

Университет ИТМО  
Факультет программной инженерии и компьютерной техники

# Распределённые системы хранения данных. Лабораторная работа №3.

Группа: Р33131  
Студент: Смирнов Виктор Игоревич  
Преподаватель: Афанасьев Дмитрий Борисович  
Вариант: 736

## Ключевые слова

База данных, конфигурация PostgreSQL.

## Содержание

<b>1</b>	<b>Цель работы и контекст</b>	<b>1</b>
<b>2</b>	<b>Этап 0. Контекст работы</b>	<b>1</b>
2.1	Переменные окружения . . . . .	2
2.2	Конфигурация базы данных . . . . .	2
2.3	Создание <code>pgpass</code> . . . . .	3
2.4	Инициализация базы данных . . . . .	3
2.5	Запуск базы данных . . . . .	4
2.6	Настройка базы данных . . . . .	4
<b>3</b>	<b>Этап 1. Резервное копирование</b>	<b>4</b>
3.1	Задача . . . . .	4
3.2	Подготовка секретов . . . . .	5
3.3	Конфигурация <code>primary</code> узла для резервного копирования . . . . .	5
3.4	Создание базовой резервной копии . . . . .	5
3.5	Подготовка <code>standby</code> . . . . .	6
3.6	Полная настройка <code>primary</code> узла . . . . .	6
3.7	Действия системного администратора по первоначальной настройке системы . . . . .	6
<b>4</b>	<b>Этап 2. Потеря основного узла</b>	<b>7</b>
4.1	Задача . . . . .	7
4.2	Восстановление СУБД на резервном узле . . . . .	7
4.3	Действия на <code>primary</code> узле . . . . .	8
4.4	Действия на <code>standby</code> узле . . . . .	8
<b>5</b>	<b>Этап 3. Повреждение файлов БД</b>	<b>9</b>
5.1	Задача . . . . .	9
5.2	Решение . . . . .	10
<b>6</b>	<b>Этап 4. Логическое повреждение данных</b>	<b>12</b>
6.1	Задача . . . . .	12
6.2	Фиксируем состояние БД . . . . .	12
6.3	Восстановление дампа из бэкапа на резервном узле . . . . .	12
<b>7</b>	<b>Вывод</b>	<b>14</b>
<b>8</b>	<b>Бонус</b>	<b>14</b>

## 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

```
1 #!/bin/sh
2
3 export DDB_PG_CONF="."
4 export DDB_PG_USER="postgres0"
5 export DDB_PG_PASS="pleasehelp"
6 export DDB_PG_PASS_FILE="$DDB_PG_CONF/pgpass.txt"
7 export DDB_PG_PORT=9666
8 export DDB_PG_DATABASE=postgres
9
10 export DDB_TABLESPACE_NAME=yqy90
11 export DDB_TABLESPACE_LOCATION="$HOME/$DDB_TABLESPACE_NAME"
12 export DDB_NEW_DATABASE_NAME=lazyorangehair
13 export DDB_NEW_USER=root
14 export DDB_NEW_USER_PASSWORD=rootik
15
16 export PGDATA="$HOME/kop67"
17
18 export DDB_PG_BIN_DIR=/usr/lib/postgresql/14/bin
19 export DDB_INITDB=$DDB_PG_BIN_DIR/initdb
20 export DDB_PG_BIN=$DDB_PG_BIN_DIR/postgres
21 export DDB_PG_BASEBACKUP=$DDB_PG_BIN_DIR/pg_basebackup
22 export DDB_PG_CTL=$DDB_PG_BIN_DIR/pg_ctl
23 export DDB_PG_VERIFYBACKUP=$DDB_PG_BIN_DIR/pg_verifybackup
24 export DDB_PGDUMP=$DDB_PG_BIN_DIR/pg_dump
25 export DDB_PGRESTORE=$DDB_PG_BIN_DIR/pg_restore
26
27 export DDB_BACKUP_DIR="primary/backup"
28 export DDB_BACKUP_BASE_DIR="$DDB_BACKUP_DIR/base"
29 export DDB_BACKUP_WAL_DIR="$DDB_BACKUP_DIR/wal"
30 export DDB_BACKUP_DUMP_DIR="$DDB_BACKUP_DIR/dump"
31
32 export DDB_PRIMARY_USER=$DDB_PG_USER
33
34 export DDB_STANDBY_HOST=ddb-standby
35 export DDB_STANDBY_USER=$DDB_PG_USER
```

Листинг 1: Переменные окружения

## 2.2 Конфигурация базы данных

```
1 # $PGDATA/pg_hba.conf (Host-based authentication)
2
3 # TYPE DATABASE USER ADDRESS METHOD
4 host all all 127.0.0.1/32 scram-sha-256 # Permit only localhost
5 host all all ::1/128 scram-sha-256 # Permit only localhost
6 host replication all localhost scram-sha-256 # Permit base backup
```

Листинг 2: Конфигурационный файл pg\_hba.conf

```
1 # $PGDATA/postgresql.conf (PostgreSQL configuration file)
2
3 # Note: Optimized for OLAP load:
4 # 5 users, packet r/w 128MB
5
6 ## CONNECTIONS
7
8 listen_addresses = '127.0.0.1' # Available only from localhost
9 port = 9666 # For security
10 unix_socket_directories = '' # Only TCP/IP
11
12 max_connections = 6 # 5 users + 1 extra
13 superuser_reserved_connections = 3
14
15
16 ## AUTHENTICATION
17
18 authentication_timeout = 20s # Type password faster
19 password_encryption = scram-sha-256 # Strong password hashing
20
21 ## RESOURCE USAGE
22
```

```

23 shared_buffers           = 1024MB # 128MB * (5 + 3) users
24 temp_buffers             = 128MB   # 128MB
25 max_prepared_transactions = 0       # We don't use transactions
26 work_mem                 = 256MB    # Expected packet size
27 hash_mem_multiplier      = 1.5     # Smaller hash tables
28 maintenance_work_mem    = 64MB     # ?
29 autovacuum_work_mem      = -1
30 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*: $DDB_STANDBY_USER:$DDB_PG_PASS" >> ~/.pgpass
10 echo "localhost:$DDB_PG_PORT*: $DDB_PRIMARY_USER:$DDB_PG_PASS" >> ~/.pgpass
11 echo "localhost:$DDB_PG_PORT*: $DDB_NEW_USER:$DDB_NEW_USER_PASSWORD" >> ~/.pgpass
12 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

```

1 #!/bin/sh
2
3 set -e
4
5 cd "$(dirname "$0")"
6
7 "$DDB_PGBIN" -D "$PGDATA"

```

Листинг 6: Запуск базы данных

## 2.6 Настройка базы данных

```

1 #!/bin/sh
2
3 set -e
4
5 cd "$(dirname "$0")"
6
7 sql() {
8     psql -h localhost -p "$DDB_PG_PORT" -c "$1" "$DDB_PG_DATABASE"
9 }
10
11 mkdir "$DDB_TABLESPACE_LOCATION" 2>/dev/null
12
13 sql "CREATE TABLESPACE $DDB_TABLESPACE_NAME LOCATION '$DDB_TABLESPACE_LOCATION';"
14 sql "ALTER DATABASE template1 SET TABLESPACE $DDB_TABLESPACE_NAME;"
15 sql "CREATE DATABASE $DDB_NEW_DATABASE_NAME TEMPLATE template1;"
16 sql "CREATE ROLE tester;"
17 sql "CREATE USER $DDB_NEW_USER WITH LOGIN PASSWORD '$DDB_NEW_USER_PASSWORD';"
18 sql "GRANT tester TO $DDB_NEW_USER;"
19
20 sql() {
21     psql -U "$DDB_NEW_USER" -h localhost -p $DDB_PG_PORT -c "$2" "$1"
22 }
23
24 PRV="$DDB_PG_DATABASE"
25 NEW="$DDB_NEW_DATABASE_NAME"
26
27 sql "$PRV" "CREATE TABLE note_prv (id serial PRIMARY KEY, content text NOT NULL);"
28 sql "$NEW" "CREATE TABLE note_new (id serial PRIMARY KEY, content text NOT NULL);"
29
30 sql "$PRV" "INSERT INTO note_prv (content) VALUES ('Note at postgres');"
31 sql "$NEW" "INSERT INTO note_new (content) VALUES ('Note at lazyorangehair');"
32
33 sql "$PRV" "SELECT * FROM note_prv;"
34 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),

- (е) скопировать на резервный узел (rsync).
- 2. Подсчитать, каков будет объем резервных копий спустя месяц работы системы, исходя из следующих условий:
  - (а) Средний объем новых данных в БД за сутки: 650МБ.
  - (б) 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.

```
1 #!/bin/sh
2
3 set -e
4
5 echo "[primary] Generating ssh key..."
6 ssh-keygen -t rsa -f ~/.ssh/id_rsa -N ""
7
8 echo "[primary] Generated ssh key:"
9 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.

```
1 #!/bin/sh
2
3 set -e
4
5 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.

```
1 ## Archiving
2
3 archive_mode = on
4 archive_timeout = 16s
5 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

```
1 #!/bin/sh
2
3 set -e
4
5 echo "[primary] Creating base backup..."
6 "$DDB_PGBASEBACKUP" \
7   --host="localhost" \
8   --port="$DDB_PG_PORT" \
9   --pgdata="$HOME/$DDB_BACKUP_BASE_DIR" \
10  --format="tar" \
11  --wal-method="fetch" \
12  --no-password
13
14 echo "[primary] Sending to '$DDB_STANDBY_USER@$DDB_STANDBY_HOST'..."
15 rsync -ave ssh \
```

```

16 "$HOME/$DDB_BACKUP_BASE_DIR" \
17 $DDB_STANDBY_USER@$DDB_STANDBY_HOST:~/ $DDB_BACKUP_DIR

```

Листинг 11: Создание базовой резервной копии

### 3.5 Подготовка standby

```

1 #!/bin/sh
2
3 set -e
4
5 echo "[standby] Preparing..."
6
7 mkdir -p "$HOME/$DDB_BACKUP_WAL_DIR"
8 mkdir -p "$HOME/$DDB_BACKUP_BASE_DIR"
9 mkdir -p "$HOME/$DDB_BACKUP_DUMP_DIR"

```

Листинг 12: Подготовка standby

### 3.6 Полная настройка primary узла

```

1 #!/bin/sh
2
3 set -e
4
5 echo "[primary] Creating '.pgpass' file..."
6 sh common/pgpass.sh
7
8 echo "[primary] Editing 'postgresql.conf' file..."
9 sh primary/config.sh
10
11 echo "[primary] Initializing the database..."
12 sh primary/init.sh
13
14 echo "[primary] Starting the database..."
15 sh common/start.sh &
16
17 echo "[primary] Waiting the database startup..."
18 sleep 2
19
20 echo "[primary] Settings up the database..."
21 sh primary/setup.sh
22
23 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/ssh-keygen.sh и авторизовать публичный ключ на узле standby
5. Проверить на primary узле возможность ssh соединения с standby узлом при помощи common/ssh-test.sh
6. Подготовить standby узел к резервированию, выполнив на нем source common/env.sh && sh standby/prepare.sh
7. Запустить primary узел, выполнив на нем source common/env.sh && sh common/pgpass.sh && sh primary/full.sh

8. Создать базовую резервную копию на `primary` узле и отправить ее на `standby` узел: `sh primary/backup.sh`
9. Наполнить данными базу данных на `primary` узле: `sh primary/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/env.sh && sh common/restore.sh standby anon-tblspc` 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.

```

1  #!/bin/sh
2
3  # set -e
4
5  INSTANCE=$1
6  echo "[restore] INSTANCE: $INSTANCE"
7
8  MODE=$2
9  echo "[restore] MODE: $MODE"
10
11 TARGET_TIME=$3
12 echo "[restore] TARGET_TIME: $TARGET_TIME"
13
14 echo "[restore] Removing existing database installation..."
15 sh common/clear.sh
16
17 mkdir -p $PGDATA
18 chmod 0700 $PGDATA
19
20 echo "[restore] Extracting base backup..."
21 tar \
22     --extract \
23     -f "$HOME/$DDB_BACKUP_BASE_DIR/base.tar" \
24     --directory=$PGDATA
25
26 echo "[restore] Restoring tablespaces..."
27 while read -r line; do
28     TABLESPACE_OID=$(echo $line | awk '{print $1}')
29
30     if [ "$MODE" = "anon-tblspc" ]; then
31         TABLESPACE_DIR="$HOME/tablespace/$TABLESPACE_OID"
32     else
33         TABLESPACE_DIR=$(echo $line | awk '{print $2}')
34     fi
35
36     echo "[restore][$TABLESPACE_OID] Restoring tablespace..."
37
38     echo "[restore][$TABLESPACE_OID] Directory: $TABLESPACE_DIR"
39     mkdir -p $TABLESPACE_DIR
40
41     echo "[restore][$TABLESPACE_OID] Extracting..."
42     tar --extract \
43         -f "$HOME/$DDB_BACKUP_BASE_DIR/$TABLESPACE_OID.tar" \
44         --directory=$TABLESPACE_DIR
45
46     echo "[restore][$TABLESPACE_OID] Creating symbolic link..."
47     ln -s $TABLESPACE_DIR $PGDATA/pg_tblspc/$TABLESPACE_OID

```



```

48 done < $PGDATA/tablespace_map
49
50 echo "[restore] Checking base backup integrity..."
51 "$DDB_PGVERIFYBACKUP" \
52   --manifest-path="$HOME/$DDB_BACKUP_BASE_DIR/backup_manifest" \
53   --wal-directory="$HOME/$DDB_BACKUP_WAL_DIR" \
54   $PGDATA
55
56 echo "[restore] Removing 'tablespace_map'..."
57 rm $PGDATA/tablespace_map
58
59 # Patching is used as we need to change values depending on env variables
60 echo "[restore] Patching postgresql.conf: add 'restore_command'..."
61 RESTORE_CMD="restore_command = 'cp ~/$DDB_BACKUP_WAL_DIR/%f %p'"
62 echo "$RESTORE_CMD" >> $PGDATA/postgresql.conf
63
64 if [ "$INSTANCE" = "standby" ]; then
65   echo "[restore] Patching postgresql.conf: disable archive..."
66   sed -i -e "s+archive_mode+#archive_mode+g" $PGDATA/postgresql.conf
67 fi
68
69 if [ "$TARGET_TIME" != "" ]; then
70   echo "[restore] Patching postgresql.conf: setting target time..."
71   echo "recovery_target_time = '$TARGET_TIME'" >> $PGDATA/postgresql.conf
72   echo "recovery_target_inclusive = false" >> $PGDATA/postgresql.conf
73 fi
74
75 echo "[restore] Signalling of recovery..."
76 touch $PGDATA/recovery.signal
77
78 echo "[restore] Is ready for startup!"

```

Листинг 14: Восстановление СУБД

### 4.3 Действия на primary узле

```

1 postgres002d09031f584d:~$ history
2 1  ls
3 2  source common/env.sh
4 3  sh common/ssh-keygen.sh
5 4  sh common/ssh-test.sh ddb-standby
6 5  sh common/pgpass.sh
7 6  sh primary/full.sh
8 7  sh primary/backup.sh
9 8  sh primary/fill.sh
10 9  history

```

Листинг 15: Действия на primary узле

### 4.4 Действия на standby узле

```

1 postgres00c33884c20d42:~$ history
2 2  ls
3 3  source common/env.sh
4 4  vim .ssh/authorized_keys
5 5  sh standby/prepare.sh
6 6  ls
7 7  ls primary/backup/base/
8 8  ls primary/backup/wal/
9 9  ls primary/backup/base
10 10 ls primary/backup/wal/
11 11 sh standby/restore.sh
12 12 sh common/start.sh
13 13 bg
14 14 history

```

Листинг 16: Действия на standby узле

```

$ sh common/start.sh
2024-05-28 17:29:22.989 GMT [61131] СООБЩЕНИЕ: запускается PostgreSQL 14.2 on amd64-portbld-freebsd1

```

```

compiled by FreeBSD clang version 11.0.1 (git@github.com:llvm/llvm-project.git llvmorg-11.0.1-0-g43f
64-bit
2024-05-28 17:29:22.989 GMT [61131] СООБЩЕНИЕ: для приёма подключений по адресу IPv4 "127.0.0.1"откр
порт 9666
2024-05-28 17:29:22.991 GMT [61132] СООБЩЕНИЕ: работа системы БД была прервана; последний
момент работы: 2024-05-28 17:19:51 GMT
ср: /var/db/postgres1/primary/backup/wal/00000002.history: No such file or directory
2024-05-28 17:29:23.012 GMT [61132] СООБЩЕНИЕ: начинается восстановление архива
2024-05-28 17:29:23.014 GMT [61132] СООБЩЕНИЕ: файл журнала "00000001000000000000000002"восстановлен
из архива
2024-05-28 17:29:23.017 GMT [61132] СООБЩЕНИЕ: запись REDO начинается со смещения 0/2000028
2024-05-28 17:29:23.017 GMT [61132] СООБЩЕНИЕ: согласованное состояние восстановления достигнуто
по смещению 0/2000138
2024-05-28 17:29:23.018 GMT [61131] СООБЩЕНИЕ: система БД готова принимать подключения в
режиме "только чтение"
2024-05-28 17:29:23.019 GMT [61132] СООБЩЕНИЕ: файл журнала "00000001000000000000000003"восстановлен
из архива
2024-05-28 17:29:23.029 GMT [61132] СООБЩЕНИЕ: файл журнала "00000001000000000000000004"восстановлен
из архива
ср: /var/db/postgres1/primary/backup/wal/00000001000000000000000005: No such file or directory
2024-05-28 17:29:23.031 GMT [61132] СООБЩЕНИЕ: записи REDO обработаны до смещения 0/4000110,
нагрузка системы: CPU: пользов.: 0.00 с, система: 0.00 с, прошло: 0.01 с
2024-05-28 17:29:23.031 GMT [61132] СООБЩЕНИЕ: последняя завершённая транзакция была выполнена
в 2024-05-28 17:20:13.468706+00
2024-05-28 17:29:23.034 GMT [61132] СООБЩЕНИЕ: файл журнала "00000001000000000000000004"восстановлен
из архива
ср: /var/db/postgres1/primary/backup/wal/00000002.history: No such file or directory
2024-05-28 17:29:23.035 GMT [61132] СООБЩЕНИЕ: выбранный ID новой линии времени: 2
2024-05-28 17:29:23.043 GMT [61132] СООБЩЕНИЕ: восстановление архива завершено
ср: /var/db/postgres1/primary/backup/wal/00000001.history: No such file or directory
2024-05-28 17:29:23.054 GMT [61131] СООБЩЕНИЕ: система БД готова принимать подключения

```

```

1 $ sh common/connect.sh postgres
2 postgres=# select * from note_prv;
3  id |          content
4  ---+-----
5    1 | Note at postgres
6    2 | Another note at postgres
7 (2 rows)
8
9 $ sh common/connect.sh lazyorangehair
10 lazyorangehair=# select * from note_new;
11  id |          content
12  ---+-----
13    1 | Note at lazyorangehair
14    2 | Another note at lazyorangehair
15 (2 rows)

```

Листинг 17: Состояние СУБД на standby узле

Как мы видим, в базе данных сохранились не только данные с базовой копии, но и подтянулись изменения из WAL сегментов.

## 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

Попробуем удалить таблицу `note_prv`. Для этого узнаем, где представлены ее данные в файловой системе.

```
1 $ psql -h localhost -p "$DDB_PG_PORT" "$DDB_PG_DATABASE"
2 postgres=# SELECT pg_relation_filepath('note_prv');
3 pg_relation_filepath
4 -----
5 base/14115/16389
6 (1 строка)
```

Листинг 18: Получаем физическую локацию таблицы

Начинаем уничтожение.

```
1 $ rm $PGDATA/base/14374/16389
```

Листинг 19: Удаляем файл таблицы

Наблюдаем эффекты.

```
$ psql -h localhost -p "$DDB_PG_PORT"$DDB_PG_DATABASE"
postgres=# select * from note_prv;
ERROR: could not open file "base/14374/16389": No such file or directory
```

Попробуем перезапустить базу данных.

```
1 $ sh common/stop.sh
2 waiting for server to shut down.... done
3 server stopped
4
5 $ sh common/start.sh
6 2024-05-28 10:52:21.905 GMT [9200] LOG: starting PostgreSQL 14.12 (Ubuntu 14.12-1.
   pgdg20.04+1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2)
   9.4.0, 64-bit
7 2024-05-28 10:52:21.905 GMT [9200] LOG: listening on IPv4 address "127.0.0.1", port
   9666
8 2024-05-28 10:52:21.908 GMT [9206] LOG: database system was shut down at 2024-05-28
   10:52:16 GMT
9 2024-05-28 10:52:21.916 GMT [9200] LOG: database system is ready to accept connections
10
11 $ psql -h localhost -p "$DDB_PG_PORT" "$DDB_PG_DATABASE"
12 postgres=# select * from note_prv;
13 2024-05-28 10:54:18.117 GMT [9230] ERROR: could not open file "base/14374/16389": No
   such file or directory
14 2024-05-28 10:54:18.117 GMT [9230] STATEMENT: select * from note_prv;
15 ERROR: could not open file "base/14374/16389": No such file or directory
```

Листинг 20: Перезапускаем базу

Видим, что postgres не заметил пропажи, значит не проверил целостность файлов. Наверное, должен существовать способ осуществить проверку целостность его файлов. `pg_checksums` не обнаружил проблему. По идее, можно использовать для проверки утилиту `pg_verifybackup`, которая проверит, соответствует ли содержимое PGDATA заданному бэкапу.

Скачиваем бэкап с `standby`, сносим нашу большую базу, запускаем восстановление и проверяем результат.

```

1 $ sh primary/download.sh
2
3 $ ls primary/backup/base primary/backup/wal/
4 primary/backup/base:
5 16384.tar backup_manifest base.tar
6
7 primary/backup/wal/:
8 00000001000000000000000001
9 00000001000000000000000002
10 00000001000000000000000002.00000028.backup
11 00000001000000000000000003
12 00000001000000000000000004
13 00000001000000000000000005
14
15 $ sh common/restore.sh

```

Листинг 21: Восстанавливаем базу данных

```

$ sh common/start.sh
2024-05-28 17:42:12.835 GMT [61895] СООБЩЕНИЕ: запускается PostgreSQL 14.2 on amd64-portbld-freebsd1
compiled by FreeBSD clang version 11.0.1 (git@github.com:llvm/llvm-project.git llvmorg-11.0.1-0-g43f
64-bit
2024-05-28 17:42:12.835 GMT [61895] СООБЩЕНИЕ: для приёма подключений по адресу IPv4 "127.0.0.1"откр
порт 9666
2024-05-28 17:42:12.838 GMT [61896] СООБЩЕНИЕ: работа системы БД была прервана; последний
момент работы: 2024-05-28 17:19:51 GMT
cp: /var/db/postgres0/primary/backup/wal/00000002.history: No such file or directory
2024-05-28 17:42:12.840 GMT [61896] СООБЩЕНИЕ: начинается восстановление архива
2024-05-28 17:42:12.842 GMT [61896] СООБЩЕНИЕ: файл журнала "000000010000000000000002"восстановлен
из архива
2024-05-28 17:42:12.843 GMT [61896] СООБЩЕНИЕ: запись REDO начинается со смещения 0/2000028
2024-05-28 17:42:12.843 GMT [61896] СООБЩЕНИЕ: согласованное состояние восстановления достигнуто
по смещению 0/2000138
2024-05-28 17:42:12.844 GMT [61895] СООБЩЕНИЕ: система БД готова принимать подключения в
режиме "только чтение"
2024-05-28 17:42:12.845 GMT [61896] СООБЩЕНИЕ: файл журнала "000000010000000000000003"восстановлен
из архива
2024-05-28 17:42:12.852 GMT [61896] СООБЩЕНИЕ: файл журнала "000000010000000000000004"восстановлен
из архива
2024-05-28 17:42:12.854 GMT [61896] СООБЩЕНИЕ: файл журнала "000000010000000000000005"восстановлен
из архива
2024-05-28 17:42:12.856 GMT [61896] СООБЩЕНИЕ: файл журнала "000000010000000000000006"восстановлен
из архива
cp: /var/db/postgres0/primary/backup/wal/000000010000000000000007: No such file or directory
2024-05-28 17:42:12.858 GMT [61896] СООБЩЕНИЕ: записи REDO обработаны до смещения 0/60000D8,
нагрузка системы: CPU: пользов.:0.00 с, система: 0.00 с, прошло: 0.01 с
2024-05-28 17:42:12.858 GMT [61896] СООБЩЕНИЕ: последняя завершённая транзакция была выполнена
в 2024-05-28 17:20:13.468706+00
2024-05-28 17:42:12.860 GMT [61896] СООБЩЕНИЕ: файл журнала "000000010000000000000006"восстановлен
из архива
cp: /var/db/postgres0/primary/backup/wal/00000002.history: No such file or directory
2024-05-28 17:42:12.862 GMT [61896] СООБЩЕНИЕ: выбранный ID новой линии времени: 2
2024-05-28 17:42:12.865 GMT [61896] СООБЩЕНИЕ: восстановление архива завершено
cp: /var/db/postgres0/primary/backup/wal/00000001.history: No such file or directory
2024-05-28 17:42:12.876 GMT [61895] СООБЩЕНИЕ: система БД готова принимать подключения
$ psql -h localhost -p "$DDB_PG_PORT$DDB_PG_DATABASE"
postgres=# select * from note_prv;
id | content
---+-----
1 | Note at postgres
2 | Another note at postgres
(2 rows)

```

Починили.

Вторая часть выполняется аналогично, но нужно вызывать восстановление через `sh common/restore.bash primary anon-tblspc`.

## 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 Фиксируем состояние БД

Сперва запустим вставку нескольких строк на основном узле через скрипт, после чего фиксируем текущее время 2024-05-28 17:46:57.818.

```
1 #!/bin/sh
2
3 set -e
4
5 cd "$(dirname "$0")"
6
7 sql() {
8     psql -U "$DDB_PG_USER" -h localhost -p $DDB_PG_PORT -c "$1" $DDB_PG_DATABASE
9 }
10
11 sql "INSERT INTO note_prv (content) VALUES ('The first testing note');"
12 sql "INSERT INTO note_prv (content) VALUES ('The second testing note');"
13 sql "INSERT INTO note_prv (content) VALUES ('The third testing note');"
```

Листинг 22: Вставка дополнительных строк

Теперь испортим. Не очень понятно, как я должен портить данные, поэтому просто вставляю строку-индикатор: `insert into note_prv (content) values ('aaaaaaaaaaaa');`

### 6.3 Восстановление дампа из бэкапа на резервном узле

Теперь летим на резервный узел и там восстанавливаем бд до заданного момента времени: `sh standby/backup2dump.sh "2024-05-28 17:46:57"`.

```
1 #!/bin/sh
2
3 set -e
4
5 TARGET_TIME=$1
6 if [ "$TARGET_TIME" = "" ]; then
7     echo "[backup2dump] Expected TARGET_TIME parameter!"
8     exit 1
9 fi
10
11 echo "[backup2dump] Starting converting backup to sql dump..."
12
```

```

13 echo "[backup2dump] Starting restoration..."
14 sh common/restore.sh standby anon-tblspc "$TARGET_TIME"
15
16 echo "[backup2dump] Starting the database..."
17 sh common/start.sh &
18
19 echo "[backup2dump] Waiting the database..."
20 sleep 2
21
22 echo "[backup2dump] Starting dumping..."
23 sh common/sql-dump.sh
24
25 echo "[backup2dump] Shutting down the database..."
26 sh common/stop.sh
27
28 echo "[backup2dump] Done!"

```

Листинг 23: Скрипт backup2dump

```

1 #!/bin/sh
2
3 set -e
4
5 dump() {
6     echo "[dump] Dumping '$1'..."
7     pg_dump \
8         --host=localhost \
9         --port=$DDB_PG_PORT \
10        --username=$DDB_PG_USER \
11        --format=tar \
12        --clean \
13        --if-exists \
14        --blobs \
15        --file=$DDB_BACKUP_DUMP_DIR/$1.tar \
16        $1
17 }
18
19 dump $DDB_PG_DATABASE
20 dump $DDB_NEW_DATABASE_NAME

```

Листинг 24: Скрипт sql-dump

После этого в директории с бэкапами появились нужные нам архивы.

Теперь летим на главный узел и запускаем наш кошмар там: `sh primary/download.sh && sh common/sql-restore.sh`.

```

1 #!/bin/sh
2
3 set -e
4
5 echo "[sql-restore] Restoring the database from an SQL dump..."
6
7 restore() {
8     echo "[sql-restore] Restoring '$1'..."
9     "$DDB_PGRESTORE" \
10        --dbname=$1 \
11        --host=localhost \
12        --port=$DDB_PG_PORT \
13        --clean \
14        --if-exists \
15        $DDB_BACKUP_DUMP_DIR/$1.tar
16 }
17
18 restore $DDB_PG_DATABASE
19 restore $DDB_NEW_DATABASE_NAME
20
21 echo "[sql-restore] Done!"

```

Листинг 25: Скрипт sql-restore

Проверяем, что в таблицах.

```

postgres=# select * from note_prv;
id | content

```

```
1 | Note at postgres
2 | Another note at postgres
35 | The first testing note
36 | The second testing note
37 | The third testing note
```

## 7 Вывод

Данная лабораторная работа помогла мне по-настоящему ощутить, какого это - быть администратором PostgreSQL. Я научился создавать резервные копии и накатывать их. В предложенной реализации есть проблема, связанная с тем, что я не накатываю валы периодически, из-за чего через 20 лет я не смогу восстановить БД за разумное время. Также я научился писать переносимые шел скрипты для автоматизации развертывания СУБД.

## 8 Бонус

```
1 FROM ubuntu:20.04
2
3 RUN apt-get update
4 RUN apt-get -y install wget gpg gnupg2 lsb-release
5
6 RUN wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | apt-key add -
7 RUN sh -c 'echo "deb https://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg
   main" > /etc/apt/sources.list.d/pgdg.list'
8 RUN apt-get update
9
10 RUN DEBIAN_FRONTEND=noninteractive apt-get -y install postgresql-14
11
12 RUN apt-get -y install locales locales-all
13
14 RUN apt-get -y install vim rsync
15
16 RUN apt-get -y install openssh-client openssh-server
17 RUN mkdir -p /run/sshd && chmod 755 /run/sshd
18 RUN echo "PasswordAuthentication yes" >> /etc/ssh/sshd_config
19 RUN echo "PermitRootLogin no" >> /etc/ssh/sshd_config
20
21 RUN useradd -ms /bin/bash postgres0
22 RUN echo "postgres0:changeme" | chpasswd
23 USER postgres0
24 WORKDIR /home/postgres0
25 RUN mkdir ~/.ssh
26 RUN touch ~/.ssh/authorized_keys
27
28 USER root
29 RUN cd /etc/ssh && ssh-keygen -A
30
31 CMD service ssh start && bash
32 EXPOSE 22
```

## Список литературы

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