

★ Member-only story

# Postgres Database Audit Policies



Oz · Following

3 min read · Mar 20, 2024



Listen



Share

... More

The PostgreSQL Audit (pgaudit) facilitates detailed session and object audit logging using PostgreSQL's standard logging facility. Its primary objective is to generate comprehensive audit logs necessary for regulatory compliance and security audits. Here's a breakdown of essential parameters and configurations within pgaudit:



## pgaudit.log\_catalog

- Enables session logging when all relations in a statement are within the pg\_catalog schema
- Reduces log noise from tools querying the catalog extensively (e.g., psql, PgAdmin)
- Default: On

### **pgaudit.log\_client**

- Determines visibility of log messages to client processes (e.g., psql)
- Typically disabled but useful for debugging purposes
- Note: Requires pgaudit.log\_level to be enabled
- Default: Off

### **pgaudit.log\_level**

- Specifies the log level for entries (excluding ERROR, FATAL, and PANIC)
- Useful for regression testing and end-user testing purposes
- Default: Log

### **pgaudit.log\_parameter**

- Controls inclusion of parameters passed with statements in audit logging
- Parameters are included in CSV format after the statement text.
- Default: Off

### **pgaudit.log\_relation**

- Enables logging of each relation referenced in SELECT or DML statements.
- Offers a shortcut for exhaustive logging without object audit logging.
- Default: Off

### **pgaudit.log\_rows**

- Specifies inclusion of retrieved or affected rows in audit logging
- Default: Off

### **pgaudit.log\_statement**

- Determines whether statement text and parameters are logged
- Enhances verbosity of logs.

- Default: On

### **pgaudit.log\_statement\_once**

- Controls logging frequency for statement text and parameters
- Default: Off

### **pgaudit.log\_parameter\_max\_size**

- Specifies the maximum size (in bytes) for logged parameter values
- Default: 0 (logs all parameters regardless of length)

### **pgaudit.role**

- Defines the master role for object audit logging
- Allows multiple audit roles for distinct auditing responsibilities

## **Installing and Configuring PostgreSQL Audit**

```
dnf install pgaudit14_12-1.4.3-1.rhel9.x86_64
```

Please configure your purposes as shown below

```
pgaudit.log: true
pgaudit.log_catalog: true
pgaudit.log_client: true
pgaudit.log_parameter: true
pgaudit.log_relation: true
pgaudit.log_statement_once: true
shared_preload_libraries: pg_stat_statements,pgaudit,powa,pg_stat_kcache,pg_stat_statements
```

```
pgaudit.log: true
pgaudit.log_catalog: true
pgaudit.log_client: true
pgaudit.log_parameter: true
pgaudit.log_relation: true
pgaudit.log_statement_once: true
restore_command: pgbackrest --stanza=cbs_backup archive-get %f "%p"
shared_buffers: 8GB
shared_preload_libraries: pg_stat_statements,pgaudit,powa,pg_stat_kcache,pg_qualstats,pg_wait_sampling
superuser_reserved_connections: 3
```

Example of patroni configuration

Configuring audit logging in PostgreSQL using the pgaudit extension is crucial for maintaining security and compliance standards within your database environment. Here's a step-by-step guide to setting up audit logging:

Before making any changes, it's essential to inspect the current settings using the

[Open in app](#) ↗

Medium

Search



```
SELECT name, setting FROM pg_settings WHERE name LIKE 'pgaudit%',
```

Use the `ALTER SYSTEM` command to adjust the audit logging settings based on your requirements. The following commands demonstrate different configurations:

```
# Enable logging for read operations only
ALTER SYSTEM SET pgaudit.log TO 'read';
SELECT pg_reload_conf();

# Enable logging for read and write operations
ALTER SYSTEM SET pgaudit.log TO 'read, write';

# Enable logging for all types of operations including miscellaneous and DDL (DDL)
ALTER SYSTEM SET pgaudit.log TO 'all, misc, ddl';

# Enable logging for all operations except miscellaneous statements
SET pgaudit.log = 'all, -misc';

# Define the master role for object audit logging
SET pgaudit.role = 'kema1.oz';
```

Adjust these settings according to your specific auditing requirements and security policies. Ensure that you reload the PostgreSQL configuration after making changes. For more detailed and technical articles like this, keep following our blog on

Medium. If you have any questions or need further assistance, feel free to reach out in the comments below and directly.



Following

## Written by Oz

149 Followers · 13 Following

Database Administrator 🐘

## No responses yet



Gvadakte

What are your thoughts?

## More from Oz



Oz

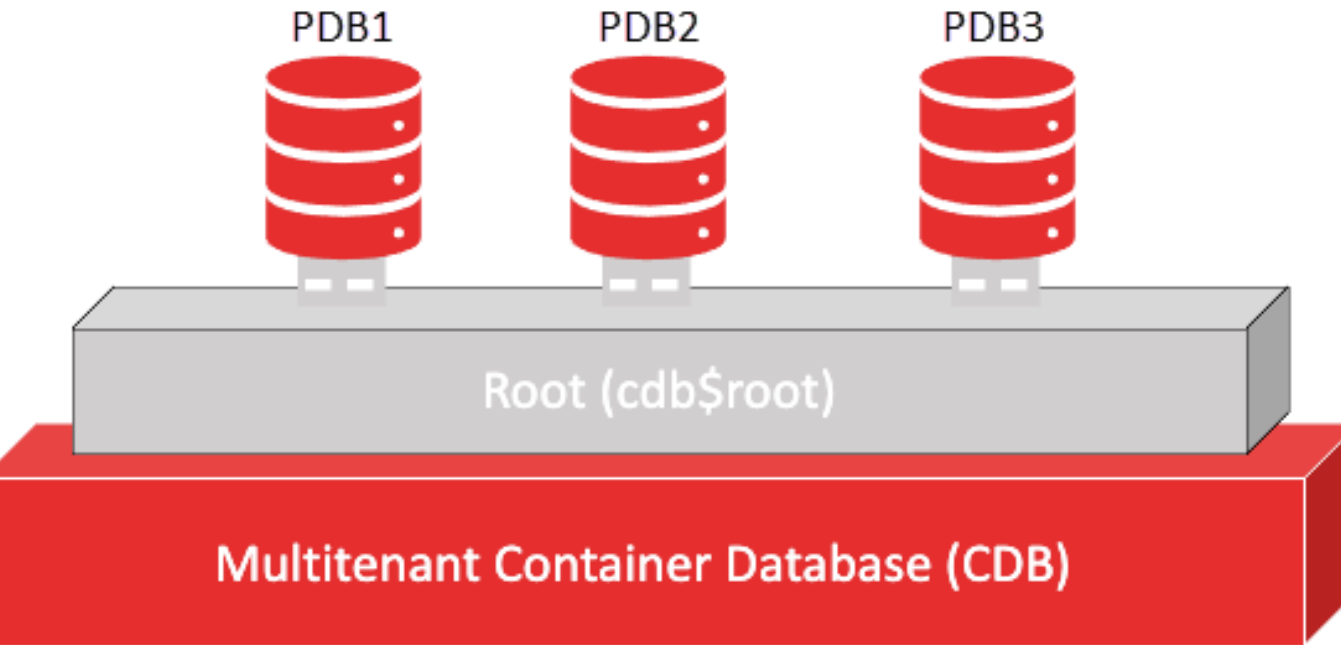
# Installing Percona Monitoring & Management (PMM) with Postgres

Introduction:

Sep 26, 2024

54

1

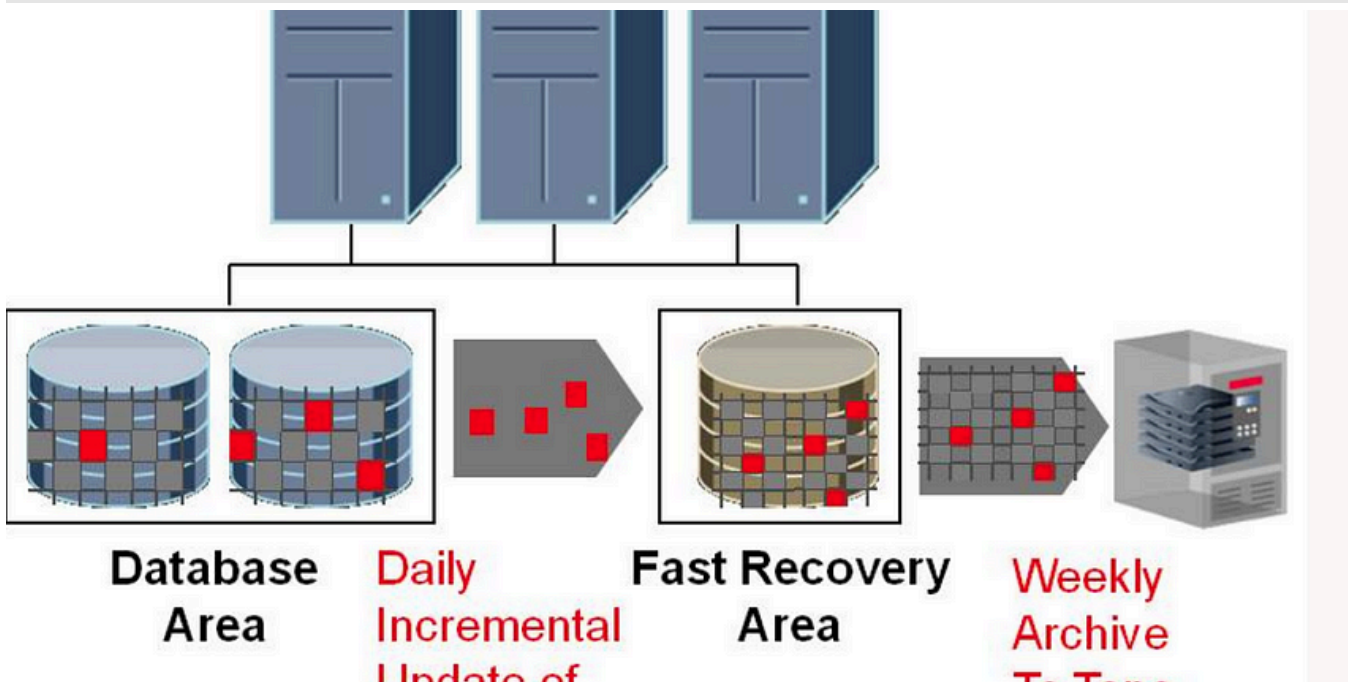


Oz

## Pluggable Database Command

----- - create pluggable database pdb1 admin user root identified by test123; alter pluggable database...

✦ May 12, 2023



Oz

## RMAN Backup Basic Commands

```
rman target / rman target sys/password@YDKTST; backup database; backup database format '/backup/path/%d_%t_%s.rman'; backup tablespace...
```

✦ May 11, 2023    👤 1



Oz

## delete jobs




★ May 8, 2023



See all from Oz

## Recommended from Medium



 Tihomir Manushev

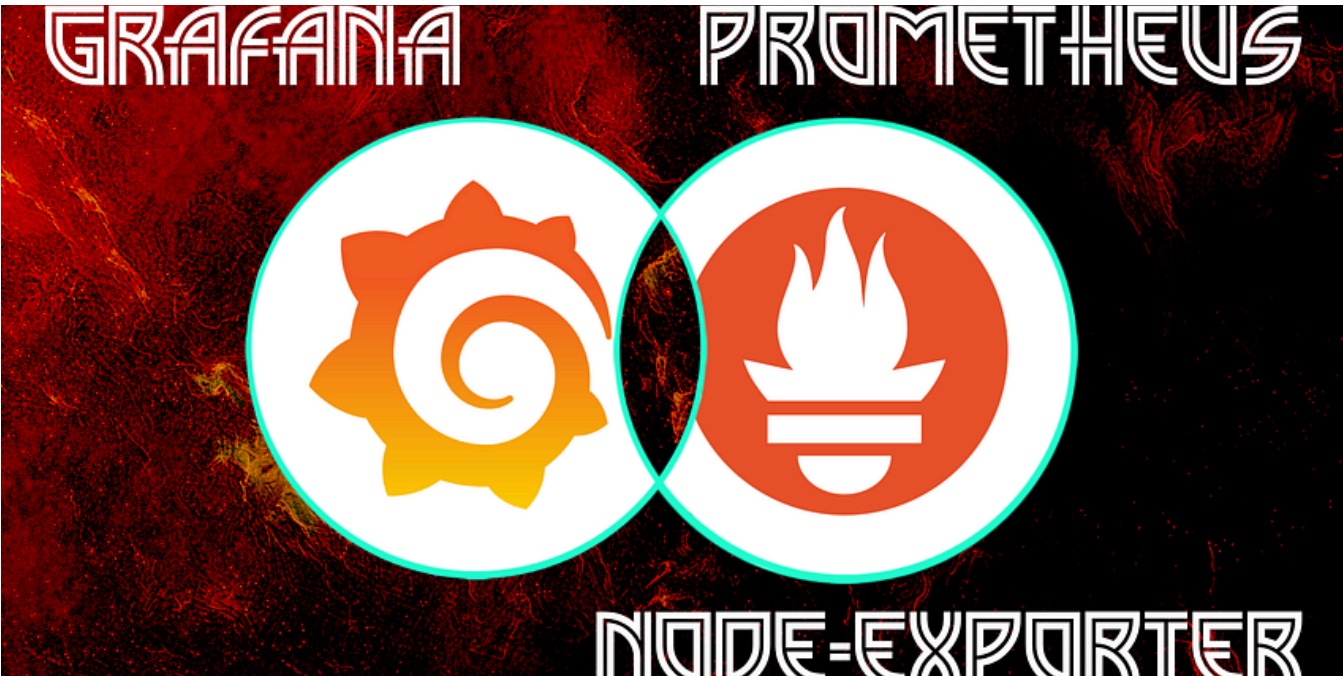
### Vector Search with pgvector in PostgreSQL


Simple AI-powered similarity search

★ Mar 9








 crptcpchk

### Grafana, Prometheus & Node-Exporter | Setup Guide

Grafana open source software enables you to query, visualize, alert on, and explore your metrics, logs, and traces wherever they are stored...

Oct 30, 2024


 8

 1





at it Means	Best Used For
e directly in the row	Simple data like INT
e in the row (unless large)	Larger types, but tri
mpress + store out-of-row	Long texts, large ob
e out-of-row, no compression	When compression

 Udbhav Singh

### Storages in PostgreSQL

What Are Storage Types in PostgreSQL — And Why Should You Care?

6d ago 18

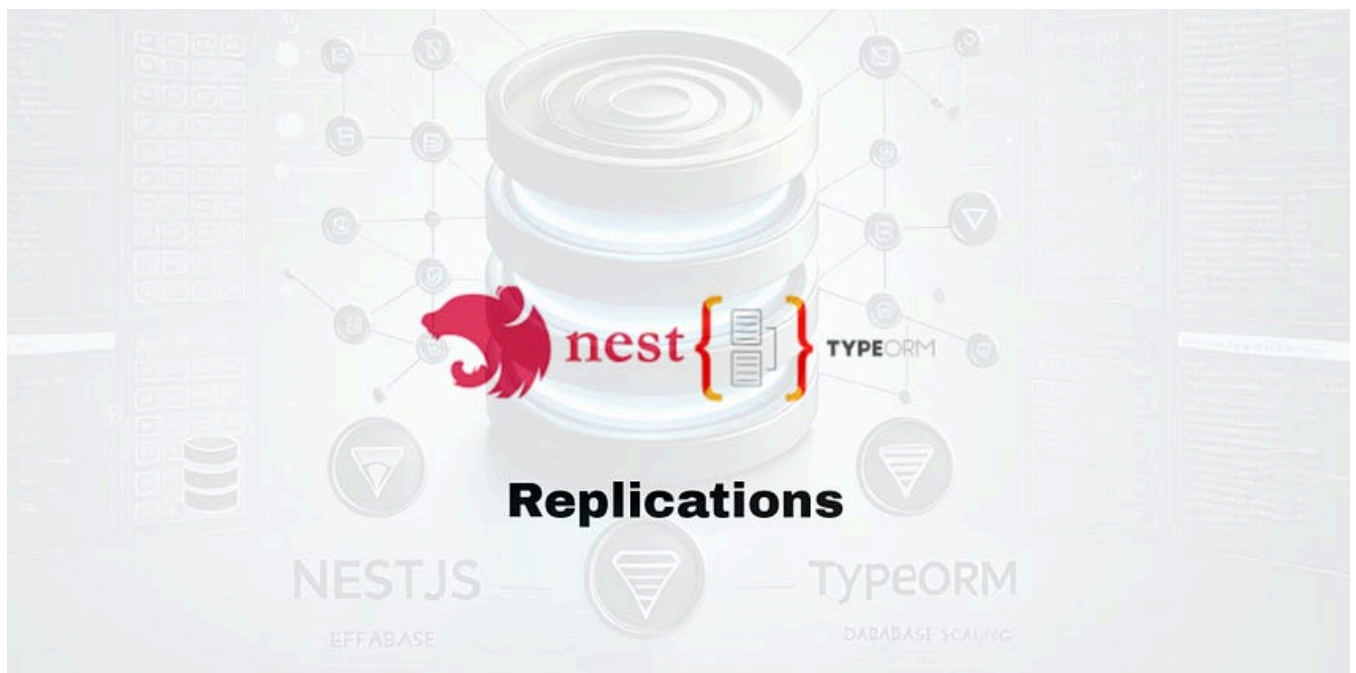


Dickson Gathima

## Patroni PostgreSQL High Availability with pgBackRest

In my previous post, I configured a 4-node PostgreSQL HA cluster to ensure high availability and failover capabilities. Now, I'd like to...

Mar 24 52 1



MD OZAIR QAYAM

## Efficient Database Scaling in NestJS with TypeORM: Best Practices for Enforcing Slave Reads

Scaling database reads is a critical challenge in modern applications. By default, NestJS with TypeORM sends all queries to the master...

Jan 31 🖱 8



# podman

## with PostgreSQL

@mehmetozanguven



mehmetozanguven

### Running PostgreSQL with Podman

Instead of running PostgreSQL locally, we can easily run with Podman. Here are the basic steps you should follow.

Mar 28 🖱 2



See more recommendations