Datadog – Database Monitoring

sudo apt update

sudo apt install postgresql-contrib

psql -h 127.0.0.1 -p 5432 -d postgres -U postgres

password: admin@123

**Find the Postgres config file:**

sudo -u postgres psql -c "SHOW config\_file;"

**/etc/postgresql/14/main/postgresql.conf**

sudo vi /etc/postgresql/14/main/postgresql.conf

# Enable pg\_stat\_statements extension

shared\_preload\_libraries = 'pg\_stat\_statements'

# Allow larger query texts to be captured

track\_activity\_query\_size = 4096

# Additional tuning for pg\_stat\_statements

pg\_stat\_statements.track = all

pg\_stat\_statements.max = 10000

pg\_stat\_statements.track\_utility = off

# Enable I/O timing

track\_io\_timing = on

**🔁 3. Restart PostgreSQL**

sudo systemctl restart postgresql

psql -h 127.0.0.1 -p 5432 -d postgres -U postgres

password: admin@123

[**Grant the Agent access**](https://docs.datadoghq.com/database_monitoring/setup_postgres/selfhosted?tab=postgres15#grant-the-agent-access)

CREATE USER datadog WITH password 'admin@123';

ALTER ROLE datadog INHERIT;

-- Enable UUID generation

CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

-- Enable query statistics

CREATE EXTENSION IF NOT EXISTS "pg\_stat\_statements";

-- Enable case-insensitive text

CREATE EXTENSION IF NOT EXISTS "citext";

CREATE SCHEMA datadog;

GRANT USAGE ON SCHEMA datadog TO datadog;

GRANT USAGE ON SCHEMA public TO datadog;

GRANT pg\_monitor TO datadog;

CREATE EXTENSION IF NOT EXISTS pg\_stat\_statements;

CREATE EXTENSION IF NOT EXISTS pg\_stat\_statements;

SELECT \* FROM pg\_stat\_statements LIMIT 5;

SHOW shared\_preload\_libraries;

SHOW track\_activity\_query\_size;

SHOW pg\_stat\_statements.track;

SHOW pg\_stat\_statements.max;

SHOW pg\_stat\_statements.track\_utility;

SHOW track\_io\_timing;

sudo -u postgres psql -d postgres -c "CREATE SCHEMA IF NOT EXISTS datadog;"

sudo -u postgres psql -d postgres -c "GRANT USAGE ON SCHEMA datadog TO datadog;"

sudo -u postgres psql -d postgres -c "GRANT USAGE ON SCHEMA public TO datadog;"

sudo -u postgres psql -d postgres -c "GRANT pg\_monitor TO datadog;"

sudo -u postgres psql -d postgres -c "CREATE EXTENSION IF NOT EXISTS pg\_stat\_statements;"

sudo -u postgres psql -d springbootdb -c "CREATE SCHEMA IF NOT EXISTS datadog;"

sudo -u postgres psql -d springbootdb -c "GRANT USAGE ON SCHEMA datadog TO datadog;"

sudo -u postgres psql -d springbootdb -c "GRANT USAGE ON SCHEMA public TO datadog;"

sudo -u postgres psql -d springbootdb -c "GRANT pg\_monitor TO datadog;"

sudo -u postgres psql -d springbootdb -c "CREATE EXTENSION IF NOT EXISTS pg\_stat\_statements;"

export PGPASSWORD='admin@123'

PGPASSWORD='admin@123' psql -U datadog -h 127.0.0.1 -d postgres -c "SELECT 1;"

**Create the function in every database to enable the Agent to collect explain plans.**

CREATE OR REPLACE FUNCTION datadog.explain\_statement(

l\_query TEXT,

OUT explain JSON

)

RETURNS SETOF JSON AS

$$

DECLARE

curs REFCURSOR;

plan JSON;

BEGIN

OPEN curs FOR EXECUTE pg\_catalog.concat('EXPLAIN (FORMAT JSON) ', l\_query);

FETCH curs INTO plan;

CLOSE curs;

RETURN QUERY SELECT plan;

END;

$$

LANGUAGE 'plpgsql'

RETURNS NULL ON NULL INPUT

SECURITY DEFINER;

To verify the permissions are correct, run the following commands to confirm the Agent user is able to connect to the database and read the core tables:

✅ Test database connection:

psql -h localhost -U datadog postgres -A \

-c "select \* from pg\_stat\_database limit 1;" \

&& echo -e "\e[0;32mPostgres connection - OK\e[0m" \

|| echo -e "\e[0;31mCannot connect to Postgres\e[0m"

✅ Test access to pg\_stat\_activity:

psql -h localhost -U datadog postgres -A \

-c "select \* from pg\_stat\_activity limit 1;" \

&& echo -e "\e[0;32mPostgres pg\_stat\_activity read OK\e[0m" \

|| echo -e "\e[0;31mCannot read from pg\_stat\_activity\e[0m"

✅ Test access to pg\_stat\_statements:

psql -h localhost -U datadog postgres -A \

-c "select \* from pg\_stat\_statements limit 1;" \

&& echo -e "\e[0;32mPostgres pg\_stat\_statements read OK\e[0m" \

|| echo -e "\e[0;31mCannot read from pg\_stat\_statements\e[0m"

sudo systemctl reload postgresql

**Modify pg\_hba.conf to use md5 authentication for datadog**

**vi /etc/postgresql/14/main/pg\_hba.conf**

# Allow the datadog user to connect using password (md5) — must be first!

host all datadog 127.0.0.1/32 md5

host all datadog ::1/128 md5

# Database administrative login by Unix domain socket

local all postgres peer

# TYPE DATABASE USER ADDRESS METHOD

# "local" is for Unix domain socket connections only

# local all all peer

local all all md5

# IPv4 local connections:

host all all 127.0.0.1/32 scram-sha-256

# IPv6 local connections:

host all all ::1/128 scram-sha-256

# Allow replication connections from localhost, by a user with the

# replication privilege.

# local replication all peer

host replication all 127.0.0.1/32 scram-sha-256

host replication all ::1/128 scram-sha-256

host all all 0.0.0.0/0 md5

sudo systemctl reload postgresql

sudo systemctl restart postgresql

sudo systemctl restart datadog-agent

sudo datadog-agent status | grep postgres -A 10

psql -h 127.0.0.1 -p 5432 -d postgres -U postgres

password: admin@123

[**Missing explain function**](https://docs.datadoghq.com/database_monitoring/setup_postgres/troubleshooting/#undefined-explain-function)

**Problem:** The Agent is unable to execute a required function in the datadog schema of the database.

Create the function **in every database** to enable the Agent to collect explain plans.

CREATE OR REPLACE FUNCTION datadog.explain\_statement(

l\_query TEXT,

OUT explain JSON

)

RETURNS SETOF JSON AS

$$

DECLARE

curs REFCURSOR;

plan JSON;

BEGIN

OPEN curs FOR EXECUTE pg\_catalog.concat('EXPLAIN (FORMAT JSON) ', l\_query);

FETCH curs INTO plan;

CLOSE curs;

RETURN QUERY SELECT plan;

END;

$$

LANGUAGE 'plpgsql'

RETURNS NULL ON NULL INPUT

SECURITY DEFINER;

🔹 1. Run in postgres database

sudo -u postgres psql -d postgres -c "

CREATE OR REPLACE FUNCTION datadog.explain\_statement(

l\_query TEXT,

OUT explain JSON

)

RETURNS SETOF JSON AS

\$\$

DECLARE

curs REFCURSOR;

plan JSON;

BEGIN

OPEN curs FOR EXECUTE pg\_catalog.concat('EXPLAIN (FORMAT JSON) ', l\_query);

FETCH curs INTO plan;

CLOSE curs;

RETURN QUERY SELECT plan;

END;

\$\$

LANGUAGE 'plpgsql'

RETURNS NULL ON NULL INPUT

SECURITY DEFINER;"

**🔹 2. Run in springbootdb database**

sudo -u postgres psql -d springbootdb -c "

CREATE OR REPLACE FUNCTION datadog.explain\_statement(

l\_query TEXT,

OUT explain JSON

)

RETURNS SETOF JSON AS

\$\$

DECLARE

curs REFCURSOR;

plan JSON;

BEGIN

OPEN curs FOR EXECUTE pg\_catalog.concat('EXPLAIN (FORMAT JSON) ', l\_query);

FETCH curs INTO plan;

CLOSE curs;

RETURN QUERY SELECT plan;

END;

\$\$

LANGUAGE 'plpgsql'

RETURNS NULL ON NULL INPUT

SECURITY DEFINER;"

**✅ Verify the Function (Optional)**

To check if it exists in both databases:

sudo -u postgres psql -d postgres -c "\df datadog.explain\_statement"

sudo -u postgres psql -d springbootdb -c "\df datadog.explain\_statement"

**sudo systemctl reload postgresql**

**sudo systemctl restart postgresql**

**sudo systemctl restart datadog-agent**

**sudo datadog-agent status | grep postgres -A 10**

sudo systemctl reload postgresql

sudo systemctl restart postgresql

sudo systemctl restart datadog-agent

sudo datadog-agent status | grep postgres -A 10

psql -h 127.0.0.1 -p 5432 -d postgres -U postgres

password: admin@123

[**Install the Agent**](https://docs.datadoghq.com/database_monitoring/setup_postgres/selfhosted?tab=postgres15#install-the-agent)

After you’ve installed the Host Agent, edit the Agent’s conf.d/postgres.d/conf.yaml file to point the Postgres instance you want to monitor. For a complete list of configuration

**vi /etc/datadog-agent/conf.d/postgres.d/conf.yaml**

init\_config:

instances:

- dbm: true # Enables Database Monitoring

host: 127.0.0.1 # Avoid Unix socket fallback

port: 5432

username: datadog

password: 'admin@123'

dbname: postgres

database\_autodiscovery:

enabled: true # Auto-discovers new databases

collect\_schemas:

enabled: true # Collect schema-level metrics

relations:

- relation\_regex: .\* # Collect metrics for all relations

query\_samples:

explain\_parameterized\_queries: true # Enable EXPLAIN for prepared statements

tags:

- 'env:prod'

- 'team:team-discovery'

- 'service:example-service'

sudo systemctl restart datadog-agent

sudo datadog-agent status | grep postgres -A 20

psql -h localhost -U datadog -d postgres -c "select \* from pg\_stat\_statements LIMIT 1;"

**sudo systemctl reload postgresql**

**sudo systemctl restart postgresql**

**sudo systemctl restart datadog-agent**

**sudo datadog-agent status | grep postgres -A 10**

sudo systemctl reload postgresql

sudo systemctl restart postgresql

sudo systemctl restart datadog-agent

sudo datadog-agent status | grep postgres -A 10

psql -h 127.0.0.1 -p 5432 -d postgres -U postgres

password: admin@123