revoke

- Description
- Notes
- Examples
- Compatibility
- See Also

REVOKE

REVOKE

7

SQL - Language Statements

REVOKE

remove access privileges

REVOKE [GRANT OPTION FOR] { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | TRIGGER | MAINTAIN } [, ...] | ALL [PRIVILEGES] } ON { [TABLE]

table_name

[, ...] | ALL TABLES IN SCHEMA

schema_name

[, ...] } FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { { SELECT | INSERT | UPDATE | REFERENCES } (

column_name

[, ...]) [, ...] | ALL [PRIVILEGES] (

column_name

[, ...]) } ON [TABLE]

table_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { { USAGE | SELECT | UPDATE } [, ...] | ALL [PRIVILEGES] } ON { SEQUENCE

sequence_name

[, ...] | ALL SEQUENCES IN SCHEMA

schema_name

[, ...] } FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { { CREATE | CONNECT | TEMPORARY | TEMP } [, ...] | ALL [PRIVILEGES] } ON DATABASE

database_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { USAGE | ALL [PRIVILEGES] } ON DOMAIN

domain_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { USAGE | ALL [PRIVILEGES] } ON FOREIGN DATA WRAPPER

fdw_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { USAGE | ALL [PRIVILEGES] } ON FOREIGN SERVER

server_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { EXECUTE | ALL [PRIVILEGES] } ON { { FUNCTION | PROCEDURE | ROUTINE }

function_name

[([[

argmode

] [

arg_name

]

arg_type

[, ...]])] [, ...] | ALL { FUNCTIONS | PROCEDURES | ROUTINES } IN SCHEMA

schema_name

[, ...] } FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { USAGE | ALL [PRIVILEGES] } ON LANGUAGE

lang_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { { SELECT | UPDATE } [, ...] | ALL [PRIVILEGES] } ON LARGE OBJECT

loid

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { { SET | ALTER SYSTEM } [, ...] | ALL [PRIVILEGES] } ON PARAMETER

configuration_parameter

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { { CREATE | USAGE } [, ...] | ALL [PRIVILEGES] } ON SCHEMA

schema_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { CREATE | ALL [PRIVILEGES] } ON TABLESPACE

tablespace_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [GRANT OPTION FOR] { USAGE | ALL [PRIVILEGES] } ON TYPE

type_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT] REVOKE [{ ADMIN | INHERIT | SET } OPTION FOR]

role_name

[, ...] FROM

role_specification

[, ...] [GRANTED BY

role_specification

] [CASCADE | RESTRICT]

where

role_specification

can be:

[GROUP]

role_name

| PUBLIC | CURRENT_ROLE | CURRENT_USER | SESSION_USER

Description

The `REVOKE` command revokes previously granted privileges from one or more roles. The key word `PUBLIC` refers to the implicitly defined group of all roles.

See the description of the `GRANT` command for the meaning of the privilege types.

Note that any particular role will have the sum of privileges granted directly to it, privileges granted to any role it is presently a member of, and privileges granted to `PUBLIC`. Thus, for example, revoking `SELECT` privilege from `PUBLIC` does not necessarily mean that all roles have lost `SELECT` privilege on the object: those who have it granted directly or via another role will still have it. Similarly, revoking `SELECT` from a user might not prevent that user from using `SELECT` if `PUBLIC` or another membership role still has `SELECT` rights.

If `GRANT OPTION FOR` is specified, only the grant option for the privilege is revoked, not the privilege itself. Otherwise, both the privilege and the grant option are revoked.

If `GRANTED BY` is specified, only privileges granted by the specified role are revoked. A role can only revoke grants by another role if it inherits the privileges of that role.

If a user holds a privilege with grant option and has granted it to other users then the privileges held by those other users are called dependent privileges. If the privilege or the grant option held by the first user is being revoked and dependent privileges exist, those dependent privileges are also revoked if `CASCADE` is specified; if it is not, the revoke action will fail. This recursive revocation only affects privileges that were granted through a chain of users that is traceable to the user that is the subject of this `REVOKE` command. Thus, the affected users might effectively keep the privilege if it was also granted through other users.

When revoking privileges on a table, the corresponding column privileges (if any) are automatically revoked on each column of the table, as well. On the other hand, if a role has been granted privileges on a table, then revoking the same privileges from individual columns will have no effect.

When revoking membership in a role, `GRANT OPTION` is instead called `ADMIN OPTION`, but the behavior is similar. Note that, in releases prior to PostgreSQL 16, dependent privileges were not tracked for grants of role membership, and thus `CASCADE` had no effect for role membership. This is no longer the case. Note also that this form of the command does not allow the noise word `GROUP` in <role_specification>.

Just as `ADMIN OPTION` can be removed from an existing role grant, it is also possible to revoke `INHERIT OPTION` or `SET OPTION`. This is equivalent to setting the value of the corresponding option to `FALSE`.

Notes

A user can only revoke privileges that were granted directly by that user. If, for example, user A has granted a privilege with grant option to user B, and user B has in turn granted it to user C, then user A cannot revoke the privilege directly from C. Instead, user A could revoke the grant option from user B and use the `CASCADE` option so that the privilege is in turn revoked from user C. For another example, if both A and B have granted the same privilege to C, A can revoke their own grant but not B's grant, so C will still effectively have the privilege.

When a non-owner of an object attempts to `REVOKE` privileges on the object, the command will fail outright if the user has no privileges whatsoever on the object. As long as some privilege is available, the command will proceed, but it will revoke only those privileges for which the user has grant options. The `REVOKE ALL PRIVILEGES` forms will issue a warning message if no grant options are held, while the other forms will issue a warning if grant options for any of the privileges specifically named in the command are not held. (In principle these statements apply to the object owner as well, but since the owner is always treated as holding all grant options, the cases can never occur.)

If a superuser chooses to issue a `GRANT` or `REVOKE` command, the command is performed as though it were issued by the owner of the affected object. (Since roles do not have owners, in the case of a `GRANT` of role membership, the command is performed as though it were issued by the bootstrap

superuser.) Since all privileges ultimately come from the object owner (possibly indirectly via chains of grant options), it is possible for a superuser to revoke all privileges, but this might require use of `CASCADE` as stated above.

`REVOKE` can also be done by a role that is not the owner of the affected object, but is a member of the role that owns the object, or is a member of a role that holds privileges `WITH GRANT OPTION` on the object. In this case the command is performed as though it were issued by the containing role that actually owns the object or holds the privileges `WITH GRANT OPTION`. For example, if table `t1` is owned by role `g1`, of which role `u1` is a member, then `u1` can revoke privileges on `t1` that are recorded as being granted by `g1`. This would include grants made by `u1` as well as by other members of role `g1`.

If the role executing `REVOKE` holds privileges indirectly via more than one role membership path, it is unspecified which containing role will be used to perform the command. In such cases it is best practice to use `GRANTED BY` to specify which role you want to do the `REVOKE` as. Failure to do so might lead to revoking privileges other than the ones you intended, or not revoking anything at all.

See ??? for more information about specific privilege types, as well as how to inspect objects' privileges.

Examples

Revoke insert privilege for the public on table `films`:

```
REVOKE INSERT ON films FROM PUBLIC;
```

Revoke all privileges from user `manuel` on view `kinds`:

```
REVOKE ALL PRIVILEGES ON kinds FROM manuel;
```

Note that this actually means “revoke all privileges that I granted”.

Revoke membership in role `admins` from user `joe`:

```
REVOKE admins FROM joe;
```

Compatibility

The compatibility notes of the `GRANT` command apply analogously to `REVOKE`. The keyword `RESTRICT` or `CASCADE` is required according to the standard, but PostgreSQL assumes `RESTRICT` by default.

See Also

□□□ #2
□ □□□ □□□ 2026-10-31 14:14:26 UTC
□ □□□ □□□ 2026-10-31 14:14:27 UTC