CREATE TABLE AS

Name

CREATE TABLE AS -- define a new table from the results of a query

Synopsis

```
CREATE [ [ GLOBAL | LOCAL ] { TEMPORARY | TEMP } ] TABLE table_name
    [ (column_name [, ...] ) ] [ [ WITH | WITHOUT ] OIDS ]
    AS query
```

Description

`CREATE TABLE AS` creates a table and fills it with data computed by a `SELECT` command or an `EXECUTE` that runs a prepared `SELECT` command. The table columns have the names and data types associated with the output columns of the `SELECT` (except that you can override the column names by giving an explicit list of new column names).

`CREATE TABLE AS` bears some resemblance to creating a view, but it is really quite different: it creates a new table and evaluates the query just once to fill the new table initially. The new table will not track subsequent changes to the source tables of the query. In contrast, a view re-evaluates its defining `SELECT` statement whenever it is queried.

Parameters

`GLOBAL or LOCAL`

Ignored for compatibility. Refer to **CREATE TABLE** for details.

`TEMPORARY or TEMP`

If specified, the table is created as a temporary table. Refer to **CREATE TABLE** for details.

table_name

The name (optionally schema-qualified) of the table to be created.

column_name

The name of a column in the new table. If column names are not provided, they are taken from the output column names of the query. If the table is created from an `EXECUTE` command, a column name list cannot be specified.

`WITH OIDS`
`WITHOUT OIDS`

This optional clause specifies whether the table created by `CREATE TABLE AS` should include OIDs. If neither form of this clause is specified, the value of the **default_with_oids** configuration parameter is used.

query

A query statement (that is, a `SELECT` command or an `EXECUTE` command that runs a prepared `SELECT` command). Refer to **SELECT** or **EXECUTE**, respectively, for a description of the allowed syntax.

Notes

This command is functionally similar to **SELECT INTO**, but it is preferred since it is less likely to be confused with other uses of the `SELECT INTO` syntax. Furthermore, `CREATE TABLE AS` offers a superset of the functionality offered by `SELECT INTO`.

Prior to PostgreSQL 8.0, `CREATE TABLE AS` always included OIDs in the table it created. As of PostgreSQL 8.0, the `CREATE TABLE AS` command allows the user to explicitly specify whether OIDs should be included. If the presence of OIDs is not explicitly specified, the **default_with_oids** configuration variable is used. As of PostgreSQL 8.1, this variable is false by default, so the default behavior is not identical to pre-8.0 releases. Applications that require OIDs in the table created by `CREATE TABLE AS` should explicitly specify `WITH OIDS` to ensure proper behavior.

Examples

Create a new table `films_recent` consisting of only recent entries from the table `films`:

```
CREATE TABLE films_recent AS
  SELECT * FROM films WHERE date_prod >= '2002-01-01';
```

Compatibility

`CREATE TABLE AS` conforms to the SQL standard, with the following exceptions:

- The standard requires parentheses around the subquery clause; in PostgreSQL, these parentheses are optional.
- The standard defines an `ON COMMIT` clause; this is not currently implemented by PostgreSQL.
- The standard defines a `WITH [ NO ] DATA` clause; this is not currently implemented by PostgreSQL. The behavior provided by PostgreSQL is equivalent to the standard's `WITH DATA` case.
- `WITH/WITHOUT OIDS` is a PostgreSQL extension.
- PostgreSQL handles temporary tables in a way rather different from the standard; see **CREATE TABLE** for details.

See Also

CREATE TABLE, **EXECUTE**, **SELECT**, **SELECT INTO**

Prev	Home	Next
CREATE TABLE	Up	CREATE TABLESPACE