Université des Sciences et de la Technologie Houari Boumediene (USTHB)

Faculté d'Informatique

Master 2 Big Data

Conception d'un entrepot de données dans power bi

Module : Business Intelligence (BI)

Réalisé par :
IHADDADENE Chakib 181831091825

Guide Complet : Power BI Desktop

16 décembre 2025

Table des matières

Téléchargement Rapide

`https://aka.ms/pbidesktopstore`

1 Installation de Power BI Desktop

1.1 Prérequis Système

Composant	Configuration minimale
Système d'exploitation	Windows 10 ou 11 (64-bit)
Processeur	1 GHz ou plus rapide
Mémoire RAM	4 GB (8 GB recommandé)
Espace disque	2 GB d'espace libre
Résolution écran	1440x900 ou supérieure

TABLE 1 – Configuration système requise

1.2 Étapes d'Installation

1.2.1 Méthode 2 : Téléchargement Direct

1. Rendez-vous sur : `https://aka.ms/pbidesktopstore`
2. Cliquez sur **"Télécharger gratuitement"**
3. Exécutez le fichier `PBIDesktopSetup.exe`
4. Suivez l'assistant d'installation
5. Acceptez les conditions d'utilisation

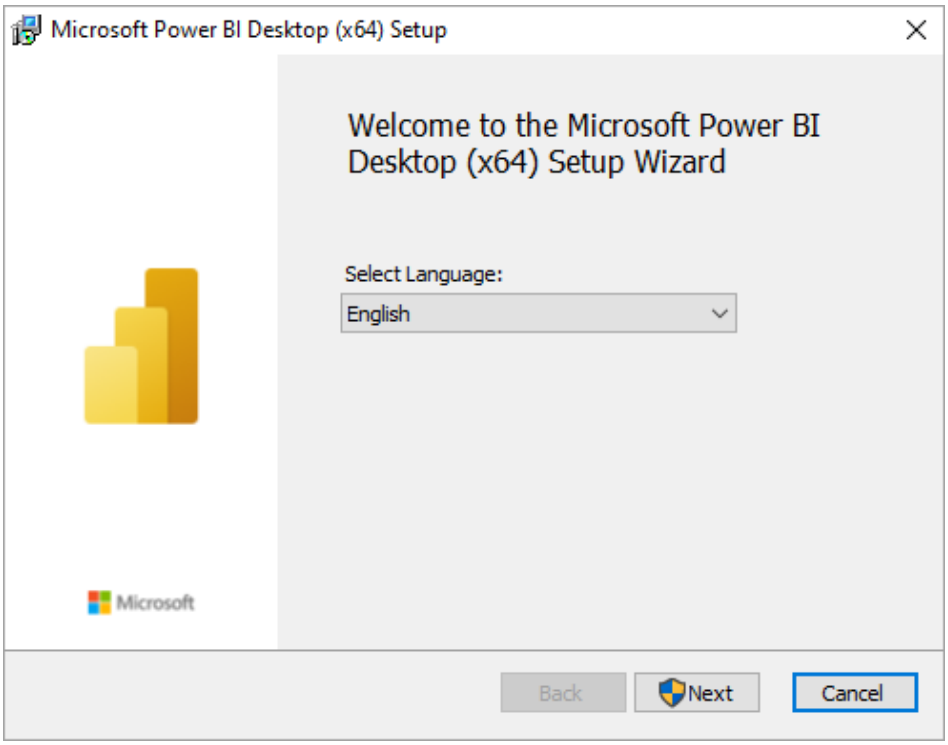

FIGURE 1 – Interface d'installation de power bi desktop

2 Premier Lancement et Configuration

2.1 Configuration Initiale

1. **Écran d'accueil** : Sélectionnez "Commencer"
2. **Connexion** : Connectez-vous avec votre compte Microsoft
3. **Thème** : Choisissez "Clair" ou "Sombre"
4. **Paramètres régionaux** : Sélectionnez "Français (France)"

2.2 Interface Utilisateur

Zone	**Fonctionnalité**
1. Ruban	Commandes principales (Fichier, Accueil, etc.)
2. Volet Visualisations	Types de graphiques disponibles
3. Volet Champs	Tables et colonnes de données
4. Zone de dessin	Construction des rapports
5. Volet Filtres	Application des filtres
6. Barre d'état	Informations et progression

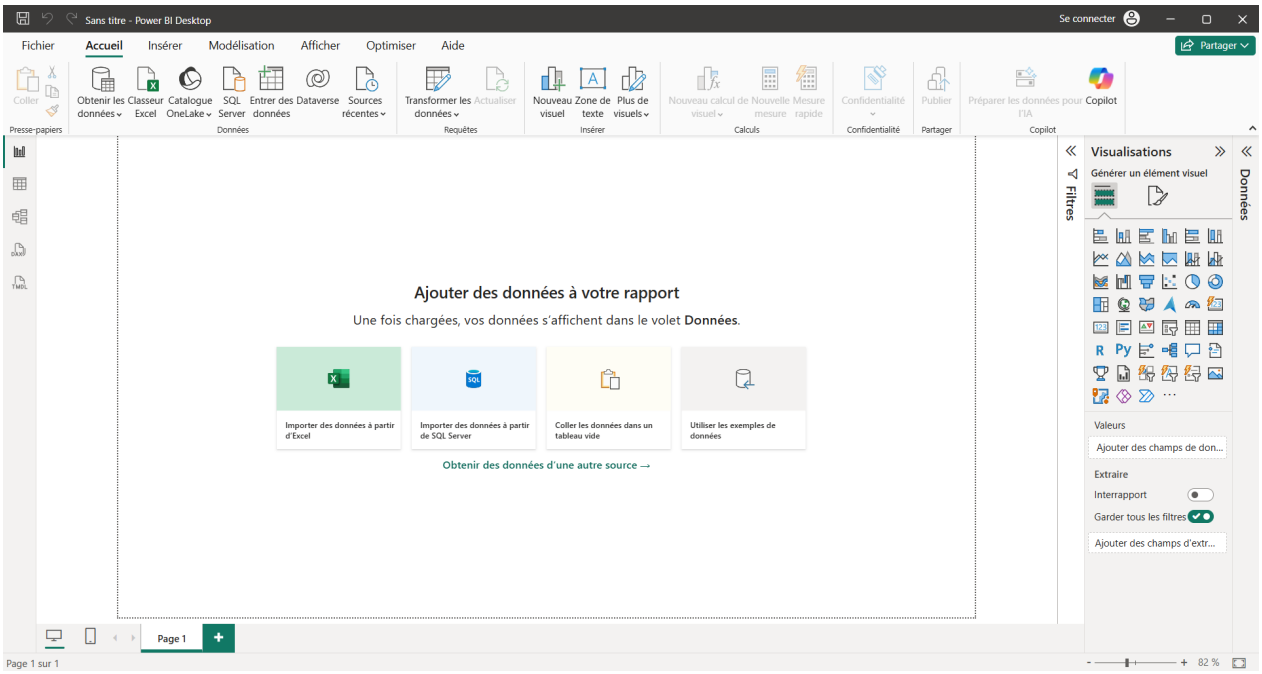

FIGURE 2 – Anatomie de l'interface Power BI Desktop

3 Architecture de l'Entrepôt de Données Northwind

3.1 Modèle en Étoile (Star Schema)

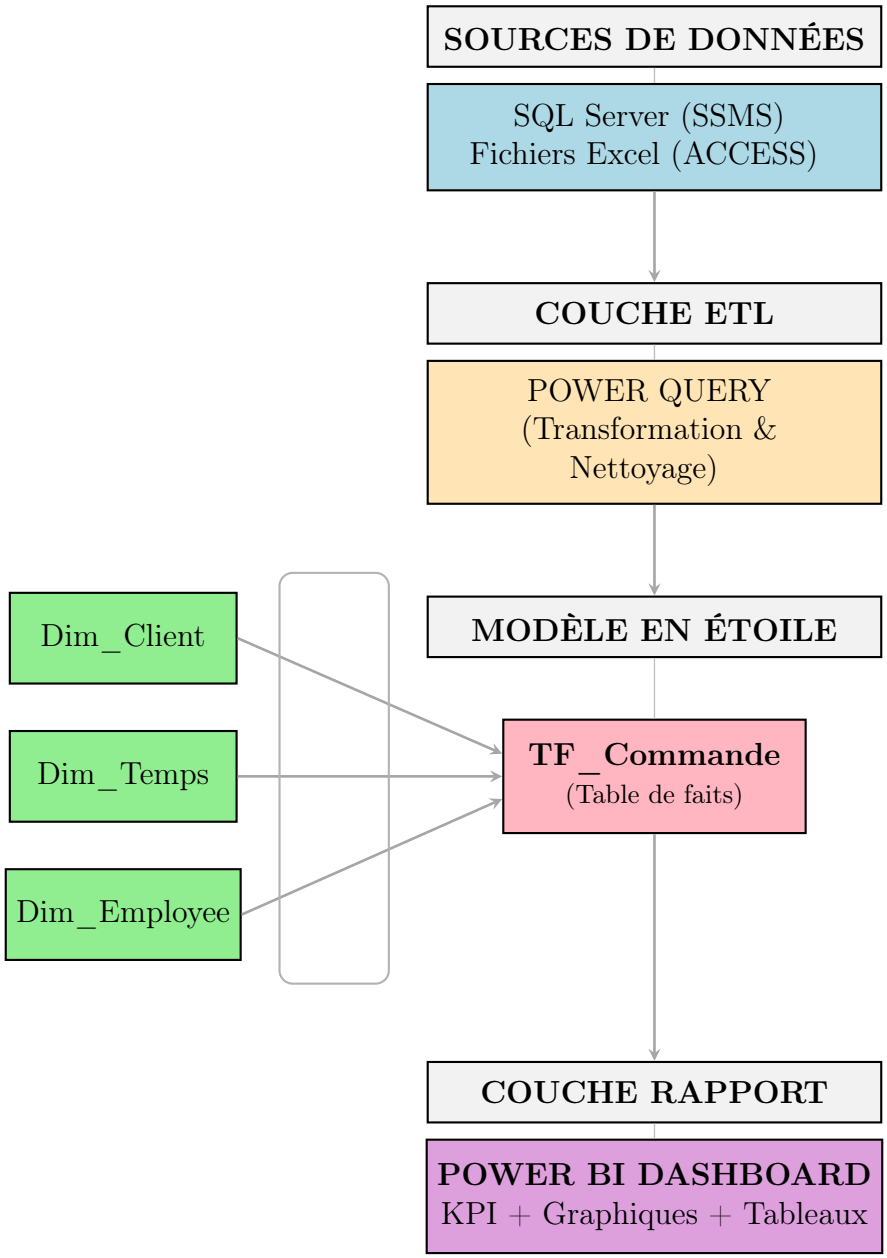

FIGURE 3 – Architecture complète de l'entrepôt de données Northwind avec modèle en étoile à 3 dimensions

3.2 Composants de l'Architecture

Composant	Description
Sources de Données	SQL Server (Northwind) + Fichiers Excel
Couche ETL	Power Query pour l'extraction et transformation
Tables Dimensions	Dim_Employee, Dim_Client, Dim_Temps
Table de Faits	TF_Commande
Couche Présentation	Rapports Power BI + Dashboard

TABLE 2 – Composants de l'architecture de l'entrepôt

3.3 Flux de Données

1. **Extraction** : Données brutes depuis SSMS et Excel
2. **Transformation** : Nettoyage et standardisation via Power Query
3. **Chargement** : Création des dimensions et table de faits
4. **Modélisation** : Établissement des relations 1 :*
5. **Visualisation** : Création des rapports et dashboards

4 Création du Projet Northwind

4.1 Structure du Projet

1. **Créer un dossier projet** :

```
C:\Users\IHADDADENE Chakib\Documents\Power BI Projects\Northwind_DW
              data\          # Donn es sources
              scripts\       # Codes Power Query
              exports\       # Exports et rapports

```

2. **Lancer Power BI Desktop**
3. **Nouveau fichier** : Fichier → Nouveau
4. **Enregistrer** : Fichier → Enregistrer sous → "Northwind_DW.pbix"

4.2 Importation des Données Sources

4.2.1 Connexion à SQL Server

1. Cliquez sur **"Obtenir des données"** → **Base de données** → **SQL Server**
2. Entrez les informations de connexion :
   — Serveur : `DESKTOP-VEO1CEQ\SQLCHAKIB`

— Base de données : `Northwind`

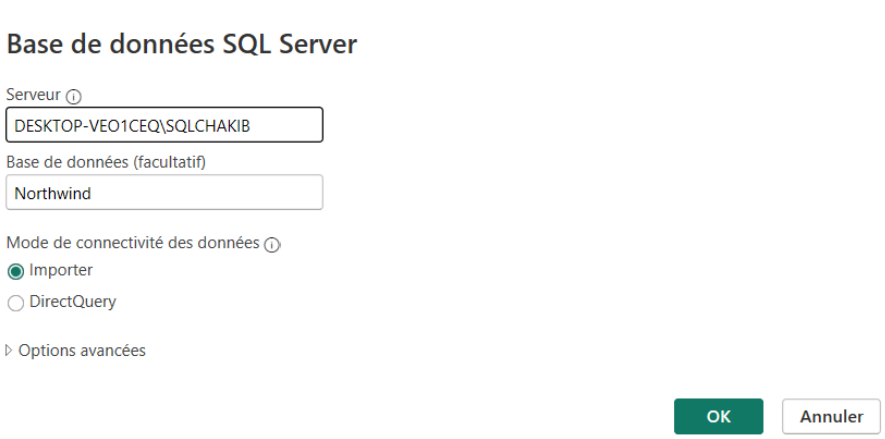

FIGURE 4 – Connexion a la base de données SSMS

— Mode de connexion : **Importer** (recommandé)

3. Sélectionnez les tables nécessaires :
   — Customers
   — Employees
   — EmployeeTerritories
   — Orders
   — Territories

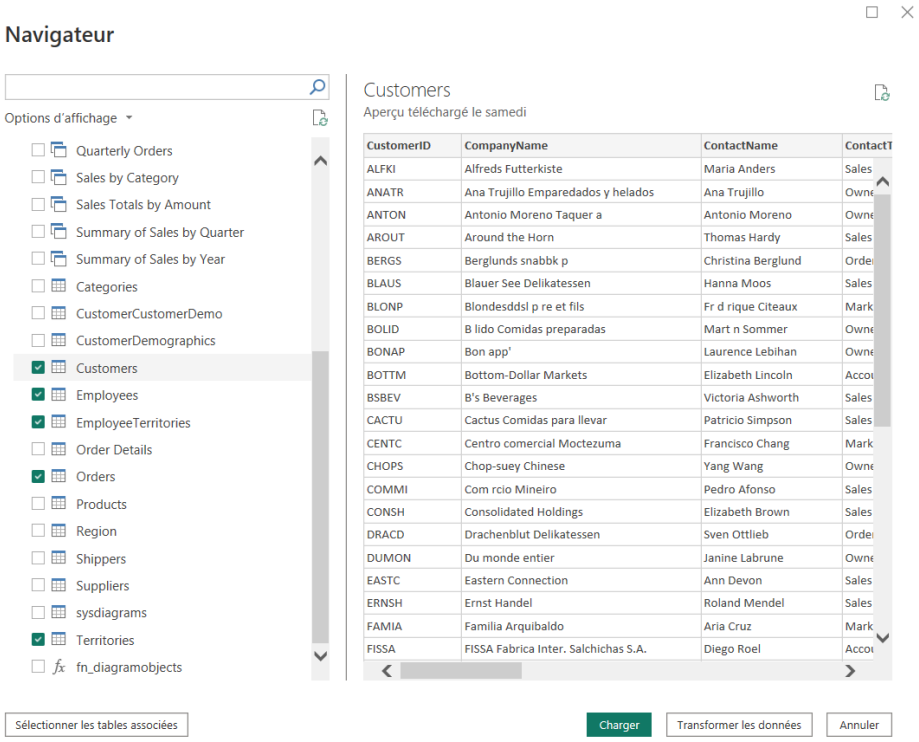

FIGURE 5 – Selection des tables nécessaires

4.2.2 Importation des Fichiers Excel

1. Cliquez sur **"Obtenir des données"** → **Fichier** → **Excel**
2. Naviguez vers : `C:\Users\PC\Documents\M2 BIGDATA\tp bi\csv du dm`
3. Sélectionnez les fichiers :
   - Customers.xlsx
   - Employees2.xlsx
   - Orders.xlsx

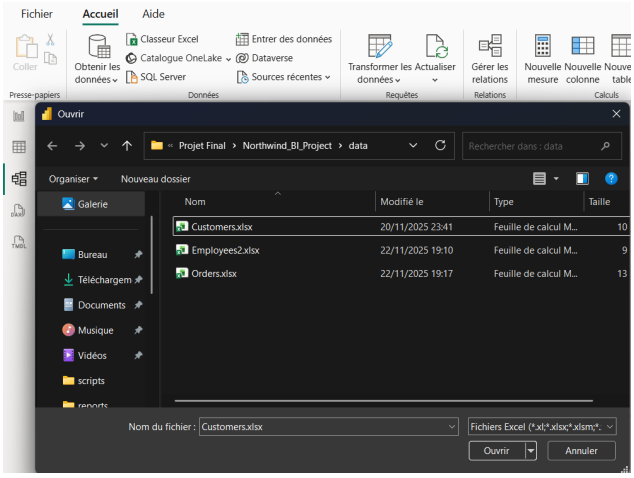

FIGURE 6 – Importation des données excel

4.3 Transformation avec Power Query

4.3.1 Accès à l'Éditeur Power Query

— Méthode 1 : **Accueil → Transformer les données**

— Méthode 2 : Clic droit sur une requête → **Éditer la requête**

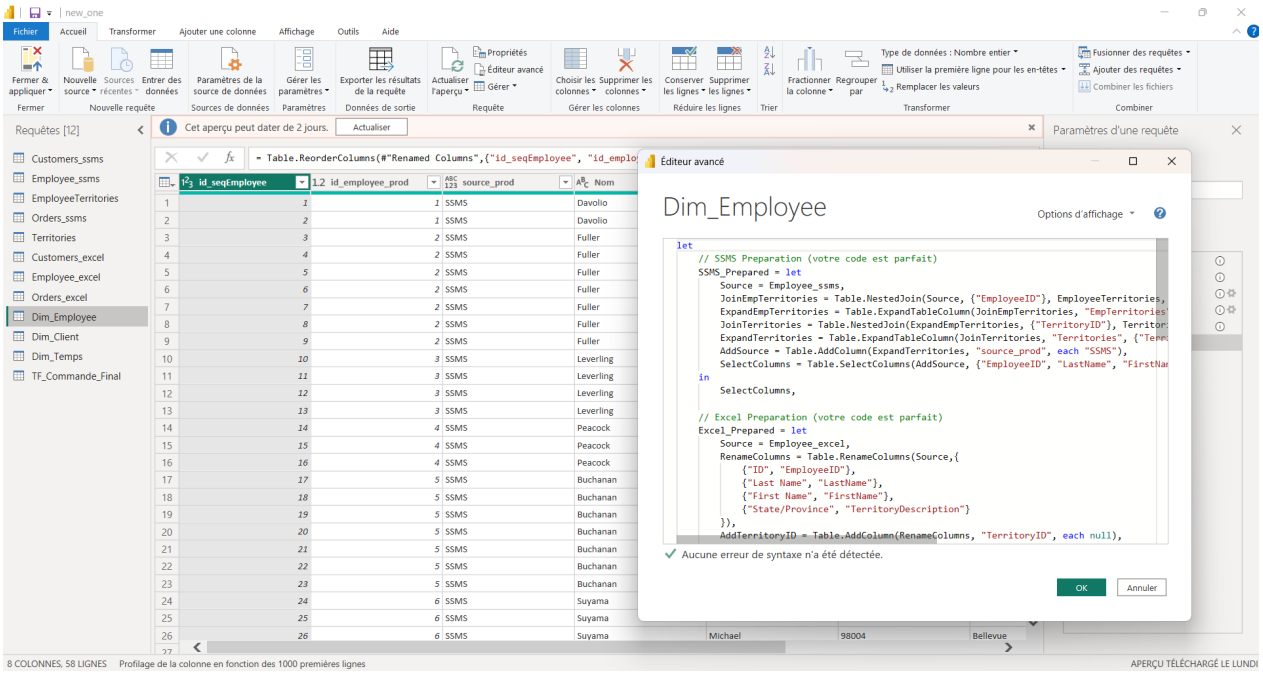

FIGURE 7 – Éditeur Power Query avec requêtes Northwind

4.3.2 Renommage des Requêtes

1. Dans le volet de navigation Power Query :
2. Renommez chaque requête avec le suffixe `_ssms` ou `_excel` :
   — Customers → Customers_ssms
   — Employees → Employee_ssms
   — etc.

5 Implémentation du Modèle de Données

5.1 Création des Dimensions

5.1.1 Dim_Employee

1. **Nouvelle requête** : Accueil → Nouvelle source → Requête vide
2. **Coller le code** : Collez le code M de Dim_Employee
3. **Renommer** : Renommez la requête en `Dim_Employee`
4. **Fermer et appliquer**

5.1.2 Dim_Client et Dim_Temps

— Répétez le processus pour chaque dimension
— Vérifiez les types de données
— Activez le chargement pour les tables dimensionnelles

5.2 Création de la Table de Faits

5.2.1 TF_Commande

1. Créez une nouvelle requête vide
2. Collez le code complet de TF_Commande
3. Renommez en `TF_Commande`
4. Vérifiez le nombre de lignes (devrait être 878)

5.3 Établissement des Relations

1. Retournez dans la vue **Modèle** de Power BI
2. Établissez les relations suivantes :
   — `TF_Commande[id_temps] → Dim_Temps[id_temps]`
   — `TF_Commande[id_seqEmployee] → Dim_Employee[id_seqEmployee]`
   — `TF_Commande[id_seqClient] → Dim_Client[id_seqClient]`
3. Vérifiez que toutes les relations sont **1 à plusieurs (*)**

FIGURE 8 – Modèle de données final avec relations

6 Comparaison Power BI vs Talend pour l'ETL

6.1 Tableau Comparatif

Aspect	**Power BI (Power Query)**	**Talend**
Type d'outil	Outil de BI avec ETL intégré	Plateforme ETL/ELT dédiée
Complexité	Courbe d'apprentissage douce	Courbe d'apprentissage plus raide
Langage	M (Power Query)	Java + Composants visuels
Connexions	Connecteurs natifs limités	1000+ connecteurs
Transformation	Interface utilisateur + M	Interface graphique + code
Orchestration	Basic (rafraîchissements)	Avancée (workflows complexes)
Coût	Gratuit (Desktop)	Licence payante (Enterprise)
Performance	Optimisé pour données de taille moyenne	Scalable pour gros volumes
Maintenance	Facile (tout intégré)	Complexe (infrastructure séparée)
Utilisation projet Northwind	Parfait pour POC et démonstrations	Surdimensionné pour ce besoin

TABLE 3 – Comparaison Power BI vs Talend pour l'ETL

6.2 Points Forts par Outil

6.2.1 Power BI (Power Query)

— Intégration native avec les rapports
— Interface utilisateur intuitive
— Pas de contexte switching entre ETL et reporting
— Version Desktop gratuite
— Fonctionnalités de modélisation intégrées
— Support DAX pour calculs avancés

6.2.2 Talend

— Orchestration de workflows complexes
— Support de gros volumes de données
— Connecteurs très variés
— Qualité et gouvernance des données
— Métadonnées et documentation
— Intégration avec l'écosystème Big Data

6.3 Exemple de Code Comparé

Power Query (M)

```
let
    Source = Orders_ssms,
    #"Filtered Rows" =
        Table.SelectRows(
            Source,
            each [ShippedDate] <>
    null
        )
in
    #"Filtered Rows"
```

Talend (Java)

```
// Composant tFilterRow
row1.ShippedDate != null

// Ou en Java dans tJavaRow
if(input_row.ShippedDate != null)
    {
    output_row = input_row;
}
```

TABLE 4 – Comparaison de syntaxe pour un filtre simple

6.4 Recommandation pour le Projet Northwind

Choix Optimal pour Northwind

Power BI avec Power Query est le meilleur choix pour ce projet car :

- Volume de données modeste (878 lignes)
- Besoin d'intégration directe avec les rapports
- Temps de développement réduit
- Aucune infrastructure supplémentaire nécessaire
- Coût nul avec Power BI Desktop gratuit
- Facilité de maintenance et de partage

7 Expérience Pratique : Talend vs Power BI pour le projet Northwind

7.1 Contexte d'Implémentation

Dans le cadre de ce projet, j'ai implémenté la même solution d'entrepôt de données Northwind avec deux outils différents : Power BI et Talend. L'objectif était de remplir un entrepôt de données vide sur SQL Server Management Studio (SSMS) à partir de deux sources distinctes :

- **Source 1** : Base de données Northwind existante sur SSMS
- **Source 2** : Données complémentaires au format Microsoft Access
- **Cible** : Entrepôt SSMS vide avec modèle en étoile

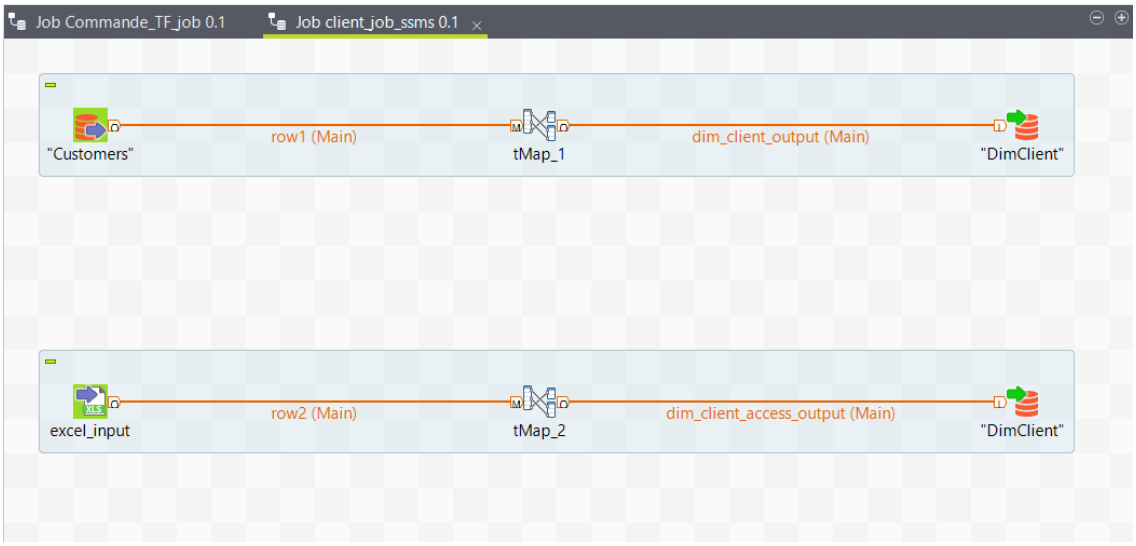

FIGURE 9 – Job Talend complet pour la dimension client

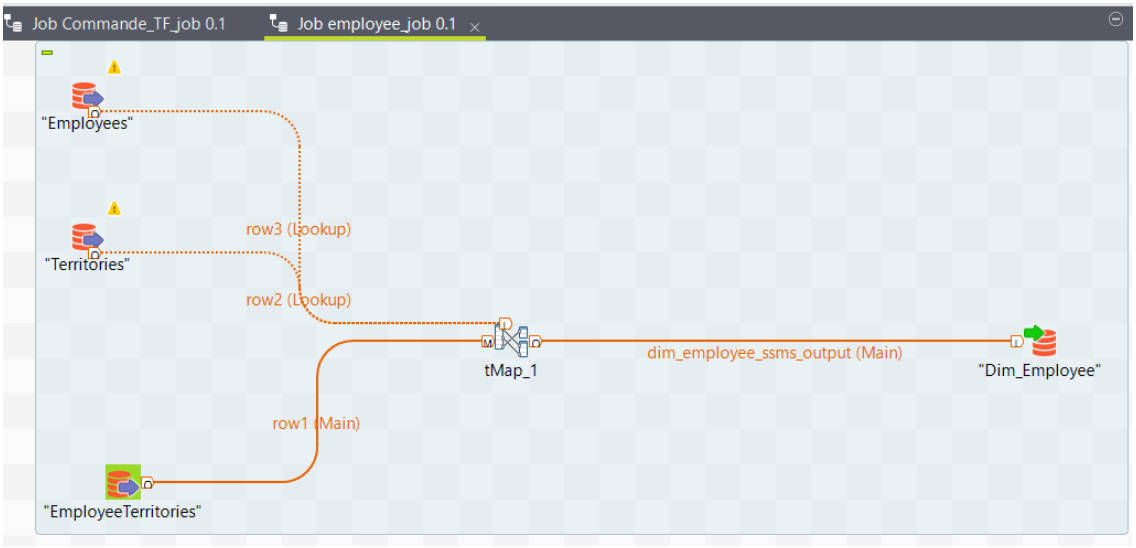

FIGURE 10 – Job Talend complet pour la dimension employee

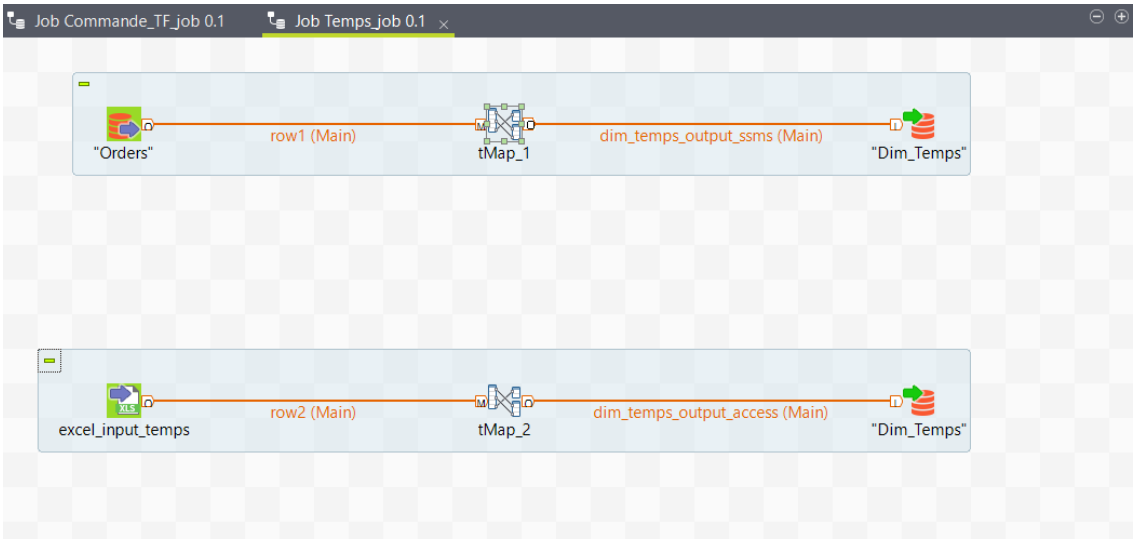

FIGURE 11 – Job Talend complet pour la dimension temps

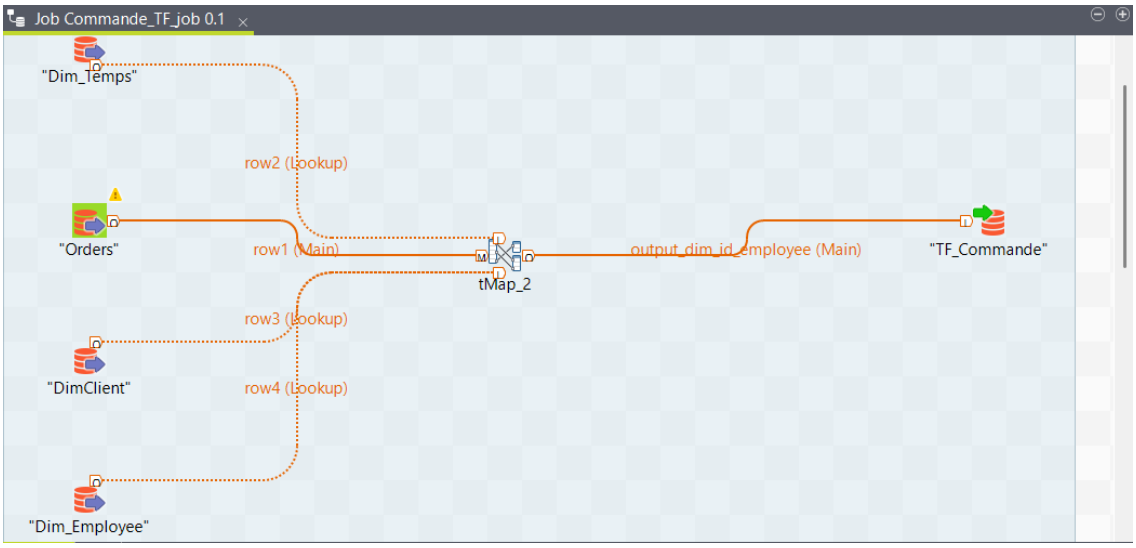

FIGURE 12 – Job Talend complet pour la table de fait

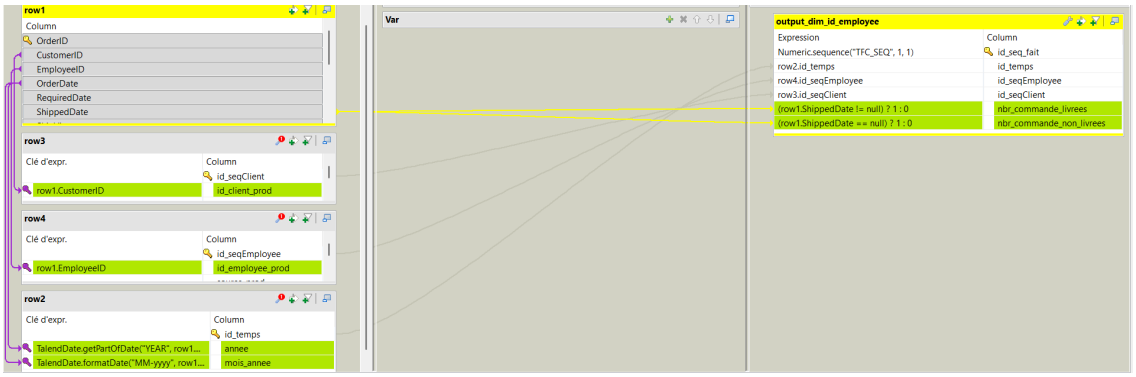

FIGURE 13 – Jointure de creation de la table de fait

7.2 Implémentation avec Talend

7.2.1 Architecture du Job Talend

Le job Talend a été structuré comme suit :

1. **Extraction des données sources** :
   - Connexion à SQL Server via `tMSSqlInput`
   - Lecture des fichiers Access via `tFileInputDelimited`
2. **Transformations principales** :
   - `tMap_2` : Jointures et transformations complexes
   - `tMap` : Nettoyage et standardisation
   - `tJavaRow` : Calculs personnalisés en Java
3. **Chargement vers SSMS** :
   - `tMSSqlOutput` pour chaque table dimensionnelle

— Gestion des clés primaires et étrangères

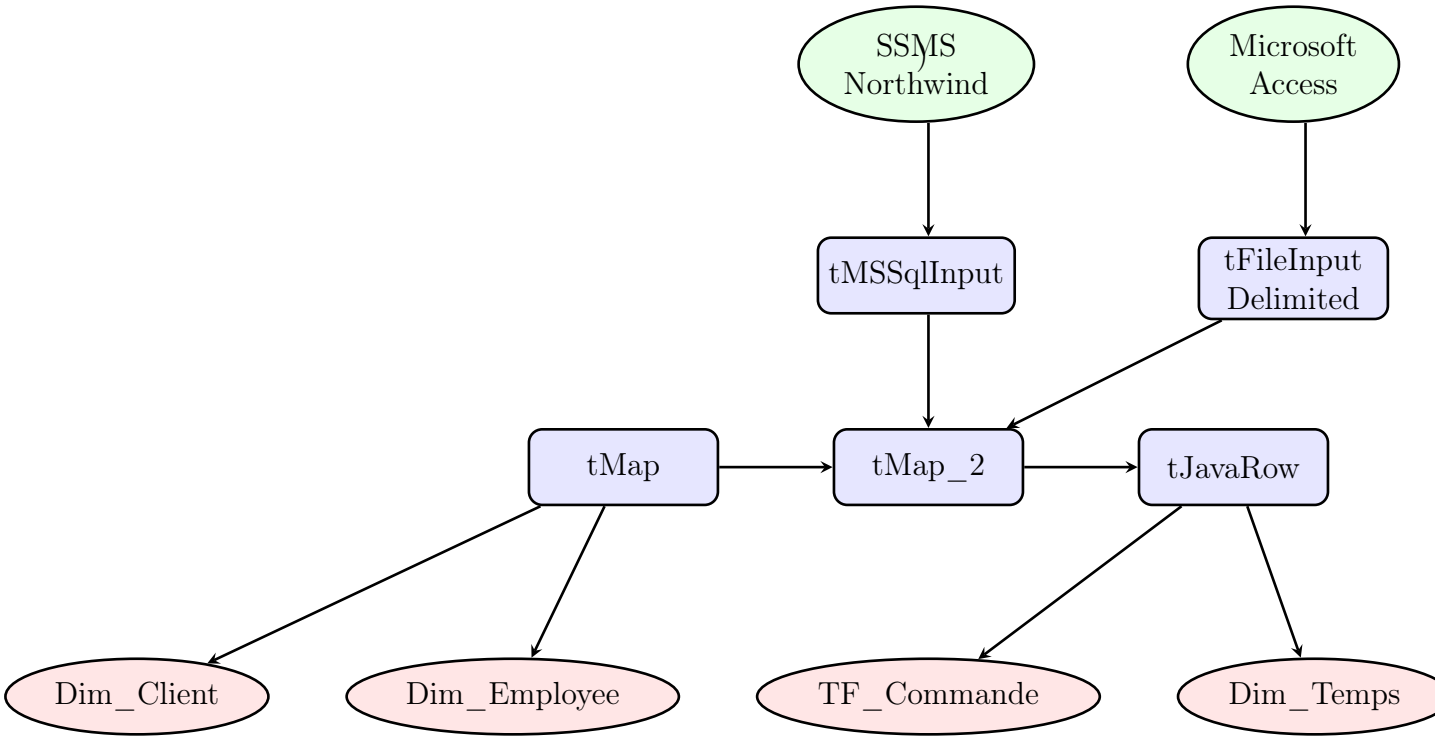

FIGURE 14 – Architecture simplifiée du flux de données Talend

7.3 Comparaison Pratique des Résultats

7.3.1 Volumes de Données Traités

Table	**Lignes Sources**	**Talend**	**Power BI**
Dim_Temps	878 dates	878	878
Dim_Client	120 clients	120	120
Dim_Employee	58 employés	58	58
TF_Commande	878 commandes	878	878

TABLE 5 – Volumes identiques traités par les deux outils

7.3.2 Métriques Comparatives

Métrique	Talend	Power BI
Temps de développement	8-10 heures	2-3 heures
Temps d'exécution	15-20 secondes	8-12 secondes
Complexité	Élevée (Java + XML)	Moyenne (M)
Apprentissage requis	2-3 semaines	2-3 jours
Maintenance	Infrastructure lourde	Fichier unique
Déploiement	Serveur requis	Fichier .pbix

TABLE 6 – Comparaison pratique des deux approches

7.4 Leçons Apprises et Recommandations

7.4.1 Pour les POC et Démonstrations

Power BI Recommandé

- **Vitesse** : Développement 3x plus rapide
- **Simplicité** : Pas d'infrastructure supplémentaire
- **Intégration** : ETL + modélisation + visualisation intégrés
- **Collaboration** : Partage facile avec Power BI Service

7.4.2 Pour les Environnements de Production

Talend Recommandé

- **Robustesse** : Gestion avancée des erreurs
- **Monitoring** : Suivi détaillé des exécutions
- **Scalabilité** : Traitement des gros volumes
- **Orchestration** : Workflows complexes et planifiés

7.5 Conclusion de l'Expérience

Cette expérience pratique démontre que le choix entre Talend et Power BI dépend du contexte :

- **Power BI** excelle pour la rapidité, la simplicité et l'intégration BI complète
- **Talend** est supérieur pour la robustesse, le monitoring et les environnements de production
- Pour le projet Northwind (1,456 lignes), Power BI était suffisant et plus rapide
- Pour des volumes plus importants ou des besoins de production, Talend serait nécessaire

Recommandation Finale

Utilisez Power BI pour : POC, analyses ad-hoc, petites volumétries, équipes métier
Utilisez Talend pour : Pipelines de production, gros volumes, environnements réglementés, équipes techniques

8 Visualisation avec Python dans Power BI

8.1 Configuration de Python pour Power BI

8.1.1 Installation des Prérequis

1. **Installer Python** : Téléchargez Python 3.8+ depuis `https://python.org`
2. **Installer les packages** :

```
pip install pandas matplotlib numpy seaborn

```

3. **Configurer Power BI** :
   - Fichier → Options et paramètres → Options
   - Scripting Python → Spécifier le chemin d'installation Python
   - Exemple : `C:\Users\IHADDADENE Chakib\AppData\Local\Programs\Python\Python39`

8.1.2 Création d'un Visuel Python

1. **Sélectionner le visuel Python** :
   - Dans le volet Visualisations, cliquez sur l'icône **Python**
   - Une zone de script vide apparaît sur la page

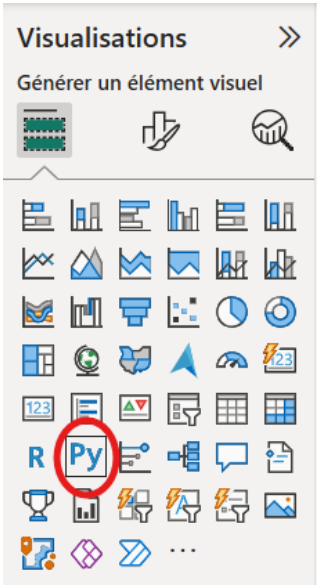

FIGURE 15 – Selection de l'icone PY

2. **Glisser les champs nécessaires** :
   - Depuis le volet Données, glissez-déposez les colonnes dans la partie Valeurs
   - Exemple pour notre projet Northwind :
     - Depuis `TF_Commande` : `nbr_commande_livrees`
     - Depuis `Dim_Employee` : `TerritoryDescri`

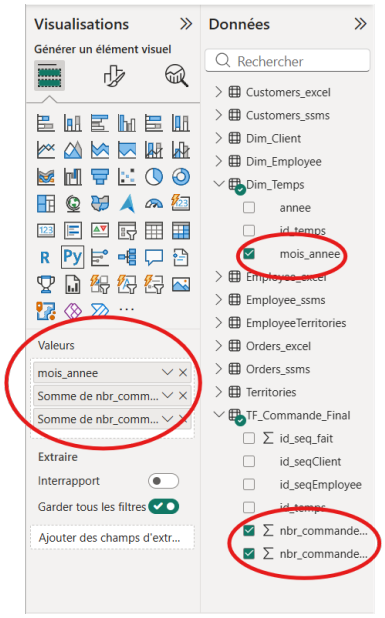

FIGURE 16 – Glissement des champs nécessaires

3. **Écrire le script Python** :
   - Le champ de script s'active automatiquement
   - Power BI crée automatiquement un DataFrame `dataset` contenant vos données

FIGURE 17 – Interface du visuel Python dans Power BI

8.2 Les 6 Graphiques Python Implémentés

8.2.1 Graphique 1 : Volume des Commandes par Mois

```
import pandas as pd
import matplotlib.pyplot as plt

# V rification des colonnes
colonnes_requises = ['id_temps', 'nbr_commande_livrees', '
    nbr_commande_non_livrees', 'mois_annee']
colonnes_manquantes = [col for col in colonnes_requises if col not in
    dataset.columns]

if colonnes_manquantes:
    fig, ax = plt.subplots(figsize=(10, 2))
    ax.text(0.5, 0.5, f"Glissez ces colonnes:\n{', '.join(
    colonnes_manquantes)}",
            ha='center', va='center', fontsize=12, color='red',
            bbox=dict(boxstyle="round,pad=0.5", facecolor="yellow", alpha
    =0.7))
    ax.axis('off')
    plt.show()

else:
    # Agr gation par mois
    commandes_par_mois = dataset.groupby('mois_annee').agg({
        'nbr_commande_livrees': 'sum',
        'nbr_commande_non_livrees': 'sum'
    }).reset_index()

    commandes_par_mois['total_commandes'] = (
        commandes_par_mois['nbr_commande_livrees'] +
        commandes_par_mois['nbr_commande_non_livrees']
    )

    # Trier par date
    try:
        commandes_par_mois['date_sort'] = pd.to_datetime(
            commandes_par_mois['mois_annee'] + '/01',
            format='%m/%Y/%d'
        )
        commandes_par_mois = commandes_par_mois.sort_values('date_sort')
    except:
        commandes_par_mois = commandes_par_mois.sort_values('mois_annee')

    # Cr ation du graphique
    fig, ax = plt.subplots(figsize=(14, 7))

    x = range(len(commandes_par_mois))
    mois_labels = commandes_par_mois['mois_annee'].tolist()

    # Barres empil es
    ax.bar(x, commandes_par_mois['nbr_commande_livrees'],
           label='Commandes Livr es', color='green', alpha=0.7)
    ax.bar(x, commandes_par_mois['nbr_commande_non_livrees'],
           bottom=commandes_par_mois['nbr_commande_livrees'],
           label='Commandes Non Livr es', color='red', alpha=0.7)
```

```
    ax.set_xlabel('Mois', fontsize=12)
    ax.set_ylabel('Nombre de Commandes', fontsize=12)
    ax.set_title(' VOLUME DES COMMANDES PAR MOIS', fontsize=14, fontweight
    ='bold')
    ax.set_xticks(x)
    ax.set_xticklabels(mois_labels, rotation=45, ha='right')
    ax.legend()
    ax.grid(True, alpha=0.3)

    # Ajouter les totaux au-dessus des barres
    for i, total in enumerate(commandes_par_mois['total_commandes']):
        ax.text(i, total + (total*0.01), f'{total:,}',
                ha='center', va='bottom', fontsize=9)

    # Statistiques
    total_periode = commandes_par_mois['total_commandes'].sum()
    stats_text = f"Total p riode: {total_periode:,} commandes"

    plt.figtext(0.02, 0.02, stats_text, fontsize=10,
                bbox=dict(boxstyle="round,pad=0.5", facecolor="lightgray",
    alpha=0.8))

    plt.tight_layout()
    plt.show()
```

Listing 1 – Volume des commandes livrées et non livrées par mois

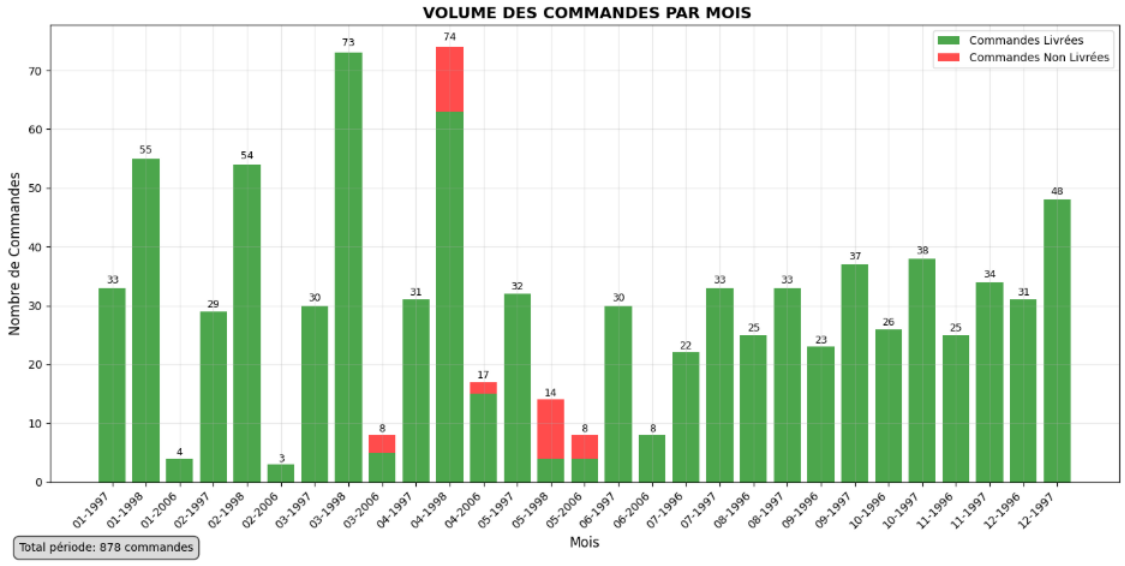

FIGURE 18 – Volume des commandes livrées et non livrées par mois

8.2.2 Graphique 2 : Top 10 Clients par Commandes Livrées

```
import pandas as pd
```

```
import matplotlib.pyplot as plt

# Grouper par client et sommer les commandes livr es
df = dataset.groupby('CompanyName')['nbr_commande_livrees'].sum().nlargest
    (10)

plt.figure(figsize=(10, 6))
df.sort_values().plot(kind='barh', color='skyblue')
plt.title("Top 10 Clients par commandes livr es", fontsize=14, fontweight
    ='bold')
plt.xlabel("Nombre de commandes livr es", fontsize=12)
plt.grid(axis='x', alpha=0.3)

# Ajouter les valeurs sur les barres
for i, v in enumerate(df.sort_values()):
    plt.text(v + (v*0.01), i, f'{v:,}', va='center', fontsize=10)

plt.tight_layout()
plt.show()
```

Listing 2 – Top 10 clients par nombre de commandes livrées

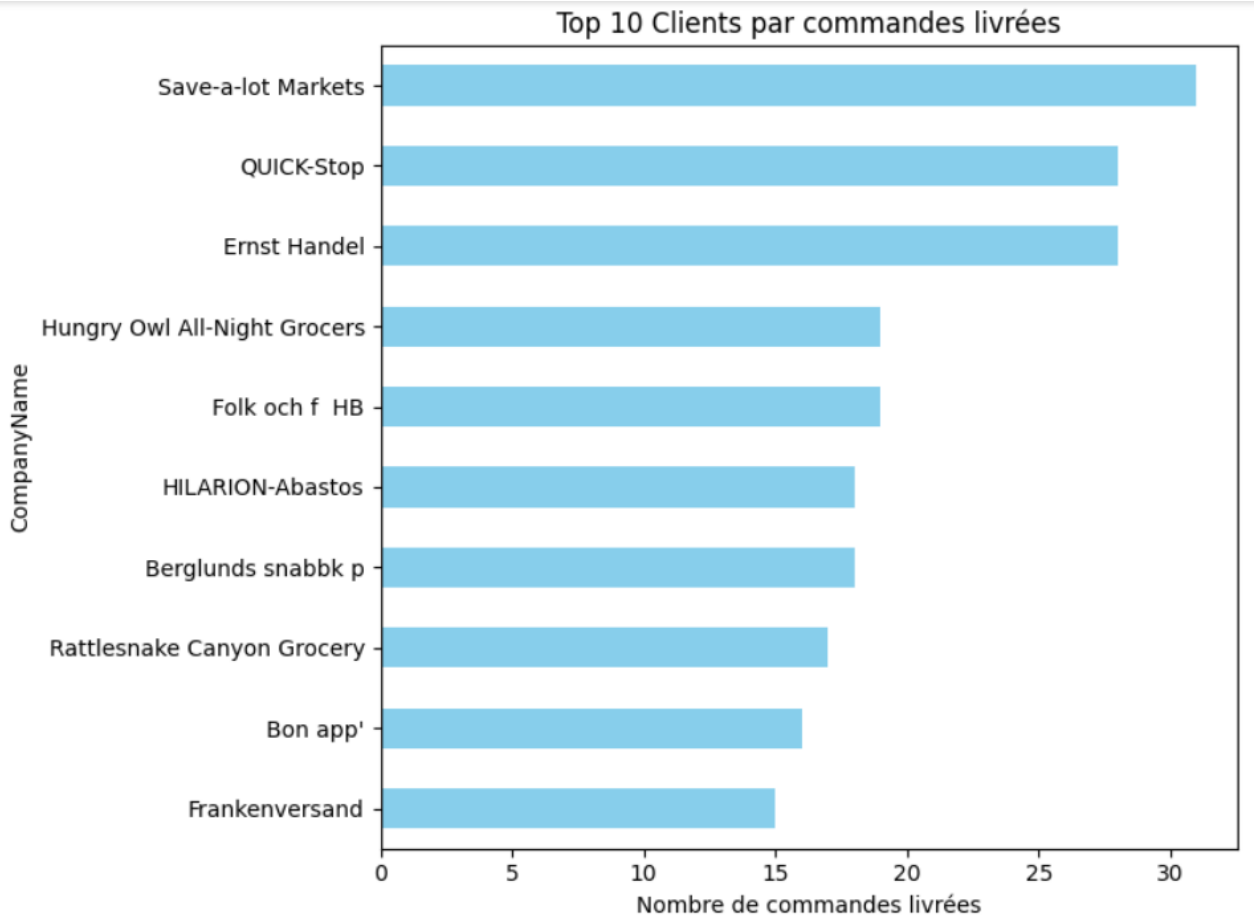

FIGURE 19 – Top 10 clients par nombre de commandes livrées

8.2.3 Graphique 3 : Top 5 Territoires par Commandes Livrées

```
import pandas as pd
import matplotlib.pyplot as plt

df = dataset.groupby('TerritoryDescri')['nbr_commande_livrees'].sum().
    nlargest(5)

plt.figure(figsize=(10, 6))
df.sort_values().plot(kind='barh', color='purple')
plt.title("Top 5 Territoires par commandes livr es", fontsize=14,
    fontweight='bold')
plt.xlabel("Nombre de commandes", fontsize=12)
plt.grid(axis='x', alpha=0.3)

# Ajouter les valeurs sur les barres
for i, v in enumerate(df.sort_values()):
    plt.text(v + (v*0.01), i, f'{v:,}', va='center', fontsize=10, color='
    white')

```

```
plt.tight_layout()
plt.show()
```

Listing 3 – Top 5 territoires par performance

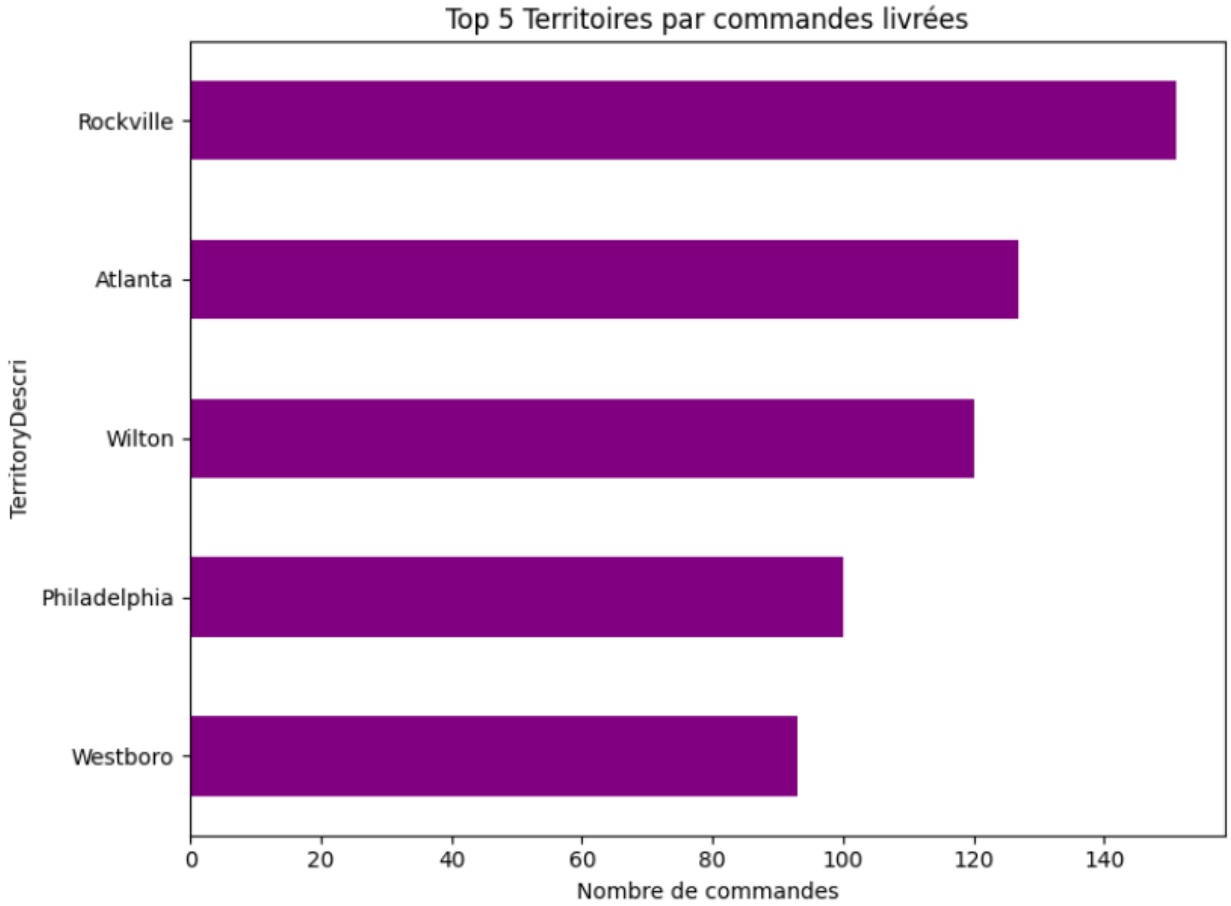

FIGURE 20 – Top 5 territoires par nombre de commandes livrées

8.2.4 Graphique 4 : Heatmap Clients vs Employés

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Cr er nom complet de l'employ
dataset['Employe'] = dataset['Nom'] + " " + dataset['Prenom']

# Calcul total commandes par client
total_client = dataset.groupby('CompanyName')['nbr_commande_livrees'].sum
    ()

# Top 20 clients
top_clients = total_client.nlargest(20).index
```

```
df_top = dataset[dataset['CompanyName'].isin(top_clients)]

# Pivot table : clients en lignes, employ s en colonnes
df_pivot = df_top.pivot_table(index='CompanyName', columns='Employe',
                              values='nbr_commande_livrees', aggfunc='sum'
    , fill_value=0)

plt.figure(figsize=(14, 8))
plt.imshow(df_pivot, cmap='YlGnBu', aspect='auto', interpolation='nearest'
    )
plt.colorbar(label='Commandes livr es', shrink=0.8)
plt.xticks(range(len(df_pivot.columns)), df_pivot.columns, rotation=90,
    fontsize=9)
plt.yticks(range(len(df_pivot.index)), df_pivot.index, fontsize=9)
plt.title("Heatmap : Top 20 Clients x Employ s", fontsize=14, fontweight=
    'bold')

# Ajouter les valeurs dans les cellules
for i in range(len(df_pivot.index)):
    for j in range(len(df_pivot.columns)):
        value = df_pivot.iloc[i, j]
        if value > 0:
            plt.text(j, i, f'{int(value)}', ha='center', va='center',
                     color='black' if value < df_pivot.values.max()/2 else
    'white',
                     fontsize=8)

plt.tight_layout()
plt.show()
```

Listing 4 – Heatmap des commandes livrées par client et employé

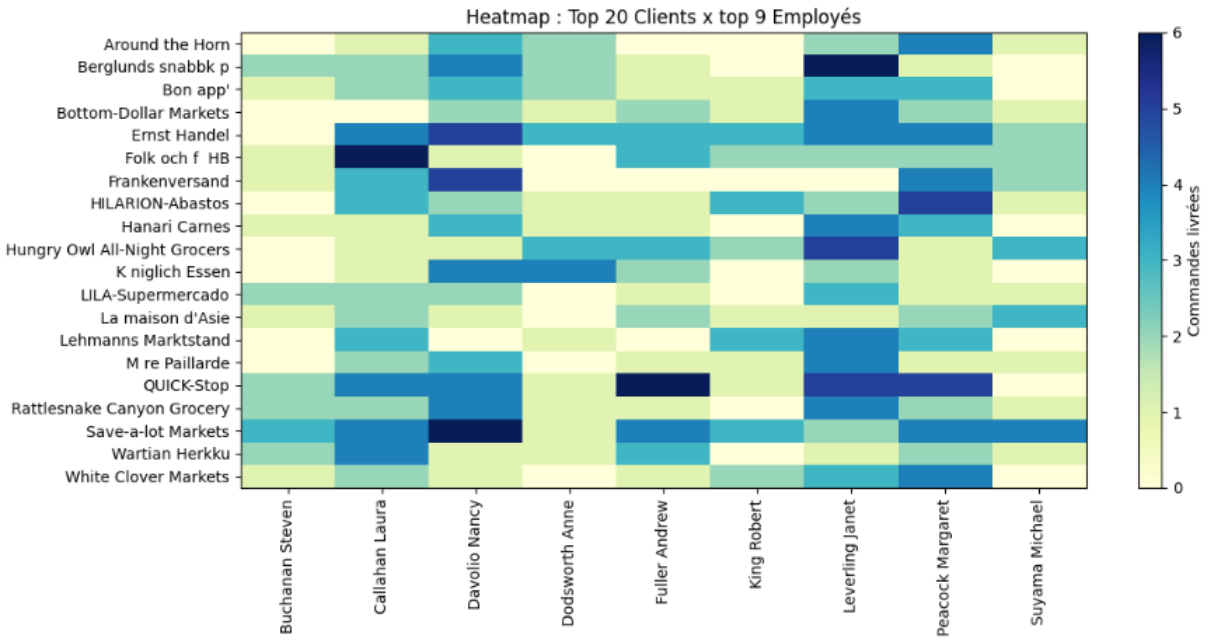

FIGURE 21 – Heatmap des commandes livrées par client et employé

8.2.5 Graphique 5 : Répartition par Région

```
import pandas as pd
import matplotlib.pyplot as plt

df = dataset.groupby('Region')['nbr_commande_livrees'].sum()

plt.figure(figsize=(9, 9))
wedges, texts, autotexts = plt.pie(df.values, labels=df.index, autopct='
    %1.1f%%',
                                   colors=plt.cm.Set3.colors, startangle
    =90,
                                   wedgeprops=dict(edgecolor='white',
    linewidth=2))

# Am liorer l'apparence des pourcentages
for autotext in autotexts:
    autotext.set_color('black')
    autotext.set_fontsize(10)
    autotext.set_fontweight('bold')

plt.title("R partition des commandes livr es par r gion", fontsize=14,
    fontweight='bold')

# Ajouter une l gende
plt.legend(wedges, df.index, title="R gions", loc="center left",
           bbox_to_anchor=(1, 0, 0.5, 1), fontsize=10)

plt.tight_layout()
plt.show()
```

Listing 5 – Répartition des commandes livrées par région

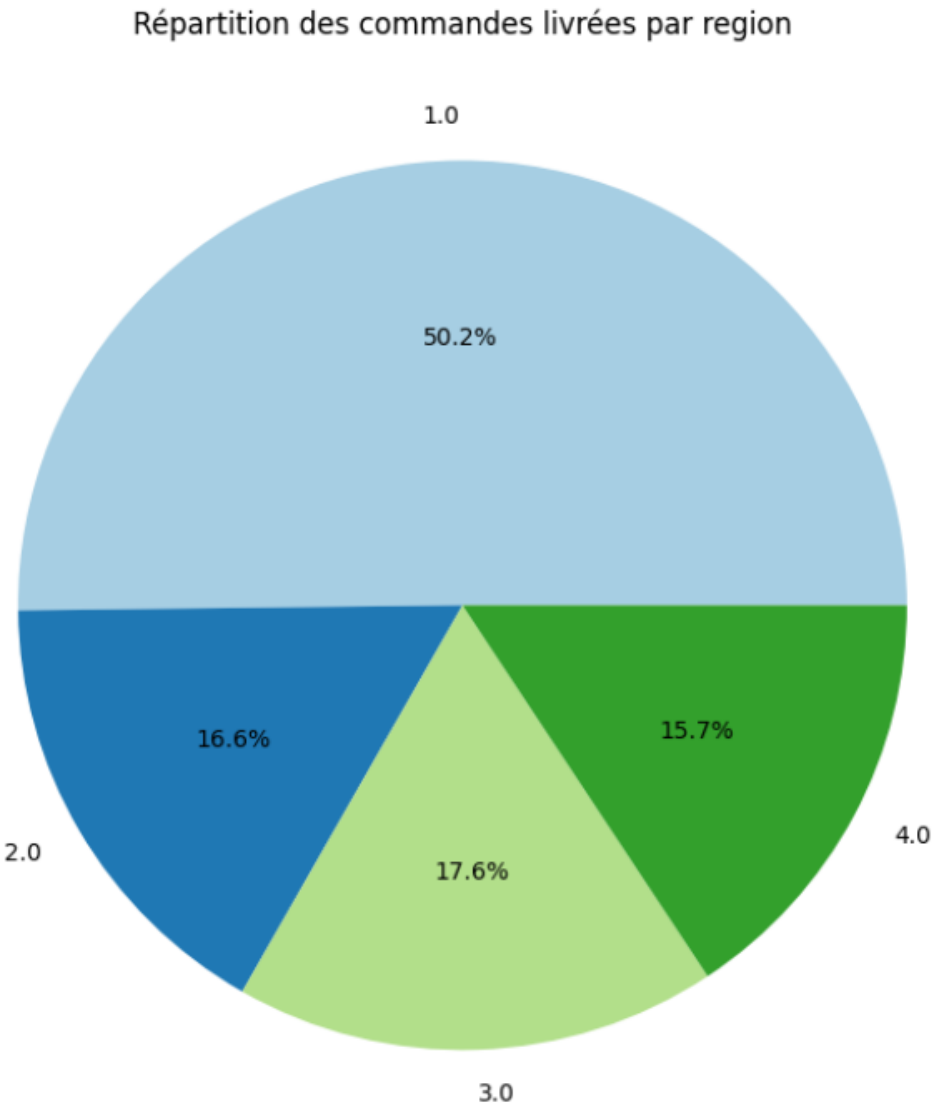

FIGURE 22 – Répartition des commandes livrées par région

8.2.6 Graphique 6 : Évolution Mensuelle des Commandes

```
import pandas as pd
import matplotlib.pyplot as plt

# Calculer le total des commandes par mois
df = dataset.groupby('mois_annee')[['nbr_commande_livrees','
    nbr_commande_non_livrees']].sum()
df = df.sort_index()  # trier par mois

plt.figure(figsize=(12, 7))
```

```
plt.plot(df.index, df['nbr_commande_livrees'], marker='o', linestyle='-',
         color='green', label='Livr es', linewidth=2, markersize=8)
plt.plot(df.index, df['nbr_commande_non_livrees'], marker='s', linestyle='
    --',
         color='red', label='Non-livr es', linewidth=2, markersize=8)

plt.title(" volution  mensuelle des commandes livr es et non-livr es",
          fontsize=14, fontweight='bold')
plt.xlabel("Mois", fontsize=12)
plt.ylabel("Nombre de commandes", fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.legend(fontsize=12)
plt.grid(True, alpha=0.3)

# Remplir l'aire entre les courbes
plt.fill_between(df.index, df['nbr_commande_livrees'], alpha=0.2, color='
    green')
plt.fill_between(df.index, df['nbr_commande_non_livrees'], alpha=0.2,
    color='red')

# Ajouter les valeurs sur les points
for i, (liv, nliv) in enumerate(zip(df['nbr_commande_livrees'], df['
    nbr_commande_non_livrees'])):
    plt.text(i, liv + (liv*0.05), f'{liv:,}', ha='center', va='bottom',
    fontsize=9)
    plt.text(i, nliv - (nliv*0.05), f'{nliv:,}', ha='center', va='top',
    fontsize=9)

plt.tight_layout()
plt.show()
```

Listing 6 – Évolution des commandes livrées et non livrées

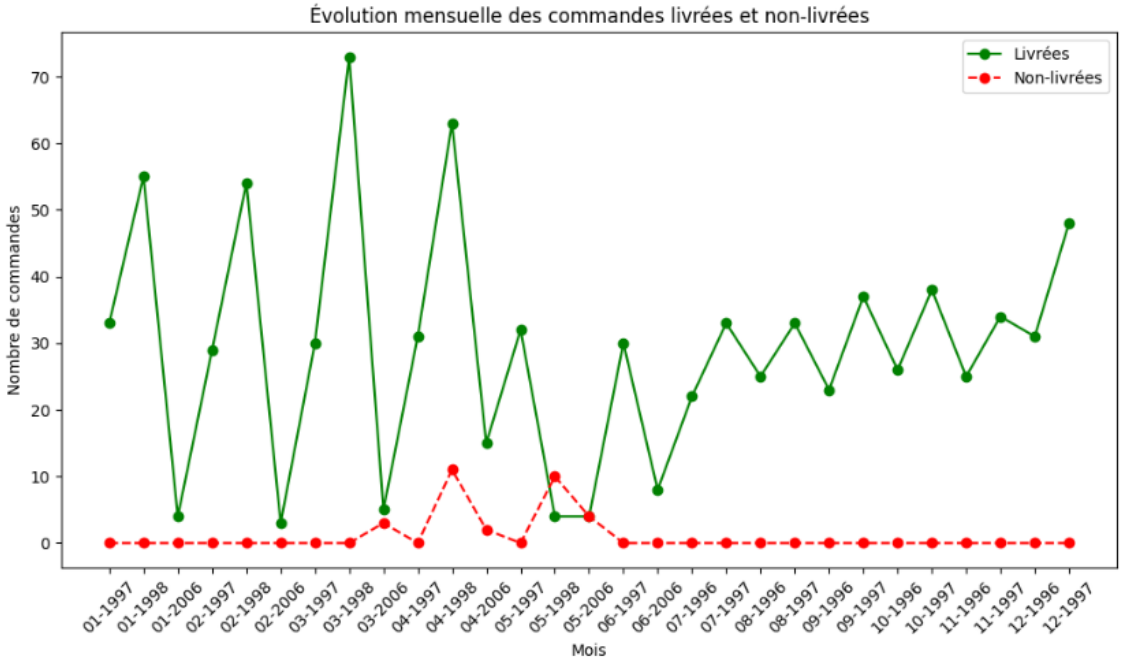

FIGURE 23 – Évolution mensuelle des commandes livrées et non livrées

8.3 Explications des Codes Python

8.3.1 Structure Commune des Scripts

Chaque script Python suit la même structure logique :

1. **Importation des bibliothèques** : pandas pour la manipulation des données, matplotlib pour la visualisation
2. **Vérification des données** : Validation des colonnes nécessaires dans le DataFrame
3. **Agrégation des données** : Groupement et calcul des métriques
4. **Création du graphique** : Configuration du type, des couleurs et des labels
5. **Personnalisation** : Ajout de valeurs, de grilles et de légendes
6. **Affichage** : Utilisation de `plt.show()` pour afficher dans Power BI

8.3.2 Techniques Avancées Utilisées

Technique	Application dans les scripts
Groupby + Agg	Agrégation des données par différentes dimensions (mois, client, territoire)
Pivot Table	Création de heatmaps avec des matrices clients × employés
Data Validation	Vérification des colonnes nécessaires avant le traitement
Visual Customization	Personnalisation fine des couleurs, tailles et polices
Annotations	Ajout de valeurs numériques directement sur les graphiques
Error Handling	Gestion des erreurs de format de date avec try-except
Color Mapping	Utilisation de palettes de couleurs professionnelles (Set3, YlGnBu)

TABLE 7 – Techniques Python avancées utilisées dans les visualisations

8.4 Bonnes Pratiques pour les Visuals Python

Pratique	Explication
Vérification des données	Vérifier les colonnes nécessaires avant le traitement
Gestion des erreurs	Utiliser try-except pour les conversions de date
Optimisation des performances	Limiter le nombre de lignes avec des filtres
Personnalisation visuelle	Utiliser des palettes de couleurs professionnelles
Annotations	Ajouter des valeurs pour améliorer la lisibilité
Titres descriptifs	Utiliser des titres clairs et informatifs
Grilles légères	Ajouter des grilles pour faciliter la lecture
Export des scripts	Sauvegarder les scripts dans des fichiers .py externes

TABLE 8 – Bonnes pratiques pour les visuals Python dans Power BI

8.5 Avantages du Visuel Python

- **Flexibilité totale** : Accès à toutes les bibliothèques Python (pandas, matplotlib, seaborn)
- **Personnalisation avancée** : Contrôle complet sur chaque élément du graphique
- **Interactivité** : Réaction aux filtres Power BI en temps réel
- **Intégration native** : Pas besoin d'exporter/importer les données
- **Reproducibilité** : Scripts reproductibles et versionnables
- **Analyses complexes** : Possibilité d'implémenter des analyses statistiques avancées

Limitations à connaître

- Performance : Les scripts complexes peuvent ralentir Power BI
- Mémoire : Limité par les ressources de Power BI Desktop
- Dépendances : Packages à installer sur chaque machine
- Version Python : Doit correspondre à celle configurée dans Power BI
- Courbe d'apprentissage : Requiert des compétences en Python
- Debugging : Plus difficile que les visuels natifs de Power BI

8.6 Dashboard Final Northwind

Notre dashboard final intègre 6 visuals Python interconnectés, permettant une analyse complète des données Northwind :

FIGURE 24 – Dashboard Northwind avec 6 visuals Python intégrés

9 Résolution des Problèmes Courants

9.1 Problèmes d'Installation

Problème	**Solution**
Échec d'installation Microsoft Store	Téléchargez la version hors Store
Erreur "Accès refusé"	Exécutez en tant qu'administrateur
Manque de dépendances	Installez .NET Framework 4.8
Antivirus bloque l'installation	Désactivez temporairement l'antivirus

9.2 Problèmes de Connexion

Problème	**Solution**
SQL Server inaccessible	Vérifiez le nom du serveur et les permissions
Fichier Excel non trouvé	Vérifiez le chemin et les permissions
Erreur d'authentification	Utilisez l'authentification Windows
Limite de mémoire	Augmentez la mémoire allouée à Power BI

9.3 Problèmes avec Python

Problème	**Solution**
Python non détecté	Vérifiez le chemin dans Options → Scripting Python
Packages manquants	Exécutez `pip install pandas matplotlib`
Erreurs d'import	Redémarrez Power BI après installation des packages
Graphiques vides	Vérifiez que les champs nécessaires sont glissés
Performance lente	Limitez le nombre de lignes avec des filtres

Table 9 – Problèmes courants avec Python dans Power BI

10 Conclusion

Ce guide vous a accompagné de l'installation de Power BI Desktop à la création d'un projet d'entrepôt de données complet avec Northwind, incluant une analyse comparative avec Talend et deux approches Python. Vous disposez maintenant de toutes les ressources nécessaires pour :

- Installer et configurer Power BI Desktop
- Comprendre l'architecture d'un entrepôt de données en étoile
- Comparer Power BI et Talend pour l'ETL
- Créer un projet structuré Northwind
- Importer et transformer des données multi-sources
- Construire un modèle de données optimisé
- Utiliser Python dans Power BI pour des visualisations avancées
- Développer des dashboards Python autonomes sans Power BI
- Implémenter les 6 graphiques d'analyse Northwind
- Choisir l'approche adaptée à vos besoins
- Résoudre les problèmes courants
- Publier et partager vos analyses

Compétences Acquises

À travers ce projet, vous avez développé des compétences précieuses en :

- **ETL** : Transformation de données avec Power Query et Talend
- **Modélisation** : Conception de modèle en étoile
- **Visualisation** : Création de dashboards interactifs
- **Programmation** : Scripting Python avancé
- **Analyse** : Interprétation de données business
- **Comparaison** : Évaluation d'outils BI/ETL

Perspectives futures : Cette base solide vous permettra d'aborder des projets plus complexes, d'intégrer des sources de données supplémentaires, et d'explorer des techniques avancées comme le machine learning intégré ou le streaming de données en temps réel.