Im going to walk you through the process of setting up PostgreSQL on an Ubuntu EC2 instance and configuring restricted access via pgAdmin from your local machine. Let's get started!

**1: INTRODUCTION**

**Host:**  
Before we dive into the setup, let's quickly go over what we'll be doing. First, we'll launch an Ubuntu EC2 instance on AWS. Then, we'll install PostgreSQL on this instance. Finally, we'll configure the instance to allow restricted access to the PostgreSQL server using pgAdmin from our local machine.

**2: LAUNCHING UBUNTU EC2 INSTANCE**

**Host:**  
Alright, let's start by launching an Ubuntu EC2 instance.

1. **Log into your AWS Management Console.**
2. **Navigate to the EC2 dashboard.**
3. **Click on "Launch Instance."**
4. **Choose an Amazon Machine Image (AMI). For this demo, we'll use the latest Ubuntu Server.**
5. **Select an instance type. A t2.micro instance is sufficient for our setup.**
6. **Configure instance details. Make sure to allow the necessary ports like SSH for connecting to the instance.**
7. **Add storage, tags, and configure the security group. Allow SSH (port 22) and add a new rule for PostgreSQL (port 5432) from your local machine's IP address.**
8. **Review and launch the instance. Make sure to download the key pair for SSH access.**

3: CONNECTING TO THE INSTANCE

[PostgreSQL: Linux downloads (Ubuntu)](https://www.postgresql.org/download/linux/ubuntu/)

**5: CONFIGURING POSTGRESQL FOR REMOTE ACCESS]**

To allow pgAdmin on your local machine to connect to PostgreSQL on the EC2 instance, we need to modify a few configuration files.

1. **Edit the postgresql.conf file to listen on all IP addresses:**

sudo nano /etc/postgresql/16/main/postgresql.conf

1. **Find the line that starts with #listen\_addresses and change it to:**

listen\_addresses = '\*'

1. **Save and exit the file.**
2. **Edit the pg\_hba.conf file to allow connections from your local machine's IP address:**

sudo nano /etc/postgresql/16/main/pg\_hba.conf

1. **Add the following line at the end of the file:**

host all all your-local-ip/32 md5

1. **Save and exit the file.**
2. **Restart PostgreSQL to apply the changes:**

sudo systemctl restart postgresql

we'll create a new PostgreSQL user with a password and grant them the privileges to create databases. Let's dive right in!

**ACCESSING POSTGRESQL SHELL]**

**Host:**  
To switch to the PostgreSQL user and access the PostgreSQL shell, run the following commands:

1. **Switch to the PostgreSQL user:**

sudo -i -u postgres

1. **Access the PostgreSQL shell:**

Psql

Check Postgres sql connection

# \conninfo

You're now in the PostgreSQL shell. Let's create a new user with a password.

**[SECTION 3: CREATING A NEW USER]**  
To create a new PostgreSQL user, use the following SQL command. Replace new\_username with the desired username and new\_password with a strong password.

CREATE USER new\_username WITH PASSWORD 'new\_password';

CREATE USER aseem WITH PASSWORD 'aseem!@#$';

CREATE DATABASE test\_db;

**VERIFYING USER PERMISSIONS]**

**To check if a user exists in PostgreSQL on an EC2 instance, you can use the following SQL query:**

**SELECT \* FROM pg\_user WHERE usename = '** username**;**

Next, we'll grant this user the privileges to create databases.

**[SECTION 4: GRANTING PRIVILEGES]**

**Premissions to user :**

grant a user all privileges, including the ability to create databases and tables, you can use the **ALTER USER** command along with the **SUPERUSER** and **CREATEDB**

ALTER USER new\_username CREATEDB;

ALTER USER username WITH SUPERUSER CREATEDB;

Alter role indicates its succesfully

To grant the new user the privilege to create databases, use the following SQL command:

Now, the new user has the necessary privileges to create databases. Let's exit the PostgreSQL shell.

1. **Exit the PostgreSQL shell:**

\q

1. **Exit the PostgreSQL user session:**

Exit

Exit and restart   
sudo systemctl restart postgresql

**[SECTION 5: VERIFYING USER PERMISSIONS]**

1. **Log in to PostgreSQL using the new user:**

psql -U new\_username -h localhost -d postgres

1. **Enter the password when prompted.**
2. **Create a new database:**

CREATE DATABASE test\_db;

1. **List the databases to verify that test\_db was created:**

\l

1. **Exit the PostgreSQL shell:**

\q

You've successfully created a new PostgreSQL user with a password and granted them the privileges to create databases. If you found this video helpful, please give it a thumbs up, subscribe to our channel, and hit the bell icon to stay updated with our latest tutorials. If you have any questions or run into any issues, leave a comment below, and I'll be happy to help. Thanks for watching, and I'll see you in the next video!