

**Postgres Functions**

**Purpose:**

This document introduces a PostgreSQL function designed to inspect role memberships and associated privileges in a secure, auditable manner. The function supports:  
- Dry-run mode – Enables safe previewing of the intended query logic without executing any SQL.  
- Verbose mode – Provides detailed feedback on what the function would do, enhancing transparency.  
This tool is particularly useful for database administrators (DBAs) who need to audit, troubleshoot, or manage role memberships with confidence and control.

**Function Definition**

Attempting to use a DECLARE statement like:  
  
DECLARE role\_name TEXT := 'dba\_team';  
SELECT \* FROM pg\_roles WHERE rolname = role\_name;  
  
will not work in standard SQL execution outside of a function or DO block. PostgreSQL’s DECLARE syntax is only valid within PL/pgSQL procedural contexts—such as functions, DO $$ ... $$ blocks, or stored procedures. Outside those contexts, PostgreSQL expects standard SQL and will throw a syntax error if it encounters DECLARE.  
  
To encapsulate parameterized logic and return results properly, you must wrap the logic in a function or a DO block. That’s exactly what the get\_role\_members() function achieves: it encapsulates logic, handles inputs, supports dry-run and verbose output, and returns structured data—while remaining fully compliant with PostgreSQL’s procedural execution model.

**Function Logic**

The get\_role\_members function in PostgreSQL is designed to retrieve both a role's privileges and the privileges of its members, providing essential visibility into Role-Based Access Control (RBAC) structures. Built in PL/pgSQL, this function includes optional parameters for dry-run and verbose logging, making it ideal for secure, auditable inspections.

Example:

CREATE OR REPLACE FUNCTION get\_role\_members(

role\_name TEXT,

dry\_run BOOLEAN DEFAULT FALSE,

verbose BOOLEAN DEFAULT FALSE

)

RETURNS TABLE(

member\_name TEXT,

role\_name TEXT,

rolsuper BOOLEAN,

rolinherit BOOLEAN,

rolcreaterole BOOLEAN,

rolcreatedb BOOLEAN,

rolreplication BOOLEAN,

rolbypassrls BOOLEAN

) AS

$$

BEGIN

IF dry\_run THEN

IF verbose THEN

RAISE NOTICE 'DRY-RUN MODE ON - Query not executed.';

RAISE NOTICE 'SELECT FROM pg\_roles WHERE rolname = %', role\_name;

RAISE NOTICE 'JOIN pg\_auth\_members and pg\_roles to show membership.';

ELSE

RAISE NOTICE 'DRY-RUN: Would retrieve role and membership info for role: %', role\_name;

END IF;

RETURN;

END IF;

RETURN QUERY

-- Role itself

SELECT

rolname AS member\_name,

rolname AS role\_name,

rolsuper,

rolinherit,

rolcreaterole,

rolcreatedb,

rolreplication,

rolbypassrls

FROM

pg\_roles

WHERE

rolname = role\_name

UNION ALL

-- Members of the role

SELECT

member.rolname AS member\_name,

role.rolname AS role\_name,

member.rolsuper,

member.rolinherit,

member.rolcreaterole,

member.rolcreatedb,

member.rolreplication,

member.rolbypassrls

FROM

pg\_auth\_members m

JOIN pg\_roles role ON m.roleid = role.oid

JOIN pg\_roles member ON m.member = member.oid

WHERE

role.rolname = role\_name;

END;

$$ LANGUAGE plpgsql;

**Usage Summary**

- Basic Usage  
 SELECT \* FROM get\_role\_members('your\_role\_name');  
 Retrieves the specified role and its members, along with their privilege attributes.  
  
- Dry-Run Mode  
 SELECT \* FROM get\_role\_members('your\_role\_name', TRUE);  
 Simulates the query without executing it. Useful for safe previews.  
  
- Verbose Dry-Run  
 SELECT \* FROM get\_role\_members('your\_role\_name', TRUE, TRUE);  
 Provides detailed notices about what the function would do, without executing any queries.

**Use Cases**

- Auditing role privileges and memberships  
- Verifying changes before applying them  
- Troubleshooting access issues  
- Enhancing security reviews and compliance audits

**TL;DR**

- Use get\_role\_members('role\_name') to list the role and its members with their privileges.  
- Use dry\_run = TRUE to preview actions without running the query.  
- Use verbose = TRUE with dry-run to view detailed notices.  
- Ideal for auditing, troubleshooting, and safely managing role memberships.

**References**

[PostgreSQL Role Management Documentation](https://www.postgresql.org/docs/current/user-manag.html)

[PostgreSQL pg\_roles System Catalog](https://www.postgresql.org/docs/current/catalog-pg-roles.html)

[PostgreSQL pg\_auth\_members System Catalog](https://www.postgresql.org/docs/current/catalog-pg-auth-members.html)

[PostgreSQL PL/pgSQL Language Reference](https://www.postgresql.org/docs/current/plpgsql.html)