NAME

dblink_connect − opens a persistent connection to a remote database

SYNOPSIS

dblink_connect(text connstr) returns text
dblink_connect(text connname, text connstr) returns text

DESCRIPTION

dblink_connect() establishes a connection to a remote PostgreSQL database. The server and database to be contacted are identified through a standard libpq connection string. Optionally, a name can be assigned to the connection. Multiple named connections can be open at once, but only one unnamed connection is permitted at a time. The connection will persist until closed or until the database session is ended.

The connection string may also be the name of an existing foreign server. It is recommended to use the foreign−data wrapper dblink_fdw when defining the foreign server. See the example below, as well as CREATE SERVER (**CREATE_SERVER**(7)) and CREATE USER MAPPING (**CREATE_USER_MAPPING**(7)).

ARGUMENTS

connname

The name to use for this connection; if omitted, an unnamed connection is opened, replacing any existing unnamed connection.

connstr

libpq−style connection info string, for example hostaddr=127.0.0.1 port=5432 dbname=mydb user=postgres password=mypasswd options=−csearch_path=. For details see Section 32.1.1. Alternatively, the name of a foreign server.

RETURN VALUE

Returns status, which is always OK (since any error causes the function to throw an error instead of returning).

NOTES

If untrusted users have access to a database that has not adopted a secure schema usage pattern, begin each session by removing publicly−writable schemas from *search_path*. One could, for example, add options=−csearch_path= to *connstr*. This consideration is not specific to dblink; it applies to every interface for executing arbitrary SQL commands.

The foreign−data wrapper dblink_fdw has an additional Boolean option use_scram_passthrough that controls whether dblink will use the SCRAM pass−through authentication to connect to the remote database. With SCRAM pass−through authentication, dblink uses SCRAM−hashed secrets instead of plain−text user passwords to connect to the remote server. This avoids storing plain−text user passwords in PostgreSQL system catalogs. See the documentation of the equivalent use_scram_passthrough option of postgres_fdw for further details and restrictions.

Only superusers may use **dblink_connect** to create connections that use neither password authentication, SCRAM pass−through, nor GSSAPI−authentication. If non−superusers need this capability, use **dblink_connect_u** instead.

It is unwise to choose connection names that contain equal signs, as this opens a risk of confusion with connection info strings in other dblink functions.

EXAMPLES

```
SELECT dblink_connect('dbname=postgres options=−csearch_path=');
 dblink_connect
−−−−−−−−−−−−−−−−
 OK
(1 row)

SELECT dblink_connect('myconn', 'dbname=postgres options=−csearch_path=');
 dblink_connect
```

```
----------------
 OK
(1 row)

-- FOREIGN DATA WRAPPER functionality
-- Note: local connections that don't use SCRAM pass-through require password
--       authentication for this to work properly. Otherwise, you will receive
--       the following error from dblink_connect():
--       ERROR:  password is required
--       DETAIL:  Non-superuser cannot connect if the server does not request a password.
--       HINT:  Target server's authentication method must be changed.

CREATE SERVER fdtest FOREIGN DATA WRAPPER dblink_fdw OPTIONS (hostaddr '127.0.0.1', dbname 'contrib_regress

CREATE USER regress_dblink_user WITH PASSWORD 'secret';
CREATE USER MAPPING FOR regress_dblink_user SERVER fdtest OPTIONS (user 'regress_dblink_user', password 'secre
GRANT USAGE ON FOREIGN SERVER fdtest TO regress_dblink_user;
GRANT SELECT ON TABLE foo TO regress_dblink_user;

\set ORIGINAL_USER :USER
\c - regress_dblink_user
SELECT dblink_connect('myconn', 'fdtest');
 dblink_connect
----------------
 OK
(1 row)

SELECT * FROM dblink('myconn', 'SELECT * FROM foo') AS t(a int, b text, c text[]);
 a  | b |       c
----+---+---------------
  0 | a | {a0,b0,c0}
  1 | b | {a1,b1,c1}
  2 | c | {a2,b2,c2}
  3 | d | {a3,b3,c3}
  4 | e | {a4,b4,c4}
  5 | f | {a5,b5,c5}
  6 | g | {a6,b6,c6}
  7 | h | {a7,b7,c7}
  8 | i | {a8,b8,c8}
  9 | j | {a9,b9,c9}
 10 | k | {a10,b10,c10}
(11 rows)

\c - :ORIGINAL_USER
REVOKE USAGE ON FOREIGN SERVER fdtest FROM regress_dblink_user;
REVOKE SELECT ON TABLE foo FROM regress_dblink_user;
DROP USER MAPPING FOR regress_dblink_user SERVER fdtest;
DROP USER regress_dblink_user;
DROP SERVER fdtest;
```