

# Setting Up Bi-Directional Replication in PostgreSQL with BDR Clustering

## Description:

BDR (Bi-Directional Replication) is a tool for PostgreSQL that lets you set up a group of servers where you can make changes on any server, and those changes will be shared with all the other servers in the group. This allows multiple servers to work together, keeping the data in sync across them.

## Installation : For installation follow these Steps

1. Create a user account on the EnterpriseDB website to obtain a token.



2. Set up the following repositories after completing this step.

```
1 curl -sLf 'https://downloads.enterprisedb.com/<Token>/enterprise/setup.rpm.sh' | sudo -E bash
2
```

```
1 dnf install yum-utils
2 rpm --import 'https://downloads.enterprisedb.com/<Token>/enterprise/gpg.E71EB0829F1EF813.key'
3 curl -sLf 'https://downloads.enterprisedb.com/<Token>/enterprise/config.rpm.txt?distro=el&codename=8' >
  /tmp/enterprise.repo
4 dnf config-manager --add-repo '/tmp/enterprise.repo'
5 dnf -q makecache -y --disablerepo='*' --enablerepo='enterprisedb-enterprise'
```

3. Install the following BDR packages next:

```
1 sudo dnf -y install edb-bdr5-pg16 edb-pgd5-proxy edb-pgd5-cli
```

Upon executing this command, the following packages will be installed:

1	324 B/s   659 B
2	00:02
3	Dependencies resolved.
4	=====
5	=====
6	Package Architecture Version Repository
7	Size
8	=====
9	=====
10	Installing:
11	edb-bdr5-pg16 x86_64 4:5.5.1-3.el8
12	enterprisedb-postgres_distributed 715 k
13	edb-pgd5-cli x86_64 5.5.0-1
14	enterprisedb-postgres_distributed 4.4 M
15	edb-pgd5-proxy x86_64 5.5.0-1
16	enterprisedb-postgres_distributed 3.7 M

```

10 Installing dependencies:
11 postgresql16-contrib          x86_64          16.4-1PGDG.rhel8          pgdg16
12 760 k
13 Transaction Summary
14 =====
15 =====

```

4. Set the password for the postgres OS user.

```

1 passwd postgres
2 Changing password for user postgres.
3 New password:
4 BAD PASSWORD: The password is shorter than 8 characters
5 Retype new password:
6 passwd: all authentication tokens updated successfully.

```

5. Add the postgres user to the sudoer list.

Execute this command.

```

1 sudo visudo

```

Add the following line

```

1 postgres ALL=(ALL) ALL

```

6. Performing two initial database setups.

```

1 ./initdb -D /var/lib/pgsql/16/node_data -A trust

```

```

1 ./initdb -D /var/lib/pgsql/16/node2_data -A trust

```

7. Configure the following settings in both instances of the database:

DB1 Instance

```

1 echo -e "shared_preload_libraries = '\$libdir/bdr'" | sudo -u postgres tee -a
  /var/lib/pgsql/16/node_data/postgresql.conf >/dev/null
2 echo -e "wal_level = 'logical'" | sudo -u postgres tee -a /var/lib/pgsql/16/node_data/postgresql.conf
  >/dev/null
3 echo -e "track_commit_timestamp = 'on'" | sudo -u postgres tee -a /var/lib/pgsql/16/node_data/postgresql.conf
  >/dev/null
4 echo -e "listen_addresses = '*' " | sudo -u postgres tee -a /var/lib/pgsql/16/node_data/postgresql.conf
  >/dev/null
5 echo -e "max_worker_processes = '16'" | sudo -u postgres tee -a /var/lib/pgsql/16/node_data/postgresql.conf
  >/dev/null
6 echo -e "host all all all md5\nhost replication all all md5" | sudo tee -a
  /var/lib/pgsql/16/node_data/pg_hba.conf >/dev/null
7
8 ./psql -U postgres
9 psql (16.4)
10 Type "help" for help.
11
12 postgres=# ALTER USER postgres WITH PASSWORD 'test'

```

DB2 Instance

```

1 echo -e "shared_preload_libraries = '\$libdir/bdr'" | sudo -u postgres tee -a
  /var/lib/pgsql/16/node2_data/postgresql.conf >/dev/null
2 echo -e "wal_level = 'logical'" | sudo -u postgres tee -a /var/lib/pgsql/16/node2_data/postgresql.conf
  >/dev/null

```

```

3 echo -e "track_commit_timestamp = 'on'" | sudo -u postgres tee -a /var/lib/pgsql/16/node2_data/postgresql.conf >/dev/null
4 echo -e "listen_addresses = '*' " | sudo -u postgres tee -a /var/lib/pgsql/16/node2_data/postgresql.conf >/dev/null
5 echo -e "port = 6000" | sudo -u postgres tee -a /var/lib/pgsql/16/node2_data/postgresql.conf >/dev/null
6 echo -e "max_worker_processes = '16'" | sudo -u postgres tee -a /var/lib/pgsql/16/node2_data/postgresql.conf >/dev/null
7 echo -e "host all all all md5\nhost replication all all md5" | sudo tee -a /var/lib/pgsql/16/node2_data/pg_hba.conf >/dev/null
8
9 ./psql -U postgres -p 5433
10 psql (16.4)
11 Type "help" for help.
12
13 postgres=# ALTER USER postgres WITH PASSWORD 'test'

```

8. Start both database instances.

```
1 ./pg_ctl -D /var/lib/pgsql/16/node_data/ start
```

```
1 ./pg_ctl -D /var/lib/pgsql/16/node2_data/ start
```

## Setting Up the Bi-Directional Replication Cluster

1. Setup first master node

Create database

```
1 postgres=# CREATE DATABASE node1;
2 CREATE DATABASE

```

Create BDR extension

```
1 CREATE EXTENSION bdr CASCADE;
2 CREATE EXTENSION

```

Execute the creation of a node to register it as a PGD node.

```

1 SELECT bdr.create_node(node_name := 'master', local_dsn := 'port=5432 dbname=node1 host=localhost
user=postgres');
2 create_node
3 -----
4 3582890138
5 (1 row)

```

Create the PGD node group

```

1 node1=# SELECT bdr.create_node_group(node_group_name := 'bdrtest-group');
2 create_node_group
3 -----
4 117160161
5 (1 row)

```

2. Set up the second master node.

Create database

```
1 postgres=# CREATE DATABASE node2;
2 CREATE DATABASE

```

Create BDR extension

```
1 CREATE EXTENSION bdr CASCADE;
```

```
2 CREATE EXTENSION
```

Create the second node

```
1 pgdtest=# SELECT bdr.create_node(node_name := 'secondmasternode', local_dsn := 'port=5433 dbname=node2
host=localhost user=postgres');
2 create_node
3 -----
4 3139556418
5 (1 row)
```

Execute the command to join the existing node group by using the "join node group" .

```
1 node2=# SELECT bdr.join_node_group(join_target_dsn := 'port=5432 dbname=node1 host=localhost user=postgres');
2 join_node_group
3 -----
4
5 (1 row)
```

## Sample data and replication

1. Create the following table on the first node and insert some data into the table.

```
1 CREATE TABLE cities (
2     id SERIAL PRIMARY KEY,
3     name VARCHAR(50),
4     population INTEGER
5 );
6 INSERT INTO cities (name, population)
7 VALUES
8 ('New York', 8419000),
9 ('Los Angeles', 3980000),
10 ('Chicago', 2716000);
11
```

2. Validate the data on the 2nd master node.

```
1 [postgres@f8949e21db0c bin]$ ./psql -U postgres -p 5433
2 psql (16.4)
3 Type "help" for help.
4
5 postgres=# \c node2
6 You are now connected to database "pgdtest" as user "postgres".
7 pgdtest=# select * from cities;
8  id |   name   | population
9  ---+-----+-----
10  2 | New York |    8419000
11  3 | Los Angeles |   3980000
12  4 | Chicago  |   2716000
13 (3 rows)
14
15 pgdtest=#
```

3. Now, input data on the 2nd master node and proceed to verify it on the 1st master.

```
1 INSERT INTO cities (name, population)
2 VALUES ('Houston', 2328000),
3 ('Phoenix', 1690000);
4
```

Check the availability of data on the first master node.

```
1 ./psql -U postgres
2 psql (16.4)
3 Type "help" for help.
4
5 postgres=# \c node1
6 You are now connected to database "pgdtest" as user "postgres".
7 pgdtest=# select * from cities;
8      id      |      name      | population
9 -----+-----+-----
10          2 | New York      |    8419000
11          3 | Los Angeles   |    3980000
12          4 | Chicago       |    2716000
13 20000002 | Houston       |    2328000
14 20000003 | Phoenix       |    1690000
15 (5 rows)
16
```

#### 4. Get BDR nodes detail

```
1 pgdtest=# select node_id,node_name,node_state,target_state,node_dsn from bdr.node;
2      node_id      |      node_name      | node_state | target_state |
3 -----+-----+-----+-----+-----
4 3582890138 | master              |          80 |          80 | port=5432 dbname=node1 host=localhost
5 user=postgres
6 3139556418 | secondmasternode    |          80 |          80 | port=5433 dbname=node2 host=localhost
7 user=postgres
8 (2 rows)
9
```

#### 5. Print the node group details

```
1 pgdtest=# select node_group_id,node_group_name from bdr.node_group;
2      node_group_id | node_group_name
3 -----+-----
4      117160161 | pgdtest-group
5 (1 row)
```