2федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**ОТЧЕТ**

по лабораторной работе №3

по дисциплине «**Распределенные системы хранения данных**»

Вариант 1

Автор: Кулаков Н. В.

Факультет: ПИиКТ

Группа: P33312

Преподаватель: Шешуков Д. М.



Санкт-Петербург 2023

# **Постановка задачи и исходные данные**

Лабораторная работа включает настройку резервного копирования данных с основного узла на резервный, а также несколько сценариев восстановления. Узел из предыдущей лабораторной работы используется в качестве основного; новый узел используется в качестве резервного. В сценариях восстановления необходимо использовать копию данных, полученную на первом этапе данной лабораторной работы.

## Этапы выполнения работы:

### Резервное копирование

1. Настроить резервное копирование с основного узла на резервный следующим образом:

Периодические полные копии с помощью SQL Dump.

По расписанию (cron) раз в сутки, методом SQL Dump с сжатием. Созданные архивы должны сразу перемещаться на резервных хост, они не должны храниться на основной системе. Срок хранения архивов на резервной системе - 4 недели. По истечении срока хранения, старые архивы должны автоматически уничтожаться.

2. Подсчитать, каков будет объем резервных копий спустя месяц работы системы, исходя из следующих условий:

Средний объем новых данных в БД за сутки: ~100 МБ.

3. Проанализировать результаты.

### Потеря основного узла

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

### 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:

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 директории PGDATA недоступно - разместить в другой директории и скорректировать конфигурацию.

4. Запустить СУБД, проверить работу и доступность данных, проанализировать результаты.

### 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:

1. В каждую таблицу базы добавить 2-3 новые строки, зафиксировать результат.

2. Зафиксировать время и симулировать ошибку:

Удалить любые две таблицы (DROP TABLE)

3. Продемонстрировать результат.

4. Выполнить восстановление данных указанным способом.

5. Продемонстрировать и проанализировать результат.

# Выполнение

## Резервное копирование

### Первичная вставка данных в обычное табличное пространство

create table t1 as

(select

i as first,

i::text as second,

i::text as third

from generate\_series(1,1000000) i);

create table t2 as

(select

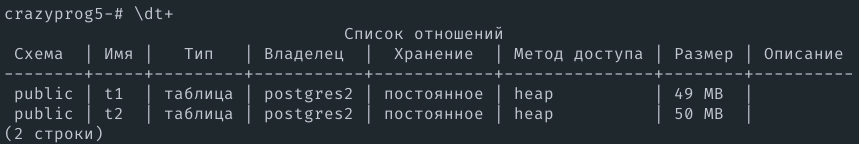
i as first,

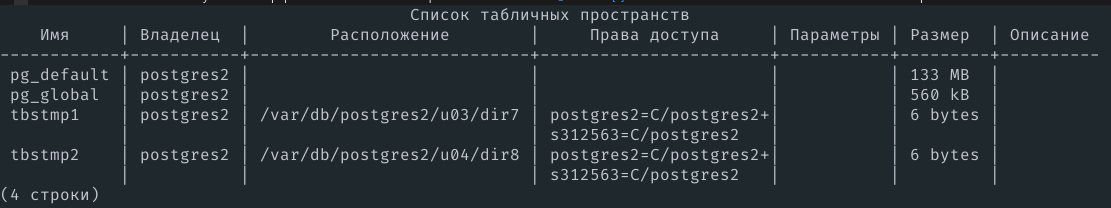
i::text as second,

now() as third

from generate\_series(1,10000) i);

insert into t2 (first,second,third) select i,i::text,now() from generate\_series(1,1000000) i;





### SQL\_DUMP

Предварительно добавим резервный узел с основного в число знакомых узлов ssh и сделаем ssh-copy-id, чтобы не вводить пароль при передаче файлов.

ssh-keygen

…

ssh-copy-id [postgres0@pg120](mailto:postgres0@pg120)-i ~/.ssh/id\_rsa.pub

Имя файла для бекапа генерируется по формату:

"backup\_`date +"%Y\_%m\_%d\_%H:%M:%S"`"

backup\_2023\_04\_24\_10:00:58

Для передачи данных на резервный узел бы использовался rsync, так как он значительно быстрее по сравнению с scp, однако узел не предоставляет данный инструмент.

Для того, чтобы не вводить пароль, была добавлена соответствующая запись в .pgpass.

Создание логического бекапа и отправка его на резервный узел:

#!/usr/local/bin/bash

BACKUP\_FILE=/tmp/pg\_crazyprog5\_`date +"%Y\_%m\_%d\_%H:%M:%S"`\_bak

TARGET\_HOST=pg120

TARGET\_USER=postgres0

TARGET\_PATH=bak

mkdir -p `dirname $BACKUP\_FILE`

pg\_dump --create \

-d crazyprog5 -h pg107 -p 9035 -U postgres2 -Fc -Z4 -f $BACKUP\_FILE

scp $BACKUP\_FILE ${TARGET\_USER}@${TARGET\_HOST}:$TARGET\_PATH

rm $BACKUP\_FILE

Создание задачи cron (запуск каждый день в 2 часа ночи):

0 2 \* \* \* /var/db/postgres2/cron\_pg\_dump.sh

Cron устанавливает директорию как обычный пользователь (crontab -e), поэтому он должен прочитать .pgpass.

Удаление на резервном узле каждые 28 дней (также задача cron):

#/usr/local/bin/bash

BAK\_DIR=/var/db/postgres0/bak

/usr/bin/find $BAK\_DIR -name "\*" -type f -mtime +28 -exec rm -f {} \;

### Подсчет размера бекапов

Коэффициент сжатия: 100 / 6.6 = 15

Тогда каждый день размер файла увеличиваться на 100 / 15 = 6.6 Мб

Тогда к концу месяца (30 дней) будет по формуле арифм суммы: 2.5 Гб

## Потеря основного узла

Восстановление будет происходить на основании логического бекапа. БД на резервном узле не будет иметь wal файлов изменений/добавлений, а только будет иметь идентичные данные, как и на основном узле.

Устанавливаем PGDATA и инициализируем базу данных

export PGDATA="$HOME/u05/dir5"

initdb

Создаем директории для табличных пространств:

mkdir -p u04/dir8

mkdir -p u03/dir7

Создадим БД crazyprog5 и пользователя postgres2:

create role postgres2 login password 'postgres' superuser;

create database crazyprog5 with template = template1;

Для восстановления требуется воспользоваться командой pg\_restore:

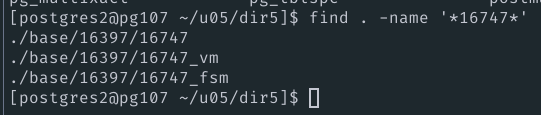
pg\_restore -d crazyprog5 bak/pg\_crazyprog5\_2023\_04\_24\_11\:35\:45\_bak

В консоли не высветилось уведомлений, значит все прошло успешно:



## Повреждение файлов БД

По oid находим местоположение таблицы:



Удаляем файл с соответствующим именем.

Данные из таблицы были удалены:



crazyprog5=# select \* from t1;

ОШИБКА: не удалось открыть файл "base/16397/16747": No such file or directory

Изменяем исходное расположение PGDATA и пересоздаем там базу данных. Сначала передаем бекап на основную машину:

scp bak/pg\_crazyprog5\_2023\_04\_24\_11\:35\:45\_bak postgres2@pg107

Изменяем директорию PGDATA:

export PGDATA=/var/db/postgres2/u06/dir5

Запускаем initdb.

Запускаем новый инстанс, создаем бд, делаем pg\_restore:

pg\_ctl start

create database crazyprog5 with template = template1;

pg\_restore pg\_crazyprog5\_2023\_04\_24\_11\:35\:45\_bak -d crazyprog5 -p 9035

Меняем конфигурационные файлы:

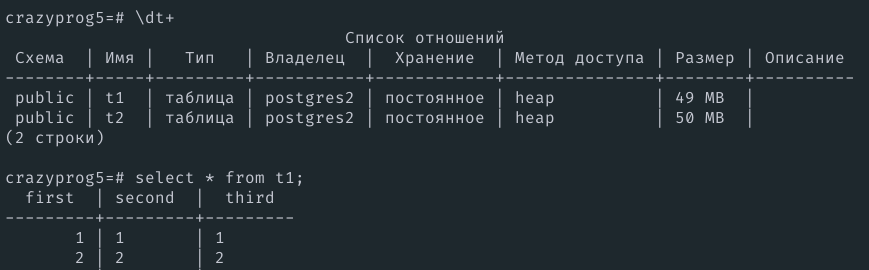
cp u05/dir5/pg\_hba.conf u05/dir5/postgresql.conf u06/dir5/

Останавливаем старый инстанс, перезапускаем новый:

PGDATA=/var/db/postgres2/u05/dir5 pg\_ctl stop -mf

pg\_ctl stop -mf

pg\_ctl start



Теперь снова можем получить доступ к данным в таблице t1.

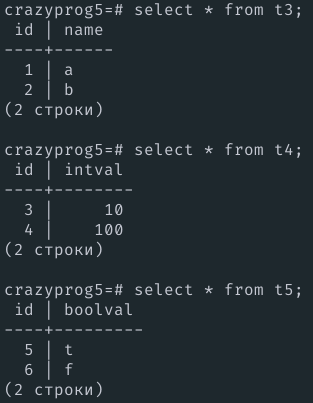
## Логическое повреждение данных

Для демонстрации создадим 3 новых таблицы, добавим в них несколько строчек данных.

create table t3 (id serial primary key, name varchar(255) not null unique);

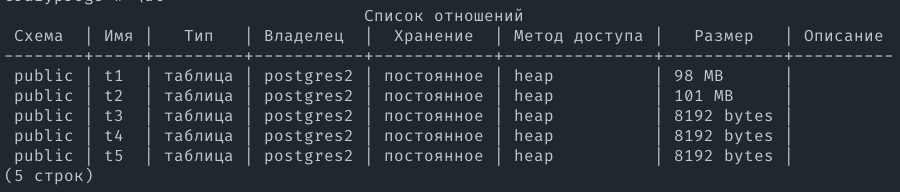
create table t4 (id serial primary key, intval integer not null);

create table t5 (id serial primary key, boolval boolean not null);

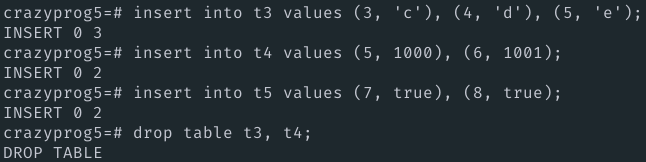


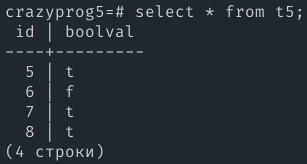
Содержимое созданных таблиц.

Теперь создадим backup, добавим его на резервный узел и произведем restore с помощью backup файла bak/pg\_crazyprog5\_2023\_04\_24\_12\:52\:22\_bak.



Вернемся к основному узлу, добавим несколько записей в таблицы и удалим «по ошибке» таблицы t3, t4.





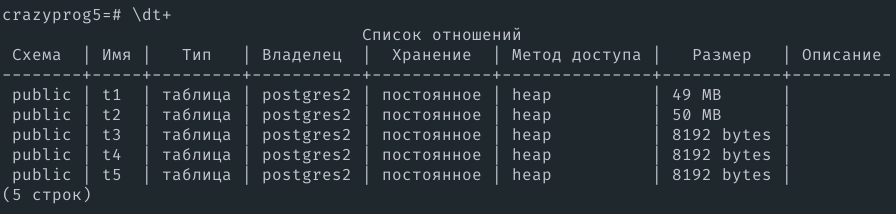
Выполним восстановление данных (восстановим таблицы t3, t4):

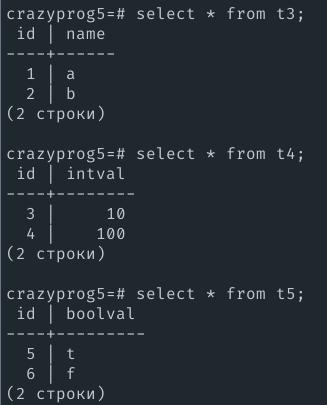
scp bak/pg\_crazyprog5\_2023\_04\_24\_12\:52\:22\_bak postgres2@pg107

Дропнем таблицы t3, t4, t5 и восстановим их из бекапа:

pg\_restore --clean -p 9035 -d crazyprog5 pg\_crazyprog5\_2023\_04\_24\_12\:52\:22\_bak

Посмотрим какие данные теперь хранятся в бд:





Как видим, последние изменения, которые не были забекаплены, не были восстановлены, то есть данные частично потеряны.

# Вывод

В ходе выполнения данной лабораторной работы были разобраны способы создания бекапов, отличие логических и физический бекапов, предназначение wal файлов, а также возможности востановления рабостоспособности базы данных при помощи логического бекапа в случае различных сбоев.

Кроме того, изучено как пользоваться утилитой cron.