

на робота 11. Створення користувачів та права доступу. Резервне копіювання та відновлення.

O.B., AI-243

Мета: Навчити студентів створювати користувачів і ролі, а також надавати їм права доступу до різних об'єктів бази даних у PostgreSQL. Ознайомити студентів з основними підходами до резервного копіювання та відновлення даних у PostgreSQL, іхньою практичною реалізацією та особливостями в різних сценаріях.

Завдання:

1. Ознайомитися з мовою Data Control Language;
2. Виконати SQL-оператори для створення користувачів та ролей;
 3. Виконати SQL-оператори для надання прав доступу для відповідних користувачів та ролей;
 4. Виконати SQL-оператори для зняття прав доступу для відповідних користувачів та ролей;
5. Виконати SQL-оператор(-и) для створення резервної копії бази даних;
6. Виконати SQL-оператор(-и) для відновлення копії бази даних.

Результат:

Студенти повинні подати SQL-скрипти, що відображають створення користувачів, ролей та надання їм різних прав доступу до різних об'єктів згідно завдання та предметної області, їх опис, а також звіт з результатами тестування. А також SQL-скрипти створення резервної копії бази даних та її відновлення.

1. Створіть трьох користувачів.

словесна постановка задачі, що вирішується:

Надати право першому користувачу (Адміністратору "i.petrenko") на зміну декількох стовпців (назва, ціна) у конкретній таблиці (Страва). Це необхідно для обмеження прав оновлення лише ключовими полями.

SQL-код рішення:

```
CREATE ROLE "I.petrenko" LOGIN PASSWORD 'pass_petrenko' CREATEROLE;
CREATE ROLE "M.ivanova" LOGIN PASSWORD 'pass_ivanova';
CREATE ROLE "A.sydorenko" LOGIN PASSWORD 'pass_sydorenko';
```

скриншот отриманого результату:

The screenshot shows the pgAdmin 4 interface with three tabs open: 'restaurant_bd/pos...' (closed), 'restaurant_bd/pos...' (closed), and 'restaurant_bd/postgres@Local PostgreSQL'. The central pane is titled 'Query' and contains the following SQL code:

```
1 CREATE ROLE "I.petrenko" LOGIN PASSWORD 'pass_petrenko' CREATEROLE;
2 CREATE ROLE "M.ivanova" LOGIN PASSWORD 'pass_ivanova';
3 CREATE ROLE "A.sydorenko" LOGIN PASSWORD 'pass_sydorenko';
```

Below the query pane, the 'Data Output' tab is selected, showing the results of the query:

```
CREATE ROLE
Query returned successfully in 99 msec.
```

2. Надайте право першому користувачу на зміну декількох стовпців будь-якої таблиці.

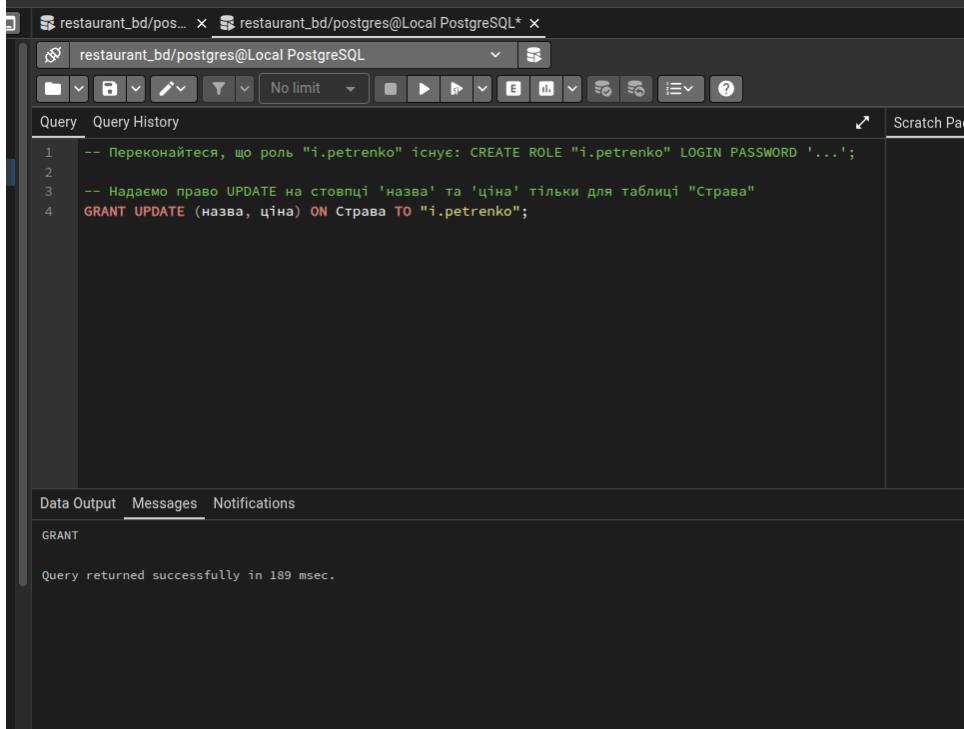
словесна постановка задачі, що вирішується:

Надати право першому користувачу (Адміністратору "i.petrenko") на зміну (UPDATE) декількох стовпців (назва, ціна) у конкретній таблиці (Страва). Це реалізує принцип найменших привілеїв, обмежуючи адміністратора лише тими полями, які він має право коригувати.

SQL-код рішення:

```
-- Переконайтесь, що роль "i.petrenko" існує: CREATE ROLE "i.petrenko" LOGIN  
PASSWORD '...';  
  
-- Надаємо право UPDATE на стовпці 'назва' та 'ціна' тільки для таблиці  
"Страва"  
GRANT UPDATE (назва, ціна) ON Страва TO "i.petrenko";
```

скриншот отриманого результату:



The screenshot shows a pgAdmin 4 interface with two tabs open: 'restaurant_bd/pos...' and 'restaurant_bd/postgres@Local PostgreSQL'. The query tab contains the four lines of SQL code provided above. Below the code, the 'Messages' tab is active, showing the word 'GRANT' in green, indicating the command was successfully executed. A message at the bottom states 'Query returned successfully in 189 msec.'

3. Надайте право всім користувачам системи на перегляд будь-якої таблиці.

словесна постановка задачі, що вирішується:

Надати право перегляду (SELECT) для всіх користувачів системи (PUBLIC) на всі існуючі таблиці у публічній схемі. Це необхідно для забезпечення базової прозорості даних (для читання) для всіх, хто може підключитися до бази даних.

SQL-код рішення:

```
-- Переконайтесь, що роль "i.petrenko" існує: CREATE ROLE "i.petrenko" LOGIN  
PASSWORD '...';  
  
-- Надаємо право UPDATE на стовпці 'назва' та 'ціна' тільки для таблиці "Страва"  
GRANT UPDATE (назва, ціна) ON Страва TO PUBLIC;
```

скриншот отриманого результату:

The screenshot shows a pgAdmin 4 interface with three tabs open. The current tab is 'restaurant_bd/postgres@Local PostgreSQL'. In the query editor, two lines of SQL code are shown:

```
-- Надаємо право SELECT (перегляд) для всіх існуючих таблиць у схемі public групі PUBLIC (всі крім системних)
GRANT SELECT ON ALL TABLES IN SCHEMA public TO PUBLIC;
```

Below the query, the output pane shows:

```
GRANT
Query returned successfully in 113 msec.
```

4. Надайте право другому користувачу вставляти або модифікувати значення таблиці з правом передавати іншим користувачам вказані права.

словесна постановка задачі, що вирішується:

Надати другому користувачу (Офіціант "m.ivanova") права вставляти (INSERT) та модифікувати (UPDATE) значення у таблиці Замовлення, а також право передавати ці привілеї іншим користувачам (за допомогою опції WITH GRANT OPTION).

SQL-код рішення:

```
-- Переконайтесь, що роль "m.ivanova" існує: CREATE ROLE "m.ivanova"
LOGIN PASSWORD '...';

-- Надаємо право INSERT та UPDATE на таблицю "Замовлення"
-- з правом передачі цих привілеїв (WITH GRANT OPTION)
GRANT INSERT, UPDATE ON Замовлення TO "m.ivanova" WITH GRANT OPTION;
```

скриншот отриманого результату:

The screenshot shows a pgAdmin 4 interface with three tabs open. The current tab is 'restaurant_bd/postgres@Local PostgreSQL'. In the query editor, five lines of SQL code are shown:

```
-- Переконайтесь, що роль "m.ivanova" існує: CREATE ROLE "m.ivanova" LOGIN PASSWORD '...';
-- Надаємо право INSERT та UPDATE на таблицю "Замовлення"
-- з правом передачі цих привілеїв (WITH GRANT OPTION)
GRANT INSERT, UPDATE ON Замовлення TO "m.ivanova" WITH GRANT OPTION;
```

Below the query, the output pane shows:

```
GRANT
Query returned successfully in 110 msec.
```

5. Надайте третьому користувачу права адміністратора (всі права на всі таблиці).

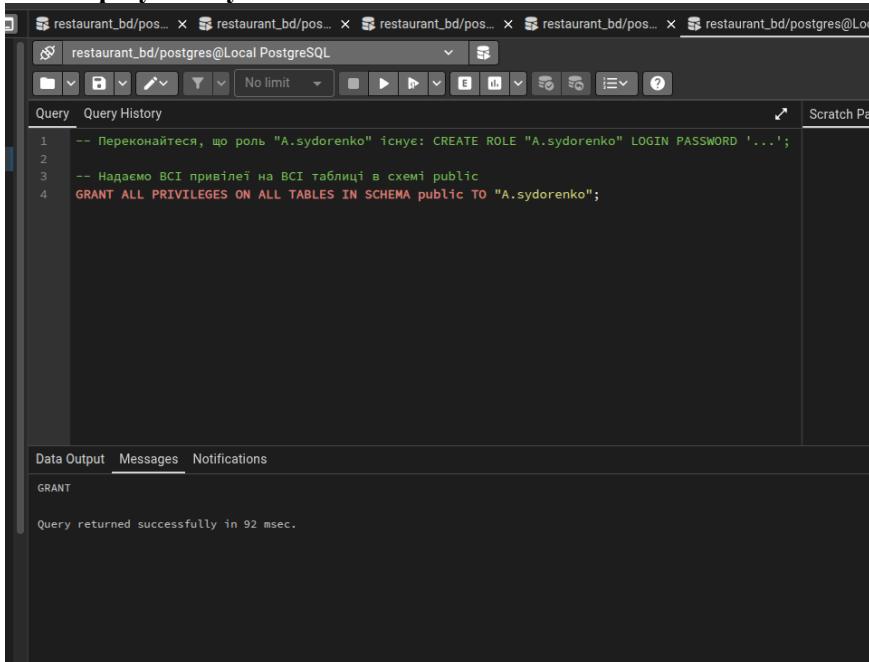
словесна постановка задачі, що вирішується:

Надати третьому користувачу (Кухар "A.sydorenko") всі можливі привілеї (ALL PRIVILEGES) на всі табличі у публічній схемі. Це дозволяє йому виконувати будь-які операції (SELECT, INSERT, UPDATE, DELETE) з даними.

SQL-код рішення:

```
-- Переконайтесь, що роль "A.sydorenko" існує: CREATE ROLE "A.sydorenko"  
LOGIN PASSWORD '...';  
  
-- Надаємо ВСІ привілеї на ВСІ табличі в схемі public  
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO "A.sydorenko";
```

скриншот отриманого результату:



The screenshot shows a pgAdmin 4 interface with a single query window titled 'restaurant_bd/postgres@Local PostgreSQL'. The query is:

```
-- Переконайтесь, що роль "A.sydorenko" існує: CREATE ROLE "A.sydorenko" LOGIN PASSWORD '...';  
-- Надаємо ВСІ привілеї на ВСІ табличі в схемі public  
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO "A.sydorenko";
```

The results pane shows the output:

```
GRANT  
Query returned successfully in 92 msec.
```

6. Зніміть привілей INSERT для третього користувача для будь-якої таблиці.

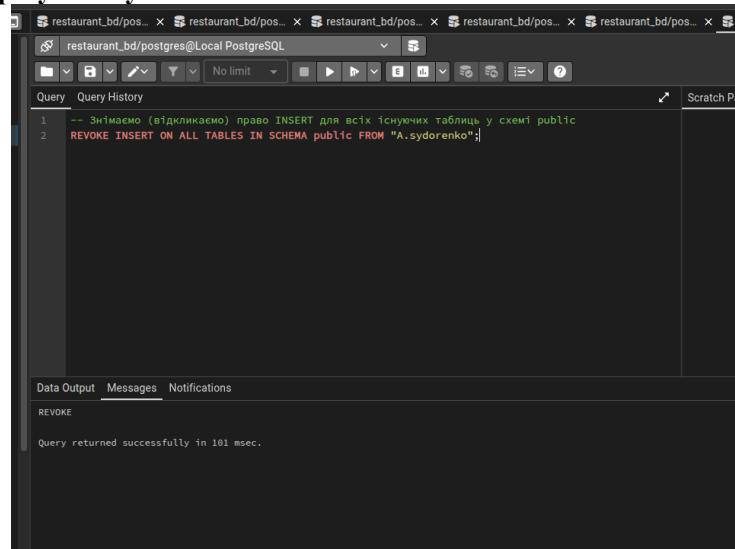
словесна постановка задачі, що вирішується:

Зняти привілей вставляти нові рядки (INSERT) для користувача "A.sydorenko" на всіх табличах у публічній схемі. Це демонструє, що навіть після надання всіх прав (ALL PRIVILEGES), адміністратор може вибірково відкликати певні привілеї.

SQL-код рішення:

```
-- Знімаємо (відкликаємо) право INSERT для всіх існуючих таблиць у схемі public  
REVOKE INSERT ON ALL TABLES IN SCHEMA public FROM "A.sydorenko";
```

скриншот отриманого результату:



The screenshot shows a pgAdmin 4 interface with a single query window titled 'restaurant_bd/postgres@Local PostgreSQL'. The query is:

```
-- Знімаємо (відкликаємо) право INSERT для всіх існуючих таблиць у схемі public  
REVOKE INSERT ON ALL TABLES IN SCHEMA public FROM "A.sydorenko";
```

The results pane shows the output:

```
REVOKE  
Query returned successfully in 101 msec.
```

7. Надайте право першому користувачу на доступ тільки до певних рядків будь-якої таблиці.

словесна постановка задачі, що вирішується:

Надати право першому користувачу (Адміністратору "i.petrenko") на доступ (перегляд та модифікацію) тільки до певних рядків таблиці Замовлення — а саме, до тих, що мають статус_замовлення = 'нове'. Це реалізується через Політику безпеки на рівні рядків (RLS).

SQL-код рішення:

```
ALTER TABLE Замовлення ENABLE ROW LEVEL SECURITY;
```

```
CREATE POLICY new_orders_policy ON Замовлення
```

```
AS PERMISSIVE
```

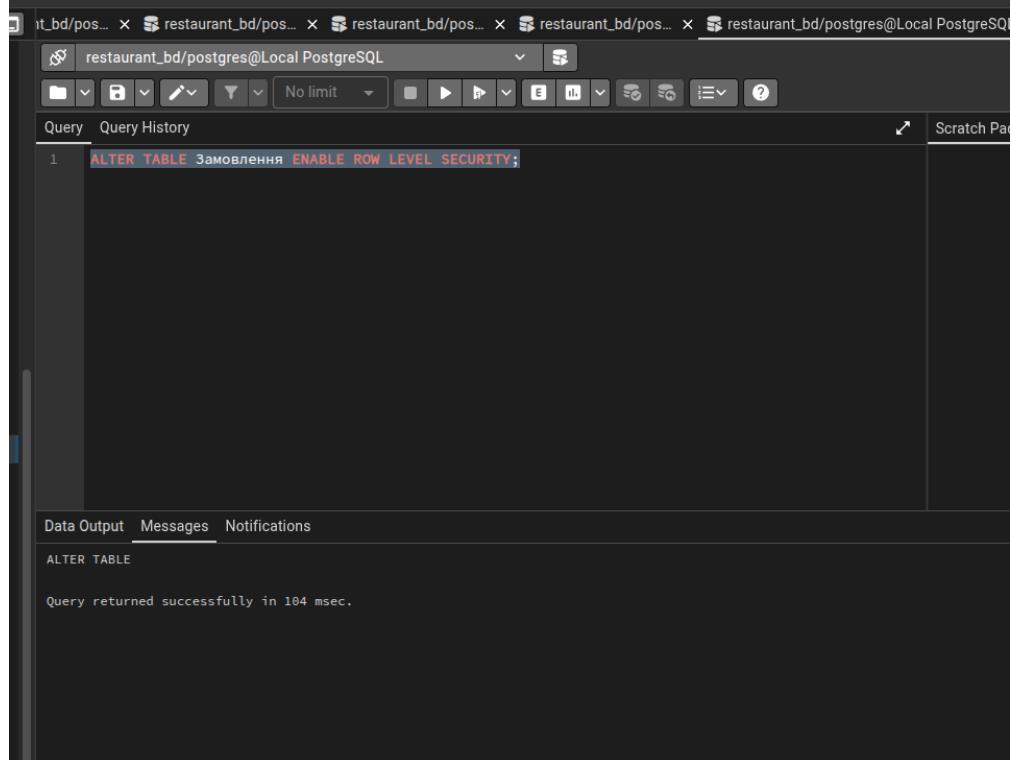
```
FOR ALL
```

```
TO "i.petrenko"
```

```
USING (статус = 'нове')
```

```
WITH CHECK (статус = 'нове');
```

скриншот отриманого результату:



The screenshot shows a pgAdmin 4 interface with a single query window. The query is:

```
ALTER TABLE Замовлення ENABLE ROW LEVEL SECURITY;
```

The results pane at the bottom shows the message:

```
ALTER TABLE
Query returned successfully in 104 msec.
```

```

1 CREATE POLICY new_orders_policy ON Замовлення
2
3 AS PERMISSIVE
4
5 FOR ALL
6
7 TO "i.petrenko"
8
9 USING (статус = 'нове')
10
11 WITH CHECK (статус = 'нове');

```

Data Output Messages Notifications

CREATE POLICY

Query returned successfully in 98 msec.

8. Розподіліть інші привілеї на свій розсуд.

словесна постановка задачі, що вирішується:

Розподілити додаткові привілеї між користувачами:

1. Надати офіціанту (m.ivanova) право видаляти рядки з таблиці Деталі_замовлення (для скасування позиції).
2. Надати кухарю (A.sydorenko) право використовувати послідовності, пов'язані з його робочими таблицями (Склад_страви).

SQL-код рішення:

GRANT DELETE ON Деталі_замовлення TO "m.ivanova";

GRANT USAGE ON ALL SEQUENCES IN SCHEMA public TO "A.sydorenko";

скриншот отриманого результату:

```

1 GRANT DELETE ON деталі_замовлення TO "m.ivanova";

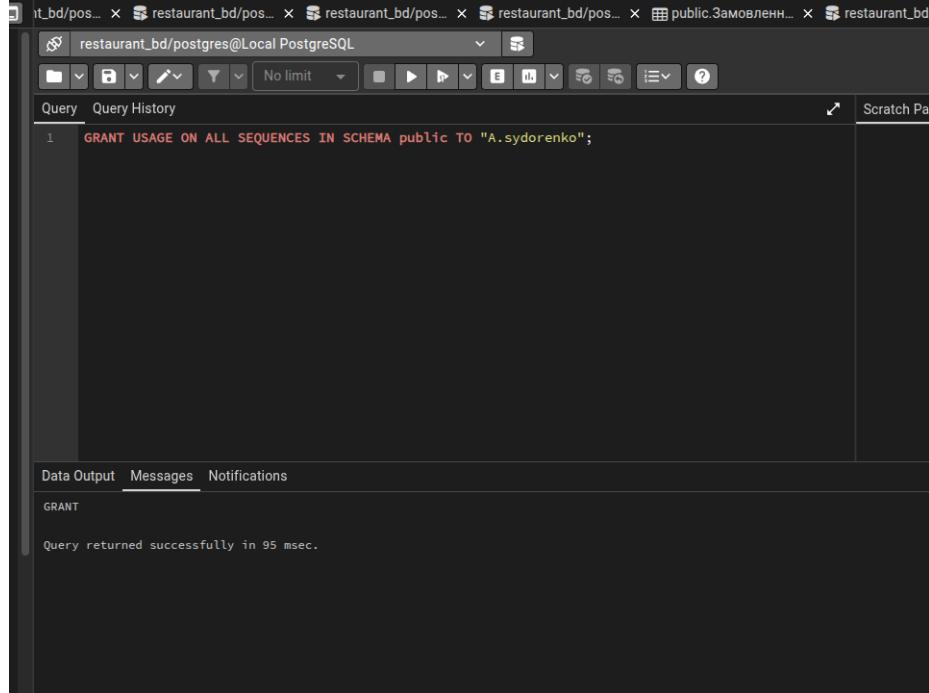
```

Data Output Messages Notifications

GRANT

Query returned successfully in 100 msec.

9. Виконайте логічне резервне копіювання бази даних, використовуючи відповідну



The screenshot shows a pgAdmin 4 interface with multiple tabs open. The active tab is 'restaurant_bd/postgres@Local PostgreSQL'. In the main pane, there is a single query listed:

```
GRANT USAGE ON ALL SEQUENCES IN SCHEMA public TO "A.sydorenko";
```

Below the query, the output shows:

```
GRANT
Query returned successfully in 95 msec.
```

утиліту.

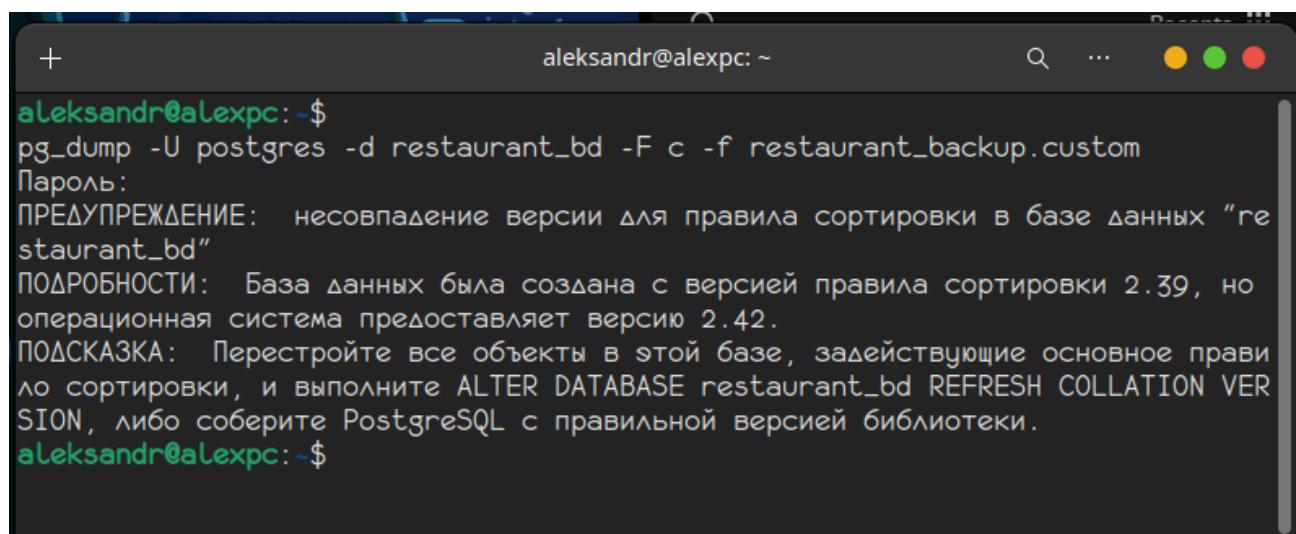
словесна постановка задачі, що вирішується:

Створити повну логічну резервну копію бази даних restaurant_bd (включно зі структурою та даними) у форматі SQL-скрипту.

SQL-код рішення:

```
pg_dump -U postgres -d restaurant_bd -F c -f restaurant_backup.custom
```

скриншот отриманого результату:



The terminal window shows the command being run:

```
aleksandr@alexp:~$ pg_dump -U postgres -d restaurant_bd -F c -f restaurant_backup.custom
```

Followed by a password prompt:

```
Пароль:
```

Then several informational messages:

```
ПРЕДУПРЕЖДЕНИЕ: несовпадение версии для правила сортировки в базе данных "ge staurant_bd"
ПОДРОБНОСТИ: База данных была создана с версией правила сортировки 2.39, но операционная система предоставляет версию 2.42.
ПОДСКАЗКА: Перестройте все объекты в этой базе, задействующие основное правило сортировки, и выполните ALTER DATABASE restaurant_bd REFRESH COLLATION VERSION, либо соберите PostgreSQL с правильной версией библиотеки.
```

```
aleksandr@alexp:~$
```

10. Виконайте відновлення з логічного резервного копіювання бази даних.

словесна постановка задачі, що вирішується:

Відновити дані зі створеної логічної резервної копії (restaurant_backup.custom) до нової, порожньої бази даних (restaurant_bd_new), використовуючи утиліту pg_restore.

SQL-код рішення:

```
pg_restore -U postgres -d restaurant_bd_new -F c
restaurant_backup.custom
```

скриншот отриманого результату:

```
postgres=# \q
aleksandr@alexp: $ pg_restore -U postgres -d restaurant_bd_new -F c restaurant_backup.custom
Пароль:
aleksandr@alexp: $
```

11. Виконайте фізичне резервне копіювання бази даних.

словесна постановка задачі, що вирішується:

Виконати повне фізичне копіювання кластера бази даних (всі файли даних) у plain форматі (-F p), включаючи журнали WAL, використовуючи утиліту pg_basebackup.

SQL-код рішення:

```
pg_basebackup -U postgres -D /tmp/restaurant_physical_backup -F p -X
stream -P
```

скриншот отриманого результату:

```
aleksandr@alexp: $ pg_basebackup -U postgres -D /tmp/restaurant_physical_backup -F p -X stream -P
Пароль:
48177/48177 КБ (100%), табличне пространство 1/1
aleksandr@alexp: $ |
```

12. Виконайте відновлення з фізичного резервного копіювання бази даних.

словесна постановка задачі, що вирішується:

Виконати відновлення кластера бази даних до стану на момент створення фізичної копії, використовуючи резервний каталог, створений у Завданні 11 (pg_basebackup), з подальшим застосуванням WAL-журналів (Write-Ahead Logs) при запуску сервера.

SQL-код рішення:

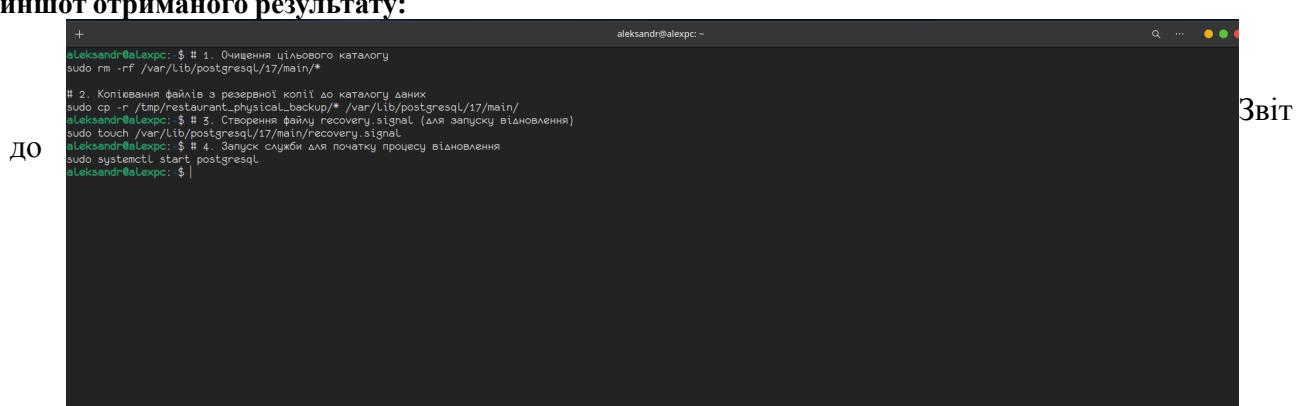
```
# 1. Зупинка сервера
sudo systemctl stop postgresql

# 2. Очищення та копіювання файлів (ЗМІНІТЬ ШЛЯХ 17/main/ НА ВАШ PG_DATA)
sudo rm -rf /var/lib/postgresql/17/main/*
sudo cp -r /tmp/restaurant_physical_backup/* /var/lib/postgresql/17/main/

# 3. Створення файлу-тригера відновлення
sudo touch /var/lib/postgresql/17/main/recovery.signal

# 4. Запуск сервера
sudo systemctl start postgresql
```

скриншот отриманого результату:



лабораторної роботи 11 можна здати онлайн на сайті ДО edu.op.edu.ua до початку вашого заняття.