Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА №3**

по дисциплине

‘Распределённые системы хранения данных’

Вариант №39645

Выполнил:

Студент группы P33312

Соболев Иван Александрович

Преподаватель:

Осипов Святослав Владимирович



Санкт-Петербург, 2024

**Задание:**

### **Этап 1. Резервное копирование**

* Настроить резервное копирование с основного узла на резервный следующим образом:

Периодические полные копии + непрерывное архивирование.  
Включить для СУБД режим архивирования WAL; настроить копирование WAL (scp) на резервный узел; настроить полное резервное копирование (pg\_basebackup) по расписанию (cron) раз в неделю. Созданные полные копии должны сразу копироваться (scp) на резервный хост. Срок хранения копий на основной системе - 1 неделя, на резервной - 4 недели. По истечении срока хранения, старые архивы и неактуальные WAL должны автоматически уничтожаться.

* Подсчитать, каков будет объем резервных копий спустя месяц работы системы, исходя из следующих условий:
  + Средний объем новых данных в БД за сутки: 650МБ.
  + Средний объем измененных данных за сутки: 150МБ.
* Проанализировать результаты.

### **Этап 2. Потеря основного узла**

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

### **Этап 3. Повреждение файлов БД**

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:

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

### **Этап 4. Логическое повреждение данных**

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:

* В каждую таблицу базы добавить 2-3 новые строки, зафиксировать результат.
* Зафиксировать время и симулировать ошибку:
  + перезаписать строки любой таблицы “мусором” (INSERT, UPDATE)
* Продемонстрировать результат.
* Выполнить восстановление данных указанным способом.
* Продемонстрировать и проанализировать результат.

**Выполнение:**

### **1. Резервное копирование**

Подключение к узлам:

ssh -J s336760@helios.cs.ifmo.ru:2222 postgres6@pg157

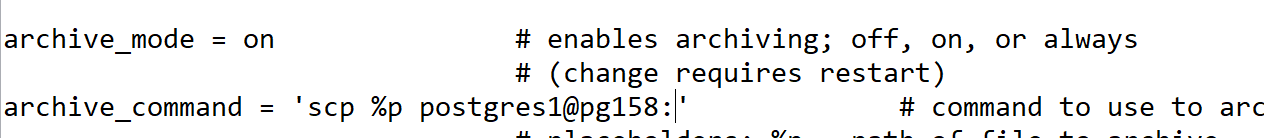
ssh -J s336760@helios.cs.ifmo.ru:2222 postgres1@pg158

Создаем пользователя для будущих реплик:

create role replica replication login password '1234';

postgesql.conf





Основной узел:

#!/usr/local/bin/bash

BACKUPS\_DIR="/var/db/postgres6/backups"

TABLE\_SPACE="/var/db/postgres6/zzo32"

CURRENT\_DATE=$(date "+%Y-%m-%d-%H:%M:%S")

BACKUP\_DIR=$BACKUPS\_DIR/BACKUP\_${CURRENT\_DATE}

DATE=$(date "+%Y%m%d%H%M%S")

BACKUP\_NAME="backup\_${DATE}"

NEW\_TABLE\_SPACE="/var/db/postgres6/tables\_backups/${BACKUP\_NAME}"

pg\_basebackup -h 127.0.0.1 -p 9455 -U postgres6 -D $BACKUP\_DIR -T "${TABLE\_SPACE}"="${NEW\_TABLE\_SPACE}"

scp -r $BACKUP\_DIR postgres1@pg158:~/backups/${BACKUP\_NAME}/

scp -r $NEW\_TABLE\_SPACE postgres1@pg158:~/tables\_backups/${BACKUP\_NAME}/

SECONDS\_IN\_WEEK=$(( 7 \* 24 \* 3600 ))

current\_time=$(date +%s)

for backup\_dir in "$BACKUP\_DIR"/\*; do

file\_modified\_time=$(stat -f %m "$backup\_dir")

time\_diff=$((current\_time - file\_modified\_time))

if [ "$time\_diff" -gt "$SECONDS\_IN\_WEEK" ]; then

rm -rf "$backup\_dir"

echo "file(dir) deleted $backup\_dir"

fi

done

for tables\_dir in "$TABLE\_SPACE"/\*; do

file\_modified\_time=$(stat -f %m "$tables\_dir")

time\_diff=$((current\_time - file\_modified\_time))

if [ "$time\_diff" -gt "$SECONDS\_IN\_MONTH" ]; then

rm -rf "$tables\_dir"

echo "file(dir) deleted $tables\_dir"

fi

done

ssh postgres1@pg158 "bash ~/remove\_script.sh"

Резервный узел:

#!usr/local/bin/bash

BACKUPS\_DIR=\~/backups/

SECONDS\_IN\_MONTH=$(( 28 \* 24 \* 3600 ))

current\_time=$(date +%s)

for backup\_dir in "$BACKUPS\_DIR"/\*; do

file\_modified\_time=$(stat -f %m "$backup\_dir")

time\_diff=$((current\_time - file\_modified\_time))

if [ "$time\_diff" -gt "$SECONDS\_IN\_MONTH" ]; then

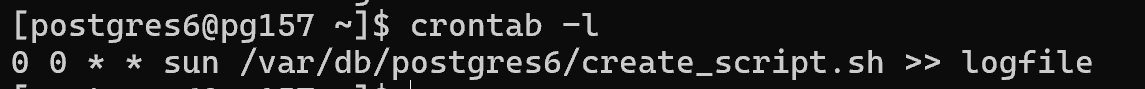
rm -rf "$backup\_dir"

echo "file(dir) deleted $backup\_dir"

fi

done

На основном узле создадим cron-файл через команду (crontab -e), в котором опишем правило для запуска нашего скрипта раз в неделю по воскресеньям в 00:00:00. Проверим список запланированных задач:

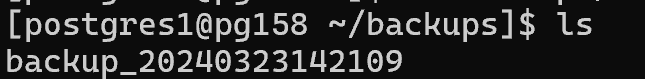


Проверим работу скрипта:

Изображение выглядит как текст, снимок экрана, черный, Шрифт

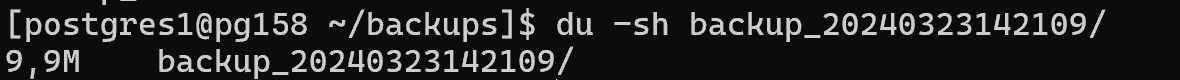
Автоматически созданное описание

Резервная копия создалась:



**Расчет объема:**

Размер одного бэкапа (изначально):



Количество данных для перезаписывания: 650+150 Мб в день. Копии создаются 1 раз в неделю, следовательно за месяц будет создано 4 копии. Так как на основном узле копии хранятся 7 дней, а на резервном 28 дней, то на момент прохождения месяца на основном будет 1 копия, а на резервном – 4.

Посчитаем с помощью арифметической прогрессии размер копий за 28 дней:

S1 = (9.9\*2+650(7-1))/2\*7 = 13719 Мб

S2 =9.9\*2+650(14-1))/2\*14 = 59289 Мб

S3 =9.9\*2+650(21-1))/2\*21 = 136708 Мб

S3 =9.9\*2+650(28-1))/2\*28 = 245977 Мб

Сумма = 455693 Мб = 445 Гб

Так как wal файлы у нас архивируются, то их объем не будет превышать 1Гб (max\_wal\_size)

Общий размер = 446 Гб

**Потеря основного узла**

Воссоздадим файловую структуру кластера для восстановления:

mkdir -p $HOME/ifg51

chmod 700 ifg51/

cp -r ~/backups/backup\_20240323160134/\* ~/ifg51/

Файлы табличного пространства

mkdir -p $HOME/zzo32

chmod 700 zzo32/

cp -r ~/backups/backup\_20240323160134/pg\_tblspc/16384/ ~/zzo32/

Меняем символические ссылки:

ln -s /var/db/postgres1/zzo32/ ~/ifg51/pg\_tblspc/16384

Укажем в postgresql.conf команду для загрузки wal файлов:

restore\_command = 'cp /var/db/postgres1/wal\_archive/%f %p'

Создадим в директории кластера файл, сигнализирующий о восстановлении:

touch ~/ifg51/recovery.signal

Запускаем резервный кластер: postgres -D $HOME/ifg51 >~/logfile 2>&1 &

Проверяем работоспособность: psql -h localhost -p 9455 -U postgres6 wetbluelove

Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

Анализ выполнения

Восстановление завершилось успешно. Но для корректного запуска необходимо подкорректировать ссылку

**Повреждение файлов БД**

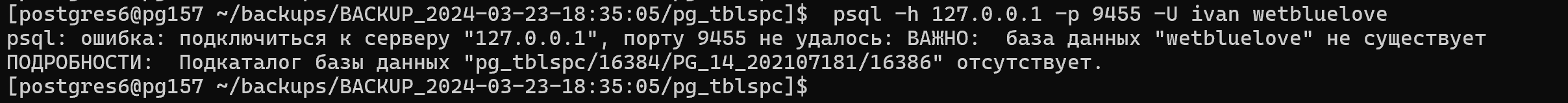
Я запустил кластер с одного из бэкапов и удалил табличное пространство.

Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

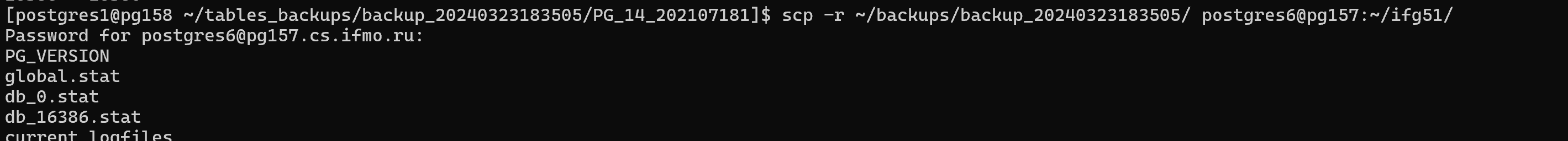
Изображение выглядит как текст, снимок экрана, Шрифт, черный

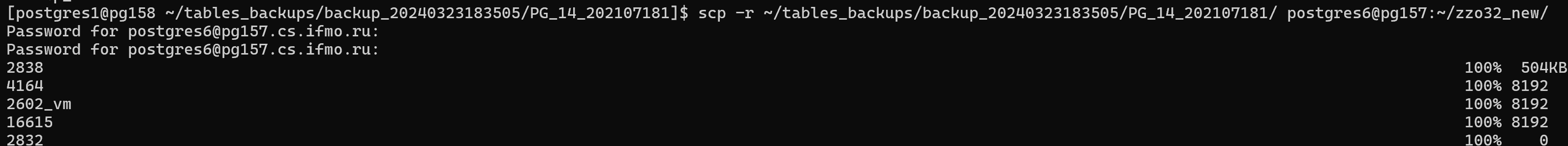
Автоматически созданное описание



Теперь при попытке подключения возникает ошибка.

Переносим копию на основной узел:



Переносим табличные пространства в новую директорию 

Запускаем поврежденный кластер в режиме восстановления wal-файлов и указываем команду для восстановления

Меняем команду восстановления в файле postgresql.conf для текущего бэкапа:



Меняем символическую ссылку:

ln -s /var/db/postgres6/zzo32\_new/ /var/db/postgres6/backups/BACKUP\_2024-03-23-18\:35\:05/pg\_tblspc/16384

Запускаем в режиме восстановления

touch ~/backups/BACKUP\_2024-03-23-18\:35\:05/recovery.signal

postgres -D \$HOME/backups/BACKUP\_2024-03-23-18\:35\:05/ >~/logfile 2>&1 &

Результат:

Изображение выглядит как текст, Шрифт, снимок экрана, черный

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Анализ

Восстановление завершилось успешно. Но для корректного запуска необходимо подкорректировать ссылку

**Логическое повреждение данных**

Сначала БД будет наполнена новыми данными, с которыми потом будут производиться неаккуратные действия.

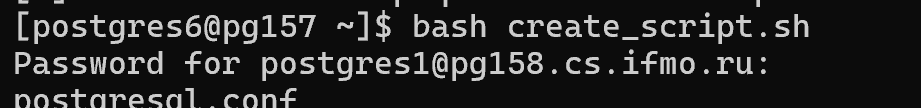
Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

После этого сделаем РК и отправим её на резервный узел:



Далее после запуска на резервном узле появятся следующие данные и та самая пока не «испорченная» БД (данные) на которой будет сделан дамп:

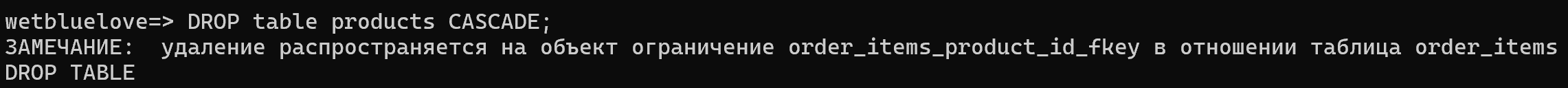
Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Далее делается сам дамп БД wetbluelove, которая будет испорчена на основном узле:

pg\_dump -h 127.0.0.1 -p 9455 -U ivan -f ~/dump\_file wetbluelove

Удалим таблицу с продуктами на основном узле:



Теперь нужно скопировать дамп с резервного узла и применить его на основном:

scp dump\_file postgres6@pg157:~/

Далее восстанавливаем данные из файла в текстовом формате:  
psql -h 127.0.0.1 -p 9455 -d wetbluelove < ~/dump\_file

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

Видно, что таблица опять появилась с теми же данными.

**Выводы**

Во время выполнения лабораторной работы я изучил способы непрерывного бекапа кластера PostgreSQL, на практике настроил и применил его при различных сбоях.