Университет ИТМО

Факультет программной инженерии и компьютерной техники

Распределённые системы хранения данных. Лабораторная работа №3.

Группа: P33131
Студент: Смирнов Виктор Игоревич
Преподаватель: Афанасьев Дмитрий Борисович
Вариант: 736

2024

Ключевые слова

База данных, конфигурация PostgreSQL.

Содержание

1 Цель работы и контекст

Цель работы - настроить процедуру периодического резервного копирования базы данных, сконфигурированной в ходе выполнения лабораторной работы №2, а также разработать и отладить сценарии восстановления в случае сбоев.

Узел из предыдущей лабораторной работы используется в качестве основного. Новый узел используется в качестве резервного. Учётные данные для подключения к новому узлу выдаёт преподаватель. В сценариях восстановления необходимо использовать копию данных, полученную на первом этапе данной лабораторной работы.

2 Этап 0. Контекст работы

В предыдущей лабораторной работы была создана база данных. Приведу здесь скрипты для ее инициализации.

2.1 Переменные окружения

```
#!/bin/sh

export DDB_PG_CONF="."
export DDB_PG_USER="postgres0"
export DDB_PG_PASS="pleasehelp"
export DDB_PG_PASS_FILE="$DDB_PG_CONF/pgpass.txt"
export DDB_PG_PORT=9666
export DDB_PG_DATABASE=postgres

export DDB_TABLESPACE_NAME=yqy90
export DDB_TABLESPACE_LOCATION="$HOME/$DDB_TABLESPACE_NAME"
export DDB_NEW_DATABASE_NAME=lazyorangehair
export DDB_NEW_USER=root
export DDB_NEW_USER_PASSWORD=rootik

export PGDATA="$HOME/kop67"

export DDB_INITDB=/usr/lib/postgresql/14/bin/initdb
export DDB_PGBIN=/usr/lib/postgresql/14/bin/postgres
export DDB_PGBASEBACKUP=/usr/lib/postgresql/14/bin/pg_basebackup

export DDB_PRIMARY_BACKUP_DIR="primary/backup"
export DDB_PRIMARY_BACKUP_BASE_DIR="$DDB_PRIMARY_BACKUP_DIR/base"

export DDB_STANDBY_HOST=ddb-standby
export DDB_STANDBY_BACKUP_BASE_DIR=$DDB_PRIMARY_BACKUP_BASE_DIR
export DDB_STANDBY_BACKUP_WAL_DIR="$DDB_PRIMARY_BACKUP_DIR/wal"
```

Листинг 1: Переменные окружения

2.2 Конфигурация базы данных

```
# $PGDATA/pg_hba.conf (Host-based authentication)

# TYPE  DATABASE    USER     ADDRESS        METHOD
host    all         all      127.0.0.1/32   scram-sha-256 # Permit only localhost
host    all         all      ::1/128        scram-sha-256 # Permit only localhost
host    replication all      localhost      scram-sha-256 # Permit base backup
```

Листинг 2: Конфигурационный файл pg_hba.conf

```
# $PGDATA/postgresql.conf (PostgreSQL configuration file)

# Note: Optimized for OLAP load:
# 5 users, packet r/w 128MB

## CONNECTIONS

listen_addresses  = '127.0.0.1' # Available only from localhost
port              = 9666        # For security
unix_socket_directories = ''    # Only TCP/IP

max_connections                 = 6 # 5 users + 1 extra
superuser_reserved_connections  = 3


## AUTHENTICATION

authentication_timeout  = 20s           # Type password faster
password_encryption     = scram-sha-256 # Strong password hashing

## RESOURCE USAGE

shared_buffers             = 1024MB  # 128MB * (5 + 3) users
temp_buffers               = 128MB   # 128MB
max_prepared_transactions  = 0       # We don't use transactions
work_mem                   = 256MB   # Expected packet size
hash_mem_multiplier        = 1.5     # Smaller hash tables
maintenance_work_mem       = 64MB    # ?
autovacuum_work_mem        = -1
max_stack_depth            = 4MB     # Be prepared for complex queries
```

```
31
32 temp_file_limit            = 4GB      # Something is wrong if we reach this
33
34 ## WRITE-AHEAD LOG
35
36 checkpoint_timeout     = 5min
37 fsync                  = off          # Lost data is not critical, as we can recreate
38 synchronous_commit     = off          # Same
39 wal_level              = replica      # Enable replication
40 wal_compression        = off          # WAL must not be so huge?
41 commit_delay           = 200          # Acceptable to lose 200mc of data
42 effective_cache_size   = 4GB          # OK?
43
44
45 ## REPORTING AND LOGGING
46
47 log_destination    = 'stderr'
48 logging_collector = off
49 log_directory      = 'log'
50 log_filename       = 'postgresql-%Y-%m-%d_%H%M%S.log'
51 log_min_messages   = warning
52
53 log_connections    = on
54 log_disconnections = on
55
56 ## Archiving
57
58 archive_mode    = on
59 archive_timeout = 16s
60 archive_command = 'ssh -q <STANDBY_HOST> "test ! -e <STANDBY_WAL_DIR>/%f" && scp %p <
       STANDBY_HOST>:~/<STANDBY_WAL_DIR>'
```

Листинг 3: Конфигурационный файл postgresql.conf

2.3 Создание .pgpass

```
1 #!/bin/sh
2
3 set -e
4
5 cd "$(dirname "$0")"
6
7 echo "" > ~/.pgpass
8 echo "localhost:$DDB_PG_PORT:*:$DDB_PG_USER:$DDB_PG_PASS" >> ~/.pgpass
9 echo "localhost:$DDB_PG_PORT:*:$DDB_NEW_USER:$DDB_NEW_USER_PASSWORD" >> ~/.pgpass
10 chmod 0600 ~/.pgpass
```

Листинг 4: Файл .pgpass

2.4 Инициализация базы данных

```
1 #!/bin/sh
2
3 set -e
4
5 cd "$(dirname "$0")"
6
7 mkdir "$PGDATA" 2> /dev/null
8
9 echo "$DDB_PG_PASS" > "$DDB_PG_PASS_FILE"
10
11 "$DDB_INITDB" \
12   --pgdata="$PGDATA" \
13   --locale="ru_RU.CP1251" \
14   --encoding="WIN1251" \
15   --pwfile="$DDB_PG_PASS_FILE"
16
17 cp "$DDB_PG_CONF/pg_hba.conf" "$PGDATA/pg_hba.conf"
18 cp "$DDB_PG_CONF/postgresql.conf" "$PGDATA/postgresql.conf"
```

Листинг 5: Инициализация базы данных

2.5 Запуск базы данных

```
#!/bin/sh

set -e

cd "$(dirname "$0")"

"$DDB_PGBIN" -D "$PGDATA"
```

Листинг 6: Запуск базы данных

2.6 Настройка базы данных

```
#!/bin/sh

set -e

cd "$(dirname "$0")"

sql() {
    psql -h localhost -p "$DDB_PG_PORT" -c "$1" "$DDB_PG_DATABASE"
}

mkdir "$DDB_TABLESPACE_LOCATION" 2>/dev/null

sql "CREATE TABLESPACE $DDB_TABLESPACE_NAME LOCATION '$DDB_TABLESPACE_LOCATION';"
sql "ALTER DATABASE template1 SET TABLESPACE $DDB_TABLESPACE_NAME;"
sql "CREATE DATABASE $DDB_NEW_DATABASE_NAME TEMPLATE template1;"
sql "CREATE ROLE tester;"
sql "CREATE USER $DDB_NEW_USER WITH LOGIN PASSWORD '$DDB_NEW_USER_PASSWORD';"
sql "GRANT tester TO $DDB_NEW_USER;"

sql() {
  psql -U "$DDB_NEW_USER" -h localhost -p $DDB_PG_PORT -c "$2" "$1"
}

PRV="$DDB_PG_DATABASE"
NEW="$DDB_NEW_DATABASE_NAME"

sql "$PRV" "CREATE TABLE note_prv (id serial PRIMARY KEY, content text NOT NULL);"
sql "$NEW" "CREATE TABLE note_new (id serial PRIMARY KEY, content text NOT NULL);"

sql "$PRV" "INSERT INTO note_prv (content) VALUES ('Note at postgres');"
sql "$NEW" "INSERT INTO note_new (content) VALUES ('Note at lazyorangehair');"

sql "$PRV" "SELECT * FROM note_prv;"
sql "$NEW" "SELECT * FROM note_new;"
```

Листинг 7: Настройка базы данных

3 Этап 1. Резервное копирование

3.1 Задача

1. Настроить резервное копирование с основного узла на резервный следующим образом:
   (a) Первоначальная полная копия + непрерывное архивирование.
   (b) Включить для СУБД режим архивирования WAL;
   (c) настроить копирование WAL (scp) на резервный узел;
   (d) создать первоначальную резервную копию (pg_basebackup),
   (e) скопировать на резервный узел (rsync).
2. Подсчитать, каков будет объем резервных копий спустя месяц работы системы, исходя из следующих условий:
   (a) Средний объем новых данных в БД за сутки: 650МБ.
   (b) Средний объем измененных данных за сутки: 950МБ.
3. Проанализировать результаты.

3.2 Подготовка секретов

Нам необходимо будет отправлять базовую резервную копию, а так WAL файлы на резервный узел, так что сперва следует сгенерировать и распределить ключи шифрования для безопасной передачи данных между узлами.

```
#!/bin/sh

set -e

echo "[primary] Generating ssh key..."
ssh-keygen -t rsa -f ~/.ssh/id_rsa -N ""

echo "[primary] Generated ssh key:"
cat ~/.ssh/id_rsa.pub
```

Листинг 8: Генерация ключей

Далее я авторизовал публичный ключ primary узла на standby. Теперь можно проверить, что передача данных скорее всего будет работать.

```
#!/bin/sh

set -e

ssh -q $1 "echo Hello, World!"
```

Листинг 9: Проверка подключения

3.3 Конфигурация primary узла для резервного копирования

Включаем архивирование, будем отправлять WAL файлы на standby каждые 16 секунд.

```
## Archiving

archive_mode    = on
archive_timeout = 16s
archive_command = 'ssh -q <STANDBY_HOST> "test ! -e <STANDBY_WAL_DIR>/%f" && scp %p <
    STANDBY_HOST>:~/<STANDBY_WAL_DIR>'
```

Листинг 10: Ключевые строчки в конфигурационном файле

3.4 Создание базовой резервной копии

```
#!/bin/sh

set -e

echo "[primary] Creating base backup..."
"$DDB_PGBASEBACKUP" \
    --host="localhost" \
    --port="$DDB_PG_PORT" \
    --pgdata="$HOME/$DDB_PRIMARY_BACKUP_BASE_DIR" \
    --format="tar" \
    --wal-method="fetch" \
    --no-password

echo "[primary] Sending to '$DDB_STANDBY_HOST'..."
rsync -ave ssh "$HOME/$DDB_PRIMARY_BACKUP_BASE_DIR" $DDB_STANDBY_HOST:~/
    $DDB_PRIMARY_BACKUP_DIR
```

Листинг 11: Создание базовой резервной копии

3.5 Подготовка standby

```
#!/bin/sh

set -e

echo "[standby] Prepairing..."

```

```
mkdir -p "$HOME/$DDB_STANDBY_BACKUP_WAL_DIR"
mkdir -p "$HOME/$DDB_PRIMARY_BACKUP_BASE_DIR"
```

Листинг 12: Подготовка standby

3.6 Полная настройка primary узла

```
#!/bin/sh

set -e

echo "[primary] Creating '.pgpass' file..."
sh common/0-pgpass.sh

echo "[primary] Editing 'postgresql.conf' file..."
sh primary/1-config.sh

echo "[primary] Initializing the database..."
sh primary/1-init.sh

echo "[primary] Starting the database..."
sh common/2-start.sh &

echo "[primary] Waiting the database startup..."
sleep 2

echo "[primary] Settings up the database..."
sh primary/3-setup.sh

echo "[primary] All right!"
```

Листинг 13: Полная настройка primary узла

3.7 Действия системного администратора по первоначальной настройке системы

1. Получить конфигурационные файлы системы из репозитория `https://github.com/vityaman-edu/ddb-homework/tree/trunk/lab-3`
2. Доставить директории `db/common` и `db/primary` на `primary` узел, разместив их в домашней директории пользователя, от лица которого будет запущена система
3. Доставить директории `db/common` и `db/standby` на `standby` узел, разместив их в домашней директории пользователя, от лица которого будет запущена система
4. На `primary` узле сгенерировать ключи для `primary` узла при помощи скрипта `common/0-ssh-keygen.sh` и авторизовать публичный ключ на узле standby
5. Проверить на `primary` узле возможность ssh соединения с `standby` узлом при помощи `common/0-ssh-test.sh`
6. Подготовить `standby` узел к резервированию, выполнив на нем `source common/0-env.sh && sh standby/1-prepare.sh`
7. Запустить `primary` узел, выполнив на нем `source common/0-env.sh && sh common/0-pgpass.sh && sh primary/9-full.sh`
8. Создать базовую резервную копию на `primary` узле и отправить ее на `standby` узел: `sh primary/4-backup.sh`
9. Наполнить данными базу данных на `primary` узле: `sh primary/5-fill.sh`
10. Убедиться, что базовая резервная копия и WAL файлы доставлены на `standby` узел

4 Этап 2. Потеря основного узла

4.1 Задача

Этот сценарий подразумевает полную недоступность основного узла. Необходимо восстановить работу СУБД на РЕЗЕРВНОМ узле, продемонстрировать успешный запуск СУБД и доступность данных.

4.2 Восстановление СУБД на резервном узле

Для этого необходимо просто выполнить `source common/0-env.sh && sh standby/2-restore.sh` на `standby` узле, предварительно убедившись, что в директории `primary/backup/base` находятся файлы базовой резервной копии, а в директории `primary/backup/wal` есть WAL сегменты.

```
#!/bin/sh

set -e

mkdir -p $PGDATA
chmod 0700 $PGDATA

echo "[standby] Extracting base backup..."
tar \
    --extract \
    -f "$HOME/$DDB_PRIMARY_BACKUP_BASE_DIR/base.tar" \
    --directory=$PGDATA

echo "[standby] Restoring tablespaces..."
while read -r line; do
    tablespace_oid=$(echo $line | awk '{print $1}')
    echo "[standby][$tablespace_oid] Restoring tablespace..."

    TABLESPACE_DIR=$HOME/tablespace/$tablespace_oid
    echo "[standby][$tablespace_oid] Directory: $TABLESPACE_DIR"
    mkdir -p $TABLESPACE_DIR

    echo "[standby][$tablespace_oid] Extracting..."
    tar --extract \
        -f $HOME/$DDB_PRIMARY_BACKUP_BASE_DIR/$tablespace_oid.tar \
        --directory=$TABLESPACE_DIR

    echo "[standby][$tablespace_oid] Creating symbolic link..."
    ln -s $TABLESPACE_DIR $PGDATA/pg_tblspc/$tablespace_oid
done < $PGDATA/tablespace_map

echo "[standby] Removing 'tablespace_map'..."
rm $PGDATA/tablespace_map

echo "[standby] Patching postgresql.conf: add 'restore_command'..."
RESTORE_CMD="restore_command = 'cp $HOME/$DDB_STANDBY_BACKUP_WAL_DIR/%f %p'"
echo "\n$RESTORE_CMD\n" >> $PGDATA/postgresql.conf

echo "[standby] Patching postgresql.conf: disable archive..."
sed -i -e "s+archive_mode+#archive_mode+g" $PGDATA/postgresql.conf

echo "[standby] Signalling of recovery..."
touch $PGDATA/recovery.signal

echo "[standby] Is ready for startup!"
```

Листинг 14: Восстановление СУБД на резервном узле

5 Этап 3. Повреждение файлов БД

5.1 Задача

Этот сценарий подразумевает потерю данных (например, в результате сбоя диска или файловой системы) при сохранении доступности основного узла. Необходимо выполнить полное восстановление данных из резервной копии и перезапустить СУБД на ОСНОВНОМ узле.

Ход работы:

1. Симулировать сбой: удалить с диска директорию любой таблицы со всем содержимым.
2. Проверить работу СУБД, доступность данных, перезапустить СУБД, проанализировать результаты.
3. Выполнить восстановление данных из резервной копии, учитывая следующее условие: исходное расположение дополнительных табличных пространств недоступно - разместить в другой директории и скорректировать конфигурацию.
4. Запустить СУБД, проверить работу и доступность данных, проанализировать результаты.

5.2 Решение

TODO

6 Этап 4. Логическое повреждение данных

6.1 Задача

Этот сценарий подразумевает частичную потерю данных (в результате нежелательной или ошибочной операции) при сохранении доступности основного узла. Необходимо выполнить восстановление данных на ОСНОВНОМ узле следующим способом:

Генерация файла на резервном узле с помощью pg_dump и последующее применение файла на основном узле.

Ход работы:

1. В каждую таблицу базы добавить 2-3 новые строки, зафиксировать результат.
2. Зафиксировать время и симулировать ошибку: в любой таблице с внешними ключами подменить значения ключей на случайные (INSERT, UPDATE)
3. Продемонстрировать результат.
4. Выполнить восстановление данных указанным способом.
5. Продемонстрировать и проанализировать результат.

6.2 Решение

TODO

7 Вывод

Данная лабораторная работа помогла мне изучить конфигурацию PostgreSQL.

Список литературы

[1] PostgreSQL Documentation: сайт. - 2024. - URL: https://www.postgresql.org/docs/14/index.html (дата обращения: 06.04.2024) - Текст : электронный.