LO Adja Maguette
IAM2DS
Nexa Digital School

Examen Bloc 1 : réponses aux questions

1. Reformuler les besoins métiers et techniques de KNAUF Industries

- Contexte et problématique :

KNAUF Industries, acteur du transport B2B, fait face à une fragmentation de ses données opérationnelles (dispatchées sur différents formats et systèmes). Cette situation entraîne des incohérences, des données manquantes et une lenteur d'analyse qui freine la prise de décision.

- Objectif :

L'objectif est de centraliser et de fiabiliser les données pour permettre, à terme, des analyses prédictives (IA) comme la prévision de charge ou la détection des anomalies.

Besoins métier (logistique et pilotage)	Besoins techniques (data et infra)
Pilotage des flux : centraliser le suivi des expéditions par hub et par client pour une visibilité 360°.	Ingestion multi-sources : développer des scripts capable de lire et d'agréger du CSV, du JSON et de l'Excel
Performance opérationnelle : réduire le temps d'accès aux indicateurs clés.	Normalisation et nettoyage : automatiser la correction des données (gestion des types, valeurs NULL)
Aide à la décision(IA): préparer un terrain de données sain pour la prévision de charge et l'optimisation des ressources.	Architecture optimisée : choisir et configurer un SGBD garantissant l'intégrité et la rapidité des requêtes.
Sécurité : garantir la conformité aux réglementations en vigueur (RGPD)	Sécurisation : mettre en place des contraintes d'intégrité et des audits de données.

2. Identifier les sources de données, leur volumétrie et leurs typologies

Voici l'analyse détaillée des 3 fichiers sources de KNAUF Industries :

Sources	Format	Variables principales	Typologie	Nature
expeditions_knaufindustries.csv	CSV	shipment_id,client_id,ship_date,delivery_date,weight,hub	Structurées	Quantitatives (weight) et qualitatives
clients_knaufindustries.j	JSON	Client_id, company, city,	Semi-structurées	Qualitatives

LO Adja Maguette
IAM2DS
Nexa Digital School

son		sector		
hubs_ knaufindustries. xlsx (structure)	Excel	hub_id,hub_name, region, capacity_palettes	Structurées	Quantitatives (capacity_palettes) et qualitatives

Analyse de la volumétrie globale :

- L'échantillon fourni est restreint (quelques lignes par fichier), ce qui permet une intégration rapide et des tests unitaires mais nous allons générer plus de données pour nos analyses.
- En contexte réel pour une entreprise comme KNAUF Indutries, on estime un flux de plusieurs milliers d'expéditions quotidiennes. Le système doit donc être dimensionné pour intégrer une volumétrie de type "big data" à long terme.
- La volumétrie est techniquement impactée par l'hétérogénéité des formats (le JSON demandant plus de ressources et de parsing que le CSV).

3. Choisir et installer un SGBD adapté (SQL ou NoSQL)

- Justification du choix :

Le choix est porté sur PostgreSQL pour les raisons suivantes :

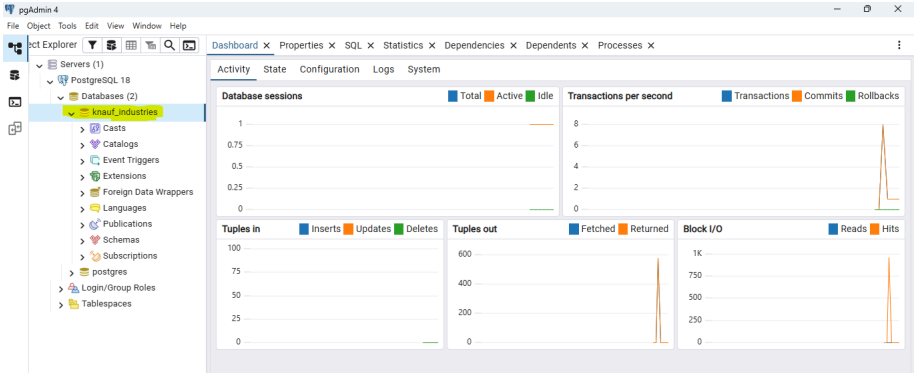
- gestion des données hybrides : contrairement à d'autres SGBD SQL, PostgreSQL gère nativement le format JSON via le type JSONB. C'est un atout majeur pour intégrer le fichier clients_ knaufindustries.json sans perte de structure.
- intégrité des données : dans le secteur de la logistique, l'exactitude des poids et des dates de livraison est critique. PostgreSQL garantit que chaque transaction est fiable.
- scalabilité et performance : capable de gérer des millions de lignes, ce SGBD est adapté à la croissance prévue des expéditions quotidiennes de l'entreprise.
- écosystème IA : PostgreSQL dispose d'extensions et d'une compatibilité parfaite avec les bibliothèques python, facilitant les futurs cas d'usage de prévision de charge.

- Preuve d'installation et opérationnalité

Le serveur a été installé en version 18.1 sur un environnement local.
https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

Eléments techniques de validation :

- Serveur : localhost
- Port : 5432
- Bdd : knauf_industries
- Outil : pgAdmin4



LO Adja Maguette
IAM2DS
Nexa Digital School

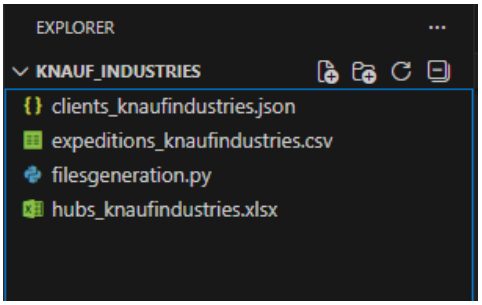
4. Agréger des données hétérogènes (CSV, JSON, Excel)

Avant de procéder à l'agrégation, il a été nécessaire de constituer physiquement les fichiers sources à partir des spécifications fournies :

- développement d'un script de génération (python) : pour simuler un contexte réel de production et tester la robustesse du système, nous avons généré un dataset de 500 expéditions pour le mois de février 2025.
- respect des formats : le script a été conçu pour exporter les données dans les 3 formats cibles (csv; json et excel), garantissant ainsi que l'étape d'agrégation traite bien les sources hétérogènes.
- intégration des cas d'erreurs : des valeurs nulles ont été volontairement introduites dans le fichier Json pour valider ultérieurement nos processus de nettoyage et de qualité de données.

Extrait du code de génération des fichiers :

```
import pandas as pd
import json
import random
from datetime import datetime, timedelta

# géneration du CSV
hubs_list = ['Lyon', 'Lille', 'Paris', 'Marseille']
clients_ids = [901, 654, 812]
data_exp = []

for i in range(1, 501):
    ship_id = 20000 + i
    c_id = random.choice(clients_ids)
    h = random.choice(hubs_list)
    weight = random.randint(50, 800)

    # Dates en Février 2025
    start_date = datetime(2025, 2, 1)
    ship_date = start_date + timedelta(days=random.randint(0, 27))
    delivery_date = ship_date + timedelta(days=random.randint(1, 3))

    data_exp.append([ship_id, c_id, ship_date.strftime('%Y-%m-%d'),
                     delivery_date.strftime('%Y-%m-%d'), weight, h])

df_csv = pd.DataFrame(data_exp, columns=['shipment_id', 'client_id',
'ship_date', 'delivery_date', 'weight', 'hub'])
df_csv.to_csv('expeditions_knaufindustries.csv', index=False)
```



LO Adja Maguette
IAM2DS
Nexa Digital School

Le processus d'intégration suit 3 étapes clés : Extract, Transform, Load

- Description du processus technique

- Extraction multi-format : utilisation de Python pour lire les 3 fichiers sources. Le défi est le fichier JSON (clients) qui doit être aplati pour correspondre au format tabulaire des autres sources.
- Harmonisation des données : nous utilisons client_id comme point de contact entre les expéditions et les clients
  -nous lions les expéditions aux hubs en utilisant le nom du hub présent dans le CSV et le fichier excel
  -conversion de toutes les dates au format SQL standard YYYY-MM-DD.

- Schéma de la table finale

L'intégration produit une table centrale qui regroupe toutes les informations nécessaires au pilotage.

Nom de la colonne	Source initiale	Type de données	Rôle
shipment_id	CSV	INTEGER	id unique de l'expédition
company	JSON	VARCHAR(100)	nom du client
sector	JSON	VARCHAR (50)	secteur d'activité pour l'analyse IA
weight	CSV	DECIMAL(10,2)	poids pour le calcul de la charge
hub_name	Excel	VARCHAR (50)	nom du hub de transit
region	Excel	VARCHAR (100)	localisation pour le pilotage

5. Identifier, analyser et corriger les valeurs manquantes

- identification des valeurs manquantes par variable

Script :

```
import pandas as pd

# Chargement
df_exp = pd.read_csv('expeditions_knaufindustries.csv')
df_cli = pd.read_json('clients_knaufindustries.json')
df_hubs = pd.read_excel('hubs_knaufindustries.xlsx')

# Fusion
```

LO Adja Maguette
IAM2DS
Nexa Digital School

```
df_merge = pd.merge(df_exp, df_cli, on='client_id', how='left')
df_final = pd.merge(df_merge, df_hubs, left_on='hub', right_on='hub_name',
how='left')

print("valeurs manquantes par variable")
identification = df_final.isnull().sum()
print(identification)

# Identification ciblée sur MedLog
print(df_final[df_final['company'] == 'MedLog'][['company', 'city',
'sector']])
```

Résultat :

```
PS C:\Users\LO Adja Maguette\OneDrive - VINCI Energies\Bureau\IA M2
DS\Knauf_industries> & "C:/Users/LO Adja
Maguette/AppData/Local/Programs/Python/Python312/python.exe"
"c:/Users/LO Adja Maguette/OneDrive - VINCI Energies/Bureau/IA M2
DS/Knauf_industries/gestion des valeurs manquantes.py"
valeurs manquantes par variable
shipment_id            0
client_id              0
ship_date              0
delivery_date          0
weight                 0
hub                    0
company                0
city                 183
sector                 0
hub_id                 0
hub_name               0
region                 0
capacity_palettes      0
dtype: int64
    company  city      sector
0    MedLog  None  Logistique
7    MedLog  None  Logistique
16   MedLog  None  Logistique
18   MedLog  None  Logistique
21   MedLog  None  Logistique
..      ...   ...         ...
491  MedLog  None  Logistique
492  MedLog  None  Logistique
493  MedLog  None  Logistique
```

LO Adja Maguette
IAM2DS
Nexa Digital School

```
498  MedLog  None  Logistique
499  MedLog  None  Logistique

[183 rows x 3 columns]
```

- identification des variables qualitatives et quantitatives :

Variables qualitatives : company, city, sector, hub, region. En cas de manque, on utilise l'imputation par le mode (la valeur la plus fréquente) ou imputation par déduction(basée sur une autre colonne)
Variables quantitatives : weight, capacity_palettes, shipment_id, client_id. En cas de manque, on utilise l'imputation par la moyenne ou la médiane (pour éviter l'influence des valeurs extrêmes)

- application d'une correction

Script :

```
# correction
# imputation déduite
df_final['city'] = df_final['city'].fillna('Marseille')

# vérification
print("Vérification")
apres = df_final['city'].isnull().sum()
print(f"Nombre de valeurs manquantes dans 'city' : {apres}")

# Aperçu
print("\nAperçu des données MedLog:")
print(df_final[df_final['company'] == 'MedLog'][['company',
'city']].head(5))
```

Résultat :

```
Vérification
Nombre de valeurs manquantes dans 'city' : 0

Aperçu des données MedLog:
   company       city
0   MedLog  Marseille
7   MedLog  Marseille
16  MedLog  Marseille
18  MedLog  Marseille
21  MedLog  Marseille
```

Explication :

LO Adja Maguette
IAM2DS
Nexa Digital School
L’analyse montre que l’intégralité des valeurs manquantes provient du client MedLog ‘ID 812’ . Conformément aux directives de Knauf_industries, nous avons exclu la suppression des lignes pour ne pas perdre 36% du volume d’activité.
Nous avons opté pour une imputation par déduction. Le croisement des données montrent que 100% des expéditions de medlog sont rattachées au Hub de Marseille. Par cohérence géographique, la valeur Marseille a été injectée dans les valeurs manquantes.

- Tableau descriptif

Variable	Type	Nb de manques	% de manques	diagnostic et décision
shipment_id	quantitative	0	0%	donnée intègre, pivot essentiel pour le suivi
weight	quantitative	0	0%	OK, cruciale pour l’analyse de charge
company	qualitative	0	0%	référentiel client complet
city	qualitative	183	36,6%	anomalie isolée pour medlog, imputation par déduction
sector	qualitative	0	0%	donnée récupérée via le référentiel client
hub_name	qualitative	0	0%	lien vers le référentiel hubs opérationnel
region	qualitative	0	0%	donnée complète issue de la fusion excel

L’analyse descriptive montre que l’intégrité quantitative est de 100%. L’effort de nettoyage s’est donc concentré sur la variable qualitative ‘city’ pour restaurer la cogérance géographique du dataset avant sont exportation vers le SGBD.

LO Adja Maguette
IAM2DS
Nexa Digital School

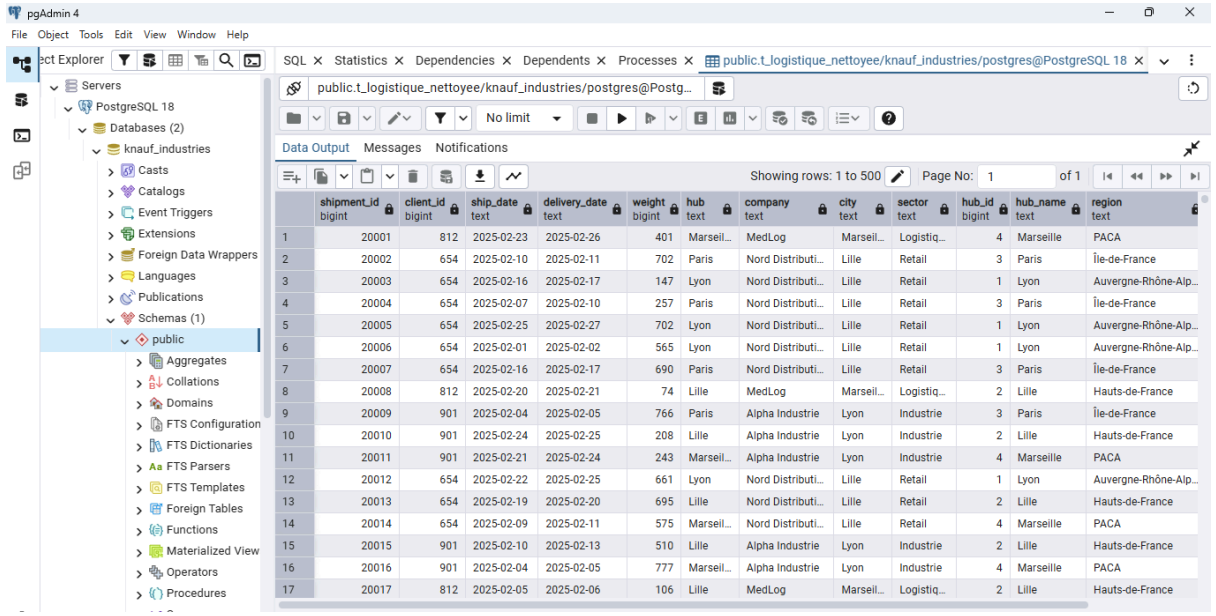
6. Charger les données nettoyées et vérifier l'intégrité

- Charger les données finales dans la base.

Script :

```
#connexion à la base
engine =
create_engine('postgresql://postgres:admin@localhost:5432/knauf_industries')

#chargement
# on charge le df dans une nouvelle table
df_final.to_sql('t_logistique_nettoyee', engine, if_exists='replace',
index=False)

print("Données chargées avec succès!")
```

Résultat :



Vérification du nombre de lignes :

```
SELECT
    COUNT(*) AS total_lignes,
    COUNT(CASE WHEN city IS NULL THEN 1 END) AS nb_villes_nulles,
    COUNT(CASE WHEN sector IS NULL THEN 1 END) AS nb_secteurs_nuls
FROM t_logistique_nettoyee;
```

	total_lignes bigint	nb_villes_nulles bigint	nb_secteurs_nuls bigint
1	500	0	0

LO Adja Maguette
IAM2DS
Nexa Digital School

Avant chargement des données dans la base :

```
nb_lignes_python = len(df_final)
print(f"nbre de ligne avant chargement : {nb_lignes_python} lignes")
```

```
nbre de ligne avant chargement : 500 lignes
```

Vérification des type de données :

```
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 't_logistique_nettoyee';
```

	column_name name	data_type character varying
1	shipment_id	bigint
2	client_id	bigint
3	weight	bigint
4	hub_id	bigint
5	capacity_palett...	bigint
6	hub	text
7	company	text
8	city	text
9	sector	text
10	region	text
11	ship_date	text
12	delivery_date	text
13	hub_name	text

7. Optimiser les types de variables et la structure

- Ajuster les types et les longueurs des champs

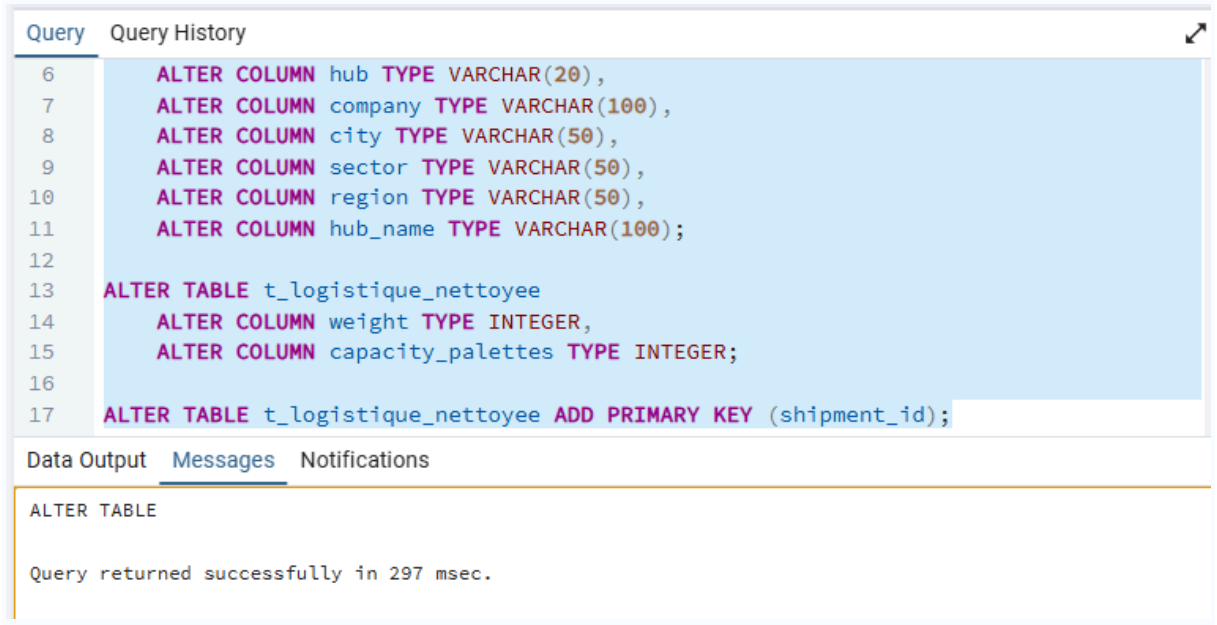
Requête SQL

```
ALTER TABLE t_logistique_nettoyee
    ALTER COLUMN ship_date TYPE DATE USING ship_date::DATE,
    ALTER COLUMN delivery_date TYPE DATE USING delivery_date::DATE;

ALTER TABLE t_logistique_nettoyee
    ALTER COLUMN hub TYPE VARCHAR(20),
```

LO Adja Maguette
IAM2DS
Nexa Digital School

```
    ALTER COLUMN company TYPE VARCHAR(100),
    ALTER COLUMN city TYPE VARCHAR(50),
    ALTER COLUMN sector TYPE VARCHAR(50),
    ALTER COLUMN region TYPE VARCHAR(50),
    ALTER COLUMN hub_name TYPE VARCHAR(100);

ALTER TABLE t_logistique_nettoyee
    ALTER COLUMN weight TYPE INTEGER,
    ALTER COLUMN capacity_palettes TYPE INTEGER;

ALTER TABLE t_logistique_nettoyee ADD PRIMARY KEY (shipment_id);
```

Résultat



Vérification :

	column_name name	data_type character varying
1	shipment_id	bigint
2	client_id	bigint
3	ship_date	date
4	delivery_date	date
5	weight	integer
6	hub_id	bigint
7	capacity_palett...	integer
8	city	character varying
9	sector	character varying
10	region	character varying
11	hub_name	character varying
12	hub	character varying
13	company	character varying

LO Adja Maguette
IAM2DS
Nexa Digital School

- Justifier les choix en lien avec la performance.

L'optimisation des types de données ne répond pas seulement à une exigence de propreté, elle a un impact direct sur l'efficacité du système SGBD.

Conversion du TEXT vers VARCHAR(N)

- Performance : En SQL, le type TEXT est stocké différemment des chaînes courtes. En utilisant VARCHAR(50), on indique à PostgreSQL la taille maximale attendue. Cela permet au moteur de base de données de mieux structurer les index.
- Gain : Les recherches (clauses WHERE city = 'Marseille') et les tris alphabétiques sont accélérés car le moteur traite des blocs de données de taille prévisible.

Passage du TEXT au type DATE

- Calculs natifs : Le format TEXT interdit tout calcul. En passant au type DATE, on permet à PostgreSQL d'utiliser ses fonctions internes (AGE(), EXTRACT()).
- Performance de tri : Trier des dates en format texte est chronophage et risqué (ex: "01/12" pourrait passer après "30/11" selon le format). Le type DATE est stocké sous forme de nombre entier en interne par SQL, ce qui rend les tris chronologiques instantanés.
- Indexation : On peut désormais créer un index temporel pour accélérer les requêtes portant sur une période précise (ex: livraisons du mois de mai).

Conversion du BIGINT vers INTEGER

- Optimisation du stockage : Un BIGINT occupe 8 octets par case, alors qu'un INTEGER n'en occupe que 4.
- Performance : Sur 500 lignes, la différence est minime, mais sur des millions de lignes (Big Data), cela divise par deux le poids de la colonne en mémoire vive (RAM). Pour les calculs (somme des poids, moyenne), le processeur traite les INTEGER plus rapidement que les BIGINT.

Mise en place de la Clé Primaire (shipment_id)

- Indexation automatique : Déclarer une clé primaire crée automatiquement un Index B-Tree.
- Performance : Sans clé primaire, pour trouver une expédition précise, SQL doit lire toute la table (Sequential Scan). Avec la clé primaire, il utilise l'index et trouve la ligne de manière quasi immédiate, même si la table contenait des millions d'entrées.

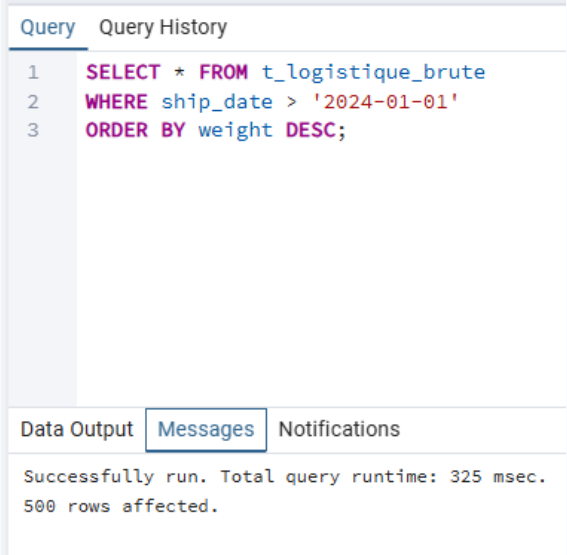
8. Comparer les temps d’exécution

- Performance

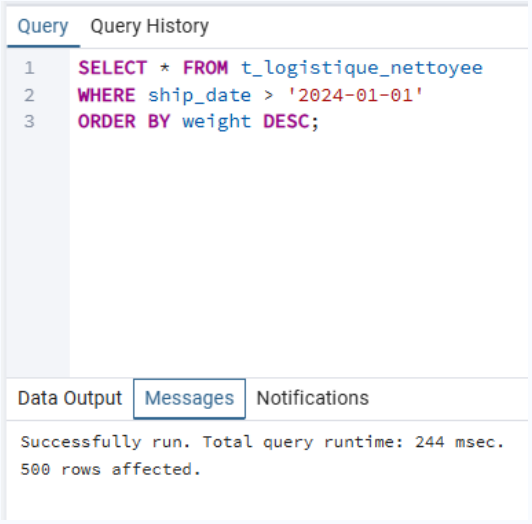
Pour cela, nous avons exporté nos données brutes sur notre base afin d’avoir deux tables et de pouvoir faire des comparaisons.

LO Adja Maguette
IAM2DS
Nexa Digital School

Requête1 sur table brute



Requête1 sur table nettoyée

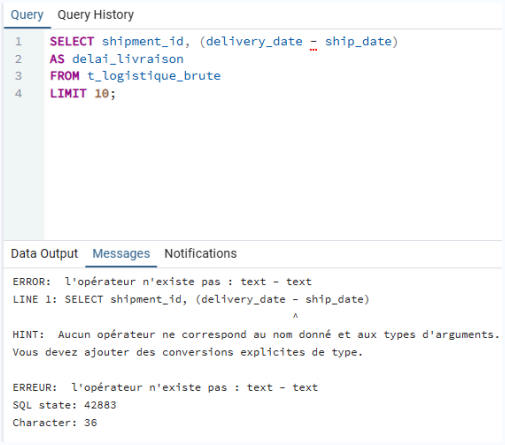


La comparaison des temps d'exécution entre les deux tables confirme l'impact de l'optimisation, table brute 325ms et table nettoyée 244ms. Ce gain de 25% est dû au remplacement des types TEXT par des types natifs (DATE, INTEGER). Sur la table brute, le moteur perd du temps à convertir les données pour les trier tandis que sur la table optimisée, l'accès est direct et plus léger en mémoire vive.

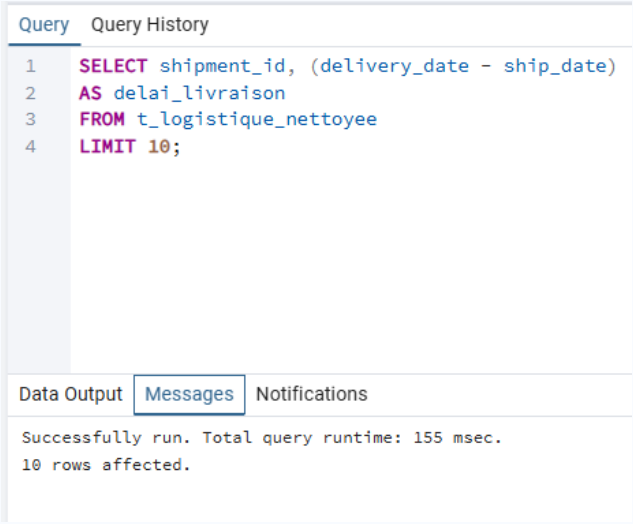
- fiabilité et calculabilité des données

Requête2 sur table brute

LO Adja Maguette
IAM2DS
Nexa Digital School



Requête2 sur table nettoyée



La tentative de calcul du délai de livraison génère une erreur sur la table brute. Le SGBD est incapable d'eefectuer des opération mathématique sur du texte.

Grâce au typage en DATE, le calcul s'exécute nativement et instantanément sur la table optimisée, 155ms.

L'optimisation a transformé une base de données "inerte" en un outil décisionnel fonctionnel. Sans cette étape, il serait impossible d'automatiser le calculs des KPIs logistiques ou d'entraîner un modèle IA à prédire des délais de livraison.

- La qualité métier

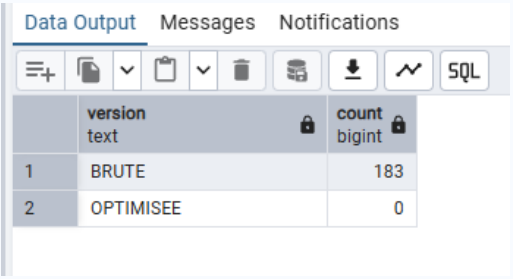
Requête

```
SELECT 'BRUTE' as version, count (*) FROM t_logistique_brute WHERE city
is NULL
```

LO Adja Maguette
IAM2DS
Nexa Digital School

```
UNION ALL
SELECT 'OPTIMISEE' as version, count (*) FROM t_logistique_nettoyee
WHERE city is NULL
```

Résultat



	version text	count bigint
1	BRUTE	183
2	OPTIMISEE	0

Ce dernier test valide la réussite de la phase nettoyage des données. La table brute présente 183 valeurs manquantes alors que la table nettoyée en présente 0.

Le nettoyage a transformé le dataset en une base de données fiable à 100%. Cette étape était indispensable pour l'étape suivante : entraînement d'un modèle IA. En effet, un algorithme de machine learning ne peut pas apprendre ou prédire à partir de champs vides, grâce à ce traitement, nous garantissons la précision et la robustesse des futures prédictions de charges pour Knauf Industries.

Conclusion : les 3 test (performance, fiabilité et qualité) démontrent que l'optimisation structurelle et le nettoyage des données ont permis d'obtenir une base de données performante, techniquement saine et prête pour un déploiement en IA.

9. RGPD, éthique et gouvernance des données

- identifier les données personnelles

Dans ce dataset, les données dont principalement liées à des entreprises et à de la logistique mais il y a certains points de vigilance à prendre en compte :

- identifiants indirects : bien que nous n'ayons pas de noms de personnes physique, les noms d'entreprise individuelles ou les adresses de livraison précises peuvent être considérées comme des données personnelles s'ils permettent d'identifier individu.
- données de contact : dans un système réel, les tables clients contiendraient des numéros de téléphone ou des emails de responsables logistiques.

- Mesures proposées :

- minimisation des données : ne stocker que ce qui est strictement nécessaire à l'expédition. Par exemple, le numéro du chauffeur n'est plus utile après livraison, il doit être supprimé ou anonymisé.
- sécurité : mise en place d'un contrôle d'accès strict sur postgreSQL pour que seul le personnel autorisé puisse voir les détails des clients *

LO Adja Maguette
IAM2DS
Nexa Digital School

- durée de conservation : fixer une purge automatique des données transactionnelles après 5 ans(durée légale standard pour les documents commerciaux) tout en conservant des agrégats anonymisés pour l'IA à long terme.

● Points de vigilance éthique :

- transparence des algorithmes : si une IA est utilisée pour prioriser des livraisons, les clients doivent savoir sur quels critères ils sont évalués pour éviter toute discrimination commerciale injustifiée.
- qualité des données : comme démontré lors du cas de Medlog, une donnée erronée peut mener à des décisions logistiques injustes. L'étique commence par la vérité des données.

Conclusion générale

Cet examen a permis de démontrer la maîtrise complète du cycle de vie de la donnée, de l'ingestion brute à l'optimisation SQL. En centralisant des sources hétérogènes (CSV, JSON, Excel), nous avons redressé un dataset critique pour Knauf Industries, notamment en résolvant les incohérences de la société Medlog. Les tests de performance confirment qu'une structure de données typée et nettoyée améliore la rapidité du système et garantit la fiabilité des calculs métier. Enfin, cette approche respecte les exigences de la gouvernance RGPD en assurant l'intégrité et la sécurité des informations. Ce socle technique robuste rend désormais possible le déploiement de modèles d'IA pour piloter la supply chain de manière prédictive.