Postgres Database Audit Policies

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

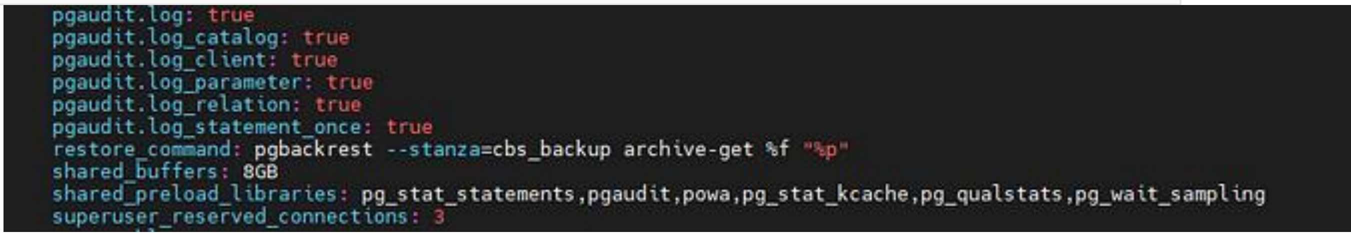
- Defines the master role for object audit logging
- Allows multiple audit roles for distinct auditing responsibilities

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_qualstats,pg_wait_sampling
```



```
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 SQL query:

```
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 (Data Definition Language) statements
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 = 'kemal.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.