
PostgreSQL Scripts Documentation

1. promote_PR.sh

Purpose: This script promotes a standby (DR) site to become the new primary (PR) site.

Description:

- Stops the PostgreSQL service on the current primary (if needed).
- Updates the PostgreSQL configuration to reflect the new primary site.

Dependencies:

- The script should be run with appropriate permissions.
- Environmental variables to be updated as per setup

Example:

bash

Copy code

Make sure the script is executable

chmod +x promote_PR.sh

Run the script

./promote_PR.sh

Script Content:

bash

#!/bin/bash

Configuration variables

PGDATA="/postgres/data/" # Path to PostgreSQL data directory

PGUSER="postgres" # PostgreSQL superuser

PGCTL="/usr/pgsql-16/bin/pg_ctl" # Path to pg_ctl command

PORT=6444

DBNAME=postgres

DBUSERNAME=postgres

Function to promote standby to primary

```
promote_standby() {
  echo "Promoting standby to primary..."
  $PGCTL promote -D "$PGDATA"

  if [ $? -eq 0 ]; then
    echo "Standby promotion successful."
  else
    echo "Standby promotion failed."
    exit 1
  fi
}

# Check if the server is currently in standby mode
is_standby() {
  local status
  status=$($psql -p 6444 -t -c "SELECT pg_is_in_recovery();" | tr -d '[:space:]')
 if [ "$status" = "t" ]; then
  echo "The server is in recovery mode."
else
  echo "The server is not in recovery mode."
fi
}

# Main script execution
echo "Checking if the server is in standby mode..."
if is_standby; then
  promote_standby
else
  echo "The server is not in standby mode. Promotion is not necessary."
  exit 0
fi
```

---

2. standby_DR.sh

Purpose: This script converts the old primary site to a standby site and updates its configuration to sync with the new primary.

Usage:

bash

Copy code

./standby_DR.sh

Description:

- Stops the PostgreSQL service on the old primary.
- Configures the old primary to act as a standby for the new primary.
- Updates postgresql.auto.conf with the new primary's details.
- Starts the PostgreSQL service on the old primary in standby mode.

Dependencies:

- Ensure correct paths and configurations.
- Environmental variables to be updated as per setup

Example:

```
#!/bin/bash

# Variables
STANDBY_DATA_DIR="/postgres/data/"
NEW_PRIMARY_HOST="10.2.0.22"
REPLICATION_USER="postgres"
PG_CTL="/usr/pgsql-16/bin/pg_ctl"  # Path to pg_ctl binary
PG_BIN_DIR="/usr/pgsql-16/bin/"  # Path to PostgreSQL binaries
LOG_FILE="/postgres/standby.log"  # Path to PostgreSQL log file
PORT=6444
PG_PORT=6444
DR_HOST="10.2.0.21"
```

```
PRIMARY_HOST="10.2.0.22"

# Function to check for errors
check_error() {
   if [ $? -ne 0 ]; then
      echo "Error: $1" >&2
      exit 1
   fi
}

# Stop PostgreSQL on the old standby (now the standby)
echo "Stopping PostgreSQL service on the standby server..."
$PG_CTL stop -D $STANDBY_DATA_DIR -s -m fast
check_error "Failed to stop PostgreSQL service on the standby server."

# Start PostgreSQL in single-user mode to update configuration
echo "Updating replication settings in postgresql.auto.conf..."
$PG_CTL start -D $STANDBY_DATA_DIR -l $LOG_FILE -w
check_error "Failed to start PostgreSQL service in single-user mode."

# Apply replication settings
echo "Applying replication settings using ALTER SYSTEM..."
PGPASSWORD=postgres@12345 $PG_BIN_DIR/psql -U $REPLICATION_USER  -p ${PORT} -d postgres -c "ALTER SYSTEM SET primary_conninfo TO 'host=${NEW_PRIMARY_HOST} port=${PG_PORT} user=${REPLICATION_USER} password=postgres@123';"

check_error "Failed to update primary_conninfo."

# Reload configuration
echo "Reloading PostgreSQL configuration..."
PGPASSWORD=postgres@12345 $PG_BIN_DIR/psql -U $REPLICATION_USER -p ${PORT} -d postgres -c "SELECT pg_reload_conf();"
```

check_error "Failed to reload PostgreSQL configuration."

touch $STANDBY_DATA_DIR/standby.signal

echo "DR changes complete now execute promte script on new PR"

---

3. replication_verify.sh

Purpose: This script checks the replication status on both the primary and standby sites.

Usage:

bash

Copy code

./replication_verify.sh

Description:

- Connects to the primary server and retrieves replication status.
- Connects to the standby server and retrieves the WAL receiver status.

Dependencies:

- Ensure correct paths and configurations.
- Environmental variables to be updated as per setup

Example:

bash

Copy code

Make sure the script is executable

chmod +x replication_verify.sh

Run the script

./replication_verify.sh

Script Content:

Define variables

PRIMARY_HOST="10.2.0.22"

PORT="6444"

REPLICA_USER="postgres"

REPLICA_HOST="10.2.0.21"

OLD_PRIMARY_HOST="10.2.0.22"

OLD_PORT="6444"

NEW_PRIMARY_HOST="10.2.0.21"

REPLICA_USER="postgres"

PRIMARY_DATA_DIR="postgres/data"

REPLICA_DATA_DIR="/postgres/data"

Connect to the primary server and check replication status

echo "Checking the replication on PR..."

psql -h "$PRIMARY_HOST" -p "$PORT" -U "$REPLICA_USER" -c \ "SELECT pid, application_name, client_addr, state, sync_state, sent_lsn, write_lsn, flush_lsn, replay_lsn FROM pg_stat_replication pg_stat_replication;"

Connect to the DR server and check replication status

echo "Checking the replication on DR..."

psql -h "$NEW_PRIMARY_HOST" -p "$PORT" -U "$REPLICA_USER" -c \ "SELECT pid,status,receive_start_lsn,written_lsn,last_msg_send_time,last_msg_receipt_time,latest_end_lsn,latest_end_time FROM pg_stat_wal_receiver;"

---

4. findprimary.sh

Purpose: This script identifies the current primary site in a PostgreSQL replication setup.

Usage:

bash

Copy code

./findprimary.sh

Description:

- Queries the replication status to determine which site is currently acting as the primary.
- If not in recovery state creates standby. Signal file.

Dependencies:

- The script should be run with appropriate permissions.
- Environmental variables to be updated as per setup

Example:

bash

Copy code

```
# Make sure the script is executable
chmod +x findprimary.sh

# Run the script
./findprimary.sh
```

Script Content:

```
# Configuration variables
PGDATA="/postgres/data/" # Path to PostgreSQL data directory
PGUSER="postgres" # PostgreSQL superuser
PGCTL="/usr/pgsql-16/bin/pg_ctl" # Path to pg_ctl command
POSTGREBIN="/usr/pgsql-16/bin/"
POSTGREDATAPATH="/postgres/data/"
PORT="6444"
DBUSERNAME="postgres"
DBNAME="postgres"
echo "Verify the current active server ip"
RECOVERY_STATUS=$(${POSTGREBIN}/psql -p ${PORT} -d ${DBNAME} -U ${DBUSERNAME} -w -c "select pg_is_in_recovery();"  -q)
value=$(echo $RECOVERY_STATUS | cut -d " " -f3);
if [ $value == "f" ]
then
    echo "Server is in Started State with master mode cannot able execute Standby.signal"
else

touch ${POSTGREDATAPATH}/standby.signal

fi

cat ${POSTGREDATAPATH}/postgresql.auto.conf | grep "^primary_conninfo" | tr -d "'" | cut -d"=" -f3- | cut -d" " -f4
```