NAME

CREATE_SCHEMA − define a new schema

SYNOPSIS

CREATE SCHEMA *schema_name* [AUTHORIZATION *role_specification*] [*schema_element* [...]]
CREATE SCHEMA AUTHORIZATION *role_specification* [*schema_element* [...]]
CREATE SCHEMA IF NOT EXISTS *schema_name* [AUTHORIZATION *role_specification*]
CREATE SCHEMA IF NOT EXISTS AUTHORIZATION *role_specification*

where *role_specification* can be:

user_name
| CURRENT_ROLE
| CURRENT_USER
| SESSION_USER

DESCRIPTION

CREATE SCHEMA enters a new schema into the current database. The schema name must be distinct from the name of any existing schema in the current database.

A schema is essentially a namespace: it contains named objects (tables, data types, functions, and operators) whose names can duplicate those of other objects existing in other schemas. Named objects are accessed either by "qualifying" their names with the schema name as a prefix, or by setting a search path that includes the desired schema(s). A CREATE command specifying an unqualified object name creates the object in the current schema (the one at the front of the search path, which can be determined with the function **current_schema**).

Optionally, **CREATE SCHEMA** can include subcommands to create objects within the new schema. The subcommands are treated essentially the same as separate commands issued after creating the schema, except that if the AUTHORIZATION clause is used, all the created objects will be owned by that user.

PARAMETERS

schema_name
The name of a schema to be created. If this is omitted, the *user_name* is used as the schema name. The name cannot begin with pg_, as such names are reserved for system schemas.

user_name
The role name of the user who will own the new schema. If omitted, defaults to the user executing the command. To create a schema owned by another role, you must be able to SET ROLE to that role.

schema_element
An SQL statement defining an object to be created within the schema. Currently, only **CREATE TABLE**, **CREATE VIEW**, **CREATE INDEX**, **CREATE SEQUENCE**, **CREATE TRIGGER** and **GRANT** are accepted as clauses within **CREATE SCHEMA**. Other kinds of objects may be created in separate commands after the schema is created.

IF NOT EXISTS
Do nothing (except issuing a notice) if a schema with the same name already exists. *schema_element* subcommands cannot be included when this option is used.

NOTES

To create a schema, the invoking user must have the CREATE privilege for the current database. (Of course, superusers bypass this check.)

EXAMPLES

Create a schema:

```
CREATE SCHEMA myschema;
```

Create a schema for user joe; the schema will also be named joe:

```
CREATE SCHEMA AUTHORIZATION joe;
```

Create a schema named test that will be owned by user joe, unless there already is a schema named test. (It does not matter whether joe owns the pre−existing schema.)

```
CREATE SCHEMA IF NOT EXISTS test AUTHORIZATION joe;
```

Create a schema and create a table and view within it:

```
CREATE SCHEMA hollywood
  CREATE TABLE films (title text, release date, awards text[])
  CREATE VIEW winners AS
    SELECT title, release FROM films WHERE awards IS NOT NULL;
```

Notice that the individual subcommands do not end with semicolons.

The following is an equivalent way of accomplishing the same result:

```
CREATE SCHEMA hollywood;
CREATE TABLE hollywood.films (title text, release date, awards text[]);
CREATE VIEW hollywood.winners AS
  SELECT title, release FROM hollywood.films WHERE awards IS NOT NULL;
```

COMPATIBILITY

The SQL standard allows a DEFAULT CHARACTER SET clause in **CREATE SCHEMA**, as well as more subcommand types than are presently accepted by PostgreSQL.

The SQL standard specifies that the subcommands in **CREATE SCHEMA** can appear in any order. The present PostgreSQL implementation does not handle all cases of forward references in subcommands; it might sometimes be necessary to reorder the subcommands in order to avoid forward references.

According to the SQL standard, the owner of a schema always owns all objects within it. PostgreSQL allows schemas to contain objects owned by users other than the schema owner. This can happen only if the schema owner grants the CREATE privilege on their schema to someone else, or a superuser chooses to create objects in it.

The IF NOT EXISTS option is a PostgreSQL extension.

SEE ALSO

ALTER SCHEMA (**ALTER_SCHEMA**(7)), DROP SCHEMA (**DROP_SCHEMA**(7))