

# PostgreSQL Automated backup on CentOS

PostgreSQL provides **pg\_dump** to backup your database in a single file easily. You can backup your database based on your schedule combined with cron job on your CentOS.

Following scripts will be beneficial for you

## pg\_backup.config

```
#####
## POSTGRESQL BACKUP CONFIG ##
#####

# Optional system user to run backups as. If the user the script is running as doesn't match this
# the script terminates. Leave blank to skip check.
BACKUP_USER=

# Optional hostname to adhere to pg_hba policies. Will default to "localhost" if none specified.
HOSTNAME=

# Optional username to connect to database as. Will default to "postgres" if none specified.
USERNAME=

# This dir will be created if it doesn't exist. This must be writable by the user the script is
# running as.
BACKUP_DIR=/home/backups/database/postgresql/

# List of strings to match against in database name, separated by space or comma, for which we only
# wish to keep a backup of the schema, not the data. Any database names which contain any of these
# values will be considered candidates. (e.g. "system_log" will match "dev_system_log_2010-01")
SCHEMA_ONLY_LIST=""

# Will produce a custom-format backup if set to "yes"
ENABLE_CUSTOM_BACKUPS=yes

# Will produce a gzipped plain-format backup if set to "yes"
ENABLE_PLAIN_BACKUPS=yes

# Will produce gzipped sql file containing the cluster globals, like users and passwords, if set to "yes"
ENABLE_GLOBALS_BACKUPS=yes

##### SETTINGS FOR ROTATED BACKUPS #####
# Which day to take the weekly backup from (1-7 = Monday-Sunday)
DAY_OF_WEEK_TO_KEEP=5

# Number of days to keep daily backups
DAYS_TO_KEEP=7

# How many weeks to keep weekly backups
WEEKS_TO_KEEP=5

#####
```

## pg\_backup.sh

```
#!/bin/bash

#####
## LOAD CONFIG #####
#####

while [ $# -gt 0 ]; do
    case $1 in
        -c)
```

```

        if [ -r "$2" ]; then
            source "$2"
            shift 2
        else
            ${ECHO} "Unreadable config file \"\$2\" 1>&2
            exit 1
        fi
        ;;
    *)
        ${ECHO} "Unknown Option \"\$1\" 1>&2
        exit 2
        ;;
    esac
done

if [ $# = 0 ]; then
    SCRIPTPATH=$(cd ${0%/*} && pwd -P)
    source $SCRIPTPATH/pg_backup.config
fi;

#####
#### PRE-BACKUP CHECKS #####
#####

# Make sure we're running as the required backup user
if [ "$BACKUP_USER" != "" -a "$(id -un)" != "$BACKUP_USER" ]; then
    echo "This script must be run as $BACKUP_USER. Exiting." 1>&2
    exit 1;
fi;

#####
### INITIALISE DEFAULTS ###
#####

if [ ! $HOSTNAME ]; then
    HOSTNAME="localhost"
fi;

if [ ! $USERNAME ]; then
    USERNAME="postgres"
fi;

#####
#### START THE BACKUPS #####
#####

FINAL_BACKUP_DIR=$BACKUP_DIR`date +\%Y-\%m-\%d`/
echo "Making backup directory in $FINAL_BACKUP_DIR"

if ! mkdir -p $FINAL_BACKUP_DIR; then
    echo "Cannot create backup directory in $FINAL_BACKUP_DIR. Go and fix it!" 1>&2
    exit 1;
fi;

#####
### GLOBALS BACKUPS ###
#####

echo -e "\n\nPerforming globals backup"
echo -e "-----\n"

if [ $ENABLE_GLOBALS_BACKUPS = "yes" ]
then
    echo "Globals backup"

    if ! pg_dumpall -g -h "$HOSTNAME" -U "$USERNAME" | gzip > $FINAL_BACKUP_DIR"globals".sql.gz.

```

```

in_progress; then
    echo "[!!ERROR!!] Failed to produce globals backup" 1>&2
else
    mv $FINAL_BACKUP_DIR"globals".sql.gz.in_progress $FINAL_BACKUP_DIR"globals".sql.gz
fi
else
    echo "None"
fi

#####
### SCHEMA-ONLY BACKUPS ###
#####

for SCHEMA_ONLY_DB in ${SCHEMA_ONLY_LIST//,/ }
do
    SCHEMA_ONLY_CLAUSE="$SCHEMA_ONLY_CLAUSE or datname ~ '$SCHEMA_ONLY_DB'"
done

SCHEMA_ONLY_QUERY="select datname from pg_database where false $SCHEMA_ONLY_CLAUSE order by datname;"` 

echo -e "\n\nPerforming schema-only backups"
echo -e "-----\n"

SCHEMA_ONLY_DB_LIST=`psql -h "$HOSTNAME" -U "$USERNAME" -At -c "$SCHEMA_ONLY_QUERY" postgres` 

echo -e "The following databases were matched for schema-only backup:\n${SCHEMA_ONLY_DB_LIST}\n"

for DATABASE in $SCHEMA_ONLY_DB_LIST
do
    echo "Schema-only backup of $DATABASE"

    if ! pg_dump -Fp -s -h "$HOSTNAME" -U "$USERNAME" "$DATABASE" | gzip > $FINAL_BACKUP_DIR"$DATABASE"_SCHEMA.sql.gz.in_progress; then
        echo "[!!ERROR!!] Failed to backup database schema of $DATABASE" 1>&2
    else
        mv $FINAL_BACKUP_DIR"$DATABASE"_SCHEMA.sql.gz.in_progress $FINAL_BACKUP_DIR"$DATABASE"_SCHEMA.sql.gz
    fi
done

#####
### FULL BACKUPS ###
#####

for SCHEMA_ONLY_DB in ${SCHEMA_ONLY_LIST//,/ }
do
    EXCLUDE_SCHEMA_ONLY_CLAUSE="$EXCLUDE_SCHEMA_ONLY_CLAUSE and datname !~ '$SCHEMA_ONLY_DB'"
done

FULL_BACKUP_QUERY="select datname from pg_database where not datistemplate and datallowconn
$EXCLUDE_SCHEMA_ONLY_CLAUSE order by datname;"` 

echo -e "\n\nPerforming full backups"
echo -e "-----\n"

for DATABASE in `psql -h "$HOSTNAME" -U "$USERNAME" -At -c "$FULL_BACKUP_QUERY" postgres` 
do
    if [ $ENABLE_PLAIN_BACKUPS = "yes" ]
    then
        echo "Plain backup of $DATABASE"

        if ! pg_dump -Fp -h "$HOSTNAME" -U "$USERNAME" "$DATABASE" | gzip >
$FINAL_BACKUP_DIR"$DATABASE".sql.gz.in_progress; then
            echo "[!!ERROR!!] Failed to produce plain backup database $DATABASE" 1>&2
        else
            mv $FINAL_BACKUP_DIR"$DATABASE".sql.gz.in_progress $FINAL_BACKUP_DIR"$DATABASE".sql.gz
        fi
    fi
done

```

```

if [ $ENABLE_CUSTOM_BACKUPS = "yes" ]
then
    echo "Custom backup of $DATABASE"

    if ! pg_dump -Fc -h "$HOSTNAME" -U "$USERNAME" "$DATABASE" -f $FINAL_BACKUP_DIR"$DATABASE".
custom.in_progress; then
        echo "[!!ERROR!!] Failed to produce custom backup database $DATABASE" 1>&2
    else
        mv $FINAL_BACKUP_DIR"$DATABASE".custom.in_progress $FINAL_BACKUP_DIR"$DATABASE".custom
    fi
fi

done

echo -e "\nAll database backups complete!"

```

### **pg\_backup\_rotated.sh**

```

#!/bin/bash

#####
##### LOAD CONFIG #####
#####

while [ $# -gt 0 ]; do
    case $1 in
        -c)
            CONFIG_FILE_PATH="$2"
            shift 2
            ;;
        *)
            ${ECHO} "Unknown Option \"$1\" 1>&2
            exit 2
            ;;
    esac
done

if [ -z $CONFIG_FILE_PATH ] ; then
    SCRIPTPATH=$(cd ${0%/*} && pwd -P)
    CONFIG_FILE_PATH="${SCRIPTPATH}/pg_backup.config"
fi

if [ ! -r ${CONFIG_FILE_PATH} ] ; then
    echo "Could not load config file from ${CONFIG_FILE_PATH}" 1>&2
    exit 1
fi

source "${CONFIG_FILE_PATH}"

#####
### PRE-BACKUP CHCKS #####
#####

# Make sure we're running as the required backup user
if [ "$BACKUP_USER" != "" -a "$(id -un)" != "$BACKUP_USER" ] ; then
    echo "This script must be run as $BACKUP_USER. Exiting." 1>&2
    exit 1
fi

#####
### INITIALISE DEFAULTS #####
#####

if [ ! $HOSTNAME ] ; then
    HOSTNAME="localhost"
fi;

```

```

if [ ! $USERNAME ]; then
    USERNAME="postgres"
fi;

#####
#### START THE BACKUPS #####
#####

function perform_backups()
{
    SUFFIX=$1
    FINAL_BACKUP_DIR=$BACKUP_DIR`date +\%Y-\%m-\%d`$SUFFIX/"

    echo "Making backup directory in $FINAL_BACKUP_DIR"

    if ! mkdir -p $FINAL_BACKUP_DIR; then
        echo "Cannot create backup directory in $FINAL_BACKUP_DIR. Go and fix it!" 1>&2
        exit 1;
    fi;

#####
### GLOBALS BACKUPS ###
#####

echo -e "\n\nPerforming globals backup"
echo -e "-----\n"

if [ $ENABLE_GLOBALS_BACKUPS = "yes" ]
then
    echo "Globals backup"

    if ! pg_dumpall -g -h "$HOSTNAME" -U "$USERNAME" | gzip > $FINAL_BACKUP_DIR"globals".sql.gz.in_progress; then
        echo "[!!ERROR!!] Failed to produce globals backup" 1>&2
    else
        mv $FINAL_BACKUP_DIR"globals".sql.gz.in_progress $FINAL_BACKUP_DIR"globals".sql.gz
    fi
else
    echo "None"
fi

#####
### SCHEMA-ONLY BACKUPS ###
#####

for SCHEMA_ONLY_DB in ${SCHEMA_ONLY_LIST//,/ }
do
    SCHEMA_ONLY_CLAUSE="$SCHEMA_ONLY_CLAUSE or datname ~ '$SCHEMA_ONLY_DB'"
done

SCHEMA_ONLY_QUERY="select datname from pg_database where false $SCHEMA_ONLY_CLAUSE order by datname;"

echo -e "\n\nPerforming schema-only backups"
echo -e "-----\n"

SCHEMA_ONLY_DB_LIST=`psql -h "$HOSTNAME" -U "$USERNAME" -At -c "$SCHEMA_ONLY_QUERY" postgres`

echo -e "The following databases were matched for schema-only backup:\n${SCHEMA_ONLY_DB_LIST}\n"

for DATABASE in $SCHEMA_ONLY_DB_LIST
do
    echo "Schema-only backup of $DATABASE"

    if ! pg_dump -Fp -s -h "$HOSTNAME" -U "$USERNAME" "$DATABASE" | gzip >
$FINAL_BACKUP_DIR"$DATABASE"_SCHEMA.sql.gz.in_progress; then
        echo "[!!ERROR!!] Failed to backup database schema of $DATABASE" 1>&2
    else
        mv $FINAL_BACKUP_DIR"$DATABASE"_SCHEMA.sql.gz.in_progress $FINAL_BACKUP_DIR"$DATABASE"
    fi
done
}

```

```

_SCHEMA.sql.gz
        fi
done

#####
##### FULL BACKUPS #####
#####

for SCHEMA_ONLY_DB in ${SCHEMA_ONLY_LIST//,/ }
do
    EXCLUDE_SCHEMA_ONLY_CLAUSE="$EXCLUDE_SCHEMA_ONLY_CLAUSE and datname !~ '$SCHEMA_ONLY_DB'"
done

FULL_BACKUP_QUERY="select datname from pg_database where not datistemplate and datallowconn
$EXCLUDE_SCHEMA_ONLY_CLAUSE order by datname;"

echo -e "\n\nPerforming full backups"
echo -e "-----\n"

for DATABASE in `psql -h "$HOSTNAME" -U "$USERNAME" -At -c "$FULL_BACKUP_QUERY" postgres` 
do
    if [ $ENABLE_PLAIN_BACKUPS = "yes" ]
    then
        echo "Plain backup of $DATABASE"

        if ! pg_dump -Fp -h "$HOSTNAME" -U "$USERNAME" "$DATABASE" | gzip >
$FINAL_BACKUP_DIR"$DATABASE".sql.gz.in_progress; then
            echo "[!!ERROR!!] Failed to produce plain backup database $DATABASE" 1>&2
        else
            mv $FINAL_BACKUP_DIR"$DATABASE".sql.gz.in_progress $FINAL_BACKUP_DIR"$DATABASE".
sql.gz
        fi
    fi

    if [ $ENABLE_CUSTOM_BACKUPS = "yes" ]
    then
        echo "Custom backup of $DATABASE"

        if ! pg_dump -Fc -h "$HOSTNAME" -U "$USERNAME" "$DATABASE" -f
$FINAL_BACKUP_DIR"$DATABASE".custom.in_progress; then
            echo "[!!ERROR!!] Failed to produce custom backup database $DATABASE"
        else
            mv $FINAL_BACKUP_DIR"$DATABASE".custom.in_progress $FINAL_BACKUP_DIR"$DATABASE".
custom
        fi
    fi

done

echo -e "\nAll database backups complete!"
}

# MONTHLY BACKUPS

DAY_OF_MONTH=`date +%d`

if [ $DAY_OF_MONTH -eq 1 ];
then
    # Delete all expired monthly directories
    find $BACKUP_DIR -maxdepth 1 -name "*-monthly" -exec rm -rf '{}' ';' 

    perform_backups "-monthly"

    exit 0;
fi

# WEEKLY BACKUPS

DAY_OF_WEEK=`date +%u` #1-7 (Monday-Sunday)
EXPIRED_DAYS=`expr $((($WEEKS_TO_KEEP * 7) + 1))`
```

```
if [ $DAY_OF_WEEK = $DAY_OF_WEEK_TO_KEEP ];
then
    # Delete all expired weekly directories
    find $BACKUP_DIR -maxdepth 1 -mtime +$EXPIRED_DAYS -name "*-weekly" -exec rm -rf '{}' +''
    perform_backups "-weekly"
    exit 0;
fi

# DAILY BACKUPS

# Delete daily backups 7 days old or more
find $BACKUP_DIR -maxdepth 1 -mtime +$DAYS_TO_KEEP -name "*-daily" -exec rm -rf '{}' +''
perform_backups "-daily"
```

For your information, above codes are copied at [https://wiki.postgresql.org/wiki/Automated\\_Backup\\_on\\_Linux](https://wiki.postgresql.org/wiki/Automated_Backup_on_Linux)