

CONTENTS

Conditions préalables

Étape 1 - Installation de PostgreSQL

Étape 2 - Utilisation des rôles et des bases de données PostgreSQL

Étape 3 - Création d'un nouveau rôle

Étape 4 - Création d'une nouvelle base de données

Étape 5 - Ouverture d'une invite Postgres avec le nouveau rôle

Étape 6 - Création et suppression de tables

Étape 7 - Ajout, requête et suppression de données dans une table

Étape 8 - Ajout et suppression de colonnes dans une table

Étape 9 - Mise à jour des données dans une table

Conclusion

RELATED

Comment passer à Ubuntu 20.04 Focal Fossa

[View](#)

Comment installer et configurer Elasticsearch sur Ubuntu 20.04

[View](#)

// Tutorial //

Comment installer et utiliser PostgreSQL sur Ubuntu 20.04

Published on May 13, 2020

[Databases](#) [Ubuntu](#) [PostgreSQL](#)By [Mark Drake](#)
Manager, Developer Education[Français](#)Try DigitalOcean
for free

Click below to sign up and get \$200 of credit to try our products over 60 days!

[Sign up →](#)

Introduction

Les systèmes de gestion de bases de données relationnelles constituent un élément clé de nombreux sites Web et applications. Ils offrent un moyen structuré de stocker les informations, de les organiser et d'y accéder.

[PostgreSQL](#), ou Postgres, est un système de gestion de base de données relationnelle qui fournit une implémentation du langage de requête [SQL](#). Il est conforme aux normes et dispose de nombreuses fonctionnalités avancées, comme des transactions fiables et une simultanéité sans verrou de lecture.

Ce guide explique comment installer Postgres sur un serveur Ubuntu 20.04. Il fournit également quelques instructions pour l'administration générale des bases de données.

Conditions préalables

Pour suivre ce tutoriel, vous aurez besoin d'un serveur Ubuntu 20.04 qui a été configuré en suivant notre guide de [configuration initiale de serveur Ubuntu 20.04](#). Au terme de ce tutoriel préalable, votre serveur devrait avoir un utilisateur non **root** avec permissions sudo et un pare-feu basique.

Étape 1 - Installation de PostgreSQL

Les référentiels par défaut d'Ubuntu contiennent des packages Postgres, vous

Popular Topics

[Ubuntu](#)[Linux Basics](#)[JavaScript](#)[React](#)[Python](#)[Security](#)[MySQL](#)[Docker](#)[Kubernetes](#)[Browse all topic tags](#)[Free Managed Hosting →](#)[All tutorials →](#)[Questions](#)[Q&A Forum](#)[Ask a question](#)[DigitalOcean Support](#)

pouvez donc les installer en utilisant le gestionnaire de packages `apt`.

Si vous ne l'avez pas fait récemment, actualisez l'index local des paquets de votre serveur :

```
$ sudo apt update
```

[Copy](#)

Puis, installez le package Postgres ainsi qu'un package `-contrib` qui ajoute quelques utilitaires et fonctionnalités supplémentaires :

```
$ sudo apt install postgresql postgresql-contrib
```

[Copy](#)

Maintenant que le logiciel est installé, nous pouvons voir comment il fonctionne et en quoi il peut être différent des autres systèmes de gestion de bases de données relationnelles que vous avez pu utiliser.

Étape 2 – Utilisation des rôles et des bases de données PostgreSQL

Par défaut, Postgres utilise un concept appelé « rôles » pour gérer l'authentification et l'autorisation. Ils sont, à certains égards, semblables aux comptes classiques de type Unix, mais Postgres ne fait pas de distinction entre utilisateurs et groupes et préfère le terme plus souple de « rôle ».

Une fois installé, Postgres est configuré pour utiliser l'authentification *ident*, ce qui signifie qu'il associe les rôles Postgres à un compte système Unix/Linux correspondant. Si un rôle existe au sein de Postgres, un nom d'utilisateur Unix/Linux portant le même nom peut se connecter à ce rôle.

La procédure d'installation a créé un compte utilisateur nommé **postgres** qui est associé au rôle Postgres par défaut. Pour utiliser Postgres, vous pouvez vous connecter à ce compte.

Il y a plusieurs façons d'utiliser ce compte pour accéder à Postgres.

Basculer sur le compte **postgres**

Basculez sur le compte **postgres** sur votre serveur en tapant :

```
$ sudo -i -u postgres
```

[Copy](#)

Vous pouvez maintenant accéder à l'invite PostgreSQL immédiatement en tapant :

```
$ psql
```

[Copy](#)

De là, vous êtes libre d'interagir avec le système de gestion de la base de données si nécessaire.

Sortez de l'invite PostgreSQL en tapant :

```
postgres=# \q
```

[Copy](#)

Cela vous ramènera à l'invite de commande `postgres` de Linux.

Accéder à une invite Postgres sans changer de compte

Vous pouvez également exécuter la commande que vous souhaitez avec le compte **postgres** directement en utilisant `sudo`.

Par exemple, dans l'exemple précédent, on vous a indiqué de vous rendre à l'invite Postgres en basculant d'abord sur l'utilisateur **postgres** puis en exécutant `psql` pour ouvrir l'invite Postgres. Vous pouvez faire cela en une seule étape en

exécutant la commande unique `psql` en tant qu'utilisateur **postgres** avec privilèges `sudo`, comme ceci :

```
$ sudo -u postgres psql
```

Copy

Cela vous connectera directement dans Postgres sans passer par le shell `bash` intermédiaire.

Là encore, vous pouvez quitter la session Postgres interactive en tapant :

```
postgres=# \q
```

Copy

De nombreux cas d'utilisation nécessitent plus d'un rôle Postgres. Lisez ce qui suit pour savoir comment les configurer.

Étape 3 - Création d'un nouveau rôle

Actuellement, le rôle **postgres** est le seul à être configuré dans la base de données. Vous pouvez créer de nouveaux rôles à partir de la ligne de commande avec la commande `createuser`. L'indicateur `--interactive` vous invitera à indiquer le nom du nouveau rôle et vous demandera également si celui-ci doit disposer des autorisations de superutilisateur.

Si vous êtes connecté en tant que compte **postgres**, vous pouvez créer un nouvel utilisateur en tapant :

```
postgres@server:~$ createuser --interactive
```

Copy

Si, au contraire, vous préférez utiliser `sudo` pour chaque commande sans quitter votre compte normal, tapez :

```
$ sudo -u postgres createuser --interactive
```

Copy

Le script vous proposera quelques choix et, en fonction de vos réponses, exécutera les bonnes commandes Postgres pour créer un utilisateur selon vos spécifications.

```
Output  
Enter name of role to add: sammy  
Shall the new role be a superuser? (y/n) y
```

Vous pouvez obtenir plus de contrôle en passant des indicateurs supplémentaires. Consultez les options en consultant la page `man` :

```
$ man createuser
```

Copy

Votre installation Postgres a maintenant un nouvel utilisateur, mais vous n'avez encore ajouté aucune base de données. La section suivante décrit ce processus.

Étape 4 - Création d'une nouvelle base de données

Une autre hypothèse posée par défaut par le système d'authentification Postgres est que tout rôle utilisé pour se connecter disposera d'une base de données du même nom à laquelle il pourra accéder.

Cela signifie que si l'utilisateur que vous avez créé dans la dernière section est appelé **sammy**, ce rôle tentera de se connecter à une base de données qui est également appelée « sammy » par défaut. Vous pouvez créer la base de données appropriée avec la commande `createdb`.

Si vous êtes connecté en tant que compte **postgres**, vous devez taper quelque

chose comme :

```
postgres@server:~$ createdb sammy
```

[Copy](#)

Si, au contraire, vous préférez utiliser `sudo` pour chaque commande sans quitter votre compte normal, vous devez taper :

```
$ sudo -u postgres createdb sammy
```

[Copy](#)

Cette flexibilité offre de multiples façons de créer des bases de données au besoin.

Étape 5 – Ouverture d'une invite Postgres avec le nouveau rôle

Pour vous connecter avec une authentification basée sur `ident`, vous aurez besoin d'un utilisateur Linux portant le même nom que votre rôle et votre base de données Postgres.

Si vous n'avez pas d'utilisateur Linux correspondant disponible, vous pouvez en créer un avec la commande `adduser`. Vous devrez le faire à partir de votre compte non `root` avec priviléges `sudo` (c'est-à-dire sans être connecté en tant qu'utilisateur `postgres`) :

```
$ sudo adduser sammy
```

[Copy](#)

Une fois ce nouveau compte disponible, vous pouvez basculer sur ce dernier et vous connecter à la base de données en tapant :

```
$ sudo -i -u sammy  
$ psql
```

[Copy](#)

Ou alors, vous pouvez le faire en ligne :

```
$ sudo -u sammy psql
```

[Copy](#)

Cette commande vous connectera automatiquement, en supposant que tous les composants ont été correctement configurés.

Si vous souhaitez que votre utilisateur se connecte à une autre base de données, vous pouvez le faire en spécifiant la base de données comme ceci :

```
$ psql -d postgres
```

[Copy](#)

Une fois connecté, vous pouvez vérifier vos informations de connexion actuelles en tapant :

```
sammy=# \conninfo
```

[Copy](#)

Output

```
You are connected to database "sammy" as user "sammy" via socket in "/var/run/p
```

Cette commande est utile si vous vous connectez à des bases de données qui ne sont pas celles par défaut ou avec des utilisateurs qui ne sont pas ceux par défaut.

Étape 6 – Crédation et suppression de tables

Maintenant que vous savez comment vous connecter au système de base de données PostgreSQL, il est temps de découvrir quelques tâches de gestion basiques de Postgres.

La syntaxe de base pour la création de tables est la suivante :

```
CREATE TABLE table_name (
    column_name1 col_type (field_length) column_constraints,
    column_name2 col_type (field_length),
    column_name3 col_type (field_length)
);
```

Comme vous pouvez le voir, ces commandes donnent un nom à la table, puis définissent les colonnes ainsi que le type de colonne et la longueur maximale des données du champ. Vous pouvez aussi éventuellement ajouter des contraintes de table pour chaque colonne.

Vous pouvez en apprendre davantage sur [comment créer et gérer des tables dans Postgres](#) ici.

À des fins de démonstration, créez la table suivante :

```
sammy=# CREATE TABLE playground (
    equip_id serial PRIMARY KEY,
    type varchar (50) NOT NULL,
    color varchar (25) NOT NULL,
    location varchar(25) check (location in ('north', 'south', 'west',
    install_date date
    sammy=# );
```

Copy

Cette commande permet de créer une table d'inventaire des équipements de terrain de jeu. La première colonne du tableau contient les numéros d'identification des équipements de la `série` qui est un entier auto-incrémenté. Cette colonne a également la contrainte de `CLÉ PRIMAIRE` ce qui signifie que les valeurs qu'elle contient doivent être uniques et non nulles.

Les deux lignes suivantes créent respectivement des colonnes pour le `type` d'équipement et la `couleur`, aucune d'entre elles ne pouvant être vide. La ligne qui suit crée une colonne de `localisation` ainsi qu'une contrainte qui exige que la valeur soit l'une des huit valeurs possibles. La dernière ligne crée une colonne de `date` qui enregistre la date à laquelle vous avez installé l'équipement.

Pour deux des colonnes (`equip_id` et `install_date`), la commande ne spécifie pas de longueur de champ. La raison est que certains types de données n'exigent pas une longueur fixe parce que la longueur ou le format est implicite.

Vous pouvez voir votre nouvelle table en tapant :

```
sammy=# \d
```

Copy

Output

List of relations			
Schema	Name	Type	Owner
public	playground	table	sammy
public	playground_equip_id_seq	sequence	sammy

(2 rows)

Votre table « playground » est là, mais il y a aussi un élément nommé `playground_equip_id_seq` qui est du type `sequence`. Il s'agit d'une représentation du type `serial` que vous avez donné à votre colonne `equip_id`. Ceci permet de garder une trace du prochain numéro de la séquence et est créé automatiquement pour les colonnes de ce type.

Si vous voulez voir uniquement la table sans la séquence, vous pouvez taper :

```
sammy=# \dt
```

Copy

Output

Schema	Name	Type	Owner
public	playground	table	sammy

(1 row)

Avec une table prête, utilisons-la pour nous entraîner à la gestion des données.

Étape 7 - Ajout, requête et suppression de données dans une table

Maintenant que vous avez une table, vous pouvez y insérer des données. Par exemple, ajoutez un toboggan et une balançoire en appelant la table à laquelle vous voulez ajouter, en nommant les colonnes et en fournissant ensuite des données pour chaque colonne, comme ceci :

```
sammy=# INSERT INTO playground (type, color, location, install_date) VA Copy  
sammy=# INSERT INTO playground (type, color, location, install_date) VALUES ('s
```

Vous devez faire attention lors de la saisie des données afin d'éviter quelques problèmes courants. Pour commencer, les noms des colonnes ne doivent pas être mis entre guillemets, mais les valeurs des colonnes que vous saisissez doivent l'être.

Une autre chose à garder à l'esprit est que vous n'entrez pas de valeur pour la colonne `equip_id`. En effet, celle-ci est générée automatiquement chaque fois que vous ajoutez une nouvelle ligne à la table.

Récupérez les informations que vous avez ajoutées en tapant :

```
sammy=# SELECT * FROM playground;
```

Copy

Output

equip_id	type	color	location	install_date
1	slide	blue	south	2017-04-28
2	swing	yellow	northwest	2018-08-16

(2 rows)

Ici, vous pouvez voir que votre `equip_id` a bien été renseigné et que toutes vos autres données ont été organisées correctement.

Si le toboggan de l'aire de jeu se casse et que vous devez l'enlever, vous pouvez également enlever la ligne de votre table en tapant :

```
sammy=# DELETE FROM playground WHERE type = 'slide';
```

Copy

Interrogez à nouveau la table :

```
sammy=# SELECT * FROM playground;
```

Copy

Output

equip_id	type	color	location	install_date
2	swing	yellow	northwest	2018-08-16

(1 row)

Notez que la rangée de `toboggans` ne fait plus partie de la table.

Étape 8 - Ajout et suppression de colonnes dans une table

Après avoir créé une table, vous pouvez la modifier en ajoutant ou en supprimant des colonnes. Ajoutez une colonne pour indiquer la dernière visite de maintenance pour chaque équipement en tapant :

```
sammy=# ALTER TABLE playground ADD last_maint date;
```

[Copy](#)

Si vous consultez à nouveau les informations de votre table, vous constaterez que la nouvelle colonne a été ajoutée mais qu'aucune donnée n'a été saisie :

```
sammy=# SELECT * FROM playground;
```

[Copy](#)

Output

equip_id	type	color	location	install_date	last_maint
2	swing	yellow	northwest	2018-08-16	

(1 row)

Si vous constatez que vos collègues utilisent un outil distinct pour suivre l'historique de la maintenance, vous pouvez supprimer la colonne en tapant :

```
sammy=# ALTER TABLE playground DROP last_maint;
```

[Copy](#)

Cela supprime la colonne `last_maint` et toutes les valeurs qui s'y trouvent, mais laisse toutes les autres données intactes.

Étape 9 - Mise à jour des données dans une table

Jusqu'à présent, vous avez appris comment ajouter des enregistrements à une table et comment les supprimer, mais ce tutoriel n'a pas encore évoqué comment modifier des entrées existantes.

Vous pouvez mettre à jour les valeurs d'une entrée existante en interrogeant l'enregistrement que vous souhaitez et en paramétrant la colonne sur la valeur que vous souhaitez utiliser. Vous pouvez interroger l'enregistrement `balançoir` (cela correspondra à *chaque* balançoir de votre table) et changer sa couleur en `rouge`. Cela pourrait être utile si vous donnez au set de balançoirs une retouche de peinture:

```
sammy=# UPDATE playground SET color = 'red' WHERE type = 'swing';
```

[Copy](#)

Vous pouvez vérifier que l'opération a réussi en effectuant une nouvelle requête :

```
sammy=# SELECT * FROM playground;
```

[Copy](#)

Output

equip_id	type	color	location	install_date
2	swing	red	northwest	2018-08-16

(1 row)

Comme vous pouvez le voir, votre diapositive est maintenant enregistrée comme étant rouge.

Conclusion

Vous avez maintenant configuré PostgreSQL sur votre serveur Ubuntu 20.04. Si vous souhaitez en savoir plus sur Postgres et sur la manière de l'utiliser, nous vous invitons à consulter les guides suivants :

- [Une comparaison des systèmes de gestion de bases de données relationnelles](#)
- [S'entraîner à exécuter des requêtes avec SQL](#)

If you've enjoyed this tutorial and our broader community, consider checking out our DigitalOcean products which can also help you achieve your development goals.

[Learn more here →](#)

Get \$200 to try DigitalOcean - and do all the below for free!

Build applications, host websites, run open source software, learn cloud computing, and more – every cloud resource you need. If you've never tried DigitalOcean's products or services before, we'll cover your first \$200 in the next 60 days.

[Sign up now to activate this offer →](#)

About the authors



[Mark Drake](#) Author

Manager, Developer Education

Technical Writer @ DigitalOcean

Still looking for an answer?

[Ask a question](#)

[Search for more help](#)

Was this helpful?

[Yes](#)

[No](#)



Comments

Leave a comment

B I U S O H₁ H₂ H₃ ““ <>

Leave a comment...

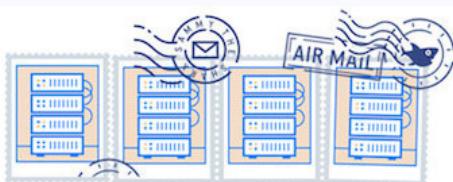
This textbox defaults to using [Markdown](#) to format your answer.

You can type `!ref` in this text area to quickly search our full set of tutorials, documentation & marketplace offerings and insert the link!

[Sign In or Sign Up to Comment](#)



This work is licensed under a Creative Commons Attribution-NonCommercial- ShareAlike 4.0 International License.



[GET OUR BIWEEKLY NEWSLETTER](#)

Sign up for Infrastructure as a Newsletter.



[HOLLIE'S HUB FOR GOOD](#)

Working on improving health and education, reducing inequality, and spurring economic growth?
We'd like to help.



[BECOME A CONTRIBUTOR](#)

You get paid; we donate to tech nonprofits.

Featured on Community Kubernetes Course Learn Python 3 Machine Learning in Python Getting started with Go Intro to Kubernetes

DigitalOcean Products Virtual Machines Managed Databases Managed Kubernetes Block Storage Object Storage Marketplace VPC Load Balancers



Search by Droplet name or IP (Cmd+B)

Create



\$0.00

Welcome to the developer cloud

DigitalOcean makes it simple to launch in the cloud and scale up as you grow – whether you're running one virtual machine or ten thousand.

[Learn More](#)

The screenshot shows the DigitalOcean developer cloud interface. At the top, there's a navigation bar with 'PROJECTS' (jonsmith), '+ New Project', 'MANAGE' (Droplets, Kubernetes, Volumes, Databases, Spaces, Images, Networking, Monitoring, API), 'DISCOVER' (Marketplace), and 'More Resources'. Below this is a 'Resources' section with tabs for 'Resources', 'Activity', and 'Settings'. It lists three 'SPACES': 'jon-db' (https://jon-db.nyc3.digitaloceanspaces.com), 'jon-website' (https://jon-website.nyc3.digitaloceanspaces.com), and 'jon-backups' (https://jon-backups.nyc3.digitaloceanspaces.com). There are also sections for 'Create something new' (Create a Droplet, Create a Managed Database, Spin up a Load Balancer) and 'Learn more' (Product Docs, Tutorials). A sidebar on the left shows 'PROJECTS' (jonsmith) and 'MANAGE' (Droplets, Kubernetes, Volumes, Databases, Spaces, Images, Networking, Monitoring, API).

Company	Products	Community	Solutions	Contact
About	Products Overview	Tutorials	Website Hosting	Support
Leadership	Droplets	Q&A	VPS Hosting	Sales
Blog	Kubernetes	CSS-Tricks	Web & Mobile Apps	Report Abuse
Careers	App Platform	Write for DOnations	Game Development	System Status
Customers	Functions	Currents Research	Streaming	Share your ideas
Partners	Cloudways	Hatch Startup Program	VPN	
Channel Partners	Managed Databases	deploy by DigitalOcean	SaaS Platforms	
Referral Program	Spaces	Shop Swag	Cloud Hosting for Blockchain	
Affiliate Program	Marketplace	Research Program	Startup Resources	
Press	Load Balancers	Open Source		
Legal	Block Storage	Code of Conduct		
Security	Tools & Integrations	Newsletter Signup		
Investor Relations	API	Meetups		
DO Impact	Pricing			
	Documentation			
	Release Notes			
	Uptime			



© 2023 DigitalOcean, LLC. All rights reserved.



COOKIE PREFERENCES