



**CYBERTEC**  
The PostgreSQL Database Company

PostgreSQL Database Links:

# **ODBC-Link 1.0**

Compilation, installation,  
setup and usage

© 2010 Cybertec Schoenig & Schoenig GmbH  
Reyergasse 9 / 2 A-2700 Wiener Neustadt, Austria

[www.postgresql-support.de](http://www.postgresql-support.de)

# PostgreSQL Database Links: ODBC-Link 1.0

ODBC-Link 1.0 is a dblink-style implementation of database links for PostgreSQL. It allows people to connect from PostgreSQL to any ODBC compliant data source and fetch data from any ODBC compliant remote data source. Using ODBC Link you can easily integrate PostgreSQL with other databases and easily send data across a network. In contrast to other implementations ODBC-Link is entirely written in C and offers superior performance and little overhead.

## How to use PostgreSQL database links via ODBC-Link?

### Compilation and Installation

Requirements are:

- a recent version of unixODBC
- ODBC driver for the required DBMS

Compile the module:

```
$ USE_PGXS=1 make
```

Install as root:

```
# USE_PGXS=1 make install
```

### Setup

Load the odbclink.sql to the appropriate database:

```
$ psql -f odbclink.sql dbname
```

The module requires an ODBC DSN correctly set up, e.g. for Informix. Download and install the Informix Client-SDK from IBM and use this information to set up the ODBC DataSource:

<http://www.unixodbc.org/doc/informix.html>

## Usage

First, the connection must be performed:

```
dbname=# select odbclink.connect('dsn', 'username', 'password');
```

```
connect
-----
1
(1 row)
```

or:

```
dbname=# select odbclink.connect('DSN=dsn;UID=username;PWD=password');
```

```
connect
-----
2
(1 row)
```

The function returns the # of the connection. Multiple connections can be active at a time, they don't have to use the same remote server.

Connections can be queried:

```
dbname=# select * from odbclink.connections();
```

```
id | connected | dsn | uid | pwd | connstr
---+-----+---+---+---+-----
 1 | t         | dsn | uid | pwd |
 2 | t         |     |     |     | DSN=dsn;UID=username;PWD=password;
 3 | f         |     |     |     |
 4 | f         |     |     |     |
(4 rows)
```

Disconnection can be performed explicitly:

```
dbname=# select odbclink.disconnect(1);
```

```
disconnect
-----
(1 row)
```

It's not mandatory to disconnect manually, it's automatic upon disconnecting from the PostgreSQL server.

Queries that require no parameters can be performed using either the connection number:

```
dbname=# select odbclink.query(1, 'SELECT * FROM mytable') as result(id int4, t text, d decimal);
```

```
id | t | d
---+---+---
 1 | first text | 1.0
 2 | second row | 1.5
 3 | third client | 20
(3 rows)
```

or the DSN/UID/PWD triplet:

```
dbname=# select odbclink.query('dsn', 'username', 'password', 'SELECT * FROM mytable') as result(id
int4, t text, d decimal);
```

```
 id |      t      | d
----+-----+---
  1 | first text  | 1.0
  2 | second row  | 1.5
  3 | third client| 20
(3 rows)
```

or the connection string:

```
dbname=# select odbclink.query('DSN=dsn;UID=username;PWD=password;', 'SELECT * FROM mytable') as
result(id int4, t text, d decimal);
```

```
 id |      t      | d
----+-----+---
  1 | first text  | 1.0
  2 | second row  | 1.5
  3 | third client| 20
(3 rows)
```

In the last two cases, the DSN/UID/PWD triplets and the connection strings are cached in the internal tables, so no re-connection occurs if the same DSN/UID/PWD triplet or connection string is used for different queries, instead it uses the already open connection.

All three forms of the function odbclink.query() returns "SETOF RECORD", so

1. it can return different query results
2. it must be properly casted to the expected result structure

**Types defined by ODBC are supported.**

**The SQL\_\*BINARY types are only supported from PostgreSQL 8.5/9.0+.**