Installation de Postgre

1 - Afficher la liste des bases de données dans PostgreSQL :

On affiche la liste des bases de données avec la commande :

\l ou \list

```
rayanew=> \l
                                                List of databases
   Name    |  Owner   | Encoding | Locale Provider |   Collate   |    Ctype    | ICU Locale | ICU Rules |   Access privileges
-----------+----------+----------+-----------------+-------------+-------------+------------+-----------+-----------------------
 postgres  | postgres | UTF8     | libc            | fr_FR.UTF-8 | fr_FR.UTF-8 |            |           |
 rayanew   | rayanew  | UTF8     | libc            | fr_FR.UTF-8 | fr_FR.UTF-8 |            |           |
 template0 | postgres | UTF8     | libc            | fr_FR.UTF-8 | fr_FR.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |             |             |            |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | fr_FR.UTF-8 | fr_FR.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |             |             |            |           | postgres=CTc/postgres
(4 rows)
```

2 - Afficher les tables de la base courante :

On affiche les tables des bases de données avec la commande :

\dt

3 - Créer une nouvelle base de données nommée

On créer une base de donnée avec la commande :

CREATE DATABASE test2;

```
rayanew=> CREATE DATABASE test2;
CREATE DATABASE
rayanew=>
```

4 - Se connecter à la nouvelle base de données

On se connecte à la base de donnée avec la commande :

\c test2

```
rayanew=> \c test2
You are now connected to database "test2" as user "rayanew".
test2=>
```

5 - Afficher la liste des utilisateurs dont le nom commence par "t" :

On affiche la liste des utilisateurs commencent par ‘t’ avec la commande :

SELECT rolname FROM pg_roles WHERE rolname LIKE 't%';

```
test2=> SELECT rolname FROM pg_roles WHERE rolname LIKE 't%';
 rolname
---------
(0 rows)

test2=>
```

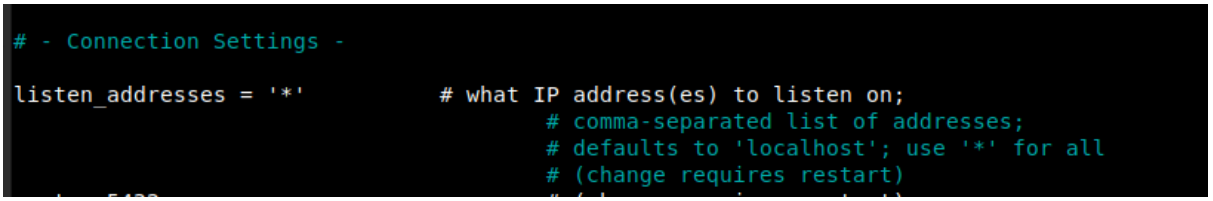
(0 utilisateur commençant par "t" car il n’y a que mon utilisateur "rayanew")

Connexion à l'aide d'un client

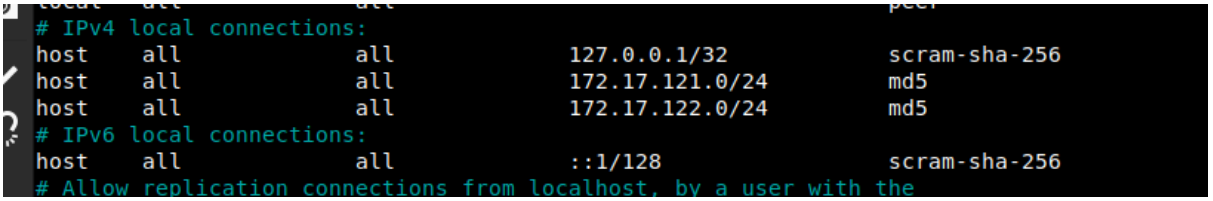
1- Éditez le fichier /etc/postgresql/version/main/postgresql.conf.
Dans quel mode faut-il se trouver ?

Le mode dans lequel il faut se trouver est nano (“sudo” avant pour avoir les droits).

2- Modifiez la ligne suivante pour autoriser n'importe qu'elle machine à se connecter listen_addresses = '*' (n'oubliez d'enlever le commentaire #)

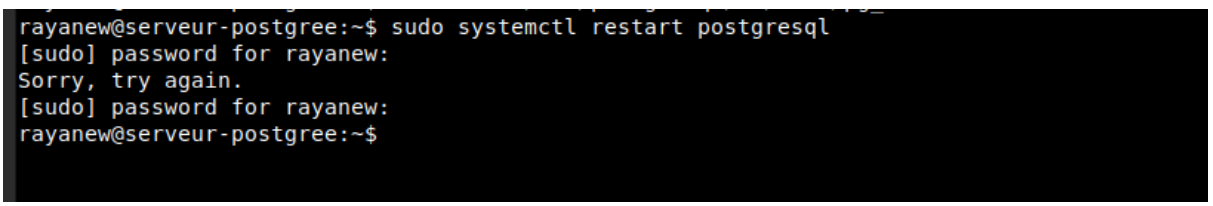


3. Editez également le fichier /etc/postgresql/ version /main/pg_hba.conf
Ajoutez la nouvelle ligne suivante pour autoriser les connexions provenant de tout le réseau des salles 121 et 122 :
host all all 172.17.121.0/24 md5
host all all 172.17.122.0/24 md5



5.Redémarrez le service postgresql à l'aide de la commande Commande à taper ?

Pour redémarrer le service postgre on utilise la commande :
sudo systemctl restart postgresql



6. Récupérez l'adresse IP de votre serveur : Commande à taper ?

Pour obtenir l'adresse ip du serveur on peut utiliser :

hostname-I

```
rayanew@serveur-postgree:~$ hostname -I
172.17.122.61
rayanew@serveur-postgree:~$
```

7. Sur votre machine hôte, installez le paquet postgresql-client : Commande à taper ?

Pour installer postegresql-client on fait :

sudo apt update

sudo apt install postgresql-client

```
test@modele:~$ sudo apt install postgresql-client
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  libpq5 postgresql-client-14 postgresql-client-common
Paquets suggérés :
  postgresql-14 postgresql-doc-14
Les NOUVEAUX paquets suivants seront installés :
  libpq5 postgresql-client postgresql-client-14 postgresql-client-common
0 mis à jour, 4 nouvellement installés, 0 à enlever et 70 non mis à jour.
Il est nécessaire de prendre 1 435 ko dans les archives.
Après cette opération, 4 881 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
Réception de :1 http://fr.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libpq5 amd64 14.19-0ubuntu0.22.04.1 [152 kB]
Réception de :2 http://fr.archive.ubuntu.com/ubuntu jammy/main amd64 postgresql-client-common all 238 [29,6 kB]
Réception de :3 http://fr.archive.ubuntu.com/ubuntu jammy-updates/main amd64 postgresql-client-14 amd64 14.19-0ubuntu0.22.04.1 [1 249 kB]
Réception de :4 http://fr.archive.ubuntu.com/ubuntu jammy/main amd64 postgresql-client all 14+238 [3 292 B]
1 435 ko réceptionnés en 0s (6 597 ko/s)
Sélection du paquet libpq5:amd64 précédemment désélectionné.
(Lecture de la base de données... 371864 fichiers et répertoires déjà installés.
)
Préparation du dépaquetage de .../libpq5_14.19-0ubuntu0.22.04.1_amd64.deb ...
Dépaquetage de libpq5:amd64 (14.19-0ubuntu0.22.04.1) ...
Sélection du paquet postgresql-client-common précédemment désélectionné.
Préparation du dépaquetage de .../postgresql-client-common_238_all.deb ...
Dépaquetage de postgresql-client-common (238) ...
Sélection du paquet postgresql-client-14 précédemment désélectionné.
Préparation du dépaquetage de .../postgresql-client-14_14.19-0ubuntu0.22.04.1_am
d64.deb ...
Dépaquetage de postgresql-client-14 (14.19-0ubuntu0.22.04.1) ...
Sélection du paquet postgresql-client précédemment désélectionné.
Préparation du dépaquetage de .../postgresql-client_14+238_all.deb ...
Dépaquetage de postgresql-client (14+238) ...
Paramétrage de postgresql-client-common (238) ...
Paramétrage de libpq5:amd64 (14.19-0ubuntu0.22.04.1) ...
Paramétrage de postgresql-client-14 (14.19-0ubuntu0.22.04.1) ...
update-alternatives: utilisation de « /usr/share/postgresql/14/man/man1/psql.1.g
z » pour fournir « /usr/share/man/man1/psql.1.gz » (psql.1.gz) en mode automatiq
ue
Paramétrage de postgresql-client (14+238) ...
Traitement des actions différées (« triggers ») pour man-db (2.10.2-1) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.35-0ubuntu3.10)
```

8. Connectez-vous depuis la machine hôte avec la commande suivante : Commande à taper ?

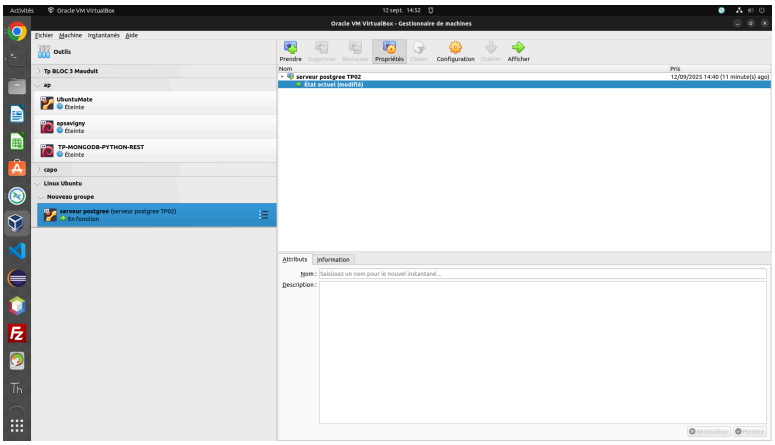
Pour se connecter depuis la machine hôte on utilise la commande :

psql -h <adresse_IP_du_serveur> -U <nom_utilisateur> -d <nom_base>

psql -h 172.17.122.61 -U rayanew -d rayanew

```
test@modele:~$ psql -h 172.17.122.61 -U rayanew -d rayanew
Password for user rayanew:
psql (14.19 (Ubuntu 14.19-0ubuntu0.22.04.1), server 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
WARNING: psql major version 14, server major version 16.
         Some psql features might not work.
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

rayanew=>
```

9. Lorsque tout fonctionne bien, réalisez un nouvel instantané de la machine virtuelle, que vous nommerez TP02



Administration de bases de données postgresql

1- Comment redémarrer le service du serveur Postgresql ?

On redémarre le service avec :
sudo systemctl restart postgresql

2 - Créer un utilisateur de la base de données nommé picsou qui sera un utilisateur sans privilèges particuliers (attention il faut que cet utilisateur puisse se connecter).

J'ai créé l'utilisateur *picsou* avec la commande CREATE USER picsou;.

```
postgres=# create user picsou ;
CREATE ROLE
postgres=#
```

3 - Créer un utilisateur nommé admin (mot de passe : mpadmin) qui aura des privilèges d'administrateur

J'ai créé l'utilisateur *admin* et lui ai attribué le mot de passe *mpadmin. Avec* les commandes suivantes

create user admin;
alter user admin with encrypted password 'mpadmin';
alter user admin with superuser;

```
postgres=# create user admin;
CREATE ROLE
postgres=# alter users admin with encrypted password 'mpadmin';
ERROR:  syntax error at or near "users"
LINE 1: alter users admin with encrypted password 'mpadmin';
            ^
postgres=# alter user admin with encrypted password 'mpadmin';
ALTER ROLE
postgres=#
```

```
postgres=# alter user admin with superuser;
ALTER ROLE
postgres=#
```

4 – Vérifier que ces 2 utilisateurs ont bien été créés.

```
postgres=# SELECT rolname FROM pg_roles;
           rolname
----------------------------
 postgres
 pg_database_owner
 pg_read_all_data
 pg_write_all_data
 pg_monitor
 pg_read_all_settings
 pg_read_all_stats
 pg_stat_scan_tables
 pg_read_server_files
 pg_write_server_files
 pg_execute_server_program
 pg_signal_backend
 pg_checkpoint
 pg_use_reserved_connections
 pg_create_subscription
 rayanew
 picsou
 admin
(18 rows)

postgres=#
```

La commande SELECT rolname FROM pg_roles; montre bien *picsou* et *admin* dans la liste des rôles.
On remarque dans les 2 dernières lignes picsou et admin

4 – Ajouter un mot de passe au compte utilisateur Picsou.

J'ai attribué le mot de passe *picsou* à l'utilisateur *picsou*.

```
postgres=# alter user picsou with encrypted password 'picsou';
ALTER ROLE
postgres=#
```

5 – Créer un groupe (rôle) gr_admin et un groupe (rôle) gr_canards.
J'ai créé deux groupes de rôles : *gr_admin* et *gr_canards*.
create role gr_admin;
create role gr_canards;

```
postgres=# create role gr_admin;
CREATE ROLE
postgres=# create role gr_canards;
CREATE ROLE
postgres=#
```

6 – Faire en sorte que Picsou appartienne au groupe gr_canards et que admin appartienne à gr_admin.

J'ai ajouté *picsou* au groupe *gr_canards* et *admin* au groupe *gr_admin* avec GRANT.

GRANT gr_canards TO picsou;
GRANT gr_admin TO admin;

```
postgres=# GRANT gr_canards TO picsou;
GRANT gr_admin TO admin;
GRANT ROLE
GRANT ROLE
postgres=#
```

7 – Vérifier que ces 2 groupes ont bien été créés. Que signifie le contenu de chaque colonne ?

La commande \du montre les deux groupes.
La colonne des attributs indique leurs droits (droite)

```
postgres=# \du
                             List of roles
 Role name  |                         Attributes
------------+------------------------------------------------------------
 admin      |
 gr_admin   | Cannot login
 gr_canards | Cannot login
 picsou     |
 postgres   | Superuser, Create role, Create DB, Replication, Bypass RLS
 rayanew    | Create DB
```

8 – Vérifier que Picsou peut se connecter depuis le client avec son compte utilisateur à la base de données template1.

La commande psql -h 172.17.122.61 -U picsou -d template1 fonctionne et permet à *picsou* de se connecter à la base *template1*.

```
test@modele:~$ psql -h 172.17.122.61 -U picsou -d template1
Password for user picsou:
psql (14.19 (Ubuntu 14.19-0ubuntu0.22.04.1), server 16.10 (Ubuntu 16.10-0ubuntu0
.24.04.1))
WARNING: psql major version 14, server major version 16.
         Some psql features might not work.
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, co
mpression: off)
Type "help" for help.

template1=>
```

9 - Pouvez-vous créer une table (si oui, n'oubliez pas de la supprimer ensuite) ? Pouvez-vous créer un utilisateur (si oui, n'oubliez pas de le supprimer ensuite)

```
template1=> CREATE TABLE test_table (id SERIAL PRIMARY KEY, nom TEXT);
ERROR:  permission denied for schema public
LINE 1: CREATE TABLE test_table (id SERIAL PRIMARY KEY, nom TEXT);
                    ^
template1=>
```

```
template1=> CREATE USER testuser;
ERROR:  permission denied to create role
DETAIL:  Only roles with the CREATEROLE attribute may create roles.
template1=>
```

Picsou ne peut pas créer de table ni d'utilisateur car il n'a pas les privilèges nécessaires.

Tp PSQL CSV

b) Connexion à une base de données PostgreSQL

```
test@modele:~$ psql -h 172.17.122.61 -U rayanew
Password for user rayanew:
psql (14.19 (Ubuntu 14.19-0ubuntu0.22.04.1), server 16.10 (Ubuntu 16.10-0ubuntu
.24.04.1))
WARNING: psql major version 14, server major version 16.
         Some psql features might not work.
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, c
mpression: off)
Type "help" for help.

rayanew=>
```

c) Créer une base de données avec PostgreSQL

CREATE TABLE etu (

pknumsecu CHAR(13) PRIMARY KEY,

knumetu VARCHAR(20) UNIQUE NOT NULL,

nom VARCHAR (50),

prenom VARCHAR (50));

INSERT INTO etu (pknumsecu, knumetu, nom, prenom) VALUES ('1800675001066', 'AB3937098X', 'Dupont', 'Pierre');

INSERT INTO etu (pknumsecu, knumetu, nom, prenom) VALUES ('2820475001124', 'XGB67668', 'Durand', ‘Anne');

Question 2:

Select * from etu;

```
rayanew=> select * from etu;
   pknumsecu   |  knumetu   |  nom   | prenom
---------------+------------+--------+--------
 1800675001066 | AB3937098X | Dupont | Pierre
 2820475001124 | XGB67668   | Durand | Anne
(2 rows)
```

Select * from uv;

```
rayanew=> select * from uv;
 pkcode |     fketu
--------+---------------
 NF17   | 1800675001066
 NF26   | 1800675001066
 NF29   | 1800675001066
(3 rows)
```

d) Import de données depuis un fichier CSV

Question 1 : Supprimer les données existantes dans les tables

```
rayanew=> DELETE FROM uv;
DELETE FROM etu;
DELETE 3
DELETE 2
```

Question 2:

Créer les deux fichiers de données suivants, respectivement etus.csv et uvs.csv. Regarder le contenu des fichiers. Pourquoi les appelle-t-on des fichiers CSV ?

Créer deux fichiers texte (au format CSV) :

```
rayanew@serveur-postgree:~$ nano etu.csv
rayanew@serveur-postgree:~$ nano uvs.csv
rayanew@serveur-postgree:~$ cat etu.csv
1;A;Dupont;Pierre
2;B;Durand;Georges
3;C;Duchemin;Paul
4;D;Dugenou;Alain
5;E;Dupied;Albert
rayanew@serveur-postgree:~$ cat uvs.csv
1;NF17
1;NF18
1;LA13
1;PH01
2;NF17
2;NF18
2;NF19
2;TN01
2;LA14
2;PH01
3;NF17
3;NF18
3;NF19
3;NF21
3;LA14
3;PH01
4;NF17
4;NF20
4;NF21
4;GE10
4;LA14
4;PH01
5;NF17
5;NF18
5;NF20
5;GE10
5;PH01
5;PH02
```

Parce que ce sont des **fichiers texte structurés** où les valeurs sont **séparées par un délimiteur** (`;` ou `,`).

Question 3 Insérer les données en complétant les instructions suivantes.
Importer les fichiers csv

```
rayanew=> \copy etu FROM 'etu.csv' DELIMITER ';' CSV;
COPY 5
rayanew=> \copy etu FROM 'uvs.csv' DELIMITER ';' CSV;
ERROR:  missing data for column "nom"
CONTEXT:  COPY etu, line 1: "1;NF17"
```

Question 4 Écrivez les requêtes permettant d'obtenir le nombre d'UV suivies par un étudiant et le nombre d'étudiants inscrits par UV.

Nombre d'UV suivies par étudiant :

```
SELECT fketu AS etudiant, COUNT(pkcode) AS nb_uv
FROM uv
GROUP BY fketu;
```

```
rayanew=> SELECT fketu AS etudiant, COUNT(pkcode) AS nb_uv
FROM uv
GROUP BY fketu;
   etudiant    | nb_uv
---------------+-------
 2             |     6
 4             |     6
 3             |     6
 5             |     6
 1             |     4
(5 rows)
```

Nombre d'étudiants inscrits par UV :

```
SELECT pkcode AS code_uv, COUNT(fketu) AS nb_etudiants
FROM uv
GROUP BY pkcode;
```

```
rayanew=> SELECT pkcode AS code_uv, COUNT(fketu) AS nb_etudiants
FROM uv
GROUP BY pkcode;
 code_uv | nb_etudiants
---------+--------------
 NF18    |            4
 PH02    |            1
 NF20    |            2
 NF21    |            2
 PH01    |            5
 TN01    |            1
 LA14    |            3
 NF19    |            2
 GE10    |            2
 LA13    |            1
 NF17    |            5
(11 rows)
```

e) Manipulation de schémas

Question 1 Visualiser la liste des schémas existants dans votre base de données.

On réalise la commande \dn

```
rayanew=> \dn
        List of schemas
    Name    |       Owner
------------+--------------------
 public     | pg_database_owner
 schematest | rayanew
(2 rows)
```

Question 2 Créer un second schéma loisir.

CREATE SCHEMA loisir;

Question 3 : Créer une table sport dans le schéma loisir

```
CREATE TABLE loisir.sport (
  fk_etu CHAR(13) REFERENCES etu(pknumsecu),
  sport VARCHAR(30),
  PRIMARY KEY (fk_etu, sport)
);
```

```
rayanew=> CREATE SCHEMA loisir;
CREATE SCHEMA
rayanew=> CREATE TABLE loisir.sport (
  fk_etu CHAR(13) REFERENCES etu(pknumsecu),
  sport VARCHAR(30),
  PRIMARY KEY (fk_etu, sport)
);
CREATE TABLE
rayanew=> \d loisir.sport
                    Table "loisir.sport"
 Column |         Type          | Collation | Nullable | Default
--------+-----------------------+-----------+----------+---------
 fk_etu | character(13)         |           | not null |
 sport  | character varying(30) |           | not null |
Indexes:
    "sport_pkey" PRIMARY KEY, btree (fk_etu, sport)
Foreign-key constraints:
    "sport_fk_etu_fkey" FOREIGN KEY (fk_etu) REFERENCES etu(pknumsecu)
```

Partie f) Exécution de fichiers SQL et scripts Linux

Question 1 Exécuter ce fichier depuis psql.

```
rayanew@serveur-postgree:~$ nano init.sql
rayanew@serveur-postgree:~$ cat init.sql
DELETE FROM uv;
DELETE FROM etu;

CREATE TABLE etu (...);
CREATE TABLE uv (...);
INSERT INTO ...;
rayanew@serveur-postgree:~$ psql -d ma_base -U postgres -f init.sql
```

ensuite faire la commande psql -d ma_base -U postgres -f init.sql les éléments en vert sont à modifier

Question 2 Exécuter ce fichier depuis un script Linux.

```
rayanew@serveur-postgree:~$ nano run.sh
rayanew@serveur-postgree:~$ cat run.sh
#!/bin/bash
psql -U rayanew -d rayanew -f init.sql
rayanew@serveur-postgree:~$ chmod +x run.sh
rayanew@serveur-postgree:~$ ./run.sh
```

même chose modifier le nom de la base et l'utilisateur et le chemin du init pour convenir à votre serveur

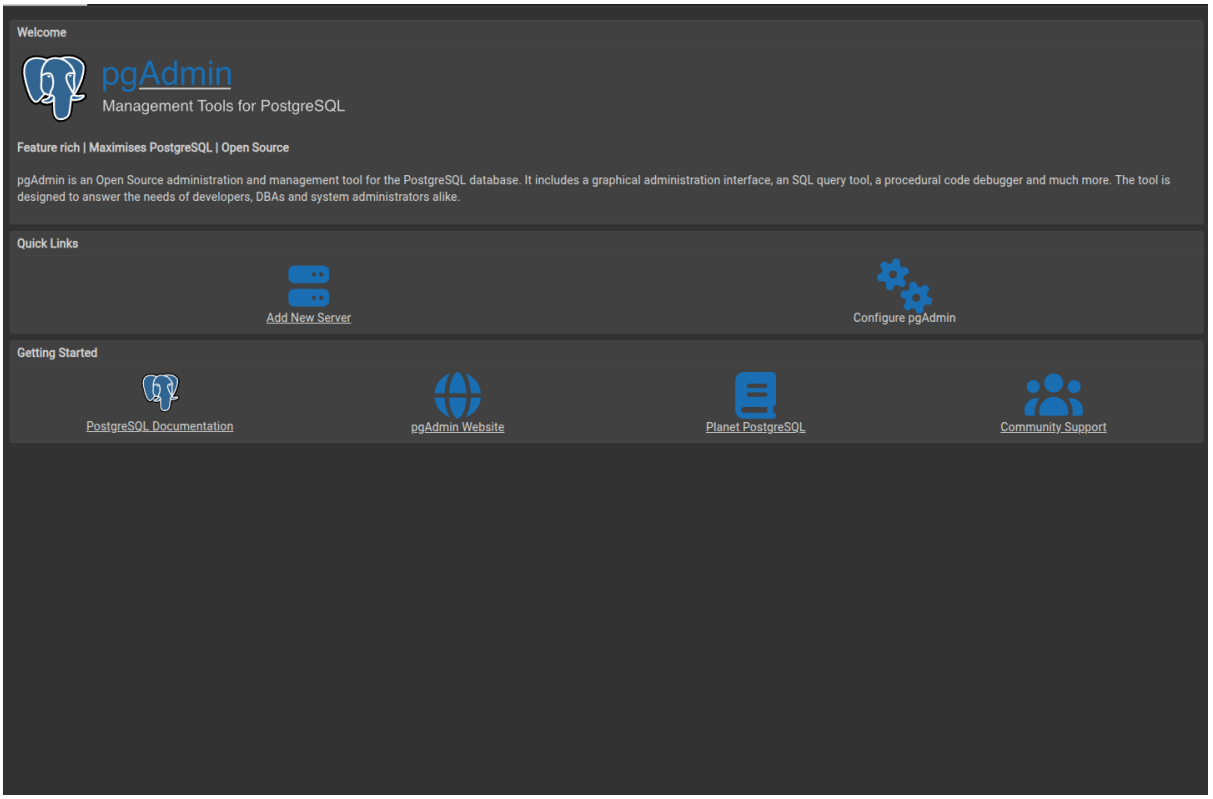
Installation d’un client graphique :

1. Les dépôts Ubuntu contiennent une application graphique d’administration d’une base de données : PgAdmin. Installez le paquet pgadmin3 Sur quelle machine allez-vous installer ?

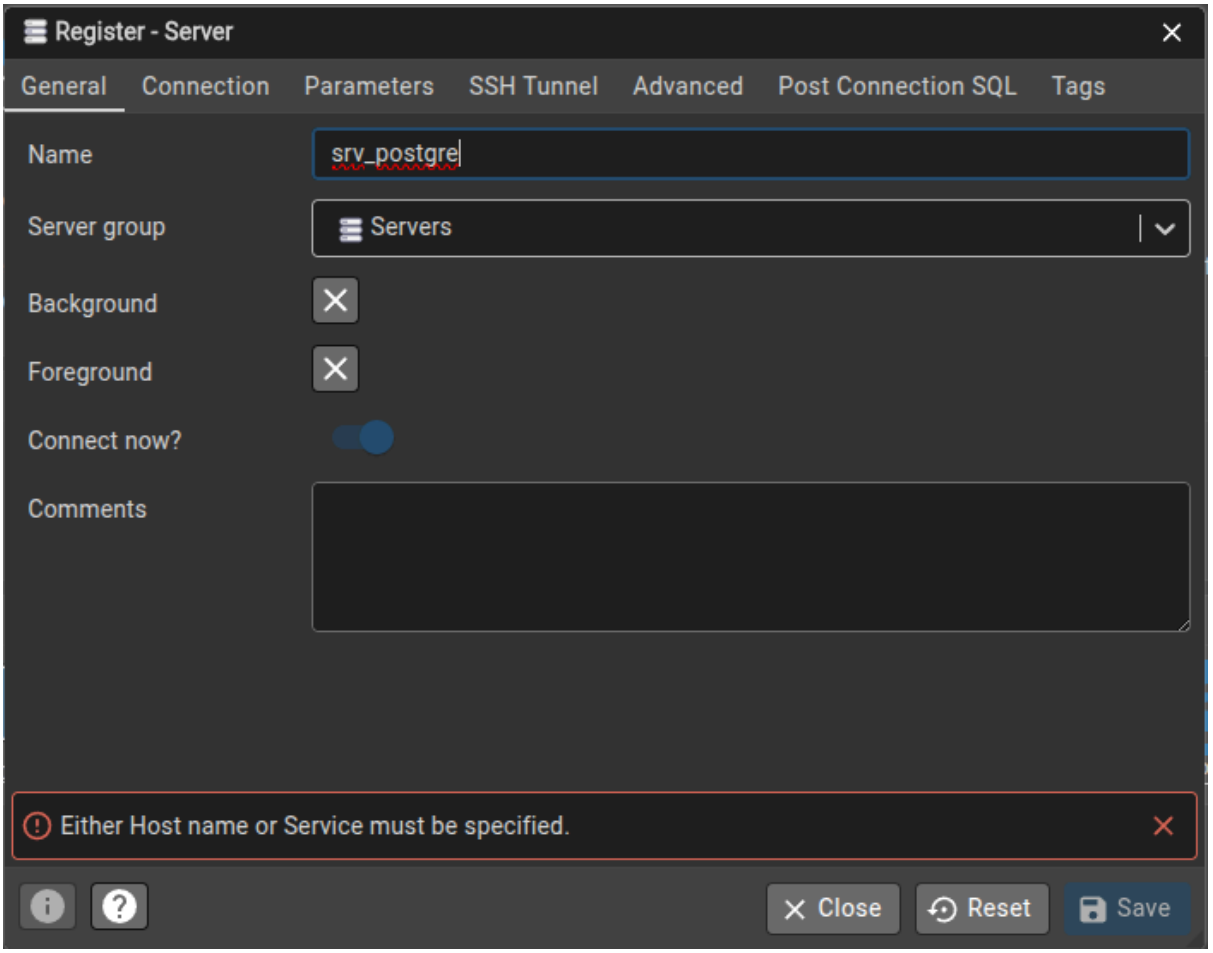
Sur notre machine physique pour cela il faut faire c’est commande :

```
# Met à jour la liste des paquets
sudo apt update

# Installe curl
sudo apt install curl -y

# Ajoute la clé publique du dépôt pgAdmin
curl -fsS https://www.pgadmin.org/static/packages_pgadmin_org.pub | sudo gpg --dearmor
-o /usr/share/keyrings/packages-pgadmin-org.gpg

# Ajoute le dépôt pgAdmin à la liste des sources et met à jour
sudo sh -c 'echo "deb [signed-by=/usr/share/keyrings/packages-pgadmin-org.gpg]
https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) pgadmin4 main" >
/etc/apt/sources.list.d/pgadmin4.list && apt update'

# Met à jour la liste des paquets
sudo apt update

# Met à jour les paquets installés
sudo apt upgrade

# Installe pgAdmin 4 (version web)
sudo apt install pgadmin4

# Installe pgAdmin 4 (version desktop)
sudo apt install pgadmin4-desktop
```

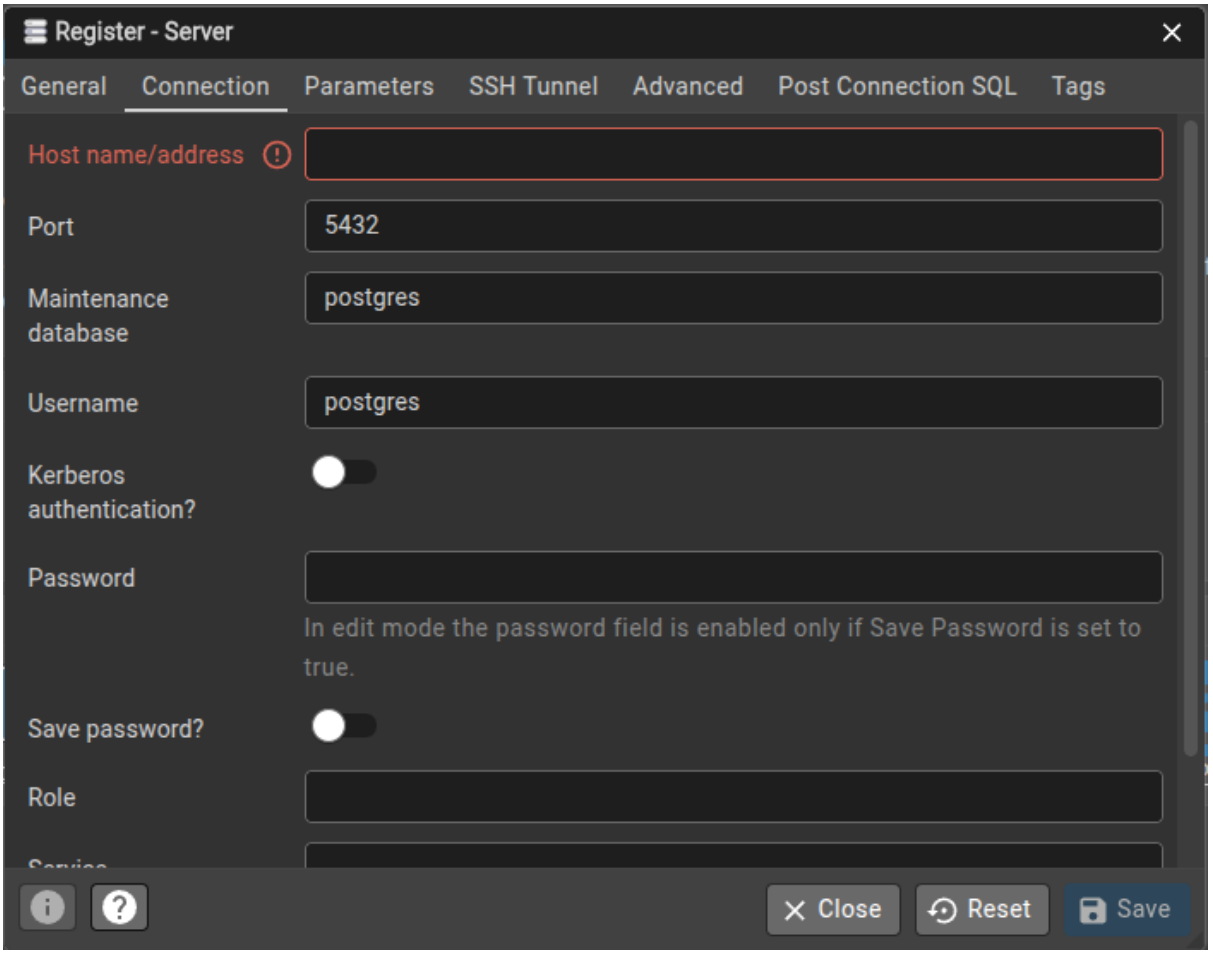
2. Connectez-vous à votre serveur « virtuel » de base de données. Les manipulations de la section précédentes doivent fonctionner pour que cela fonctionne.



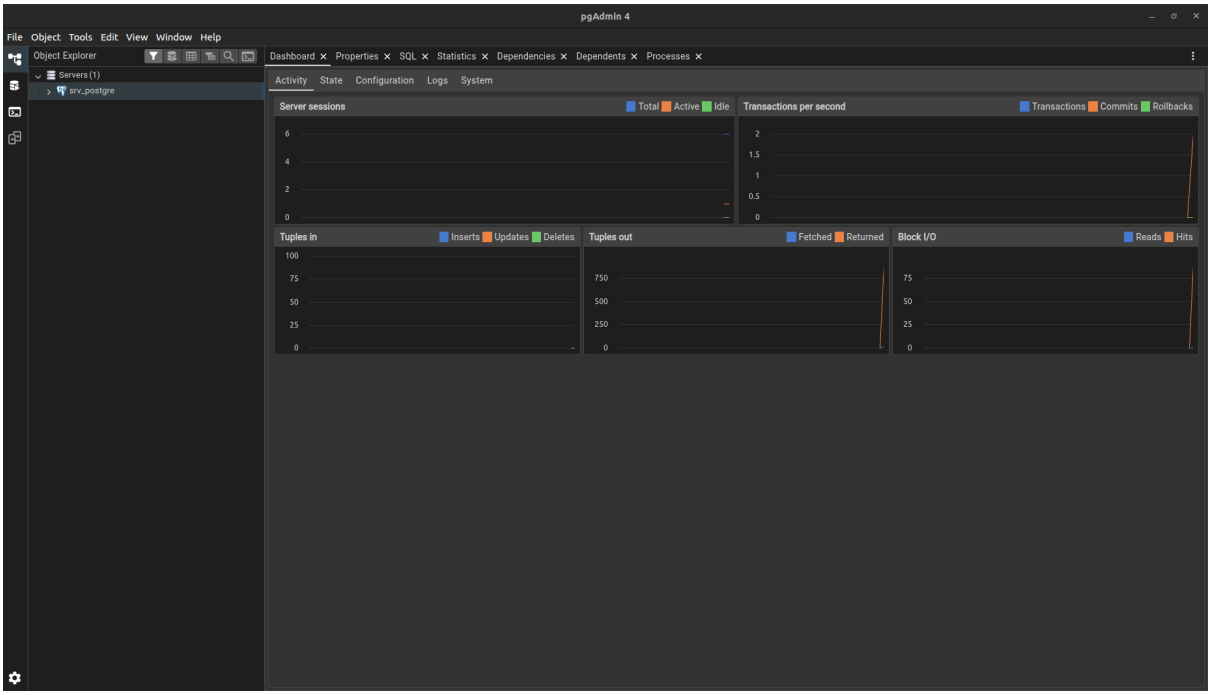
aller dans **“add new server”**



mettre le nom de ton serveur ici **srv_postgre**



ensuite ajouter **l'ip, l'user et le mdp**



nous voila connectés