Campus de Rimouski

Département de mathématiques, informatique et génie

Récupération des données en lien avec la consommation énergétique au Canada à partir de différentes sources de données

Par

Sokhna Tahiratou Mbaye

Travail présenté à Yacine Yaddaden

Dans le cadre du cours : Projet en informatique I (INF34515)

Mercredi 25 Octobre 2023

Table des matières

Introduction

Ce projet se concentre sur l'extraction et l'analyse des données relatives à la consommation énergétique au Canada, avec une attention particulière portée sur Énergie NB, une source cruciale d'informations pour le Nouveau-Brunswick. Guidés par l'objectif d'obtenir une compréhension approfondie des tendances de consommation énergétique dans différentes régions, nous avons choisi de mettre en œuvre des technologies avancées telles que Python, Selenium, BeautifulSoup, SQLAlchemy.

Cette exploration complète s'articule autour d'un pipeline de données, débutant par la collecte initiale et aboutissant à une représentation visuelle finale. Notre démarche cherche à répondre de manière exhaustive aux questions cruciales entourant la consommation énergétique dans la région. Tout au long du projet, nous mettrons en évidence des approches innovantes et des méthodes spécifiques pour garantir une gestion efficace des ressources énergétiques.

Nous anticipons des résultats significatifs qui contribueront non seulement à l'avancement de notre compréhension de la consommation énergétique, mais également à la démonstration de l'efficacité des technologies avancées dans ce domaine spécifique.

I. Description des Outils et Technologies Utilisés

- Python : Python est le langage de programmation central de notre projet en raison de sa simplicité, de sa polyvalence et de sa riche bibliothèque. Sa syntaxe claire facilite la manipulation des données, en faisant de Python un choix idéal pour le traitement et l'analyse de données.
- Selenium : Selenium est utilisé pour automatiser les interactions avec les pages web. Il permet d'émuler des actions humaines telles que le clic sur des éléments, facilitant ainsi la récupération de données à partir de pages dynamiques, notamment celles générées par JavaScript.
- BeautifulSoup : BeautifulSoup est une bibliothèque Python dédiée à l'extraction d'informations à partir de fichiers HTML et XML. Grâce à ses fonctionnalités, il simplifie la tâche d'analyse de la structure des pages web, en particulier pour extraire des données structurées.
- SQLAlchemy : SQLAlchemy est un ORM (Object-Relational Mapping) qui simplifie la communication avec des bases de données relationnelles. Il offre une abstraction puissante au-dessus du langage SQL, facilitant ainsi la création, la gestion et la manipulation des données.
- Requests : Requests est une bibliothèque HTTP pour Python, conçue pour simplifier l'envoi de requêtes HTTP et la gestion des réponses. Elle est utilisée pour récupérer des données à partir de sources accessibles via des API web, offrant ainsi une méthode efficace pour l'interaction avec des services en ligne.
- Pandas : Pandas est une bibliothèque de manipulation et d'analyse de données en Python. Elle propose des structures de données puissantes, notamment les DataFrames, qui facilitent le nettoyage, la transformation et l'analyse des données, rendant le processus plus efficace et lisible.
- Matplotlib : Matplotlib est une bibliothèque de visualisation de données qui nous permet de créer des graphiques et des visualisations. Cela nous aide à mieux comprendre les tendances et les modèles présents dans les données, facilitant ainsi l'interprétation visuelle des résultats.

- Streamlit - Streamlit est utilisé pour créer des applications web interactives et des tableaux de bord. Il offre une interface conviviale pour présenter et partager dynamiquement les résultats du projet, rendant ainsi l'exploration des données plus accessible.

II. Récupération des données

Le processus de récupération des données commence par l'utilisation de Selenium pour automatiser l'interaction avec les pages web d'Énergie NB. Nous identifions et cliquons sur les éléments pertinents, déclenchant ainsi le chargement des données dynamiques.

Requests est employé pour récupérer des données depuis des sources accessibles via des API web. Nous envoyons des requêtes HTTP, recevons les réponses, puis extrayons et traitons les données nécessaires.

a. Données du réseau (mise à jour au 5 minutes)

Les données sont structurées sous forme de tableau, comprenant plusieurs colonnes telles que "Charge au NB", "Demande au NB", "ISO-NE", etc. Chaque colonne contient des informations numériques associées à chaque catégorie.

Le processus d'extraction des données commence par l'utilisation de la bibliothèque Requests, qui permet de récupérer le contenu HTML de la page web. Ensuite, la bibliothèque BeautifulSoup est employée pour analyser ce contenu. Elle identifie de manière précise les balises pertinentes dans le code HTML, ce qui facilite l'extraction des données spécifiques contenues dans ces balises.

Les valeurs numériques, représentant des informations telles que la charge au Nouveau-Brunswick, la demande au Nouveau-Brunswick, ISO-NE, etc., sont extraites des cellules du tableau. Ce processus de récupération des données garantit une précision et une fiabilité dans l'acquisition des informations nécessaires pour une analyse ultérieure.

b. Données du réseau archivé

La page est conçue pour consulter l'historique de l'information sur le réseau électrique d'Énergie NB Power, couvrant la période de janvier 2016 à novembre 2023. Les données horaires sont disponibles sous forme de fichiers CSV mensuels. Les utilisateurs ont la possibilité de sélectionner un mois et une année spécifiques, puis de cliquer sur le lien "Obtenir les données" pour télécharger le fichier CSV correspondant.

Les données énergétiques, exprimées en MWh, sont organisées mensuellement dans des fichiers au format «.aspx ». Un fichier avec l'extension « .aspx » est une page Web générée par le framework Microsoft ASP.NET, exécuté sur des serveurs Web (source : https://docs.fileformat.com/fr/web/aspx). Ces informations sont structurées en 8 colonnes.

Dans le cadre de ce projet, l'utilisation combinée de Selenium et BeautifulSoup a été privilégiée pour effectuer du web scraping. Selenium a permis une navigation automatisée sur le site à l'aide du webdriver de Chrome, en utilisant le module By de Selenium pour sélectionner les éléments et JavaScript pour définir la valeur de l'élément à sélectionner. Ensuite, BeautifulSoup a été utilisé pour extraire le contenu de la balise 'pre', qui contient les données nécessaires, et les stocker dans une variable

c. Interruptions

La page "Interruptions de transport" présente des informations sous forme de tableau, où chaque ligne représente une interruption de transport avec les détails associés tels que le numéro d'interruption, le statut, l'heure de début, l'heure d'arrêt et le but. Pour accéder aux détails spécifiques d'une interruption, l'utilisateur se doit de cliquer sur le numéro d'interruption correspondant, ce qui ouvre un dialogue contenant des informations détaillées.

Pour cette étape, nous avons utilisé le webdriver de Chrome pour accéder à la page, faisant appel à Selenium. Ensuite, nous avons utilisé Selenium pour extraire tous les éléments <a> ayant la classe 'popup', lesquels représentent les numéros d'interruption sur la page. Par la suite, nous avons itéré sur ces numéros d'interruption obtenus. À chaque itération, la fonction a exécuté un

script JavaScript, déclenchant l'ouverture du dialogue spécifique à l'interruption en cours et chargeant les données extraites dans des variables.

d. Prévisions des charges

La page « Prévisions des charges » contient des rapports et évaluations répartis dans 4 dossiers : jour/5-jour, horaire, Trimestre/18-mois, semaine/28-jours. Dans chaque dossier, se trouve un sous-dossier nommé ‘Archive’ et des fichiers CSV. À l'intérieur de chaque sous-dossier ‘Archive’, on retrouve des dossiers pour chaque année de 2004 à 2023. Dans chaque dossier d'année, il y a des sous-dossiers pour chaque mois, et à l'intérieur de ces dossiers de mois se trouvent des fichiers CSV.

Le processus d'extraction des données a débuté par la navigation sur la page, suivie du clic sur chaque dossier identifié par leur ID avec Selenium. Ensuite, nous avons sélectionné le sous-dossier ‘Archive’, et une fois la page chargée, nous avons cliqué sur les sous-dossiers annuels, puis mensuels, avant de télécharger les fichiers CSV. Après chaque téléchargement, nous avons utilisé driver.back() pour revenir en arrière et accéder aux autres dossiers.

Les sous-dossiers ont été identifiés grâce à leur ID qui suit le format 'lv_' suivi du numéro du mois. Quant aux fichiers CSV, nous les avons téléchargés en extrayant les liens de chaque fichier dans les sous-dossiers, utilisant la méthode de Selenium pour localiser toutes les balises « a » dont l'attribut href se termine par « .csv ».

III. Traitement et transformation

Une fois la récupération dans les quatre fonctions d’Energie NB terminée, les données ont été traitées et transformées comme suit :

- Donnée du réseau (5min) : La bibliothèque Pandas est utilisée pour organiser les données dans des structures tabulaires, facilitant ainsi leur manipulation. Elle crée des DataFrames à partir des listes extraites et effectue des opérations de nettoyage et de formatage.
- Pour la fonction ‘données archivées’, le contenu de la balise <pre>, extrait avec la méthode get_text(), est transformé en une structure de données exploitable. Le texte est divisé en lignes en utilisant le caractère de saut de ligne ('\n'). La première ligne (lines[0]) est traitée comme l'en-tête, ses éléments sont séparés par des virgules (','). Les lignes suivantes sont

également divisées en éléments par des virgules, créant ainsi un dictionnaire pour chaque ligne avec l'en-tête comme clé et les données comme valeur. Ces dictionnaires sont ensuite ajoutés à la liste all_data.

- Pour la page des 'interruptions de transport', le traitement et la transformation des données ont lieu après la récupération des données de la base de données. L'objectif principal est de convertir les champs de date (planned_start et planned_stop) en objets datetime distincts. Deux listes distinctes, dates_start et dates_stop, sont créées pour stocker les dates planifiées de début et de fin d'interruption.
- En ce qui concerne la page 'prévisions des charges', les fichiers CSV sont traités en divisant le contenu en lignes. Chaque ligne est à son tour divisée en parties en utilisant la virgule comme séparateur. Ces parties sont ensuite transformées en un dictionnaire avec les clés 'number' et 'charge'. Seules les lignes avec une virgule sont traitées pour s'assurer que seules les lignes de données sont incluses. Les dictionnaires ainsi créés sont ajoutés à la liste csv_content_list, représentant les données extraites de tous les fichiers CSV dans le sous-dossier du mois en cours.

IV. Exportation des données

Pour exporter efficacement nos données, nous avons utilisé les DataFrames créés pour les enregistrer dans divers formats de fichiers tels qu'Excel, CSV ou texte.

- Données du réseau (mise à jour au 5min) : Nous avons commencé par vérifier l'existence du fichier Excel, nommé `donneesdureseau_5min.xlsx`, dans le répertoire actuel. Si le fichier existe, Pandas est utilisé pour charger le DataFrame existant depuis le fichier Excel. Ensuite, nous concaténons ce DataFrame avec le nouveau DataFrame créé précédemment, garantissant ainsi l'ajout des nouvelles données aux existantes. Si le fichier Excel n'existe pas, le code utilise simplement le nouveau DataFrame. Enfin, le DataFrame combiné est exporté vers le fichier Excel.

- Données archivées : Nous avons ouvert le fichier CSV en mode écriture ('w') et créé un objet `DictWriter` pour écrire les données dans le fichier. Les noms de champ (fieldnames) sont extraits des clés du premier dictionnaire dans la liste `all_data`. L'en-tête du fichier CSV est écrit

uniquement si le fichier CSV est nouvellement créé, puis chaque ligne de données est écrite dans le fichier.

- Interruptions de transport : Le fichier texte `interruptions_transport.txt` est ouvert en mode ajout ('a'), ce qui garantit que les données existantes ne sont pas écrasées, mais que les nouvelles données sont ajoutées à la fin du fichier. Une ligne indiquant le numéro de l'interruption de transport est ajoutée, suivie de l'itération à travers les clés et les valeurs du dictionnaire `data`, où chaque paire clé-valeur est écrite sur une nouvelle ligne dans le fichier texte.

- Prévisions de charge : Nous avons ouvert le fichier CSV en mode écriture ('w'), créé un objet avec un délimiteur d'espace (' ') et un séparateur de ligne. Ensuite, nous avons parcouru la liste `csv_content_list` contenant des dictionnaires représentant les données extraites de chaque fichier CSV, et chaque ligne du fichier CSV a été écrite en utilisant ces dictionnaires.

V. Stockage dans une base de donnée

Pour garantir l'efficacité et la gestion optimale d'un grand volume de données, notre script Python utilise une base de données relationnelle en local pour stocker les informations extraites des différentes sources d'Energie NB.

La configuration de la base de données est réalisée avec SQLAlchemy, en utilisant le moteur 'sqlite:///energieNB.db', assurant une connexion transparente entre le script Python et la base de données locale. Cette configuration permet non seulement une récupération efficace des données, mais également un stockage structuré et organisé. L'utilisation de sessions SQLAlchemy offre une gestion efficace des transactions, permettant ainsi une manipulation sécurisée et cohérente des données tout au long du processus.

Un ORM facilite la manipulation des données en les représentant sous forme d'objets Python, tout en les persistant dans une base de données relationnelle. SQLAlchemy offre une couche d'abstraction puissante, permettant ainsi de travailler avec les données de manière plus naturelle et orientée objet, sans avoir à écrire directement du SQL.

Dans notre script, nous avons défini des classes de modèle, telles que DonneeDuRéseau_5min, donnees_archivées, Interruption, et prevision_charges, qui représentent les différentes entités de notre système. Ces classes sont basées sur la classe Base fournie par SQLAlchemy, créant ainsi une structure cohérente pour stocker les données.

L'utilisation d'un ORM simplifie également la gestion des connexions à la base de données, les transactions, et l'ensemble du processus de stockage des données. Cela rend notre script plus modulaire et maintenable, en garantissant une intégration harmonieuse avec la base de données relationnelle locale

VI. Visualisation

La visualisation des données est une étape essentielle de notre projet, permettant de présenter de manière claire et interactive les informations extraites et traitées. Nous utilisons principalement la bibliothèque Matplotlib pour créer des graphiques et visualisations, offrant ainsi un aperçu visuel des tendances et des modèles inhérents aux données énergétiques.

- Données du réseau : Les données sont extraites de la table 'donnees' à l'aide d'une requête SQLAlchemy. Une liste des colonnes à extraire est préalablement définie, comprenant des éléments tels que la date, la charge, la demande, les réserves, etc. Un dictionnaire, donnees_par_colonne, est initialisé pour stocker les listes de données pour chaque colonne. Une boucle itère à travers chaque colonne et récupère les valeurs correspondantes pour chaque donnée dans la base de données. Une boucle supplémentaire parcourt le dictionnaire des données par colonne. Pour chaque colonne, un tracé est généré en utilisant la date comme axe des x et les valeurs spécifiques de cette colonne comme axe des y. La colonne 'date' est exclue du tracé pour éviter des graphiques redondants.
- Données archivées : Pour cette source de données, les données extraites étaient vraiment nombreuses. Pour une visualisation plus intéressante nous avons choisi de tracer les charges au Nouveau-Brunswick de l’année 2017. Les données de la charge au Nouveau-

Brunswick pour l'année 2017 sont extraites à partir de la liste all_data. La liste donnees_2017 est créée en récupérant les valeurs spécifiques liées à la colonne "CHARGE_AU_NB". Un graphique est créé en utilisant plt.plot() avec les données extraites pour l'année 2017.

- Interruptions de transport : Un graphique à barres est créé pour représenter les dates planifiées de début (en bleu) et de fin (en rouge) pour chaque interruption. Les numéros d'interruption (interruption_numbers) sont utilisés sur l'axe des x, et les dates planifiées (dates_start et dates_stop) sont représentées sur l'axe des y.
- Prévisions des charges : Dans cette section, nous explorons l'évolution des charges associées à un numéro spécifique. Tout commence par le chargement des données depuis le fichier CSV, extrait précédemment. Le numéro choisi pour cette exploration est "20041029000000AD". Nous filtrons les données pour isoler celles associées à ce numéro spécifique, permettant ainsi une analyse ciblée de son évolution au fil du temps. Une vérification est effectuée pour s'assurer que des données sont disponibles pour ce numéro particulier. En cas d'absence, un avertissement est affiché, indiquant qu'aucune donnée n'est disponible. Si des données sont présentes, nous extrayons les charges associées et générons une courbe pour visualiser leur évolution. La courbe est représentée en fonction de l'index des charges, offrant ainsi une perspective graphique de la fluctuation des charges pour le numéro "20041029000000AD".

Conclusion

En conclusion, ce projet axé sur l'extraction et l'analyse des données de consommation énergétique au Canada, avec une focalisation particulière sur Énergie NB, a représenté une exploration approfondie des technologies avancées dans le domaine. Nous avons utilisé des outils tels que Python, Selenium, BeautifulSoup, et SQLAlchemy pour collecter, traiter, transformer, et stocker efficacement des informations cruciales.

La première partie du projet a mis en lumière les outils et technologies clés utilisés.
La deuxième partie a détaillé le processus de récupération des données, mettant en évidence des méthodes spécifiques pour extraire des informations variées depuis les pages web d'Énergie NB. De la collecte de données en temps réel au téléchargement d'historiques et à la capture des prévisions de charges, chaque étape a été soigneusement orchestrée à l'aide d'outils adaptés.

Le traitement et la transformation des données ont été abordés de manière rigoureuse dans la troisième partie. Des structures tabulaires organisées avec Pandas aux manipulations complexes des données archivées, chaque source a été traitée de manière à faciliter une analyse ultérieure, tout en conservant la précision et la fiabilité des informations.

L'exportation des données a été réalisée de manière efficace dans différentes formats, offrant ainsi une flexibilité d'utilisation. Les données ont été enregistrées dans des fichiers Excel, CSV, texte, et intégrées à une base de données locale grâce à SQLAlchemy.

Enfin, la visualisation des données a permis de donner vie aux informations extraites. Des graphiques de tendances de consommation aux représentations visuelles des interruptions de transport, la présentation graphique a facilité la compréhension des résultats.

Ce projet a démontré l'efficacité de l'intégration de technologies avancées dans l'analyse de la consommation énergétique. Les méthodologies mises en œuvre, du web scraping à la visualisation, ont ouvert la voie à une compréhension approfondie des tendances énergétiques

au Nouveau-Brunswick. Les résultats obtenus ne sont pas seulement des données brutes, mais des informations structurées et exploitables qui peuvent contribuer à des décisions éclairées dans le domaine de la gestion de l'énergie.

En anticipant des résultats significatifs, ce projet a relevé le défi de traiter une variété de données énergétiques avec des approches innovantes et des technologies modernes. Cette exploration exhaustive a non seulement enrichi notre compréhension de la consommation énergétique, mais a également illustré la puissance des outils technologiques dans la résolution de problématiques complexes liées à l'énergie.

Annexes

Image 1 : Appel des différentes fonctions dans le fichier run.py

```python
#Appel de la fonction Interruption
energyDataHarvest.interruption(engine, Interruption)

#Appel de la fonction donnée_archivées
energyDataHarvest.donnée_archivées(donnees_archivées, engine)


# Appel de la fonction prévisions des charges
energyDataHarvest.previsions_de_charges(prevision_charges, engine)

# Boucle principale pour exécuter le scraper de la page "données du réseau" toutes les 5 minutes
while True:
    energyDataHarvest.scraper(engine, DonneeDuRéseau_5min)
    # Attendre 5 minutes (300 secondes) avant la prochaine exécution
    time.sleep(300)
```

Image 2 : Importation des différents modules dans le fichier EnergyDataHarvest.py

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd
import os
from sqlalchemy.orm import sessionmaker
from sqlalchemy.sql.expression import func
import time
import csv
from selenium import webdriver # Importation du module Selenium
from selenium.webdriver.common.by import By # Importation du module By pour les éléments du navigateur
from datetime import datetime
from selenium.common.exceptions import NoSuchElementException
import matplotlib.pyplot as plt
from matplotlib.ticker import MultipleLocator
import streamlit as st
```

Image 3 : Traitement des données de la page « Données du réseau (5min) »

```python
# Traitement et transformation
#-----------------------------------------------------------------------------------------------
df = pd.DataFrame([tr2_data, tr3_data, font6_data])
```

Image 4 : Récupération des données dans la page « Données archivées »

```
#Récuperation des données
#-------------------------------------------------------------------------------
# Sélectionner l'année dans la liste déroulante
select_annee = driver.find_element(By.ID, "ctl00_cphMainContent_ddlYear")
# Utiliser JavaScript pour définir la valeur de l'élément "select"
driver.execute_script(f"arguments[0].value = '{annee}';", select_annee)

# Attendre un court instant pour que la page se mette à jour
time.sleep(1)

# Sélectionner le mois dans la liste déroulante
select_mois = driver.find_element(By.ID, "ctl00_cphMainContent_ddlMonth")
# Utiliser JavaScript pour définir la valeur de l'élément "select"
driver.execute_script(f"arguments[0].value = '{mois}';", select_mois)
# Attendre un court instant pour que la page se mette à jour
time.sleep(1)

# Cliquez sur le lien "Obtenir les données"
obtenir_donnees_link = driver.find_element(By.ID, "ctl00_cphMainContent_lbGetData")
obtenir_donnees_link.click()

# Attendre un court instant pour le chargement des données
time.sleep(5)

#Récupérer les données directement depuis le site avec BeautifulSoup
soup = BeautifulSoup(driver.page_source, 'html.parser')
pre_data = soup.find('pre')
```

Image 5 : Traitement des données dans la page « Données archivées »

```
# Traitement et transformation
#-------------------------------------------------------------------------------
if pre_data:
    data_text = pre_data.get_text()
    # Traiter chaque ligne pour extraire les données
    lines = data_text.split('\n')
    header = lines[0].split(',')
    for line in lines[1:]:
        data = line.split(',')
        if len(data) == len(header):
            data_dict = dict(zip(header, data))
            all_data.append(data_dict)
```

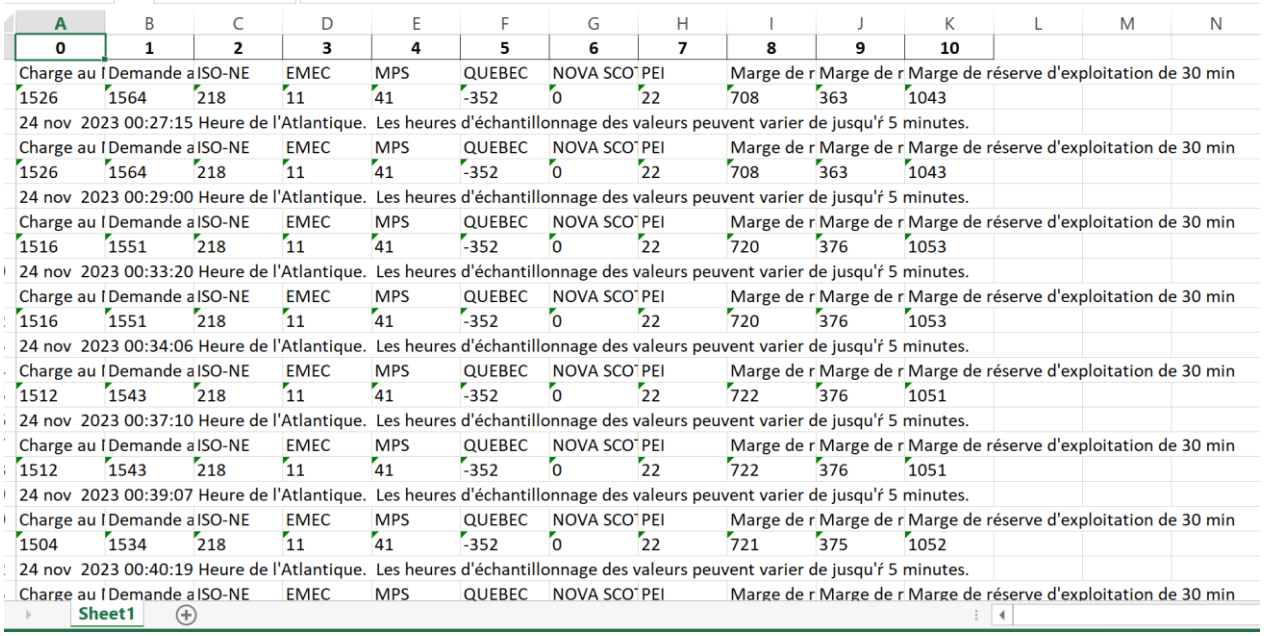
Image 6 : Visualisation des données dans la page « Interruptions de transport »

```
# Visualisation
#-----------------------------------------------------------------------------------------------------
# Tracé des dates planifiées de début
plt.figure(figsize=(10, 6))
plt.bar(interruption_numbers, dates_start, label='Planned Start', color='blue', alpha=0.7)

# Tracé des dates planifiées de fin
plt.bar(interruption_numbers, dates_stop, label='Planned Stop', color='red', alpha=0.7)

plt.xlabel('Numéro d\'interruption')
plt.ylabel('Date')
plt.title('Planned Start and Planned Stop for Each Interruption')
plt.xticks(interruption_numbers)
plt.gca().xaxis.set_major_locator(MultipleLocator(1))  # Afficher uniquement des numéros entiers sur l'axe x
plt.legend()
plt.show(block=False)
plt.pause(10)
plt.close()
```

Image 7 : Partie sur la récupération, l'exportation et le stockage des données dans la page « Prévisions des charge »

```
# Récupérer les liens des fichiers
links = driver.find_elements(By.CSS_SELECTOR, 'a[href$=".csv"]')

# Récupérer le contenu de chaque fichier et l'ajouter à la liste
for link in links:
    file_url = link.get_attribute('href')
    response = requests.get(file_url)
    csv_content = response.text
    lines = csv_content.split('\n')

    for line in lines:
        if ',' in line:
            parts = line.split(',')
            number = parts[0]
            charge = int(parts[1])  # Convertir la partie après la virgule en entier
            csv_content_list.append({'number': number, 'charge': charge})
            # Exportation des données
            #-----------------------------------------------------------------------------------------------

            for db_data in csv_content_list:
                writer.writerow([db_data['number'], db_data['charge']])

    # Ajouter les données à la base de données
    for db_data in csv_content_list:
        prevision_charge = prevision_charges(number=db_data['number'], charge=db_data['charge'])
        session.add(prevision_charge)

    # Valider les changements dans la base de données
    session.commit()
```

Image 8 : Partie du fichier excel données du réseau (5 min)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	0	1	2	3	4	5	6	7	8	9	10			
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1526	1564	218	11	41	-352	0	22	708	363	1043			
	24 nov 2023 00:27:15 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1526	1564	218	11	41	-352	0	22	708	363	1043			
	24 nov 2023 00:29:00 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1516	1551	218	11	41	-352	0	22	720	376	1053			
	24 nov 2023 00:33:20 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1516	1551	218	11	41	-352	0	22	720	376	1053			
	24 nov 2023 00:34:06 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1512	1543	218	11	41	-352	0	22	722	376	1051			
	24 nov 2023 00:37:10 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1512	1543	218	11	41	-352	0	22	722	376	1051			
	24 nov 2023 00:39:07 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			
	1504	1534	218	11	41	-352	0	22	721	375	1052			
	24 nov 2023 00:40:19 Heure de l'Atlantique. Les heures d'échantillonnage des valeurs peuvent varier de jusqu'ŕ 5 minutes.													
	Charge au I	Demande a	ISO-NE	EMEC	MPS	QUEBEC	NOVA SCO	PEI	Marge de r	Marge de r	Marge de réserve d'exploitation de 30 min			

Sheet1

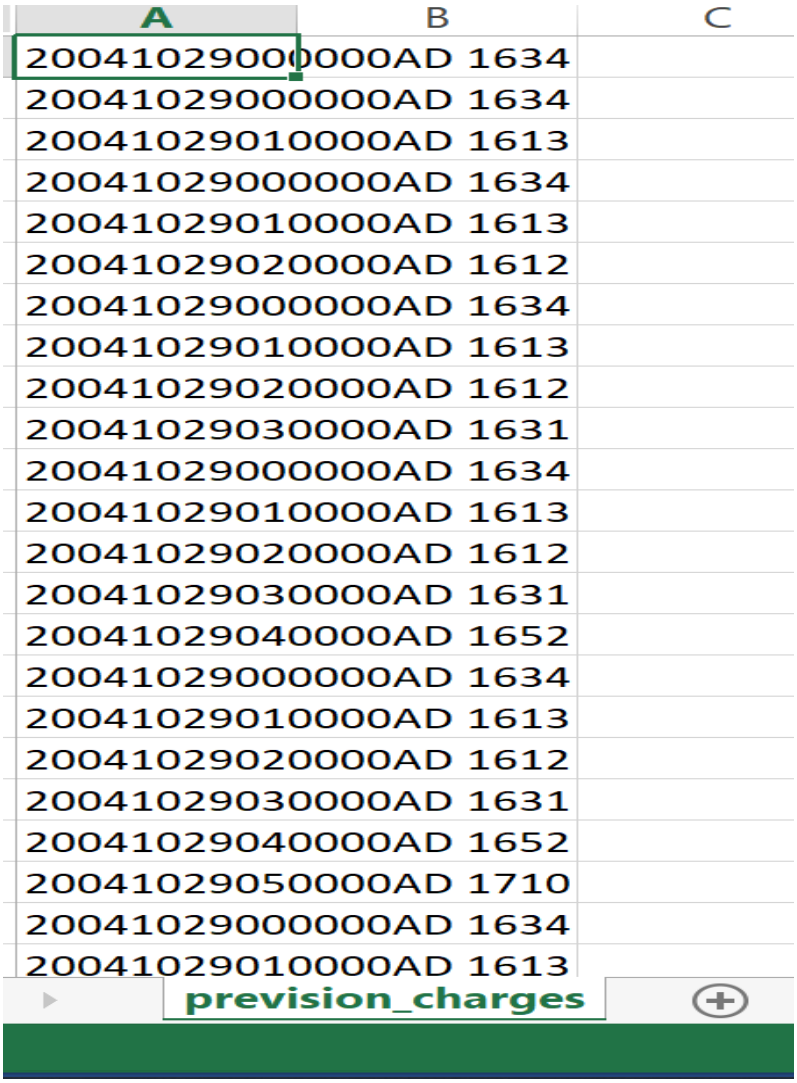
Image 9 : Fichier csv données archivées

A	B	C	D	E	F	G	H
HEURE,CHARGE_AU_NB,DEMANDE_AU_NB,ISO_NE,NMISA,QUEBEC,NOUVELLE_ECOSSE,IPE							
2016-01-01 00:00,1646,1693,945,-12,-871,0,87							
2016-01-01 01:00,1610,1649,780,-15,-729,0,84							
2016-01-01 02:00,1585,1627,804,-25,-908,0,83							
2016-01-01 03:00,1569,1611,813,-29,-890,0,68							
2016-01-01 04:00,1568,1610,849,-30,-890,0,26							
2016-01-01 05:00,1586,1628,862,-30,-890,0,12							
2016-01-01 06:00,1610,1653,882,-29,-920,0,17							
2016-01-01 07:00,1666,1711,824,-26,-920,0,11							
2016-01-01 08:00,1727,1772,662,-24,-920,0,40							
2016-01-01 09:00,1780,1826,659,-21,-939,0,52							
2016-01-01 10:00,1841,1890,666,-15,-939,0,51							
2016-01-01 11:00,1878,1929,648,-12,-939,0,67							
2016-01-01 12:00,1886,1938,686,-10,-939,0,77							
2016-01-01 13:00,1868,1922,752,-13,-939,0,82							
2016-01-01 14:00,1843,1895,674,-14,-939,0,60							
2016-01-01 15:00,1812,1864,738,-14,-939,0,67							
2016-01-01 16:00,1813,1864,726,-18,-937,0,71							
2016-01-01 17:00,1894,1951,845,-15,-919,0,89							
2016-01-01 18:00,1982,2043,863,-6,-920,0,81							
2016-01-01 19:00,1968,2027,798,-7,-939,0,69							
2016-01-01 20:00,1948,2005,800,-9,-939,0,63							
2016-01-01 21:00,1929,1984,791,-18,-939,0,69							

Image 10 : Fichier Interruptions de transport

```
Interruption De Transport #26693
additional_information: See STOP 23-096-KC
affected_interfaces: MECL, NSPI
category: 1
forced_outage: False
interaction_impacts:
mailto: mailto:ECCOutagePlanning@nbpower.com
planned_start: 2023/12/15 08:30
planned_stop: 2023/12/15 16:00
purpose: urgent outage for repairing NO3-4D1
queued_time: 2023/12/12 15:56
status: APPROVED
status_comments:

Interruption De Transport #28047
additional_information: See STOP 23-096-KC
affected_interfaces: MECL, NSPI
category: 1
forced_outage: False
interaction_impacts:
mailto: mailto:ECCOutagePlanning@nbpower.com
planned_start: 2023/12/15 08:30
planned_stop: 2023/12/15 16:00
purpose: urgent outage for repairing NO3-4D1
queued_time: 2023/12/12 15:56
status: APPROVED
status_comments:

Interruption De Transport #28173
additional_information: See STOP 23-096-KC
affected_interfaces: MECL, NSPI
category: 1
forced_outage: False
interaction_impacts:
mailto: mailto:ECCOutagePlanning@nbpower.com
planned_start: 2023/12/15 08:30
planned_stop: 2023/12/15 16:00
```

Image 11 : Fichier excel prévisions des charges



A	B	C
20041029000000AD 1634		
20041029000000AD 1634		
20041029010000AD 1613		
20041029000000AD 1634		
20041029010000AD 1613		
20041029020000AD 1612		
20041029000000AD 1634		
20041029010000AD 1613		
20041029020000AD 1612		
20041029030000AD 1631		
20041029000000AD 1634		
20041029010000AD 1613		
20041029020000AD 1612		
20041029030000AD 1631		
20041029040000AD 1652		
20041029000000AD 1634		
20041029010000AD 1613		
20041029020000AD 1612		
20041029030000AD 1631		
20041029040000AD 1652		
20041029050000AD 1710		
20041029000000AD 1634		
20041029010000AD 1613		

prevision_charges

Références bibliographiques

https://openclassrooms.com/fr/courses/6204541-initiez-vous-a-python-pour-lanalyse-de-donnees

https://www.crummy.com/software/BeautifulSoup/

https://www.python.org/

https://requests.readthedocs.io/en/latest/

https://pandas.pydata.org/

https://matplotlib.org/

https://docs.sqlalchemy.org/en/20/orm/

https://code.visualstudio.com/

https://dev.mysql.com/downloads/mysql/

https://selenium-python.readthedocs.io/

https://streamlit.io/

https://docs.python.org/3/library/time.html