**Министерство науки и высшего образования Российской Федерации**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Факультет безопасности информационных технологий**

**Дисциплина:**

«Информационная безопасность баз данных»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5**

«Резервирование БД и восстановление по контрольным точкам»

**Выполнили:**

Бардышев Артём Антонович,

студент группы N3346

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

**Проверил:**

Салихов Максим Русланович,

преподаватель, ФБИТ

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(отметка о выполнении)

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

Содержание

[Введение 3](#_Toc211089947)

[1 Подготовка 4](#_Toc211089948)

[1.1 Включить архивирование WAL 4](#_Toc211089949)

[1.2 Создаем папки для архивов и логов 4](#_Toc211089950)

[1.3 Применяем настройки 4](#_Toc211089951)

[2 Ночное резервирование (по расписанию) 5](#_Toc211089952)

[2.1 Скрипт base-backup 5](#_Toc211089953)

[2.2 Планировщик задач 5](#_Toc211089954)

[3 Создать контрольную точку восстановления и “сломать” данные 6](#_Toc211089955)

[3.1 Создаём restore-point 6](#_Toc211089956)

[3.2 Вносим случайные изменения 6](#_Toc211089957)

[4 Откат к контрольной точке 7](#_Toc211089958)

[4.1 Подготовим «чистое» восстановление 7](#_Toc211089959)

[4.2 Настроить восстановление из WAL и таргет-точку 7](#_Toc211089960)

[4.3 Старт 7](#_Toc211089961)

[4.4 Проверка «отката» 7](#_Toc211089962)

[Заключение 9](#_Toc211089963)

[Список использованных источников 10](#_Toc211089964)

Введение

Цель работы – Получение навыков по резервированию и восстановлению БД.

Задание

1. Создание резервной копии БД согласно выбранному расписанию.
2. Внести случайные изменения в таблицы созданной вами базы данных (изменения вносятся до момента создания контрольной точки).
3. Продемонстрировать процесс отката к последней контрольной точке. Откатите изменения, выполненные в пункте 2.

Проанализируйте возможность анализа/просмотра изменений, которые были «откачены», с помощью системы логирования СУБД (в том числе сделанной ЛР 3) или с помощью средств системы резервирования

# Подготовка

## Включить архивирование WAL

Открываем postgresql.conf (по умолчанию: C:\Program Files\PostgreSQL\18\data\postgresql.conf) и прописываем/проверяем:

wal\_level = replica

archive\_mode = on

archive\_command = 'cmd /c copy /Y "%p" "D:\\pg\_wal\_archive\\%f"'

logging\_collector = on

log\_directory = 'C:\Users\ububk\AppData' # ВНЕ data каталога — чтобы логи не «откатились»

log\_statement = 'mod' # Логировать INSERT/UPDATE/DELETE

## Создаем папки для архивов и логов

New-Item -ItemType Directory -Force -Path C:\Users\ububk\AppData\pg\_wal\_archive| Out-Null

New-Item -ItemType Directory -Force -Path C:\Users\ububk\AppData\pg\_logs|Out-Null

## Применяем настройки

Перезапускаем PostgreSQL (через Services или PowerShell от админа):

net stop postgresql-x64-18

net start postgresql-x64-18

Проверим, что WAL-архивы «сыпятся»:

psql -U postgres -d apple\_store -c "SELECT pg\_switch\_wal();"

dir C:\Users\ububk\AppData\pg\_wal\_archive

Должны появиться новые файлы в C:\Users\ububk\AppData\pg\_wal\_archive

# Ночное резервирование (по расписанию)

## Скрипт base-backup

Создаем C:\Users\ububk\AppData\pg\_backup\_scripts\full\_backup.ps1:

C:\Users\ububk\AppData\pg\_backup\_scripts\full\_backup.ps1

$stamp = Get-Date -Format "yyyyMMdd\_HHmm"

$dest = "D:\pg\_backups\base\_$stamp"

New-Item -ItemType Directory -Force -Path $dest | Out-Null

# Архивируем кластер + стримим WAL

& "C:\Program Files\PostgreSQL\18\bin\pg\_basebackup.exe" `

-D $dest `

-Fp `

-X stream `

-P `

-U postgres `

-d "host=localhost dbname=postgres user=postgres" `

2>&1 | Tee-Object -FilePath "$dest\backup.log"

Подготовим папки:

New-Item -ItemType Directory -Force -Path C:\Users\ububk\AppData\pg\_backups, C:\Users\ububk\AppData\pg\_backup\_scripts | Out-Null

## Планировщик задач

Каждую ночь, например 03:00

$A = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "-NoProfile -ExecutionPolicy Bypass -File C:\Users\ububk\AppData\pg\_backup\_scripts\full\_backup.ps1"

$T = New-ScheduledTaskTrigger -Daily -At 3:00AM

$S = New-ScheduledTaskSettingsSet -Compatibility Win8

Register-ScheduledTask -TaskName "PostgresFullBackupNightly" -Action $A -Trigger $T -Settings $S -Description "Nightly base backup PostgreSQL"

Это закрывает требование «резервная копия по расписанию».

# Создать контрольную точку восстановления и “сломать” данные

## Создаём restore-point

psql -U postgres -d apple\_store -c "SELECT pg\_create\_restore\_point('lab5\_before\_changes');"

## Вносим случайные изменения

psql -U postgres -d apple\_store -c "INSERT INTO product(title, price, stock) VALUES ('FAKE\_ITEM', 999999, 1);"

psql -U postgres -d apple\_store -c "UPDATE orders SET status='cancelled' WHERE orderid IN (SELECT orderid FROM orders ORDER BY orderdate DESC LIMIT 2);"

psql -U postgres -d apple\_store -c "DELETE FROM payments WHERE status='refunded' AND paid\_at < now() - interval '180 days';"

# Откат к контрольной точке

## Подготовим «чистое» восстановление

Останавливаем PostgreSQL:

net stop postgresql-x64-16

**ВАЖНО:** сохранить текущий data на всякий случай:

Rename-Item "C:\Program Files\PostgreSQL\16\data" "C:\Program Files\PostgreSQL\16\data\_bad\_$(Get-Date -Format yyyyMMdd\_HHmm)"

Возьмем **последний успешный base-backup** (например D:\pg\_backups\base\_20251011\_0300) и скопируем его как новый data:

Copy-Item "D:\pg\_backups\base\_YYYYMMDD\_HHMM\\*" "C:\Program Files\PostgreSQL\18\data" -Recurse

## Настроить восстановление из WAL и таргет-точку

В кластере ≥ PostgreSQL 12 для PITR создаём recovery.signal и прописываем параметры в postgresql.conf.

Добавляем в C:\Program Files\PostgreSQL\18\data\postgresql.conf:

restore\_command = 'cmd /c copy "D:\\pg\_wal\_archive\\%f" "%p"'

recovery\_target\_name = 'lab5\_before\_changes' # наша restore-point метка

recovery\_target\_action = 'promote'

Создай пустой файл-флажок:

New-Item -ItemType File "C:\Program Files\PostgreSQL\16\data\recovery.signal" | Out-Null

## Старт

net start postgresql-x64-16

Сервер поднимется, «накрутит» WAL-журналы до **restore-point** lab5\_before\_changes и автоматически **promote**-нётся

## Проверка «отката»

Проверяем, что наш мусор исчез:

psql -U postgres -d apple\_store -c "SELECT \* FROM product WHERE title='FAKE\_ITEM';"

psql -U postgres -d apple\_store -c "SELECT orderid,status FROM orders ORDER BY orderdate DESC LIMIT 5;"

psql -U postgres -d apple\_store -c "SELECT \* FROM payments WHERE status='refunded' AND paid\_at < now() - interval '180 days';"

Заключение

В рамках лабораторной работы №5 была реализована система резервного копирования и восстановления базы данных **apple\_store** средствами PostgreSQL.  
Настроено **архивирование журналов транзакций (WAL)**, регулярное **создание резервных копий (base-backup)** и выполнение **восстановления по контрольной точке (Point-in-Time Recovery, PITR)**.

Список использованных источников

1. Новиков Б. А., Горшкова Е. А., Графеева Н. Г. **Основы технологий баз данных.** – 2-е изд. – М.: ДМК Пресс, 2020. – 582 с.
2. Хомоненко А. Д. (ред.). **Базы данных.** – 6-е изд., доп. – СПб.: КОРОНА-Век, 2009. – 736 с.
3. Документация PostgreSQL: <https://www.postgresql.org/docs/>