# deploy.SetRolePermissions – How-To Guide

Version: 2.0  
Date: 2025-08-01  
Page: 1 of 1

## Summary

Creates PostgreSQL roles if missing, optionally assigns users to them, and applies privileges on schemas, tables, and functions. Can run as dry-run or live execution.

## Usage Instructions

Run via psql using:  
  
 SELECT deploy.SetRolePermissions(  
 p\_role\_names,  
 p\_users\_to\_assign,  
 p\_apply\_schema\_privs\_to\_roles,  
 p\_table\_privileges,  
 p\_function\_privileges,  
 p\_apply\_to\_existing\_tables,  
 p\_set\_default\_for\_future\_tables,  
 p\_set\_default\_for\_future\_functions,  
 p\_revoke\_all\_first,  
 p\_execute\_flag  
 );

## Examples

1. Create role + assign user:  
 SELECT deploy.SetRolePermissions(ARRAY['reporting'], ARRAY['bob'], NULL, NULL, NULL, FALSE, FALSE, FALSE, FALSE, TRUE);  
  
2. Grant SELECT on all tables in schema:  
 SELECT deploy.SetRolePermissions(ARRAY['reader'], NULL, ARRAY['public'], ARRAY['SELECT'], NULL, TRUE, FALSE, FALSE, FALSE, TRUE);  
  
3. Grant EXECUTE on functions (future only):  
 SELECT deploy.SetRolePermissions(ARRAY['exec\_role'], NULL, ARRAY['api'], NULL, ARRAY['EXECUTE'], FALSE, FALSE, TRUE, FALSE, TRUE);

## Parameters

- `p\_role\_names`: TEXT[] – Roles to create/grant privileges to  
- `p\_users\_to\_assign`: TEXT[] – Users to assign to those roles  
- `p\_apply\_schema\_privs\_to\_roles`: TEXT[] – Target schemas  
- `p\_table\_privileges`: TEXT[] – Privileges like SELECT, INSERT  
- `p\_function\_privileges`: TEXT[] – e.g., EXECUTE  
- `p\_apply\_to\_existing\_tables`: BOOL – Apply to current tables  
- `p\_set\_default\_for\_future\_tables`: BOOL – Future table privileges  
- `p\_set\_default\_for\_future\_functions`: BOOL – Future function privileges  
- `p\_revoke\_all\_first`: BOOL – Revoke all before re-granting  
- `p\_execute\_flag`: BOOL – TRUE to run, FALSE to dry-run

## TL;DR

Create or update roles and their privileges across schemas. Set to dry-run for audit, TRUE for live action.