Open-Data des accidents de la circulation 2019

Chargement d'un fichier CSV

Nous allons charger le fichier `caracteristiques` de 2019 dans un tableau de dictionnaire de donnée

```
import csv

def load_csv_file (filename):
  """Charge un fichier CSV"""
  with open(filename, newline='') as csvfile:
    # On utilise la bibliothèque csv de Python pour transformer chaque enregistrement du fichier en un objet
    reader = csv.DictReader(csvfile, delimiter=';')
    # On construit le tableau en compréhension contenant chaque enregistrement sous la forme d'un dictionnaire
    return [record for record in reader]

accidents_2019 = load_csv_file('./data/2019-caracteristiques.csv')
print(accidents_2019[0])
```

```
{'Num_Acc': '201900000001', 'jour': '30', 'mois': '11', 'an': '2019', 'hrmn': '01:30', 'lum': '4', 'dep': '93', 'com': '93053', 'agg': '1', 'int': '1', 'atm': '1', 'col': '2', 'adr': 'AUTOROUTE A3', 'lat': '48,8962100', 'long': '2,4701200'}
```

Réponses aux questions

Nous allons principalement utiliser des listes en compréhension pour filtrer les données

```
# Liste des accidents du Rhône
accidents_2019_rhones = [accident for accident in accidents_2019 if accident['dep'] == '69']
print(f'1. {len(accidents_2019_rhones)} accidents dans le département du Rhônes')

# Liste des numéros d'accident impliquant au moins un vélo
vehicules_2019 = load_csv_file('./data/2019-vehicules.csv')
velos_2019 = [vehicule for vehicule in vehicules_2019 if vehicule['catv'] == '1']
numeros_accidents_avec_velos = [numero['Num_Acc'] for numero in velos_2019]

accidents_avec_velos_2019 = [accident for accident in accidents_2019 if accident['Num_Acc'] in numeros_accidents_avec_velos]
```

```
print(f'2. {len(accidents_avec_velos_2019)} accidents impliquant au moins un vélo')

# Fonction retournant la date de l'accident
def date_accident(accident):
  return f'{accident["an"]}{accident["mois"]:0>2}{accident["jour"]:0>2}
{accident["hrmn"]}'

# Les accidents doivent être ordonnés par date, pour cela il faut construire la date à
partir des propriétés an, mois, jour et heures/minutes
accidents_2019_sorted = sorted(accidents_2019, key=date_accident)
premier_accident = accidents_2019_sorted[0]
print(f'3. Le premier accident de l\'année a eu lieu au coordonnées
({premier_accident["lat"]}, {premier_accident["long"]})')
```

```
1. 2523 accidents dans le département du Rhônes
2. 4834 accidents impliquant au moins un vélo
3. Le premier accident de l'année a eu lieu au coordonnées ( -20,9611120, 55,6577380)
```

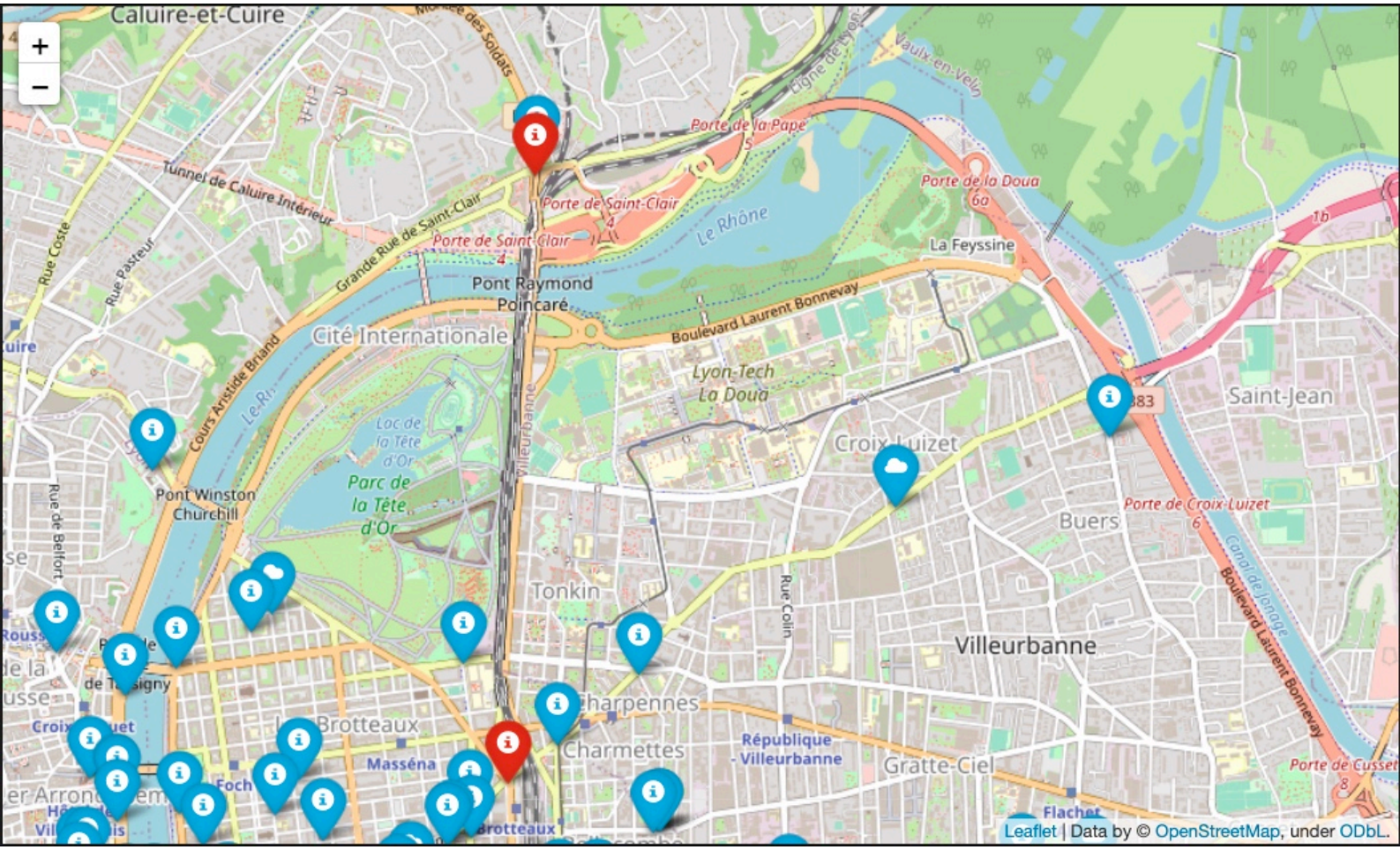
```
import os

# Constuction du jeu de données indiquant les accidents de vélos de 2019

usagers_2019 = load_csv_file('./data/2019-usagers.csv')
numeros_accidents_mortels_2019 = [usager['Num_Acc'] for usager in usagers_2019 if
usager['grav'] == '2']
accidents_velos_2019 = [{
  'Num_Acc': accident['Num_Acc'],
  'dep': accident['dep'],
  'lat': accident['lat'].replace(',', '.'),
  'long': accident['long'].replace(',', '.'),
  'date': f'{accident["an"]}-{accident["mois"]:0>2}-{accident["jour"]:0>2}',
  'hrmn': accident['hrmn'],
  'atm': accident['atm'],
  'mortel': accident['Num_Acc'] in numeros_accidents_mortels_2019,
} for accident in accidents_avec_velos_2019]

print(accidents_velos_2019[0:2])

def save_csv_file(data, filepath):
  """ Sauvegarde les données au format CSV"""
  if os.path.exists(filepath):
    os.remove(filepath)
  fd = open(filepath, 'w')
  headers = list(data[0].keys())
  csv_writer = csv.DictWriter(fd, headers)
```

```
  csv_writer.writeheader()
  csv_writer.writerows(data)

save_csv_file(accidents_velos_2019, './output/acc-velos-2019.csv')
```

```
[{'Num_Acc': '201900000037', 'dep': '67', 'lat': '48.5708480', 'long': '7.7587680',
'date': '2019-11-29', 'hrmn': '13:40', 'atm': '2', 'mortel': False}, {'Num_Acc':
'201900000038', 'dep': '66', 'lat': '42.6969500', 'long': '2.8993500', 'date': '2019-
11-29', 'hrmn': '13:50', 'atm': '1', 'mortel': False}]
```

```
import folium

accidents_velos_2019_rhones = [accident for accident in accidents_velos_2019 if
accident['dep'] == '69']

map = folium.Map(location=[45.7, 4.8], zoom_start=11)

COND_ATM = {
  '-1': 'Non renseigné',
  '1': 'Normale',
  '2': 'Pluie légère',
  '3': 'Pluie forte',
  '4': 'Neige - grêle',
  '5': 'Brouillard - fumée',
  '6': 'Vent fort - tempête',
  '7': 'Temps éblouissant',
  '8': 'Temps couvert',
  '9': 'Autre',
}

for accident in accidents_velos_2019_rhones:
  coordonnees = [accident['lat'], accident['long']]

  cond_atm = COND_ATM[accident['atm']]

  popup=f'<b>N°{accident["Num_Acc"]}</b><br>{accident["date"]} {accident["hrmn"]}<br>
{cond_atm}'

  # Si l'accident est mortel, il est affiché en rouge
  if accident['mortel']:
    color = 'red'
  else:
    color = 'blue'

  # L'icône est dépendant des conditions atmosphériques
  if accident['atm'] != '1':
```

```
    icon = folium.Icon(icon="cloud", color=color)
  else:
    icon = folium.Icon(color=color)

  # Construction et ajout du marqueur
  folium.Marker(
    coordonnees,
    popup=popup,
    icon=icon,
  ).add_to(map)

map
```


Caluire-et-Cuire
Tunnel de Caluire Intérieur
Rue Coste
Rue Pasteur
Grande Rue de Saint-Clair
Porte de Saint-Clair
Le Rhône
Porte de la Pape
Ligne de Lyon
Vaulx-en-Velin
Porte de la Doua
La Feyssine
Pont Raymond Poincaré
Boulevard Laurent Bonnevay
Cité Internationale
Cours Aristide Briand
Lyon-Tech La Doua
Croix-Luizet
Saint-Jean
Parc de la Tête d'Or
Pont Winston Churchill
Rue de Belfort
Buers
Porte de Croix-Luizet
Canal de Jonage
Tonkin
Villeurbanne
Rue Colin
Charpennes
Brotteaux
Charmettes
Masséna
Foch
République - Villeurbanne
Gratte-Ciel
Flachet
Porte de Cusset
Leaflet | Data by © OpenStreetMap, under ODbL.