TP DE VISUALISATION ELABORE PAR CT GIPOY MULENDA RODRIGUE DIRIGE PAR LE PROF MASAKUNA FELICIEN

Objectif du Projet

Le but du projet est de fournir des outils visuels pour explorer et comprendre les données démographiques en République Démocratique du Congo. Plus précisément, les visualisations permettent de :

1. Visualiser la Répartition Géographique de la Population :
   - Identifier les zones avec une forte ou faible densité de population.
   - Repérer les régions les plus peuplées et celles qui pourraient nécessiter des interventions spécifiques.
2. Analyser la Distribution de la Population par Tranche d'Âge :
   - Comprendre la structure démographique du pays.
   - Identifier les groupes d'âge qui dominent la population et les implications pour les politiques publiques et les services sociaux.

1. Carte Interactive avec Folium

Entrée [1]:
```
import folium
import pandas as pd
from folium.plugins import HeatMap
from IPython.display import display, IFrame
```

Dans le cadre de notre travail, nous considerons de données démographiques de la RDC que nous nous sommes reprentés à titre d'exemple.

Entrée [2]:
```
data = {
    'Region': ['Kinshasa', 'Lubumbashi', 'Goma', 'Kisangani', 'Bukavu'],
    'Latitude': [-4.3214, -11.6687, -1.6701, 0.5167, -2.5250],
    'Longitude': [15.3134, 27.4797, 29.2130, 25.1944, 28.8552],
    'Population': [12300000, 700000, 400000, 300000, 200000]
}
df = pd.DataFrame(data)
```

Entrée [3]:
```
m = folium.Map(location=[-2.5, 23], zoom_start=5) # Création de la carte
```

Marqueurs Circulaires : Les marqueurs de taille variable indiquent la population dans différentes régions de la RDC. Par exemple, Kinshasa, étant la capitale, a un marqueur plus grand indiquant une population plus élevée comparée à d'autres villes comme Bukavu.

Entrée [4]:
```
# Ajout de marqueur
for _, row in df.iterrows():
    folium.CircleMarker(
        location=[row['Latitude'], row['Longitude']],
        radius=row['Population'] / 2000000,  # Taille proportionnelle
        color='blue',
        fill=True,
        fill_color='blue',
        fill_opacity=0.6,
        popup=folium.Popup(f"<strong>{row['Region']}</strong><br>Population: {row['Popul
    ).add_to(m)
```

Couche de Chaleur : La couche de chaleur montre les zones avec une concentration plus élevée de population. Les zones plus chaudes sur la carte indiquent des densités de population plus élevées.

Entrée [5]:
```
# Ajouter une couche de chaleur
heat_data = [[row['Latitude'], row['Longitude'], row['Population']] for _, row in df.ite
HeatMap(heat_data, radius=15).add_to(m)
```

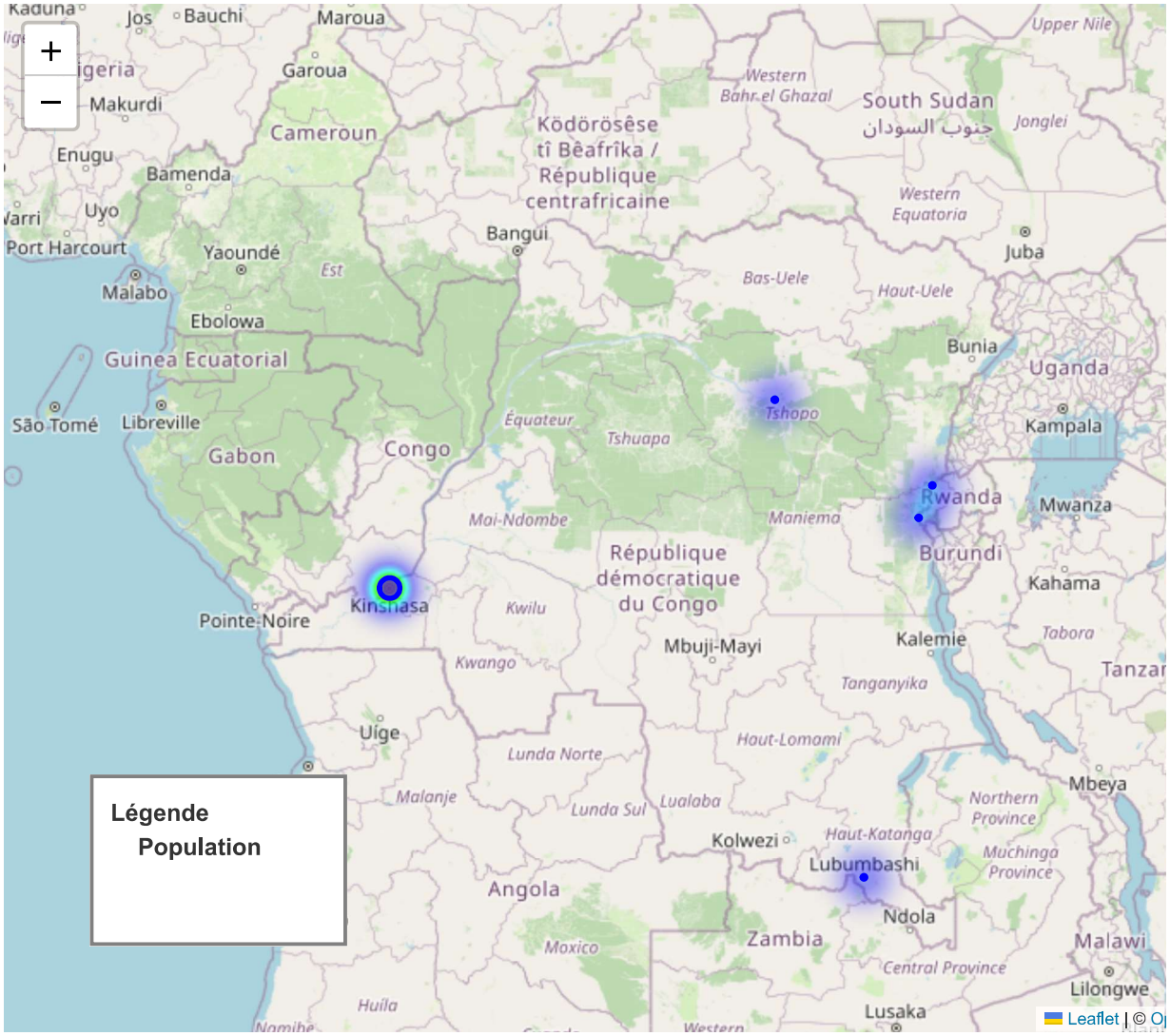
Out[5]:
```
<folium.plugins.heat_map.HeatMap at 0x1fcc5f969d0>
```

Entrée [6]:
```
# Ajouter une légende
legend_html = '''
    <div style="position: fixed;
                bottom: 50px; left: 50px; width: 150px; height: 100px;
                border:2px solid grey; background-color: white; z-index:9999;
                font-size:14px; font-weight: bold; padding: 10px;">
        <div><strong>Légende</strong></div>
        <div><i style="background:blue; width: 12px; height: 12px; display: inline-block
    </div>
'''
m.get_root().html.add_child(folium.Element(legend_html))
```

Out[6]:
```
<branca.element.Element at 0x1fcc614a340>
```

Entrée [7]:
```
# Sauvegarde de la carte
map_path = 'carte_demographique_rdc.html'
m.save(map_path)
```

Entrée [8]:
```
# Affichage de la carte
display(IFrame(src=map_path, width=800, height=600))
```



Interprétation :

* Kinshasa : Avec la plus grande population, Kinshasa est clairement le centre démographique le plus important du pays. Cela suggère un besoin potentiel élevé en infrastructure et en services dans cette région.
* Autres Régions : Les autres villes comme Lubumbashi et Goma montrent une population significative, mais moindre en comparaison avec Kinshasa. Les zones avec une densité de population plus faible pourraient bénéficier de plus d'attention pour le développement économique ou les services sociaux.
* Densité : La couche de chaleur aide à visualiser les zones avec une population plus concentrée. Les zones moins peuplées peuvent avoir des besoins différents en termes de services et d'infrastructure.

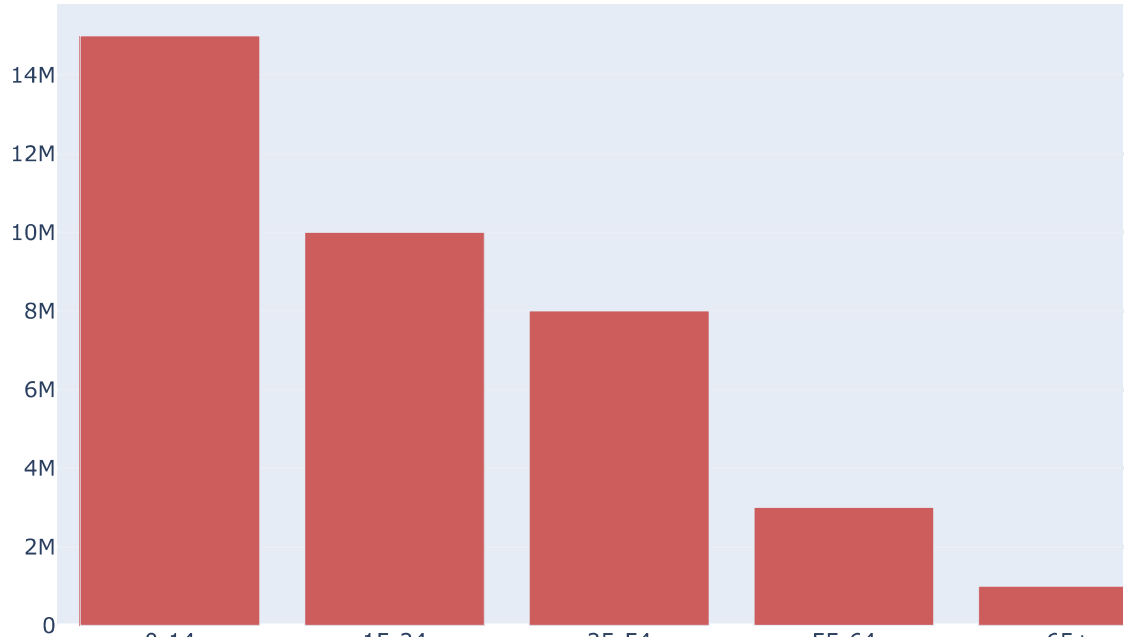
2. Graphiques Interactifs avec Plotly

Entrée [9]:
```
# Importation de données
import plotly.graph_objects as go
import plotly.express as px
import pandas as pd
```

Entrée [10]:
```
# Exemple de données démographiques par tranche d'âge pour la RDC
# Nous considerons 5 tranches d'âges
data = {
    'Age Group': ['0-14', '15-24', '25-54', '55-64', '65+'],
    'Population': [15000000, 10000000, 8000000, 3000000, 1000000]
}

df = pd.DataFrame(data)
```

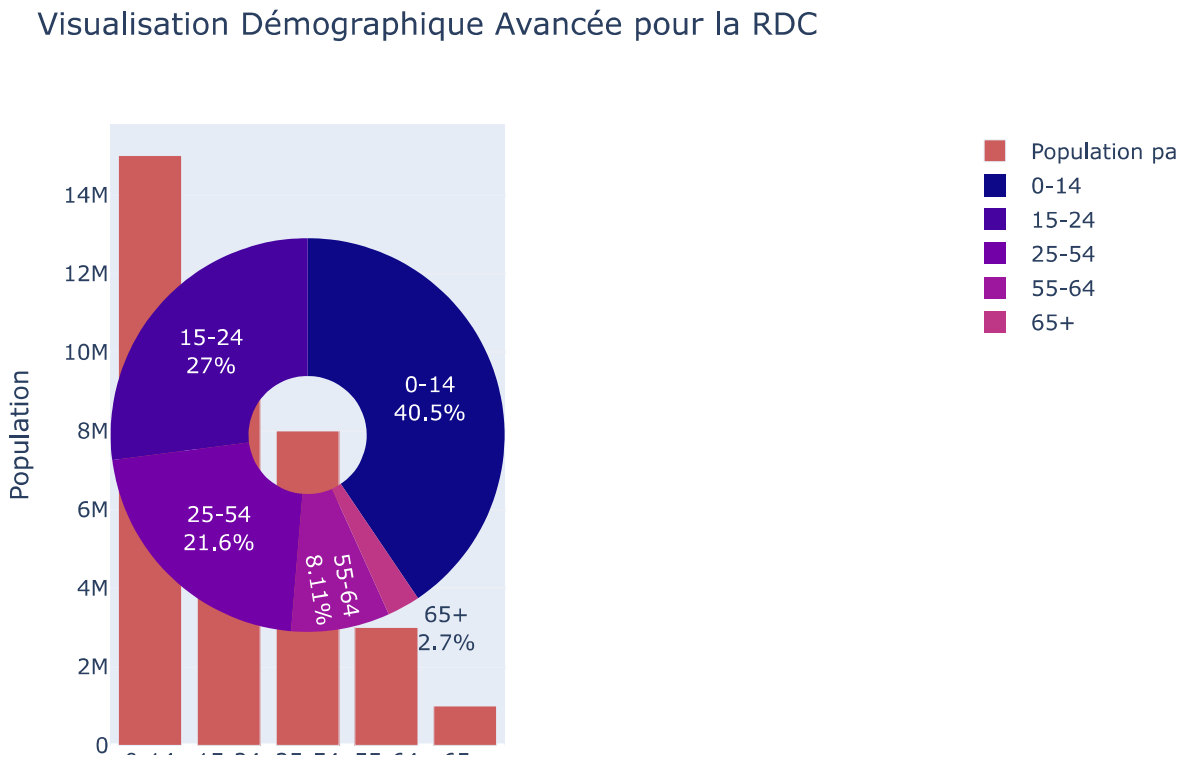
Graphique à Barres : Montre la population par tranche d'âge. Par exemple, les tranches d'âge 0-14 et 15-24 montrent des populations significatives par rapport aux tranches plus âgées.

Entrée [11]:
```
# Création du graphique à barres
fig = go.Figure()

# Ajouter les barres
fig.add_trace(go.Bar(
    x=df['Age Group'],
    y=df['Population'],
    name='Population par Tranche d\'Âge',
    marker_color='indianred'
))
```



Graphique en Camembert : Illustre la répartition proportionnelle de la population par tranche d'âge. Les tranches plus jeunes (0-14 et 15-24) occupent une plus grande part du camembert, suggérant une population jeune majoritaire.

Entrée [12]:
```
# Ajout d'un graphique en camembert
fig.add_trace(go.Pie(
    labels=df['Age Group'],
    values=df['Population'],
    name='Distribution par Tranche d\'Âge',
    hole=0.3,
    textinfo='label+percent',
    marker=dict(colors=px.colors.sequential.Plasma)
))
# Mettre à jour la disposition
fig.update_layout(
    title_text='Visualisation Démographique Avancée pour la RDC',
    grid=dict(rows=1, columns=2),
    xaxis_title='Tranche d\'Âge',
    yaxis_title='Population',
    showlegend=True
)

# Affichage du graphique
fig.show()
```



Entrée [13]:
```
import nbformat
```

Entrée [14]:
```
with open('TP_Visualisation.ipynb', 'r', encoding='utf-8') as f:
    notebook = nbformat.read(f, as_version=4)
```

Entrée [15]:
```
text_content = ""
for cell in notebook.cells:
    if cell.cell_type == 'code':
        text_content += f"### CODE CELL ###\n{cell.source}\n\n"
    elif cell.cell_type == 'markdown':
        text_content += f"### MARKDOWN CELL ###\n{cell.source}\n\n"
```

Entrée [16]:
```
with open('TP_Visualisation.txt', 'w', encoding='utf-8') as f:
    f.write(text_content)
```

Entrée [17]:
```
print("Le contenu du notebook a été extrait avec succès dans le fichier 'TP_Visualisatio
```

```
Le contenu du notebook a été extrait avec succès dans le fichier 'TP_Visualisation.tx
t'.
```