14 novembre 2025

BASE RELATIONNELLE POSTGRESQL

MODÉLISER L'ÉCOSYSTÈME DE LA RECHERCHE

ÉTUDIANTS :

Younes SHIMI
Meldi AHISSOU

ENSEIGNANTE

EVELYNE VITTORI

Sommaire

Introduction

Ce premier travail pratique s'inscrit dans le cadre du projet fil rouge intitulé « Écosystème de la recherche : de la modélisation relationnelle à l'entrepôt de données ». Il constitue la première étape d'un ensemble de trois travaux successifs visant à modéliser, stocker et analyser les données liées à la recherche universitaire.

L'objectif principal de ce TP1 est de concevoir et implémenter une base de données relationnelle sous PostgreSQL permettant de gérer l'ensemble des informations relatives aux institutions, laboratoires, chercheurs, projets, contrats, publications et jeux de données.
Cette base servira de fondation pour les prochaines étapes du projet, notamment l'intégration des données scientifiques dans MongoDB (TP2) et la création d'un entrepôt de données décisionnel (TP3).
Ce travail met en œuvre plusieurs compétences clés :

- la conception d'un schéma relationnel cohérent à l'aide d'outils d'IA générative ;
- le peuplement automatique de la base à l'aide de Python Faker ;
- la gestion sécurisée des accès par rôles, vues et privilèges ;
- l'optimisation des requêtes SQL pour améliorer les performances ;
- et la programmation PL/pgSQL à travers la création de triggers et procédures stockées.

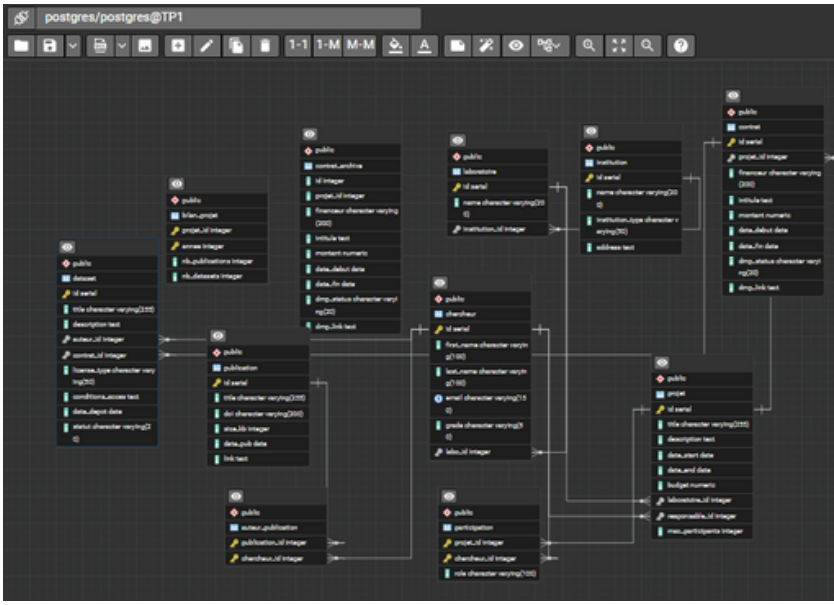
L'ensemble de ces étapes vise à démontrer la capacité à concevoir une base de données complète, robuste et performante, tout en adoptant une démarche critique face aux outils d'intelligence artificielle utilisés dans le processus de conception.

- Ce travail est accompagné d'un dépôt GitHub contenant l'ensemble des scripts SQL, procédures et données générées :

https://github.com/youneshimi/TP1_BD_Projet.git

Partie 1 - Conception de la Base de Données

MODÈLE CONCEPTUEL DE DONNÉE (MCD):



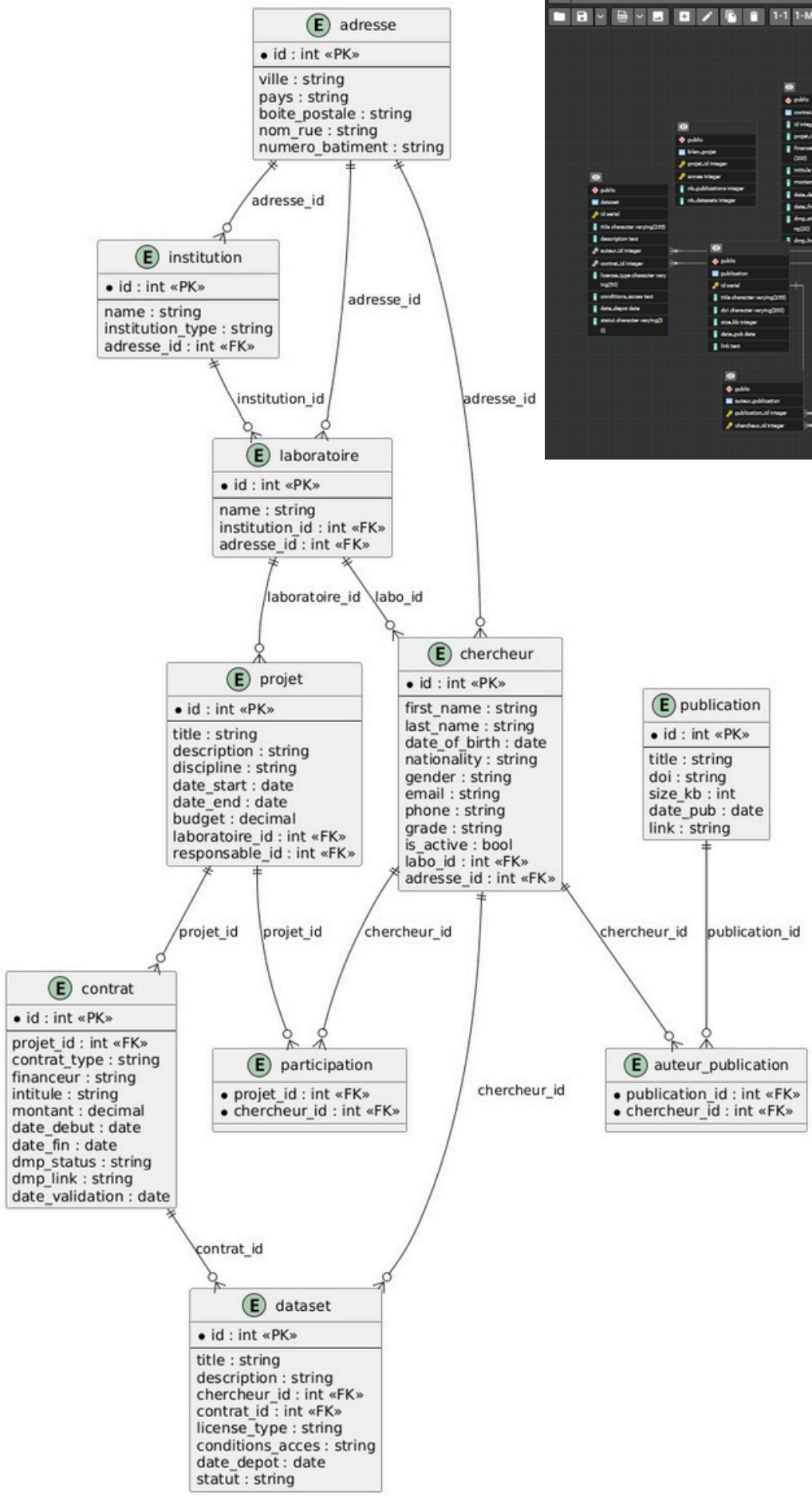


Schéma relationnel :

- ADRESSE (id, ville, pays, boite_postale, nom_rue, numero_batiment)
- INSTITUTION (id, name, institution_type, **#adresse_id**)
- LABORATOIRE (id, name, **#institution_id**, **#adresse_id**)
- CHERCHEUR(id , first_name, last_name, date_of_birth, nationality, gender, email, phone, grade, is_active, **#labo_id** , **#adresse_id**)
- PROJET(id, title, description, discipline, date_start, date_end, budget, **#laboratoire_id**, **#responsable_id**)
- PARTICIPATION (**#projet_id**, **#chercheur_id**)
- CONTRAT (id, **#projet_id**, contrat_type ,financeur, intitule, montant, date_debut, date_fin, dmp_status, dmp_link, date_validation)
- DATASET (id , title, description, **#chercheur_id**, **#contrat_id**, license_type, conditions_acces, date_depot, statut)
- PUBLICATION(id, title, doi, size_kb, date_pub, link)
- AUTEUR_PUBLICATION (**#publication_id, #chercheur_id**)

Le schéma relationnel modélise l'écosystème de la recherche scientifique en reliant les institutions, laboratoires, chercheurs, projets, contrats, publications et jeux de données.
Chaque table représente une entité principale avec ses relations logiques :

- **ADRESSE** : centralise les informations géographiques.
- **INSTITUTION** et **LABORATOIRE** : décrivent les structures de recherche et leurs localisations.
- **CHERCHEUR** : contient les informations personnelles et professionnelles des chercheurs.
- **PROJET** : regroupe les projets de recherche avec leur responsable et laboratoire associé.
- **PARTICIPATION** : gère la relation N-N entre chercheurs et projets.
- **CONTRAT** : décrit les contrats de financement liés aux projets.
- **DATASET** : représente les jeux de données produits dans le cadre des contrats.
- **PUBLICATION** et **AUTEUR_PUBLICATION** : gèrent les publications scientifiques et leurs auteurs.

Utilisation des outils d'IA:

Dans le cadre de la création de notre base de données, nous avons élaboré manuellement le Modèle Conceptuel de Données (MCD) ainsi que le schéma relationnel, en nous appuyant sur le cahier des charges pour définir les entités, les relations et les contraintes principales. L'intelligence artificielle, en l'occurrence **Perplexity**, n'a été utilisée que pour la création du schéma côté visualisation, notamment pour générer l'image du MCD à l'aide de PlantUML. Cette approche nous a permis de conserver une maîtrise complète de la conception tout en bénéficiant d'une aide automatisée pour la représentation graphique.

Partie 3 - Gestion des Utilisateurs et Création de Vues

Création des Rôles (CreateRole.sql)**:**

```
DO
$$
BEGIN
    IF NOT EXISTS (SELECT FROM pg_roles WHERE rolname = 'chercheur') THEN
        CREATE ROLE chercheur LOGIN PASSWORD 'chercheur@123';
    END IF;

    IF NOT EXISTS (SELECT FROM pg_roles WHERE rolname = 'datamanager') THEN
        CREATE ROLE datamanager LOGIN PASSWORD 'datamanager@123';
    END IF;

    IF NOT EXISTS (SELECT FROM pg_roles WHERE rolname = 'admin_bd') THEN
        CREATE ROLE admin_bd LOGIN PASSWORD 'admin@123';
    END IF;
END
$$;
```

Ce script vérifie si les rôles "**chercheur**", "**datamanager**" et "**admin_bd**" existent dans PostgreSQL et les crée automatiquement avec un mot de passe si besoin. Cela permet d'automatiser et de sécuriser la gestion des comptes utilisateurs dans la base de données, tout en évitant les doublons et les erreurs lors de la réexécution du script.

Création des vues : (les Scripts sont dans le dossier Partie3/CreateVues.sql du ZIP)

1- Liste des informations liées aux projets d'un chercheur:

Cette vue affiche la liste des chercheurs et des laboratoires auxquels ils sont associés.

	chercheur_id integer	first_name character varying (100)	last_name character varying (100)	email character varying (150)	phone character varying (20)	grade character varying (50)	is_active boolean	date_of_ date
1	1	Philippine	David	fvallet@example.com	+33 (0)4 44 96 82 11	MaitreConf	true	1970-10-
2	2	Adèle	Perrin	noel43@example.com	0554972885	Doctorant	true	1993-08-
3	3	Martine	Dumas	cousinsabine@example.net	+33 4 67 05 16 50	Professeur	true	1980-06-
4	4	Clémence	Rossi	maggiedevaux@example.net	+33 3 65 77 98 82	Professeur	false	1969-02-
5	5	Isaac	Fouquet	lefortlouise@example.org	+33 7 80 72 25 08	MaitreConf	false	1973-03-
6	6	Andrée	Baudry	uribeiro@example.com	0487689704	MaitreConf	false	1996-01-
7	7	Charles	Daniel	laurencebrun@example.net	+33 (0)3 10 17 05 55	Professeur	false	1969-08-
8	8	Julien	Léger	constancegaillard@example....	+33 2 90 74 53 09	MaitreConf	true	1991-01-
9	9	Alexandre	Marion	cvoisin@example.org	0153243684	MaitreConf	true	1998-07-
10	10	Nathalie	Wagner	jeanpires@example.com	02 54 92 23 94	Doctorant	true	1960-06-
11	11	Dorothée	Hamel	marcellesage@example.org	+33 5 94 35 13 99	Professeur	false	1960-07-
12	12	Aimée	Weiss	lefebvrenathalie@example.co...	+33 (0)5 24 18 70 08	Doctorant	true	1980-02-

2- Liste des publications et de leurs auteurs:

Cette vue affiche la liste des informations en rapport avec une publication.

iblication_id teger	publication_titre character varying (255)	date_pub date	doi character varying (200)	auteur_nom text	laboratoire character varying (200)	lien text
1	Digne ce marquer espèce très français.	2023-12-03	3e3ceee1-532c-4bd7-a2c9-1ec0b6aa22...	Dorothée Hamel	Labo Avenir	http://brun.fr/
1	Digne ce marquer espèce très français.	2023-12-03	3e3ceee1-532c-4bd7-a2c9-1ec0b6aa22...	Clémence Rossi	Labo Avenir	http://brun.fr/
1	Digne ce marquer espèce très français.	2023-12-03	3e3ceee1-532c-4bd7-a2c9-1ec0b6aa22...	Suzanne Roux	Labo Moyen	http://brun.fr/
2	Expression perdre espèce.	2024-08-26	7e66bd38-36bf-4b11-838b-076e3914ce...	Charles Techer	Labo Endormir	https://mahe.fr/
2	Expression perdre espèce.	2024-08-26	7e66bd38-36bf-4b11-838b-076e3914ce...	Danielle Fernan...	Labo Endormir	https://mahe.fr/
2	Expression perdre espèce.	2024-08-26	7e66bd38-36bf-4b11-838b-076e3914ce...	René Rodrigues	Labo Moyen	https://mahe.fr/
2	Expression perdre espèce.	2024-08-26	7e66bd38-36bf-4b11-838b-076e3914ce...	Paulette Lévy	Labo Moyen	https://mahe.fr/
3	Précis fou appeler endroit prêter moment.	2025-08-07	60b075d9-ef47-4e27-a1dc-3ce049f96f6c	Christophe Per...	Labo Moyen	https://www.san
3	Précis fou appeler endroit prêter moment.	2025-08-07	60b075d9-ef47-4e27-a1dc-3ce049f96f6c	Robert Berger	Labo Occasion	https://www.san
3	Précis fou appeler endroit prêter moment.	2025-08-07	60b075d9-ef47-4e27-a1dc-3ce049f96f6c	Pierre Fouquet	Labo Moyen	https://www.san

3:Informations des contrats et datasets des chercheurs du même projet

Cette vue affiche les informations relatives à un contrat de projets et les informations des datasets qui y sont liés

	chercheur_nom text	projet_titre character varying (255)	contrat_id integer	financeur character varying (200)	contrat_type character varying (50)	montant numeric	dataset_id integer	dataset_titre character varying (255)	license_ charact
1	Claude Rocher	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
2	Marcelle Jacquet	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
3	Paulette Prévost	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
4	Guy Dupré	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
5	Raymond Mary	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
6	Christophe Robin	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
7	Théophile Vailla...	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
8	Lucie Guillou	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
9	Suzanne Roux	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
10	Luc Barbier	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte
11	Suzanne Besson	Projet Structurant Accent	68	Royer Boucher et Fils	H2020	60616	1	Dataset Projet 1	restricte

4:Informations complètes sur un chercheur

chercheur_id integer	first_name character varying (100)	last_name character varying (100)	email character varying (150)	phone character varying (20)	grade character varying (50)	is_active boolean	date_of_birt date
1	Philippine	David	fvallet@example.com	+33 (0)4 44 96 82 11	MaitreConf	true	1970-10-07
2	Adèle	Perrin	noel43@example.com	0554972885	Doctorant	true	1993-08-03
3	Martine	Dumas	cousinsabine@example.net	+33 4 67 05 16 50	Professeur	true	1980-06-20
4	Clémence	Rossi	maggiedevaux@example.net	+33 3 65 77 98 82	Professeur	false	1969-02-21
5	Isaac	Fouquet	lefortlouise@example.org	+33 7 80 72 25 08	MaitreConf	false	1973-03-24
6	Andrée	Baudry	uribeiro@example.com	0487689704	MaitreConf	false	1996-01-21
7	Charles	Daniel	laurencebrun@example.net	+33 (0)3 10 17 05 55	Professeur	false	1969-08-14
8	Julien	Léger	constancegaillard@example....	+33 2 90 74 53 09	MaitreConf	true	1991-01-17
9	Alexandre	Marion	cvoisin@example.org	0153243684	MaitreConf	true	1998-07-14
10	Nathalie	Wagner	jeanpires@example.com	02 54 92 23 94	Doctorant	true	1960-06-05
11	Dorothée	Hamel	marcellesage@example.org	+33 5 94 35 13 99	Professeur	false	1960-07-13

5:Vue de supervision administrative (ADMIN)

Cette vue affiche les statistiques permettant à l'administrateur d'avoir une vision globale des informations de la base de données.

	projet character varying (255)	nb_chercheurs bigint	nb_datasets bigint	nb_publications bigint	total_budget numeric
1	Projet Structurant Accent	30	159	233	134833968480
2	Projet Structurant Affirmer	51	100	233	130682520105
3	Projet Structurant Avant	32	57	233	50285612640
4	Projet Structurant Commen...	49	104	233	142620330559
5	Projet Structurant Enlever	42	93	233	58916847864
6	Projet Structurant Mériter	41	128	233	106420868991
7	Projet Structurant Posséder	42	105	233	83223950754
8	Projet Structurant Résister	36	103	233	87228078588
9	Projet Supérieur	38	59	161	48434217650
10	Projet Tantôt	39	92	161	57287241375

Création des PRIVILÈGES : (les Scripts sont dans le dossier Partie3/Privileges_vues duZIP)

Rôle	Description fonctionnelle	Vues accessibles	Droits sur tables
Chercheur	Accès restreint à ses projets et données personnelles	vue_projets_chercheur, vue_publications_auteurs, vue_infos_chercheur	Lecture seule
Data Manager	Accès élargi aux métadonnées et supervision	toutes les vues (5)	Lecture seule
Admin BD	Accès complet à toutes les structures et données	toutes les vues	Lecture + écriture + gestion

Partie 4 - Requêtes et optimisation

(Script disponible dans le projet rendu)

- Comparaison des performances

		R1	M.V	R2	M.V	R3	M.V
Script 1	***Plan d'exécution***	19.999 ms		29.261 ms		0.470 ms	
	Temps d'exécution réel	7.510 ms		0.458 ms		0.370 ms	
Script 2	***Plan d'exécution***	0.788 ms	**2**	1.756 ms	**2**	0.363 ms	**2**
	Temps d'exécution réel	3.202 ms		7.104 ms		0.765 ms	

Tab1 : Tableau comparatif des requêtes (*M.V = Meilleur Version)*

Requête 1 : (Script 1)

```sql
SELECT
    p.id AS projet_id,
    p.title AS projet_titre,
    EXTRACT(YEAR FROM d.date_depot) AS annee_depot,
    COUNT(d.id) AS nb_datasets,
    ROUND(AVG(d.date_depot - c.date_debut), 2) AS delai_moyen_jours
FROM projet p
JOIN contrat c ON c.projet_id = p.id
JOIN dataset d ON d.contrat_id = c.id
WHERE p.id IN (
    SELECT projet_id
    FROM participation
    GROUP BY projet_id
    HAVING COUNT(chercheur_id) > 5
)
AND d.date_depot >= '2018-01-01'
GROUP BY p.id, p.title, EXTRACT(YEAR FROM d.date_depot)
ORDER BY annee_depot DESC, nb_datasets DESC;
```

```
QUERY PLAN
text
Rows Removed by Filter: 1
-> Hash (cost=19.11..19.11 rows=30 width=528) (actual time=4.893..4.898 rows=100 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 16kB
-> Nested Loop (cost=8.31..19.11 rows=30 width=528) (actual time=4.530..4.827 rows=100 loops=1)
-> Hash Join (cost=8.16..11.54 rows=30 width=16) (actual time=3.842..3.937 rows=100 loops=1)
Hash Cond: (c.projet_id = participation.projet_id)
-> Seq Scan on contrat c (cost=0.00..3.00 rows=100 width=12) (actual time=0.007..0.027 rows=100 loops=1)
-> Hash (cost=8.12..8.12 rows=3 width=4) (actual time=3.785..3.786 rows=10 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 9kB
-> HashAggregate (cost=8.00..8.12 rows=3 width=4) (actual time=3.759..3.764 rows=10 loops=1)
Group Key: participation.projet_id
Filter: (count(participation.chercheur_id) > 5)
Batches: 1 Memory Usage: 24kB
-> Seq Scan on participation (cost=0.00..6.00 rows=400 width=8) (actual time=3.481..3.577 rows=400 loops=1)
-> Memoize (cost=0.15..0.65 rows=1 width=520) (actual time=0.008..0.008 rows=1 loops=100)
Cache Key: c.projet_id
Cache Mode: logical
Hits: 90 Misses: 10 Evictions: 0 Overflows: 0 Memory Usage: 2kB
-> Index Scan using projet_pkey on projet p (cost=0.14..0.64 rows=1 width=520) (actual time=0.072..0.072 rows=1 loo...
Index Cond: (id = c.projet_id)
Planning Time: 19.999 ms
Execution Time: 7.510 ms
```

Requête 1 : (Script 2)

```
QUERY PLAN
text
Rows Removed by Filter: 1
-> Hash (cost=19.11..19.11 rows=3 width=528) (actual time=0.572..0.577 rows=100 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 15kB
-> Nested Loop (cost=8.31..19.11 rows=3 width=528) (actual time=0.314..0.519 rows=100 loops=1)
-> Hash Join (cost=8.16..11.54 rows=30 width=16) (actual time=0.287..0.375 rows=100 loops=1)
Hash Cond: (c.projet_id = participation.projet_id)
-> Seq Scan on contrat c (cost=0.00..3.00 rows=100 width=12) (actual time=0.010..0.028 rows=100 loops=1)
-> Hash (cost=8.12..8.12 rows=3 width=4) (actual time=0.265..0.267 rows=10 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 9kB
-> HashAggregate (cost=8.00..8.12 rows=3 width=4) (actual time=0.255..0.259 rows=10 loops=1)
Group Key: participation.projet_id
Filter: (count(participation.chercheur_id) > 5)
Batches: 1 Memory Usage: 24kB
-> Seq Scan on participation (cost=0.00..6.00 rows=400 width=8) (actual time=0.011..0.063 rows=400 loops=1)
-> Memoize (cost=0.15..0.65 rows=1 width=520) (actual time=0.001..0.001 rows=1 loops=100)
Cache Key: c.projet_id
Cache Mode: logical
Hits: 90 Misses: 10 Evictions: 0 Overflows: 0 Memory Usage: 2kB
-> Index Scan using projet_pkey on projet p (cost=0.14..0.64 rows=1 width=520) (actual time=0.003..0.004 rows=1 loo...
Index Cond: (id = c.projet_id)
Planning Time: 0.788 ms
Execution Time: 3.202 ms
```

```sql
WITH projets_valides AS (
    SELECT projet_id
    FROM participation
    GROUP BY projet_id
    HAVING COUNT(chercheur_id) > 5
)
SELECT
    p.id AS projet_id,
    p.title AS projet_titre,
    EXTRACT(YEAR FROM d.date_depot) AS annee_depot,
    COUNT(d.id) AS nb_datasets,
    ROUND(AVG(d.date_depot - c.date_debut), 2) AS delai_moyen_jours
FROM dataset d
JOIN contrat c ON d.contrat_id = c.id
JOIN projet p ON p.id = c.projet_id
JOIN projets_valides pv ON pv.projet_id = p.id
WHERE d.date_depot >= '2018-01-01'
GROUP BY p.id, p.title, EXTRACT(YEAR FROM d.date_depot)
ORDER BY annee_depot DESC, nb_datasets DESC;
```

Requête 2 : (Script 1)

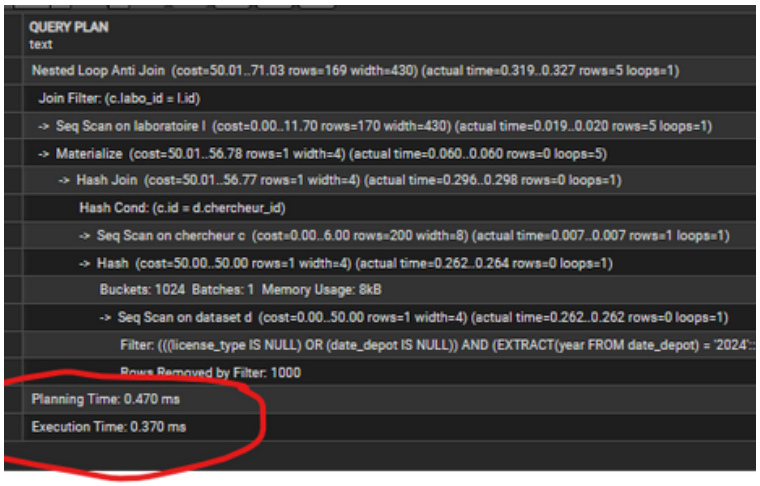
```
QUERY PLAN
text
Index Cond: ((publication_id = pub.id) AND (chercheur_id = ch.id))
Heap Fetches: 0
SubPlan 2
-> Aggregate (cost=26.19..26.20 rows=1 width=32) (never executed)
-> GroupAggregate (cost=26.15..26.17 rows=1 width=12) (never executed)
Group Key: c2.id
-> Sort (cost=26.15..26.16 rows=1 width=8) (never executed)
Sort Key: c2.id
-> Nested Loop (cost=0.42..26.14 rows=1 width=8) (never executed)
-> Nested Loop (cost=0.28..25.22 rows=5 width=8) (never executed)
-> Seq Scan on publication pub2 (cost=0.00..16.50 rows=2 width=4) (never executed)
Filter: (EXTRACT(year FROM date_pub) = '2024'::numeric)
-> Index Only Scan using auteur_publication_pkey on auteur_publication ap2 (cost=0.28..4.33 rows=3 width=8) (never exe...
Index Cond: (publication_id = pub2.id)
Heap Fetches: 0
-> Index Scan using chercheur_pkey on chercheur c2 (cost=0.14..0.18 rows=1 width=4) (never executed)
Index Cond: (id = ap2.chercheur_id)
Filter: (labo_id = l.id)
-> Index Scan using chercheur_pkey on chercheur ch_resp (cost=0.14..0.40 rows=1 width=18) (never executed)
Index Cond: (id = p.responsable_id)
Planning Time: 29.261 ms
Execution Time: 0.458 ms
Total rows: 50 Query complete 00:00:00.861
```

```sql
SELECT
    p.title AS projet_titre,
    ch_resp.first_name || ' ' || ch_resp.last_name AS responsable
FROM projet p
JOIN chercheur ch_resp ON ch_resp.id = p.responsable_id
JOIN laboratoire l ON p.laboratoire_id = l.id
WHERE l.name = 'LISA'
  AND NOT EXISTS (
      SELECT 1
      FROM participation pa
      JOIN chercheur ch ON ch.id = pa.chercheur_id
      WHERE pa.projet_id = p.id
        AND ch.labo_id = l.id
        AND (
            SELECT COUNT(ap.publication_id)
            FROM auteur_publication ap
            JOIN publication pub ON pub.id = ap.publication_id
            WHERE ap.chercheur_id = ch.id
              AND EXTRACT(YEAR FROM pub.date_pub) = 2024
        ) < (
            SELECT AVG(pub_count)
            FROM (
                SELECT COUNT(ap2.publication_id) AS pub_count
                FROM chercheur c2
                JOIN auteur_publication ap2 ON ap2.chercheur_id = c2.id
                JOIN publication pub2 ON pub2.id = ap2.publication_id
                WHERE c2.labo_id = l.id
                  AND EXTRACT(YEAR FROM pub2.date_pub) = 2024
                GROUP BY c2.id
            ) AS moy_lab
        )
```

Requête 2 : (Script 2)

```
QUERY PLAN
text
Hash Cond: (pa.chercheur_id = p24.chercheur_id)
-> Seq Scan on participation pa (cost=0.00..6.00 rows=400 width=8) (actual time=0.011..0.090 rows=400 loops=1)
-> Hash (cost=0.36..0.36 rows=2 width=4) (actual time=6.632..6.635 rows=109 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 12kB
-> Hash Join (cost=0.25..0.36 rows=2 width=4) (actual time=6.200..6.577 rows=109 loops=1)
Hash Cond: (p24.labo_id = pub_chercheur_2024.labo_id)
Join Filter: ((p24.nb_pub_2024)::numeric < (avg(pub_chercheur_2024.nb_pub_2024)))
Rows Removed by Join Filter: 84
-> CTE Scan on pub_chercheur_2024 p24 (cost=0.00..0.10 rows=5 width=16) (actual time=5.552..5.586 rows=193 loops=1)
-> Hash (cost=0.19..0.19 rows=5 width=36) (actual time=0.599..0.600 rows=5 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 9kB
```

Requête 3 : (Script 1):

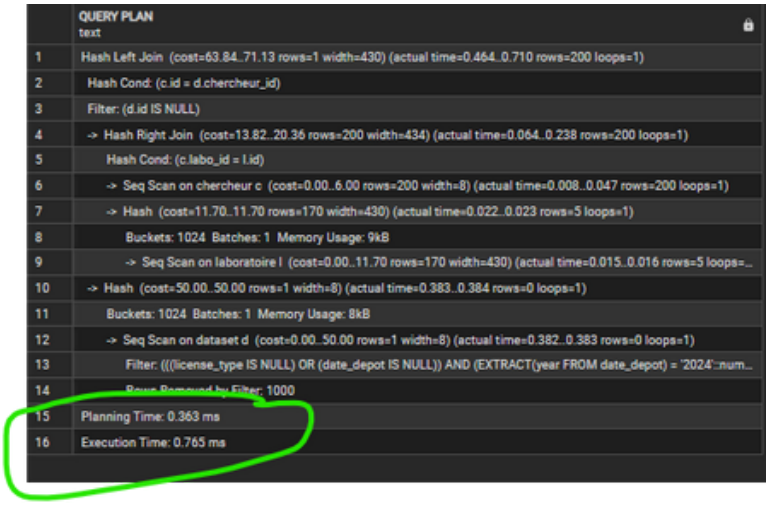


QUERY PLAN text
Nested Loop Anti Join (cost=50.01..71.03 rows=169 width=430) (actual time=0.319..0.327 rows=5 loops=1)
Join Filter: (c.labo_id = l.id)
-> Seq Scan on laboratoire l (cost=0.00..11.70 rows=170 width=430) (actual time=0.019..0.020 rows=5 loops=1)
-> Materialize (cost=50.01..56.78 rows=1 width=4) (actual time=0.060..0.060 rows=0 loops=5)
-> Hash Join (cost=50.01..56.77 rows=1 width=4) (actual time=0.296..0.298 rows=0 loops=1)
Hash Cond: (c.id = d.chercheur_id)
-> Seq Scan on chercheur c (cost=0.00..6.00 rows=200 width=8) (actual time=0.007..0.007 rows=1 loops=1)
-> Hash (cost=50.00..50.00 rows=1 width=4) (actual time=0.262..0.264 rows=0 loops=1)
Buckets: 1024 Batches: 1 Memory Usage: 8kB
-> Seq Scan on dataset d (cost=0.00..50.00 rows=1 width=4) (actual time=0.262..0.262 rows=0 loops=1)
Filter: (((license_type IS NULL) OR (date_depot IS NULL)) AND (EXTRACT(year FROM date_depot) = '2024'::...
Rows Removed by Filter: 1000
Planning Time: 0.470 ms
Execution Time: 0.370 ms

```sql
SELECT l.*
FROM laboratoire l
WHERE NOT EXISTS (
  SELECT 1
  FROM chercheur c
  JOIN dataset d ON d.chercheur_id = c.id
  WHERE c.labo_id = l.id
    AND (
      d.license_type IS NULL OR d.date_depot IS NULL
    )
    AND EXTRACT(YEAR FROM d.date_depot) = 2024
);

```

Requête 3 : (Script 2)



	QUERY PLAN text
1	Hash Left Join (cost=63.84..71.13 rows=1 width=430) (actual time=0.464..0.710 rows=200 loops=1)
2	Hash Cond: (c.id = d.chercheur_id)
3	Filter: (d.id IS NULL)
4	-> Hash Right Join (cost=13.82..20.36 rows=200 width=434) (actual time=0.064..0.238 rows=200 loops=1)
5	Hash Cond: (c.labo_id = l.id)
6	-> Seq Scan on chercheur c (cost=0.00..6.00 rows=200 width=8) (actual time=0.008..0.047 rows=200 loops=1)
7	-> Hash (cost=11.70..11.70 rows=170 width=430) (actual time=0.022..0.023 rows=5 loops=1)
8	Buckets: 1024 Batches: 1 Memory Usage: 9kB
9	-> Seq Scan on laboratoire l (cost=0.00..11.70 rows=170 width=430) (actual time=0.015..0.016 rows=5 loops=...
10	-> Hash (cost=50.00..50.00 rows=1 width=8) (actual time=0.383..0.384 rows=0 loops=1)
11	Buckets: 1024 Batches: 1 Memory Usage: 8kB
12	-> Seq Scan on dataset d (cost=0.00..50.00 rows=1 width=8) (actual time=0.382..0.383 rows=0 loops=1)
13	Filter: (((license_type IS NULL) OR (date_depot IS NULL)) AND (EXTRACT(year FROM date_depot) = '2024'::num...
14	Rows Removed by Filter: 1000
15	Planning Time: 0.363 ms
16	Execution Time: 0.765 ms

```sql
SELECT l.*
FROM laboratoire l
LEFT JOIN chercheur c ON c.labo_id = l.id
LEFT JOIN dataset d
  ON d.chercheur_id = c.id
  AND (
    d.license_type IS NULL OR d.date_depot IS NULL
  )
  AND EXTRACT(YEAR FROM d.date_depot) = 2024
WHERE d.id IS NULL;

```

3- Définition des index pertinents pour optimiser les performances

(Les scripts de création d'index sont dans le projet rendu)

L'objectif est d'accélérer les requêtes en réduisant les parcours séquentiels complets (SeqScan) observés dans les plans d'exécution, notamment sur les jointures, les clauses WHERE, et les agrégations par année.

Requête R1 – Jeux de données déposés par projet et par année

Requête choisie : Version 2 (avec CTE)

Requête plus claire, meilleure exécution grâce à la présélection des projets validés avant les agrégations.

Justification :

- dataset.date_depot est utilisée pour le **filtrage temporel** (>= 2018).
- contrat.projet_id et dataset.contrat_id optimisent les **jointures multiples**.
- L'index composite (contrat_id, date_depot) permet à PostgreSQL de filtrer et d'agréger plus rapidement.
- `participation.projet_id` est utile pour la sous-requête `HAVING COUNT(chercheur_id) >5`.

- **Requête R2 – Projets du labo LISA sans chercheurs sous la moyenne**

Requête choisie : Version 2 (avec CTE)

Meilleure lisibilité et réutilisation des CTE (pub_chercheur_2024, moyenne_labo, etc.).

Justification :

- labo_id et publication.date_pub sont clés dans les **CTE**.
- Les jointures sur auteur_publication (N-N) sont fréquentes : index sur les deux colonnes.
- laboratoire.name est souvent utilisé dans les filtres (pour "LISA").
- Ces index réduisent drastiquement les sous-requêtes imbriquées et améliorent la performance des agrégations par laboratoire.

- **Requête R3 – Laboratoires sans jeux de données non conformes**

Requête choisie : Version 2 (avec LEFT JOIN)

Généralement plus performant pour les vérifications avec (IS NULL) qu'un NOT EXISTS imbriqué.

Justification :

- dataset(chercheur_id) évite les scans complets sur les datasets.
- L'index composite (date_depot, license_type) accélère la détection des "non conformes" (nulls sur license_type ou date_depot en 2024).
- Le LEFT JOIN s'exécute plus rapidement quand les colonnes jointes sont indexées.

Partie 5 - Triggers et Procédures Stockées :

Cette partie a pour objectif d'automatiser certaines contraintes de gestion, notamment l'intégrité fonctionnelle des projets de recherche, et de fournir des fonctions opérationnelles permettant de gérer et d'analyser efficacement ces projets. Elle facilite ainsi le suivi des données et assure que les informations manipulées restent cohérentes tout au long du cycle de vie du projet.

❖ Question 1

1- Trigger : Limite de participants à un projet:

➢ Le trigger trg_check_participation_limit empêche l'ajout d'un chercheur à un projet lorsque la capacité maximale est atteinte.

Ce mécanisme garantit la cohérence métier sans dépendre d'un contrôle applicatif.

Empêcher qu'un projet dépasse une capacité maximale de participants.

➢ On ajoute d'abord un champ capacite_max dans la table projet pour pouvoir vérifier la limite.

```sql
ALTER TABLE projet ADD COLUMN capacite_max INT DEFAULT 10;
```

- Fonction du trigger:

Query | Query History

```sql
CREATE OR REPLACE FUNCTION check_participation_limit()
RETURNS TRIGGER AS $$
DECLARE
    nb_participants INT;
    max_participants INT;
BEGIN
    -- Récupérer le nombre actuel de participants du projet
    SELECT COUNT(*) INTO nb_participants
    FROM participation
    WHERE projet_id = NEW.projet_id;

    -- Récupérer la capacité maximale du projet
    SELECT capacite_max INTO max_participants
    FROM projet
    WHERE id = NEW.projet_id;

    -- Vérification de la limite
    IF nb_participants >= max_participants THEN
        RAISE EXCEPTION 'Impossible d''ajouter un nouveau participant : capacité maximale de % atteinte pour le projet %.',
            max_participants, NEW.projet_id;
    END IF;

    RETURN NEW;
```

Data Output | Messages | Notifications

```
CREATE FUNCTION

Query returned successfully in 658 msec.
```

- Définition du trigger :

```
Query  Query History
1  CREATE TRIGGER trg_check_participation_limit
2  BEFORE INSERT ON participation
3  FOR EACH ROW
4  EXECUTE FUNCTION check_participation_limit();
5
```

```
Data Output  Messages  Notifications
CREATE TRIGGER

Query returned successfully in 105 msec.
```

- Test du trigger :

```
Query  Query History
1  -- Exemple de projet avec une capacité max
2  INSERT INTO projet (title, capacite_max) VALUES ('Projet IA', 2);
3
4  -- Deux chercheurs
5  INSERT INTO chercheur (first_name, last_name) VALUES ('Meldi', 'User');
6  INSERT INTO chercheur (first_name, last_name) VALUES ('Sara', 'Khalil');
7
8  -- Ajouter leurs participations (OK)
9  INSERT INTO participation VALUES (1, 1);
10 INSERT INTO participation VALUES (1, 2);
11
12 -- Tentative d'ajout d'un troisième chercheur (Erreur)
13 INSERT INTO chercheur (first_name, last_name) VALUES ('Younes', 'Shimi');
14 INSERT INTO participation VALUES (1, 3); -- Doit lever une exception
15
```

```
Data Output  Messages  Notifications
ERROR:  Impossible d'ajouter un nouveau participant : capacité maximale de 10 atteinte pour le projet 1.
CONTEXT:  PL/pgSQL function check_participation_limit() line 18 at RAISE

SQL state: P0001
```

2- Trigger : Vérification DMP avant dépôt d'un dataset

Empêcher qu'un **dataset** soit mis au statut `validé` si le **contrat** associé n'a pas un **DMP validé**.

- Fonction du trigger:

```
1  CREATE OR REPLACE FUNCTION check_dmp_before_dataset_validation()
2  RETURNS TRIGGER AS $$
3  DECLARE
4      dmp_statut VARCHAR(20);
5  BEGIN
6      -- Vérifier le DMP associé
7      SELECT dmp_status INTO dmp_statut
8      FROM contrat
9      WHERE id = NEW.contrat_id;
10
11     -- Si DMP non validé et statut du dataset = 'validé', bloquer
12     IF NEW.statut = 'validé' AND dmp_statut <> 'valide' THEN
```

```
Data Output  Messages  Notifications
CREATE FUNCTION

Query returned successfully in 625 msec.
```

- Définition du trigger :

Query Query History

```
1  CREATE TRIGGER trg_check_dmp_before_validation
2  BEFORE INSERT OR UPDATE ON dataset
3  FOR EACH ROW
4  EXECUTE FUNCTION check_dmp_before_dataset_validation();
5
```

Data Output Messages Notifications

```
CREATE TRIGGER

Query returned successfully in 842 msec.
```

- Test du trigger :

Query Query History

```
1   -- Créer un contrat non validé
2   INSERT INTO contrat (projet_id, dmp_status) VALUES (1, 'brouillon');
3
4   -- Dataset lié (devrait échouer)
5   INSERT INTO dataset (title, contrat_id, statut) VALUES ('Données Test', 1, 'validé');
6
7   -- Contrat validé
8   UPDATE contrat SET dmp_status = 'valide' WHERE id = 1;
9
10  -- Cette fois, le dataset passe
11  INSERT INTO dataset (title, contrat_id, statut) VALUES ('Données Valides', 1, 'validé');
12
```

Data Output Messages Notifications

```
INSERT 0 1

Query returned successfully in 129 msec.
```

❖ Question 2

1- Fonction — Nombre de publications d'un projet et d'une année

Compter le nombre de publications liées à un projet donné via ses chercheurs participants.

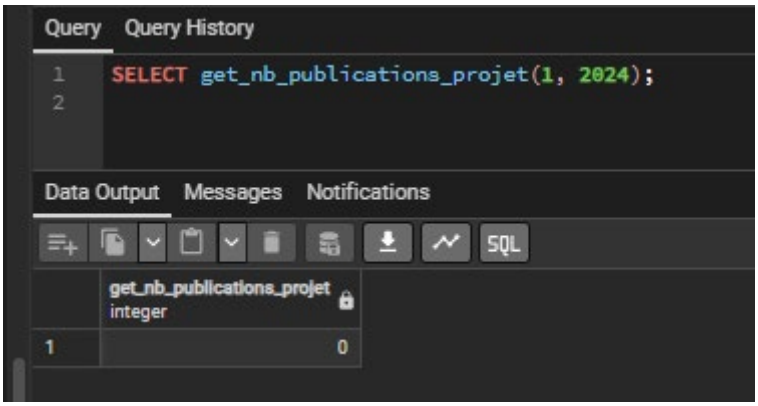
```
1   CREATE OR REPLACE FUNCTION get_nb_publications_projet(
2       p_projet_id INT,
3       p_annee INT
4   )
5   RETURNS INT AS $$
6   DECLARE
7       nb_pub INT;
8   BEGIN
9       SELECT COUNT(DISTINCT pub.id) INTO nb_pub
10      FROM publication pub
11      JOIN auteur_publication ap ON ap.publication_id = pub.id
12      JOIN participation pa ON pa.chercheur_id = ap.chercheur_id
13      WHERE pa.projet_id = p_projet_id
14        AND EXTRACT(YEAR FROM pub.date_pub) = p_annee;
15
16      RETURN nb_pub;
17  END;
18  $$ LANGUAGE plpgsql;
19
```

Data Output Messages Notifications

```
CREATE FUNCTION

Query returned successfully in 128 msec.
```

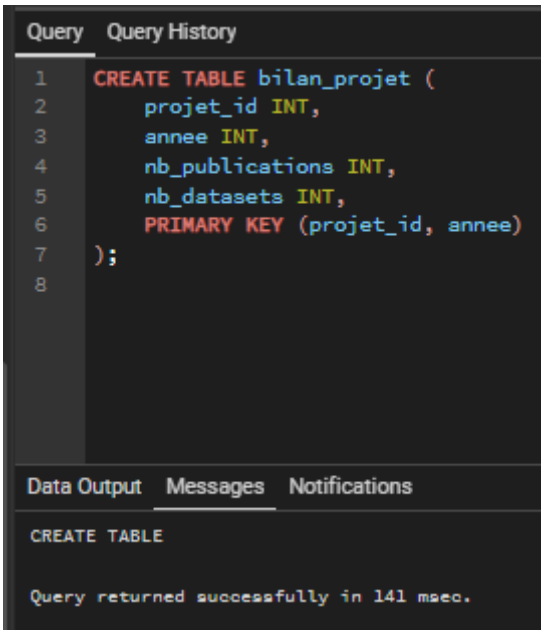
Test :



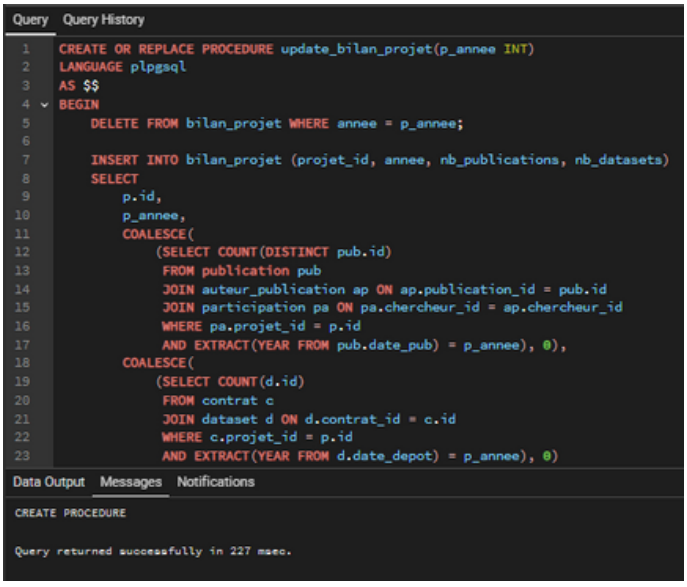
2- Procédure — Mise à jour du bilan annuel des projets

Mettre à jour une table **bilan_projet** contenant pour chaque projet et année le nombre de publications et datasets.

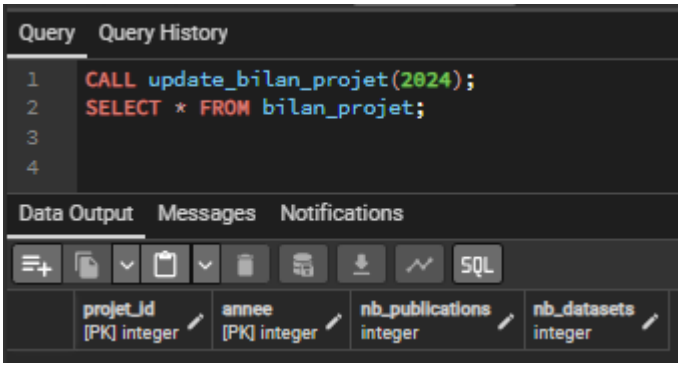
- Création de la table de bilan:



- Procédure:

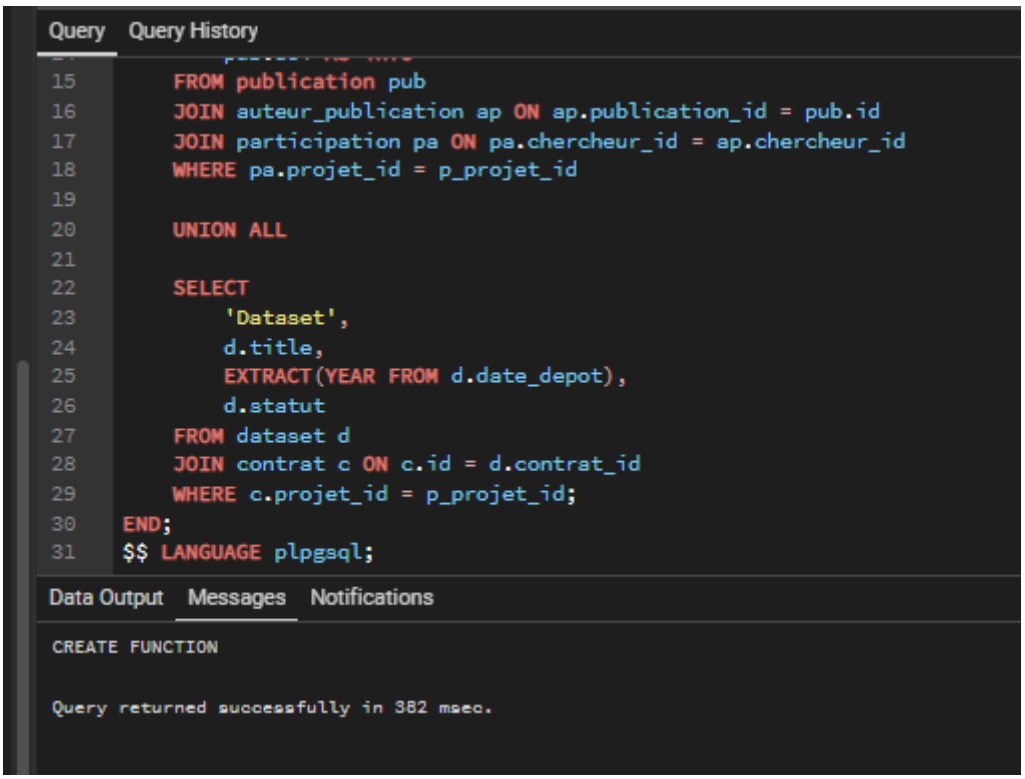


- Test:

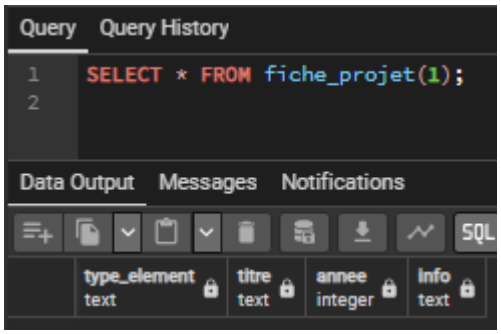


3- Fonction — Fiche projet détaillée :

Afficher les publications et datasets associés à un projet



Test :



4- Procédure — Archivage des contrats échus :

Déplacer les contrats expirés avant une date donnée vers une table d'archives.

Création de la table archive:

```
Query  Query History
1  CREATE TABLE contrat_archive AS TABLE contrat WITH NO DATA;
2

Data Output  Messages  Notifications
CREATE TABLE AS

Query returned successfully in 105 msec.
```

Procédure :

```
1  CREATE OR REPLACE PROCEDURE archiver_contrats(p_date_seuil DATE)
2  LANGUAGE plpgsql
3  AS $$
4  BEGIN
5      INSERT INTO contrat_archive
6      SELECT * FROM contrat
7      WHERE date_fin < p_date_seuil;
8
9      DELETE FROM contrat
10     WHERE date_fin < p_date_seuil;
11 END;
12 $$;
13

Data Output  Messages  Notifications
CREATE PROCEDURE

Query returned successfully in 104 msec.
```

Test :

```
Query  Query History
1  CALL archiver_contrats('2025-01-01');
2  SELECT * FROM contrat_archive;
3

Data Output  Messages  Notifications
```

	id integer	projet_id integer	financeur character varying (200)	intitule text	montant numeric	date_debut date	date_fin date	dmp_status character varying (20)	dmp_link text

• Fonctions et procédures stockées

Fonction / Procédure	Description	Type
`get_nb_publications_projet`	Retourne le nombre de publications d'un projet pour une année donnée	Fonction
`update_bilan_projet`	Met à jour la table **bilan_projet** avec le nombre de publications et datasets par projet et année	Procédure
`fiche_projet`	Renvoie la fiche détaillée d'un projet (publications et datasets)	Fonction
`archiver_contrats`	Déplace les contrats expirés dans une table d'archives	Procédure

❖ Tests et validation

Chaque trigger et procédure a été testé avec des jeux de données fictifs.
Les tests ont confirmé :

- le refus d'insertion lors du dépassement de capacité,
- le blocage de validation sans DMP,
- la génération correcte du bilan et des fiches projets,
- l'archivage automatique des contrats expirés.