

**CertScan Postgres Database : HA Failover and Failback setup in Kubernets**



|  |  |
| --- | --- |
| S2 Global, Inc.  1800 Eller Drive, Suite 400  Fort Lauderdale, Florida 33316 USA |  |

Information Statement

PROPRIETARY STATEMENT: This document contains S2 Global, an OSI Systems Company proprietary information. Recipient, by accepting this document, agrees that neither the document nor any information contained within it shall be reproduced, transferred, used, or disclosed to others for manufacturing or any other purposes except as specifically authorized in writing by S2 Global.

EXPORT CONTROL STATEMENT: This document contains technical data whose export, reexport or transfer is restricted by the International Traffic in Arms Regulations (22 CFR Parts 120-130) or the Export Administration Regulations (15 CFR Parts 730-774). Violations of these export regulations are subject to severe criminal or civil penalties. This document and the technology described herein may not be transferred, or otherwise disposed of, to any person other than the authorized recipient, either in its original form or after being incorporated into other items, without first obtaining approval from the U.S. government or as otherwise authorized by U.S. law and regulations. Please contact S2 Global with any questions.

Table of Contents

1. Setup High Availablity for Postgresql Database 4

1.1 Pre-requisites 4

1.2 Master server (10.205.211.107) level setup 4

1.3 Standby server (10.205.211.11) level setup 14

1.4 Failback Old Primary Server (10.205.211.107) as Standby 29

Revision Record

| Rev. | Description | Author | Date |
| --- | --- | --- | --- |
| 1.0 | Draft Release | Surendra | 25/09/2024 |
|  |  |  |  |
|  |  |  |  |

1. Setup High Availablity for Postgresql Database

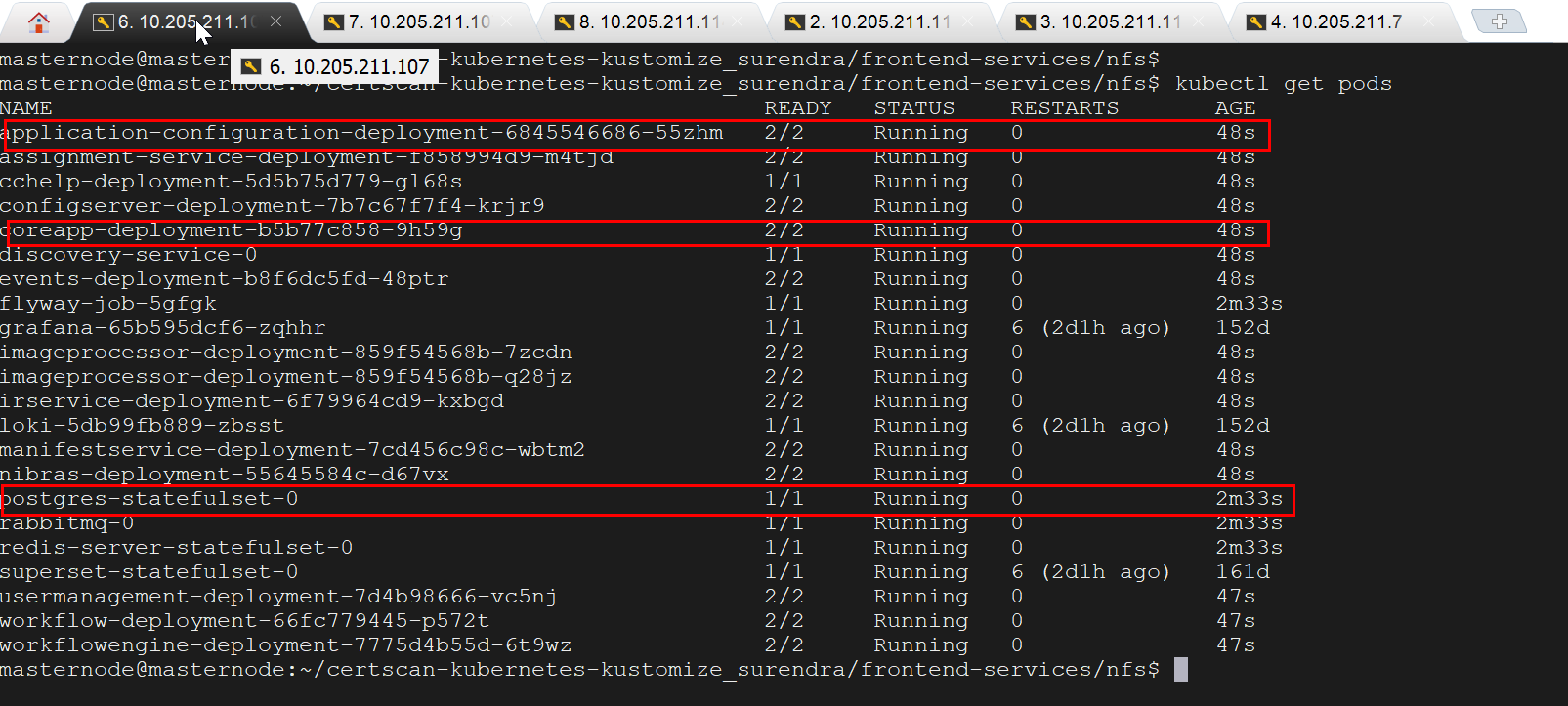
This document provides a detailed procedure on how to setup the streaming replication in kubernets environment for High Availability and repmgr setup for automatic failover to standby server in case of primary server failed.Aslo provides a detailed procedure on how to failback old primary as standby and data streaming from primary to standby. High availability refers to a system’s ability to operate continuously by removing the possibility of a single point of failure. PostgreSQL Database High Availability can be achieved by using streaming replication from Master to Slave (standby) servers. Streaming replication allows a standby server to stay more up-to-date than is possible with file-based log shipping. The standby connects to the primary, which streams WAL records to the standby as they're generated, without waiting for the WAL file to be filled. With the PostgreSQL and repmgr configurations in place, we can now use the repmgr tool to register our primary server node into the cluster. This will enable repmgr to monitor the health of our primary node, as well as any standby nodes, and to facilitate automatic failover in the event of a primary node failure.

* 1. Pre-requisites

We need 2 kubernetes clusters and NFS storage attached to those clusters and postgres ports are exposed outside. Connections are allowed between those clusters.

* 1. Master server (10.205.211.107) level setup

1. Certscan applications pods and postgres database should be running in the kubernets environment as shown in below screenshot.



1. Login to database as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

run the below commands

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**psql -U certscanuser -d certscandb**

**CREATE ROLE repuser REPLICATION LOGIN PASSWORD 'password';**

**alter system set wal\_level = replica;**

**alter system set max\_wal\_senders = 20;**

**alter system set max\_replication\_slots = 300;**

**alter system set hot\_standby = on;**

**select pg\_create\_physical\_replication\_slot('replication\_slot\_slave1');**

**alter system set primary\_slot\_name = 'replication\_slot\_slave1';**

**create user repmgr superuser ;**

**create database repmgr with owner repmgr;**

**alter user repmgr with password 'password';**

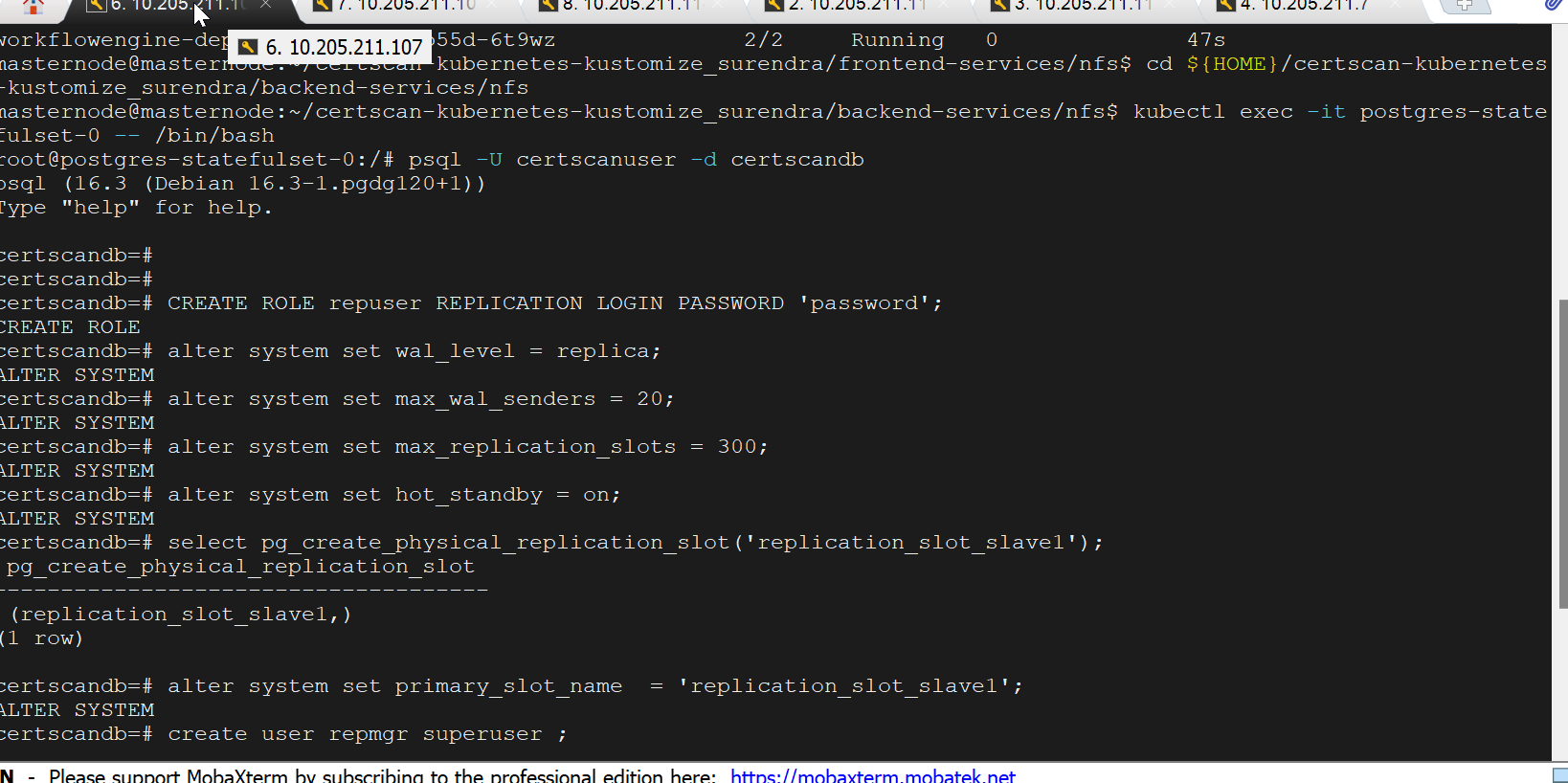
**alter system set archive\_mode = on;**

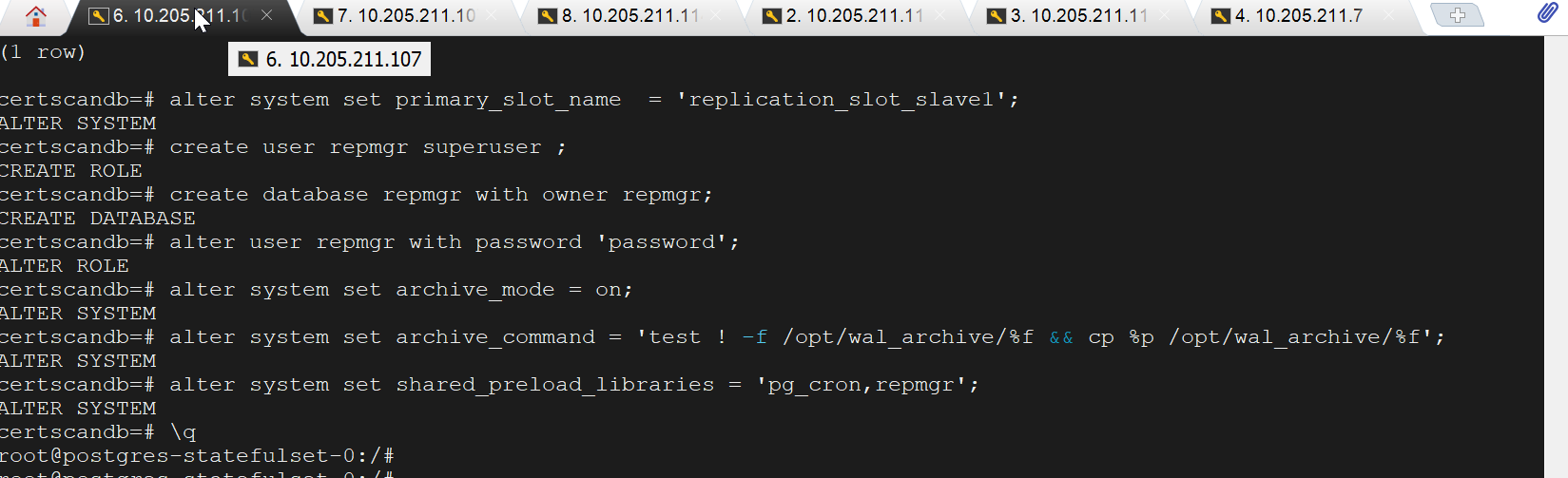
**alter system set wal\_log\_hints = 'on';**

**alter system set archive\_command = 'test ! -f /opt/wal\_archive/%f && cp %p /opt/wal\_archive/%f';**

**alter system set shared\_preload\_libraries = pg\_cron,repmgr;**

**\q**





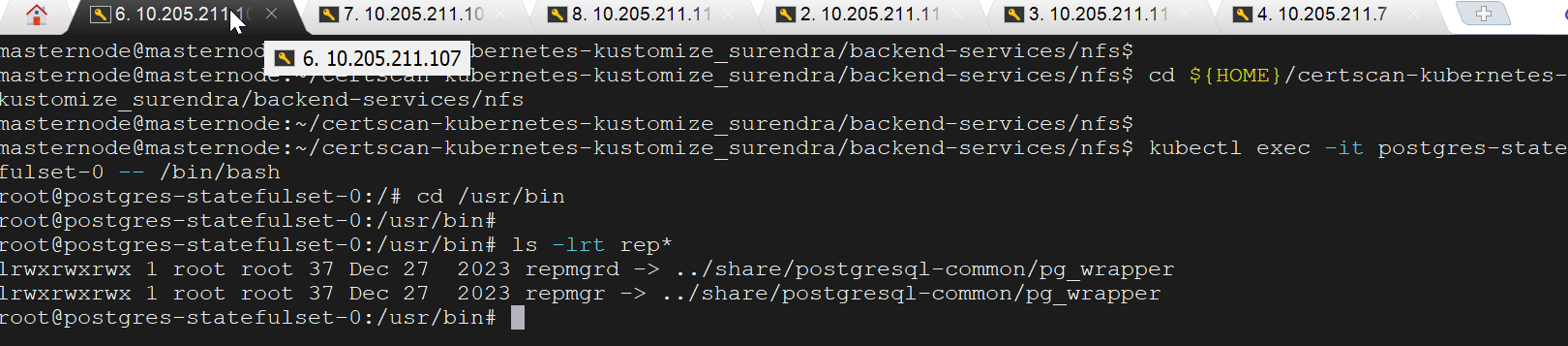
1. Check the repmgr is installed in kubernets postgres pod as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**cd /usr/bin**

**ls -lrt rep\***



1. Go to data directory mapped in nfs server (10.205.211.114) and edit the pg\_hba.conf file and add the below entries as shown below.

**vi config\_changes.sh**

Press **“ i”** to insert the below entries (copy and paste) , press **Esc + :wq** to save the changes, and then press Enter to exit.

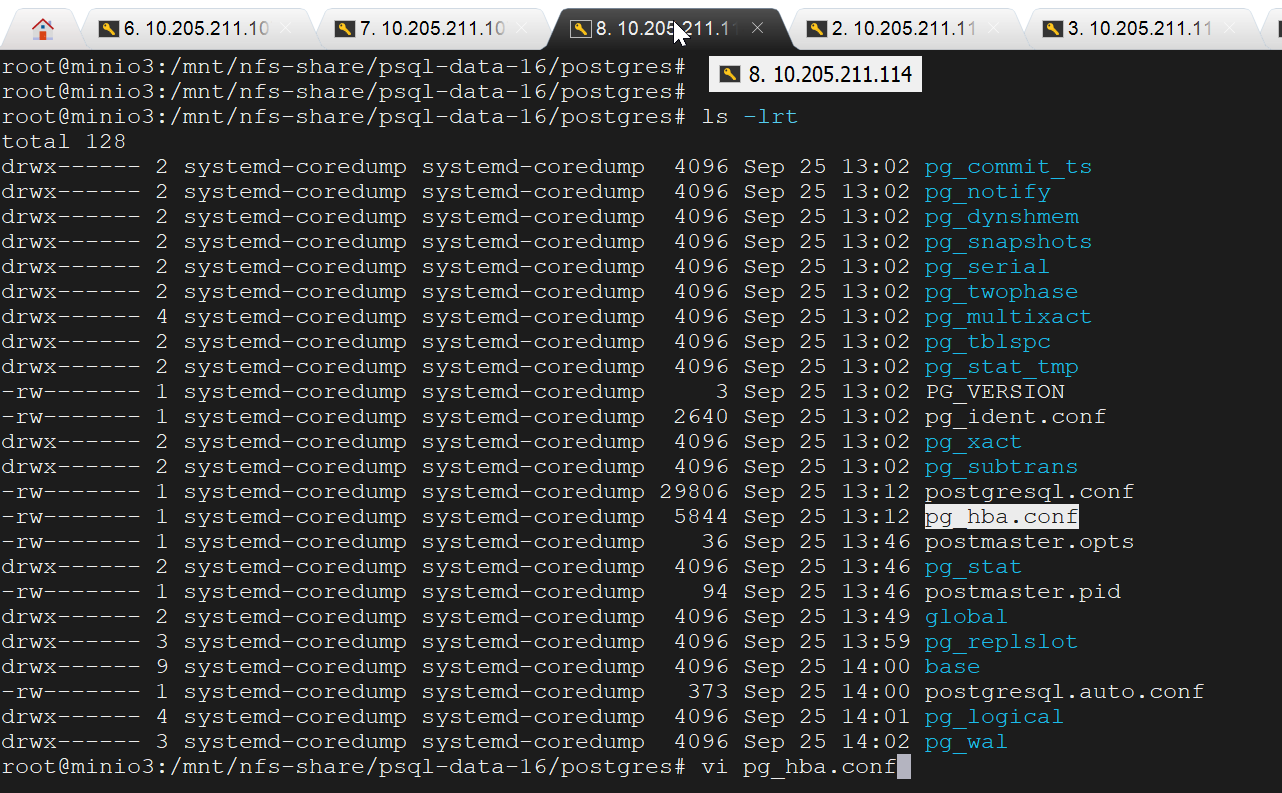
host all repuser all trust

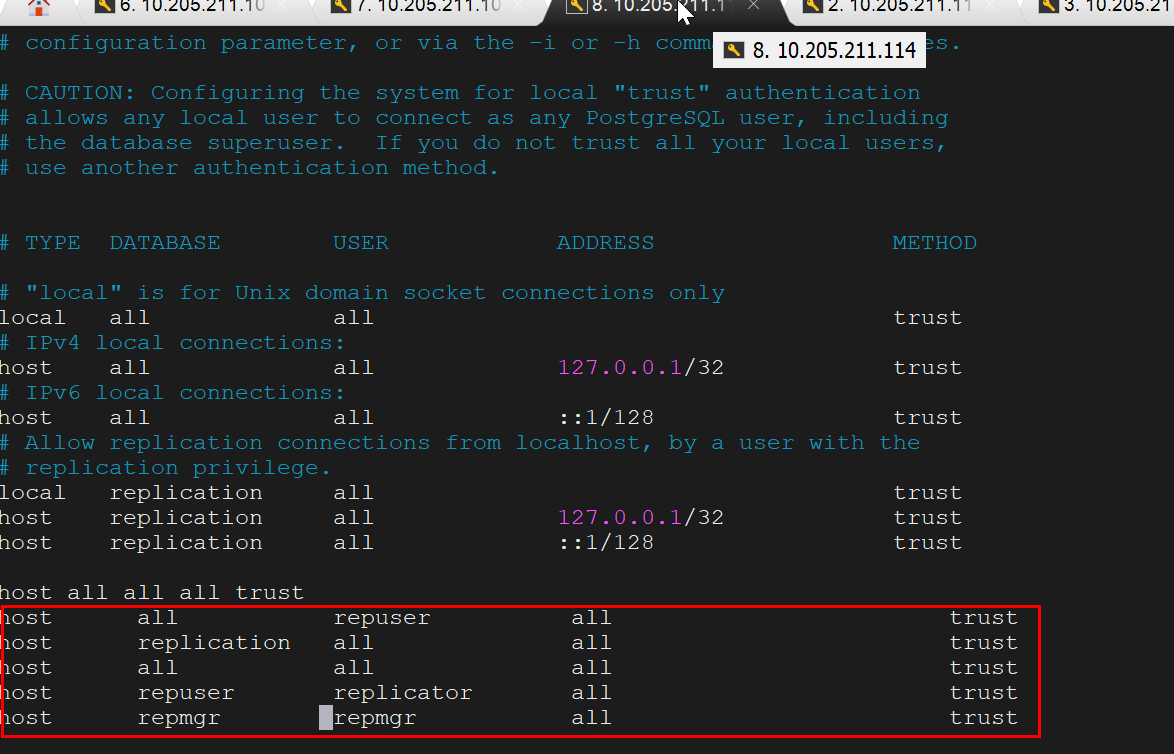
host replication all all trust

host all all all trust

host repuser replicator all trust

host repmgr repmgr all trust



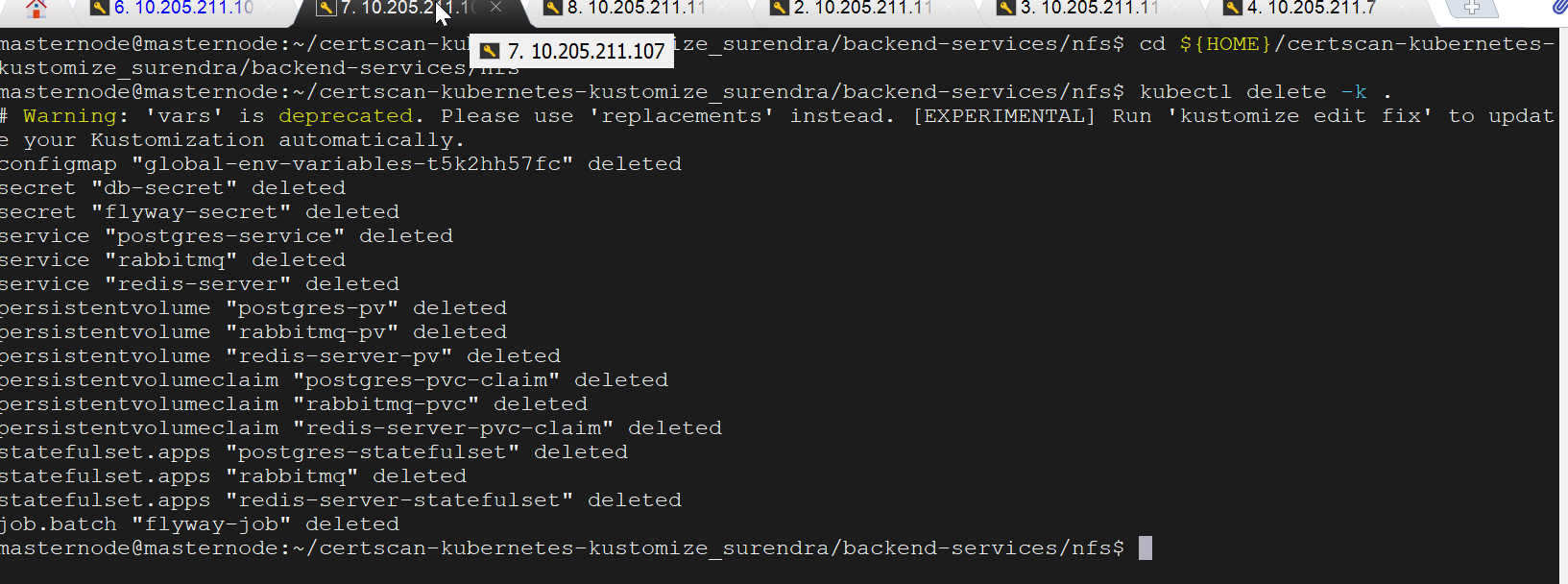


1. Stop and start the kubernets pods as shown below to reload the parameter changes executed in step 2 and step 4.

Execute the below command to stop pods.

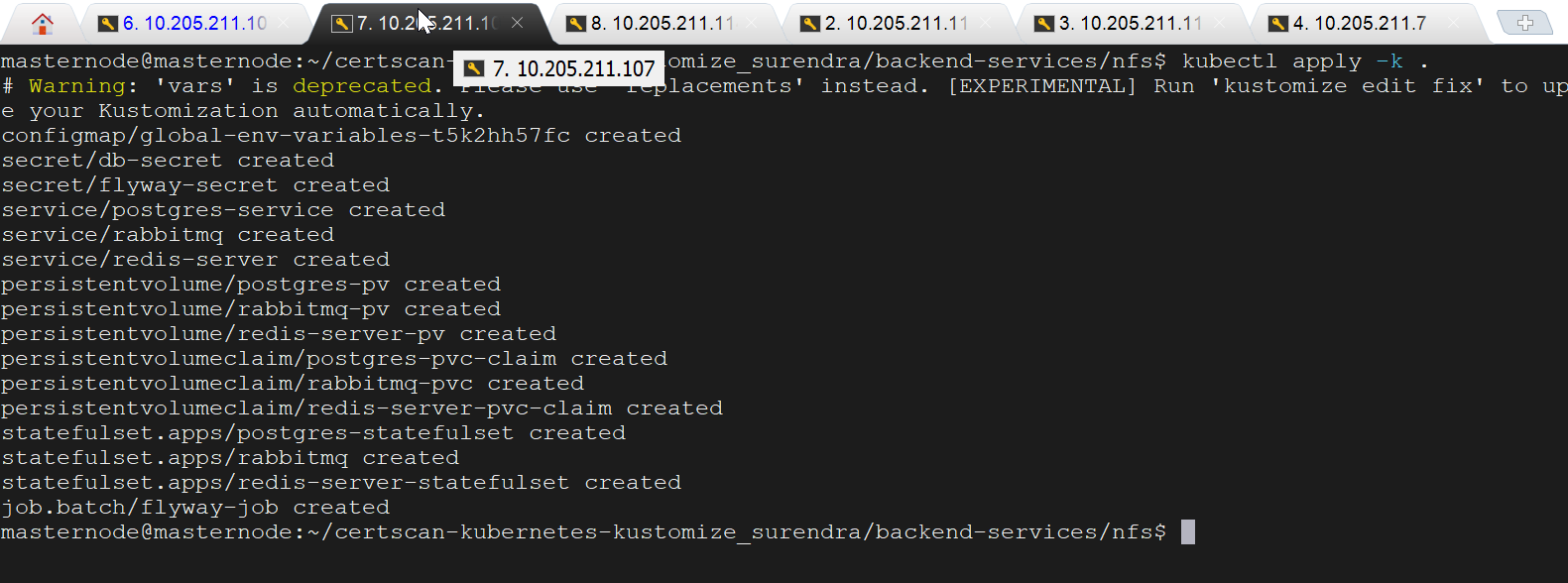
**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl delete -k .**



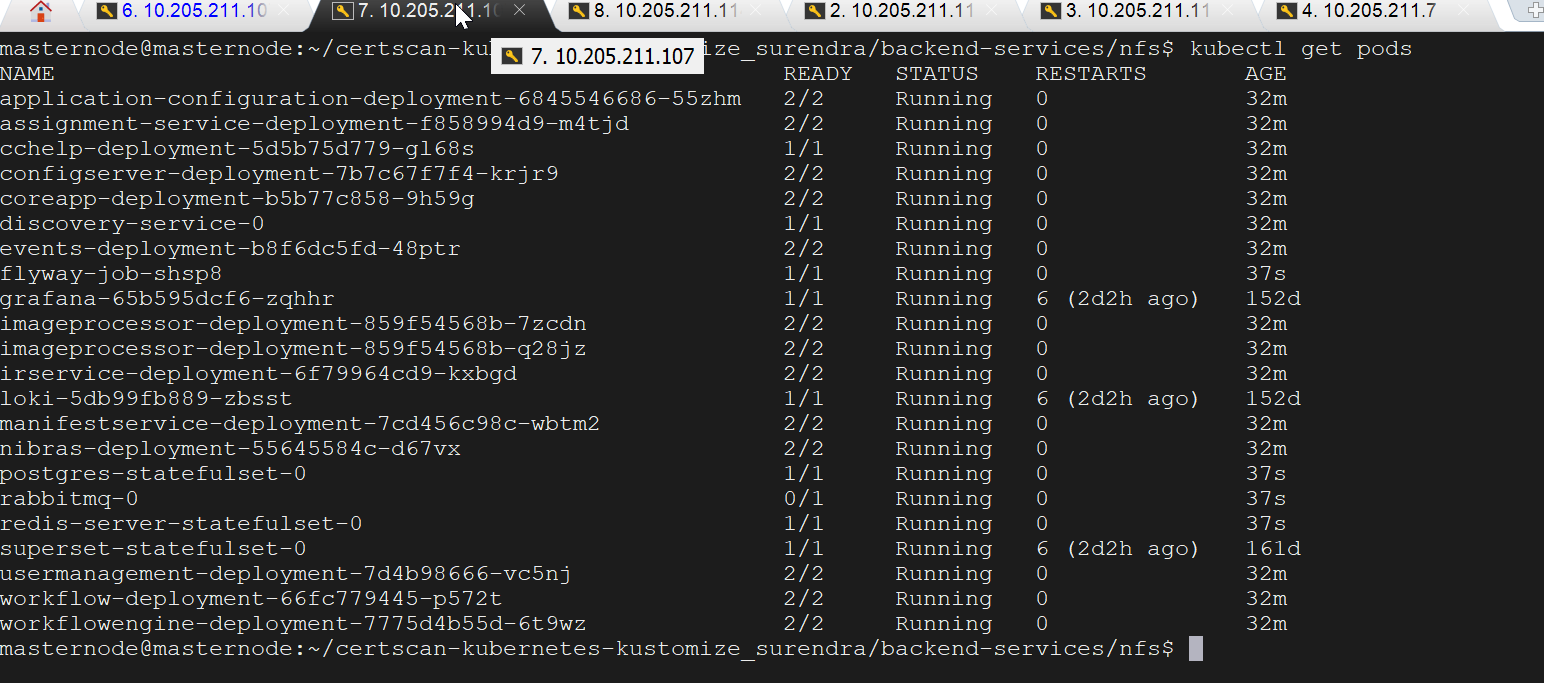
Execute the below command to start pods.

**kubectl apply -k .**



Execute the below command and check all pods are running in status.

kubectl get pods

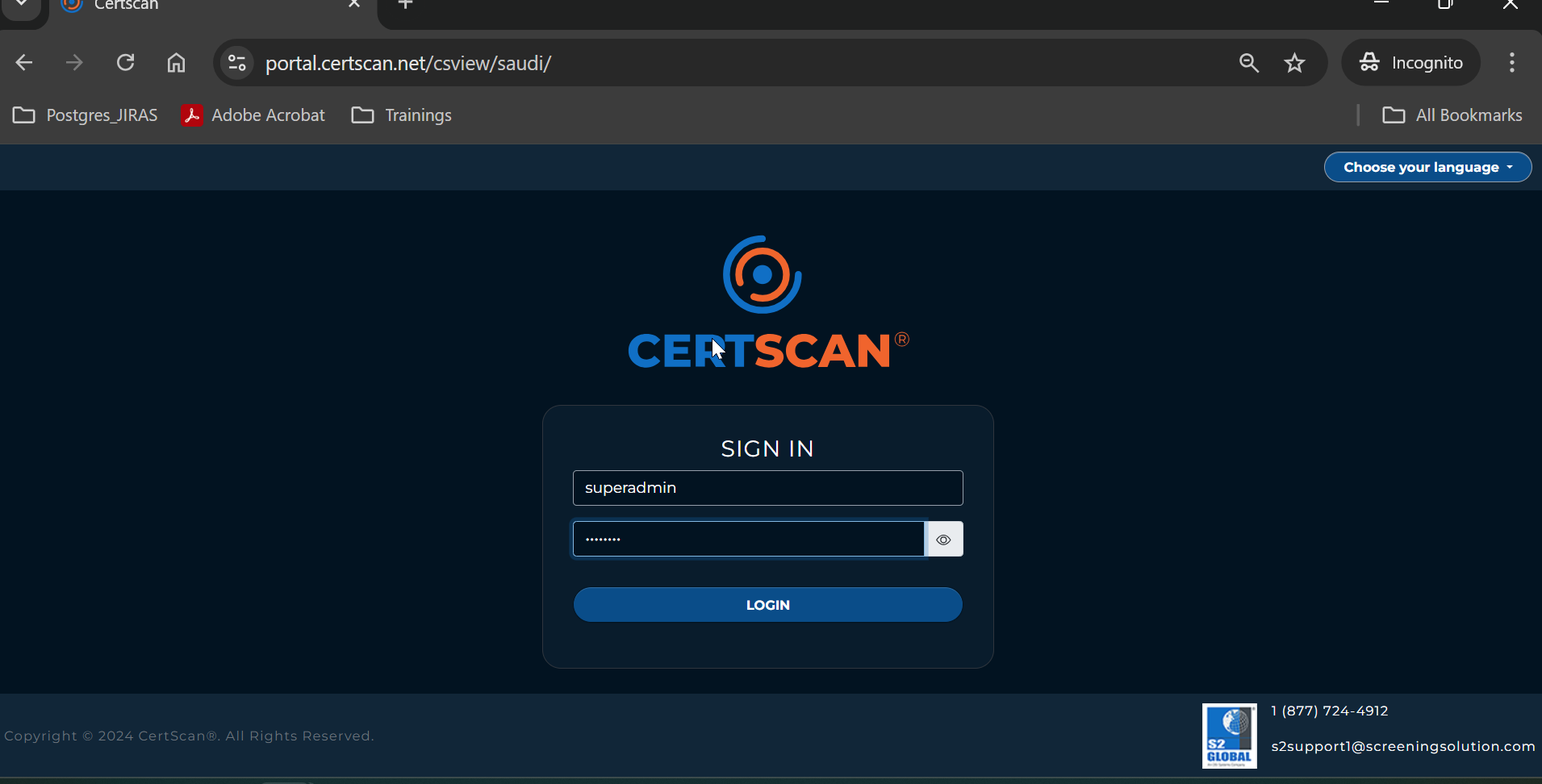


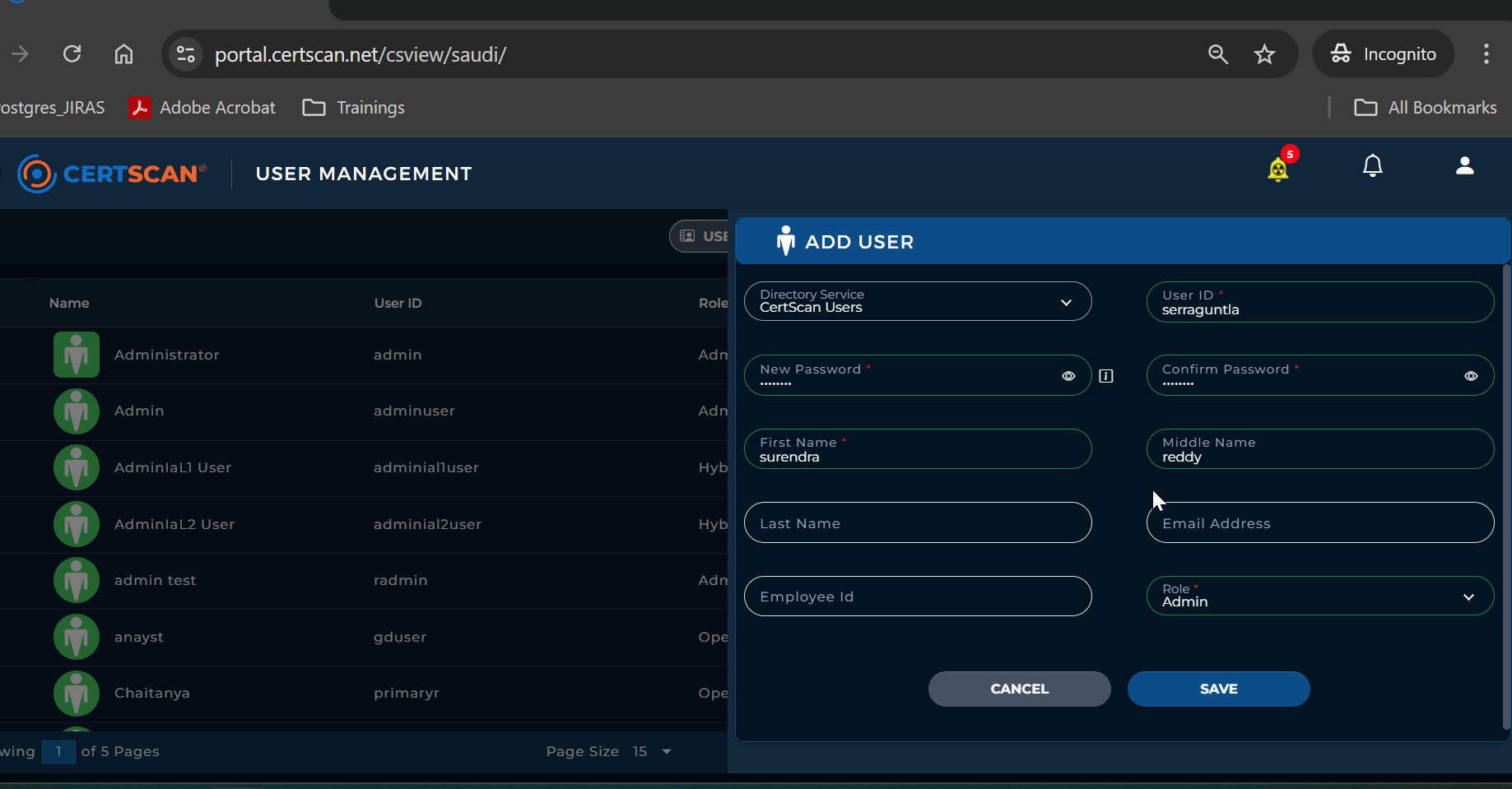
1. Login to application and add user and note the user details replicated and displayed in standby sever database after streaming replication started from primary to standby server.

url: <https://portal.certscan.net/csview/cbp/>

username: superadmin

Password: test@123





A screenshot of a computer

Description automatically generated

1. Login to standby server database after streaming replication started and check the user data which is added in step 6 is replicated and displayed as expected.

**cd ${HOME}/ certscan-kubernetes-kustomize/backend-services/nfs**

run the below commands

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**psql -U certscanuser -d certscandb**

**\c certscandb**

**Set search\_path=saudi;**

**Select \* from cfg\_users;**

A screen shot of a computer

Description automatically generated



1. Login to nfs server (10.205.211.114) and prepare the repmgr.conf file for primary server register with repmgr and failover in case of primary server failure.

**sudo su**

**cd /mnt/nfs-share/dbscript/scripts**

Note: This is the volume path and mapped to postgres pod directory /home/dbscripts/scripts

**vi repmgr.conf**

Press **“ i”** to insert the below entries (copy and paste) , press **Esc + :wq** to save the changes, and then press Enter to exit.

**node\_id=1**

**node\_name=masternode**

**conninfo='host=10.205.211.107 port=30009 user=repmgr dbname=repmgr password=password connect\_timeout=2'**

**data\_directory='/var/lib/postgresql/data'**

**failover=automatic**

**promote\_command='/usr/bin/repmgr -f /home/dbscript/scripts/repmgr.conf standby promote'**

**follow\_command='/usr/bin/repmgr -f /home/dbscript/scripts/repmgr.conf stamdby follow'**

1. Login to postgres pod and register the primary server with repmgr for automatic failover in case for primary server failure.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

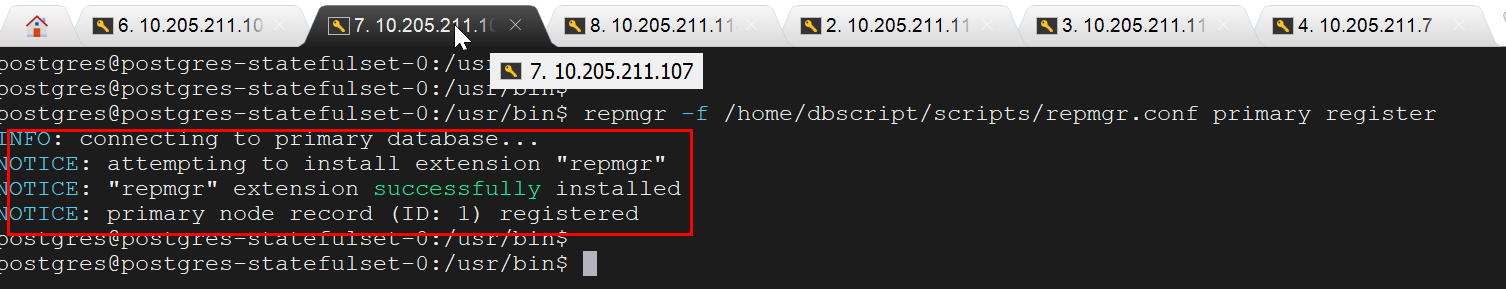
run the below commands

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**su postgres**

**cd /usr/bin**

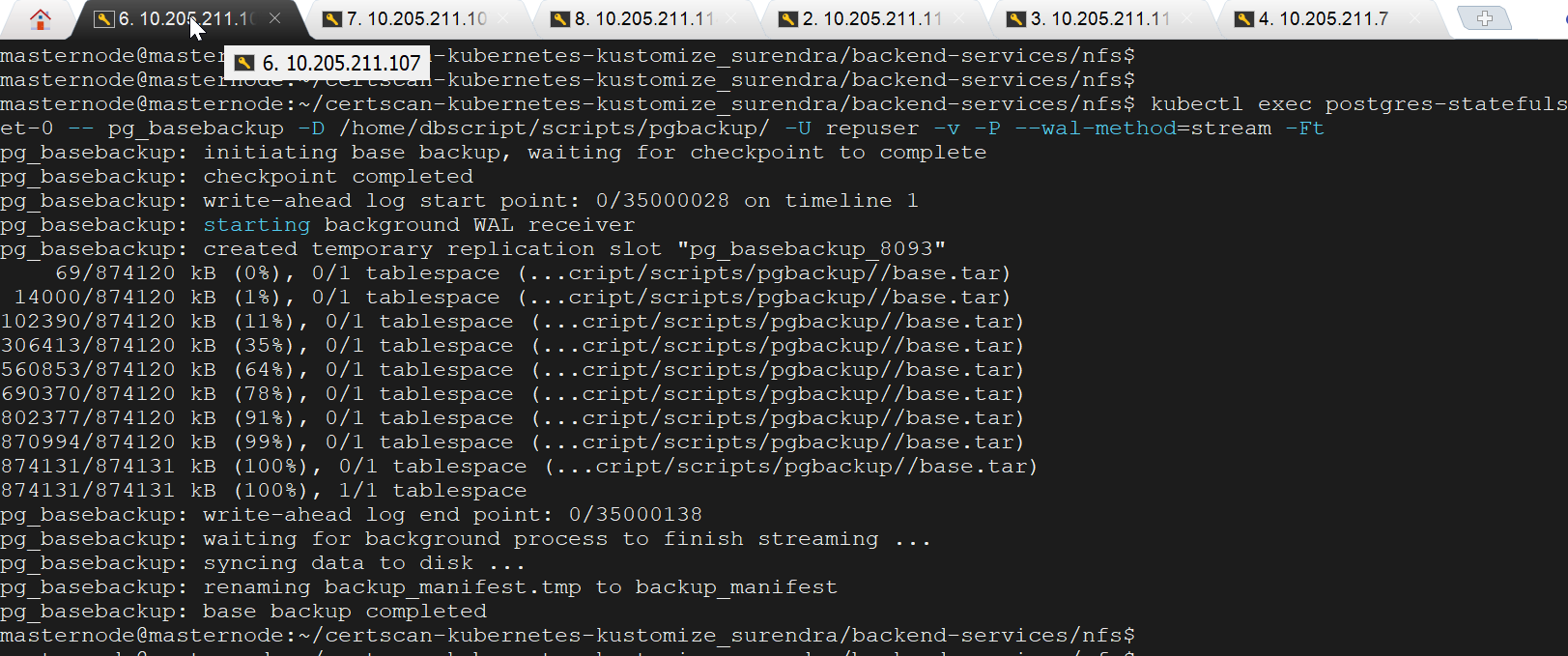
**repmgr -f /home/dbscript/scripts/repmgr.conf primary register**

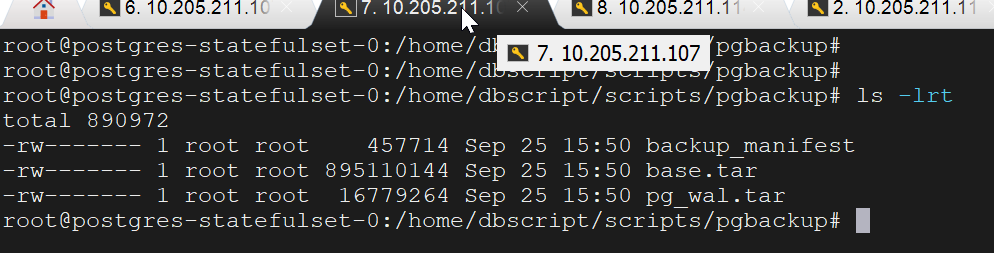
****

1. Take database backup by executing the below command for streaming replication.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

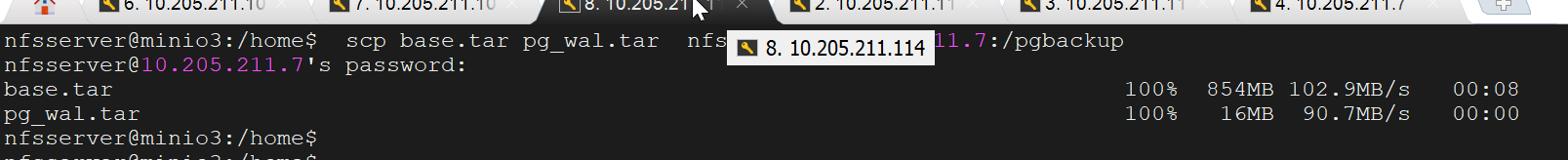
**kubectl exec postgres-statefulset-0 -- pg\_basebackup -D /home/dbscript/scripts/pgbackup/ -U repuser -v -P --wal-method=stream -Ft**

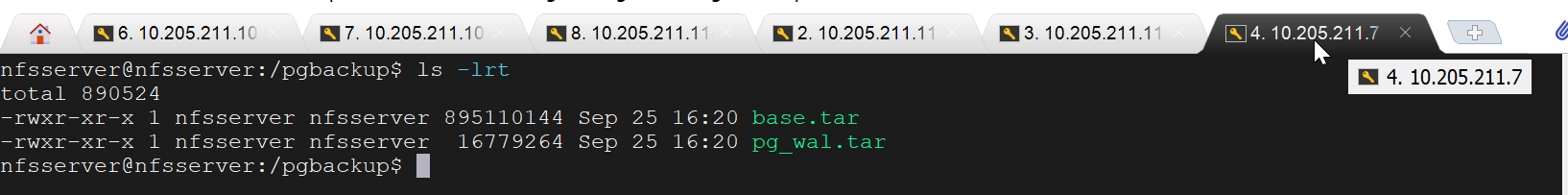
****

****

1. Copy the backup files(base.tar,pg\_wal.tar) from primary server nfs server (10.205.211.114) to standby server nfs server (10.205.211.7) using scp command.

**scp base.tar pg\_wal.tar** [nfsserver@10.205.211.7:/](mailto:nfsserver@10.205.211.7:/)**pgbackup**





1. Enable the repmgrd demon process for monitoring and failover handling.

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**su postgres**

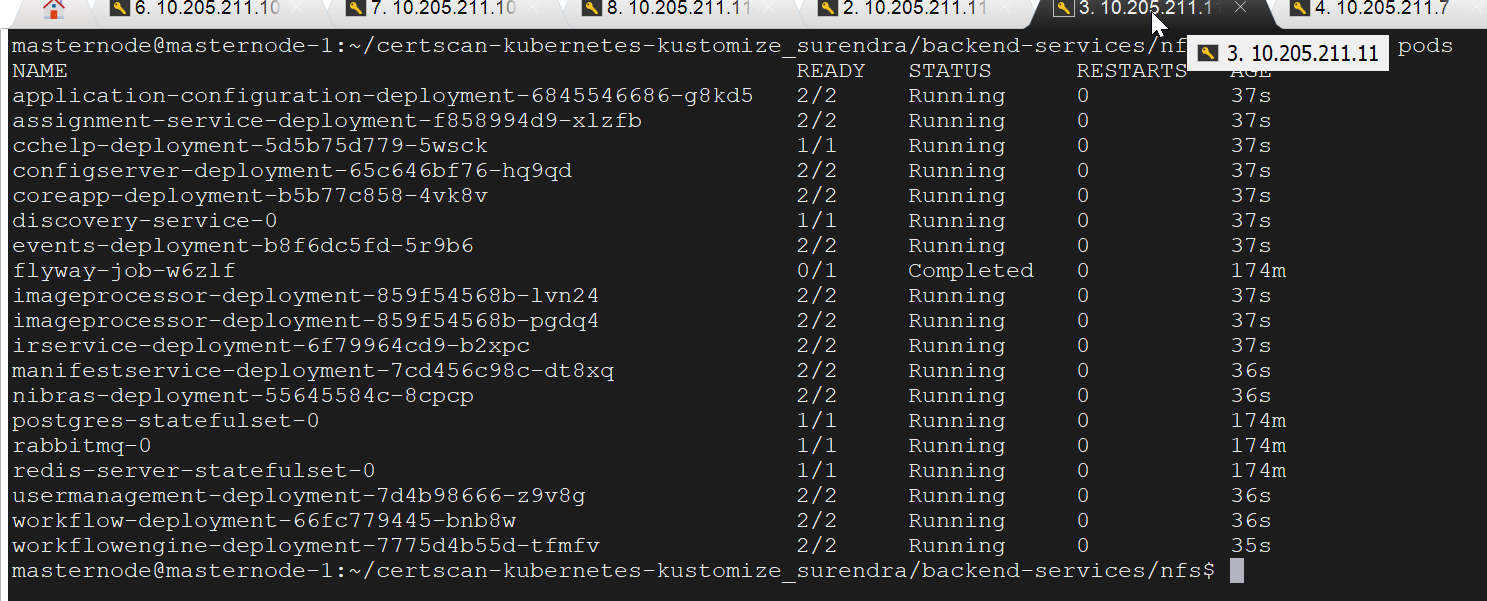
**cd /usr/bin**

**repmgrd -f /home/dbscript/scripts/repmgr.conf -d**



* 1. Standby server (10.205.211.11) level setup

Certscan application pods and postgres database should be running in the kubernets environment as shown in below screenshot.



Stop the application and postgres database pods as shown below.

To stop pods execute the below command.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

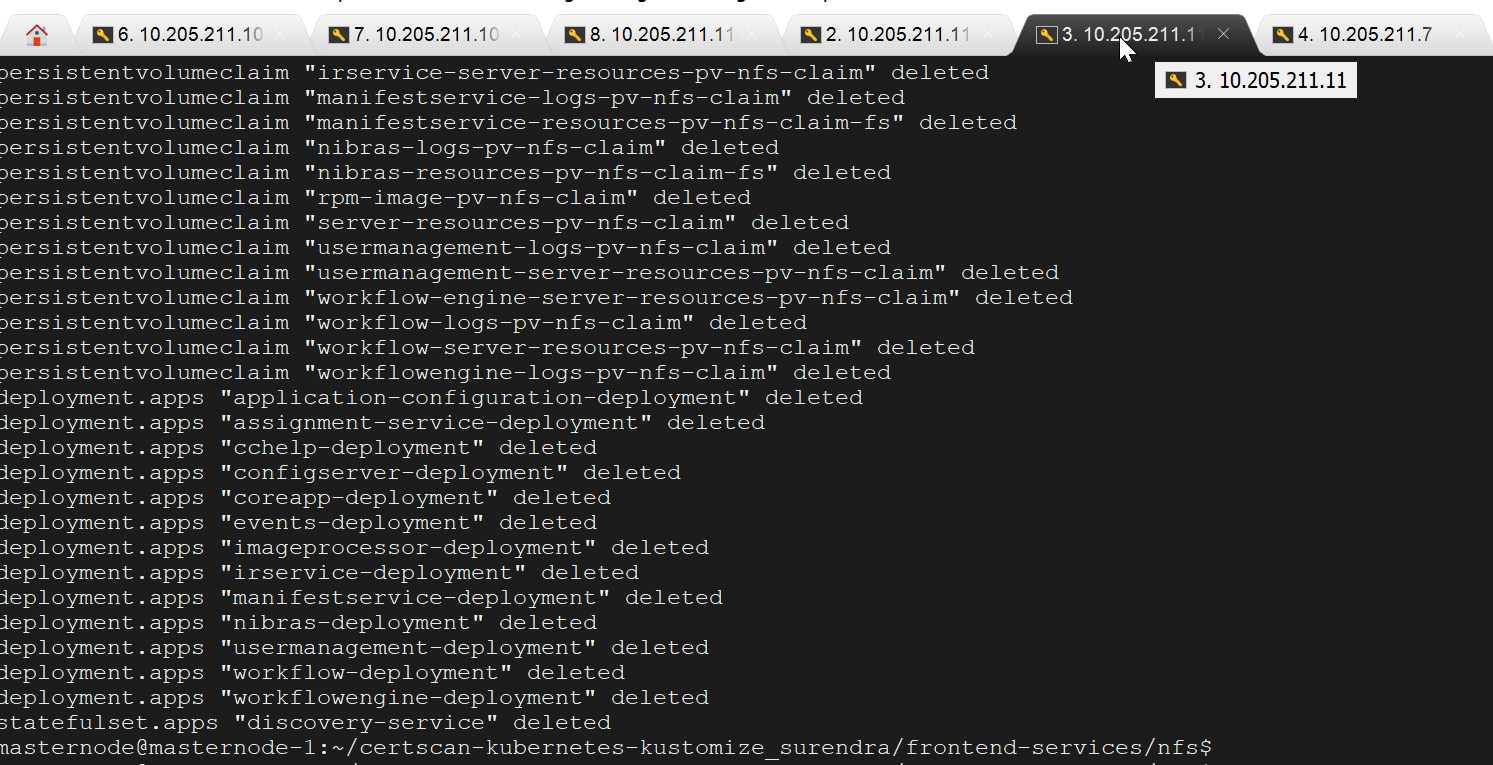
**kubectl delete -k .**

A computer screen shot of a black screen

Description automatically generated

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/** **frontend-services/nfs**

**kubectl delete -k .**

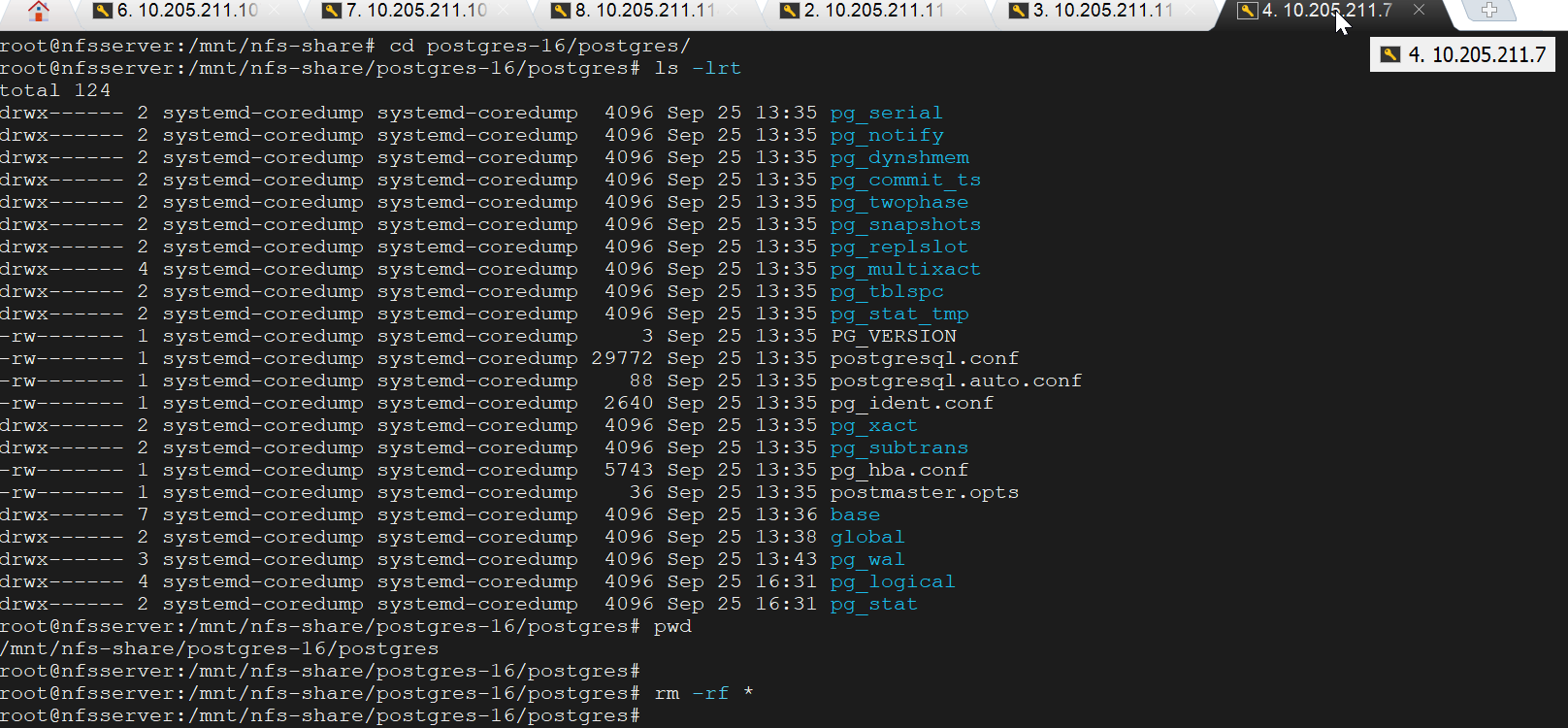


Login to nfs server (10.205.211.7) and go to data directory volume path and remove the contents as shown below.

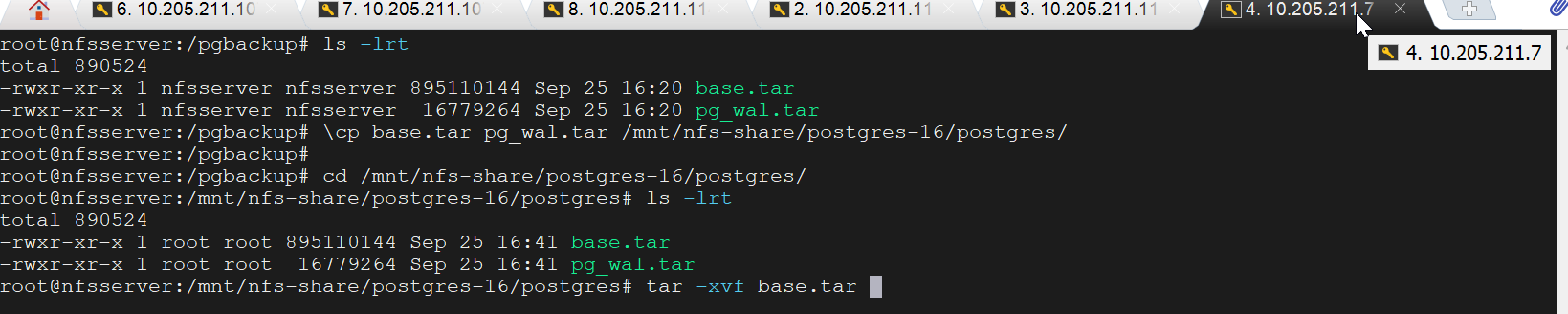
**sudo su**

**cd /mnt/nfs-share/postgres-16/postgres**

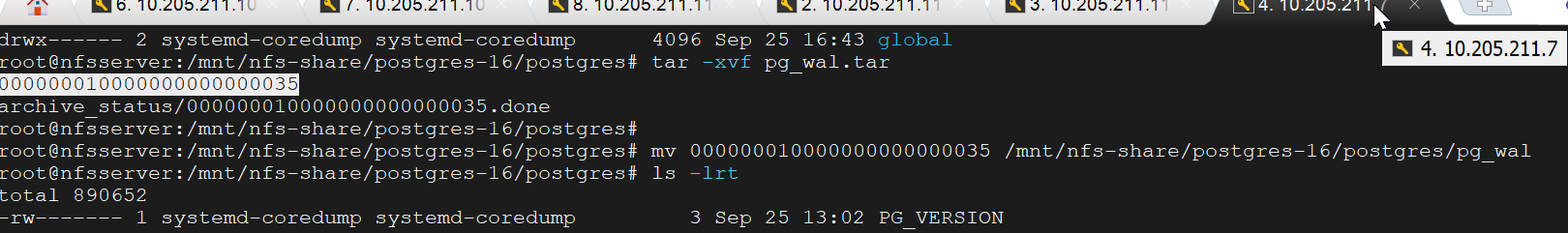
**rm -rf \***



Copy the backup files from /pgbackup to data directory volume path and extract the files (base.tar,pg\_wal.tar) as shown below.



Move the WAL file to pg\_wal directory after extract the pg\_wal.tar as shown below



Edit the postgresql.conf and set the below parameters as shown below.

**vi postgresql.conf**

Press **“ i”** to insert the below entries (copy and paste) , press **Esc + :wq** to save the changes, and then press Enter to exit.

**listen\_addresses = '\*'**

**wal\_level = replica**

**hot\_standby = on**

**archive\_mode = on**

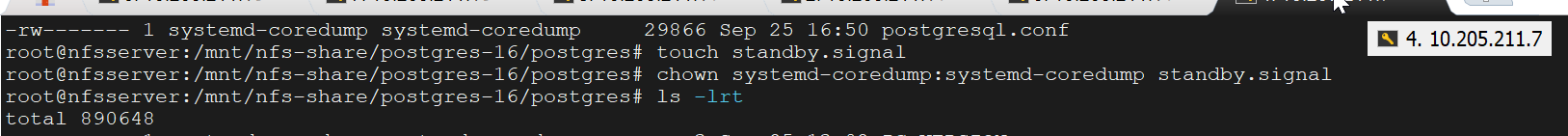
**archive\_command = 'test ! -f /opt/wal\_archive/%f && cp %p /opt/wal\_archive/%f'**

**primary\_conninfo = 'host=10.205.211.107 port=30009 user=repuser password=password'**

Create standby.sigan file and change the permissions as shown below.

**touch standby.signal**

**chown systemd-coredump:systemd-coredump standby.signal**



Create the repmgr.conf in the **/mnt/nfs-share/dbscript/scripts** directory for automatic failover and register the standy server.

**vi repmgr.conf**

Press **“ i”** to insert the below entries (copy and paste) , press **Esc + :wq** to save the changes, and then press Enter to exit.

**node\_id=2**

**node\_name=masternode1**

**conninfo='host=10.205.211.11 port=30009 user=repmgr dbname=repmgr password=password connect\_timeout=2'**

**data\_directory='/var/lib/postgresql/data'**

**failover=automatic**

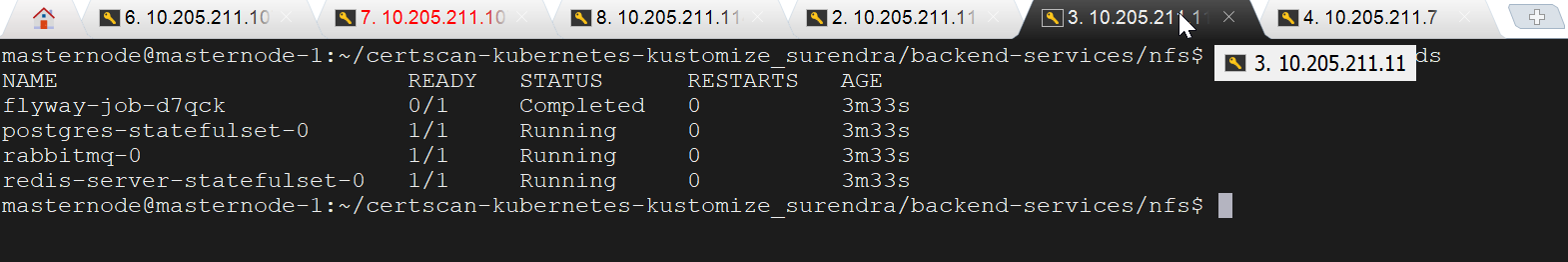
**promote\_command='repmgr -f /home/repmgr.conf standby promote'**

**follow\_command='repmgr -f /home/repmgr.conf stamdby follow'**

Start the postgres database pods as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl apply -k .**

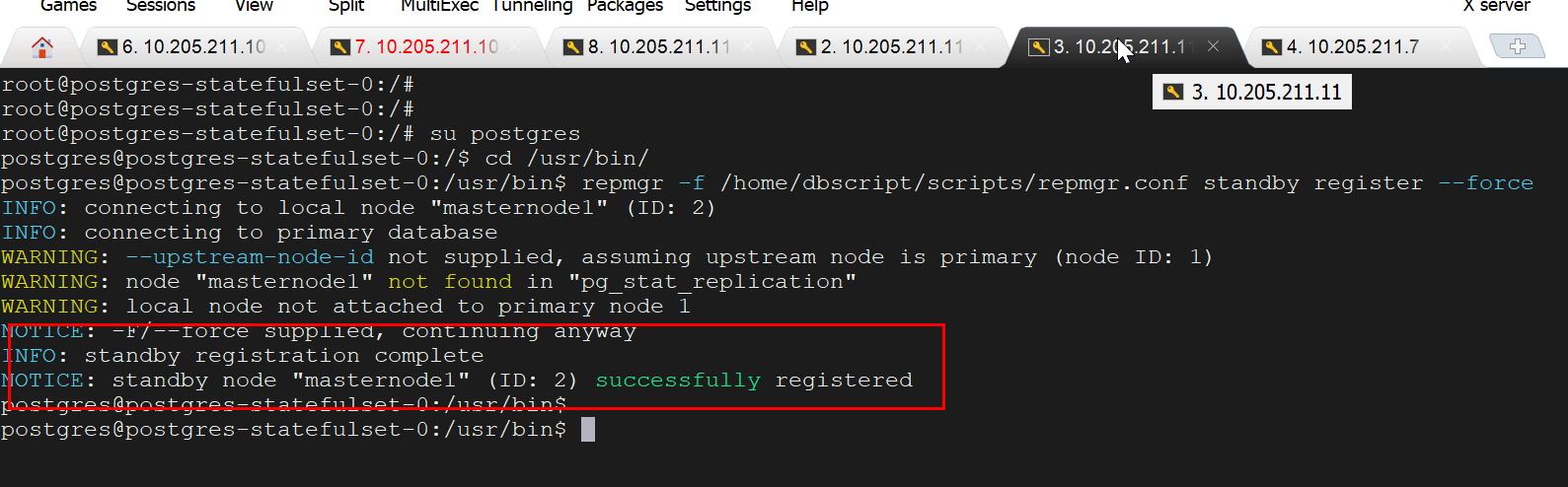


Register the standby server.

**su postgres**

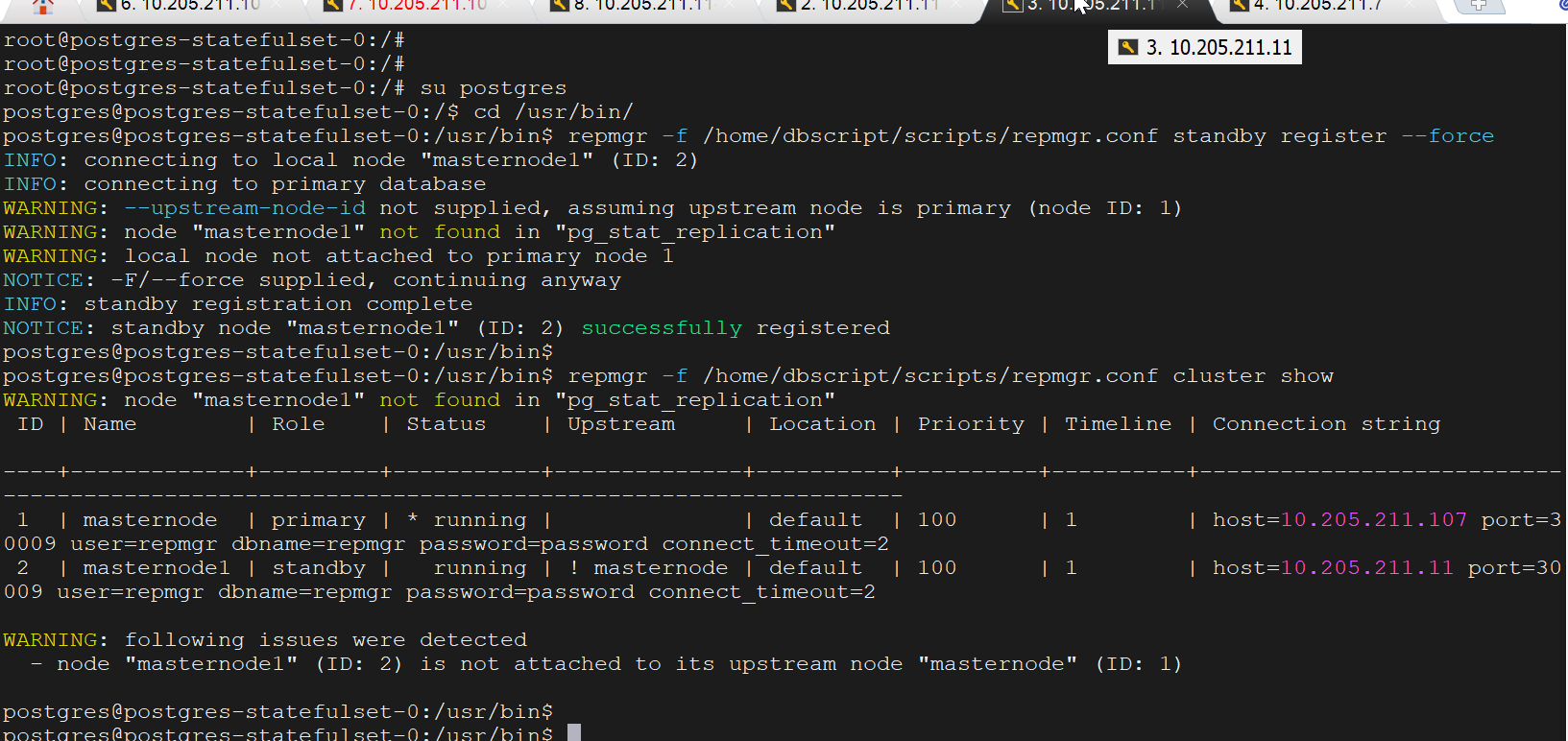
**cd /usr/bin**

**repmgr -f /home/dbscript/scripts/repmgr.conf standby register --force**

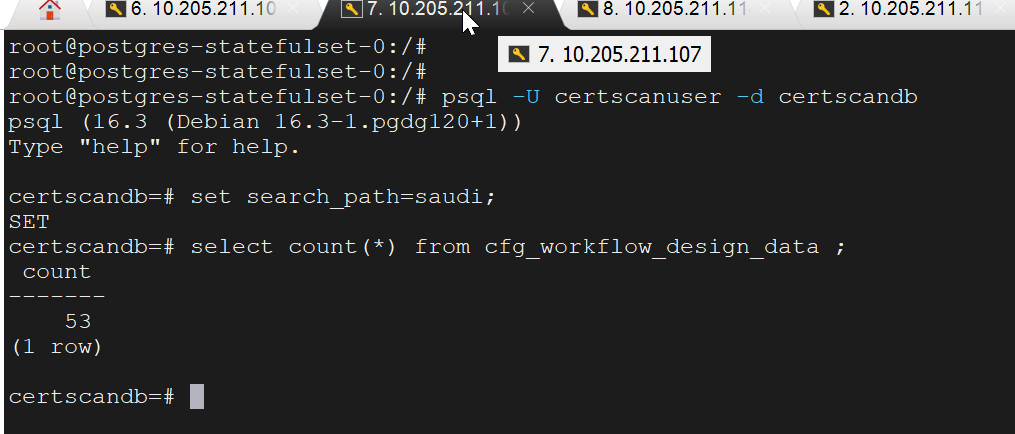


Check the available cluster nodes by executing the below command. It shows primary and standby server roles and status.

**repmgr -f /home/dbscript/scripts/repmgr.conf cluster show**



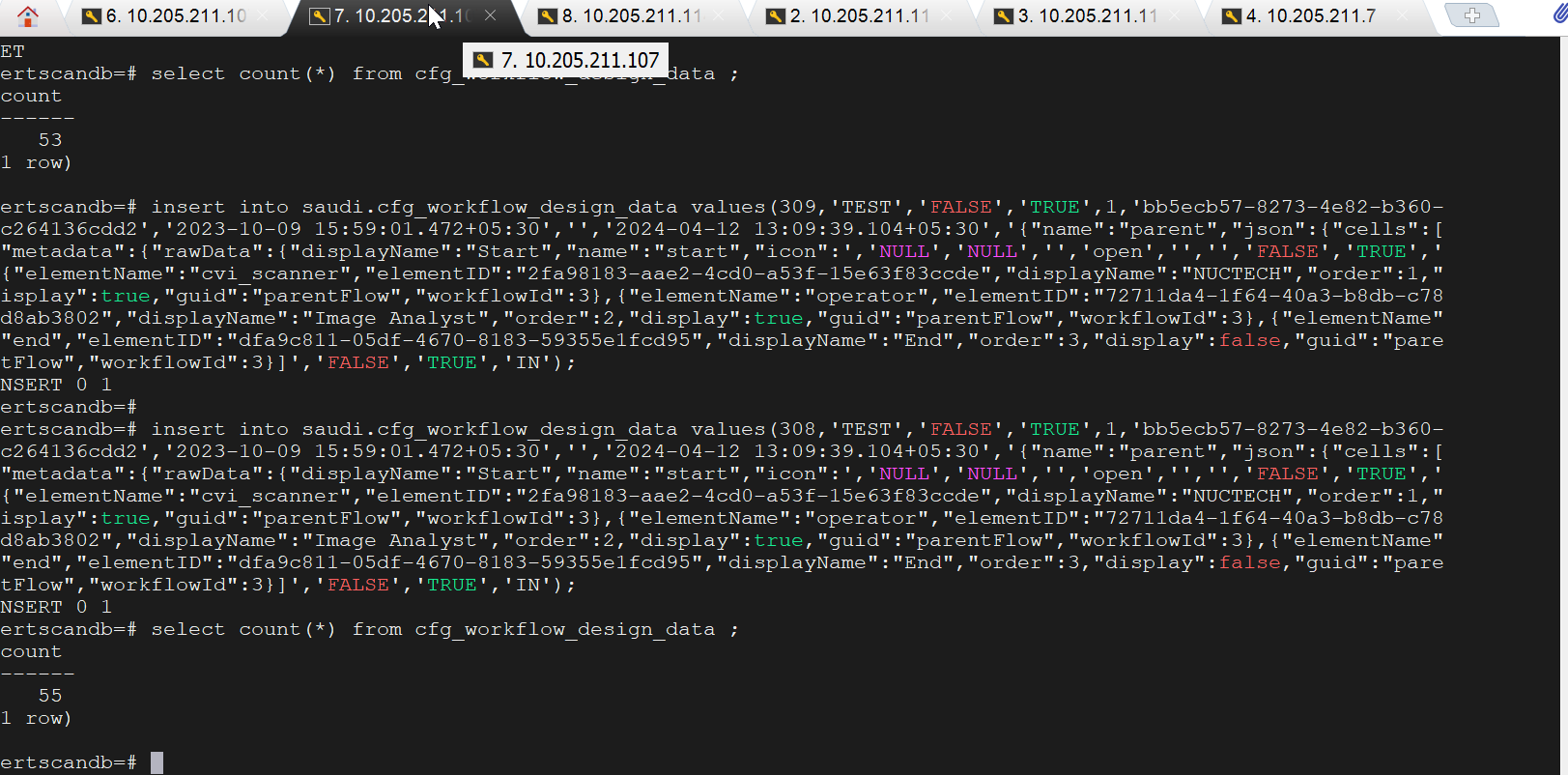
Add some records through primary server application.



A screenshot of a computer

Description automatically generated

Check the records added through primary server database is replicated to standby server database .



A screenshot of a computer

Description automatically generated

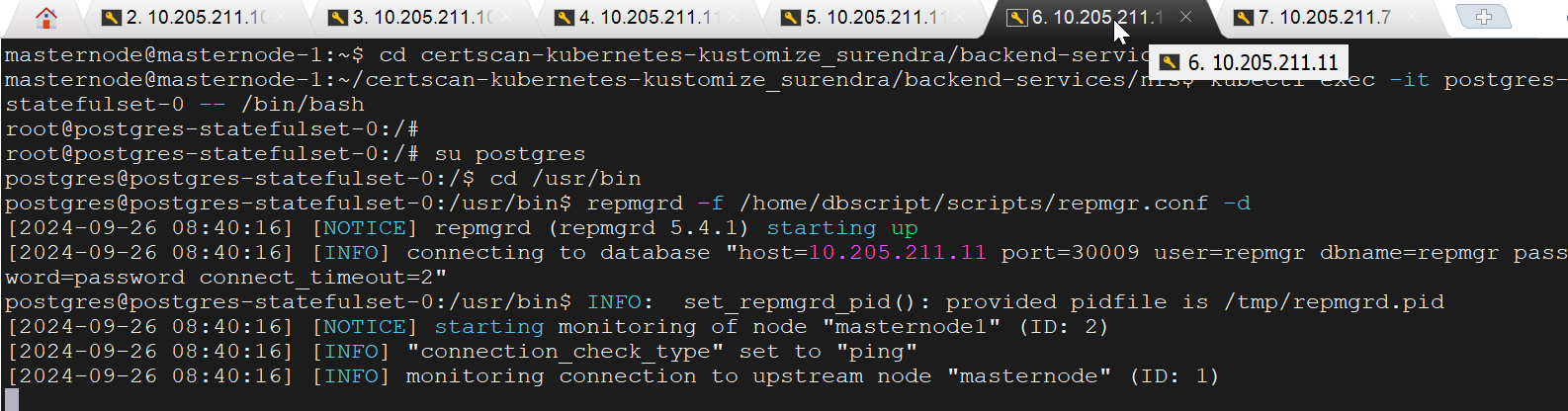
Enable repmgrd for monitoring and failover handling.

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**su postgres**

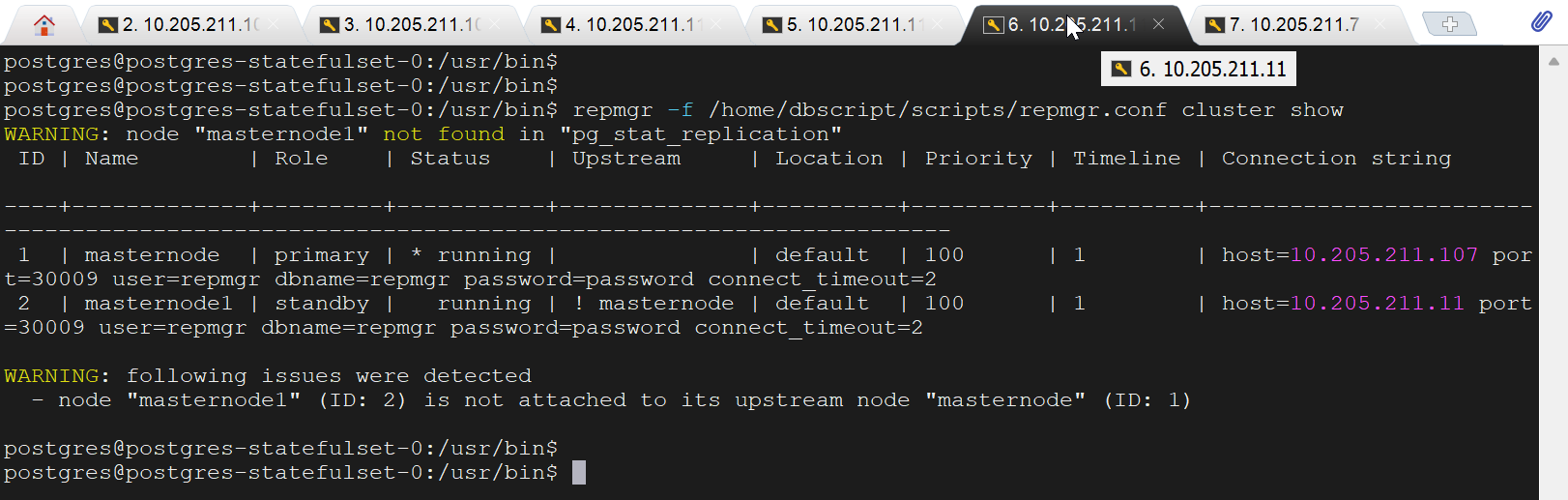
**cd /usr/bin**

**repmgrd -f /home/dbscript/scripts/repmgr.conf -d**



Check the current status of the cluster. It shows primary and standby server roles and status.

**repmgr -f /home/dbscript/scripts/repmgr.conf cluster show**



Stop the primary server.

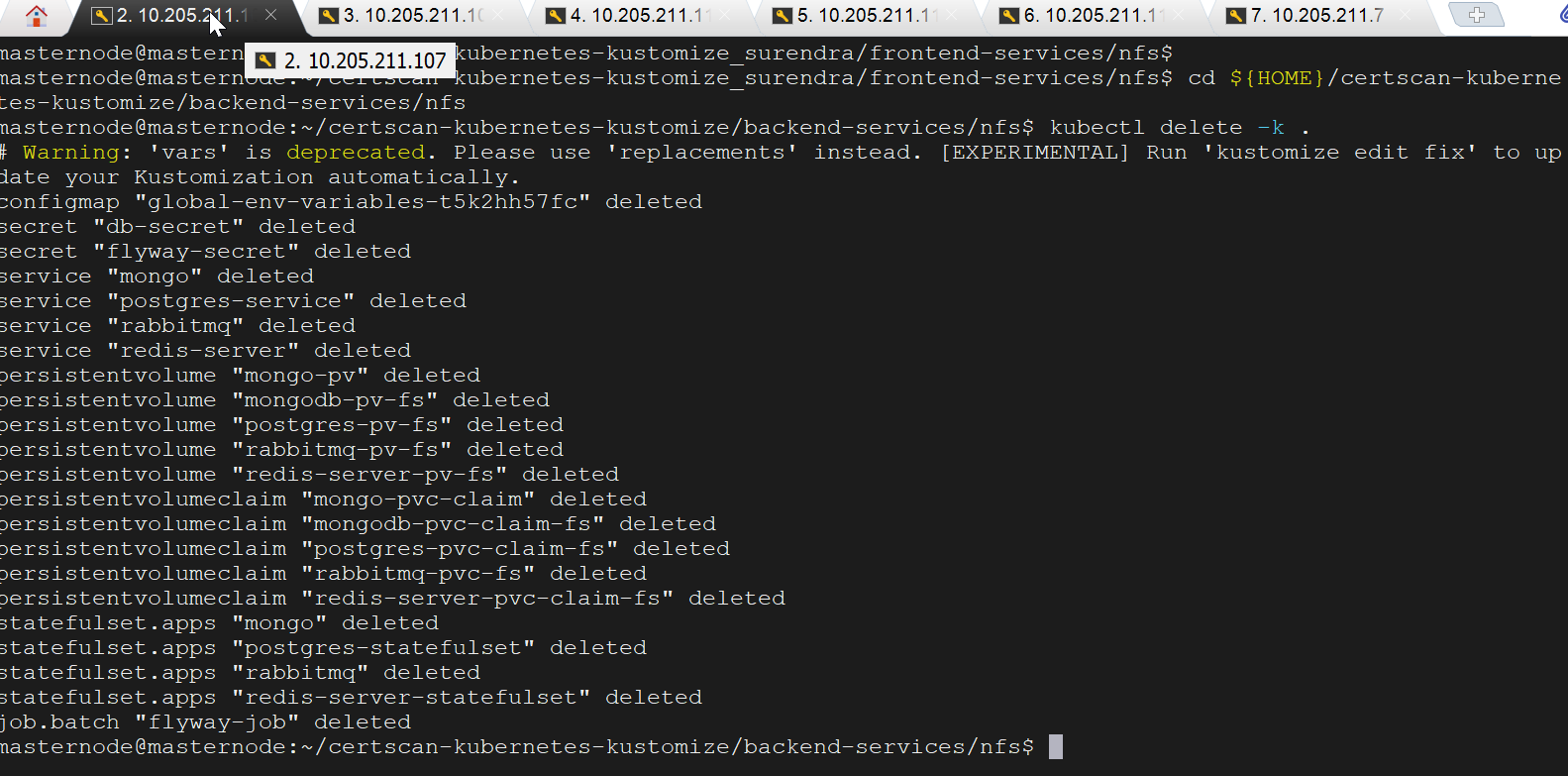
Execute the below command to stop primary server pods.

**cd ${HOME}/certscan-kubernetes-kustomize/backend-services/nfs**

**kubectl delete -k .**

**cd ${HOME}/certscan-kubernetes-kustomize/frontend-services/nfs**

**kubectl delete -k .**



Check the cluster status from standby server. Standby server role should be primary.

**repmgr -f /home/dbscript/scripts/repmgr.conf cluster show**

Roles before failover:

A computer screen shot of a black screen

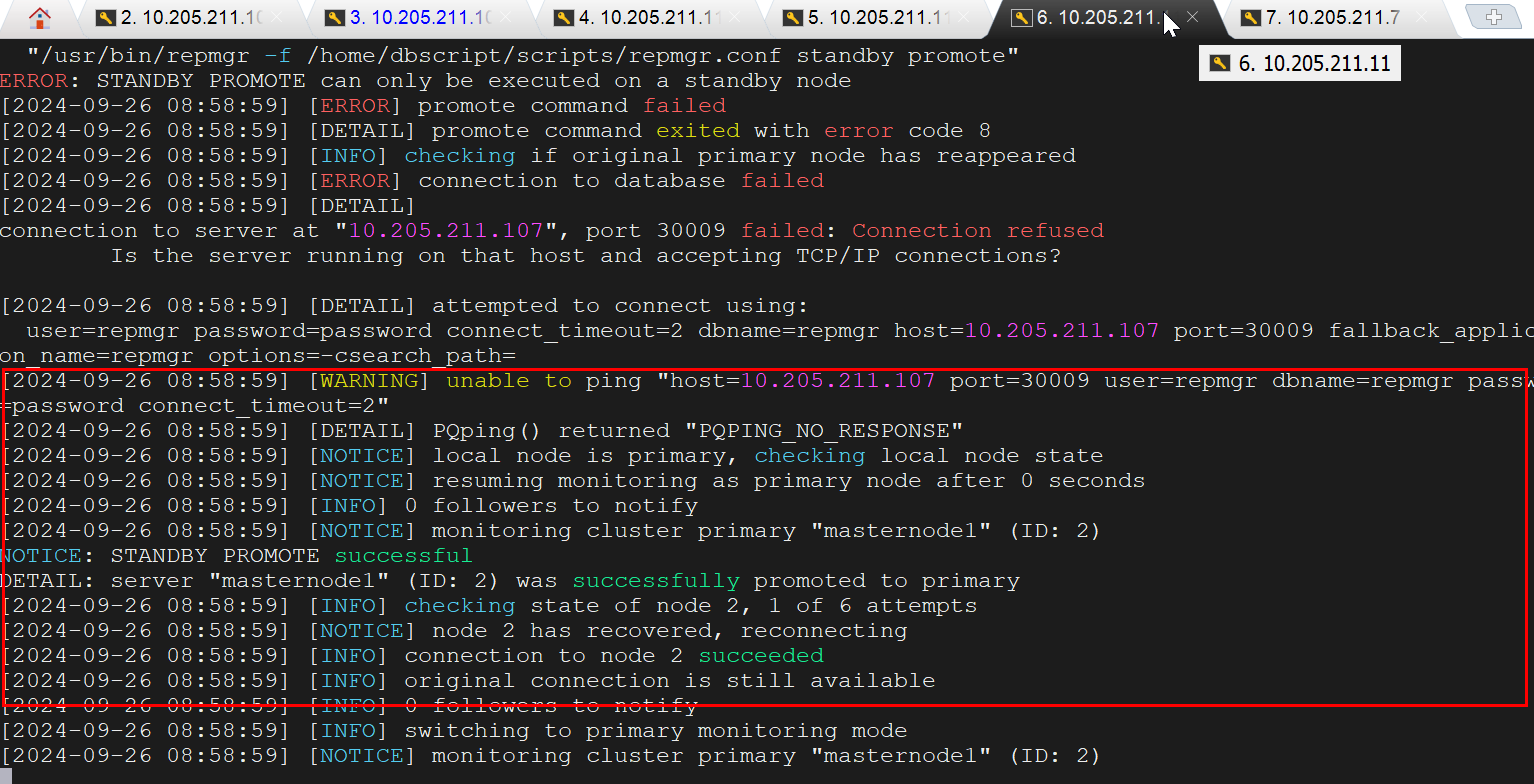
Description automatically generated

Roles after failover:

Automatically failover from primary server to standby server and standby server role has been changed from standby to primary after primary database failed.

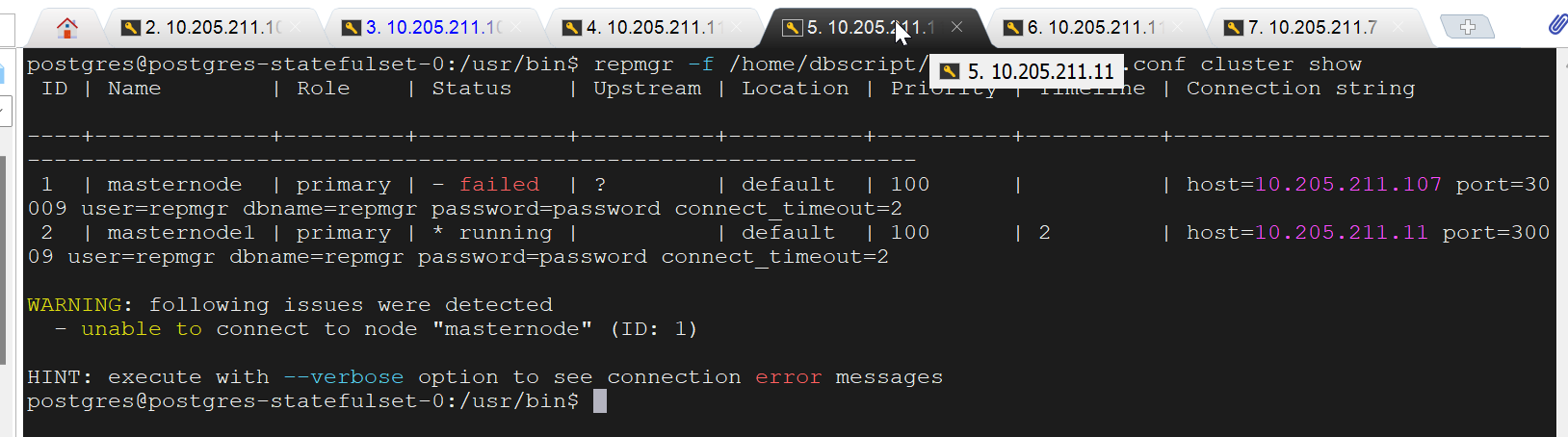
A screenshot of a computer

Description automatically generated



A computer screen shot of a black screen

Description automatically generated



Stop and start application pods from current primary server (10.205.211.11) to start application as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/frontend-services/nfs**

**kubectl delete -k .**

**A computer screen shot of text

Description automatically generated**

**cd ${HOME}/certscan-kubernetes-kustomize/frontend-services/nfs**

**kubectl apply -k .**

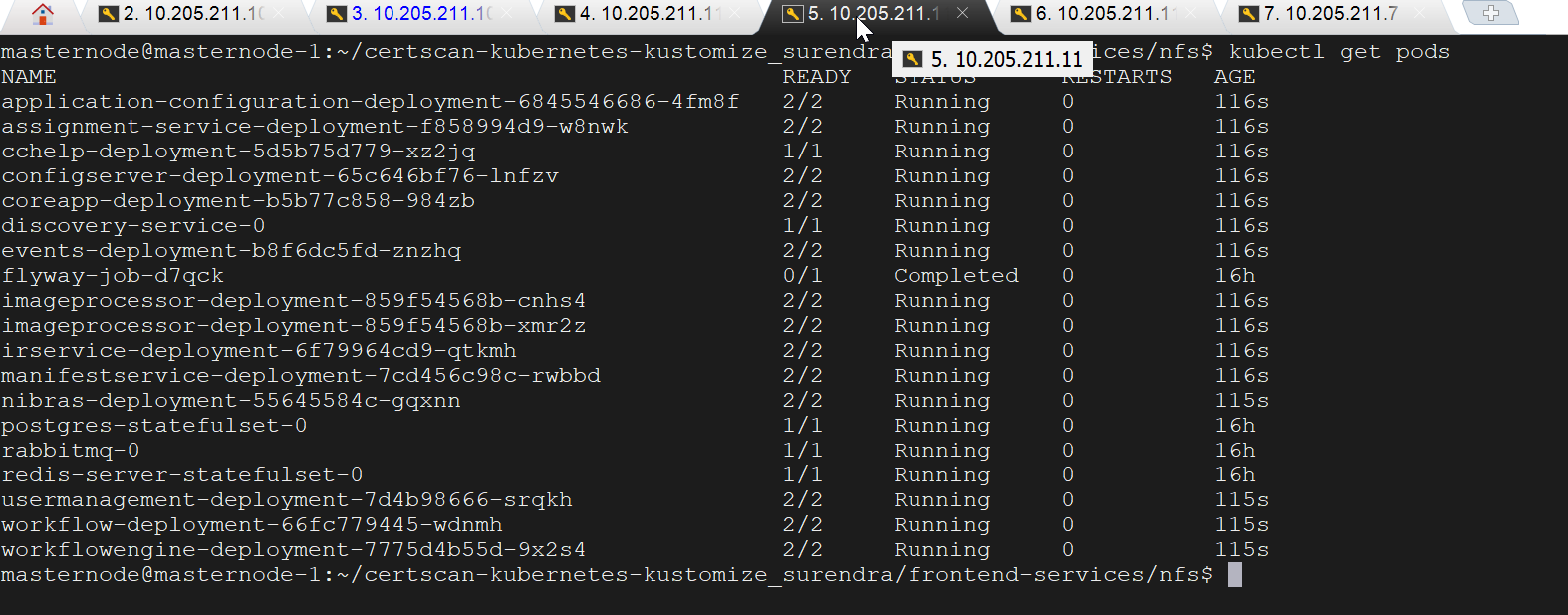
**A screen shot of a computer code

Description automatically generated**

Check the status of the pods, all pods should be running in status.

**cd ${HOME}/certscan-kubernetes-kustomize/frontend-services/nfs**

**kubectl get pogs**



Login to the application and check the data displaying as expected. Also perform some transactions and all transactions displayed.

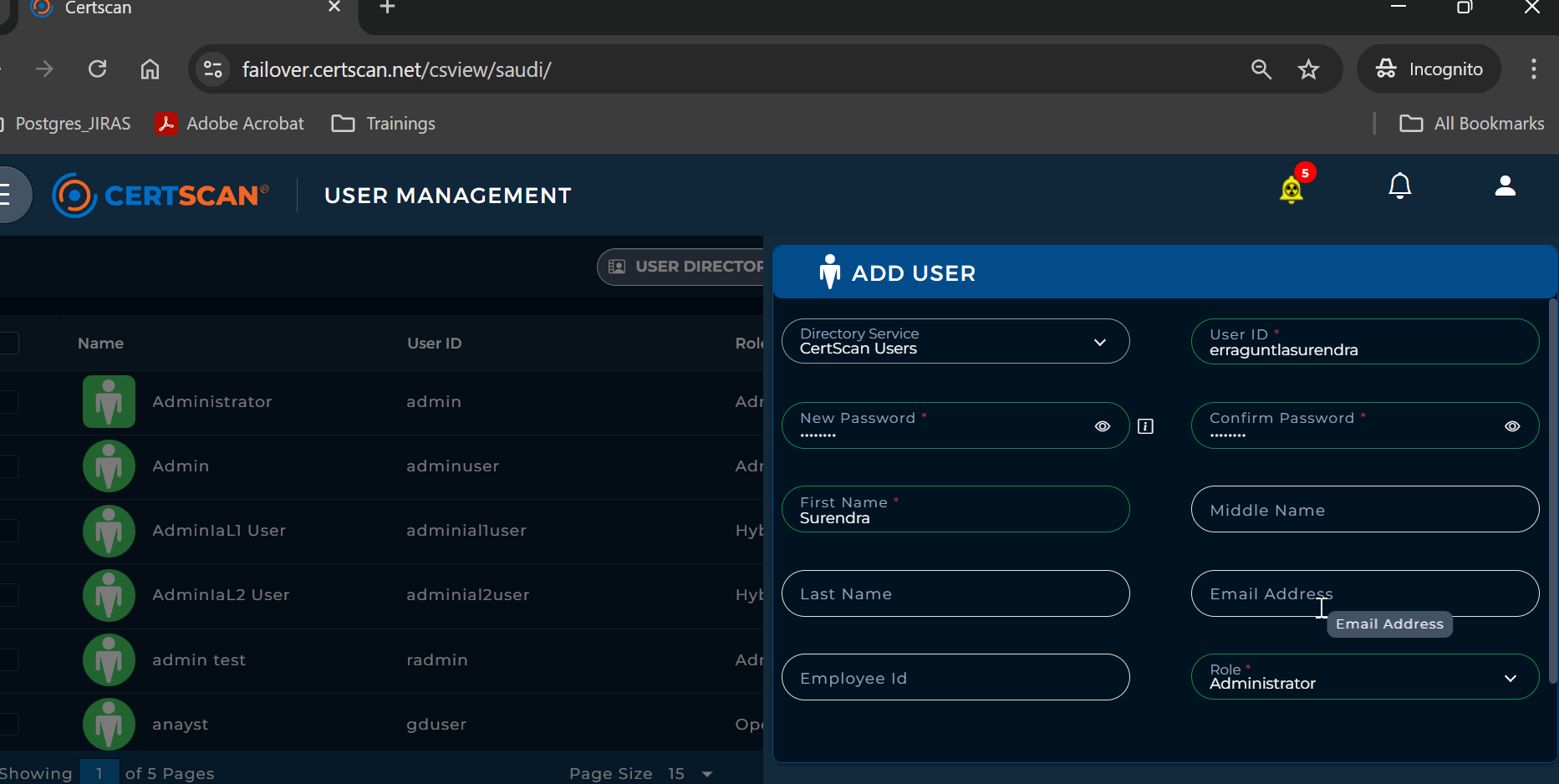
url: <https://failover.certscan.net/csview/saudi/>

username: superadmin

password: test@123

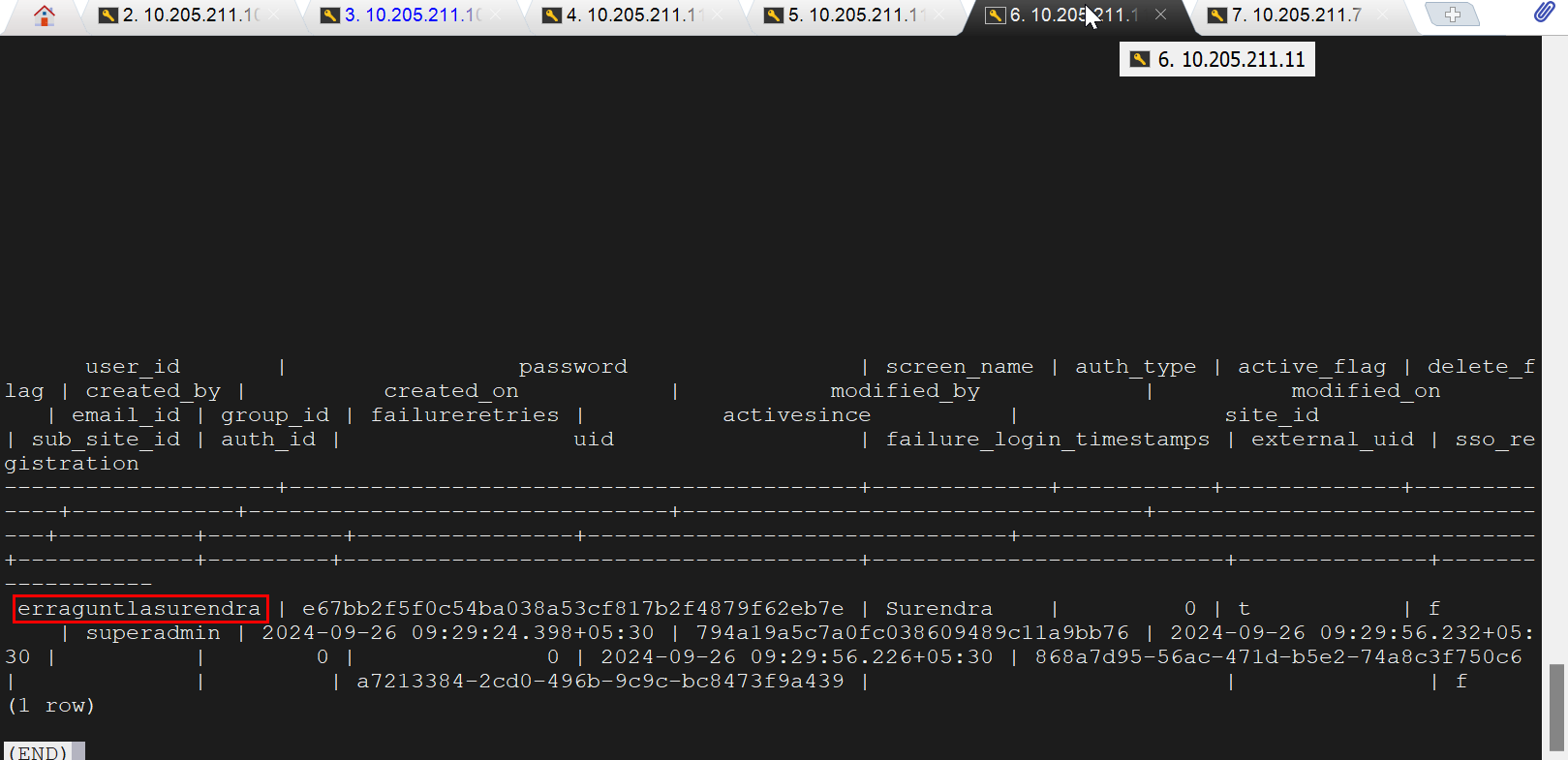
A screenshot of a computer

Description automatically generated

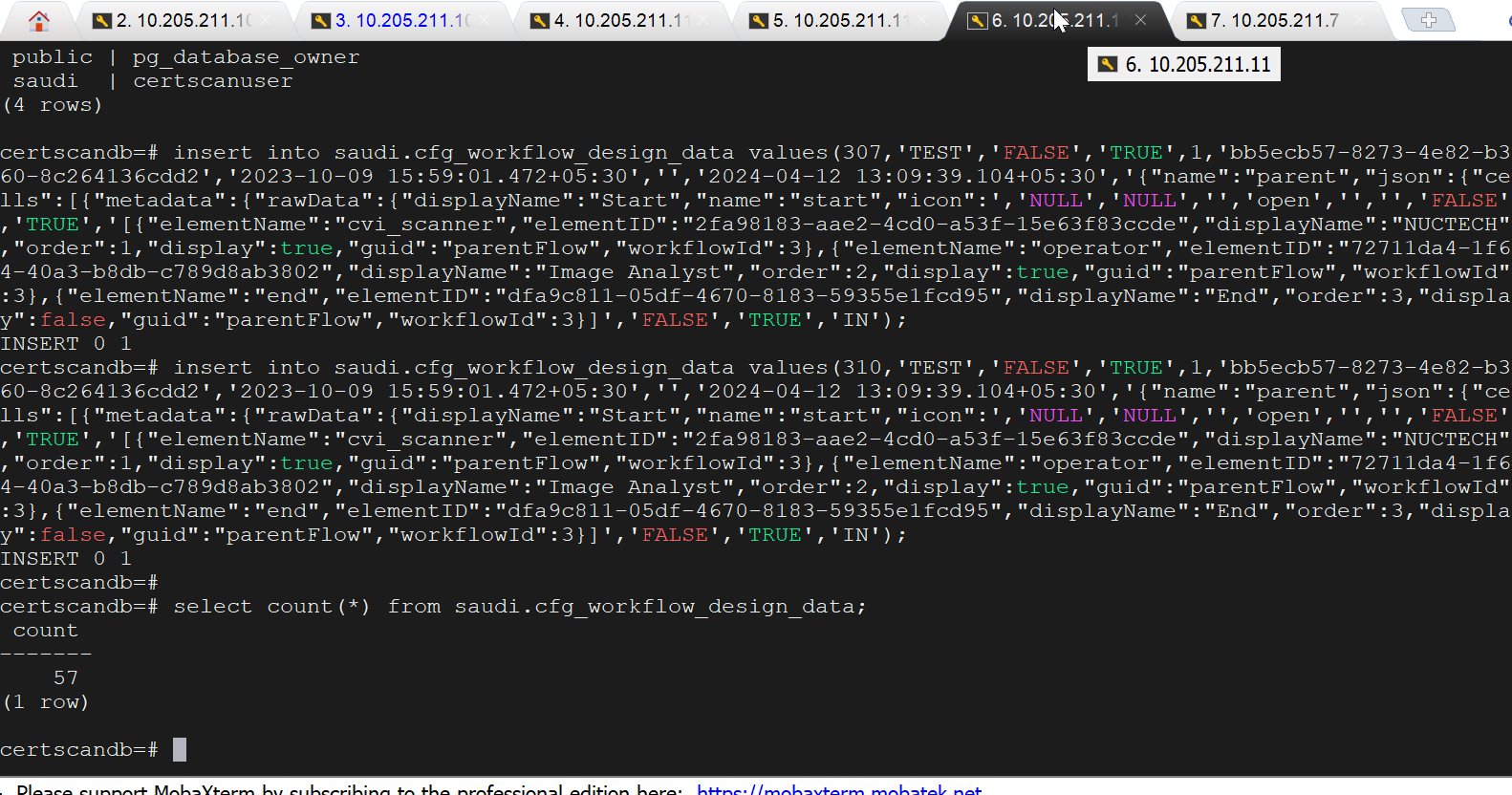


A screenshot of a computer

Description automatically generated









* 1. Failback Old Primary Server (10.205.211.107) as Standby

1. Login to primary database server (10.205.211.11) as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**psql -U certscanuser -d certscandb**

Run the below commands

**select pg\_create\_physical\_replication\_slot('replication\_slot\_slave1');**

**alter system set wal\_log\_hints = 'on';**

**alter system set restore\_command = 'cp /opt/wal\_archive/%f %p';**

**alter system set primary\_conninfo = 'host=10.205.211.11 port=30009 user=repuser password=password';**

**\q**

1. Stop and start the postgres database pod from primary server (10.205.211.11) as shown below to effect the changes made in step1.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend -services/nfs**

**kubectl delete statefulset -n default postgres-statefulset-0 --force**

**kubectl apply -k .**

1. Stop the postgres database pod from old primary server (10.205.211.107) as shown below.

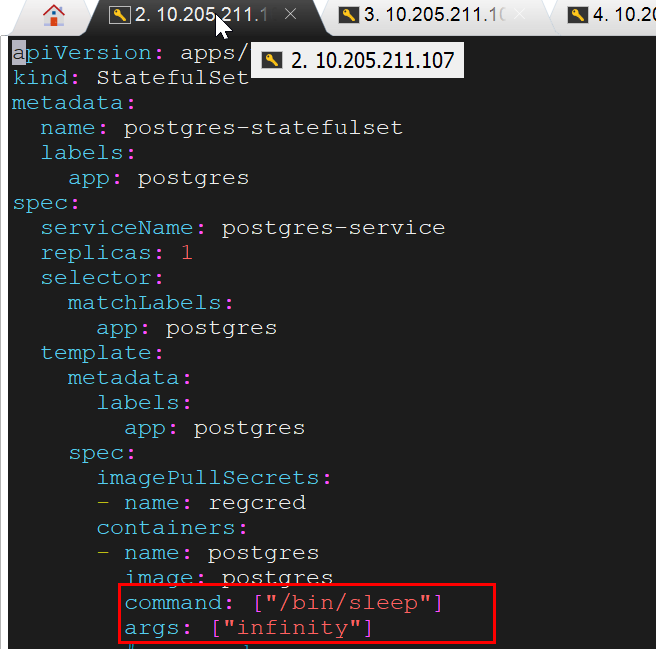
**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl delete statefulset -n default postgres-statefulset-0 --force**

1. Edit the ‘**postgres-statefulset.yaml’** from old primary server(10.205.211.107) as shown below to add sleep command for executing pg\_rewind tool for synching data from primary to standby as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/postgres/base**

**vi postgres-statefulset.yaml**

****

1. Start the postgres database pods from old primary server (10.205.211.107) as shown below.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl apply -k .**

1. Execute the ‘pg\_rewind’ tool from old primary server database as shown below to synch with primary server database.

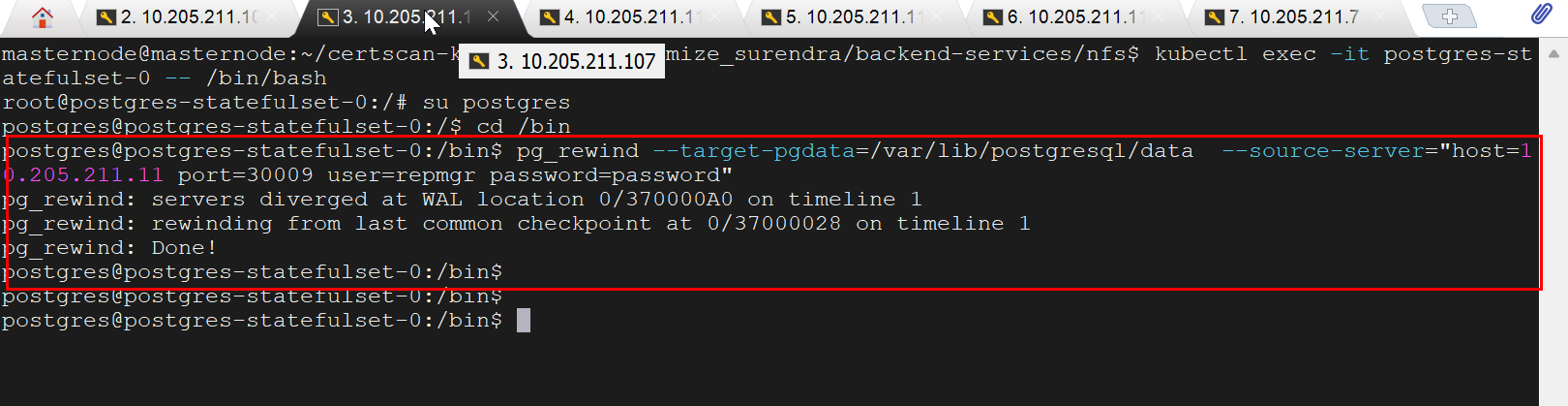
**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**su postgres**

**cd /bin**

**pg\_rewind --target-pgdata=/var/lib/postgresql/data --source-server="host=10.205.211.11 port=30009 user=repmgr password=password"**



1. Stop the postgres database pod from old primary server (10.205.211.107) as shown below

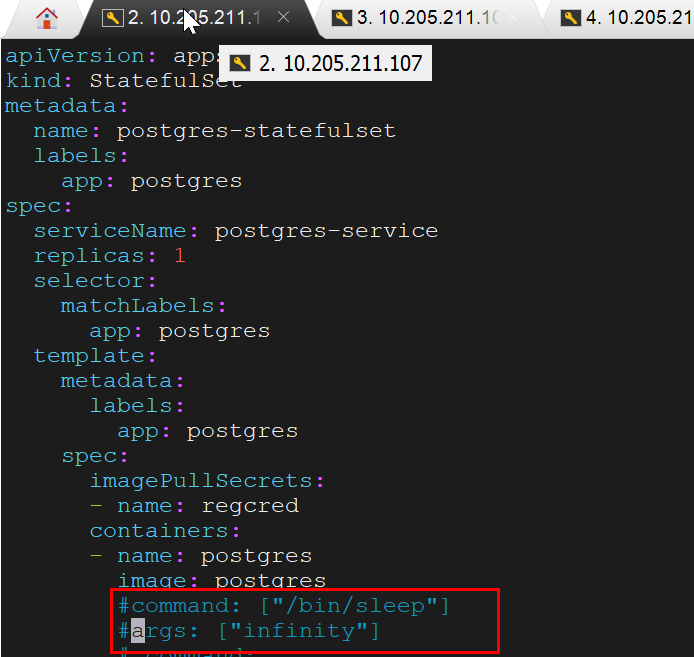
**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl delete statefulset -n default postgres-statefulset-0 --force**

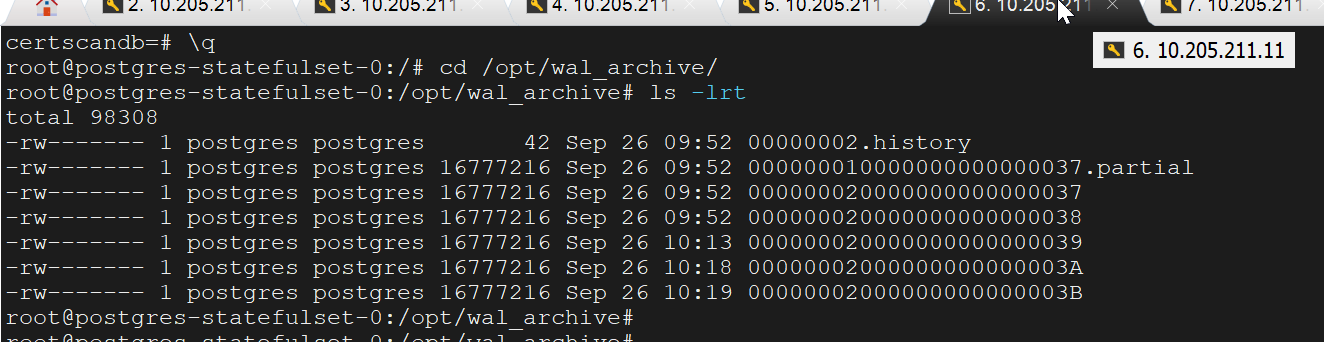
1. Edit the ‘**postgres-statefulset.yaml’** from old primary server(10.205.211.107) as shown below and comment **sleep command**.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/postgres/base**

**vi postgres-statefulset.yaml**



1. Copy the required WAL files from primary server (10.205.211.11) database /opt/wal\_archive directory to old primary server database /opt/wal\_archive directory as shown below.



1. Create ‘standby.signal’ file in the data directory of the old primary and change the owner for this file.

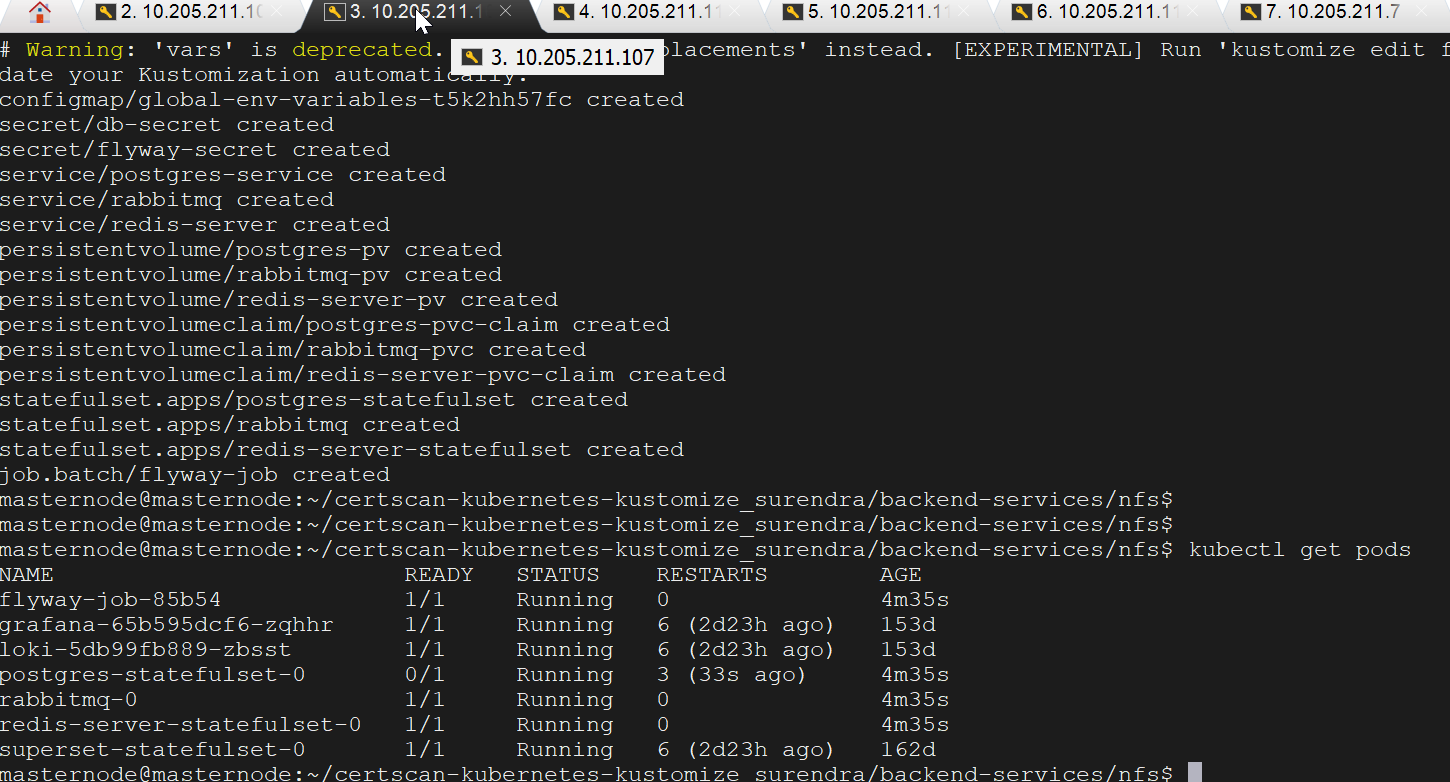
**touch standby.signal**

**chown systemd-coredump: systemd-coredump standby.signal**

1. Start the postgres database pods from old primary server (10.205.211.107) as shown below to effect the changes made in step1.

**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl apply -k .**

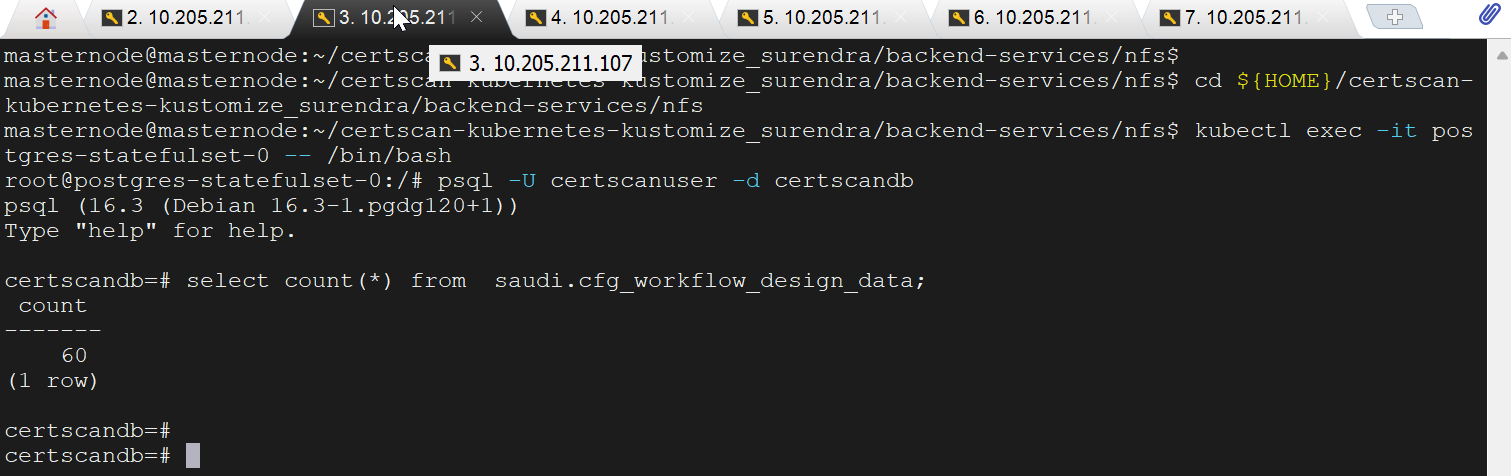
****

1. Login to old primary database server (10.205.211.107) as shown below and check the data is replicated as expected from primary database server.

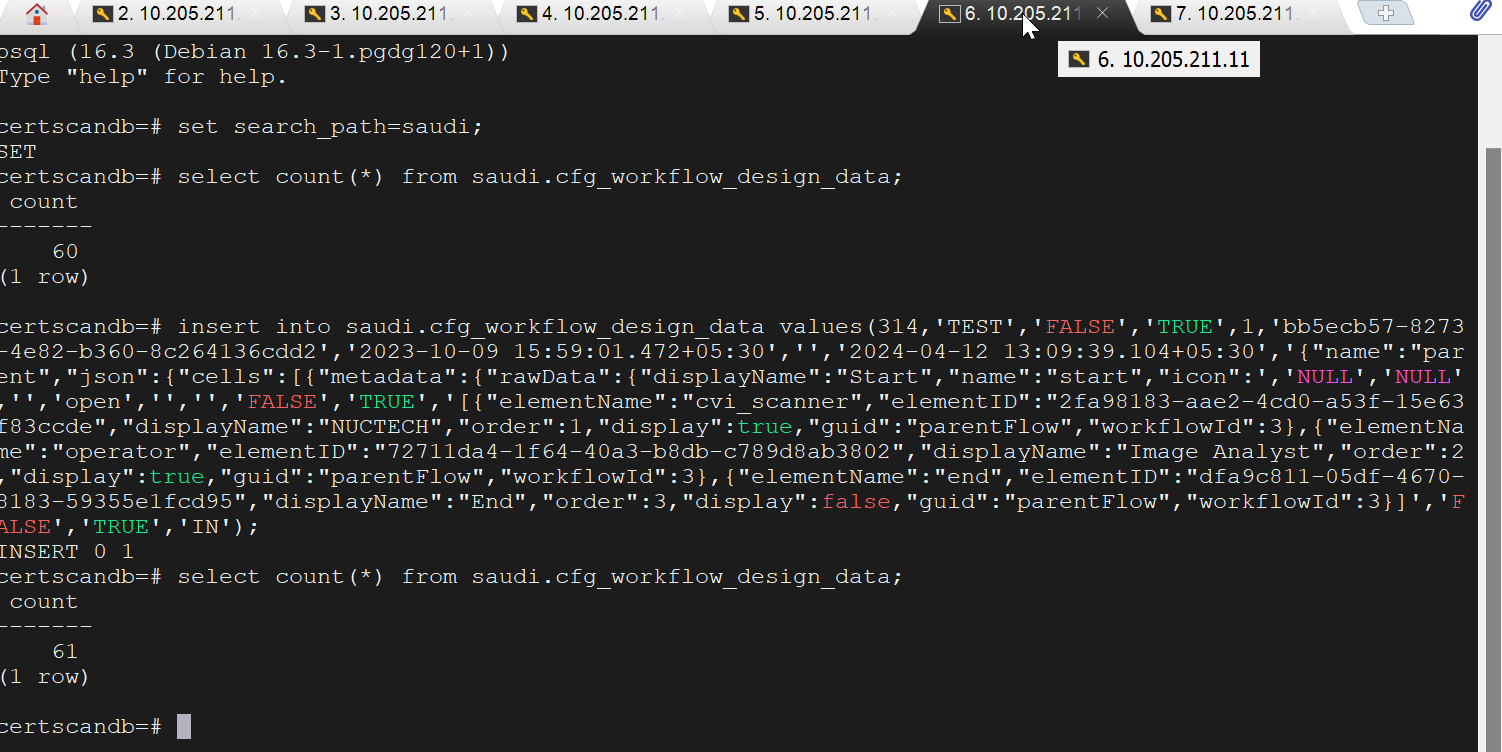
**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**psql -U certscanuser -d certscandb**

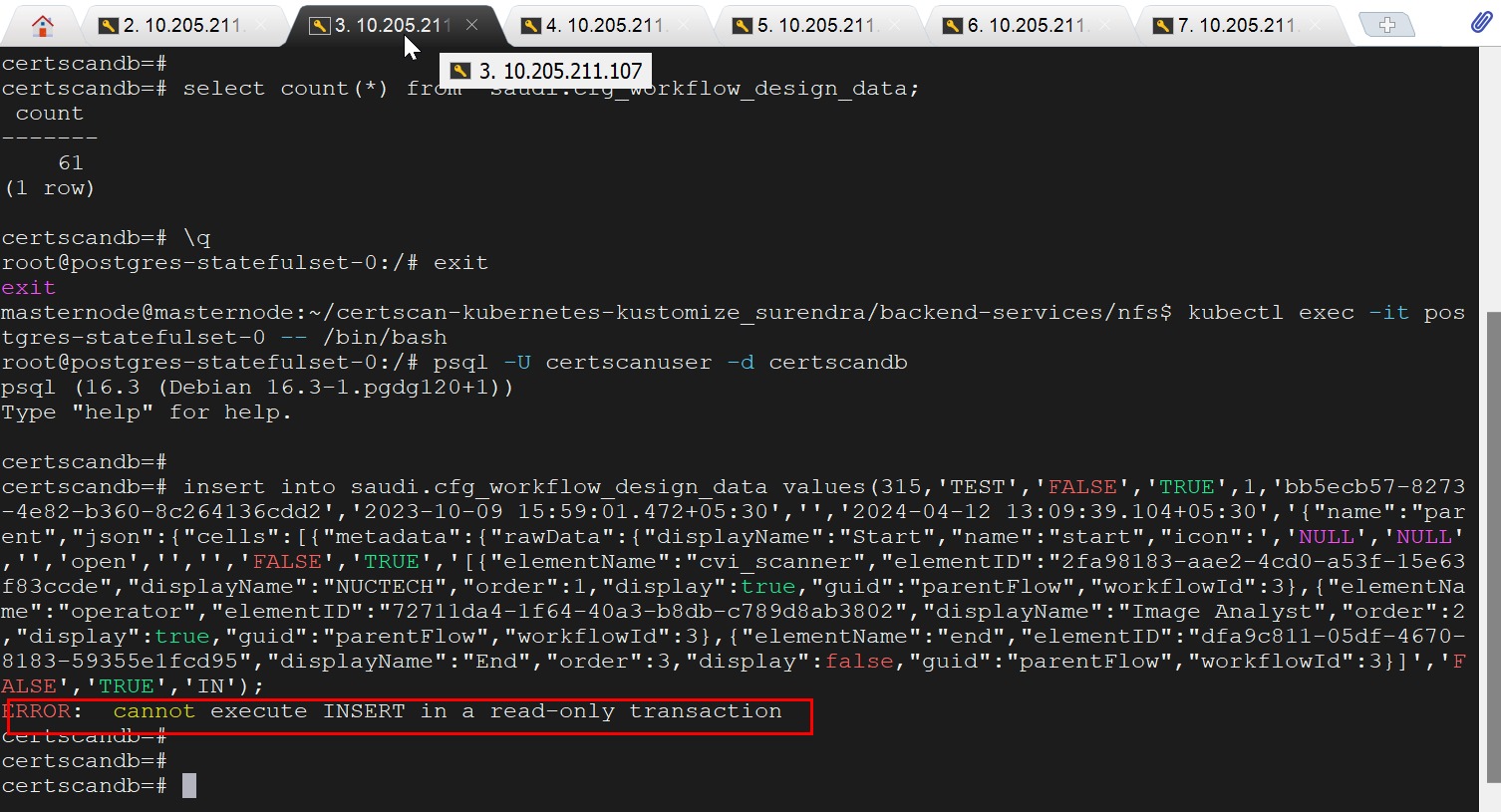
****

1. Add some records from primary server (10.205.211.11) and check the data is replicated as expected in old primary server (10.205.211.107) database.



A computer screen with text on it

Description automatically generated



Register the old primary server (10.205.211.107) as standby server.

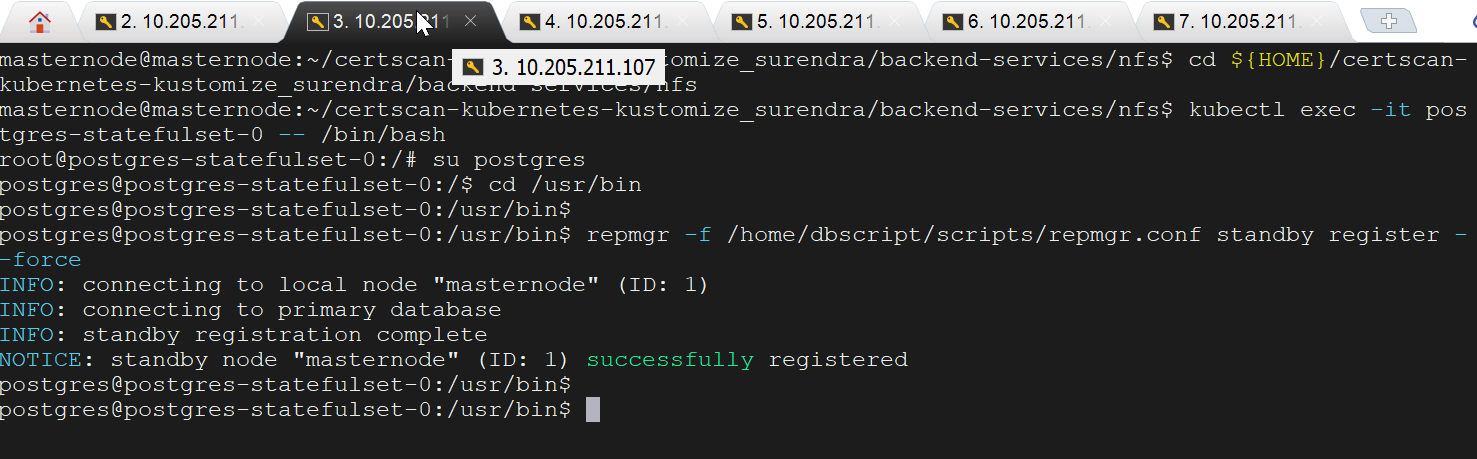
**cd ${HOME}/certscan-kubernetes-kustomize\_surendra/backend-services/nfs**

**kubectl exec -it postgres-statefulset-0 -- /bin/bash**

**su postgres**

**cd /usr/bin**

**repmgr -f /home/dbscript/scripts/repmgr.conf standby register -–force**

****

Check the available cluster nodes by executing the below command. It shows primary and standby server roles and status.

