Rapport d'Analyse de Données

📄 Description du Script

Un script Python permettant de générer un fichier SQL pour charger la base de données avec les données du fichier Excel.

🔎 Remarque importante concernant les rôles des utilisateurs dans les interactions 🔎

Afin d'éviter toute confusion lors de l'analyse des données, voici une clarification sur le sens de certains identifiants dans les tables liées aux interactions entre utilisateurs :

🟦 Table `reaction`

- `id_utilisateur_1` : correspond à **l'utilisateur qui effectue une réaction** (like, partage, commentaire...)
- `id_utilisateur` : correspond à **l'auteur de la publication sur laquelle la réaction a eu lieu**

→ Exemple : Si l'utilisateur 5 like une publication de l'utilisateur 2, alors `id_utilisateur_1 = 5`, `id_utilisateur = 2`.

🟦 Table `reponse_a`

- `id_utilisateur` : représente **l'utilisateur qui répond à une publication**
- `id_utilisateur_1` : désigne **l'auteur de la publication à laquelle la réponse est faite**

→ Exemple : Si l'utilisateur 3 répond à une publication de l'utilisateur 7, alors `id_utilisateur = 3`, `id_utilisateur_1 = 7`.

🟦 `Table Message`

- `id_utilisateur` : désigne **l'expéditeur du message**
- `id_utilisateur_1` : désigne **le destinataire du message**

→ Exemple : Si l'utilisateur 10 envoie un message à l'utilisateur 4, alors `id_utilisateur = 10`, `id_utilisateur_1 = 4`.

Nous avons souhaité expliciter cela pour garantir une bonne lecture des scripts et une compréhension claire des logiques relationnelles au sein de la base de données.

SCRIPT :

In [3]:
```
import pandas as pd


df = pd.read_excel("dataS204.xlsx", sheet_name="Data")
df.columns = df.columns.str.strip()
for col in df.select_dtypes(include='object').columns:
    df[col] = df[col].map(lambda x: x.strip() if isinstance(x, str) else x)


fichier = open("script_sae.sql", "w", encoding="utf-8")


#scripte de creation des tableaux
#------------------------------------------------------------------------------

fichier.write("CREATE TYPE Genre AS ENUM ('Homme', 'Femme', 'non-binaire');\n")
fichier.write("CREATE TYPE Typeabo AS ENUM ('Gratuit', 'Premium', 'Entreprise');
fichier.write("CREATE TYPE Typereact AS ENUM ('like', 'partage', 'commentaire',

# ------------------ creation des tables ------------------

fichier.write("""
CREATE TABLE utilisateur (
    id_utilisateur SERIAL,
    genre Genre,
    username VARCHAR NOT NULL,
    date_de_naissance DATE,
    niveau_education VARCHAR,
    type_d_abonnement Typeabo NOT NULL,
    PRIMARY KEY (id_utilisateur)
);

CREATE TABLE session (
    id_session SERIAL,
    date_debut TIMESTAMP,
    date_fin TIMESTAMP,
    localisation VARCHAR,
    id_utilisateur INTEGER,
    PRIMARY KEY (id_session),
    FOREIGN KEY (id_utilisateur) REFERENCES utilisateur(id_utilisateur)
);

CREATE TABLE page (
    id_page SERIAL,
    titre VARCHAR,
    PRIMARY KEY (id_page)
);

CREATE TABLE pub (
    id_pub SERIAL,
    contenu VARCHAR(50),
    PRIMARY KEY (id_pub)
);

CREATE TABLE notification (
    id_utilisateur INT,
```

```
    id_notification SERIAL,
    contenu VARCHAR(50),
    PRIMARY KEY (id_utilisateur, id_notification),
    FOREIGN KEY (id_utilisateur) REFERENCES utilisateur(id_utilisateur)
);

CREATE TABLE publication (
    id_page INT,
    id_utilisateur INT,
    id_publication SERIAL,
    contenu VARCHAR,
    date_publication DATE,
    PRIMARY KEY (id_page, id_utilisateur, id_publication),
    FOREIGN KEY (id_page) REFERENCES page(id_page),
    FOREIGN KEY (id_utilisateur) REFERENCES utilisateur(id_utilisateur)
);

CREATE TABLE groupe (
    id_page INT,
    id_groupe SERIAL,
    PRIMARY KEY (id_page, id_groupe),
    FOREIGN KEY (id_page) REFERENCES page(id_page)
);

CREATE TABLE reaction (
    id_page INT,
    id_utilisateur INT,
    id_utilisateur_1 INT,
    id_publication INT,
    id_reaction SERIAL,
    type Typereact NOT NULL,
    contenu VARCHAR(50),
    PRIMARY KEY (id_page, id_utilisateur, id_publication, id_utilisateur_1, id_r
    FOREIGN KEY (id_page, id_utilisateur, id_publication) REFERENCES publication
    FOREIGN KEY (id_utilisateur_1) REFERENCES utilisateur(id_utilisateur)
);

CREATE TABLE rejoint (
    id_utilisateur INT,
    id_page INT,
    id_groupe INT,
    PRIMARY KEY (id_utilisateur, id_page, id_groupe),
    FOREIGN KEY (id_utilisateur) REFERENCES utilisateur(id_utilisateur),
    FOREIGN KEY (id_page, id_groupe) REFERENCES groupe(id_page, id_groupe)
);

CREATE TABLE clique (
    id_session INT,
    id_pub INT,
    PRIMARY KEY (id_session, id_pub),
    FOREIGN KEY (id_session) REFERENCES session(id_session),
    FOREIGN KEY (id_pub) REFERENCES pub(id_pub)
);

CREATE TABLE message (
    id_utilisateur INT,
    id_utilisateur_1 INT,
    coutenu VARCHAR(50),
    PRIMARY KEY (id_utilisateur, id_utilisateur_1),
    FOREIGN KEY (id_utilisateur) REFERENCES utilisateur(id_utilisateur),
```

```
    FOREIGN KEY (id_utilisateur_1) REFERENCES utilisateur(id_utilisateur)
);

CREATE TABLE accede_a (
    id_session INT,
    id_page INT,
    PRIMARY KEY (id_session, id_page),
    FOREIGN KEY (id_session) REFERENCES session(id_session),
    FOREIGN KEY (id_page) REFERENCES page(id_page)
);

CREATE TABLE administre (
    id_utilisateur INT,
    id_page INT,
    PRIMARY KEY (id_utilisateur, id_page),
    FOREIGN KEY (id_utilisateur) REFERENCES utilisateur(id_utilisateur),
    FOREIGN KEY (id_page) REFERENCES page(id_page)
);

CREATE TABLE IF NOT EXISTS reponse_a (
    id_page INT,
    id_utilisateur INT,
    id_publication INT,
    id_page_1 INT,
    id_utilisateur_1 INT,
    id_publication_1 INT,
    PRIMARY KEY (id_page, id_utilisateur, id_publication, id_page_1, id_utilisat
    FOREIGN KEY (id_page, id_utilisateur, id_publication) REFERENCES publication
    FOREIGN KEY (id_page_1, id_utilisateur_1, id_publication_1) REFERENCES publi
);

""")
#-------------------------------------------------------------------------------

#utilisateur

i = 0
while i < len(df):
    ligne = df.iloc[i]


    id_utilisateur = ligne["Id"]
    genre = ligne['F (Sexe)']
    username = "utilisateur_" + str(id_utilisateur)
    age = ligne['G (Âge)']
    niveau = ligne["I (Niveau d'éducation)"]
    abo= ligne["J (Type d'abonnement)"]

    annee_naissance = 2025 - int(age)
    date_naissance = str(annee_naissance) + "-01-01"




    requete = "INSERT INTO utilisateur (id_utilisateur, genre, username, date_de
    requete = requete + str(id_utilisateur) + ", "
    requete = requete + "'" + genre + "', "
    requete = requete + "'" + username + "', "
    requete = requete + "'" + date_naissance + "', "
    requete = requete + "'" + str(niveau) + "', "
```

```
    requete = requete + "'" + abo + "');\n"


    fichier.write(requete)

    i = i + 1
print("Creation terminer de utilisateur")

#session
i = 0
id_utilisateur = 1
id_session = 1


while i < len(df):
    ligne = df.iloc[i]

    temps = int(ligne["E (Temps passé)"])
    loc = ligne["H (Localisation)"]
    heure_debut = 10
    minutes_debut = 0


    heure_fin = heure_debut + (temps // 60)
    minutes_fin = temps % 60

    date_debut = "2025-04-01 " + str(heure_debut).zfill(2) + ":" + str(minutes_d
    date_fin = "2025-04-01 " + str(heure_fin).zfill(2) + ":" + str(minutes_fin).

    sql = "INSERT INTO session VALUES (" + str(id_session) + ", '" + date_debut
    fichier.write(sql)

    id_session += 1
    id_utilisateur += 1
    i += 1

print("Creation terminer de session")









#Pages

max_pages = int(df["AJ (Créations de pages)"].sum())
nb_pages = int(max_pages) + 1

id_page = 1
while id_page <= nb_pages:
    titre = "Page_" + str(id_page)
    requete = "INSERT INTO page (id_page, titre) VALUES (" + str(id_page) + ", '
    fichier.write(requete)
    id_page = id_page + 1

print("Creation terminer de page")
```

```
#pub
i=1
while i <= 1000:
    contenu = "contenu_" + str(i)
    requete = "INSERT INTO pub (contenu) VALUES ('" + contenu + "');\n"

    fichier.write(requete)


    i+= 1

print("Creation terminer de pub")

#notification

notification_id = 1
i = 0
while i < len(df):
    ligne = df.iloc[i]


    id_utilisateur = ligne["Id"]
    nombre_notifications = ligne["L (Notifications)"]


    j = 0
    while j < nombre_notifications:
        contenu = "notification_" + str(notification_id)
        requete = "INSERT INTO notification (id_utilisateur, contenu) VALUES ("

        fichier.write(requete)

        # Passer à la notification suivante
        notification_id += 1
        j += 1

    i += 1
print("Creation terminer de notif")



# Création des publications
publication_list = []
date_pub = "2025-03-03"
max_pages = int(df["AJ (Créations de pages)"].sum())

id_pub = 1
i = 0
while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_publications = int(ligne["AL (Publications sur pages)"])
    nb_pages_membre = int(ligne["AK (Membres sur pages)"])

    page_id = 1
    compteur_pub = 0

    while compteur_pub < nb_publications and compteur_pub < nb_pages_membre:
```

```
        contenu = "Contenu publication " + str(compteur_pub + 1)
        requete = "INSERT INTO publication (id_page, id_utilisateur, contenu, da
        requete += str(page_id) + ", "
        requete += str(id_utilisateur) + ", "
        requete += "'" + contenu + "', "
        requete += "'" + date_pub + "');\n"

        fichier.write(requete)


        publication_list.append((page_id, id_utilisateur, id_pub))

        compteur_pub += 1
        page_id += 1
        id_pub += 1

        if page_id > max_pages:
            page_id = 1

    i += 1
print("Création terminée des publications.")

#groupe




max_groupes = int(df["V (Créations de groupes)"].sum())
nb_pages = int(df["AJ (Créations de pages)"].sum())

id_groupe = 1
id_page = 1
groupe_page_map = {}

while id_groupe <= max_groupes:
    requete = "INSERT INTO groupe (id_groupe, id_page) VALUES (" + str(id_groupe
    fichier.write(requete)

    groupe_page_map[id_groupe] = id_page

    id_groupe += 1
    id_page += 1
    if id_page > nb_pages:
        id_page = 1

print("Création terminée des groupes")


#Reaction
print("la creation des lignes de la table reaction vient de commencer cela peut
publications = []
i = 0
page_id_global = 1
while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_pages_membre = int(ligne["AL (Publications sur pages)"])
    nb_publications = int(ligne['AL (Publications sur pages)'])
    nb_pages_membre1 = int(ligne["AK (Membres sur pages)"])
    page_id_global  = 1
```

```
    compteur_pub = 0
    while compteur_pub < nb_publications and page_id_global <= nb_pages_membre1:
        pub = {
            "id_publication": len(publications) + 1,
            "id_utilisateur": id_utilisateur,
            "id_page": page_id_global
        }
        publications.append(pub)

        page_id_global += 1
        compteur_pub += 1
        if page_id_global  > nb_pages_membre :
            page_id_global  = 1

    i += 1

# Génération des réactions
i = 0
while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur_1 = ligne["Id"]

    nb_likes = int(ligne["A (Likes)"])
    nb_partages = int(ligne["B (Partages)"])
    nb_commentaires = int(ligne["C (Commentaires)"])
    nb_participations = int(ligne["U (Participations événements)"])

    type_list = [("like", nb_likes), ("partage", nb_partages), ("commentaire", n

    j = 0
    while j < len(type_list):
        type_react = type_list[j][0]
        nb_react = type_list[j][1]

        compteur = 0
        index_publi = 0
        reactions_deja_faites = []

        while compteur < nb_react and index_publi < len(publications):
            pub = publications[index_publi]
            id_publication = pub["id_publication"]
            id_utilisateur_publi = pub["id_utilisateur"]
            id_page = pub["id_page"]

            deja_reagit = False
            k = 0
            while k < len(reactions_deja_faites):
                if reactions_deja_faites[k][0] == id_publication and reactions_d
                    deja_reagit = True
                k += 1

            if id_utilisateur_1 != id_utilisateur_publi and deja_reagit == False
                contenu = "Contenu de " + type_react + " " + str(compteur + 1)

                requete = "INSERT INTO reaction (id_page, id_utilisateur, id_uti
                requete += str(id_page) + ", "
                requete += str(id_utilisateur_publi) + ", "
                requete += str(id_utilisateur_1) + ", "
                requete += str(id_publication) + ", "
                requete += "'" + type_react + "', "
```

```
                requete += "'" + contenu + "');\n"

                fichier.write(requete)



                reactions_deja_faites.append((id_publication, type_react))
                compteur += 1

            index_publi += 1

        j += 1
    i += 1
print("Enfin,on passe a la table rejoint")



# Rejoint




i = 0
id_groupe_global = 1
max_groupes = int(df["V (Créations de groupes)"].sum())

while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_groupes_rejoints = int(ligne["W (Membres dans groupes)"])
    nb_pages_membre = int(ligne["AK (Membres sur pages)"])

    if nb_pages_membre == 0:
        i += 1
        continue

    compteur = 0

    while compteur < nb_groupes_rejoints and id_groupe_global <= max_groupes:
        id_page_du_groupe = groupe_page_map[id_groupe_global]
        if id_page_du_groupe <= nb_pages_membre:
            requete = "INSERT INTO rejoint (id_utilisateur, id_page, id_groupe)
                      str(id_utilisateur) + ", " + str(id_page_du_groupe) + ", "
            fichier.write(requete)
            compteur += 1

        id_groupe_global += 1

    i += 1

print("Création terminée des lignes Rejoint")

# Clique

i = 0
id_pub = 1
nb_total_pubs = 1000

while i < len(df):
    ligne = df.iloc[i]
```

```
    id_utilisateur = ligne["Id"]
    id_session = id_utilisateur
    nb_clics = int(ligne["P (Clics publicités)"])

    compteur = 0
    while compteur < nb_clics :
        requete = "INSERT INTO clique (id_session, id_pub) VALUES ("
        requete += str(id_session) + ", "
        requete += str(id_pub) + ");\n"
        fichier.write(requete)

        compteur += 1
        id_pub += 1


    id_pub = 1
    i += 1
print("Creation terminer de click ")


# Message

i = 0
while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_messages = ligne["K (Messages privés)"]

    id_utilisateur_1 = 1
    compteur = 0

    while compteur < nb_messages :

        if id_utilisateur != id_utilisateur_1:
            contenu = "message_" + str(compteur + 1)
            requete = "INSERT INTO message (id_utilisateur, id_utilisateur_1, co
            requete += str(id_utilisateur) + ", "
            requete += str(id_utilisateur_1) + ", "
            requete += "'" + contenu + "');\n"

            fichier.write(requete)
            compteur += 1

        id_utilisateur_1 += 1

    i += 1

print("Creation terminer de message")




#Acceder a + administer

i = 0
id_session = 1
id_page_max = int(df["AJ (Créations de pages)"].sum())
nb_pages_total = id_page_max
```

```
pages_accedees_par_utilisateur = {}
pages_administrees = {}


id_page = 1
while id_page <= nb_pages_total:
    pages_administrees[id_page] = 0
    id_page += 1


i = 0
id_session = 1
while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_pages_membre = int(ligne["AK (Membres sur pages)"])

    pages_accedees_par_utilisateur[id_utilisateur] = []
    compteur = 0
    id_page = 1
    while compteur < nb_pages_membre and id_page <= nb_pages_total:
        requete = "INSERT INTO accede_a (id_session, id_page) VALUES (" + str(id
        fichier.write(requete)

        pages_accedees_par_utilisateur[id_utilisateur].append(id_page)

        compteur += 1
        id_page += 1

    id_session += 1
    i += 1

i = 0
id_page_courant = 1

while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_pages_creees = int(ligne["AJ (Créations de pages)"])


    if id_utilisateur in pages_accedees_par_utilisateur:
        pages_acces = pages_accedees_par_utilisateur[id_utilisateur]
    else:
        pages_acces = []


    compteur = 0
    index_page = 0
    while compteur < nb_pages_creees and index_page < len(pages_acces):
        page_id = pages_acces[index_page]

        requete = "INSERT INTO administre (id_utilisateur, id_page) VALUES (" +
        fichier.write(requete)
        pages_administrees[page_id] += 1

        compteur += 1
        index_page += 1

    i += 1
```

```
print("Creation terminer de accede_a et administre")

#reponse a

i = 0
while i < len(df):
    ligne = df.iloc[i]
    id_utilisateur = ligne["Id"]
    nb_publications = int(ligne["AL (Publications sur pages)"])
    nb_reponses = int(ligne["AN (Messages sur pages)"])

    pubs_utilisateur = [p for p in publication_list if p[1] == id_utilisateur]

    compteur_reponse = 0
    index_cible = 0
    index_source = 0

    while compteur_reponse < nb_reponses and len(pubs_utilisateur) > 0:
        if index_cible >= len(publication_list):
            index_cible = 0

        if index_source >= len(pubs_utilisateur):
            index_source = 0

        cible = publication_list[index_cible]
        source = pubs_utilisateur[index_source]

        if source != cible:
            requete = "INSERT INTO reponse_a (id_page, id_utilisateur, id_public
            requete += str(source[0]) + ", "
            requete += str(id_utilisateur) + ", "
            requete += str(source[2]) + ", "
            requete += str(cible[0]) + ", "
            requete += str(cible[1]) + ", "
            requete += str(cible[2]) + ");\n"

            fichier.write(requete)
            compteur_reponse += 1

        index_cible += 1
        index_source += 1

    i += 1

print("Création terminée de reponse_a.")




fichier.close()
```

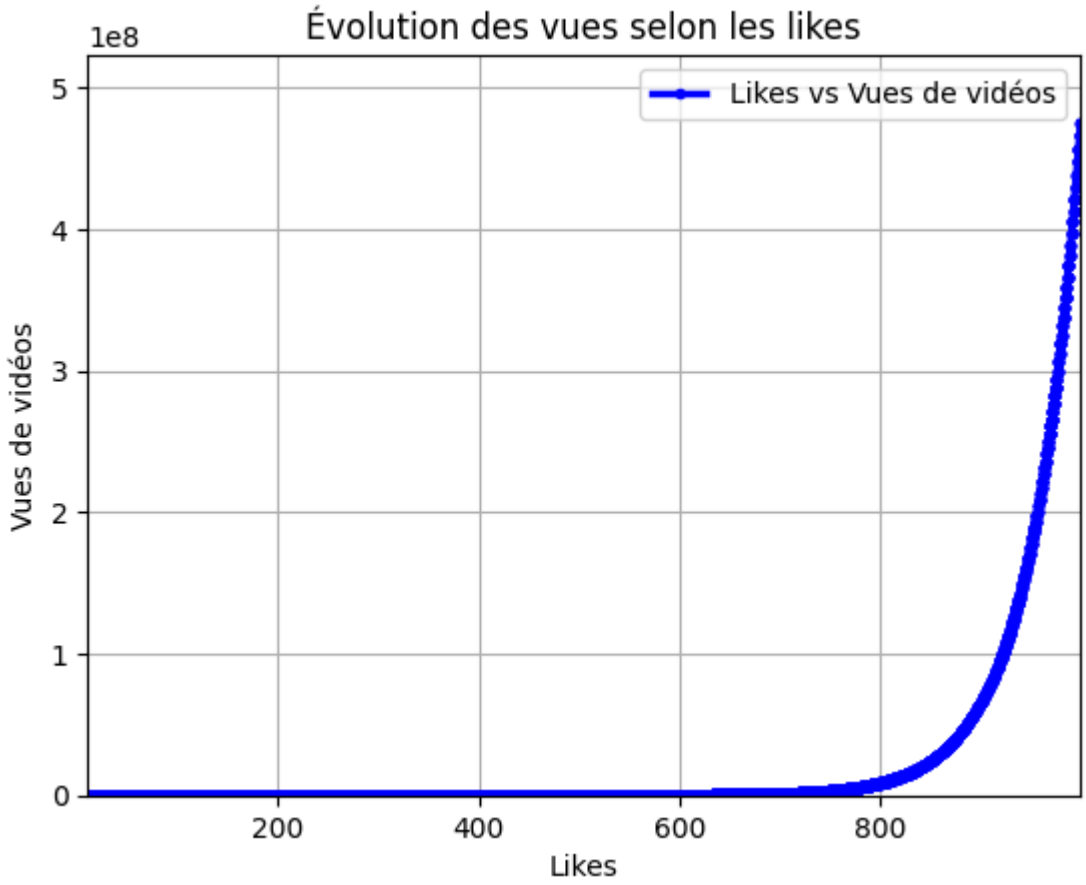
```
Creation terminer de utilisateur
Creation terminer de session
Creation terminer de page
Creation terminer de pub
Creation terminer de notif
Création terminée des publications.
Création terminée des groupes
la creation des lignes de la table reaction vient de commencer cela peut prendre
quelque minutes
Enfin,on passe a la table rejoint
Création terminée des lignes Rejoint
Creation terminer de click
Creation terminer de message
Creation terminer de accede_a et administre
Création terminée de reponse_a.
```

📄 Description du Visualisation

trois visualisations différentes mettant en avant des éléments significatifs de la base de données.

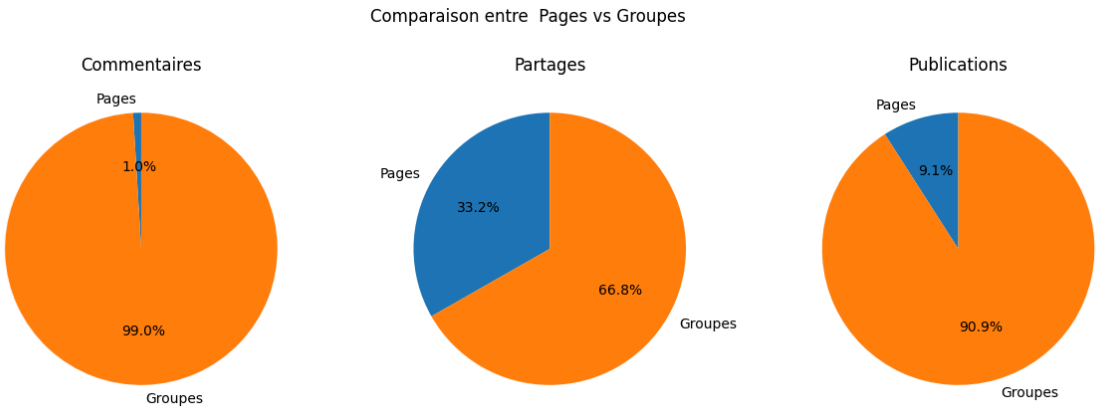
Premiere Visualisation :

In [4]:
```python
import pandas as pd
from  matplotlib.pyplot import *

df =pd.read_excel("dataS204.xlsx")
likes = df[' A (Likes) '].astype(float).tolist()
views = df['Vues de vidéos'].astype(float).tolist()
data = sorted(zip(likes, views))
X = [x for x, m in data]
Y = [y for m, y in data]
plot(X, Y, 'b.-', linewidth=2.5, label="Likes vs Vues de vidéos")
xlim(min(X), max(X))
ylim(0, max(Y) * 1.1)
title("Évolution des vues selon les likes")
xlabel("Likes")
ylabel("Vues de vidéos")
legend()
grid(True)
show()
```



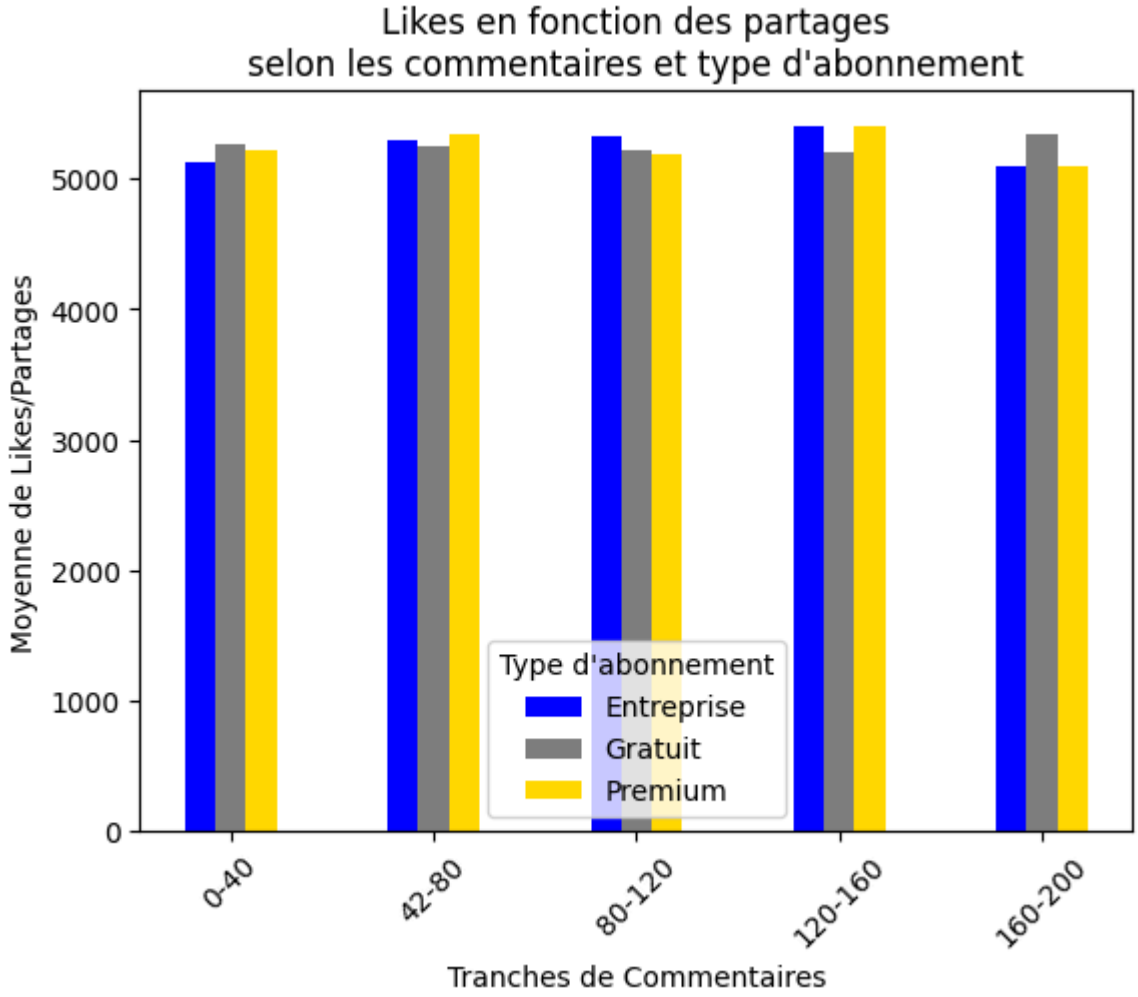
Deuxieme Visualisation :

In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_excel('dataS204.xlsx')
df.columns = [col.strip() for col in df.columns]
M = {
    'Commentaires': ('AP (Commentaires sur pages)', 'AB (Commentaires dans group
    'Partages':     ('AO (Partages sur pages)',     'AA (Partages dans groupes)'
    'Publications': ('AL (Publications sur pages)', 'X (Publications dans groupe
}
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
for ax, (M, (page_col, group_col)) in zip(axes, M.items()):
    sizes = [df[page_col].mean(), df[group_col].mean()]
    labels = ['Pages', 'Groupes']
    ax.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
    ax.set_title(M)
fig.suptitle('Comparaison entre  Pages vs Groupes ')
plt.show()
```



Trosieme Visualisation :

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt


df = pd.read_excel("dataS204.xlsx")
df.columns = df.columns.str.strip()
bins = [0, 40, 80, 120, 160, 200]
labels = ['0-40', '42-80', '80-120', '120-160', '160-200']
df['Commentaires_bin'] = pd.cut(df["C (Commentaires)"], bins=bins, labels=labels
grouped = df.groupby(["Commentaires_bin", "J (Type d'abonnement)"],observed=True
X= grouped['Commentaires_bin'].unique()
Y= [grouped[grouped['Commentaires_bin'] == x]['Likes en fonction des partages'].
H = grouped['J (Type d\'abonnement)'].unique()
fig, ax = plt.subplots()
colors = ['blue', 'gray', 'gold']
bar_width = 0.15
Tranche = range(len(X))
i = 0
for hue in H:
    y = [Y[j][i] for j in range(len(X))]
    ax.bar([p + bar_width * i for p in range(len(X))], y, bar_width, color=color
    i += 1
plt.title("Likes en fonction des partages\nselon les commentaires et type d'abon
plt.xlabel("Tranches de Commentaires")
plt.ylabel("Moyenne de Likes/Partages")
plt.xticks([p + bar_width for p in Tranche],X, rotation=45)
plt.legend(title="Type d'abonnement")
plt.show()
```


Likes en fonction des partages
selon les commentaires et type d'abonnement
Moyenne de Likes/Partages
0
1000
2000
3000
4000
5000
Type d'abonnement
Entreprise
Gratuit
Premium
0-40
42-80
80-120
120-160
160-200
Tranches de Commentaires