

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,   
МЕХАНИКИ И ОПТИКИ»

Факультет программной инженерии и компьютерной техники

Дисциплина:

Администрирование систем управления базами данных

**Лабораторная работа №3**

**Преподаватель:**

Николаев Владимир Вячеславович

**Выполнили:**

Мизевич Илья

Р32202

# Задание

**1. Резервное копирование**

Настроить резервное копирование с основного узла на резервный следующим образом:

* + Первоначальная полная копия + непрерывное архивирование.
  + ПолВключить для СУБД режим архивирования WAL; настроить копирование WAL (scp) на резервный узел; создать первоначальную резервную копию (pg\_basebackup), скопировать на резервный узел (rsync).

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

* + Средний объем новых данных в БД за сутки: ~80 МБ.

Проанализировать результаты.

**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.

**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 директории PGDATA недоступно - разместить в другой директории и скорректировать конфигурацию.
  + Запустить СУБД, проверить работу и доступность данных, проанализировать результаты.

**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:

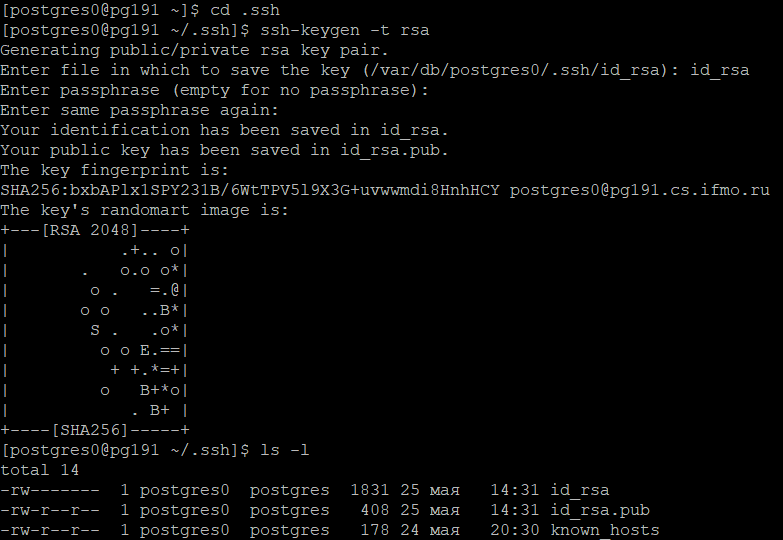
* + Генерация файла на резервном узле с помощью pg\_dump и последующее применение файла на основном узле.

Ход работы:

* + В каждую таблицу базы добавить 2-3 новые строки, зафиксировать результат.
  + Зафиксировать время и симулировать ошибку: В любой таблице с внешними ключами изменить внешние ключи случайным образом (INSERT, UPDATE)
  + Продемонстрировать результат.
  + Выполнить восстановление данных указанным способом.
  + Продемонстрировать и проанализировать результат.

# Выполнение

Для того, чтобы отправлять на резервный узел копии по расписанию, нам необходимо создать ключ-пару ssh, чтобы можно было подключаться с основного узла на резервный без пароля.:



В результате этой операции создается два ключа – приватный для авторизации машины и публичный, который нужно отправить на резервный хост. Для этого сначала запустим ssh-agent:



Затем отправим публичный ключ на резервный хост:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

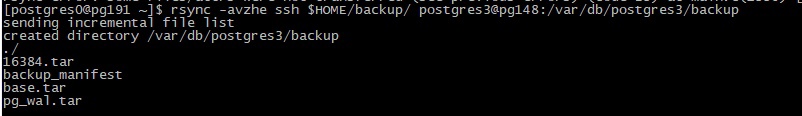
После чего, как мы можем заметить на скриншоте выше, мы можем подключаться к резервному хосту без ввода пароля.

**1. Резервное копирование**

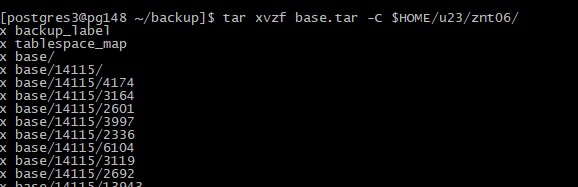
Сначала создадим все директории на резевном узле. После сделаем бэкап основного узла:

https://sun9-21.userapi.com/impg/nQ4iJuZ6QbcBm1Ue0bap18Fr612fdPwlKjVT8A/H6qRbbvpYjI.jpg?size=614x29&quality=96&sign=7e94fd8c91d7557a74ba3b35639773ac&type=album

С помощью утилиты rsync скопируем бэкап на резервный узел:



Затем разархивируем все в директорию ~u23/znt06:



Скрипт

**# Указываем, что наш скрипт использует команды на языке bash**

**#!/bin/bash**

**# Задаем значение переменной REMOTE\_HOST для подключения к удаленному серверу**

**REMOTE\_HOST=remote\_user@remote\_host**

**# Создаем директорию ~/backup на удаленном сервере через SSH**

**ssh $REMOTE\_HOST "mkdir ~/backup"**

**# Создаем директорию ~/u23 на удаленном сервере через SSH**

**ssh $REMOTE\_HOST "mkdir ~/u23"**

**# Создаем директорию ~/u34 на удаленном сервере через SSH**

**ssh $REMOTE\_HOST "mkdir ~/u34"**

**# Создаем директорию ~/new\_dir на удаленном сервере через SSH**

**ssh $REMOTE\_HOST "mkdir ~/new\_dir"**

**# Создаем директорию ~/backup/wal на удаленном сервере через SSH**

**ssh $REMOTE\_HOST "mkdir ~/backup/wal"**

**# Запускаем работу СУБД на удаленном сервере из директории ~/u23/znt06**

**pg\_ctl -D ~/u23/znt06 start**

**# Создаем базовую резервную копию в директории ~/backup на локальном хосте**

**pg\_basebackup -D $HOME/backup -p 9801 -h localhost -Ft**

**# Останавливаем работу СУБД на удаленном сервере из директории ~/u23/znt06**

**pg\_ctl -D ~/u23/znt06 stop**

**# Копируем резервную копию с локального хоста на удаленный сервер с помощью rsync**

**rsync -avzhe ssh $HOME/backup/ $REMOTE\_HOST:/var/db/postgres3/backup/**

**# Распаковываем файл pg\_wal.tar в директорию ~/backup/wal на удаленном сервере через SSH**

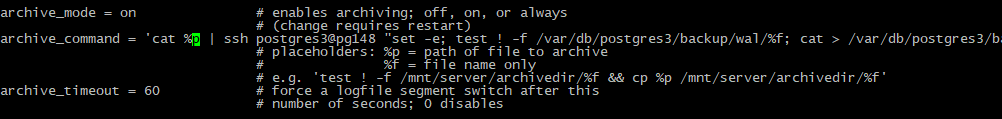
**ssh $REMOTE\_HOST "tar xvzf ~/backup/pg\_wal.tar -C ~/backup/wal"**

Повторим эту процедуру для созданного табличного пространства. После создадим папку где будут храниться архивы WAL файлов. Разархивируем в нее содержимое архива pg\_wal.tar. После нужно восстановить ссылку на табличное пространсво в директории ~/u23/znt06/pg\_tblspace

ln -s /var/db/postgres3/u34/znt08/ 16384

Так же выдадим права на все разархивированные файлы 750.

Теперь настроим непрерывное архивирование на основном узле.

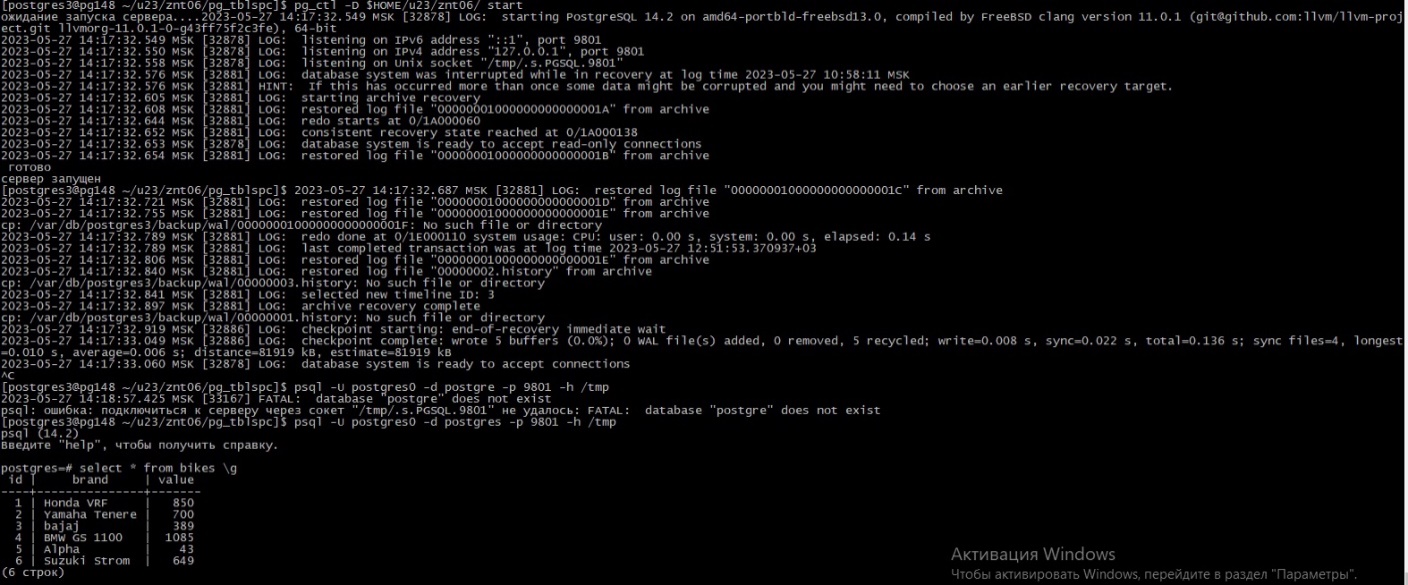


Выставим archive\_timeout 60 секунд что бы не дожидаться заполнения 16 мегабайт и wal файл архивировлся по времени.

Так же укажем в конфиге recovery\_command = 'cp /var/db/postgres3/backup/wal/%f %p'

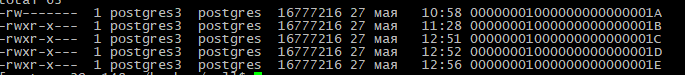
Создадим файл recovery.signal в директории touch ~/u23/znt06/.

Запустим кластер и увидем что наши данные восстановились.



Вставим новые данные на основном узле 

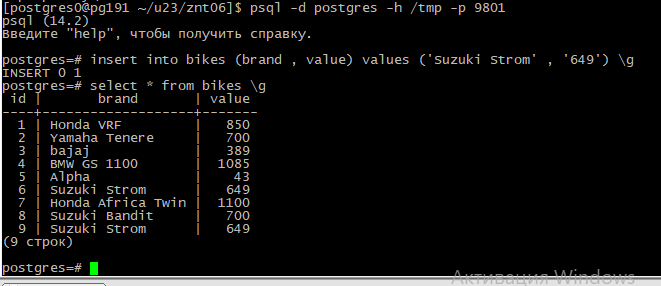
Содержимое папки wal на резевном узле сначала:



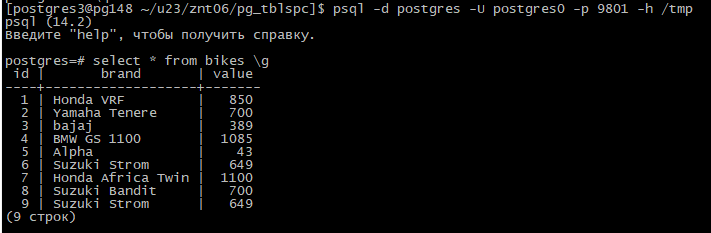
Новый вол файл пришел после минуты работы основной базы:



До восстановления данных нет:



Повторяем процедуру восстановления:



Скрипт

#!/bin/bash

# Запрашиваем у пользователя ввод пути до директории tablespaces

echo -n "Enter tablespaces dir: "

read TABLESPACES\_DIR

# Удаляем содержимое директории ~/u23/znt06

rm -r ~/u23/znt06/\*

# Удаляем содержимое директории $TABLESPACES\_DIR

rm -r $TABLESPACES\_DIR/\*

# Распаковываем файл base.tar в директорию ~/u23/znt06

tar xvzf ~/backup/base.tar -C ~/u23/znt06/

# Создаем файл recovery.signal в директории ~/u23/znt06

touch ~/u23/znt06/recovery.signal

# Распаковываем файл 16384.tar в директорию $TABLESPACES\_DIR на локальном хосте

tar xvzf ~/backup/16384.tar -C $HOME/$TABLESPACES\_DIR

# Устанавливаем права доступа на домашнюю директорию текущего пользователя

chmod -R 750 ~/

# Устанавливаем права доступа на домашнюю директорию текущего пользователя

Каждый архивированный WAL файл весить 16 мегабайт. WAL файлы архивируются при заполнении их до 16 мегабайт. Но в задании было указано установить непрерывную архивацию , поэтому был установлен таймер архивировать WAL файл по истечению времени. Для выполнения лабораторной работы был установлен таймер в 60 секунд. Для реальных условий это слишком мало , т.к любые малейшие изменения в базе будут весить 16 мб , что является расточительством. Поэтому в дальнейших рассуждениях будем считать что таймер установлен на 5 минут (600 секунд).

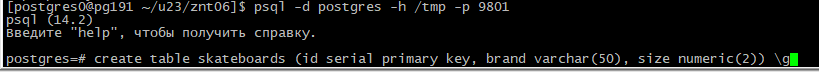
В задании указано, что средний объем измененных данных за сутки равен 80 МБ. Предположим, что в худшем случае все эти 80 МБ являются новыми данными, т.е. их нужно напрямую прибавлять к размеру копии. Т.к. мы напрямую копируем файлы postgres, то дополнительных множителей у нас нет.

Рассчитаем на сколько больше будет весить восстановленная база , относительно первоначальной резервной копии, если каждый день будет приходить по 80 мб новых данных. 80мб \* 30 = 2,34Гб.

Теперь рассчитаем вес архивированных WAL файлов. WAL файл вне зависимости от веса входных данных по истечению таймера будет весить 16 мегабайт. Поэтому возьмем сценарий что за каждые (пока не накопится 80 мб) 10 минут в день будет приходить по 1 мегабайту данных. Следовательно каждые 10 минут будет появляться 1 WAL файл весом 16мб. 16мб \* 80 = 1280 мб = 1,25 Гб. Если в день объем архивных WAL файлов будет составлять примерно 1,25 ГБ то за 30 дней объем составит 1,25 \* 30 = 37,5Гб

**2. Потеря основного узла**

Теперь восстановим данные на втором узле из холодной полной копии. Создадим на основном узле еще одну таблицу и заполним ее:



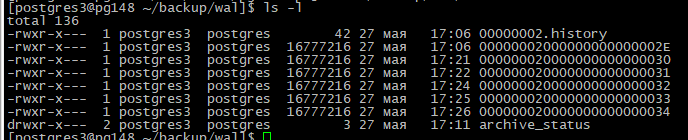




Допустим база безвозвратно упала , к ней не подключиться на основном узле:



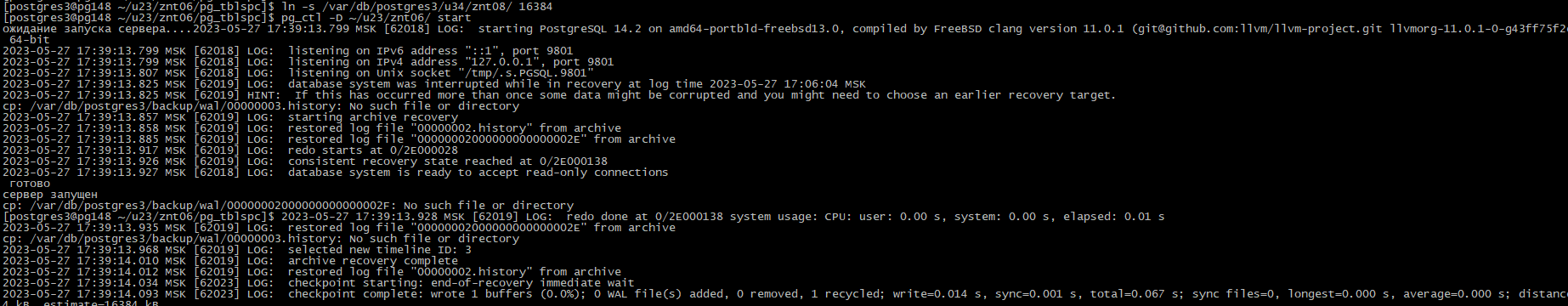
По скольку у нас была настроена архивация WAL проверяем папку с бэкапом на резеврном узле. Видим файлы появившиеся перед «смертью» основного узла:



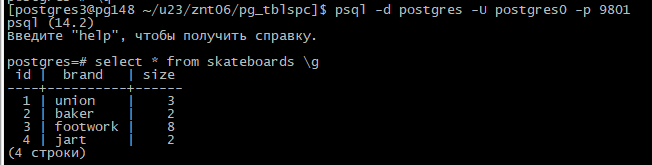
Заново накатываем из архива резервную копию.

Заново создаем символические ссылки на табличные пространства в директории pg\_tblspce.

Создаем опять файл recovery.signal и запускаем базу:

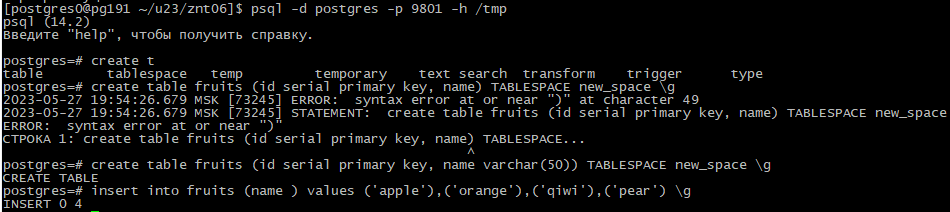


Проверяем данные:

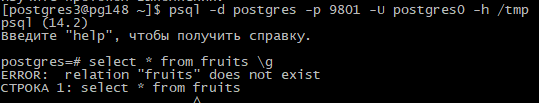


**3. Повреждение файлов БД**

Создадим в табличном пространстве new\_space новую таблицу и заполним данными на основном узле:



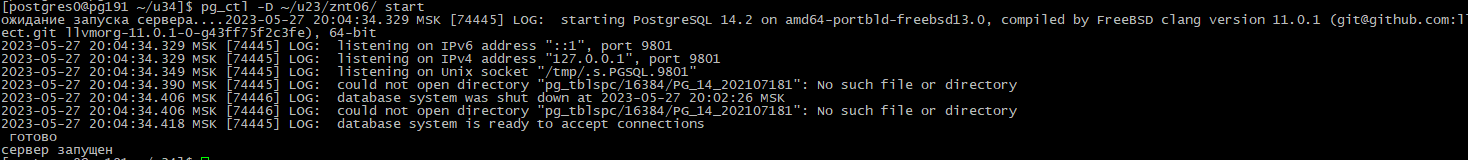
На резервном узле данных нет:



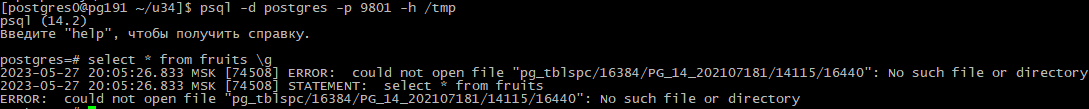
Удаляем директории табличного пространства new\_space:



Основной узел запустился:



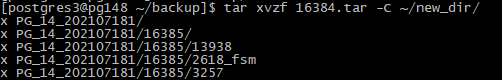
Но данные были утеряны:



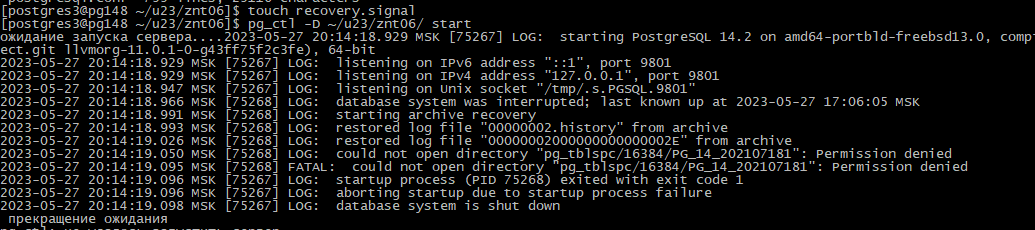
На резервном сервере заново разархивируем бекап и выдаем права на распакованные файлы:



Так как по условию директория табличного пространства недоступна , разархивируем архив с данными табличного пространства в другую директорию:



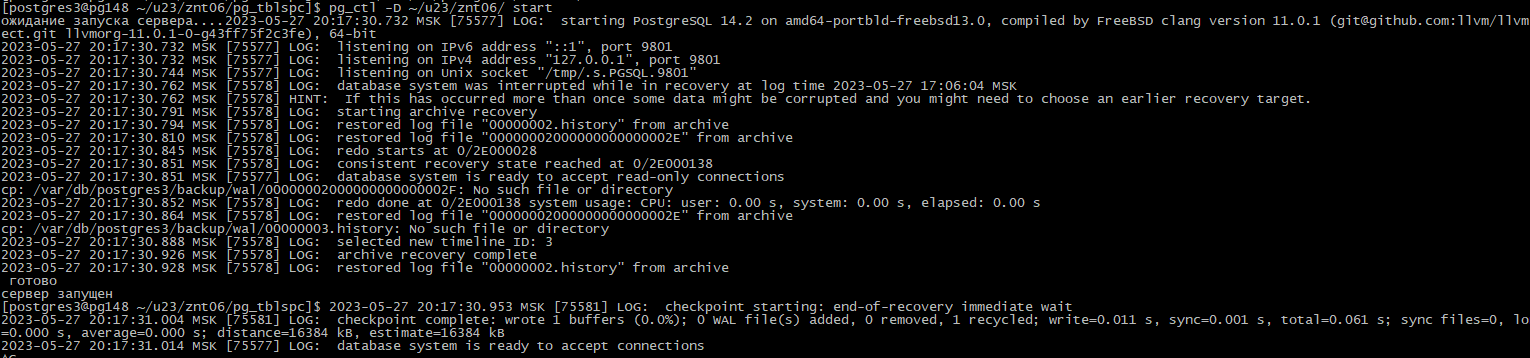
Создаем файл recovery.signal и пробуем восстановиться:



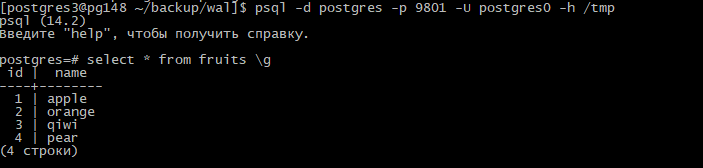
Получаем ошибку доступа к ссылке. Поскольку ссылка ссылается на старый путь табличного пространства ее нужно удалить и создать новую, которая ссылается на новую директорию табличного пространства:



Перезапускаем базу:



Проверяем данные:



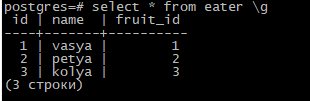
**4. Логическое повреждение данных**

По условию задания нужно изменить внешние ключи в таблицах. В нашей базе такие таблици отсутствуют , поэтому сначала их нужно создать и сделать бекап на резервном узле:



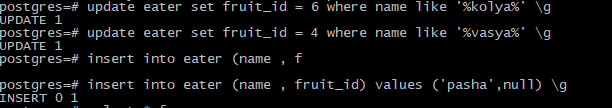


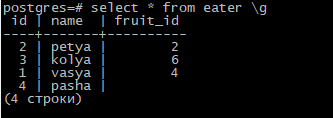
Вот так выглядят консистентные актуальные данные до ошибки:



На резервном узле устанавливаем recovery\_target\_time на минуту раньше чем произошло изменение и восстанавливаемся из текущих WAL.

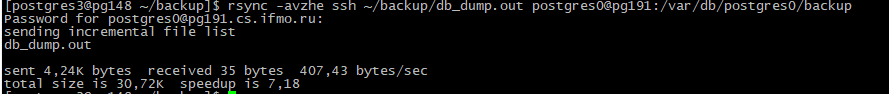
Симулируем ошибку. Были внесены изменния во внешние ключи и была добавлена без ключа:



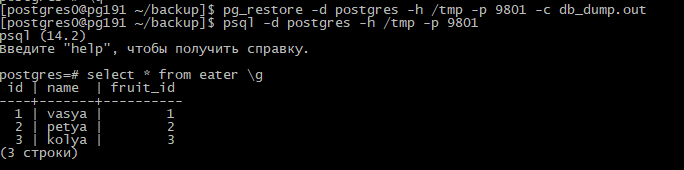


Создадим дамп базы на резервном узле и запишем в файл. Файл отправим на основной узел:





На основном узле восстанавливаем базу через утилиту pg\_restore:



Флаг –с очищает базу , перед тем как накатить содержимое файла.