



Beta

**EDB Postgres™ Replication Server
Quickstart Guide**

EDB Postgres™ Replication Server 7

January 7, 2019

EDB Postgres™ Replication Server Quickstart Guide
by EnterpriseDB® Corporation
Copyright © 2019 EnterpriseDB Corporation. All rights reserved.

Beta

EnterpriseDB Corporation, 34 Crosby Drive, Suite 201, Bedford, MA 01730, USA
T +1 781 357 3390 **F** +1 978 467 1307 **E** info@enterprisedb.com **www**.enterprisedb.com

1 Quickstart

EDB Postgres Replication Server uses message streaming using Apache Kafka for replicating the data that has changed between the databases.

When EDB Replication Server is used to build a replication network for replicating the data that has changed from producer databases to the consumer databases, the generated software components and processes are built upon the Kafka components.

The components of a replication network and their relationship to the underlying Kafka components are as follows:

- **Replication Server.** A replication server is a process running as a dedicated RESTful service on an HTTP server. The replication server uses a single Kafka broker and either one or two ZooKeeper instances. Each replication server can either have none, one, or more than one associated databases that act as a producer or consumer for Kafka. The replication server is the first EDB Replication Server component that must be created and started when you create a replication node.
- **Leader Service.** Leader service is the primary replication server in the replication network. The leader service is the replication server with which the RepCLI commands issued by the users communicate. The first replication server started while creating a new replication network acts as the leader service and has two ZooKeeper instances running in its framework. All the other additional replication servers added to the replication network have a single ZooKeeper instance. During replication, the leader service role may be transferred to another replication server. This happens due to a failover operation (the current replication server acting as the leader service aborts, so the leader service operation is transferred to an active replication server to keep the replication network active).
- **Replication Node.** A replication node consists of the following:
 - Single replication server
 - Underlying Kafka broker
 - ZooKeeper instances
 - Producer and consumer databases
 - Publications
 - Topic partitions within the Kafka broker
 - Other supporting open source components

A replication node may not have any producer or consumer databases added to its replication server to enhance high availability support. In this scenario, if another replication server aborts, its functionality can be transferred to this replication server. A replication node forms a single, complete EDB Replication Server component that can contribute to a full replication network.

- **Replication Network.** A Replication network is a group of one or more replication nodes. Each replication node must be registered on the replication

network. This is done by joining its replication server to the replication network. The first replication server started and joined to the replication network starts as the leader service. All of the registered replication nodes can then exchange changed data between any of their producer and consumer databases. Also, there may be replication nodes with no databases that are provided for high availability.

- **Publication.** Publication is defined as a set of one or more tables from a given producer database whose changed data is streamed to consumer databases. A publication is implemented as Kafka topics for storing the changed data. The changed data is streamed to consumer databases that have been registered as part of the replication network and are joined to that particular publication. A publication is identified by a name that must be unique amongst all the publications in the replication network.
- **Replication Command Line Interface (RepCLI).** RepCLI is the command line tool to do the setup, configuration, and execution process for EDB Replication Server. The RepCLI commands are supported by the RESTful architecture.

The following is an example of a system consisting of a pair of databases configured on a single replication node.

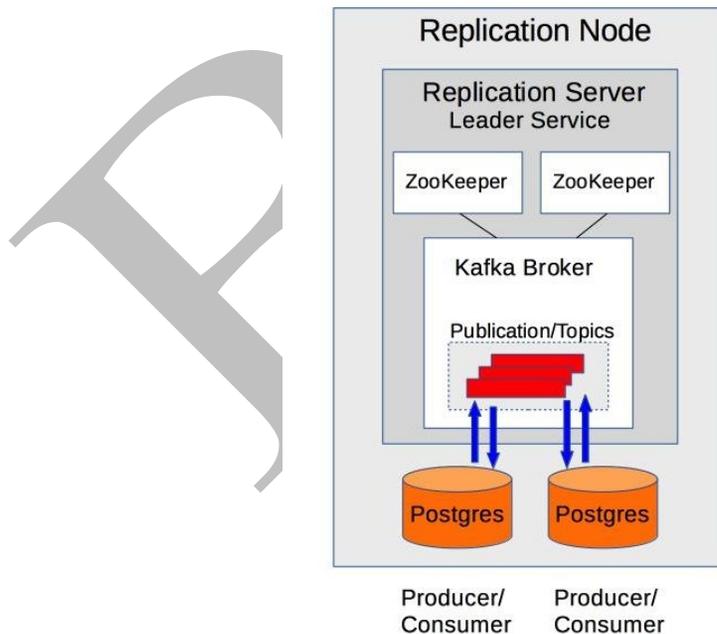


Figure 1-1 Cluster on a single replication node

To install EDB Replication Server first install EDB Postgres Advanced Server and then install EDB replication server.

Installing EDB Postgres Advanced Server and creating clusters

Perform the following steps as the `root` user for installing advanced server:

Step 1: Visit the following URL to obtain the credentials to access EDB Yum Repository:

<https://www.enterprisedb.com/repository-access-request>

Step 2: Download the EDB repository RPM.

```
rpm -Uvh http://yum.enterprisedb.com/edbrepos/edb-repo-latest.noarch.rpm
```

Step 3: Replace the username and password which you got from Step 1 in place of `<yum user>` and `<yum password>` in `edb.repo` located at:

```
vi /etc/yum.repos.d/edb.repo
```

Step 4: Enable the components to be installed.

```
sed -i "\/edbas10/,/gpgcheck/ s/enabled=0/enabled=1/"
/etc/yum.repos.d/edb.repo
sed -i "\/enterprisedb-dependencies/,/gpgcheck/
s/enabled=0/enabled=1/" /etc/yum.repos.d/edb.repo
```

Note: Do not run the following command until the proper component name has been established for `enterprisedb-xdb??` on GA release for EDB Replication Server.

```
sed -i "\/enterprisedb-xdb??./,/gpgcheck/
s/enabled=0/enabled=1/" /etc/yum.repos.d/edb.repo
```

Step 5: Install the RPM packages.

```
yum -y install epel-release
yum -y install wxBase
yum -y install edb-as10-server
```

Step 6: Create Advanced Server cluster 1 with default port 5444.

```
su - enterprisedb -c "mkdir /var/lib/edb/as10/cluster1"
su - enterprisedb -c "chmod 700 /var/lib/edb/as10/cluster1"
su - enterprisedb -c "/usr/edb/as10/bin/initdb -D
/var/lib/edb/as10/cluster1"
```

Step 7: Create Advanced Server cluster 2 with non-default port 5445.

```
su - enterprisedb -c "mkdir /var/lib/edb/as10/cluster2"
```

```
su - enterprisedb -c "chmod 700 /var/lib/edb/as10/cluster2"
su - enterprisedb -c "/usr/edb/as10/bin/initdb -D
/var/lib/edb/as10/cluster2"
su - enterprisedb -c "sed -i 's/port = 5444/port = 5445/'
/var/lib/edb/as10/cluster2/postgresql.conf"
```

Step 8: Modify the Advanced Server configuration file to allow the usage for EDB Replication Server.

```
su - enterprisedb -c "sed -i 's/#wal_level =
replica/wal_level = logical/'
/var/lib/edb/as10/cluster*/postgresql.conf"
```

```
su - enterprisedb -c "sed -i 's/#track_commit_timestamp =
off/track_commit_timestamp = on/'
/var/lib/edb/as10/cluster*/postgresql.conf"
```

Step 9: Start the two clusters.

```
su - enterprisedb -c "/usr/edb/as10/bin/pg_ctl start -D
/var/lib/edb/as10/cluster1"
su - enterprisedb -c "/usr/edb/as10/bin/pg_ctl start -D
/var/lib/edb/as10/cluster2"
```

Installing EDB Replication Server

Perform the following steps as the `root` user:

Step 1: Install Java 1.8.

```
yum -y install java-1.8.0-openjdk
```

Step 2: Install the EDB Replication Server packages. If you are not installing directly from the EDB Yum Repository by enablement of the EDB Replication Server component noted in Step 4, but you have directly downloaded the package files to your host machine, change to the directory to where the package files have been downloaded before invoking the following `install` command.

```
yum -y install edb-rs-*
```

Step 3: Set the password for the database user `enterprisedb` on each cluster.

```
su - enterprisedb -c "psql -d postgres -p 5444 -c 'alter role
enterprisedb identified by password'"
su - enterprisedb -c "psql -d postgres -p 5445 -c 'alter role
enterprisedb identified by password'"
```

Step 4: Encrypt the password which is used in some RepCLI commands. The displayed encrypted password has been included in subsequent `adddb` RepCLI commands.

```
echo 'password' > ~/passwd.in
export EPRS_HOME=/usr/edb-rs-7.0
$EPRS_HOME/client/bin/runRepCLI.sh -encrypt -input
~/passwd.in -output ~/passwd.out
cat ~/passwd.out
```

Step 5: Create the databases for replication node in each cluster.

```
su - enterprisedb -c "psql -d postgres -p 5444 -c 'create
database node1'"
su - enterprisedb -c "psql -d postgres -p 5445 -c 'create
database node2'"
```

Step 6: Create the table in database `node1` and insert the initial rows.

```
su - enterprisedb -c "psql -d node1 -p 5444 -c 'create table
public.dept (deptno numeric(2) NOT NULL CONSTRAINT dept_pk
PRIMARY KEY, dname varchar(14) CONSTRAINT dept_dname_uq
UNIQUE, loc varchar(13))'"
su - enterprisedb -c "psql -d node1 -p 5444 -c 'alter table
public.dept REPLICA IDENTITY FULL'"

su - enterprisedb -c "psql -d node1 -p 5444 -c \"insert into
dept VALUES (10, 'ACCOUNTING', 'NEW YORK')\""
su - enterprisedb -c "psql -d node1 -p 5444 -c \"insert into
dept VALUES (20, 'RESEARCH', 'DALLAS')\""
su - enterprisedb -c "psql -d node1 -p 5444 -c \"select *
from dept\""
```

Step 7: Create the table in database `node2` with no rows.

```
su - enterprisedb -c "psql -d node2 -p 5445 -c 'create table
public.dept (deptno numeric(2) NOT NULL CONSTRAINT dept_pk
PRIMARY KEY, dname varchar(14) CONSTRAINT dept_dname_uq
UNIQUE, loc varchar(13))'"
su - enterprisedb -c "psql -d node2 -p 5445 -c 'alter table
public.dept REPLICA IDENTITY FULL'"
su - enterprisedb -c "psql -d node2 -p 5445 -c \"select *
from dept\""
```

Step 8: Set the environment variables `HOST_IP` and `EPRS_HOME`, and then start the replication server.

```
export HOST_IP=127.0.0.1
export EPRS_HOME=/usr/edb-rs-7.0
```

```
$EPRS_HOME/server/bin/runServer.sh --host $HOST_IP -c
$EPRS_HOME/server/etc
```

Step 9: Open a second terminal, and as the `root` user, set the `HOST_IP` and `EPRS_HOME` environment variables, then run the `joinnetwork RepCLI` command.

```
export HOST_IP=127.0.0.1
export EPRS_HOME=/usr/edb-rs-7.0
$EPRS_HOME/client/bin/runRepCLI.sh -joinnetwork -servername
localService -host $HOST_IP -port 8082
```

Run all the remaining commands from this second terminal as the `root` user.

Step 10: Set the administrator password.

```
$EPRS_HOME/client/bin/runRepCLI.sh -setadminpassword -
savepassword
```

Step 11: Add database `node1` to the replication server.

```
$EPRS_HOME/client/bin/runRepCLI.sh -adddb -servername
localService -dbid db1 -dbtype enterprisedb -dbhost 127.0.0.1
-dbport 5444 -dbuser enterprisedb -dbpassword
ygJ9AxoJEX854elcVIJPTw== -database node1 -user admin
```

Step 12: Create a publication.

```
$EPRS_HOME/client/bin/runRepCLI.sh -createpub -pubname
deptpub -servername localService -dbid db1 -tables
public.dept -nodetype RW -user admin
```

Note:

- To modify a publication by adding tables to a publication use the `addtables` command. To use the `addtables` command you should have the `create_pub` permission.
- To modify a publication by removing tables from a publication use the `removetables` command. To use the `removetables` command you should have the `remove_pub` permission.

Examples

```
$EPRS_HOME/client/bin/runRepCLI.sh -removetables -pubname
deptpub -tables public.dept -user admin
```

```
$EPRS_HOME/client/bin/runRepCLI.sh -addtables -pubname
deptpub -tables public.dept -user admin
```

Step 13: Add databasenode2 to the replication server.

```
$EPRS_HOME/client/bin/runRepCLI.sh -adddb -servername
localService -dbid db2 -dbtype enterprisedb -dbhost 127.0.0.1
-dbport 5445 -dbuser enterprisedb -dbpassword
ygJ9AxoJEX854elcVIJPTw== -database node2 -user admin
```

Step 14: Join database node2 to the publication.

```
$EPRS_HOME/client/bin/runRepCLI.sh -joinpub -servername
localService -dbid db2 -pubname deptpub -nodetype RW -user
admin
```

Step 15: Take an initial snapshot after which you should see the initial two rows loaded into the node1 database now in the node2 database as well.

```
$EPRS_HOME/client/bin/runRepCLI.sh -startsnapshot -pubname
deptpub -dbid db2 -user admin
```

Step 16: Run the checksnapshot command (after a few seconds) to confirm if the data is replicated on the target node.

```
$EPRS_HOME/client/bin/runRepCLI.sh -checksnapshot -pubname
deptpub -dbid db2 -user admin

su - enterprisedb -c "psql -d node2 -p 5445 -c \"select *
from dept\""
```

Step 17: Start streaming.

```
$EPRS_HOME/client/bin/runRepCLI.sh -startstreaming -pubname
deptpub -user admin
```

Step 18: Insert additional rows into node1.

```
su - enterprisedb -c "psql -d node1 -p 5444 -c \"insert into
dept VALUES (30, 'SALES', 'CHICAGO')\""
```

```
su - enterprisedb -c "psql -d node1 -p 5444 -c \"insert into
dept VALUES (40, 'OPERATIONS', 'BOSTON')\""
```

Step 19: Insert rows into node2.

```
su - enterprisedb -c "psql -d node2 -p 5445 -c \"insert into
dept VALUES (50, 'HR', 'DENVER')\""
```

```
su - enterprisedb -c "psql -d node2 -p 5445 -c \"insert into
dept VALUES (60, 'FINANCE', 'CHICAGO')\""
```

Step 20: Display the rows from node1.

```
su - enterprisedb -c "psql -d node1 -p 5444 -c \"select *
from dept\""
```

Step 21: Display the rows from node2.

```
su - enterprisedb -c "psql -d node2 -p 5445 -c \"select *
from dept\""
```

Note: Node 1 and node2 should have the same rows.

Replication Cluster Cleanup

To stop the replication server and delete all the replication data from the host machine do the following:

Step 1: Leave the publication.

```
$EPRS_HOME/client/bin/runRepCLI.sh -leavepub -dbid db2 -
pubname deptpub -user admin
```

Note: `leavepub` command should be used for the databases which are joined to the publication using the `joinpub` command.

Step 2: Remove the publication.

```
$EPRS_HOME/client/bin/runRepCLI.sh -removepub -pubname
deptpub -user admin
```

Note: `removepub` command is used for publications which are created using the `createpub` command.

Step 3: Remove the database.

```
$EPRS_HOME/client/bin/runRepCLI.sh -removedb -dbid db1 -user
admin
```

Note: Before using the `removedb` command, use the `leavepub` command to disconnect the database from all the publications it had joined with the `joinpub` command. Also, make sure to remove all the publications created in the database using the `removepub` command.

Step 4: Leave the network.

```
$EPRS_HOME/client/bin/runRepCLI.sh -leavenetwork -servername
localService -user admin
```