МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский университет ИТМО»

##### ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

## ЛАБОРАТОРНАЯ РАБОТА №3

### по дисциплине

### «РАСПРЕДЕЛЕННЫЕ СИСТЕМЫ ХРАНЕНИЯ ДАННЫХ»

### Вариант № 444

##### ***Выполнил:*** Студент группы P3319

##### Билобрам Денис Андреевич

#### Преподаватель:

##### Шибаев Семён Сергеевич

Санкт-Петербург, 2025

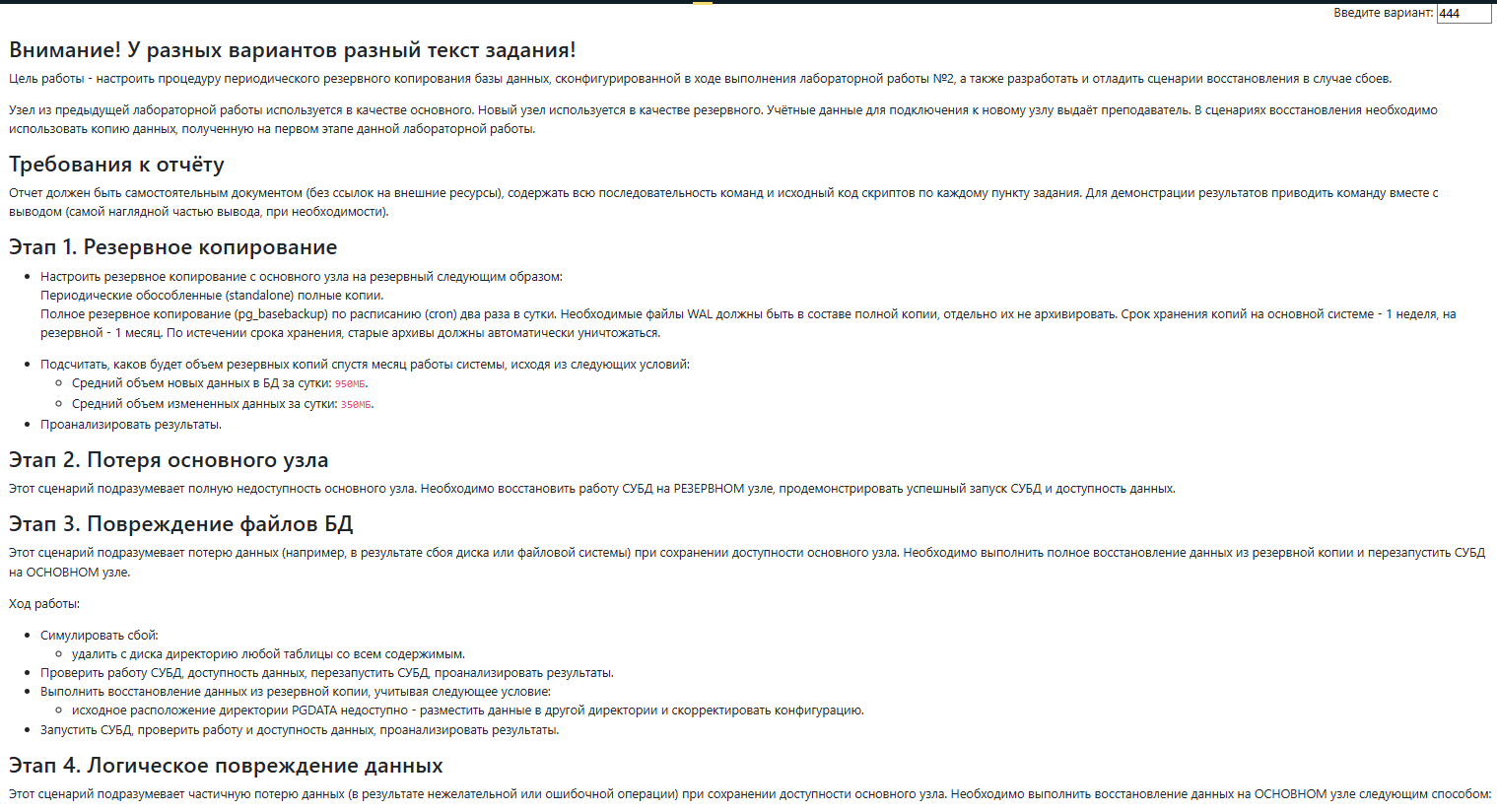
# Cодержание

[Задание: 3](#_Toc191644366)

[Выполнение: 3](#_Toc191644367)

[Вывод: 5](#_Toc191644368)

# Задание:



# Выполнение:

**Этап 1 – Резервное копирование**

1. **Создание директорий для резервных копий**

На основном узле создаем директорию для локальных резервных копий:

mkdir -p /var/db/postgres6/backups

chmod 700 /var/db/postgres6/backups

На резервном узле создаем директорию для хранения копий:

mkdir -p /var/db/postgres4/backups

chmod 700 /var/db/postgres4/backups

1. **Создание скрипта для резервного копирования**

#!/bin/bash

# Установка переменных

PRIMARY\_HOST="pg194"

BACKUP\_HOST="pg196"

PRIMARY\_USER="postgres6"

BACKUP\_USER="postgres4"

PGDATA="/var/db/${PRIMARY\_USER}/bcq76"

BACKUP\_DIR="/var/db/${PRIMARY\_USER}/backups/backup\_$(date +%Y%m%d\_%H%M%S)"

REMOTE\_BACKUP\_DIR="/var/db/${BACKUP\_USER}/backups"

# Создание директории для резервной копии

mkdir -p $BACKUP\_DIR

cd $BACKUP\_DIR

# Выполнение pg\_basebackup

pg\_basebackup --format=tar --wal-method=fetch -h localhost -p 9798 -U ${PRIMARY\_USER} -D $BACKUP\_DIR

# Передача архивов на резервный узел

scp -r $BACKUP\_DIR ${BACKUP\_USER}@${BACKUP\_HOST}:$REMOTE\_BACKUP\_DIR/

# Удаление старых резервных копий на основном узле (старше 7 дней)

find /var/db/${PRIMARY\_USER}/backups -type d -mtime +7 -exec rm -rf {} \;

1. **Настройка cron для запуска скрипта**

crontab -e

0 0,12 \* \* \* backup.sh

1. **Управление хранением на резервном узле:**

Создаем скрипт clean\_backups.sh на резервном узле для удаления копий старше 30 дней:

#!/bin/bash

BACKUP\_DIR="/var/db/postgres4/backups"

find $BACKUP\_DIR -type f -mtime +30 -exec rm {} \;

Настраиваем cron для ежедневного запуска:

crontab -e

0 0 \* \* \* clean\_backups.sh

1. **Расчет объема резервных копий:**

Условия:

* Средний объем новых данных за сутки: 950 МБ (0.95 ГБ).
* Средний объем измененных данных за сутки: 350 МБ (0.35 ГБ).

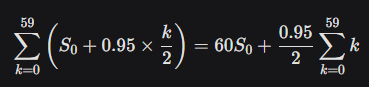
Предполагаем, что размер базы данных увеличивается на 950 МБ в сутки за счет новых данных.

За 30 дней размер базы увеличивается на:

950 МБ \* 30 = 28500 МБ = 27.8 ГБ

Поскольку резервное копирование выполняется дважды в сутки, за месяц создается 60 копий. Размер каждой копии соответствует размеру базы на момент создания. Для упрощения предполагаем линейный рост размера базы.

Общий размер данных в конце месяца можно вычислить по формуле:



Где S\_0 – это начальный размер базы данных.

Если принять S\_0 – 0, тогда объем будет 834 ГБ.

Анализ: Объем в 834 ГБ указывает на значительные требования к хранилищу из-за полных копий. Использование инкрементного копирования или WAL-архивирования могло бы значительно уменьшить объем.

**Этап 2 – Потеря основного узла**

Цель: Восстановить базу данных на резервном узле при полной недоступности основного узла, используя последнюю резервную копию, и продемонстрировать доступность данных.

Шаги восстановления:

1. Определение последней резервной копии

LATEST\_BACKUP=$(ls -d /var/db/postgres4/backups/backup\_\* | sort | tail -n 1)

1. Создание директории для восстановленных данных

RECOVERED\_DATA="/var/db/postgres4/recovered\_data"

mkdir -p $RECOVERED\_DATA

chmod 700 $RECOVERED\_DATA

1. Извлечение base.tar и обработка табличного пространства

tar -xvf $LATEST\_BACKUP/base.tar -C $RECOVERED\_DATA

mkdir -p /var/db/postgres4/fye37

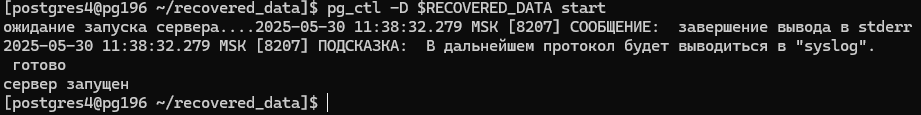
chmod 700 /var/db/postgres4/fye37

tar -xvf $LATEST\_BACKUP/16389.tar -C /var/db/postgres4/fye37

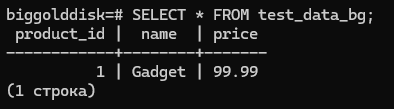
rm $RECOVERED\_DATA/pg\_tblspc/16389

ln -s /var/db/postgres4/fye37 $RECOVERED\_DATA/pg\_tblspc/16389

1. Запуск на резервном узле



1. Проверка доступности данных

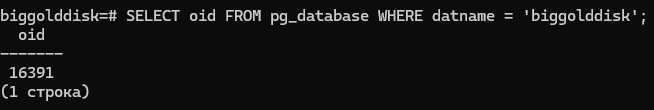


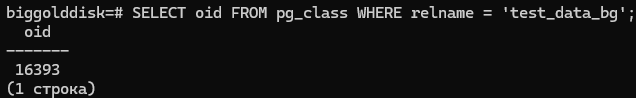
Анализ: Восстановление успешно, данные доступны. Корректировка путей для табличных пространств была необходима из-за различий в файловых системах узлов.

**Этап 3 – Повреждение файлов БД**

Цель: Смоделировать потерю данных на основном узле, восстановить из локальной резервной копии в новую директорию и проверить доступность данных.

Определение OID таблицы и базы данных:





Удаление директории таблицы

rm -rf /var/db/postgres6/bcq76/base/16391/16393\*

Проверка доступности данных



Восстановление:

Создание новой директории

RECOVERED\_DATA="/var/db/postgres6/recovered\_data"

mkdir -p $RECOVERED\_DATA

chmod 700 $RECOVERED\_DATA

Извлечение последней резервной копии

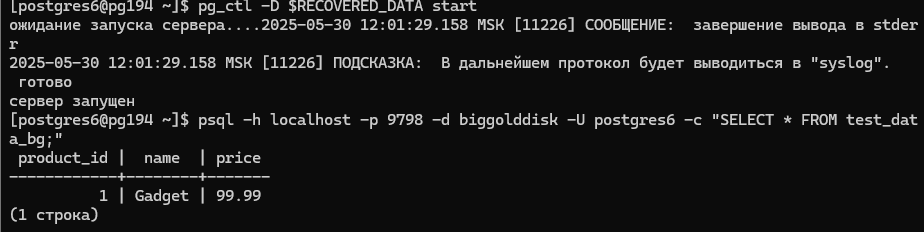
LATEST\_BACKUP=$(ls -d /var/db/postgres6/backups/backup\_\* | sort | tail -n 1)

tar -xvf $LATEST\_BACKUP/base.tar -C $RECOVERED\_DATA

rm -rf /var/db/postgres6/fye37/\*

tar -xvf $LATEST\_BACKUP/16389.tar -C /var/db/postgres6/fye37

Проверка доступности данных



Анализ: Восстановление успешно, данные доступны. Использование новой директории позволило обойти поврежденную PGDATA.

**Этап 4 – Повреждение файлов БД**

Цель: Смоделировать логическое повреждение данных путем удаления таблиц, восстановить данные с помощью pg\_dump с резервного узла и проверить результат.

Удалим таблицу на основном узле:

biggolddisk=# DROP TABLE test\_data\_bg;

Теперь восстановим базу данных с резервного узла:

Создание дампа на резервном узле

pg\_dump -h localhost -p 9799 -U postgres6 biggolddisk > /var/db/postgres4/dump.sql

Передача дампа на основной узел

scp /var/db/postgres4/dump.sql postgres6@pg194:

Восстановление на основном узле

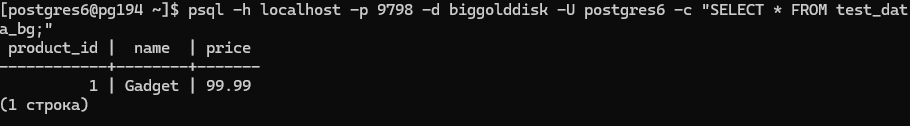
dropdb -h localhost -p 9798 -U postgres6 biggolddisk

createdb -h localhost -p 9798 -U postgres6 biggolddisk

psql -h localhost -p 9798 -U postgres6 biggolddisk < /var/db/postgres6/dump.sql



Проверка восстановления:



Анализ: Данные восстановлены до состояния последней резервной копии.

# Вывод:

В ходе лабораторной работы были успешно настроены процедуры резервного копирования с использованием pg\_basebackup и разработаны сценарии восстановления для трех типов сбоев. Полученные навыки позволяют эффективно управлять резервным копированием и восстановлением данных в PostgreSQL, обеспечивая надежность системы.