####### Replication with REPMGR #########

URL followed : https://www.enterprisedb.com/postgres-tutorials/how-implement-repmgr-postgresql-automatic-failover

Assumption :

master ip: 192.168.1.1
master data directory : /var/lib/pgsql/master

slave ip: 192.168.1.2
slave data directory : /var/lib/pgsql/standby/

We assume that Replication is already configured and is working fine

Now we are installing REPMGR for monitoring:

Installation steps of repmgr: (The tool should be installed on all nodes which we want in topology)

1:rpm --import http://packages.2ndquadrant.com/repmgr/RPM-GPG-KEY-repmgr

2: yum install http://packages.2ndquadrant.com/repmgr/yum-repo-rpms/repmgr-rhel-1.0-1.noarch.rpm

3: yum --showduplicates list repmgr96
note down the appropriate version for your machine

4:yum install repmgr96-4.0.0-1.rhel6

5: su - postgres

6: cd /usr/pgsql9.6/bin

7: you can see repmgr here

8: create repmgr.conf in your data directory

vi /var/lib/pgsql/master/repmgr.conf

```
    node_id=1
    node_name=192.168.1.1
    conninfo='host=192.168.1.1 user=postgres dbname=postgres connect_timeout=2'
    data_directory='/var/lib/pgsql/master/'
```

############update pg_hba.conf on master

```
# IPv4 local connections:
host    all             all             127.0.0.1/32            trust
host    postgres            postgres            192.168.1.1/32          trust
host  postgres    postgres         192.168.1.2/32         trust

# replication privilege.
```

```
#local   replication     postgres                                    trust
#host    replication     postgres        127.0.0.1/32            trust
#host    replication     postgres        ::1/128                 trust
host    replication     postgres        192.168.1.2/32           trust
host    replication      postgres             192.168.1.1/32            trust

save the files and restart the instance

./pg_ctl -D /var/lib/pgsql/master restart

9: register master server :
cd /usr/pgsql9.6/bin

./repmgr -f /var/lib/pgsql/master/repmgr.conf primary register

10: Verify status of the cluster like this:

psql

select * from repmgr.nodes;


#########SLAVE config ##########

1: ensure entry in pg_hba

 IPv4 local connections:
host    all             all             127.0.0.1/32            trust
host    postgres        postgres        192.168.1.2/32            trust
# IPv6 local connections:
host    all             all             ::1/128                 trust
# Allow replication connections from localhost, by a user with the
# replication privilege.
#local   replication     postgres                                    trust
#host    replication     postgres        127.0.0.1/32            trust
#host    replication     postgres        ::1/128                 trust
host    replication     postgres        192.168.1.2/32           trust




2: create repmgr.conf in data directory

      node_id=2
    node_name=192.168.1.2
    conninfo='host=192.168.1.2 user=postgres dbname=postgres connect_timeout=2'
    data_directory='/var/lib/pgsql/standby'

3 : save and restart the server

./pg_ctl -D /var/lib/pgsql/standby restart

4: dry run of repmgr to check the connectivity :

./repmgr -h 192.168.1.1 -U postgres -d postgres -f
/var/lib/pgsql/standby/rpgmgr.conf standby clone --dry-run


5: Register the slave
```

```
./repmgr -f /var/lib/pgsql/standby/repmgr.conf standby register
```

6: Monitoring from Master

```
select * from repmgr.nodes;
```

Some more operation with repmgr

->To promote a standby to be the new master, use the following command:

stop the master because if master is active then slave wont be promoted then fire the statement in slave machine

```
export PATH=$PATH:/usr/pgsql-9.6/bin/
repmgr standby promote
```

->To request a standby to follow a new master, use the following command:

```
repmgr standby follow
```

-> Check the status of each registered node in the cluster, like this:

```
repmgr cluster show
```