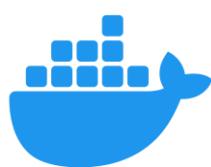


Compte Rendu TP

Cycle de vie de la donnée



PostgreSQL

EL HOUZI

Abdelmounaim

AMAZOUZ

Anas

M1 EISI INFRA

Sommaire

Table des matières

I.	Découverte et Compréhension :	2
A.	Connexion à OpenMetadata :	2
B.	Exploration du catalogue de données :	2
C.	Analyse des tables pertinentes :	4
D.	Synthèse :	7
II.	Modélisation et Transformation (PostgreSQL) :	8
A.	Préparation :	8
B.	Couche Silver (Raffinage) :	13
C.	Couche Gold (Agrégation métier) :	20
III.	Visualisation (Metabase) :	22
A.	Connexion et ajout de la source PostgreSQL à Metabase :	22
B.	Création des « charts » :	23
C.	Création du Dashboard Final :	25
IV.	Sécurité et Gouvernance (PostgreSQL +) :	26
A.	Création des rôles / utilisateurs :	26
B.	Audit (Simulation) :	26
C.	Tâche (Script SQL) :	28
D.	Row-level Security :	29

Ce document a pour objectif de décrire en détail chaque étape du TP, de répondre aux objectifs pédagogiques et de servir de référence personnelle. Il a notamment pour but de nous permettre, même après un an, de comprendre rapidement les choix réalisés et les travaux effectués.

I. Découverte et Compréhension :

Entreprise : VéloCity

Objectif : L'objectif de cette phase est d'identifier les tables sources pertinentes à partir du catalogue de données **VéloCity** afin de répondre aux besoins métiers du service Marketing pour le suivi de l'activité de location de vélos (nombre de locations, durée des trajets, répartition par ville, station, type de vélo et abonnement).

A. Connexion à OpenMetadata :

Pour cette partie, nous utilisons le fichier **catalogue_donnees.yml** fourni dans le dépôt GitHub, qui sert de documentation de secours pour toutes les tables brutes.

Ce fichier :

- Recense l'ensemble des tables.
- Fournit pour chaque table une description fonctionnelle et un schéma de colonnes.

B. Exploration du catalogue de données :

A. Objectif :

L'objectif de cette étape est d'identifier les tables pertinentes pour l'analyse de l'activité de location de vélos dans le service Marketing, en utilisant le fichier **catalogue_donnees.yml** comme source unique.

Nous voulons :

- Repérer les tables de **faits**
- Repérer les tables de **dimension**
- Identifier les tables contenant des **données personnelles (PII)**
- Marquer les tables **hors périmètre**

B. Outils utilisés :

- **VS Code** : pour ouvrir et explorer le fichier YAML.
- **Fonction de recherche intégrée (CTRL + F)** : utilisée pour simuler la barre de recherche
- **Annotations et surlignage** : pour identifier visuellement les tables pertinentes.

C. Navigation dans le catalogue de données :

Nous avons commencé par naviguer dans le catalogue de données fourni (**catalogue_donnees.yml**), qui sert de documentation de secours en cas d'indisponibilité d'**OpenMetadata**.

Le catalogue est organisé par **domaines métiers** et **sous-domaines**, chaque sous-domaine contenant plusieurs tables avec leurs colonnes et leurs descriptions.

Analyse des domaines :

```

12 domains:
13 # -----
14 # DOMAINE 1: Archives
15 #
16 > - name: "Archives" ...
17 #
18 # -----
19 # DOMAINE 2: Finance
20 #
21 > - name: "Finance" ...
22 #
23 # -----
24 # DOMAINE 3: Ressources Humaines
25 #
26 > - name: "Ressources Humaines" ...
27 #
28 # -----
29 # DOMAINE 4: IT Operations
30 #
31 > - name: "IT Operations" ...
32 #
33 # -----
34 # DOMAINE 5: Logistics & Maintenance
35 #
36 > - name: "Logistics & Maintenance" ...
37 #
38 # -----
39 # DOMAINE 6: Marketing
40 #
41 > - name: "Marketing" ...
42 #
43 # -----
44 # DOMAINE 7: Support
45 #
46 > - name: "Support" ...
47 #
48 # -----
49 # DOMAINE 8: External Data
50 #
51 > - name: "External Data" ...
52 
```

La navigation dans le catalogue a permis d'écarte rapidement certains domaines non pertinents pour le besoin Marketing :

- **Archives**, contenant des tables obsolètes comme *rentals_archive_2022* ou *daily_activity_summary_OLD*.
- **Ressources Humaines** : avec des informations sur le personnel interne (*employees*, *employee_payrolls*).
- **IT Operations** : qui contient des logs techniques et de l'inventaire matériel (*user_session_logs*, *api_request_logs*).
- **External Data** : données externes (météo, partenaires)

Les **domaines retenus** pour le TP sont :

- **Support** : qui contient les trajets et les informations sur les utilisateurs.
- **Finance** : informations sur les abonnements (**subscriptions**)
- **Logistics & Maintenance** : qui permet d'obtenir l'inventaire des vélos (**bikes**) et des stations (**bike_stations**).
- **Marketing** : référentiel des villes.

D. Utilisation de la recherche et des “Tags” (Source / PII) :

Pour aller plus loin, nous avons utilisé la recherche par mots-clés (**bike, rental, user, station, city, subscription**) et nous nous sommes appuyés sur des tags implicites, tels que :

- **Source** : table enregistrant les événements métier principaux.
- **PII (Personally Identifiable Information)** : table contenant des données personnelles sensibles.

Grâce à cette méthode, nous avons identifié les tables suivantes :

- **bike_rentals (Source)** : c'est la table principale, qui enregistre chaque trajet effectué par les utilisateurs avec le vélo utilisé, la station de départ et d'arrivée, et les horodatages. Cette table est essentielle pour calculer les indicateurs clés : nombre de trajets, durée, fréquentation par station et par ville.
- **user_accounts (PII)** : contient les informations personnelles des utilisateurs (prénom, nom, email, date de naissance, abonnement). Ces informations sont nécessaires pour rattacher les trajets à un utilisateur et analyser la base client, mais elles devront être anonymisées dans le dashboard final pour protéger la confidentialité.
- **Tables de référence / dimension** :
 - **subscriptions** : types d'abonnement pour segmenter les analyses.
 - **Bikes** : inventaire de la flotte pour identifier le type et le modèle de vélo.
 - **bike_stations** : référentiel des stations pour analyser la fréquentation par station.
 - **cities** : villes où le service est opéré, permettant de regrouper les trajets par localisation.

C. Analyse des tables pertinentes :

Après avoir identifié les tables pertinentes pour le TP, nous avons réalisé une analyse approfondie afin de comprendre leur rôle exact, leur structure et leur qualité de données. Nous avons pour chaque table examinée : le **schéma**, la **documentation**, les **profils de données**, et le **propriétaire (“Owner”)** responsable de la table.

Cette démarche nous permet de préparer une base fiable pour construire un **dashboard Marketing** efficace et sécurisé.

A. Table de faits :

bike_rentals :

- **Schéma** :
 - **rental_id** : identifiant unique du trajet (clé primaire)
 - **bike_id** : identifiant du vélo utilisé
 - **user_id** : identifiant de l'utilisateur
 - **start_station_id** : station de départ
 - **end_station_id** : station d'arrivée
 - **start_t** : horodatage de début
 - **end_t** : horodatage de fin

Cette table enregistre tous les **trajets** effectués par les utilisateurs. Elle contient les informations sur le vélo utilisé, l'utilisateur, la station de départ et d'arrivée, ainsi que les horaires de début et de fin. Elle constitue la base pour calculer les indicateurs clés : nombre de trajets, durée moyenne, fréquentation par station et par ville.

- **Profils de données :**
 - Vérification des doublons sur **rental_id**
 - Analyse de la distribution des trajets par jour et par station
 - Vérification des valeurs manquantes sur **bike_id** et **user_id**
- **Propriétaire :** Service **Support**, responsable des données liées aux trajets.

B. Tables de dimension :

user_accounts (PII) :

- **Schéma :**
 - **user_id:** identifiant unique (clé primaire)
 - **first_name, last_name:** prénom et nom
 - **email :** email utilisateur
 - **birthdate :** date de naissance
 - **registration_date:** date d'inscription
 - **subscription_id:** référence à la table **subscriptions**

Cette table contient les **informations personnelles des utilisateurs**, comme le prénom, le nom, l'email, la date de naissance et le type d'abonnement. Elle permet de rattacher chaque trajet à un utilisateur et d'analyser la base client tout en identifiant les différents types d'abonnements.

- **Profils de données :**
 - Nombre d'utilisateurs uniques
 - Répartition par âge ou type d'abonnement
 - Vérification des doublons sur **user_id** et de la validité des emails
- **Propriétaire :** Service **Support**, responsable des données liées aux trajets.
- **Remarque :** Données **PII** doivent être anonymisées pour le Dashboard.

subscriptions :

- **Schéma :**
 - **subscription_id :** identifiant unique de l'abonnement
 - **subscription_type :** type d'abonnement
 - **price_eur :** prix

Cette table recense les **différents types d'abonnement** disponibles, avec leur identifiant et le prix associé. Elle sert à segmenter les trajets selon le type d'abonnement et à analyser le comportement des utilisateurs selon leur catégorie d'abonnement.

- **Profils de données :**
 - Vérification des valeurs distinctes de **subscription_type**
 - Vérification de la cohérence des prix (**price_eur**)

- Propriétaire : Service **Finance**

bikes :

- Schéma :

- **bike_id:** identifiant unique du vélo
- **bike_type:** type de vélo
- **model_name :** modèle commercial
- **commissioning_date :** date de mise en service
- **status:** statut actuel du vélo

Cette table recense **tous les vélos du système**, avec leur type (mécanique ou électrique), leur modèle, leur date de mise en service et leur statut actuel. Elle permet d'analyser l'activité selon le type et le modèle de vélo, pour mieux comprendre l'usage et l'état du matériel.

- Profils de données :

- Vérification du nombre de vélos par type et modèle
- Vérification des vélos actifs et hors service

- Propriétaire : Service **Logistics & Maintenance**

bike_stations :

- Schéma :

- **station_id:** identifiant unique
- **station_name:** nom de la station
- **Latitude, longitude :** coordonnées géographiques
- **capacity :** capacité totale de la station
- **city_id :** référence à la table cities

Cette table contient **les informations sur toutes les stations de vélo**, incluant leur nom, leur localisation (latitude et longitude), leur capacité et la ville à laquelle elles appartiennent. Elle est utilisée pour analyser la fréquentation des stations et visualiser les trajets sur une carte.

- Profils de données :

- Vérification de la cohérence des coordonnées géographiques
- Nombre de stations par ville
- Capacité totale par station

- Propriétaire : Service **Logistics & Maintenance**

cities :

- Schéma :

- **city_id :** identifiant unique
- **city_name :** nom de la ville
- **region :** région administrative

Cette table recense **les villes dans lesquelles le service est disponible**, avec leur nom et leur région. Elle permet de regrouper et de visualiser les trajets par ville et de comparer l'activité entre différentes zones géographiques.

- **Profils de données :** Vérification du nombre de villes couvertes, absence de doublons et cohérence des noms et des régions.
- **Propriétaire :** Service **Marketing**

D. Synthèse :

Pour ce TP, nous avons sélectionné les tables suivantes pour construire le dashboard Marketing sur les trajets :

Table de faits :

- ✓ **bike_rentals** : elle contient tous les trajets des utilisateurs et constitue la base pour calculer les principaux indicateurs

Table de dimension (PII) :

- ✓ **user_accounts** : elle permet de relier chaque trajet à un utilisateur et de mieux comprendre la clientèle.

Table de dimension :

- ✓ **subscriptions** : elle donne le type d'abonnement de chaque utilisateur pour segmenter les analyses.
- ✓ **bikes** : elle fournit des informations sur les vélos pour analyser l'activité selon le type et le modèle.
- ✓ **bike_stations** : elle décrit les stations pour suivre la fréquentation et la localisation des trajets.
- ✓ **cities** : elle recense les villes pour analyser l'activité par zone géographique.

L'organisation des données autour d'une **table de faits centrale** et de **tables de dimensions** permet de construire un **modèle en étoile**, facilitant l'analyse des indicateurs et la création de dashboards clairs et performants.

II. Modélisation et Transformation (PostgreSQL) :

Objectif :

Dans cette partie, notre objectif est de préparer les données pour le dashboard Marketing en appliquant l'architecture Médailon : Raw (brut) → Silver (raffiné) → Gold (agrégé). Cette démarche permet de disposer de données propres, typées et prêtes à l'analyse, tout en conservant un historique et une traçabilité des transformations.

A. Préparation :

Démarrage des conteneurs Docker :

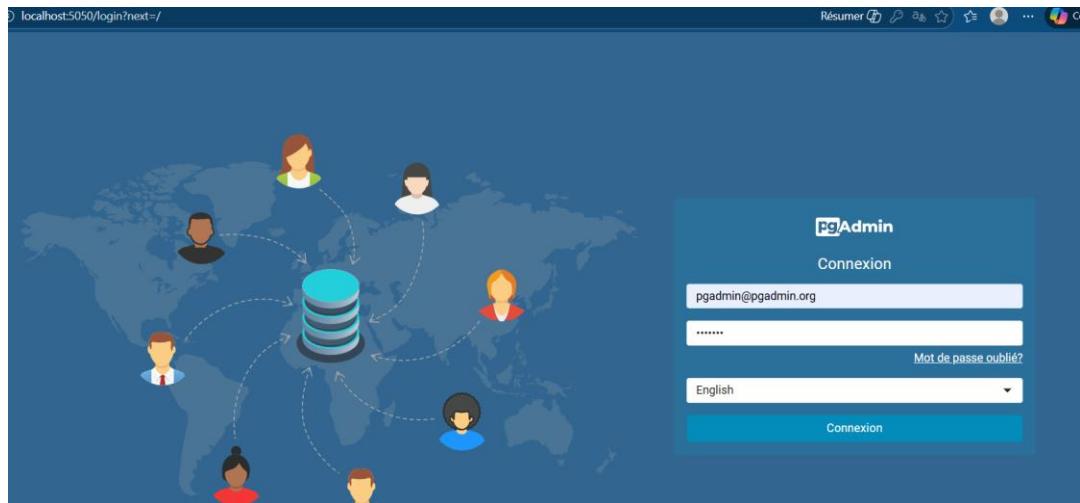
```
PS C:\Users\hz-ca\Desktop\L1-Intra-EPSI-2025-2026\Datamanag\tp\Tp_Amazouz_EL_HOUZI\docker_image_tp> docker ps
[+] Running 5/5
  ✓ Network docker_image_tp_unified_network   Created
  ✓ Container epsi_postgres_container          Healthy
  ✓ Container metabase_internal_db             Healthy
  ✓ Container epsi_pgadmin_container           Started
  ✓ Container metabase_app                   Started
PS C:\Users\hz-ca\Desktop\L1_Infra_EPSI_2025_2026\Da
```

Vérification des services démarrés :

●	docker_image_tp	-	-	-
●	metabase_app	abc4c14075c0	metabase/metabase:1 3000:3000 ↗	
●	epsi_pgadmin_container	a0a8831b39eb	dpage/pgadmin4:8 5050:80 ↗	
●	metabase_internal_db	aead899f87d0	postgres:13-alpine	
●	epsi_postgres_container	52d7b83e0ad0	postgres:18 5432:5432 ↗	

```
PS C:\Users\hz-ca\Desktop\L1-Intra-EPSI-2025-2026\Datamanag\tp\Tp_Amazouz_EL_HOUZI\docker_image_tp> docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
 NAMES
abc4c14075c0        metabase/metabase:latest    "/app/run_metabase.sh"   8 minutes ago       Up 8 minutes      0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp
tcp_metabase_app
a0a8831b39eb        dpage/pgadmin4:8          "/entrypoint.sh"        8 minutes ago       Up 8 minutes      0.0.0.0:5050->80/tcp, [::]:5050->80/tcp
  epsi_pgadmin_container
aead899f87d0        postgres:13-alpine          "docker-entrypoint.s..."  8 minutes ago       Up 8 minutes (healthy)  5432/tcp
  metabase_internal_db
52d7b83e0ad0        postgres:18                "docker-entrypoint.s..."  8 minutes ago       Up 8 minutes (healthy)  0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp
  epsi_postgres_container
PS C:\Users\hz-ca\Desktop\L1_Infra_EPSI_2025_2026\Da
```

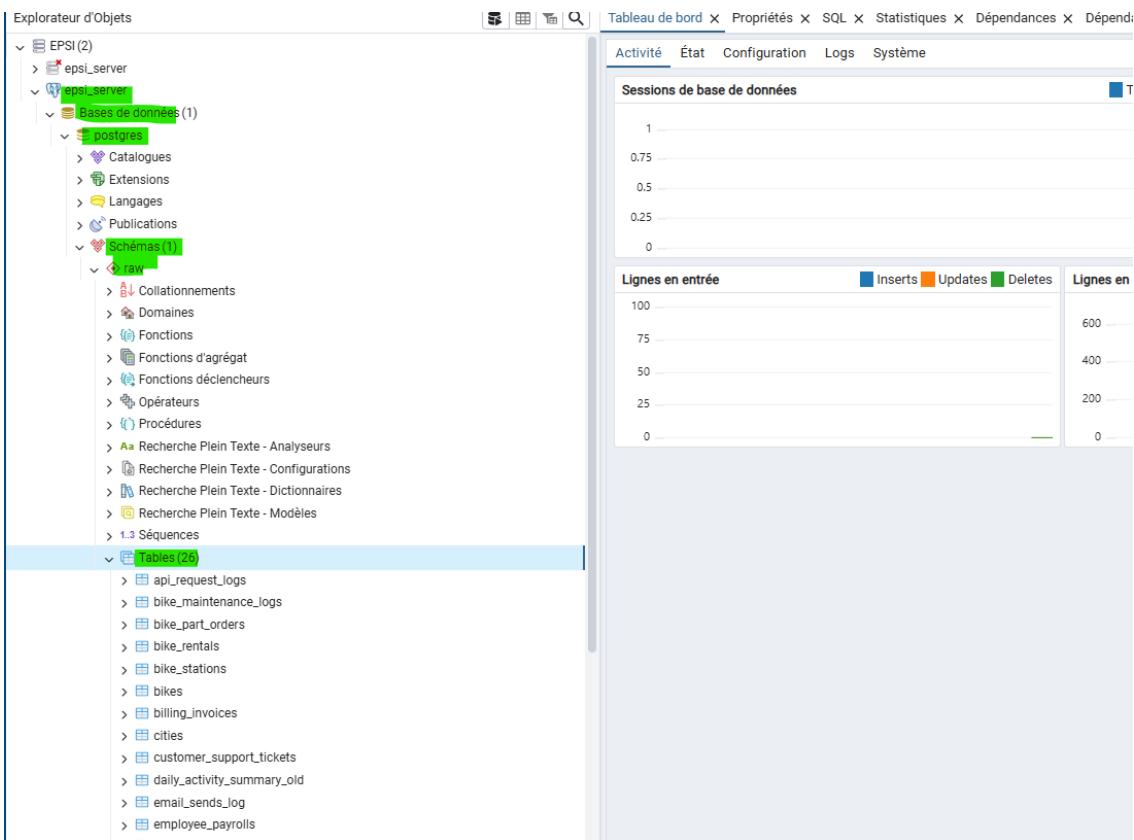
Pour explorer les tables de la base de données, nous utilisons l'outil **pgAdmin**, accessible via un navigateur web à l'adresse <http://localhost:5050>, après authentification.



Dans un premier temps, nous ajoutons un serveur PostgreSQL dans pgAdmin.
Ensuite, nous naviguons dans l'arborescence suivante :

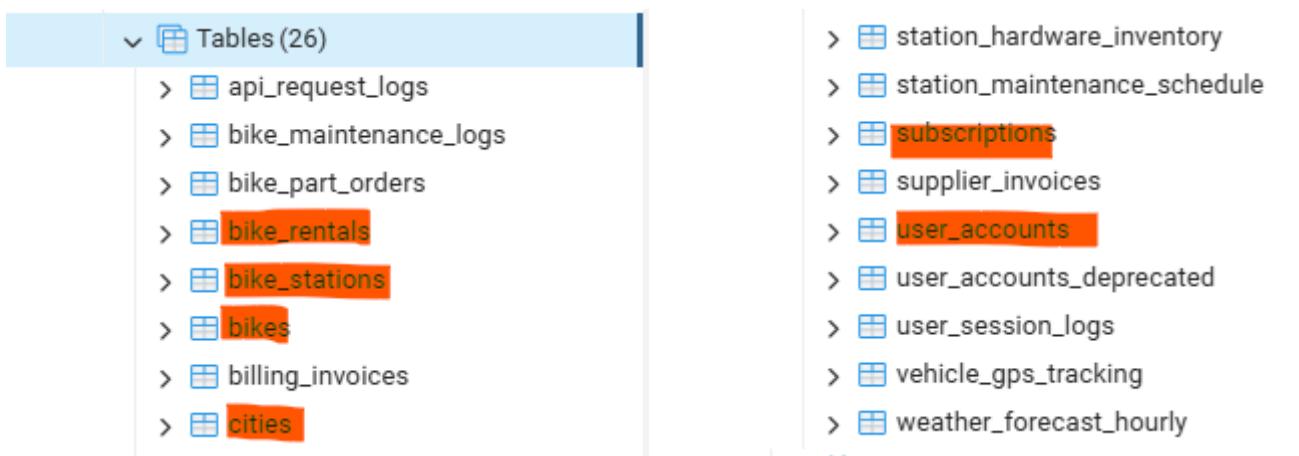
Paramètre	Valeur
Nom d'hôte / Adresse	postgres
Port	5432
Base de données de maintenance	postgres
Identifiant	postgres
Authentification Kerberos ?	Off
Mot de passe
Enregistrer le mot de passe ?	Off
Rôle	
Service	

Ensuite, nous naviguons dans l'arborescence suivante :



Nous y trouvons les **26** tables dans la base de données.

Nous avons identifié les tables pertinentes à interroger :



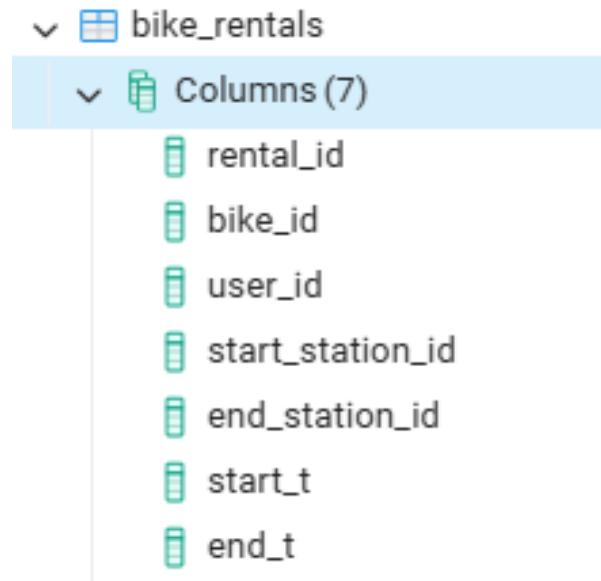
Vérification des métadonnées :

Nous avons comparé pour chaque table les informations disponibles dans OpenMetadata avec celles du catalogue interne :

- **bike_rentals** : toutes les colonnes (**rental_id**, **bike_id**, **user_id**, **start_station_id**, **end_station_id**, **start_t**, **end_t**) étaient présentes et les types de données correspondaient.
- **user_accounts** : les colonnes PII (**first_name**, **last_name**, **email**, **birthdate**, **registration_date**) et les références (**subscription_id**) étaient correctes.
- **subscriptions**, **bikes**, **bike_stations**, **cities** : les colonnes et types correspondaient également aux descriptions du catalogue.

Les métadonnées sont globalement cohérentes avec le catalogue OpenMetadata, ce qui confirme la fiabilité des informations pour la phase suivante.

```
- name: "bike_rentals"
  description: "Table de faits enregistrant
  columns:
    - name: "rental_id"
      description: "Identifiant unique du tra
    - name: "bike_id"
      description: "Clé étrangère vers la ta
    - name: "user_id"
      description: "Clé étrangère vers la ta
    - name: "start_station_id"
      description: "Clé étrangère (station)
    - name: "end_station_id"
      description: "Clé étrangère (station)
    - name: "start_t"
      description: "Horodatage de début."
    - name: "end_t"
      description: "Horodatage de fin."
```



Vérification de la qualité des données

Nous avons ensuite analysé la qualité des données pour chaque table afin de détecter les anomalies à corriger :

Pour la table **bike_rentals** :

Problème identifié	Description	Correction prévue (couche Silver)
start_t & end_t	Les colonnes start_t et end_t sont stockées en texte au lieu de timestamps	Doit être transformé en timestamp
start_station_id / end_station_id	On observe des identifiants avec des formats hétérogènes (station-11014, STA_25016, 25019, 33003)	On s'assure que les identifiants de station sont dans un format cohérent
Trajets trop courts	Présence de locations de moins de 2 minutes	Exclusion des trajets < 2 minutes

Pour la table **user_accounts** :

Problème identifié	Description	Correction prévue (couche Silver)
first_name / last_name	certains sont en majuscules d'autres en minuscules	Première lettre en majuscule, le reste en minuscule
registration_date	Mélange de formats (YYYY-MM-DD, MM/DD/YYYY, DD/MM/YYYY) et valeurs manquantes.	Convertir en format unique (YYYY-MM-DD)
email	Certaines adresses sont manquantes ou mal formées	On corrige les emails invalides
birthdate	Mélange de formats (YYYY-MM-DD, MM/DD/YYYY, DD/MM/YYYY) et valeurs manquantes.	Convertir en type DATE et uniformiser le format à YYYY-MM-DD
Phone_number	Plusieurs formats différents	Uniformiser tous les numéros dans un format standard international.

subscription_id	Incohérence de casse : sub_002, SUB_005 + Certains identifiants, comme sub_999, semblent utilisés pour des tests.	Standardiser la casse
------------------------	---	-----------------------

Pour la table **cities**:

Il y a une incohérence de langue : les villes sont en français alors que les pays et certaines régions sont en anglais.

Pour la table **bike_stations** :

Problème identifié	Description	Correction prévue (couche Silver)
Données Manquantes / Nulles	Mélange de STA_xxxx, station-xxxxxx	Adapter le format : renommer en STA_xxx
station_name	Tour Eiffel (français) et Eiffel Tower (anglais) + Trois stations différentes s'appellent Parc Nice	Garder une seule langue + Renommer pour préciser (Parc Nice Nord, Parc Nice Sud)
city_name	Villes invalides + Hyde Park est associé à Paris. Place Nantes associée à London.	Supprimer les associations incohérentes
latitude / longitude	Les colonnes sont en TEXT au lieu de DECIMAL + Mélange de points. et de virgules.	Convertir le type de la colonne en DECIMAL + Remplacer toutes les virgules par des points.

Pour la table **subscription** :

Problème identifié	Description	Correction prévue (couche Silver)
subscription_id	La plupart sont en minuscules mais la ligne 5 est en majuscules	Mettre toutes en minuscules
sub_name	La ligne 7 contient la chaîne de caractères null	Remplacer par une vraie valeur NULL
price	Trois formats différents pour la même devise + La ligne 6 (Gratuit) a une devise null	Convertir la colonne en type numérique pour permettre les calculs + garder que les valeur numérique et retirer € + définir une devise par défaut
latitude / longitude	Les colonnes sont en TEXT au lieu de DECIMAL + Mélange de points. et de virgules ,	standardiser sur le code ISO 4217 tout convertir en EUR +
end_date	Les abonnements en cours ont la valeur null (texte)	emplacer les chaînes "null" par de vrais NULL SQL pour permettre les requêtes sur les dates

Pour la table **bike_station** :

Problème identifié	Description	Correction prévue (couche Silver)
bike_type	Format pas adapter, mélange entre minuscule et majuscule	Mettre toutes en minuscules

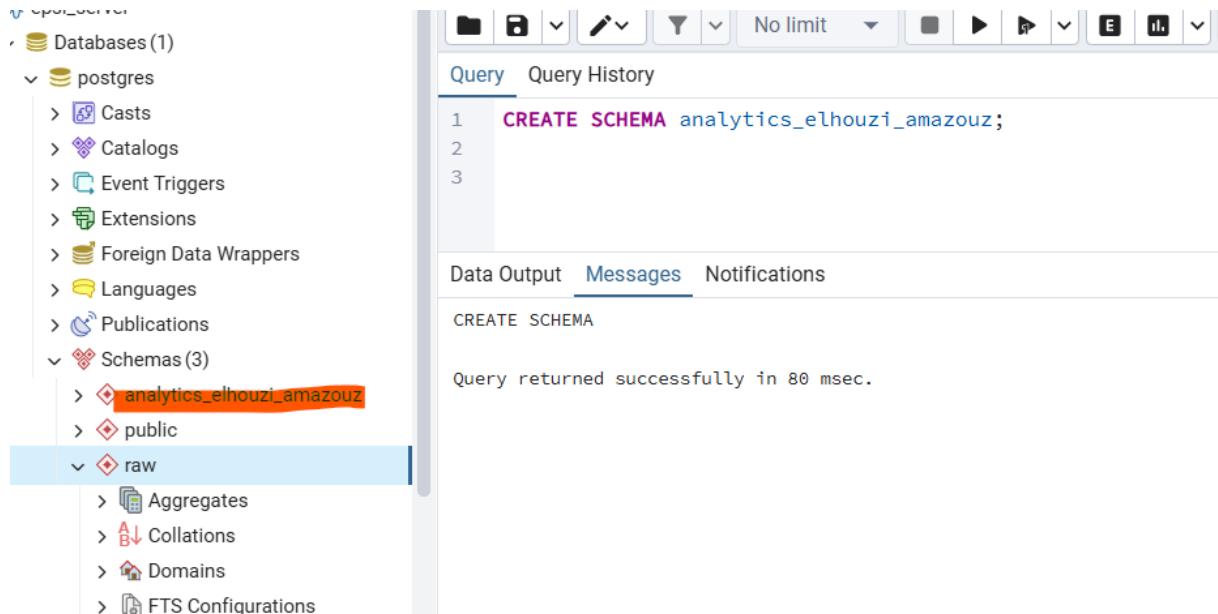
commissioning_date	La date de mise en service est 2099-01-01. C'est une date future.	Cas test : supprimer
model_name	type Mecanique mais modèle E-City Pro	Déduire le type à partir du nom du modèle

Création du schéma pour les transformations

Avant de commencer le nettoyage et la transformation des données, nous avons créé un schéma spécifique dans PostgreSQL afin de regrouper toutes les tables utilisées pour l'analyse.

Ce schéma permet de :

- Ne pas modifier les données brutes
- Organiser clairement les différentes étapes de transformation,
- Séparer les couches Silver et Gold du reste de la base.



B. Couche Silver (Raffinage) :

L'objectif de la couche **Silver** est de transformer les données brutes issues du schéma **raw** en données propres, cohérentes et directement exploitables pour l'analyse. Pour chaque table identifiée comme pertinente, nous avons créé une table équivalente dans le schéma **analytics_elhouzi_amazouz**, en appliquant des règles de nettoyage et de standardisation.

A. Raffinage de la table `bike_rentals` :

Transformations appliquées :

- Conversion des champs **start_t** et **end_t** en timestamps
- Calcul de la durée du trajet en minutes
- Suppression des trajets trop courts (durée < 2 minutes)
- Exclusion des trajets avec dates incohérentes ou valeurs manquantes

B. Raffinage de la table user_accounts :

Transformations appliquées :

- La colonne **user_id** a été convertie en **entier** afin de garantir l'identification unique et correcte de chaque utilisateur.
- Les **adresses e-mail** ont été corrigées par le remplacement des caractères incorrects (# → @) et par la conversion en **minuscules** afin d'assurer l'uniformité.
- Les **prénoms et noms** ont été normalisés : première lettre en **majuscule**, le reste en **minuscules**.
- Pour les utilisateurs dont le **prénom ou le nom était manquant**, ces informations ont été reconstituées à partir de l'adresse e-mail.
- Les **numéros de téléphone** ont été nettoyés afin de supprimer les caractères non conformes.
- Les **dates de naissance** et **dates d'inscription** ont été converties au format **DATE** pour faciliter les traitements temporels.
- Les **identifiants d'abonnement** ont été normalisés en minuscules et nettoyés.
- Les **utilisateurs incomplets** (absence de date de naissance, de date d'inscription ou d'abonnement) ont été supprimés afin de garantir la qualité des données.
- Une **colonne age** a été ajoutée, calculée à partir de la date de naissance, afin de permettre des **analyses démographiques**.

Avant Transformations :

	user_id [PK] integer	first_name character varying (100)	last_name character varying (100)	email character varying (150)	phone_number character varying (50)	birthdate character varying (50)	registration_date text	subscription_id character varying (50)
1	1	Frank	Anderson	frank.anderson@gmail.io	02.29.25.20.24	1978-02-24	2025-08-22	sub_002
2	2	Alice	Dupont	alice.dupont@protonmail.net	0731430179	2002-08-27	2025-09-05	sub_008
3	3	Grace	Garcia	grace.garcia@outlook.fr	+1-873-884-8242	2005-08-01	2025-03-23	sub_002
4	4	Nathan	Williams	nathan.williams@protonmail.fr	01.77.85.38.23	1972-08-04	2024-05-23	sub_008
5	5	Quentin	Mercier	quentin.mercier@hotmail.fr	690404833	1970-03-09	2024-01-10	sub_004
6	6	OPHELIE	Miller	ophelie.miller@yahoo.org	[null]	[null]	24 Feb 2024	[null]
7	7	Thomas	Da Silva	thomas.da.silva@yahoo.fr	667091224	1972-09-19	2025-11-08	sub_006
8	8	Thomas	Martin	thomas.martin@hotmail.org	332183109	1995-11-05	2024-01-24	sub_002
9	9	Eva	Williams	[null]	+33124381150	28/04/1987	03/15/24	SUB_005
10	10	Alice	Jones	alice.jones@icloud.com	01.47.11.76.24	1974-09-27	2025-07-29	SUB_005
11	11	Thomas	Leroy	thomas.leroy@gmail.com	05.86.13.26.88	1975-04-10	2024-02-19	sub_004
12	12	Alice	Brown	alice.brown@icloud.org	+33455398586	11-27-2000	04/28/25	sub_999
13	13	Manon	Lopez	manon.lopez@outlook.fr	0255578432	1995-07-09	2025-09-29	sub_006
14	14	NATHAN	Martin	nathan.martin#outlook.org	[null]	[null]	04 Dec 2024	[null]
15	15	grace	Lopez	gracelopez@hotmail	[null]	01/01/1900	01 Jan 2024	[null]
16	16	Laura	Davis	laura.davis@outlook.de	+1-308-117-4296	1988-03-14	2024-06-20	sub_008
17	17	Paul	Smith	paul.smith@gmail.net	+1-966-815-7340	1985-01-31	2025-02-16	sub_001
18	18	Alice	Davis	[null]	+33760657286	08/04/1994	03/10/24	SUB_008
19	19	Ugo	Lee	ugo.lee@gmail.com	+1-357-186-1033	1971-08-13	2023-12-30	sub_002
20	20	Ugo	Lee	[null]	[null]	[null]	[null]	[null]

Après Transformations :

> 📈 Languages
> 📖 Publications
✓ ❤️ Schemas (3)
↳ ⚡ analytics_elhouzi_amazouz
> 📊 Aggregates
> 🔍 Collations
> 🏫 Domains
> 📄 FTS Configurations
> 📄 FTS Dictionaries
> 📄 FTS Parsers
> 📄 FTS Templates
> 🗺 Foreign Tables
> 🎨 Functions
> 🌐 Materialized Views
> 📊 Operators
> 🎯 Procedures
> 1.3 Sequences
↳ 📁 Tables (2)
> 📁 silver_bike_rentals
> 📁 silver_user_accounts
> 🎯 Trigger Functions
↳ 📁 Types
> 🌐 Views
↳ ⚡ public
↳ ⚡ raw
=

	user_id	first_name	last_name	email	phone	birth_date	registration_date	subscription_id	age
1	1	Frank	Anderson	frank.anderson@gmail.io	0229252024	1978-02-24	2025-08-22	sub_002	47
2	2	Alice	Dupont	alice.dupont@protonmail.net	0731430179	2002-08-27	2025-09-05	sub_008	23
3	3	Grace	Garcia	grace.garcia@outlook.fr	08738848242	2005-08-01	2025-03-23	sub_002	20
4	4	Nathan	Williams	nathan.williams@protonmail.fr	0177853823	1972-08-04	2024-05-23	sub_008	53
5	5	Quentin	Mercier	quentin.mercier@hotmail.fr	0690404833	1970-03-09	2024-01-10	sub_004	55
6	7	Thomas	Da Silva	thomas.da.silva@yahoo.fr	0667091224	1972-09-19	2025-11-08	sub_006	53
7	8	Thomas	Martin	thomas.martin@hotmail.org	0332183109	1995-11-05	2024-01-24	sub_002	30
8	9	Eva	Williams	[null]	0124381150	1987-04-28	2024-03-15	sub_005	38
9	10	Alice	Jones	alice.jones@icloud.com	0147117624	1974-09-27	2025-07-29	sub_005	51
10	11	Thomas	Leroy	thomas.leroy@gmail.com	0586132688	1975-04-10	2024-02-19	sub_004	50
11	12	Alice	Brown	alice.brown@icloud.org	0455398586	2000-11-27	2025-04-28	sub_999	25
12	13	Manon	Lopez	manon.lopez@outlook.fr	0255578432	1995-07-09	2025-09-29	sub_006	30
13	14	Laura	Davis	laura.davis@outlook.de	03081174296	1988-03-14	2024-06-20	sub_008	37
14	17	Paul	Smith	paul.smith@gmail.net	09668157340	1985-01-31	2025-02-16	sub_001	40
15	18	Alice	Davis	[null]	0760657286	1994-04-08	2024-03-10	sub_008	31
16	19	Ugo	Lee	ugo.lee@gmail.com	03571861033	1971-08-13	2023-12-30	sub_002	54
17	20	Grace	Petit	grace.petit@yahoo.de	0374052221	1995-09-21	2024-08-24	sub_002	30
18	21	Nathan	Jones	nathan.jones@outlook.net	0782212361	1979-10-13	2023-12-23	sub_999	46
19	22	Bob	Smith	bob.smith@hotmail.fr	0245563201	2001-12-15	2024-12-30	sub_001	23
20	23	Laura	Johnson	laura.johnson@icloud.fr	0249201318	1991-03-14	2024-11-05	sub_001	34
21	24	Grace	Smith	grace.smith@outlook.fr	0489423196	1999-06-12	2025-04-13	sub_999	26
22	25	Manon	Miller	manon.miller@gmail.de	0208901139	1988-12-04	2025-01-29	sub_002	37
23	26	Alice	Brown	alice.brown@icloud.de	0545963478	2004-04-12	2025-08-02	sub_003	21

C. Raffinement de la table subscriptions :

Transformations appliquées :

- **subscription_id** : Suppression des espaces et mise en minuscules
- **sub_name** : remplacement des valeurs manquantes ou invalides par "Non défini" afin d'éviter les champs vides.
- **Price** : Suppression de tous les caractères non numériques + Conversion en type numérique
- **Currency** : Remplacement des valeurs manquantes par EUR + Uniformisation des valeurs pour passer toutes en EUR
- **start_date, end_date** : Conversion en type DATE
- **is_active** : Une souscription est considérée active si end_date est nulle ou supérieure à la date actuelle.

Avant Transformations :

	subscription_id [PK] character varying (50) ↴	sub_name character varying (100) ↴	price character varying (50) ↴	currency character varying (20) ↴	country_scope character varying (50) ↴	start_date date ↴	end_date date ↴
1	sub_001	Mensuel	29.99	EUR	France	2022-01-01	[null]
2	sub_002	Annuel	299.99	€	Italy	2022-01-01	[null]
3	sub_003	Pay-as-you-go	1.50	Euro	International	2022-01-01	[null]
4	sub_004	Étudiant	12.99	GBP	United Kingdom	2023-09-01	2024-08-31
5	SUB_005	Mensuel	29.99	EUR	Belgium	2023-01-01	[null]
6	sub_006	Gratuit	0.00	[null]	International	2022-01-01	[null]
7	sub_007	[null]	45.00	EUR	Germany	2024-01-01	[null]
8	sub_008	Premium Plus	50.00 €	EUR	Spain	2024-06-01	[null]

Après Transformations :

	subscription_id	sub_name	price	currency	country_scope	start_date	end_date	is_active
	text	character varying	numeric (10,2)	character varying	character varying (50)	date	date	boolean
1	sub_001	Mensuel	29.99	EUR	France	2022-01-01	[null]	true
2	sub_002	Annuel	299.99	EUR	Italy	2022-01-01	[null]	true
3	sub_003	Pay-as-you-go	1.50	EUR	International	2022-01-01	[null]	true
4	sub_004	Étudiant	12.99	GBP	United Kingdom	2023-09-01	2024-08-31	false
5	sub_005	Mensuel	29.99	EUR	Belgium	2023-01-01	[null]	true
6	sub_006	Gratuit	0.00	EUR	International	2022-01-01	[null]	true
7	sub_007	Non défini	45.00	EUR	Germany	2024-01-01	[null]	true
8	sub_008	Premium Plus	50.00	EUR	Spain	2024-06-01	[null]	true

D. Raffinage de la table bikes :

Transformations appliquées :

- **bike_id** : converti en entier
- **model_name** : suppression des espaces autour du texte pour uniformiser les valeurs.
- **bike_type** : Standardisation en “mécanique” ou “électrique” en fonction du nom du modèle ou du type renseigné.
- **commissioning_date** : Conversion en type DATE + Les dates incorrectes (1970-01-01) ou manquantes sont remplacées par NULL.
- **Status** : passage en minuscules et suppression des espaces.
- **year_service** : calcul automatique à partir de la date de mise en service pour les analyses.

Avant Transformations :

	bike_id	bike_type	model_name	commissioning_date	status
	[PK] integer	character varying (50)	character varying (100)	date	character varying (50)
1	1000	electrique	CityBike V2	2025-01-28	in_maintenance
2	1001	mecanique	CityBike V2	2023-04-01	lost
3	1002	mecanique	CityBike Sport	2024-07-13	active
4	1003	electrique	E-City Pro	2024-04-30	active
5	1004	electrique	E-City V3	2023-03-20	active
6	1005	mecanique	CityBike V2	2023-03-12	retired
7	1006	mecanique	CityBike Sport	2024-08-28	active
8	1007	mecanique	CityBike Sport	2025-11-08	in_maintenance
9	1008	electrique	E-City Pro	2024-10-25	active
10	1009	mecanique	CityBike V1	2023-12-07	active
11	1010	Velo	CityBike Sport	2025-08-03	active
12	1011	electrique	E-City V2	2025-09-12	active
13	1012	electrique	CityBike V2	2023-07-13	active
14	1013	electrique	E-City V3	2025-05-26	active
15	1014	electrique	E-City V2	2024-07-28	in_maintenance
16	1015	electrique	E-City V3	2023-12-13	retired
17	1016	mecanique	CityBike V2	2024-09-16	active
18	1017	Mecanique	E-City Pro	2024-08-29	in_maintenance
19	1018	electrique	E-City Pro	1970-01-01	active
20	1019	mecanique	CityBike V2	2025-05-23	active
21	1020	electrique	E-City Pro	2025-08-16	active
22	1021	electrique	E-City V2	1970-01-01	active
23	1022	electric	E-City V2	2024-05-27	active
24	1023	Mecanique	CityBike V1	2023-11-10	active
25	1024	mecanique	CityBike V1	2025-02-19	active
26	1025	mecanique	CityBike V1	2025-03-01	active

Après Transformations :

	bike_id integer	model_name text	bike_type text	commissioning_date date	status text	year_service numeric
1	1000	CityBike V2	mecanique	2025-01-28	in_maintenance	2025
2	1001	CityBike V2	mecanique	2023-04-01	lost	2023
3	1002	CityBike Sport	mecanique	2024-07-13	active	2024
4	1003	E-City Pro	electrique	2024-04-30	active	2024
5	1004	E-City V3	electrique	2023-03-20	active	2023
6	1005	CityBike V2	mecanique	2023-03-12	retired	2023
7	1006	CityBike Sport	mecanique	2024-08-28	active	2024
8	1007	CityBike Sport	mecanique	2025-11-08	in_maintenance	2025
9	1008	E-City Pro	electrique	2024-10-25	active	2024
10	1009	CityBike V1	mecanique	2023-12-07	active	2023
11	1010	CityBike Sport	mecanique	2025-08-03	active	2025
12	1011	E-City V2	electrique	2025-09-12	active	2025
13	1012	CityBike V2	mecanique	2023-07-13	active	2023
14	1013	E-City V3	electrique	2025-05-26	active	2025
15	1014	E-City V2	electrique	2024-07-28	in_maintenance	2024
16	1015	E-City V3	electrique	2023-12-13	retired	2023
17	1016	CityBike V2	mecanique	2024-09-16	active	2024
18	1017	E-City Pro	electrique	2024-08-29	in_maintenance	2024
19	1018	E-City Pro	electrique	[null]	active	[null]
20	1019	CityBike V2	mecanique	2025-05-23	active	2025
21	1020	E-City Pro	electrique	2025-08-16	active	2025
22	1021	E-City V2	electrique	[null]	active	[null]

E. Raffinement de la table cities:

Transformations appliquées :

- **city_id** : conversion en entier
- **city_name** : Suppression des espaces superflus autour du texte + Passage en majuscules pour assurer la cohérence avec la table **silver_bike_stations** lors des jointures.
- **Region** : suppression des espaces inutiles pour standardiser les noms de régions.
- **Country** : suppression des espaces et passage en majuscules pour homogénéiser les valeurs et faciliter les regroupements.

Avant Transformations :

	city_id [PK] integer	city_name character varying (100)	region character varying (100)	country character varying (50)
1	1	Paris	Île-de-France	France
2	2	Lyon	Auvergne-Rhône-Alpes	France
3	3	Marseille	Provence-Alpes-Côte d'Az...	France
4	4	Lille	Hauts-de-France	France
5	5	Toulouse	Occitanie	France
6	6	Bordeaux	Nouvelle-Aquitaine	France
7	7	Nantes	Pays de la Loire	France
8	8	Strasbourg	Grand Est	France
9	9	Montpellier	Occitanie	France

Après Transformations :

	city_id integer	city_name text	region text	country text
1	1	PARIS	Île-de-France	FRANCE
2	2	LYON	Auvergne-Rhône-Alpes	FRANCE
3	3	MARSEILLE	Provence-Alpes-Côte d'Azur	FRANCE
4	4	LILLE	Hauts-de-France	FRANCE
5	5	TOULOUSE	Occitanie	FRANCE
6	6	BORDEAUX	Nouvelle-Aquitaine	FRANCE
7	7	NANTES	Pays de la Loire	FRANCE
8	8	STRASBOURG	Grand Est	FRANCE
9	9	MONTPELLIER	Occitanie	FRANCE

F. Raffinage de la table bike_stations :

Transformations appliquées :

- **station_id** : Normalisation du format en ajoutant le préfixe **STA_**
- **city_id** : *conversion en entier*
- **city_name** : passage en majuscules pour les jointures avec **silver_citiy**
- **capacity** : Remplacement des valeurs manquantes ou invalides par 0 + Conversion en entier
- **latitude, longitude** : Transformation des valeurs manquantes en Null + Conversion en type DECIMAL avec remplacement des virgules en des points

Avant Transformations :

	station_id [PK] character varying (20)	station_name character varying (255)	city_name character varying (100)	latitude text	longitude text	capacity integer	city_id integer
1	ERR_001	Hyde Park	Paris	48.85	2.35	20	1
2	ERR_002	Tour Eiffel	London	51.50	-0.12	30	24
3	ERR_ORPH	Station Fantome	Nowhere	0.0	0.0	10	99
4	STA_10001	Parc Nice	Nice	42.19830721305225	6.945629305111375	49	10
5	STA_10002	Université Nice	Nice	47.947538605843775	5.9239479934031145	11	10
6	STA_10003	Centre Nice	Nice	42.30598848112203	12.772865870994362	17	10
7	STA_10004	Parc Nice	Nice	46.26700878736621	0.6678544493807204	45	10
8	STA_10005	Parc Nice	Nice	49.529433654823585	-4.8275319556923435	10	10
9	STA_1001	Gare du Nord	Paris	48.086483279967155	10.403452327754668	10	1
10	STA_1002	Bastille	Paris	42.37735836095015	2.419109438601379	25	1
11	STA_1003	Hôtel de Ville	Paris	46.73448929131848	8.193088797362975	17	1

Après Transformations :

	station_id text	city_id integer	station_name text	city text	capacity integer	latitude numeric	longitude numeric
1	STA_1001	1	Gare Du Nord	PARIS	10	48.086483	10.403452
2	STA_1002	1	Bastille	PARIS	25	42.377358	2.419109
3	STA_1003	1	Hôtel De Ville	PARIS	17	46.734489	8.193089
4	STA_1004	1	Champs-Élysées	PARIS	12	47.781049	-2.253960
5	STA_1005	1	Louvre - Rivoli	PARIS	17	47.552808	4.210199
6	STA_1006	1	Gare De Lyon	PARIS	29	49.343301	14.964065
7	STA_1007	1	Montparnasse	PARIS	31	45.736642	6.489478
8	STA_1008	1	République	PARIS	18	48.242992	13.570120
9	STA_1009	1	Opéra	PARIS	49	40.278352	1.931304

C. Couche Gold (Agrégation métier) :

Objectif : L'objectif de la couche Gold est de produire une table finale, propre et prête à l'usage, permettant d'alimenter directement le dashboard.

La table cible est : **analytics_elhouzi_amazouz.gold_daily_activity**

Elle permet d'agréger les informations des trajets selon différents axes métier : jour, ville, station, type de vélo et type d'abonnement.

Métriques calculées :

- **total_rentals** (nombre de locations)
- **average_duration_minutes** (durée moyenne)
- **unique_users** (utilisateurs distincts)

Nous allons partir de la table des faits (**silver_bike_rentals**) faire des jointures vers les dimensions pour récupérer les informations essentielles pour l'analyse :

- **CENTRE : silver_bike_rentals** (Trajets)
- **JOINTURE 1 : vers silver_stations** (Pour avoir le Nom de la Station et la Ville).
- **JOINTURE 2 : vers silver_bikes** (Pour avoir le Type : Electrique/Mécanique).
- **JOINTURE 3 : vers silver_users** (Pour faire le lien avec l'abonnement).
- **JOINTURE 4 : vers silver_subscriptions** (Pour avoir le Nom de l'abonnement)

```

CREATE TABLE analytics_elhouzi_amazouz.gold_daily_activity AS
SELECT
    r.start_ts::DATE AS rental_date,
    s.city_name AS city,
    st.station_name AS start_station,
    b.bike_type AS bike_type,
    sub.sub_name AS sub_name,

    COUNT(r.rental_id) AS total_rentals,
    ROUND(AVG(r.duration_minutes), 2) AS average_duration_minutes,
    COUNT(DISTINCT r.user_id) AS unique_users

FROM analytics_elhouzi_amazouz.silver_bike_rentals r
JOIN analytics_elhouzi_amazouz.silver_user_accounts u
    ON r.user_id = u.user_id

JOIN analytics_elhouzi_amazouz.silver_bikes b
    ON r.bike_id = b.bike_id

JOIN analytics_elhouzi_amazouz.silver_bike_stations st
    ON r.start_station_id = st.station_id

JOIN analytics_elhouzi_amazouz.silver_cities s
    ON st.city_id = s.city_id

JOIN analytics_elhouzi_amazouz.silver_subscriptions sub
    ON u.subscription_id = sub.subscription_id

GROUP BY rental_date, city_name, start_station, bike_type, sub_name
ORDER BY rental_date, city_name, start_station;

```

Cette table est prête à l'emploi pour le dashboard. Elle permet d'analyser le nombre total de locations, la durée moyenne des locations (en minutes) ainsi que le nombre d'utilisateurs distincts.

	rental_date date	city text	start_station text	bike_type text	sub_name character varying	total_rentals bigint	average_duration_minutes numeric	unique_users bigint
1	2023-01-01	ANTWERP	Centre Antwerp	electrique	Annuel	1	72.98	1
2	2023-01-01	ANTWERP	Centre Antwerp	electrique	Étudiant	1	46.52	1
3	2023-01-01	ANTWERP	Centre Antwerp	electrique	Premium Plus	1	14.30	1
4	2023-01-01	ANTWERP	Centre Antwerp	mecanique	Étudiant	1	35.18	1
5	2023-01-01	ANTWERP	Centre Antwerp	mecanique	Mensuel	1	60.93	1
6	2023-01-01	ANTWERP	Centre Antwerp	mecanique	Non défini	1	43.98	1
7	2023-01-01	ANTWERP	Centre Antwerp	mecanique	Premium Plus	2	97.98	2
8	2023-01-01	ANTWERP	Mairie De Antwerp	electrique	Mensuel	1	22.55	1
9	2023-01-01	ANTWERP	Mairie De Antwerp	mecanique	Étudiant	1	14.57	1
10	2023-01-01	ANTWERP	Mairie De Antwerp	mecanique	Pay-as-you-go	1	78.92	1
11	2023-01-01	ANTWERP	Université Antwerp	electrique	Étudiant	1	116.90	1
12	2023-01-01	ANTWERP	Université Antwerp	electrique	Gratuit	1	94.48	1
13	2023-01-01	ANTWERP	Université Antwerp	electrique	Mensuel	1	115.50	1
14	2023-01-01	ANTWERP	Université Antwerp	electrique	Pay-as-you-go	1	40.72	1
15	2023-01-01	ANTWERP	Université Antwerp	mecanique	Gratuit	1	56.00	1
16	2023-01-01	ANTWERP	Université Antwerp	mecanique	Non défini	1	29.37	1
17	2023-01-01	ANTWERP	Université Antwerp	mecanique	Premium Plus	2	43.59	2
18	2023-01-01	BARCELONA	Centre Barcelona	mecanique	Étudiant	1	116.83	1

III. Visualisation (Metabase) :

Objectif : connecter la table **analytics_elhouzi_amazouz.gold_daily_activity** et construire le Dashboard Marketing demandé.

A. Connexion et ajout de la source PostgreSQL à Metabase :

- ✓ Ouvrir un navigateur web et accéder à l'adresse <http://localhost:3000>.
- ✓ Se connecter à Metabase avec un compte disposant des droits administrateurs.
- ✓ Dans la barre supérieure, cliquer sur l'icône Administration.
- ✓ Dans le menu latéral, sélectionner Databases.
- ✓ Cliquer sur Add a database.
- ✓ Renseigner les différents champs de configuration requis :

Afficher le nom
VéloCity Prod

Hôte
postgres

Port
5432

Nom de la base de données
postgres

Nom d'utilisateur
postgres@gmail.com

Mot de passe

Schemas
Tout

Bases de données

BASES DE DONNÉES > VÉLOCITÉ PROD > ANALYTICS_ELHOUZI_AMAZOUZ

Gold Daily Activity	Silver Bike Rentals	Silver Bike Stations
Silver Bikes	Silver Cities	Silver Subscriptions
Silver User Accounts		

B. Création des « charts » :

Pour Ajouter les graphique visuels il faut :

- ✓ Cliquez sur le bouton « + New »
- ✓ Sélection la base (vélocity Prod)
- ✓ Selection la table: Gold Daily Activity dans schema analytics_elhouzi_amazouz

The screenshot shows the configuration interface for a chart. At the top, it says "VéloCity Prod / analytics_elhouzi_amazouz / Gold Daily Activity". There are buttons for "Voir SQL" and "Sauvegarde". Below this, under "Données", there is a dropdown menu set to "Gold Daily Activity". There are also icons for refresh and search. Under "Filtre", there is a placeholder "Ajouter des filtres pour préciser votre réponse". Under "Résumer", there are two boxes: "Choisir une fonction ou une métrique" and "Choisissez une colonne d'agrégation", both currently empty. At the bottom is a large blue "Visualiser" button.

Chart 1 : Évolution du nombre de locations dans le temps :

Objectif : Voir la tendance des locations jour par jour.

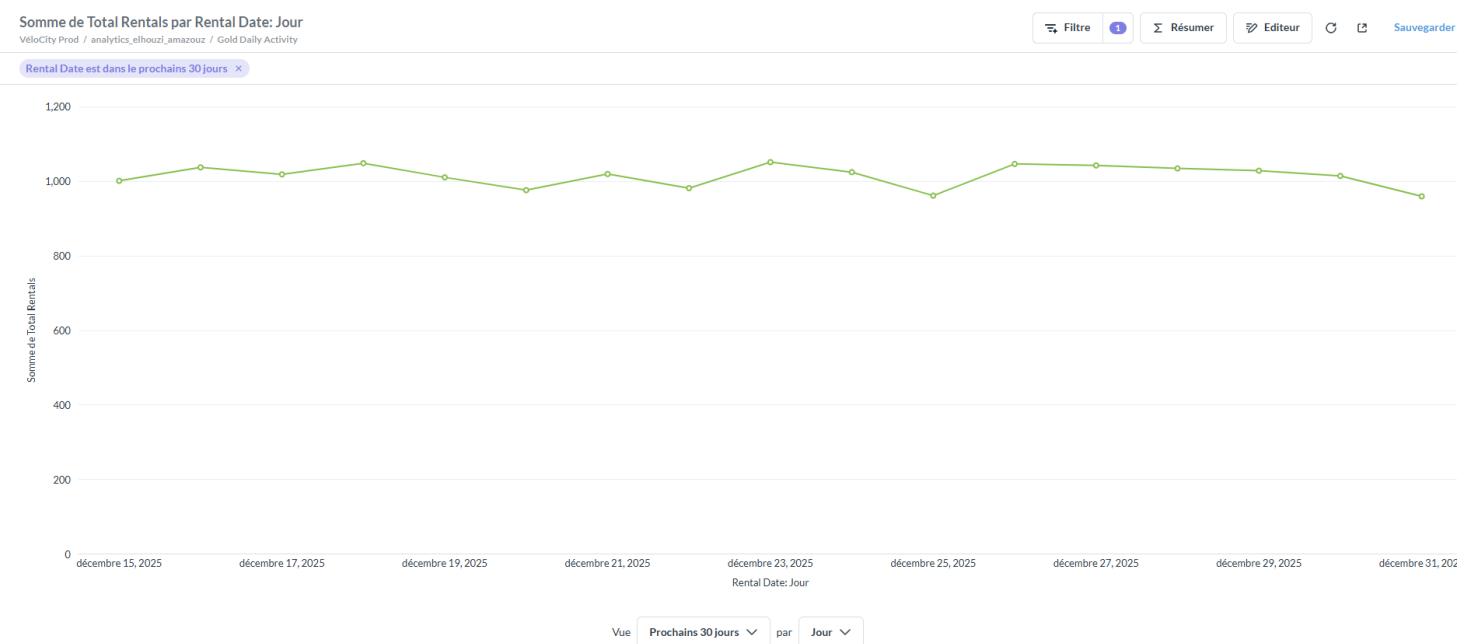


Chart 2 : Classement des trois villes ayant le plus grand nombre de locations

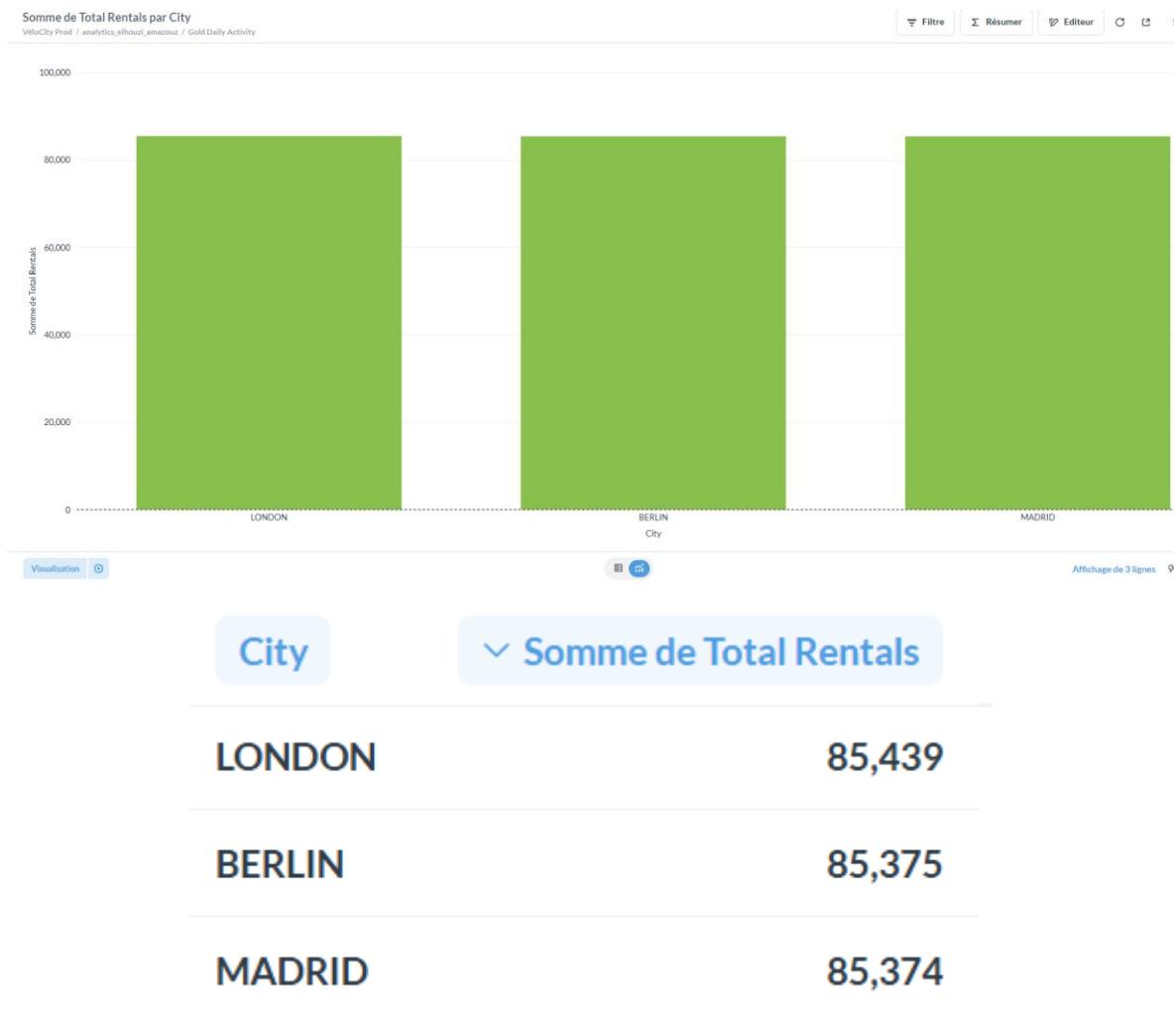
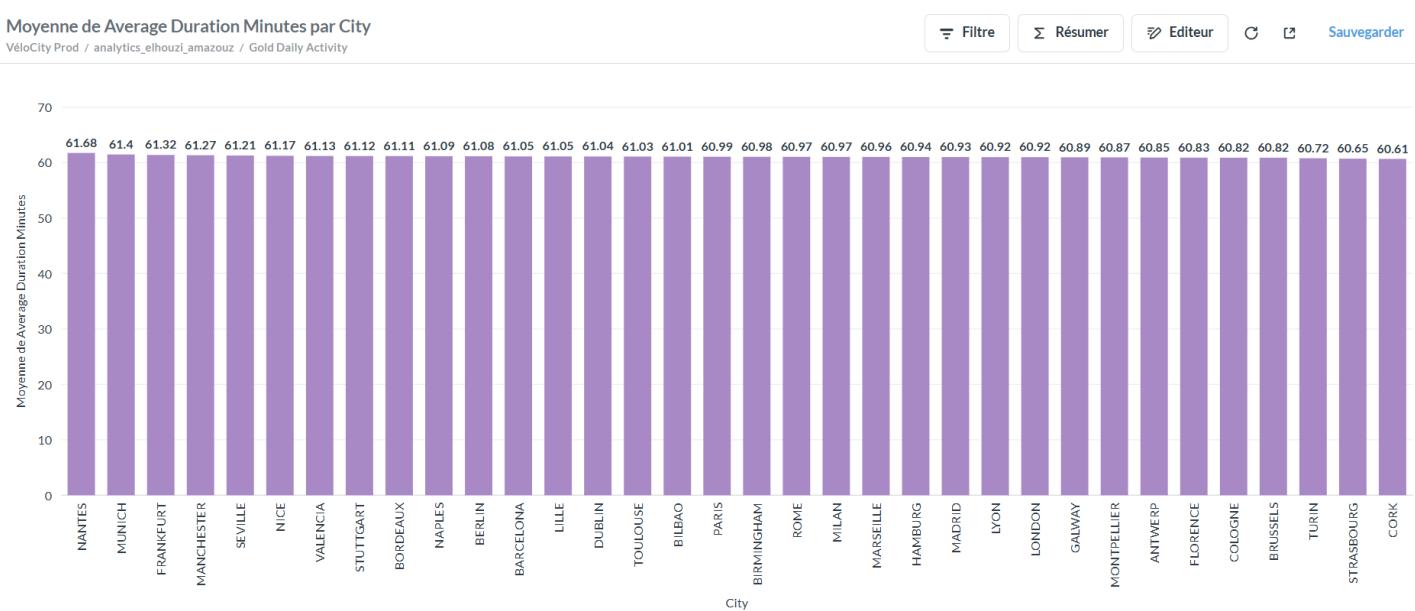
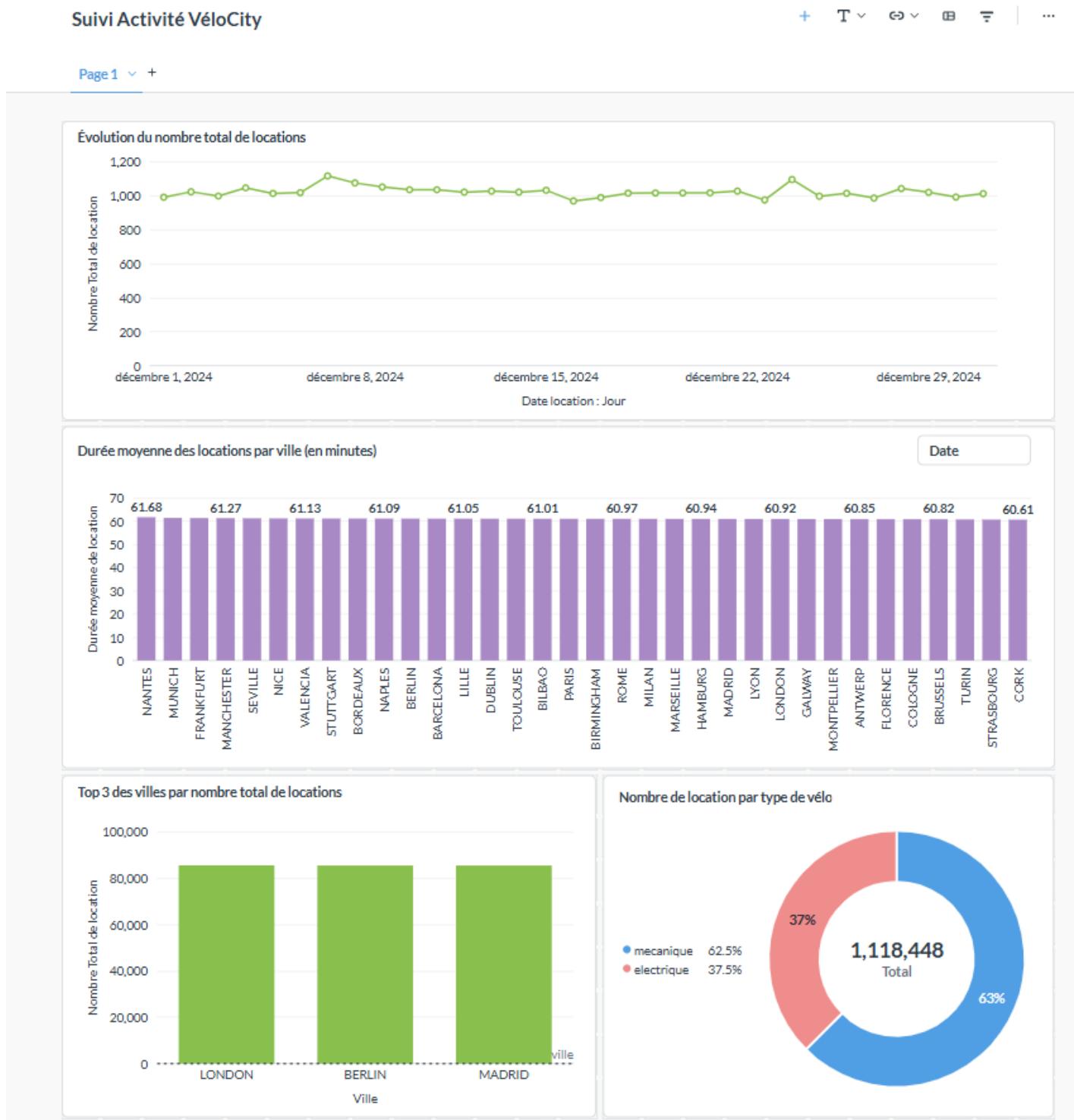


Chart 3 : Analyse de la durée moyenne des trajets par ville



C. Création du Dashboard Final :

La figure ci-dessous présente le dashboard final de suivi de l'activité VéloCity, illustrant l'évolution du nombre total de locations dans le temps, la durée moyenne des locations par ville, le top 3 des villes les plus actives ainsi que la répartition des locations par type de vélo.



IV. Sécurité et Gouvernance (PostgreSQL +) :

Objectif : Mettre en place une sécurité basée sur le moindre privilège pour deux rôles :

- **marketing_user** : accès uniquement en lecture à la couche Gold.
- **manager_lyon** : accès uniquement aux lignes correspondant à city = 'lyon' grâce à une politique RLS (Row Level Security).

A. Création des rôles / utilisateurs :

```

Query  Query History
1 CREATE ROLE marketing_user LOGIN PASSWORD 'Epsi##25';
2 CREATE ROLE manager_lyon LOGIN PASSWORD 'Epsi##25';
3

Data Output  Messages  Notifications
CREATE ROLE

Query returned successfully in 68 msec.

```

B. Audit (Simulation) :

Dans cette étape, nous testons le comportement du système lorsque l'utilisateur **marketing_user** tente d'accéder à différentes tables, notamment une table de la couche **raw** et la table **Gold** agrégée. Pour simuler cette situation, nous utilisons la commande suivante afin de se connecter en tant que **marketing_user** sans avoir à changer de session.

```
SET ROLE marketing_user;
```

Test1: **marketing_user** exécute:

```
SELECT * FROM raw.user_accounts;
```

```

5
6 SELECT * FROM raw.user_accounts LIMIT 5;
7
8 RESET ROLE;
9

Data Output  Messages  Notifications
ERROR: permission denied for schema raw
LINE 3: SELECT * FROM raw.user_accounts LIMIT 5;
^

SQL state: 42501
Character: 41

```

Résultat : accès interdit Erreur n'en peut pas accéder à la table **user_accounts** de la couche **raw**

Explication :

- La couche *raw* contient des **données sensibles (PII)**.
- L'utilisateur *marketing_user* ne doit **en aucun cas** avoir accès à cette couche.
- Cette restriction respecte le **principe du moindre privilège**, selon lequel un utilisateur ne dispose que des droits strictement nécessaires à ses besoins.

Test 2: Si *marketing_user* exécute:

```
SELECT * FROM analytics_elhouzi_amazouz.gold_daily_activity;
```

```
31 SET ROLE marketing_user;
32 SELECT * FROM analytics_elhouzi_amazouz.gold_daily_activity;
33
```

ata Output Messages Notifications

The screenshot shows a PostgreSQL client interface with the following details:

- SQL tab:** The user is currently viewing the SQL tab.
- Query:** The query executed is `SELECT * FROM analytics_elhouzi_amazouz.gold_daily_activity;`.
- Results:** The results show data from the `gold_daily_activity` table. The columns and their data types are:

rental_date	city	start_station	bike_type	sub_name	total_rentals	average_duration_minutes	unique_users
date	text	text	text	character varying	bigint	numeric	bigint

 The data rows are:

2025-04-22	MARSEILLE	Castellane 2	electrique	Étudiant	2	71.85	2
2025-04-22	MARSEILLE	Castellane 2	electrique	Mensuel	1	119.45	1
2025-04-22	MARSEILLE	Castellane 2	electrique	Pay-as-you-go	1	54.60	1
2025-04-22	MARSEILLE	Castellane 2	mecanique	Pay-as-you-go	1	31.68	1
2025-04-22	MARSEILLE	Joliette	mecanique	Gratuit	1	66.53	1
2025-04-22	MARSEILLE	Joliette	mecanique	Mensuel	1	113.88	1
2025-04-22	MARSEILLE	La Timone	electrique	Premium Plus	1	71.35	1
2025-04-22	MARSEILLE	La Timone	mecanique	Annuel	1	63.90	1
2025-04-22	MARSEILLE	La Timone	mecanique	Gratuit	3	78.41	3

Résultat : L'utilisateur **marketing_user** a été autorisé à lire la table analytique **gold_daily_activity** contenue dans le schéma **analytics_elhouzi_amazouz**.

Explication :

- La table **gold_daily_activity** appartient au schéma **analytics_elhouzi_amazouz**, qui correspond à la couche GOLD du Data Warehouse.
- Cette couche contient uniquement des données : nettoyées – agrégées – analytiques - et ne contenant plus d'informations sensibles.

C. Tâche (Script SQL) :

Les commandes ci-dessous permettent d'appliquer la règle de sécurité définie :

a. Révocation des accès par défaut :

```
REVOKE ALL ON SCHEMA raw FROM marketing_user;
REVOKE ALL ON ALL TABLES IN SCHEMA raw FROM marketing_user;
```

Explication :

- Les tables et schémas **raw** et **silver** contiennent respectivement les données brutes et intermédiaires.
- Ces commandes garantissent que **marketing_user** ne peut ni lire, ni modifier ces données sensibles.

b. Accès limité au schéma Gold :

```
GRANT USAGE ON SCHEMA analytics_elhouzi_amazouz TO marketing_user;

GRANT SELECT ON analytics_elhouzi_amazouz.gold_daily_activity TO marketing_user;
```

Explication :

- GRANT USAGE : permet au rôle de voir le schéma et de naviguer dedans.
- GRANT SELECT : autorise uniquement la lecture de la table Gold.
- Résultat : Marketing peut lire uniquement les données agrégées nécessaires pour le Dashboard.

c. Gestion des futurs objets Gold :

✓ **ALTER DEFAULT PRIVILEGES IN SCHEMA analytics_elhouzi_amazouz
GRANT SELECT ON TABLES TO marketing_user;**

Explication :

- Automatisation : toute nouvelle table Gold créée dans ce schéma sera accessible en lecture par **marketing_user**.
- Évite de devoir refaire manuellement les GRANT à chaque nouvelle table.

d. Vérification et tests :

```
--  
31 SET ROLE marketing_user;  
32 SELECT * FROM analytics_elhouzi_amazouz.gold_daily_activity;  
33  
34
```

Data Output Messages Notifications

	rental_date	city	start_station	bike_type	sub_name	total_rentals	average_duration_minutes	unique_users
	date	text	text	text	character varying	bigint	numeric	bigint
1	2025-04-22	MARSEILLE	Castellane 2	electrique	Étudiant	2	71.85	2
2	2025-04-22	MARSEILLE	Castellane 2	electrique	Mensuel	1	119.45	1
3	2025-04-22	MARSEILLE	Castellane 2	electrique	Pay-as-you-go	1	54.60	1
4	2025-04-22	MARSEILLE	Castellane 2	mecanique	Pay-as-you-go	1	31.68	1
5	2025-04-22	MARSEILLE	Joliette	mecanique	Gratuit	1	66.53	1
6	2025-04-22	MARSEILLE	Joliette	mecanique	Mensuel	1	113.88	1
7	2025-04-22	MARSEILLE	La Timone	electrique	Premium Plus	1	71.35	1
8	2025-04-22	MARSEILLE	La Timone	mecanique	Annuel	1	63.90	1
9	2025-04-22	MARSEILLE	La Timone	mecanique	Gratuit	3	78.41	3

Explication :

- Ces tests permettent de vérifier que les priviléges sont correctement appliqués.
- Marketing ne voit que les données autorisées, et aucune information sensible des utilisateurs.

D. Row-level Security :

Durant cette étape notre objectif est de Mettre en place une sécurité au niveau des lignes pour le rôle manager_lyon, afin qu'il ne puisse consulter que les données correspondant à la ville “Lyon” dans la table GOLD

a. Donne l'accès au schéma et à la table GOLD :

```
GRANT USAGE ON SCHEMA analytics_elhouzi_amazouz TO manager_lyon;  
  
GRANT SELECT ON analytics_elhouzi_amazouz.gold_daily_activity TO manager_lyon;
```

Explication :

- USAGE permet de naviguer dans le schéma.
- SELECT autorise uniquement la consultation des données, sans modification.

b. Activation du Row-level Security sur la table :

```
47  
48 ALTER TABLE analytics_elhouzi_amazouz.gold_daily_activity ENABLE ROW LEVEL SECURITY;  
49  
50  
51  
52  
53  
54
```

Data Output Messages Notifications

```
ALTER TABLE  
  
Query returned successfully in 77 msec.
```

Explication :

- La RLS bloque par défaut tout accès aux lignes pour les rôles qui ne sont pas superuser.
- Il faut ensuite créer une policy spécifique pour autoriser certaines lignes.

c. Création de la policy RLS pour lyon :

```

52 ✓ CREATE POLICY lyon_only_policy
53   ON analytics_elhouzi_amazouz.gold_daily_activity
54   FOR SELECT
55   TO manager_lyon
56   USING (city_name = 'Lyon');
57
Data Output Messages Notifications
CREATE POLICY
Query returned successfully in 70 msec.

```

Explication :

- FOR SELECT : s'applique uniquement aux lectures.
- TO manager_lyon : cible le rôle concerné.
- USING (city_name = 'Lyon') : filtre les lignes visibles par ce rôle.

d. Test du rôle :

```

412  SELECT * FROM analytics_elhouzi_amazouz.gold_daily_activity;
413
414
Data Output Messages Notifications
SQL All rows on this page are selected. Select All 45047 Rows

```

	rental_date	city	start_station	bike_type	sub_name	total_rentals	average_duration_minutes	unique_users
	date	text	text	text	character varying	bigint	numeric	bigint
1	2025-04-23	LYON	Bellecour	électrique	Mensuel	1	50.43	1
2	2025-04-23	LYON	Bellecour	mécanique	Étudiant	1	85.27	1
3	2025-04-23	LYON	Bellecour 2	électrique	Étudiant	1	93.98	1
4	2025-04-23	LYON	Bellecour 2	électrique	Gratuit	1	117.08	1
5	2025-04-23	LYON	Bellecour 2	mécanique	Étudiant	1	20.10	1
6	2025-04-23	LYON	Bellecour 2	mécanique	Pay-as-you-go	1	99.45	1
7	2025-04-23	LYON	Bellecour 2	mécanique	Premium Plus	2	79.04	2
8	2025-04-23	LYON	Charpennes	électrique	Gratuit	1	64.48	1
9	2025-04-23	LYON	Charpennes	mécanique	Premium Plus	1	98.80	1
10	2025-04-23	LYON	Confluence	électrique	Mensuel	1	9.75	1

Résultat :

- Seules les lignes où city = 'Lyon' apparaissent.
- Aucune autre ville n'est visible par **manager_lyon**.