PostgreSQL Procedure Privileges and Security

This document explains the behavior of PostgreSQL procedures and their privilege management, including the difference between SECURITY INVOKER and SECURITY DEFINER modes, and how they influence procedure execution rights. This will clarify why a user might be able to execute a procedure without explicitly granted EXECUTE privileges and how to restrict execution rights if necessary.

# 1. Rule about Procedure Privileges

By default, when a procedure is created in PostgreSQL, the following behavior occurs:  
  
1. The creator (e.g., user1) of the procedure is considered the owner of the procedure.  
2. No one except the owner or superusers can execute the procedure unless EXECUTE privilege is explicitly granted to other users.

# 2. Scenario: Procedure Execution Without Explicit EXECUTE Privileges

In this scenario, we are working with two users: user1 and user2. The procedure 'proc\_1' is created by user1. User2 has been granted SELECT and INSERT privileges on a table named 'table\_1' but has not been explicitly granted EXECUTE privileges on the procedure. Despite this, user2 is able to execute the procedure. Let’s break down the steps:  
  
Step 1: Creating the Procedure  
User1 creates the procedure 'proc\_1' on 'table\_1'. Since user1 is the owner, user1 has the EXECUTE privilege on this procedure.  
  
Step 2: Creating User2  
User2 is created and granted SELECT and INSERT privileges on 'table\_1', which allows user2 to select and insert data in 'table\_1'.  
  
Step 3: Granting Privileges on the Table  
As user1, SELECT and INSERT privileges are granted to user2 on 'table\_1':  
 GRANT SELECT, INSERT ON table\_1 TO user2;  
Now, user2 can access 'table\_1' using SELECT and INSERT commands.  
  
Step 4: Calling the Procedure  
Despite user2 not being explicitly granted EXECUTE privilege on 'proc\_1', user2 is able to call and execute the procedure successfully.

# 3. Why Did the Procedure Work Without Explicit EXECUTE Privilege?

This behavior occurs because PostgreSQL procedures default to SECURITY INVOKER mode, unless otherwise specified. Here's how it works:  
  
SECURITY INVOKER Behavior (Default)  
Procedures execute with the privileges of the user who calls them. In this case, when user2 calls the procedure, PostgreSQL checks:  
- Does user2 have the EXECUTE privilege on the procedure?  
 Answer: No, but the database does not block execution because the procedure runs under SECURITY INVOKER.  
- Does user2 have the necessary privileges on the objects (like table\_1) the procedure interacts with?  
 Answer: Yes, since SELECT and INSERT privileges were granted on table\_1.  
  
Because the procedure executes under user2's privileges (SECURITY INVOKER mode), and user2 has the necessary table privileges, PostgreSQL allows the procedure to execute.

# 4. How to Restrict Procedure Execution

If you want to restrict the execution of a procedure to specific users (e.g., only user1 or users explicitly granted access), you can control the EXECUTE privilege and potentially use SECURITY DEFINER mode.

Solution 1: Explicitly Control EXECUTE Privilege

Revoke execution rights for all users (public):  
 REVOKE EXECUTE ON PROCEDURE proc\_1 FROM PUBLIC;

Grant EXECUTE privilege to specific users (e.g., user2):  
 GRANT EXECUTE ON PROCEDURE proc\_1 TO user2;

Solution 2: Use SECURITY DEFINER

Modify the procedure to execute with the privileges of the owner (user1):  
 CREATE OR REPLACE PROCEDURE proc\_1()  
 LANGUAGE plpgsql  
 SECURITY DEFINER  
 AS $$  
 BEGIN  
 -- Procedure logic here  
 END;  
 $$;

With SECURITY DEFINER, the EXECUTE privilege is checked for the procedure only, and any underlying object privileges are based on the owner's privileges (user1).

# 5. Summary

By default, a procedure is created with SECURITY INVOKER mode and without granting EXECUTE to others. However:  
  
- If a user has sufficient privileges on the underlying objects accessed by the procedure, the procedure will execute successfully, because the procedure uses the caller’s privileges.  
- To restrict execution, you must:  
 1. Explicitly manage EXECUTE privileges.  
 2. Use SECURITY DEFINER to restrict execution rights and ensure that only authorized users can execute the procedure.