Implementing PostgreSQL Replication and Failover Solutions Using repmgr

Objective: To enhance database reliability, performance, and management through PostgreSQL replication and failover.

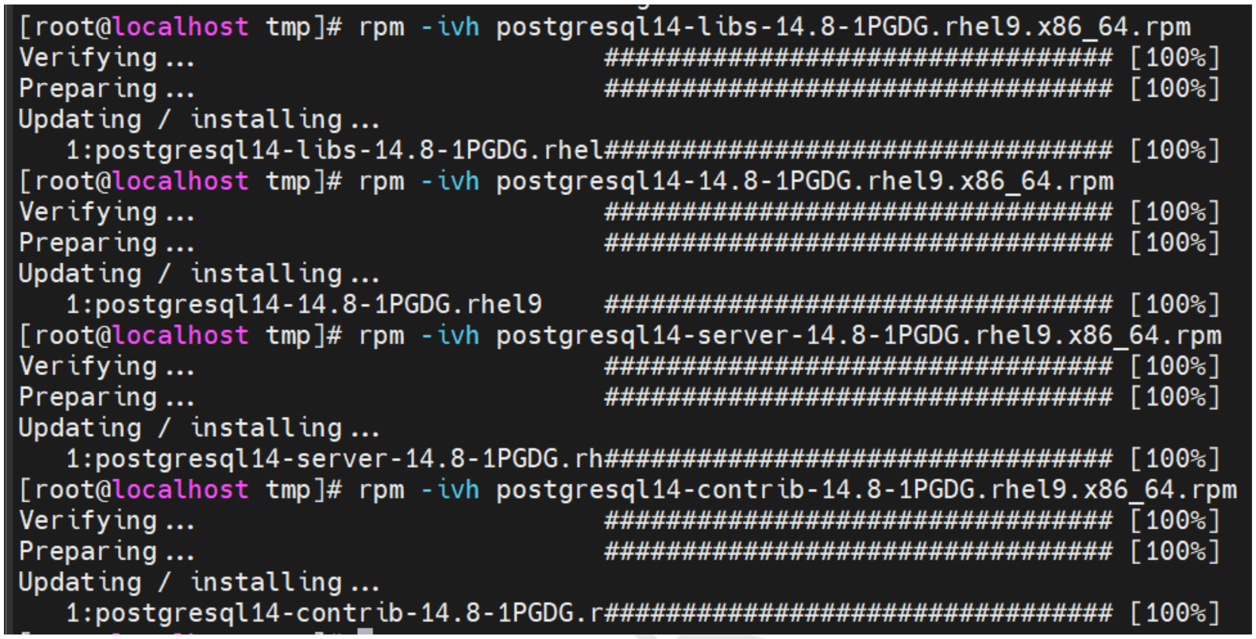
Key Benefits:

- **High Availability:**
  - Goal: Minimize downtime and maintain continuous database availability.
  - How: By replicating data from a primary server to standby servers, ensuring service continuity even during server failures.
- **Data Redundancy:**
  - Goal: Protect against data loss and ensure data integrity.
  - How: Real-time replication to standby servers provides an up-to-date backup, crucial for disaster recovery.
- **Automatic Failover:**
  - Goal: Ensure seamless transition in case of primary server failure.
  - How: repmgr facilitates automatic failover, promoting standby servers to primary status without manual intervention.
- **Scalability:**
  - Goal: Optimize database performance and handle increasing workloads.
  - How: Distributes read queries and load across multiple servers, enhancing overall system performance.
- **Backup and Recovery:**
  - Goal: Improve backup efficiency and recovery processes.
  - How: Standby servers can be utilized for backups, reducing the impact on the primary server and ensuring quick recovery.

Overall Impact: Implementing PostgreSQL replication and failover with repmgr delivers robust data protection, high availability, and streamlined database management, positioning your organization to effectively handle database challenges and growth.

On the Primary Server:

1. Install PostgreSQL 14 RPM:



```
[root@localhost tmp]# rpm -ivh postgresql14-libs-14.8-1PGDG.rhel9.x86_64.rpm
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
   1:postgresql14-libs-14.8-1PGDG.rhel################################# [100%]
[root@localhost tmp]# rpm -ivh postgresql14-14.8-1PGDG.rhel9.x86_64.rpm
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
   1:postgresql14-14.8-1PGDG.rhel9    ################################# [100%]
[root@localhost tmp]# rpm -ivh postgresql14-server-14.8-1PGDG.rhel9.x86_64.rpm
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
   1:postgresql14-server-14.8-1PGDG.rh################################# [100%]
[root@localhost tmp]# rpm -ivh postgresql14-contrib-14.8-1PGDG.rhel9.x86_64.rpm
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
   1:postgresql14-contrib-14.8-1PGDG.r################################# [100%]
```

2. Install repmgr 14:

- Install repmgr using RPM:

sudo dnf install -y repmgr_14*

3. Configure PostgreSQL for Replication:

- Edit $PGDATA/postgresql.conf

vim $PGDATA/postgresql.conf

Add or update the following parameters:

listen_addresses = '*'

wal_level = replica

max_wal_senders = 10

max_replication_slots = 10

wal_keep_size = 1GB

hot_standby = on

shared_preload_libraries = 'repmgr'

Save and exit.

The **shared_preload_libraries** setting in PostgreSQL is used to load certain libraries at server startup. When you set **shared_preload_libraries = 'repmgr',** it tells PostgreSQL to preload the repmgr extension, which is necessary for repmgr to function correctly.

Here's why you need this setting:

- Replication Management: repmgr is a tool used for managing replication and failover in PostgreSQL. For repmgr to integrate with PostgreSQL and manage replication effectively, it needs to be loaded as a shared library when PostgreSQL starts.
- Extended Functionality: Loading repmgr as a shared library allows it to extend PostgreSQL's capabilities with additional functionality required for replication and failover operations.
- Configuration and Monitoring: The repmgr extension allows you to configure and monitor replication from within PostgreSQL, and it needs to be available as soon as PostgreSQL starts to manage these tasks properly.

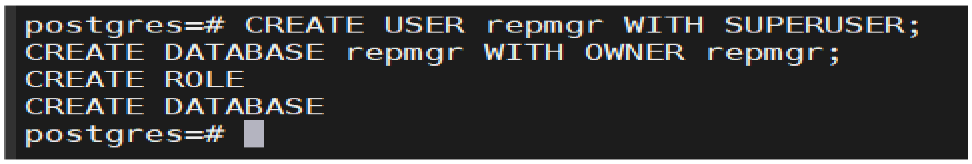
To apply this change, you need to update the PostgreSQL configuration file (postgresql.conf), add repmgr to the shared_preload_libraries list, and then restart the PostgreSQL server for the changes to take effect.

4. Restart and Check PostgreSQL Services:

systemctl restart postgresql-14.service

systemctl status postgresql-14.service

5. Create User and Database for repmgr:



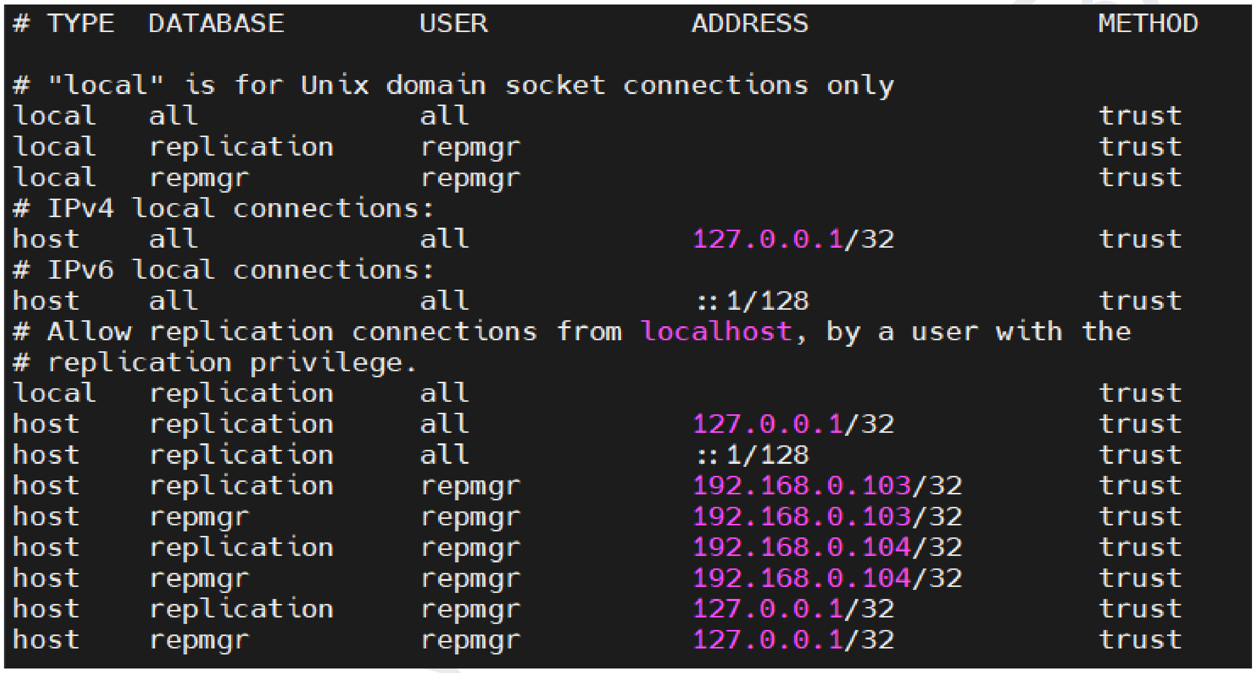
```
postgres=# CREATE USER repmgr WITH SUPERUSER;
CREATE DATABASE repmgr WITH OWNER repmgr;
CREATE ROLE
CREATE DATABASE
postgres=#
```

6. Configure Connectivity for repmgr User:

- Edit $PGDATA/pg_hba.conf:

vim $PGDATA/pg_hba.conf

Add the following entries:

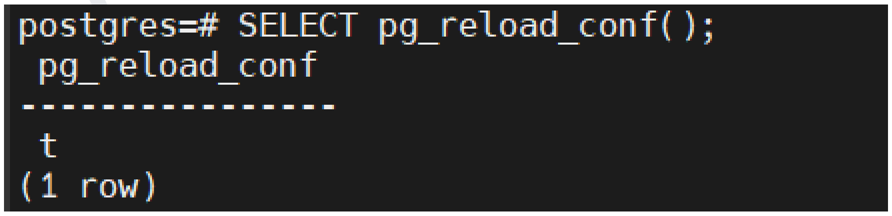


```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     trust
local   replication     repmgr                                  trust
local   repmgr          repmgr                                  trust
# IPv4 local connections:
host    all             all             127.0.0.1/32            trust
# IPv6 local connections:
host    all             all             ::1/128                 trust
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     trust
host    replication     all             127.0.0.1/32            trust
host    replication     all             ::1/128                 trust
host    replication     repmgr          192.168.0.103/32        trust
host    repmgr          repmgr          192.168.0.103/32        trust
host    replication     repmgr          192.168.0.104/32        trust
host    repmgr          repmgr          192.168.0.104/32        trust
host    replication     repmgr          127.0.0.1/32            trust
host    repmgr          repmgr          127.0.0.1/32            trust
```

Save and exit.

- Reload PostgreSQL configurations:

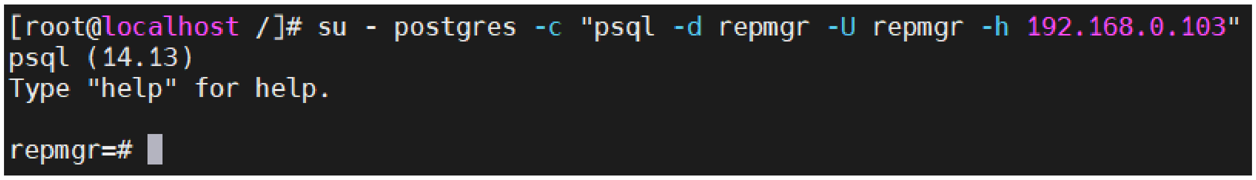
SELECT pg_reload_conf();



```
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)
```

- Check connectivity:

su - postgres -c "psql -d repmgr -U repmgr -h 192.168.0.103"

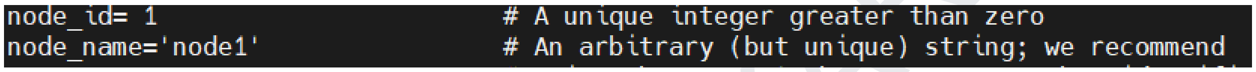

```
[root@localhost /]# su - postgres -c "psql -d repmgr -U repmgr -h 192.168.0.103"
psql (14.13)
Type "help" for help.

repmgr=#
```

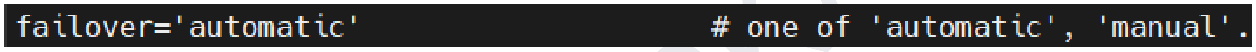
7. Edit repmgr Configuration on Primary Server:

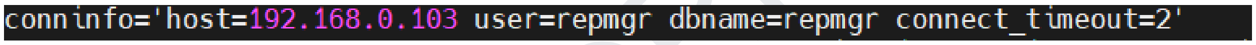
- Edit /etc/repmgr/14/repmgr.conf:

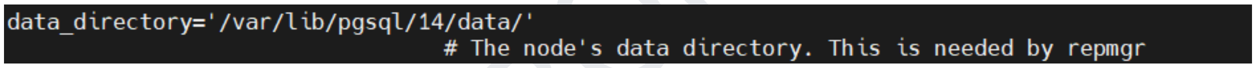
vim /etc/repmgr/14/repmgr.conf

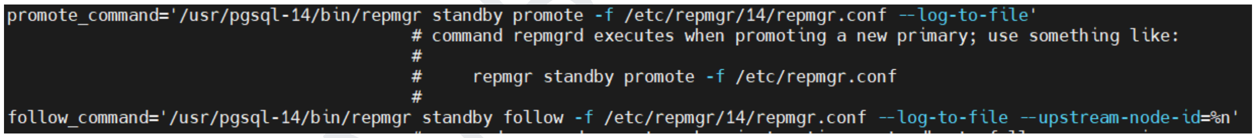
Add the following entries:


```
node_id= 1                           # A unique integer greater than zero
node_name='node1'                    # An arbitrary (but unique) string; we recommend
```


```
failover='automatic'                     # one of 'automatic', 'manual'.
```

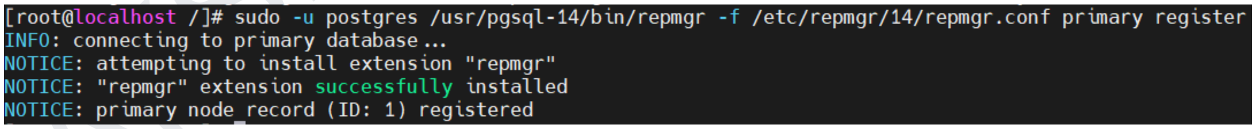

```
conninfo='host=192.168.0.103 user=repmgr dbname=repmgr connect_timeout=2'
```


```
data_directory='/var/lib/pgsql/14/data/'
                                    # The node's data directory. This is needed by repmgr
```

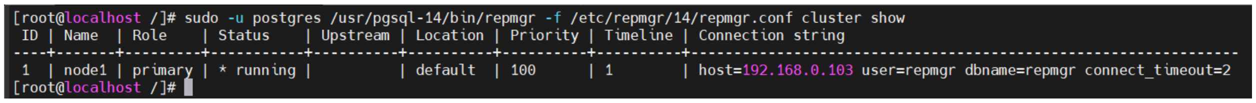

```
promote_command='/usr/pgsql-14/bin/repmgr standby promote -f /etc/repmgr/14/repmgr.conf --log-to-file'
                                        # command repmgrd executes when promoting a new primary; use something like:
                                        #
                                        #     repmgr standby promote -f /etc/repmgr.conf
                                        #
follow_command='/usr/pgsql-14/bin/repmgr standby follow -f /etc/repmgr/14/repmgr.conf --log-to-file --upstream-node-id=%n'
```

Save and exit.

8. Register the Primary Server with repmgr:


```
[root@localhost /]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf primary register
INFO: connecting to primary database...
NOTICE: attempting to install extension "repmgr"
NOTICE: "repmgr" extension successfully installed
NOTICE: primary node record (ID: 1) registered
```

- **Check the cluster status:**


```
[root@localhost /]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf cluster show
 ID | Name  | Role    | Status    | Upstream | Location | Priority | Timeline | Connection string
----+-------+---------+-----------+----------+----------+----------+----------+--------------------------------------------------------------
 1  | node1 | primary | * running |          | default  | 100      | 1        | host=192.168.0.103 user=repmgr dbname=repmgr connect_timeout=2
[root@localhost /]#
```

On the Standby Server:

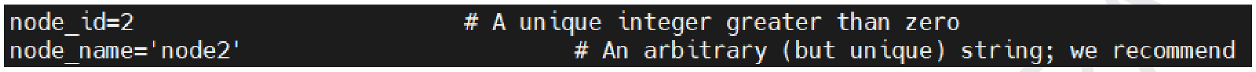
9. Install PostgreSQL & Repmgr As Above:

10. Edit repmgr Configuration on Standby Server:

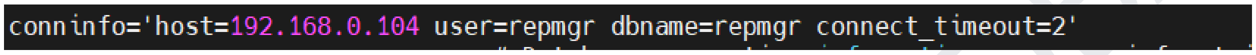
- Edit /etc/repmgr/14/repmgr.conf:

vim /etc/repmgr/14/repmgr.conf

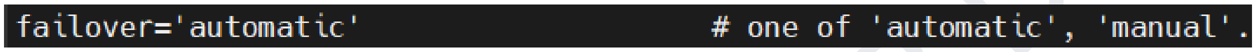
Add the following entries:



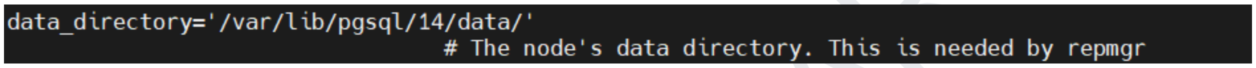
```
node_id=2                              # A unique integer greater than zero
node_name='node2'                              # An arbitrary (but unique) string; we recommend
```



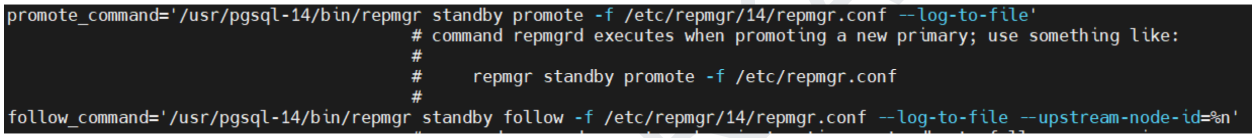
```
conninfo='host=192.168.0.104 user=repmgr dbname=repmgr connect_timeout=2'
```



```
failover='automatic'                    # one of 'automatic', 'manual'.
```



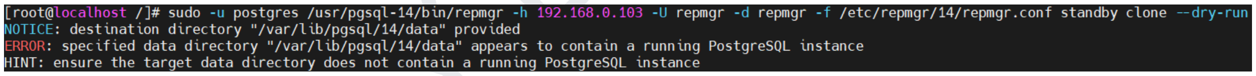
```
data_directory='/var/lib/pgsql/14/data/'
                                    # The node's data directory. This is needed by repmgr
```



```
promote_command='/usr/pgsql-14/bin/repmgr standby promote -f /etc/repmgr/14/repmgr.conf --log-to-file'
                                        # command repmgrd executes when promoting a new primary; use something like:
                                        #
                                        #     repmgr standby promote -f /etc/repmgr.conf
                                        #
follow_command='/usr/pgsql-14/bin/repmgr standby follow -f /etc/repmgr/14/repmgr.conf --log-to-file --upstream-node-id=%n'
```

Save and exit.

11. Perform a Dry Run for Configuration Validation:



```
[root@localhost /]# sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone --dry-run
NOTICE: destination directory "/var/lib/pgsql/14/data" provided
ERROR: specified data directory "/var/lib/pgsql/14/data" appears to contain a running PostgreSQL instance
HINT: ensure the target data directory does not contain a running PostgreSQL instance
```

The error indicates that the target data directory /var/lib/pgsql/14/data appears to contain a running PostgreSQL instance. Here's what you can do to resolve this issue:

- **Check if PostgreSQL is Running:** Ensure that there is no active PostgreSQL instance using the target data directory. You can check the status of PostgreSQL with:

  sudo systemctl start postgresql-14.service

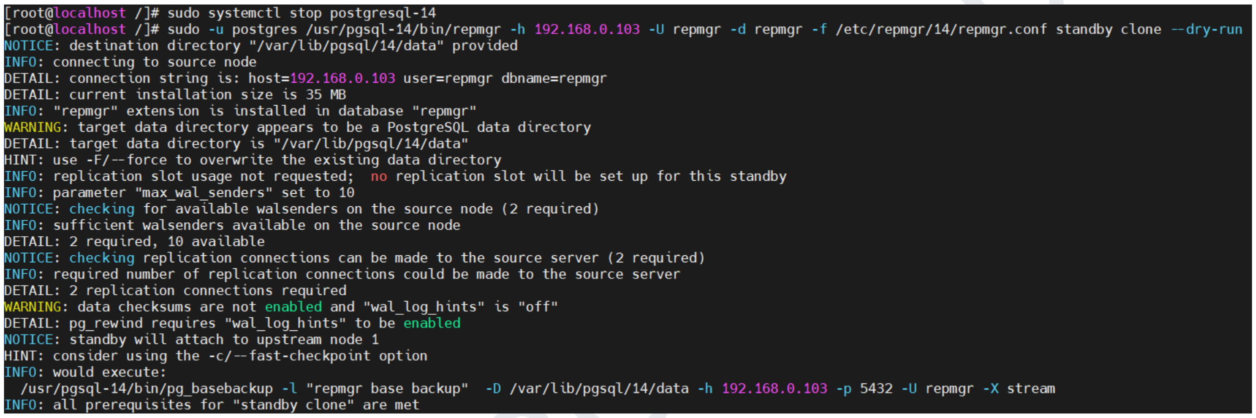
- **Verify the Data Directory:** Make sure that the data directory specified is indeed the correct one and not being used by another PostgreSQL instance. You can verify the directory with:

  sudo -u postgres psql -c "SHOW data_directory;"

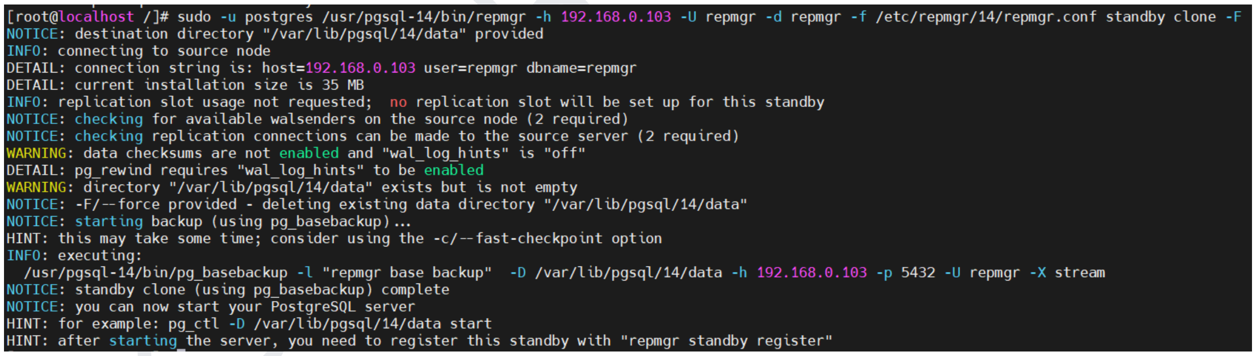
- **Stop PostgreSQL:** If PostgreSQL is running and using the data directory, stop the PostgreSQL service before proceeding with the repmgr clone operation:

  sudo systemctl stop postgresql-14.service

- **Verify the Directory Status:** Since the dry-run didn't show any issues with connecting or prerequisites, you should ensure that overwriting the existing data directory is the right step. Double-check that the /var/lib/pgsql/14/data directory should be overwritten and is not currently in use.
- **Proceed with Cloning (if appropriate):** If you're sure that overwriting the directory is correct, re-run the repmgr clone command with the -F/--force option:



```
[root@localhost /]# sudo systemctl stop postgresql-14
[root@localhost /]# sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone --dry-run
NOTICE: destination directory "/var/lib/pgsql/14/data" provided
INFO: connecting to source node
DETAIL: connection string is: host=192.168.0.103 user=repmgr dbname=repmgr
DETAIL: current installation size is 35 MB
INFO: "repmgr" extension is installed in database "repmgr"
WARNING: target data directory appears to be a PostgreSQL data directory
DETAIL: target data directory is "/var/lib/pgsql/14/data"
HINT: use -F/--force to overwrite the existing data directory
INFO: replication slot usage not requested;  no replication slot will be set up for this standby
INFO: parameter "max_wal_senders" set to 10
NOTICE: checking for available walsenders on the source node (2 required)
INFO: sufficient walsenders available on the source node
DETAIL: 2 required, 10 available
NOTICE: checking replication connections can be made to the source server (2 required)
INFO: required number of replication connections could be made to the source server
DETAIL: 2 replication connections required
WARNING: data checksums are not enabled and "wal_log_hints" is "off"
DETAIL: pg_rewind requires "wal_log_hints" to be enabled
NOTICE: standby will attach to upstream node 1
HINT: consider using the -c/--fast-checkpoint option
INFO: would execute:
  /usr/pgsql-14/bin/pg_basebackup -l "repmgr base backup"  -D /var/lib/pgsql/14/data -h 192.168.0.103 -p 5432 -U repmgr -X stream
INFO: all prerequisites for "standby clone" are met
```



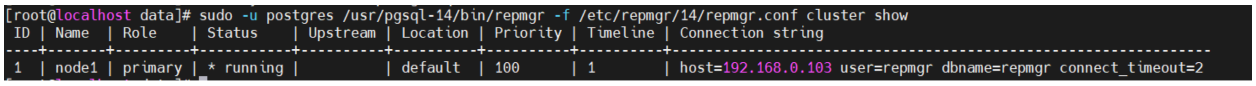
```
[root@localhost /]# sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone -F
NOTICE: destination directory "/var/lib/pgsql/14/data" provided
INFO: connecting to source node
DETAIL: connection string is: host=192.168.0.103 user=repmgr dbname=repmgr
DETAIL: current installation size is 35 MB
INFO: replication slot usage not requested;  no replication slot will be set up for this standby
NOTICE: checking for available walsenders on the source node (2 required)
NOTICE: checking replication connections can be made to the source server (2 required)
WARNING: data checksums are not enabled and "wal_log_hints" is "off"
DETAIL: pg_rewind requires "wal_log_hints" to be enabled
WARNING: directory "/var/lib/pgsql/14/data" exists but is not empty
NOTICE: -F/--force provided - deleting existing data directory "/var/lib/pgsql/14/data"
NOTICE: starting backup (using pg_basebackup)...
HINT: this may take some time; consider using the -c/--fast-checkpoint option
INFO: executing:
  /usr/pgsql-14/bin/pg_basebackup -l "repmgr base backup"  -D /var/lib/pgsql/14/data -h 192.168.0.103 -p 5432 -U repmgr -X stream
NOTICE: standby clone (using pg_basebackup) complete
NOTICE: you can now start your PostgreSQL server
HINT: for example: pg_ctl -D /var/lib/pgsql/14/data start
HINT: after starting the server, you need to register this standby with "repmgr standby register"
```

- **Start PostgreSQL:** Start the PostgreSQL service after proceeding with the repmgr clone operation:

  sudo systemctl start postgresql-14.service

- **Monitor the Process:** After running the command, monitor the PostgreSQL logs and the replication process to ensure that the standby setup proceeds as expected.

- **Post-Setup Verification:** After the cloning process completes, verify the standby server's status and ensure that it has correctly attached to the primary node. You can check this with:



```
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf cluster show
 ID | Name  | Role    | Status    | Upstream | Location | Priority | Timeline | Connection string
----+-------+---------+-----------+----------+----------+----------+----------+--------------------------------------------------------------
 1  | node1 | primary | * running |          | default  | 100      | 1        | host=192.168.0.103 user=repmgr dbname=repmgr connect_timeout=2
```

Here's what you can do to properly register the standby node:

- **Start the Standby Node** (if not already started):

  sudo systemctl start postgresql-14.service

- **Register the Standby Node**: Run the following command on the **standby node** to register it with the primary node in the repmgr cluster.

  sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf standby register

  This command registers the standby node with the primary node in the replication cluster.

- **Verify Registration**: After registering, verify the cluster's status by running the following command on **either node**:

  sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf cluster show

"Note: If you don't want to clone into the default directory with overwrite, you can delete the old directory and run the cloning process again without using the force option."

- **Navigate to the data directory:**

  cd /var/lib/pgsql/14/data/

- **Delete the contents of the directory:**

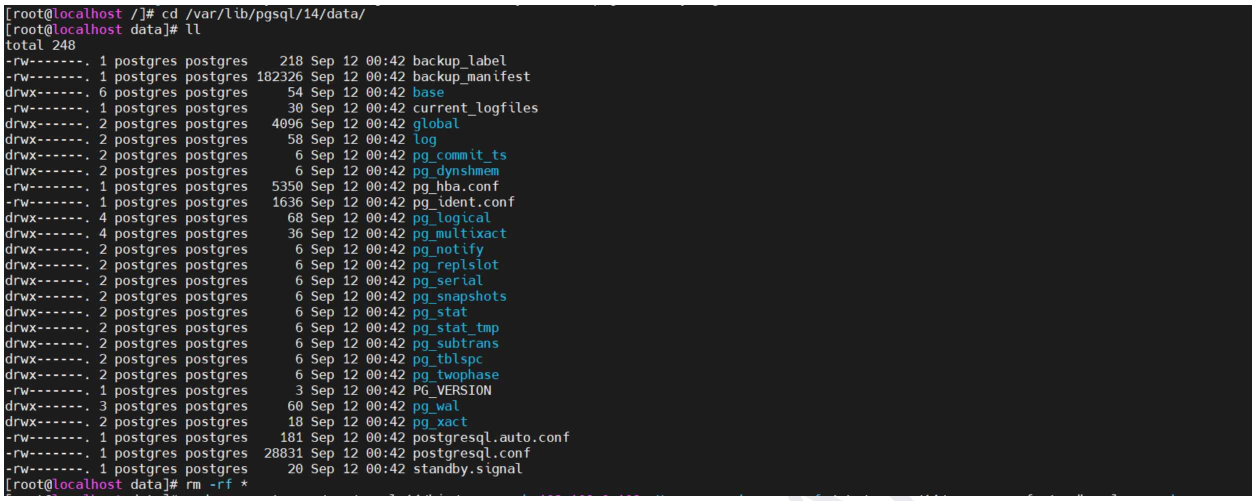
  rm -rf *

- **Run the dry-run of the standby clone:**

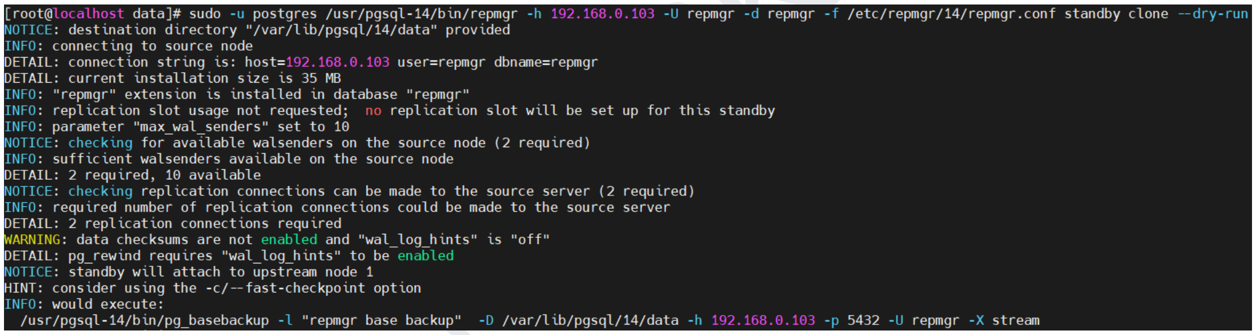
  sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone --dry-run

- **Run the actual standby clone:**

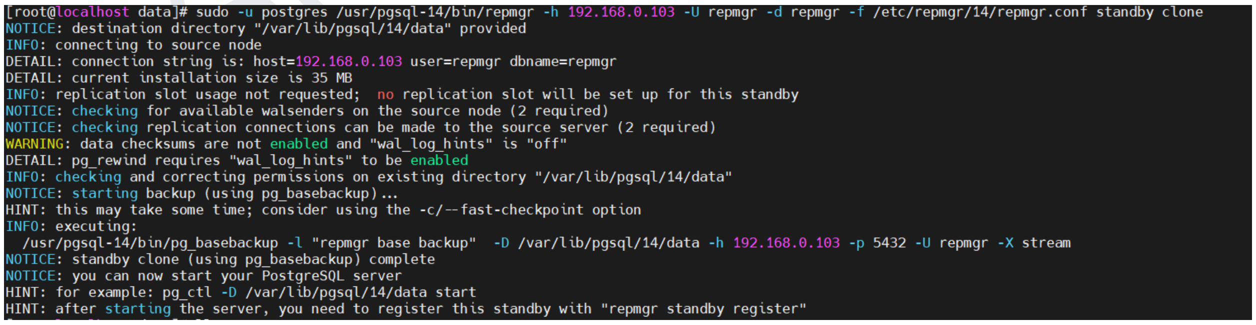
sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone



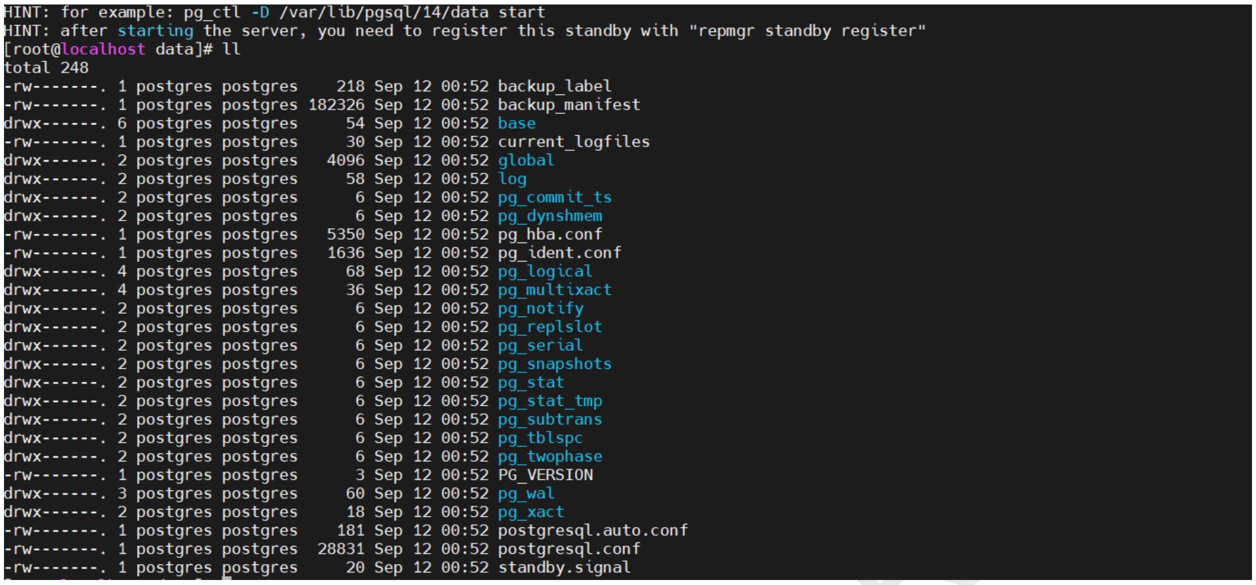
```
[root@localhost /]# cd /var/lib/pgsql/14/data/
[root@localhost data]# ll
total 248
-rw-------. 1 postgres postgres    218 Sep 12 00:42 backup_label
-rw-------. 1 postgres postgres 182326 Sep 12 00:42 backup_manifest
drwx------. 6 postgres postgres     54 Sep 12 00:42 base
-rw-------. 1 postgres postgres     30 Sep 12 00:42 current_logfiles
drwx------. 2 postgres postgres   4096 Sep 12 00:42 global
drwx------. 2 postgres postgres     58 Sep 12 00:42 log
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_commit_ts
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_dynshmem
-rw-------. 1 postgres postgres   5350 Sep 12 00:42 pg_hba.conf
-rw-------. 1 postgres postgres   1636 Sep 12 00:42 pg_ident.conf
drwx------. 4 postgres postgres     68 Sep 12 00:42 pg_logical
drwx------. 4 postgres postgres     36 Sep 12 00:42 pg_multixact
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_notify
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_replslot
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_serial
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_snapshots
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_stat
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_stat_tmp
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_subtrans
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_tblspc
drwx------. 2 postgres postgres      6 Sep 12 00:42 pg_twophase
-rw-------. 1 postgres postgres      3 Sep 12 00:42 PG_VERSION
drwx------. 3 postgres postgres     60 Sep 12 00:42 pg_wal
drwx------. 2 postgres postgres     18 Sep 12 00:42 pg_xact
-rw-------. 1 postgres postgres    181 Sep 12 00:42 postgresql.auto.conf
-rw-------. 1 postgres postgres  28831 Sep 12 00:42 postgresql.conf
-rw-------. 1 postgres postgres     20 Sep 12 00:42 standby.signal
[root@localhost data]# rm -rf *
```



```
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone --dry-run
NOTICE: destination directory "/var/lib/pgsql/14/data" provided
INFO: connecting to source node
DETAIL: connection string is: host=192.168.0.103 user=repmgr dbname=repmgr
DETAIL: current installation size is 35 MB
INFO: "repmgr" extension is installed in database "repmgr"
INFO: replication slot usage not requested;  no replication slot will be set up for this standby
INFO: parameter "max_wal_senders" set to 10
NOTICE: checking for available walsenders on the source node (2 required)
INFO: sufficient walsenders available on the source node
DETAIL: 2 required, 10 available
NOTICE: checking replication connections can be made to the source server (2 required)
INFO: required number of replication connections could be made to the source server
DETAIL: 2 replication connections required
WARNING: data checksums are not enabled and "wal_log_hints" is "off"
DETAIL: pg_rewind requires "wal_log_hints" to be enabled
NOTICE: standby will attach to upstream node 1
HINT: consider using the -c/--fast-checkpoint option
INFO: would execute:
  /usr/pgsql-14/bin/pg_basebackup -l "repmgr base backup"  -D /var/lib/pgsql/14/data -h 192.168.0.103 -p 5432 -U repmgr -X stream
```

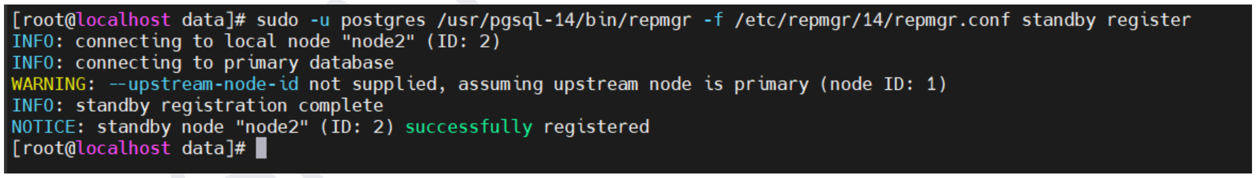


```
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -h 192.168.0.103 -U repmgr -d repmgr -f /etc/repmgr/14/repmgr.conf standby clone
NOTICE: destination directory "/var/lib/pgsql/14/data" provided
INFO: connecting to source node
DETAIL: connection string is: host=192.168.0.103 user=repmgr dbname=repmgr
DETAIL: current installation size is 35 MB
INFO: replication slot usage not requested;  no replication slot will be set up for this standby
NOTICE: checking for available walsenders on the source node (2 required)
NOTICE: checking replication connections can be made to the source server (2 required)
WARNING: data checksums are not enabled and "wal_log_hints" is "off"
DETAIL: pg_rewind requires "wal_log_hints" to be enabled
INFO: checking and correcting permissions on existing directory "/var/lib/pgsql/14/data"
NOTICE: starting backup (using pg_basebackup)...
HINT: this may take some time; consider using the -c/--fast-checkpoint option
INFO: executing:
  /usr/pgsql-14/bin/pg_basebackup -l "repmgr base backup"  -D /var/lib/pgsql/14/data -h 192.168.0.103 -p 5432 -U repmgr -X stream
NOTICE: standby clone (using pg_basebackup) complete
NOTICE: you can now start your PostgreSQL server
HINT: for example: pg_ctl -D /var/lib/pgsql/14/data start
HINT: after starting the server, you need to register this standby with "repmgr standby register"
```



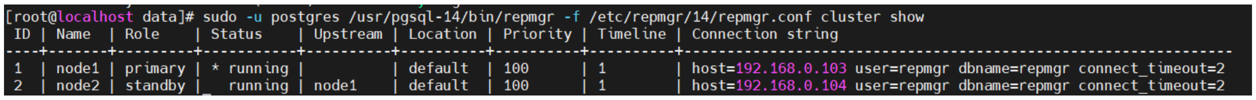
```
HINT: for example: pg_ctl -D /var/lib/pgsql/14/data start
HINT: after starting the server, you need to register this standby with "repmgr standby register"
[root@localhost data]# ll
total 248
-rw-------. 1 postgres postgres    218 Sep 12 00:52 backup_label
-rw-------. 1 postgres postgres 182326 Sep 12 00:52 backup_manifest
drwx------. 6 postgres postgres     54 Sep 12 00:52 base
-rw-------. 1 postgres postgres     30 Sep 12 00:52 current_logfiles
drwx------. 2 postgres postgres   4096 Sep 12 00:52 global
drwx------. 2 postgres postgres     58 Sep 12 00:52 log
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_commit_ts
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_dynshmem
-rw-------. 1 postgres postgres   5350 Sep 12 00:52 pg_hba.conf
-rw-------. 1 postgres postgres   1636 Sep 12 00:52 pg_ident.conf
drwx------. 4 postgres postgres     68 Sep 12 00:52 pg_logical
drwx------. 4 postgres postgres     36 Sep 12 00:52 pg_multixact
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_notify
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_replslot
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_serial
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_snapshots
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_stat
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_stat_tmp
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_subtrans
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_tblspc
drwx------. 2 postgres postgres      6 Sep 12 00:52 pg_twophase
-rw-------. 1 postgres postgres      3 Sep 12 00:52 PG_VERSION
drwx------. 3 postgres postgres     60 Sep 12 00:52 pg_wal
drwx------. 2 postgres postgres     18 Sep 12 00:52 pg_xact
-rw-------. 1 postgres postgres    181 Sep 12 00:52 postgresql.auto.conf
-rw-------. 1 postgres postgres  28831 Sep 12 00:52 postgresql.conf
-rw-------. 1 postgres postgres     20 Sep 12 00:52 standby.signal
```

12. Register the Standby Node with repmgr:



```
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf standby register
INFO: connecting to local node "node2" (ID: 2)
INFO: connecting to primary database
WARNING: --upstream-node-id not supplied, assuming upstream node is primary (node ID: 1)
INFO: standby registration complete
NOTICE: standby node "node2" (ID: 2) successfully registered
[root@localhost data]#
```

- **Check the cluster status:**

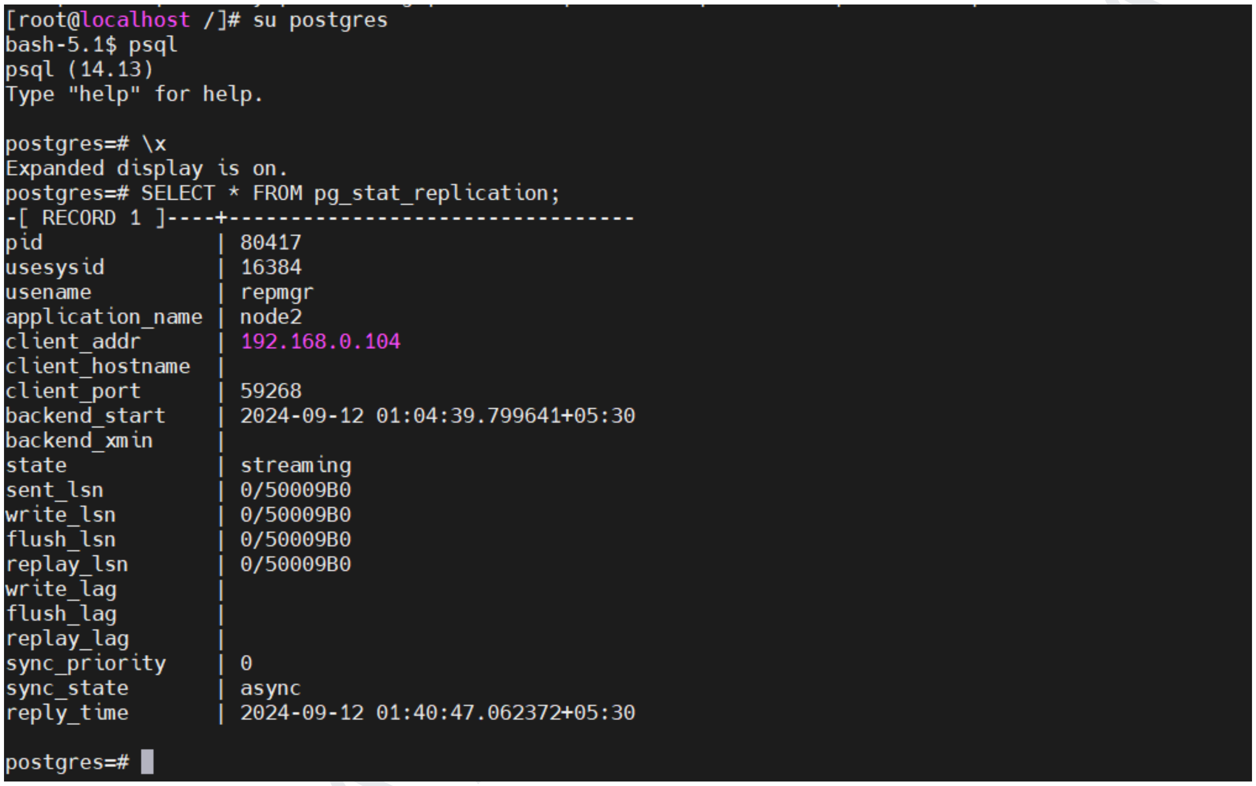


```
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf cluster show
 ID | Name  | Role    | Status    | Upstream | Location | Priority | Timeline | Connection string
----+-------+---------+-----------+----------+----------+----------+----------+--------------------------------------------------------------
 1  | node1 | primary | * running |          | default  | 100      | 1        | host=192.168.0.103 user=repmgr dbname=repmgr connect_timeout=2
 2  | node2 | standby |   running | node1    | default  | 100      | 1        | host=192.168.0.104 user=repmgr dbname=repmgr connect_timeout=2
```

13. Verify Replication:

- **On Primary Server:**

SELECT * FROM pg_stat_replication;



```
[root@localhost /]# su postgres
bash-5.1$ psql
psql (14.13)
Type "help" for help.

postgres=# \x
Expanded display is on.
postgres=# SELECT * FROM pg_stat_replication;
-[ RECORD 1 ]----+--------------------------------
pid              | 80417
usesysid         | 16384
usename          | repmgr
application_name | node2
client_addr      | 192.168.0.104
client_hostname  |
client_port      | 59268
backend_start    | 2024-09-12 01:04:39.799641+05:30
backend_xmin     |
state            | streaming
sent_lsn         | 0/50009B0
write_lsn        | 0/50009B0
flush_lsn        | 0/50009B0
replay_lsn       | 0/50009B0
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2024-09-12 01:40:47.062372+05:30

postgres=#
```

- **Create a database:**

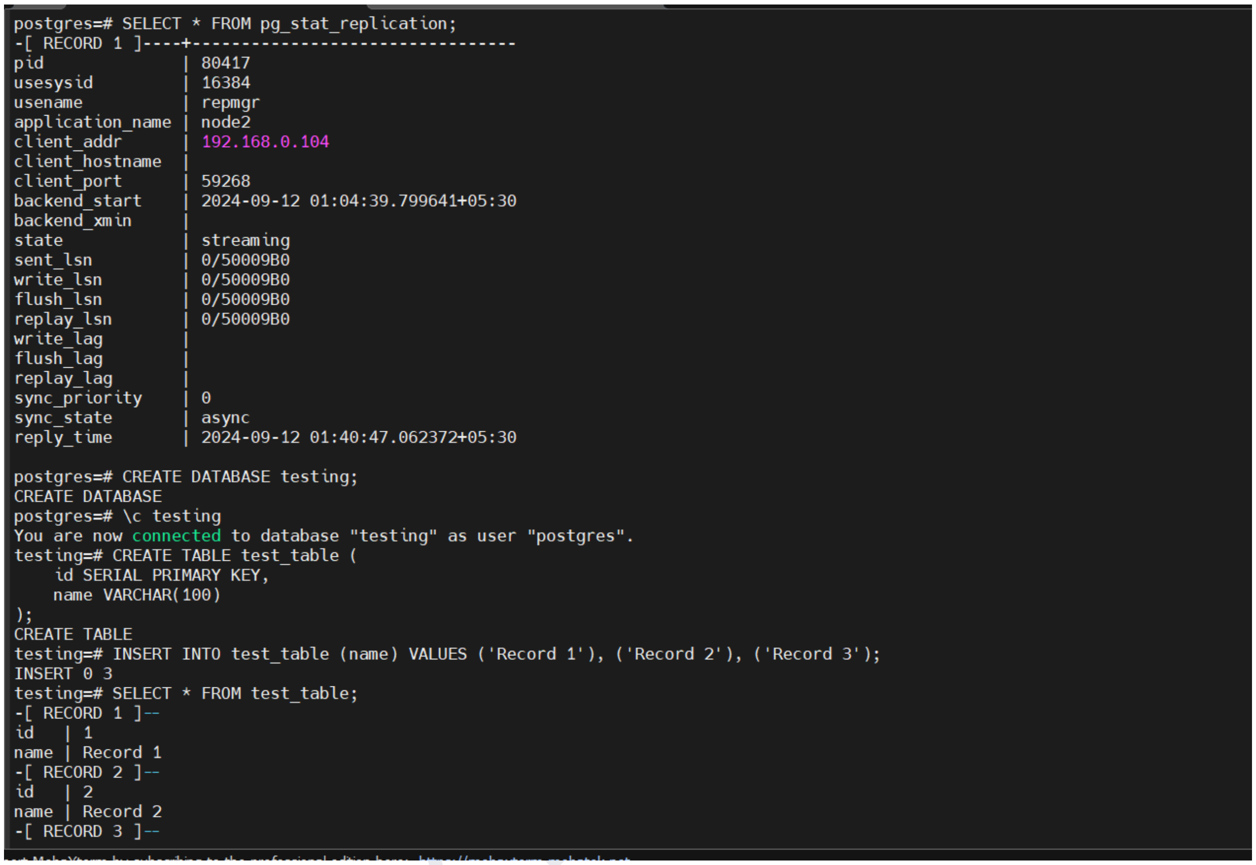
CREATE DATABASE testing;

- **Connect to the new database:**

\c testing

- **Create a table:**
- **Insert records into the table:**

- **Check that the records are inserted:**
- **On the Standby Node:**



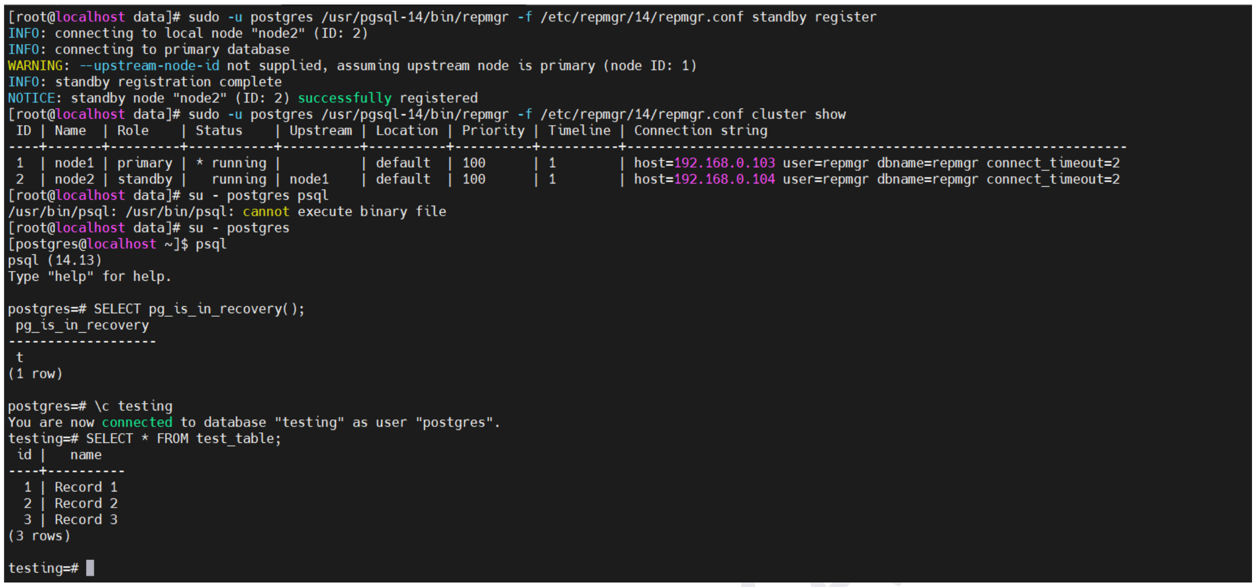
```
postgres=# SELECT * FROM pg_stat_replication;
-[ RECORD 1 ]----+---------------------------------
pid              | 80417
usesysid         | 16384
usename          | repmgr
application_name | node2
client_addr      | 192.168.0.104
client_hostname  |
client_port      | 59268
backend_start    | 2024-09-12 01:04:39.799641+05:30
backend_xmin     |
state            | streaming
sent_lsn         | 0/50009B0
write_lsn        | 0/50009B0
flush_lsn        | 0/50009B0
replay_lsn       | 0/50009B0
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2024-09-12 01:40:47.062372+05:30

postgres=# CREATE DATABASE testing;
CREATE DATABASE
postgres=# \c testing
You are now connected to database "testing" as user "postgres".
testing=# CREATE TABLE test_table (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100)
);
CREATE TABLE
testing=# INSERT INTO test_table (name) VALUES ('Record 1'), ('Record 2'), ('Record 3');
INSERT 0 3
testing=# SELECT * FROM test_table;
-[ RECORD 1 ]--
id   | 1
name | Record 1
-[ RECORD 2 ]--
id   | 2
name | Record 2
-[ RECORD 3 ]--
```

To verify replication, connect to the standby node and check if the changes made on the primary are replicated:

- **Connect to the standby node:**
- **Check if the table and records exist:**

1. Check if the standby is in recovery mode:

If it returns true, the server is in standby mode.



```
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf standby register
INFO: connecting to local node "node2" (ID: 2)
INFO: connecting to primary database
WARNING: --upstream-node-id not supplied, assuming upstream node is primary (node ID: 1)
INFO: standby registration complete
NOTICE: standby node "node2" (ID: 2) successfully registered
[root@localhost data]# sudo -u postgres /usr/pgsql-14/bin/repmgr -f /etc/repmgr/14/repmgr.conf cluster show
 ID | Name  | Role    | Status    | Upstream | Location | Priority | Timeline | Connection string
----+-------+---------+-----------+----------+----------+----------+----------+-------------------------------------------------------------
 1  | node1 | primary | * running |          | default  | 100      | 1        | host=192.168.0.103 user=repmgr dbname=repmgr connect_timeout=2
 2  | node2 | standby |   running | node1    | default  | 100      | 1        | host=192.168.0.104 user=repmgr dbname=repmgr connect_timeout=2
[root@localhost data]# su - postgres psql
/usr/bin/psql: /usr/bin/psql: cannot execute binary file
[root@localhost data]# su - postgres
[postgres@localhost ~]$ psql
psql (14.13)
Type "help" for help.

postgres=# SELECT pg_is_in_recovery();
 pg_is_in_recovery
-------------------
 t
(1 row)

postgres=# \c testing
You are now connected to database "testing" as user "postgres".
testing=# SELECT * FROM test_table;
 id |   name
----+----------
  1 | Record 1
  2 | Record 2
  3 | Record 3
(3 rows)

testing=# 
```