How to setup Postgres SQL and setup monitoring using Prometheus and Grafana.

Step 1:

First we setup PostgreSql Server.

We create one EC2 instance and install PostgreSql on it. Login to our EC2 instance and perform below steps.

[ec2-user@ip-172-31-47-67 ~]$ **sudo yum update**

Last metadata expiration check: 1 day, 2:21:18 ago on Sun Dec 24 07:04:40 2023.

Dependencies resolved.

Nothing to do.

Complete!

[ec2-user@ip-172-31-47-67 ~]$ **sudo yum install postgresql15.x86\_64 postgresql15-server -y**

Step 2:

Now we initialize the database.

ec2-user@ip-172-31-47-67 ~]$ **sudo postgresql-setup --initdb**

\* Initializing database in '/var/lib/pgsql/data'

\* Initialized, logs are in /var/lib/pgsql/initdb\_postgresql.log

[ec2-user@ip-172-31-47-67 ~]$

Now we start and enable the PostgreSQL.

[ec2-user@ip-172-31-47-67 ~]$ sudo systemctl start postgresql

[ec2-user@ip-172-31-47-67 ~]$ sudo systemctl enable postgresql

Created symlink /etc/systemd/system/multi-user.target.wants/postgresql.service → /usr/lib/systemd/system/postgresql.service.

[ec2-user@ip-172-31-47-67 ~]$ sudo systemctl status postgresql

**Step 3:**

**Now we configure PostgreSQL.**

# 3.1. Set password for ssh postgres user and admin postgres database password

# # Change the ssh user password: sudo passwd postgres # Log in using the Postgres system account: su - postgres # Now, change the admin database password: psql -c "ALTER USER postgres WITH PASSWORD 'your-password';" exit

# Below are Primary Configuration and authentication File

# /var/lib/pgsql/data/postgresql.conf

# We can enable which IP address it need to listen.

# listen\_addresses = '\*' # what IP address(es) to listen on;

# /var/lib/pgsql/data/pg\_hba.conf

# # You can change ident as md5 To allow connections from absolutely any address with password authentication host all all 0.0.0.0/0 md5

# [ec2-user@ip-172-31-47-67 ~]$ sudo systemctl restart postgresql

# Step 4:

# Now we create user and database.

# # Connect to the PostgreSQL server as the Postgres user: sudo -i -u postgres psql # Create a new database user: CREATE USER your-username WITH PASSWORD 'password'; # Create a new database: CREATE DATABASE database\_name; # Grant all privileges on the database to the user: GRANT ALL PRIVILEGES ON DATABASE database\_name TO your-username; # To list all available PostgreSQL users and databases: \l

# We can test the connection as below.

# [ec2-user@ip-172-31-47-67 ~]$ psql -h localhost -U test -d testdb

# Password for user test:

# psql (15.4)

# Type "help" for help.

# testdb=> exit

# [ec2-user@ip-172-31-47-67 ~]$

# Step 4:

# Now we download “postgres exporter”

The 'postgres\_exporter' is a prometheus exporter for the PostgreSQL server metrics. It supports multiple versions of PostgreSQL such as 9.4, 9.5, 9.6, 10, 11, 12, 13, and 14. The 'postgres\_exporter' is a single binary file application, it's mainly written in Go.

You'll download the 'postgres\_exporter' in these steps on the PostgreSQL server.

4.1. First we create “postgres\_exporter”.

sudo useradd -M -r -s /sbin/nologin postgres\_exporter

[ec2-user@ip-172-31-47-67 ~]$ sudo useradd -M -r -s /sbin/nologin postgres\_exporter

[ec2-user@ip-172-31-47-67 ~]$

4.2: Now we download the postgre exporter package.

[ec2-user@ip-172-31-47-67 ~]$ mkdir postgre\_exporter

[ec2-user@ip-172-31-47-67 ~]$ cd postgre\_exporter/

[ec2-user@ip-172-31-47-67 postgre\_exporter]$ wget <https://github.com/prometheus-community/postgres_exporter/releases/download/v0.12.0-rc.0/postgres_exporter-0.12.0-rc.0.linux-amd64.tar.gz>

[ec2-user@ip-172-31-47-67 postgre\_exporter]$ ls -l

total 7932

-rw-r--r--. 1 ec2-user ec2-user 8118459 Aug 29 2022 postgres\_exporter-0.12.0-rc.0.linux-amd64.tar.gz

[ec2-user@ip-172-31-47-67 postgre\_exporter]$

[ec2-user@ip-172-31-47-67 postgre\_exporter]$ tar xvf postgres\_exporter\*.tar.gz

[ec2-user@ip-172-31-47-67 postgre\_exporter]$ ls -l

total 7932

drwxr-xr-x. 2 ec2-user ec2-user 60 Aug 29 2022 postgres\_exporter-0.12.0-rc.0.linux-amd64

-rw-r--r--. 1 ec2-user ec2-user 8118459 Aug 29 2022 postgres\_exporter-0.12.0-rc.0.linux-amd64.tar.gz

[ec2-user@ip-172-31-47-67 postgres\_exporter-0.12.0-rc.0.linux-amd64]$ sudo mkdir /opt/postgres\_exporter

[ec2-user@ip-172-31-47-67 postgres\_exporter-0.12.0-rc.0.linux-amd64]$ ls -l /opt/postgres\_exporter

total 0

[ec2-user@ip-172-31-47-67 postgres\_exporter-0.12.0-rc.0.linux-amd64]$ mv postgres\_exporter\*/\* /opt/postgres\_exporter

[ec2-user@ip-172-31-47-67 postgres\_exporter]$ pwd

/opt/postgres\_exporter

[ec2-user@ip-172-31-47-67 postgres\_exporter]$ ls -l

total 15388

-rw-r--r--. 1 ec2-user ec2-user 11357 Aug 29 2022 LICENSE

-rw-r--r--. 1 ec2-user ec2-user 70 Aug 29 2022 NOTICE

-rwxr-xr-x. 1 ec2-user ec2-user 15740257 Aug 29 2022 postgres\_exporter

[ec2-user@ip-172-31-47-67 postgres\_exporter]$

Step 5:

Now we configure postgres\_exporter

*cd /opt/postgres\_exporter*

Now we create .env file and add our database entry.

Ex:

# Format

#DATA\_SOURCE\_NAME=postgresql://username:password@localhost:5432/postgres?sslmode=disable

# Monitor all databases via postgres\_exporter

DATA\_SOURCE\_NAME="postgresql://postgres:strongpostgrespassword@localhost:5432/?sslmode=disable"

# Monitor specific databases on the PostgreSQL server

# DATA\_SOURCE\_NAME="postgresql://username:password@localhost:5432/database-name?sslmode=disable"

[ec2-user@ip-172-31-47-67 postgres\_exporter]$ cd /opt/postgres\_exporter

[ec2-user@ip-172-31-47-67 postgres\_exporter]$ sudo vi .env

[ec2-user@ip-172-31-47-67 postgres\_exporter]$ cat .env

# Monitor all databases via postgres\_exporter

DATA\_SOURCE\_NAME="postgresql://postgres:\*\*\*\*\*\*\*@localhost:5432/?sslmode=disable"

[ec2-user@ip-172-31-47-67 postgres\_exporter]$

# Now we change the ownership of file.

# sudo chown -R postgres\_exporter: /opt/postgres\_exporter

# Step 6:

# Now we setup service file for postgres\_exporter “*/etc/systemd/system/postgres\_exporter.service”.*

*[Unit]  
Description=Prometheus exporter for Postgresql  
Wants=network-online.target  
After=network-online.target  
  
[Service]  
User=postgres\_exporter  
Group=postgres\_exporter  
WorkingDirectory=/opt/postgres\_exporter  
EnvironmentFile=/opt/postgres\_exporter/.env  
ExecStart=/opt/postgres\_exporter/postgres\_exporter --web.listen-address=:9187 --web.telemetry-path=/metrics  
Restart=always  
  
[Install]  
WantedBy=multi-user.target*

# A computer screen with white text Description automatically generated

# Step 7:

# Now we enable the service.

# sudo systemctl daemon-reload

# sudo systemctl start postgres\_exporter

# sudo systemctl enable postgres\_exporter

# A black screen with text on it Description automatically generated

# Also we need to open the postgres\_exporter in EC2.

# To access metrics.

# http://<public IP>:9187/metrics

# A screenshot of a computer Description automatically generated

# Step 8:

# Now we add postgres\_exporter to Prometheus.

# Login to our Prometheus server and edit the “/etc/prometheus/prometheus.yml”.

Under the 'scrape\_configs' section, add the following lines to the file. In this example, you'll set up a new job for gathering PostgreSQL server metrics with the name '**postgres\_exporter**', and the target server of PostgreSQL is '**<public ip of postgresql server>:9187**', which is the metrics that are exposed by the 'postgres\_exporter' service.

  # PostgreSQL Monitoring with postgres\_exporter  
  - job\_name: 'postgres\_exporter'  
    scrape\_interval: 5s  
    static\_configs:  
      - targets: ['<public ip address of Postgresql server>:9187']

# Now restart the Prometheus service.

*sudo systemctl restart prometheus*

# Niow login to Prometheus site.

# <public Ip of Prometheus server>:9090/targers

# A screenshot of a computer Description automatically generated

# We can login to Grafana site.

# <public Ip of Prometheus server>:3000

# A screenshot of a computer Description automatically generated

# Now we can set dashboard for postgresql.

# Home -> dashboard -> New ->import ->template id ->import

# We can import template from “Grafana.com”. ‘9628’ is id for postgresql database template.

# A screenshot of a computer Description automatically generated

**Now we can see the dashboard of our database. We can save the dashboard.**

**A screenshot of a computer

Description automatically generated**

# Doc: <https://hbayraktar.medium.com/how-to-install-postgresql-15-on-amazon-linux-2023-a-step-by-step-guide-57eebb7ad9fc>

# https://www.howtoforge.com/how-to-monitor-postgresql-with-prometheus-and-grafana/#postgresql-server-configuration