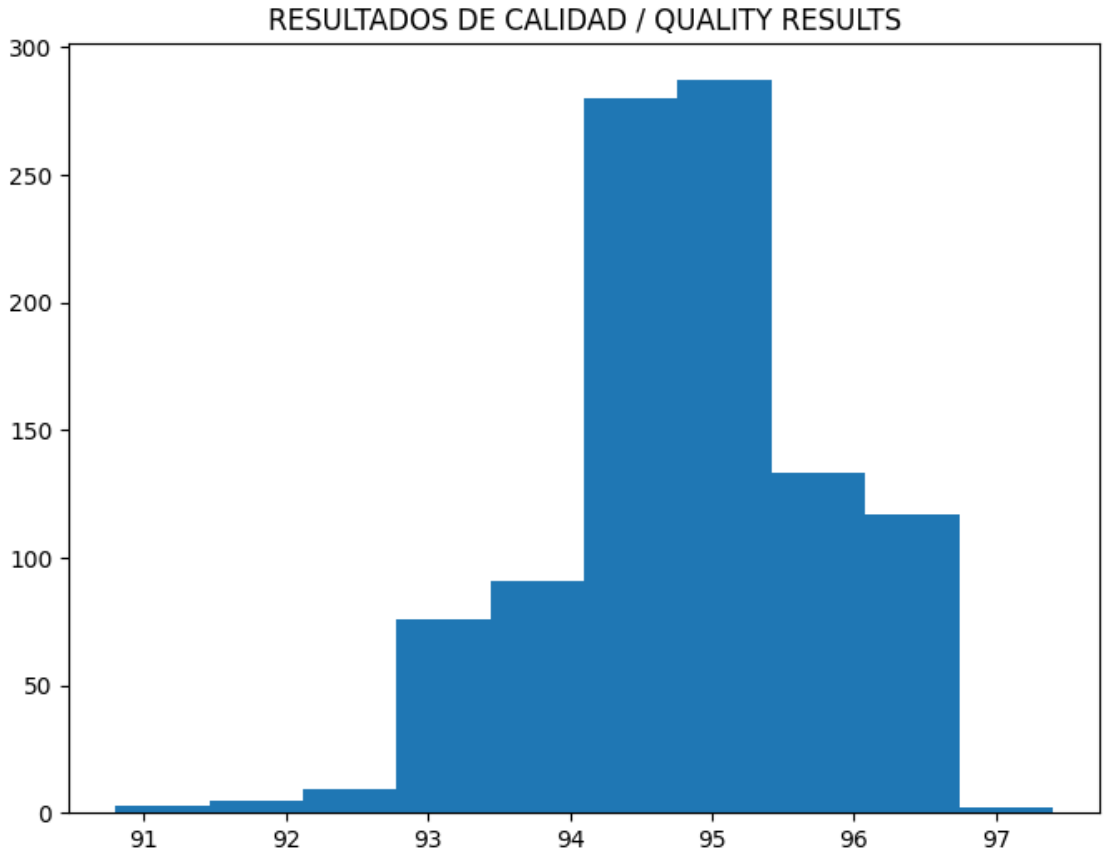
[ ]:
```
print(outliers[outliers['ANÁLISIS / ANALYSES'].str.contains('RON')].
 ↪groupby('NOMBRE ESTACION')['RESULTADOS DE CALIDAD / QUALITY RESULTS'].
 ↪apply(list))
```

```
NOMBRE ESTACION
NEXT                                  [90.8]
PETRONAN                        [92.1, 91.8]
```

```
SHELL       [92.2, 92.3, 92.0, 91.4, 91.4, 97.4]
TEXACO                                [92.2, 92.0]
TOTAL                                      [91.7]
Name: RESULTADOS DE CALIDAD / QUALITY RESULTS, dtype: object
```

Revisando el histograma de los datos, se observa que está ligeramente sesgado a la izquierda. Este comportamiento es de esperarse por los datos atípicos en el lado izquierdo que se mostraron en la imagen anterior.

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('RON')].hist(column =␣
↪'RESULTADOS DE CALIDAD / QUALITY RESULTS', figsize=(8,6), grid=False)
plt.show()
```



Al agrupar los datos por estación, podemos observar que las estaciones TOTAL, PETRONAN, NEXT, SOL y UNITED son las que tienen mayor variación en comparación con las demás. Los rangos de valores de estas estaciones van desde 91.7-96.6, 91.8 - 96.5, 90.8 - 96.2, 92.7 - 96.2 y 94.8 - 96.7 respectivamente, siendo NEXT la estación que obtuvo el valor mínimo más bajo (90.8).

Por otro lado, el 75% de las estaciones de AXXON, ESSO, SUNIX y TOTAL tienen un RON por debajo de 95. Por último, solo las estaciones PETROMOVIL, SHELL, TDC y ECOPETROLEO son las que tienen una mediana igual o mayor a 95, lo que indica que el 50% de sus estaciones

tienen un RON de 95 o más. En el caso particular de TDC, esta fue la estación con el número mínimo más alto (95.9)

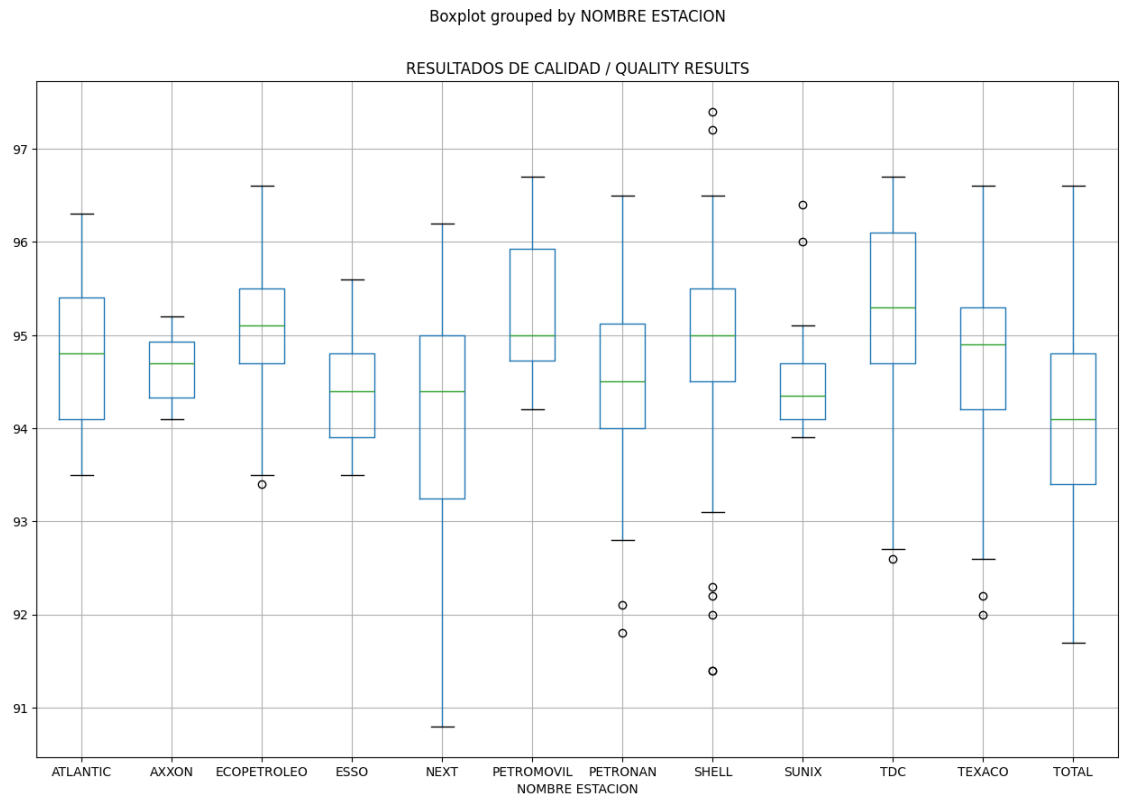
Por último, importante mencionar que el número total de estaciones AXXON, ESSO, SITRAS, TDC y UNITED es menor a 10 en cada una de ellas.

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('RON')].
  ↪groupby('NOMBRE ESTACION')['RESULTADOS DE CALIDAD / QUALITY RESULTS'].
  ↪describe()
```

[ ]:

NOMBRE ESTACION	count	mean	std	min	25%	50%	75%	max
ATLANTIC	13.0	94.823077	0.909353	93.5	94.100	94.80	95.400	96.3
AXXON	6.0	94.650000	0.432435	94.1	94.325	94.70	94.925	95.2
ECOPETROLEO	144.0	95.176389	0.697647	93.4	94.700	95.10	95.500	96.6
ESSO	5.0	94.440000	0.814248	93.5	93.900	94.40	94.800	95.6
EXCOM	42.0	94.964286	0.831930	92.7	94.500	94.90	95.750	96.3
NATIVA	40.0	95.950000	0.432642	94.7	95.900	96.10	96.300	96.6
NEXT	27.0	94.166667	1.192928	90.8	93.250	94.40	95.000	96.2
PETROMOVIL	34.0	95.220588	0.739279	94.2	94.725	95.00	95.925	96.7
PETRONAN	76.0	94.493421	1.043626	91.8	94.000	94.50	95.125	96.5
SHELL	217.0	94.931336	0.870470	91.4	94.500	95.00	95.500	97.4
SIGMA	32.0	95.040625	0.823783	92.6	94.600	94.80	95.725	96.7
SITRAS	3.0	95.700000	0.781025	94.8	95.450	96.10	96.150	96.2
SOL	16.0	94.537500	1.166119	92.7	93.350	94.85	95.425	96.2
SUNIX	24.0	94.541667	0.600664	93.9	94.100	94.35	94.700	96.4
TDC	3.0	96.233333	0.416333	95.9	96.000	96.10	96.400	96.7
TEXACO	148.0	94.811486	0.884063	92.0	94.200	94.90	95.300	96.6
TOTAL	171.0	94.206433	1.022692	91.7	93.400	94.10	94.800	96.6
UNITED	2.0	95.750000	1.343503	94.8	95.275	95.75	96.225	96.7

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('RON')].boxplot(
    by='NOMBRE ESTACION',
    column = 'RESULTADOS DE CALIDAD / QUALITY RESULTS',
    figsize=(15,10))
plt.show()
```



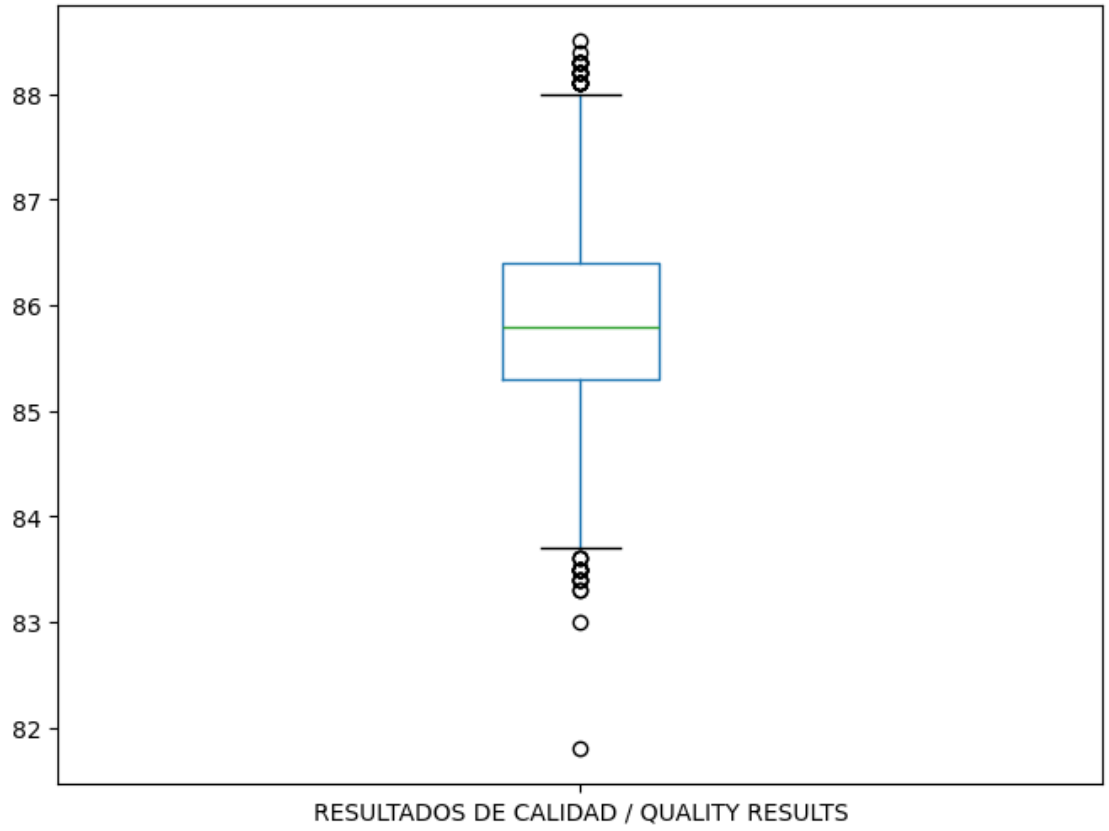
Ahora pasamos al análisis de los resultados para el método MON. Como se muestra a continuación, tanto la desviación estándar como el rango es muy similar a los resultados obtenidos por el método RON, lo que sugiere poca variación en los resultados.

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('MON')]['RESULTADOS␣
↪DE CALIDAD / QUALITY RESULTS'].describe()
```

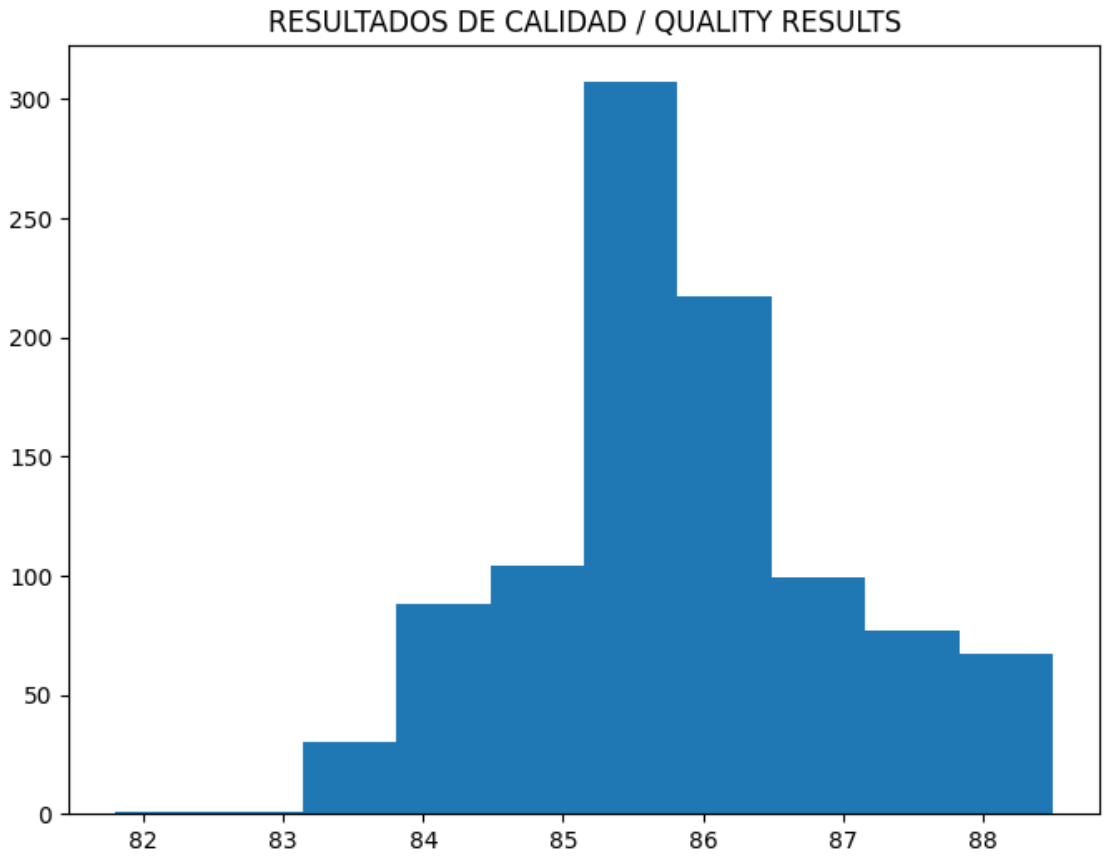
[ ]:
```
count    991.000000
mean      85.854995
std        1.113241
min       81.800000
25%       85.300000
50%       85.800000
75%       86.400000
max       88.500000
Name: RESULTADOS DE CALIDAD / QUALITY RESULTS, dtype: float64
```

Revisando el diagrama de caja, se muestra una forma ligeramente más simétrica en comparación con los resultados del método RON, aunque si se destaca mayor número de valores atípicos. Lo anterior también puede confirmarse observando el histograma que se muestra más debajo.

[ ]:
```
boxplot = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('MON')].
  ↪boxplot(column = 'RESULTADOS DE CALIDAD / QUALITY RESULTS', figsize=(8,6),␣
  ↪grid=False)
plt.show()
```



[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('MON')].hist(column =␣
  ↪'RESULTADOS DE CALIDAD / QUALITY RESULTS', figsize=(8,6), grid=False)
plt.show()
```



Como se mencionó arriba, en el caso del método MON, se puede apreciar un mayor número de valores atípicos. Además de las estaciones NEXT, PETRONAN, SHELL, TEXACO y TOTAL, también tienen valores atípicos las estaciones ECOPETROLEO, NATIVA, PETROMOVIL y SIGMA.

[ ]:
```
Q1 = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('MON')]['RESULTADOS DE CALIDAD / QUALITY RESULTS'].quantile(0.25)
Q3 = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('MON')]['RESULTADOS DE CALIDAD / QUALITY RESULTS'].quantile(0.75)
IQR = Q3 - Q1
threshold = 1.5

outliers = premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('MON') &␣
 ↪((premium_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] < Q1 - threshold *␣
 ↪IQR) |(premium_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] > Q3 +␣
 ↪threshold * IQR) )]
```

[ ]:
```
print(outliers[outliers['ANÁLISIS / ANALYSES'].str.contains('MON')].
 ↪groupby('NOMBRE ESTACION')['RESULTADOS DE CALIDAD / QUALITY RESULTS'].
 ↪apply(list))
```

```
NOMBRE ESTACION
```

```
ECOPETROLEO    [88.1, 88.1, 88.3, 88.1, 88.1, 88.2, 88.1, 88…
NATIVA                                        [88.3, 88.1, 88.2]
NEXT                        [83.5, 83.4, 83.4, 83.4, 83.5, 83.0]
PETROMOVIL                        [88.5, 88.3, 88.3, 88.3, 88.2]
PETRONAN                                [88.1, 83.5, 83.5, 83.6]
SHELL                                                [83.5, 81.8]
SIGMA                                                      [88.1]
TEXACO                                               [88.2, 83.6]
TOTAL          [83.4, 83.3, 83.6, 88.2, 83.5, 83.3, 88.4, 88…
Name: RESULTADOS DE CALIDAD / QUALITY RESULTS, dtype: object
```

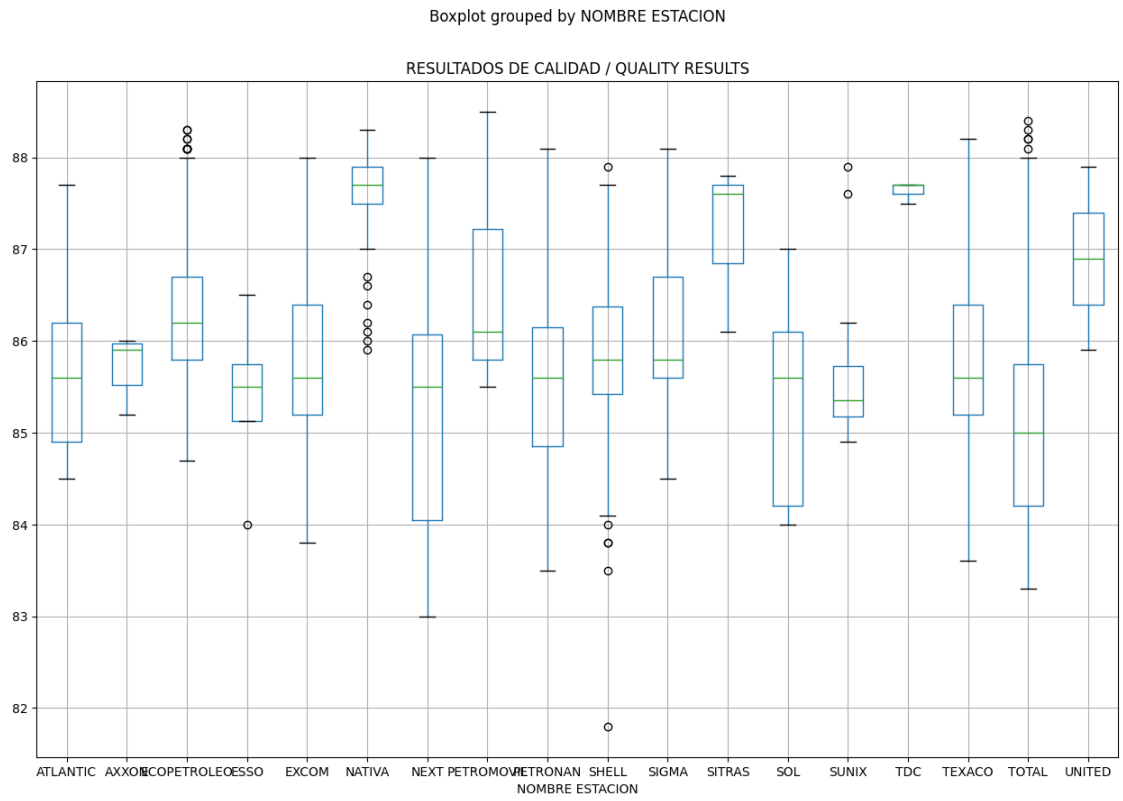
Realizando el análisis por tipo de estación se puede apreciar que todas las estaciones, a excepción de la SHELL, tienen sus valores por encima de 82, el cual es el valor de referencia mínimo. En cuanto a dispersión de lso datos, las estaciones ATLANTIC, ESSO, NEXT, PETRONAN, TOTAL y UNITED son las que tienen las desviaciones estándares más altas, con un valor igual o mayor a 1.

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('MON')].
  ↪groupby('NOMBRE ESTACION')['RESULTADOS DE CALIDAD / QUALITY RESULTS'].
  ↪describe()
```

[ ]:

NOMBRE ESTACION	count	mean	std	min	25%	50%	75%	max
ATLANTIC	13.0	85.753846	1.023568	84.5	84.900	85.60	86.200	87.7
AXXON	6.0	85.733333	0.344480	85.2	85.525	85.90	85.975	86.0
ECOPETROLEO	143.0	86.401399	0.891587	84.7	85.800	86.20	86.700	88.3
ESSO	4.0	85.375000	1.030776	84.0	85.125	85.50	85.750	86.5
EXCOM	44.0	85.797727	0.975631	83.8	85.200	85.60	86.400	88.0
NATIVA	39.0	87.474359	0.627744	85.9	87.500	87.70	87.900	88.3
NEXT	26.0	85.169231	1.351079	83.0	84.050	85.50	86.075	88.0
PETROMOVIL	34.0	86.505882	0.990238	85.5	85.800	86.10	87.225	88.5
PETRONAN	75.0	85.628000	1.152853	83.5	84.850	85.60	86.150	88.1
SHELL	210.0	85.832857	0.778796	81.8	85.425	85.80	86.375	87.9
SIGMA	32.0	86.240625	0.985003	84.5	85.600	85.80	86.700	88.1
SITRAS	3.0	87.166667	0.929157	86.1	86.850	87.60	87.700	87.8
SOL	17.0	85.335294	0.982307	84.0	84.200	85.60	86.100	87.0
SUNIX	24.0	85.604167	0.742194	84.9	85.175	85.35	85.725	87.9
TDC	3.0	87.633333	0.115470	87.5	87.600	87.70	87.700	87.7
TEXACO	145.0	85.777241	0.994551	83.6	85.200	85.60	86.400	88.2
TOTAL	171.0	85.182456	1.216377	83.3	84.200	85.00	85.750	88.4
UNITED	2.0	86.900000	1.414214	85.9	86.400	86.90	87.400	87.9

[ ]:
```
premium_df[premium_df['ANÁLISIS / ANALYSES'].str.contains('MON')].boxplot(
    by='NOMBRE ESTACION',
    column = 'RESULTADOS DE CALIDAD / QUALITY RESULTS',
    figsize=(15,10))
plt.show()
```



3.3. Análisis Gasolina Regular Ahora procederemos a realizar el análisis descriptivo para la gasolina de tipo regular. Para esto, creamos primero un dataframe que contenga solo los valores de la gasolina de este tipo.

```
[ ]: regular_df = df_ron_mon[df_ron_mon['PRODUCTO'].str.contains('REGULAR')]
```

Revisando los resultados por el método RON, se aprecia que el 75% de los datos se encuentran por encima del valor 90.4, superior al mínimo de referencia el cual es 89.

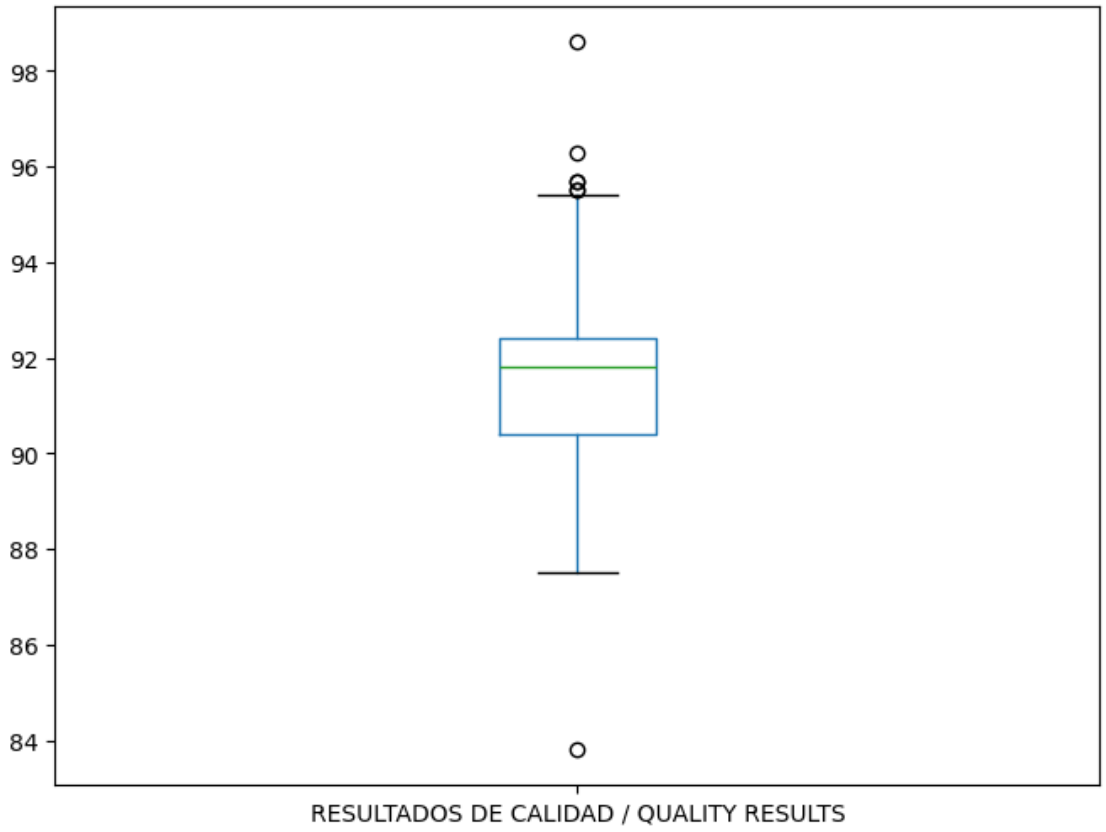
```
[ ]: regular_df[regular_df['ANÁLISIS / ANALYSES'].str.contains('RON')]['RESULTADOS␣
     ↪DE CALIDAD / QUALITY RESULTS'].describe()
```

```
[ ]: count    892.000000
     mean      91.555381
     std        1.721562
     min       83.800000
     25%       90.400000
     50%       91.800000
     75%       92.400000
     max       98.600000
     Name: RESULTADOS DE CALIDAD / QUALITY RESULTS, dtype: float64
```

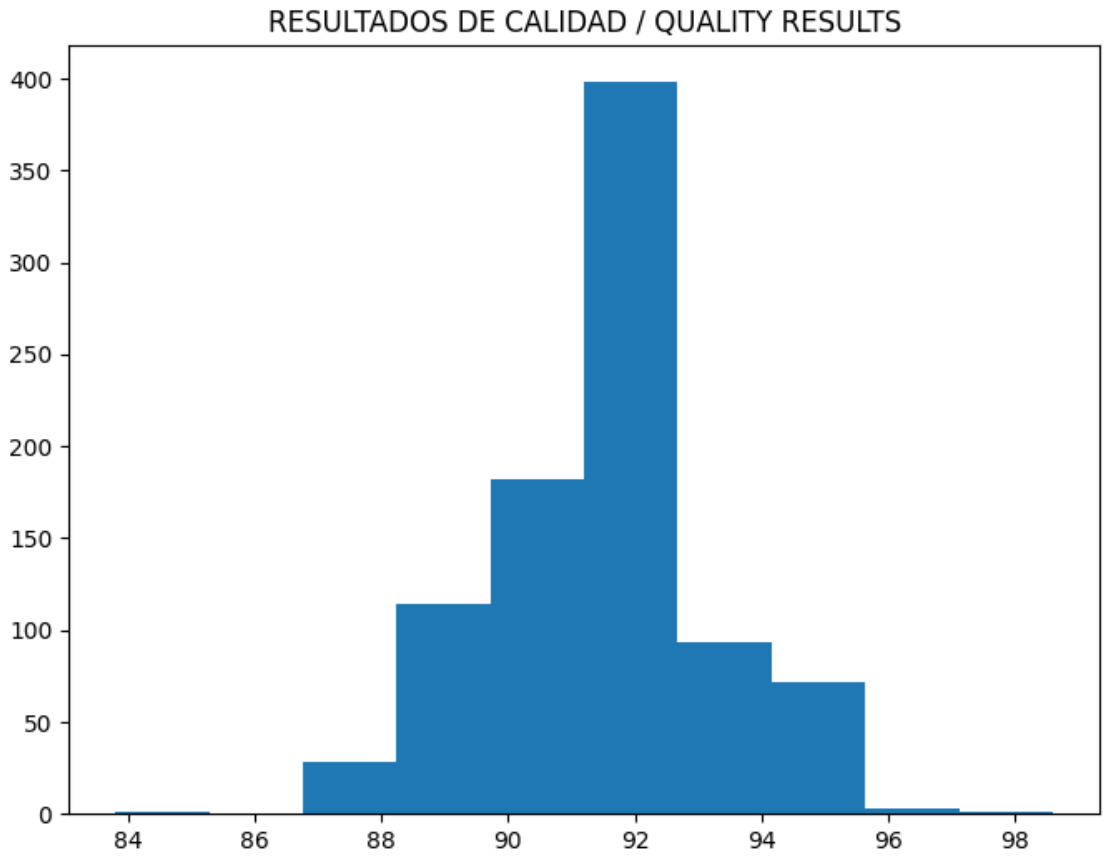
En el diagrama de caja se puede visualizar menor presencia de valores atípicos, mientras que en el

histograma se muestra una distribución centrada. Sin embargo, llama la atención que la mayoría de los valores atípicos se encuentran en el extremo derecho, es decir, con valores que rondan en 95 o más.

[ ]:
```
boxplot = regular_df[regular_df['ANÁLISIS / ANALYSES'].str.contains('RON')].
  ↪boxplot(column = 'RESULTADOS DE CALIDAD / QUALITY RESULTS', figsize=(8,6),␣
  ↪grid=False)
plt.show()
```



[ ]:
```
regular_df[regular_df['ANÁLISIS / ANALYSES'].str.contains('RON')].hist(column =␣
  ↪'RESULTADOS DE CALIDAD / QUALITY RESULTS', figsize=(8,6), grid=False)
plt.show()
```



Revisando un poco más de cerca los atípicos, se evidencian valores muy por encima del valor mínimo de referencia, el cual es 89. Por ejemplo, TOTAL las Hortensias y SHELL Alvarez presentan valores de 98.6 y 96.3 respectivamente, incluso superior al valor mínimo de referencia de la gasolina Premium.

Estos valores fueron confirmados en el reporte original para descartar cualquier error durante el preprocesamiento de los datos.

[ ]:
```
Q1 = regular_df[regular_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('RON')]['RESULTADOS DE CALIDAD / QUALITY RESULTS'].quantile(0.25)
Q3 = regular_df[regular_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('RON')]['RESULTADOS DE CALIDAD / QUALITY RESULTS'].quantile(0.75)
IQR = Q3 - Q1
threshold = 1.5

outliers = regular_df[regular_df['ANÁLISIS / ANALYSES'].str.contains('RON') &␣
 ↪((regular_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] < Q1 - threshold *␣
 ↪IQR) |(regular_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] > Q3 +␣
 ↪threshold * IQR) )]
```

[ ]:
```
outliers [['CLIENTE', 'RESULTADOS DE CALIDAD / QUALITY RESULTS', 'ARCHIVO']]
```

[ ]:

	CLIENTE	RESULTADOS DE CALIDAD / QUALITY RESULTS	ARCHIVO
5324	TOTAL HACIENDA	83.8	2023 05 114.pdf
7596	PETROMOVIL JHL HAINA	95.5	2023 02 046.pdf
9026	TOTAL LAS HORTENSIAS	98.6	2023 01 025.pdf
9355	TOTAL LA 27 DE FEBRERO	95.7	2023 03 068.pdf
10051	SHELL ALVAREZ	96.3	2023 07 129.pdf
22713	TOTAL LA ESTRELLA	95.7	2023 02 076.pdf
29123	ECOPETROLEO CAMILO	95.5	2023 05 137.pdf

Realizando el análisis para cada estación, se evidencia que para las estaciones ATLANTIC y SOL cerca del 25% de los resultados caen por debajo de 89. Para las demás, el 75% de los datos son iguales o mayores al valor mínimo de referencia, el cual es de 89.

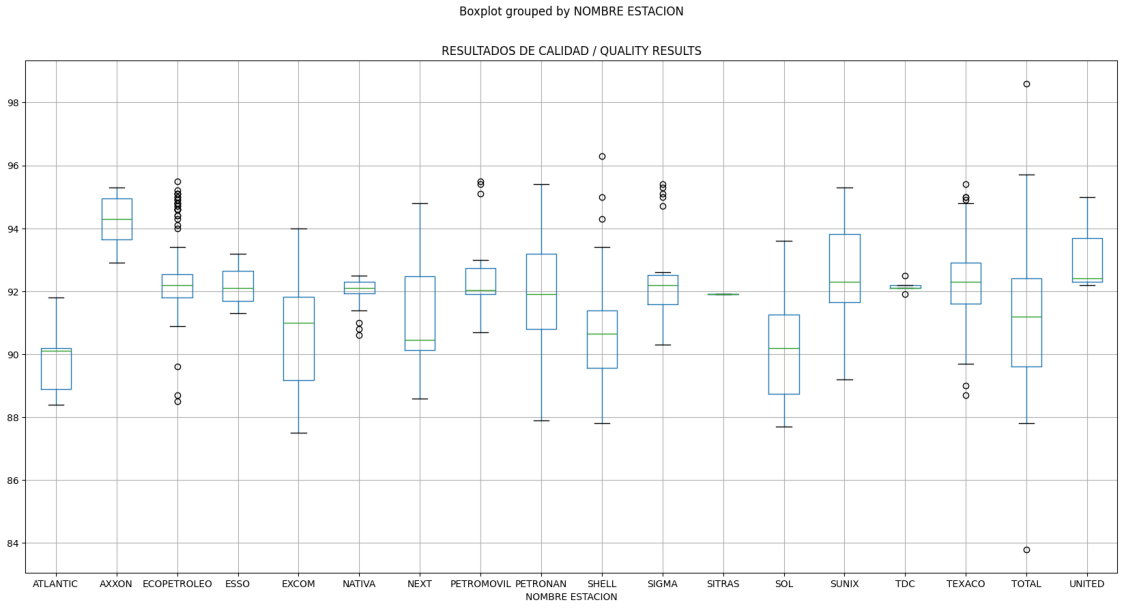
[ ]:
```
regular_df[regular_df['ANÁLISIS / ANALYSES'].str.contains('RON')].
↪groupby('NOMBRE ESTACION')['RESULTADOS DE CALIDAD / QUALITY RESULTS'].
↪describe()
```

[ ]:

NOMBRE ESTACION	count	mean	std	min	25%	50%	75%	max
ATLANTIC	9.0	89.833333	1.081665	88.4	88.900	90.10	90.200	91.8
AXXON	6.0	94.233333	0.939503	92.9	93.650	94.30	94.950	95.3
ECOPETROLEO	119.0	92.473950	1.294150	88.5	91.800	92.20	92.550	95.5
ESSO	3.0	92.200000	0.953939	91.3	91.700	92.10	92.650	93.2
EXCOM	28.0	90.660714	1.780416	87.5	89.175	91.00	91.825	94.0
NATIVA	34.0	92.023529	0.451300	90.6	91.925	92.10	92.300	92.5
NEXT	34.0	91.158824	1.654950	88.6	90.125	90.45	92.475	94.8
PETROMOVIL	32.0	92.381250	1.126925	90.7	91.900	92.05	92.725	95.5
PETRONAN	69.0	91.889855	1.902254	87.9	90.800	91.90	93.200	95.4
SHELL	196.0	90.591327	1.294000	87.8	89.575	90.65	91.400	96.3
SIGMA	28.0	92.371429	1.441413	90.3	91.575	92.20	92.525	95.4
SITRAS	1.0	91.900000	NaN	91.9	91.900	91.90	91.900	91.9
SOL	11.0	90.236364	1.992623	87.7	88.750	90.20	91.250	93.6
SUNIX	16.0	92.581250	1.573200	89.2	91.650	92.30	93.825	95.3
TDC	5.0	92.160000	0.219089	91.9	92.100	92.10	92.200	92.5
TEXACO	137.0	92.354745	1.221376	88.7	91.600	92.30	92.900	95.4
TOTAL	161.0	90.981988	2.044104	83.8	89.600	91.20	92.400	98.6
UNITED	3.0	93.200000	1.562050	92.2	92.300	92.40	93.700	95.0

Revisando más de cerca los datos para cada estación por medio del diagrama de caja, se puede apreciar un mayor número de valores atípicos en el extremo derecho, siendo estos por lo general igual o superior a 94.

[ ]:
```
regular_df[regular_df['ANÁLISIS / ANALYSES'].str.contains('RON')].boxplot(
    by='NOMBRE ESTACION',
    column = 'RESULTADOS DE CALIDAD / QUALITY RESULTS',
    figsize=(20,10))
plt.show()
```



Filtrando todos los valores mayores o iguales a 95 del método RON para la gasolina regular, se obtienen un total de 29 registros. Como se mencionó más arriba, llama la atención que el valor obtenido en esas estaciones para la gasolina regular sea mayor o igual al valor mínimo de referencia para la gasolina premium.

[ ]:
```
ron_regular_df = regular_df[regular_df['ANÁLISIS / ANALYSES'].str.
 ↪contains('RON')]
ron_regular_df[ron_regular_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] >=␣
 ↪95][['CLIENTE', 'NOMBRE ESTACION', 'ANÁLISIS / ANALYSES','RESULTADOS DE␣
 ↪CALIDAD / QUALITY RESULTS', 'ARCHIVO']]
```

[ ]:

	CLIENTE	NOMBRE ESTACION	\
809	TEXACO SAN ANDRES	TEXACO	
4050	SIGMA AV. ECOLOGICA	SIGMA	
4442	TEXACO MOCA	TEXACO	
5162	SHELL JUANA RUIZ	SHELL	
6065	ECOPETROLEO SATELITE	ECOPETROLEO	
6087	ECOPETROLEO SATELITE	ECOPETROLEO	

6364	PETROMOVIL ROMAR	PETROMOVIL
6387	PETROMOVIL ROMAR	PETROMOVIL
6876	SUNIX GENESIS	SUNIX
7596	PETROMOVIL JHL HAINA	PETROMOVIL
9026	TOTAL LAS HORTENSIAS	TOTAL
9355	TOTAL LA 27 DE FEBRERO	TOTAL
10051	SHELL ALVAREZ	SHELL
10097	PETRONAN COMBUSELL\nDÍA ANÁLISIS	PETRONAN
15548	PETRONAN BELLAMAR	PETRONAN
16677	ECOPETROLEO PERALVILLO	ECOPETROLEO
17811	SIGMA KM 25	SIGMA
19291	UNITED PETROLEUM	UNITED
20285	ECOPETROLEO SATELITE	ECOPETROLEO
20563	SIGMA LAS AMERICAS 2	SIGMA
20585	SIGMA LAS AMERICAS 2	SIGMA
20703	ECOPETROLEO VILLA MELLA\nDÍA ANÁLISIS	ECOPETROLEO
22713	TOTAL LA ESTRELLA	TOTAL
23593	TEXACO INDEPENDENCIA\nDÍA ANÁLISIS	TEXACO
27582	AXXON LUCAMI	AXXON
27605	AXXON LUCAMI	AXXON
29123	ECOPETROLEO CAMILO	ECOPETROLEO
30071	ECOPETROLEO HACIENDA ESTRELLA DÍA ANÁLISIS	ECOPETROLEO
30259	PETRONAN AV. ESPAÑA	PETRONAN

	ANÁLISIS / ANALYSES \
809	NUMERO DE OCTANO METODO RESEARCH (RON)
4050	NUMERO DE OCTANO METODO RESEARCH (RON)
4442	NUMERO DE OCTANO METODO RESEARCH (RON)
5162	NUMERO DE OCTANO METODO RESEARCH (RON)
6065	NUMERO DE OCTANO METODO RESEARCH (RON)
6087	NUMERO DE OCTANO METODO RESEARCH (RON)
6364	NUMERO DE OCTANO METODO RESEARCH (RON)
6387	NUMERO DE OCTANO METODO RESEARCH (RON)
6876	NUMERO DE OCTANO METODO RESEARCH (RON)
7596	NUMERO DE OCTANO METODO RESEARCH (RON)
9026	NUMERO DE OCTANO METODO RESEARCH (RON)
9355	NUMERO DE OCTANO METODO RESEARCH (RON)
10051	NUMERO DE OCTANO METODO RESEARCH (RON)
10097	NUMERO DE OCTANO METODO RESEARCH (RON)
15548	NUMERO DE OCTANO METODO RESEARCH (RON)
16677	NUMERO DE OCTANO METODO RESEARCH (RON)
17811	NUMERO DE OCTANO METODO RESEARCH (RON)
19291	NUMERO DE OCTANO METODO RESEARCH (RON)
20285	NUMERO DE OCTANO METODO RESEARCH (RON)
20563	NUMERO DE OCTANO METODO RESEARCH (RON)
20585	NUMERO DE OCTANO METODO RESEARCH (RON)
20703	NUMERO DE OCTANO METODO RESEARCH (RON)

```
22713  NUMERO DE OCTANO METODO RESEARCH (RON)
23593  NUMERO DE OCTANO METODO RESEARCH (RON)
27582  NUMERO DE OCTANO METODO RESEARCH (RON)
27605  NUMERO DE OCTANO METODO RESEARCH (RON)
29123  NUMERO DE OCTANO METODO RESEARCH (RON)
30071  NUMERO DE OCTANO METODO RESEARCH (RON)
30259  NUMERO DE OCTANO METODO RESEARCH (RON)
```

	RESULTADOS DE CALIDAD / QUALITY RESULTS	ARCHIVO
809	95.0	2023 02 025.pdf
4050	95.3	2023 02 037.pdf
4442	95.4	2023 03 033.pdf
5162	95.0	2023 01 040.pdf
6065	95.2	2023 02 009.pdf
6087	95.1	2023 02 009.pdf
6364	95.4	2023 03 030.pdf
6387	95.1	2023 03 030.pdf
6876	95.3	2023 01 082.pdf
7596	95.5	2023 02 046.pdf
9026	98.6	2023 01 025.pdf
9355	95.7	2023 03 068.pdf
10051	96.3	2023 07 129.pdf
10097	95.4	2023 02 078.pdf
15548	95.1	2023 02 067.pdf
16677	95.0	2023 02 112.pdf
17811	95.4	2023 02 070.pdf
19291	95.0	2023 02 101.pdf
20285	95.1	2023 02 061.pdf
20563	95.0	2023 02 060.pdf
20585	95.1	2023 02 060.pdf
20703	95.0	2023 02 074.pdf
22713	95.7	2023 02 076.pdf
23593	95.0	2023 01 003.pdf
27582	95.3	2023 03 017.pdf
27605	95.0	2023 03 017.pdf
29123	95.5	2023 05 137.pdf
30071	95.1	2023 03 004.pdf
30259	95.4	2023 02 015.pdf

Realizando un conteo por estación, la estación ECOPETROLEO es la que cuenta con un mayor número de resultados igual o mayor a 95.

[ ]:
```
ron_regular_df[ron_regular_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] >=␣
↪95][['NOMBRE ESTACION']].value_counts()
```

[ ]:
```
NOMBRE ESTACION
ECOPETROLEO        7
SIGMA              4
```

```
PETROMOVIL          3
PETRONAN            3
TEXACO              3
TOTAL               3
AXXON               2
SHELL               2
SUNIX               1
UNITED              1
Name: count, dtype: int64
```

El número de estaciones aumenta considerablemente si se filtran los valores para un RON mayor o igual a 94.

[ ]:
```
ron_regular_df[ron_regular_df['RESULTADOS DE CALIDAD / QUALITY RESULTS'] >=␣
↪94][['NOMBRE ESTACION']].value_counts()
```

[ ]:
```
NOMBRE ESTACION
ECOPETROLEO         23
TEXACO              15
PETRONAN            14
TOTAL               13
SIGMA                5
SUNIX                4
AXXON                3
NEXT                 3
PETROMOVIL           3
SHELL                3
EXCOM                1
UNITED               1
Name: count, dtype: int64
```

0.1.4 4. Conclusiones

Este análisis se enfocó solamente para los valores RON de la gasolina premium y regular, así como los valores MON para la gasolina premium. Los datos utilizados fueron extraidos directamente de los informes a través del servicio Document Intelligence de Azure. Un total de 100 archivos no pudieron ser procesados, por lo que queda pendiente de su revisión para evaluar la posibilidad de resolver los errores e incorporar los datos de esos reportes al análisis.

En cuanto a los resultados del análisis, para la gasolina premium se aprecia poca variación en los datos lo que implica que su calidad es consistente. Sin embargo, para una gran cantidad de estaciones, la media se encuentra fuera del rango de los valores mínimos establecidos. Por ejemplo, para el método RON, cerca del 50% de los datos se encuentran por debajo de 95.

Respecto a la gasolina regular, también se evidencia consistencia en sus valores, aunque en menor medida. Algunos hallazgos llaman la atención, como por ejemplo el hecho de que varios resultados por el método RON se encuentran igual o por encima de 95, valor mínimo establecido para la gasolina premium.