



Національний технічний університет України  
«Київський політехнічний інститут»

Факультет Прикладної Математики  
Кафедра Системного Програмування і Спеціалізованих  
Комп'ютерних Систем

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА  
*з дисципліни*

«Бази даних та засоби управління»

ТЕМА: «Створення додатку бази даних,  
орієнтованого на взаємодію з СУБД PostgreSQL»

Група: КВ-11

Виконав: Брюханов О.

Оцінка: \_\_\_\_

Київ – 2023

Система управління відносинами зі спонсорами та донорами	
GitHub	Telegram
<a href="https://github.com/march07th/db_labs">https://github.com/march07th/db_labs</a>	<a href="https://id155.t.me/">https://id155.t.me/</a>

**Мета роботи:** здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

#### Завдання на розрахунково-графічну роботу:

1. Реалізувати функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сущностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

# Опис бази даних

## Модель

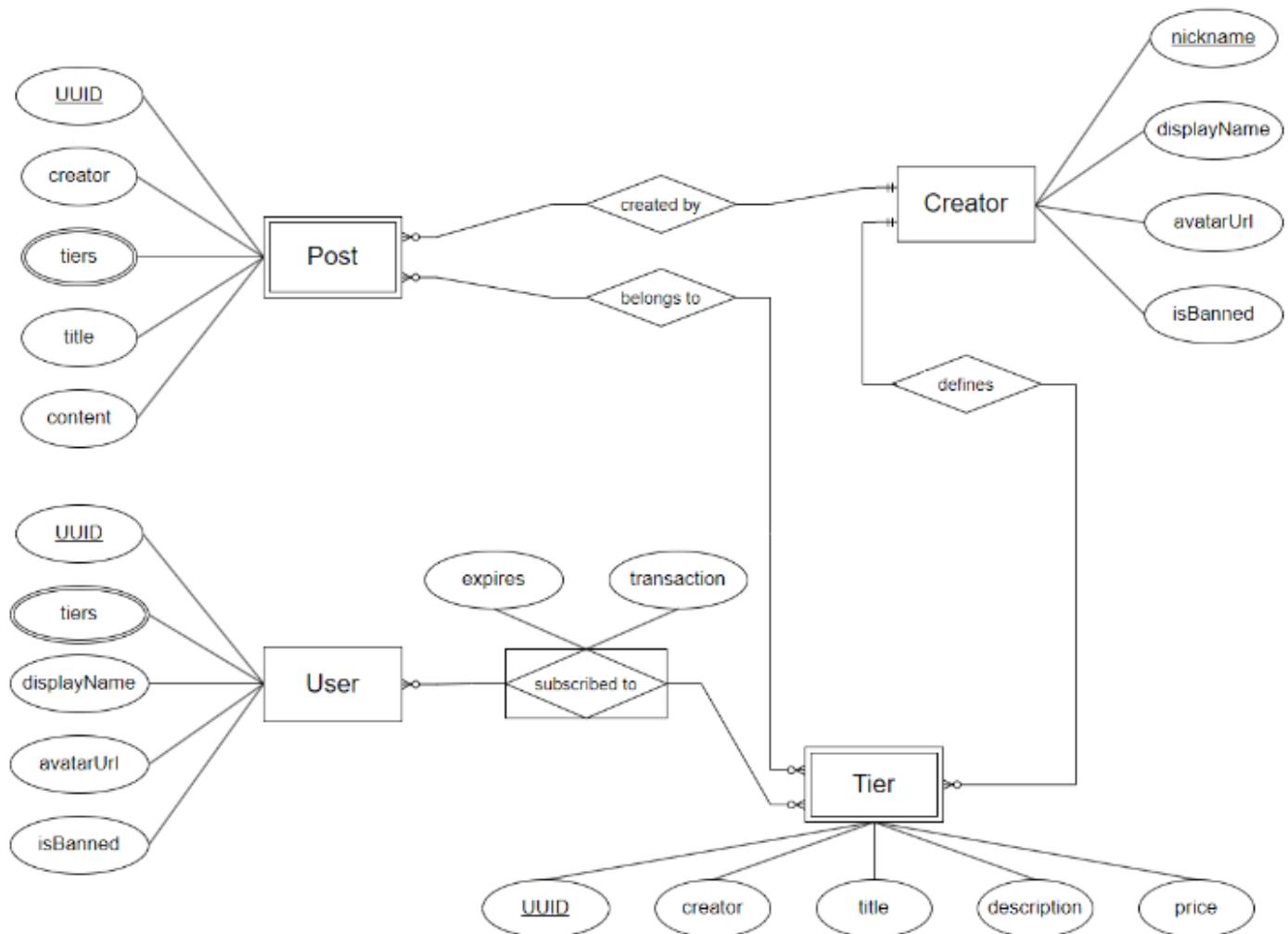


Рис 1. Модель в нотації Чена-Пташиної лапки

Для моделі було зроблено 5 сутностей, з яких дві – слабкі, одна об'єднуюча:

1. Creator
  - ☞ nickname, displayName, avatarUrl, isBanned
2. Tier
  - ☞ UUID, creator, title, description, price
3. Post
  - ☞ UUID, creator, tiers, title, content
4. User
  - ☞ UUID, tiers, displayName, avatarUrl, isBanned
5. [subscribed to]
  - ☞ expires, transaction

## Опис сутностей

<b>Creator</b>	<b>User</b>
Сутність для представлення користувача, який публікує пости для взаємодії зі своїми спонсорами. Такі користувачі також визначають рівні підписок спонсорів. Кожен такий користувач має унікальний нікнейм (nickname), який однозначно його ідентифікує, назву профілю (displayName), URL аватарки (avatarUrl) та флагок бану (isBanned).	Сутність для представлення користувача, який може виступати спонсором для користувачів-контентмейкерів (сутність Creator). Як і Creators, ці користувачі мають назву профілю (displayName), URL аватарки (avatarUrl) та флагок бану (isBanned). Але на відміну від них, вони ідентифікуються не нікнеймом, а унікальним ідентифікатором (UUID) та мають підписки/спонсорства (tiers).
<b>Tier</b>	<b>Post</b>
Сутність для представлення рівня підписки, до яких будуть прикріплятися пости. Користувачі-спонсори мають бачити тільки ті пости, які входять в рівень, на який вони підписані. Ця сутність ідентифікається унікальним ідентифікатором (UUID), містить інформацію про те, хто її створив (creator), назву (title), опис (description) та ціну (price).	Сутність для представлення посту. Пости створюють кріейтори та публікують, прикріплюючи до рівня підписки, користувачі-спонсори мають бачити тільки ті пости, які входять в рівень, на який вони підписані. Ця сутність ідентифікається унікальним ідентифікатором (UUID), містить інформацію про те, хто її створив (creator), назву (title), контент (content) та перелік рівнів в яких вона доступна (tiers).
<b>User - <i>subscribed to</i> - Tier</b>	
Зв'язуюча сутність, яка містить додаткову інформацію про те, коли підписка закінчиться та який номер у, наприклад, банківської транзакції у події покупки підписки на певний рівень спонсорства.	

## Опис зв'язків

### **Creator – *defines* – Tier, Tier – *defined by* – Creator**

Кожен крійтор може створити скільки завгодно рівнів підписок, в нього може бути їх 0, а може бути 10 (зв'язок 1:N optional), але кожен рівень підписки обов'язково має мати свого власника (1:1 required).

### **Creator – *creates* – Post, Post – *created by* – Creator**

Кожен крійтор може створити скільки завгодно постів, в нього може бути їх 0, а може бути 10 (зв'язок 1:N optional), але кожен пост обов'язково має мати свого власника (1:1 required).

### **Post – *belongs to* – Tier, Tier – *has* – Post**

У кожному рівні підписки може бути скільки завгодно постів, від 0 до  $n$ . Пост у свою чергу може бути також опублікован у різні рівні підписки, потенційно навіть у рівень іншого крійтора (репост), пост також може не мати прив'язаних рівнів, тоді він, звісно, ніде в користувачів-спонсорів показаним не буде. Зв'язок – N:M.

### **User – *subscribed to* – Tier, Tier – *has* – User**

Кожен користувач може підписатись на скільки завгодно рівнів підписки, хоч на 0. Теж саме можна сказати й про рівні підписки: в них може бути зареєстровано від 0 до  $n$  користувачів-спонсорів.

Цей зв'язок відбувається через додаткову сутність, яка містить додаткову інформацію про те, коли підписка закінчиться та який номер у, наприклад, банківської транзакції у події покупки підписки на певний рівень спонсорства.

# База даних в PostgreSQL

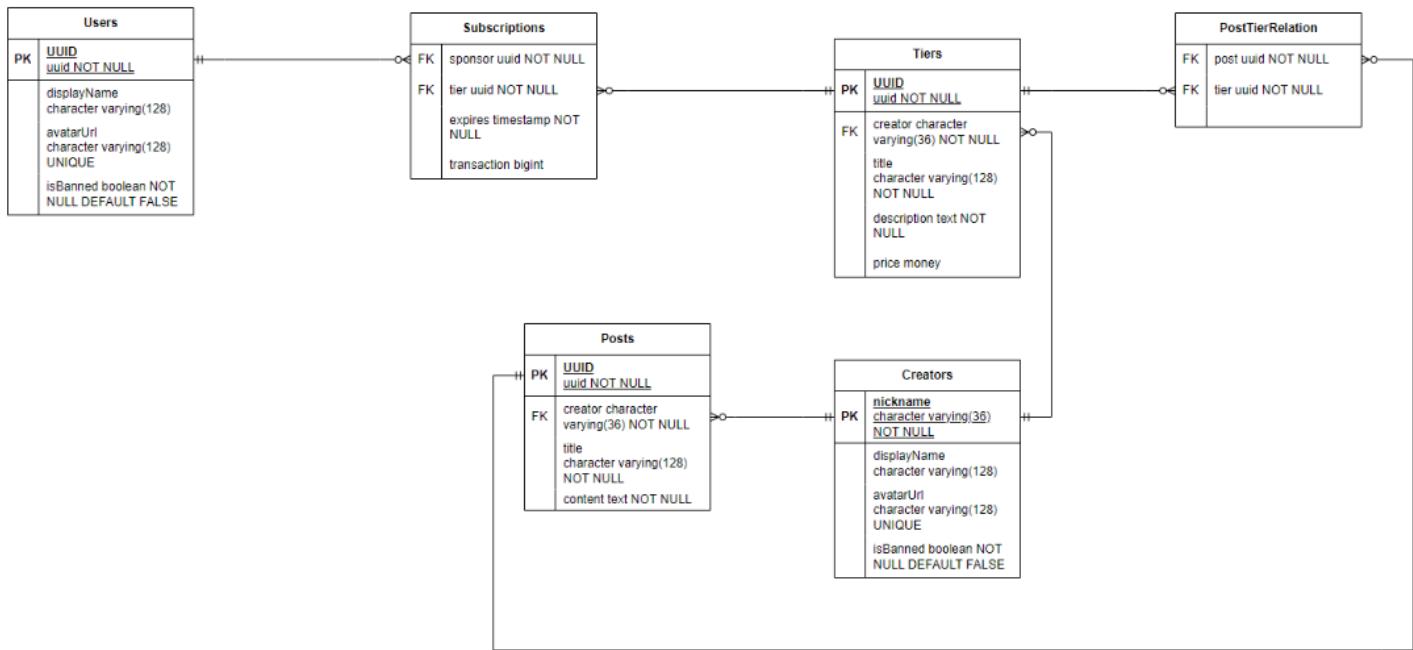


Рис 2. Схема бази даних PostgreSQL

## Відмінності від схеми бази даних у ЛР 1:

1. Був доданий індекс sponsor\_expires для полей sponsor та expires таблиці Subscriptions. Це було зроблено для пришвидшення запитів генерації даних.
2. Оскільки більшість зовнішніх ключей мали політику CASCADE для вилучення та оновлення даних, усі зовнішні ключі було переведено на CASCADE. Відповідно, додаток не контролює вилучення даних із підлеглої таблиці, якщо у батьківській є щось зв'язане, бо сервер баз даних автоматично ці дані видалить. Натомість, додаток контролює уведення даних та виводить інформацію про помилку, якщо користувач намагається ввести дані, що посилаються на неіснуючі сутності.

## **Опис додатку**

### **Технічні відомості**

Додаток виконан у вигляді веб-додатку на мові програмування PHP з дотриманням стандарту PSR-4 (автозавантаження класів по неймспейсам у стилі Java), маршрутизацією та використанням наступних бібліотек/інструментів:

1. PDO – розширення для PHP, що надає реалізацію патерну Database Object для виконання запитів та мапінгу результатів на структуру
2. CTyre – розширення для PHP, біндінги до функцій ctype\_\* для валідації деяких даних
3. Jdenticon – бібліотека для PHP, що дозволяє створювати прості «хеш-картинки», використовується для генерації автарок
4. FastRoute – бібліотека для PHP, що надає реалізацію швидкого механізму маршрутизації HTTP-запиїтв
5. Latte – бібліотека для PHP, альтернативний шаблонізатор для представлення даних у вигляді HTML-документу
6. Bootstrap – клієнтська UI-бібліотека для створення інтерфейсів

## Схема додатку

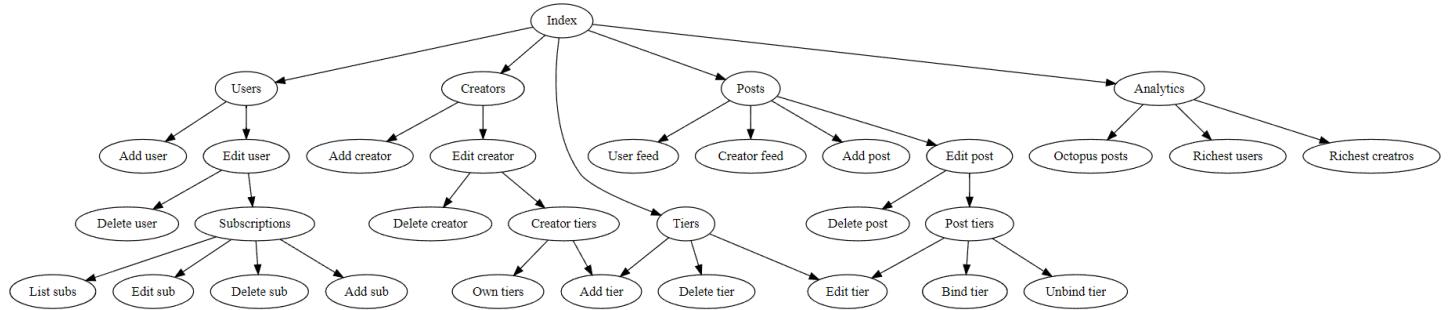


Рис 3. Схема сторінок додатку

Додаток виконаний у вигляді простого веб CRUD-додатку. Він умовно поділений на 5 розділів:

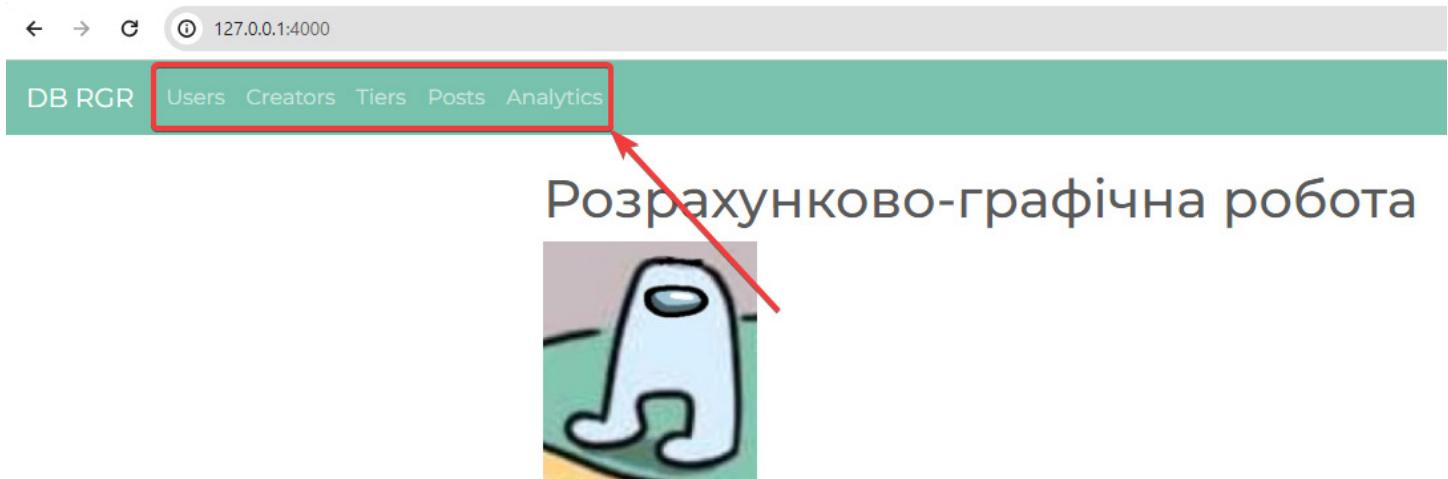


Рис 4. Головний екран (посилання на розділи обведені)

Сторінка кожного розділу, окрім Analytics, представляє собою поділений на дві частини документ, з лівої сторони знаходиться вікно з фільтрами, з правої – список сущностей та посилання на них, а знизу нього, за потреби, відображується механізм навігації по сторінкам (додаток не виводить за раз більше 10 сущностей). За умовчанням фільтри не встановлені та у виборку потрапляють усі об'єкти.

# Users

Filter

Users

 Utonuyare UUID: 9f26a554-c78a-44bc-bf1e-75131c4f290f Banned: No
 Kashinemori UUID: 970cff4-6d0e-45d6-abbe-a7c4da1e0c09 Banned: No
 Kisuniyari UUID: f3c48780-3195-4156-abb0-e83ea91b37f9 Banned: Yes
 Itonomin UUID: 4926f7d3-f6d5-4776-8597-f93f6338fdee Banned: Yes
 Kusonumuru UUID: ac6bbb30-9afe-48d9-989b-98970e8ae35b Banned: No

Рис 5. Типовий екран списку (усі виглядають приблизно однаково)

Фільтри, доступні для таблиці користувачів	<input type="button" value="Include both banned and not ▾"/> <input type="button" value="Go"/>
Фільтри, доступні для таблиці творців	<input type="button" value="Include both banned and not ▾"/> <input type="button" value="Go"/>
Фільтри, доступні для таблиці рівнів	Min. price: <input type="text" value="13.37"/> € <input type="button" value="Include both free and paid ▾"/> <input type="button" value="Go"/>

<p>Фільтри, доступні для таблиць постів</p>	<p><b>Feed filter</b></p> <p>User: <input type="text"/></p> <p><b>Fetch feed</b></p> <p><b>Creator filter</b></p> <p>Creator: <input type="text"/></p> <p><b>Fetch list</b></p>
---	---

*Табл 1. Список фільтрів*

Під фільтрами знаходиться посилання на сторінку вставки даних:

## New post

<p>Post: <input type="text" value="Sample title"/></p> <p>Sample text</p>
<p>UUID: <input type="text" value="(generated)"/></p> <p>Creator: <input type="text" value="GENxxxx"/></p>
<p><b>Save</b></p>

*Рис 6. Типовий екран вставки даних (усі виглядають приблизно однаково)*

Посилання на списках сутностей відкривають сторінку редагування сутності:

## Kusonumuru

User info

UUID: ac6bbb30-9afe-48d9-989b-98970e8ae35b  
Display name: Kusonumuru  
Avatar URL: /ava/ac6bbb30-9afe-48d9  
Banned:   
Actions: Save Delete

Subscriptions

Exclude expired

Tier LF73?26MAfc4No?eYczfExEm5LUJBI0quG9WZyjD0vkJ N7f!pRxCWrouet  
By: GEN13643  
Expires: 12/30/2023   
Actions: Save Delete

Page 1

Add subscription

Tx:   
Tier:   
Expires:  mm/dd/yyyy   
Create

Рис 7. Типовий екран редагування (усі виглядають приблизно однаково)

Оскільки у зв'язків нема свого розділу, то у випадку, якщо у сутності є зв'язки, на сторінці редагування цієї сутності є можливість додати, вилучити або відредактувати зв'язок.

Розділ аналітично-пошукових запитів має головну сторінку із 3 посиланнями на сторінку кожного запиту:

## Analytical queries

- [Octopus posts](#) (posts bound to the largest amount of tiers)
- [Users who spend the most](#)
- [Creators who earn the most](#)

Рис 8. Розділ аналітично-пошукових запитів

У додатку є три аналітично-пошукових запити:

1. Пошук постів-восьминогів: пошук постів, які прив'язані до найбільшої кількості рівнів (мають багато зв'язків, як octopus merge commits в Git, через що так називаються).
2. Пошук користувачів, що витрачають найбільшу велику кількість грошей на підписки на рівні.
3. Пошук творців, що отримують найбільше грошей з підписок.

Перший запит підтримує додаткове фільтрування за довжиною контенту та дозволяє вилучити із виборки пости, створені творцями, ім'я яких збігається початком із певним, наданим користувачем рядком.

Другий та третій запити дозволяють обрати дату, до якої треба враховувати підписки та мінімальну кількість грошей.

Третій запит також має додатковий фільтр, що дозволяє вилучити із виборки забанених творців, або навпаки вилучити тих, кого ще не забанили.

Сторінки запитів мають зверху інформацію про час виконання запиту:

The screenshot shows the 'Octopus posts' search interface. At the top, a teal bar displays the message 'Query took 321ms'. A red arrow points from this bar to the left side of the interface. On the left, there is a 'Filter' section with two input fields: 'Name does not begin with:' containing 'KFP' and 'Minimum length:' containing '0'. Below these is a green 'Apply' button. To the right, under the heading 'Posts', there is a list starting with 'Post: DdI535 xjbH'. A teal box highlights the text 'Is bound to 9 tiers, has length of 3060 bytes.' Below this, several lines of encoded post content are visible, starting with 'tVzHwxXJ4Eys3xVqgM75MZF432'. The entire interface has a clean, modern design with light gray backgrounds and green accents for buttons and status bars.

Рис 9. Час виконання аналітичного запиту

## Завдання 1. Забезпечити можливість уведення, редагування та вилучення даних

1) Показати виконання операції вилучення запису батьківської таблиці та виведення вмісту дочірньої таблиці після цього вилучення.

Відкриємо сторінку із певним творцем (батьківська сутність), в якого є рівні (підлегла сутність):

### Kusehiyuwa

Creator info

 Nickname: GEN1983  
Display name: Kusehiyuwa  
Avatar URL: /ava/1983.jpeg  
Banned:   
Actions: Save Delete

Tiers

Add new...

**Tier MpuPRyBsdbFsstC1OQ!udu7Rgp2zunQUg!aRj4bLHjuCJ**  
**UUID:** f82eac93-7d48-42b6-bed6-e52482bcb3c3  
**Price:** \$5.00  
**Description:**  
h0HfjmOXlouw6BfMgDEqFN3857OJO9faljvygLUErQgQKerpu0sRrgrA2b4YlXEEExljHTYjOzJvh8eBKlFqnlgvQqek?lg5?  
bCIKk5Hz0GoDq667t0xDrlDByYlDZmzih89x7eaGvQ6o36sRjUVYVRIRy1Y7h7hR45q1U77N?  
mAWS5LUfmvSRotd54oyGL9D!Przq1wuEunwc3f6GC6ElRpFzlXoCBmUqTkC9rH9?HtzLQJUC bx2aaxKwv  
Click the title to edit (or remove) this tier.

**Tier uA2xINihymu3**  
**UUID:** 6fb3582d-d586-4442-b450-9eb87a009e42  
**Price:** \$3.00  
**Description:**  
YUuhLclENfhx4xmHmqYBM86dlH8vV6zlc2bbJPJHzf0ksxiE!n0YrUEmtOqoGuCj!OnJ!E5bL?  
leFasE5Qq3zm3Gra9zRX4s0vYf81Pp7zVKlj7UzhLLCqdMnCd8ViQPegOQdO6T2qlR!6nnzxsgctfOGgO?  
9QPBhJpPRU4oWP9fyHx8sdYUX8UYfykeXeESt5bwDZ2cODQ4wYwXxj

Рис 10. Певний творець

Відкриємо список постів та введемо ім'я творця у фільтр, щоб показати тільки ті пости, що входять до рівнів цього творця:

# Posts

The screenshot shows a web application interface for managing posts. On the left, there is a 'Filter' section with two main sections: 'Feed filter' and 'Creator filter'. In the 'Feed filter' section, there is a 'User:' input field containing 'mlpNuGWUxbV0' and a green 'Fetch feed' button. In the 'Creator filter' section, there is a 'Creator:' input field containing 'GEN1983' and a green 'Fetch list' button. Below these sections is a green 'Add...' button. On the right, there is a 'Posts' section with a single post listed. The post has a blue header with the text 'Post: e6064T6 m7rvaFOhzCNB6Wao9c8JDLx?U1VBT1Mg2looMue0F!5 mlpNuGWUxbV0'. The post content itself is a long string of random characters: eTaYHF0oWaAVFpi!Cq?sJNg ChiofBrCOpFPbeJCu6M4tPA SbyOQmmQCW0vFgH 9KbYYByRUSqu1ei2nElUNCps8ZpG0luXhaT3j F0LaGvAnxhQXXV4WytYFPA8iHweFZiA7PmDHsoe9qsJtKLo6KI?!zgtLwmnzGV 0tDTibQvTmbbxNpFLYi?O1?N 7pr4YqqXzafLFaDX3Jpd5!yXY1xAHVnkwbJvCs98QC4neTnSYN!0mk3jYcC5oN qV6H12v!109BHlrjFO0WIUbXdMxDMuU6FEL3aj1bPrKdrQeWyhpcWXsuFkX Ty6i5H7xeovwDVeVJ!pkM7EvSdZDMw91aDTM8J6LUDN?pna? faGUbH7FJkam4DsadZR!2WeHg6BASGLZ9k32FnDhdB6G2VkBtssHe178CNYi 5jAQDdbHilPrY88KznCAllqubL YeLcG?gV359xd? PQBxBGQ86hXdle3q7NI3wPb2zOoJyhyzs320X!55fAKEdUrgGBawyA5seKXX jbe3i!yaoth72fRoREpR85 tewoivuzSVa2lX1sIYr7aJHHi1jm UIDQA lmDjVI4C4Ai7htSEABMoucy17wFdTW39aFhqqYeome P6i9207pPfpE3z7Ke3kfGVx9zTESUGYEeqMaCqaGLkiYWvvxz2G wUrJhx6UfFXLNAql1!ZVVYjBjOdDe cDhTnGzJJI? l8!a4J3l5HrqpOdemzCMSkQqjnOReQq3smLVgAEDPOjnISWBxvgz? 9iucKklqe4RToHYUjyHUIL9Il3J1DRFgz6dPqpLva6aW1DVD3aYJAtz2lwo9rDeEu LjH0TxFrFypGffGxGzgem5NiYA?Hf1!Ra3bvEl? kwOKcp4XERfqhqxguXCp6LnHoCWSxsUD5o53 B? awT 7Yomzoow!CTTjwhPaorVSUKTGGN1TfD6Thoi? FSSpYD5V4zeQENAAKo0zY3yqqfn? Dbzm4q4RS!1ZhHTfo4dJ0wz? E2mLJ!BhPlqzVPLPTB69S85SVG35L? GZHItSOWdw8kYLcjzZDxoHm7uHuK7rpzk9aFXZO1ytNRnuSKzT5bV7U!W9 YpOFmBdM9kz1Zr3oMcuEPa70h Lw6holtRYOA0rSHkNCe8:i1SxJN8dhpPQqIMyEMUw8SFzR!UCe29T8mYWr? JbxQPOwww0RILL5lhG9eQs6x5dMiF6PB9TdxEKOjLaxtu cptcTrk4drorsWNUDcdS1Q g9 ORux3n0h0yPue

Рис 11. Пости певного творця, їх там дві штуки

Оскільки пости також є підлеглою сутністю творців, то при видалені творця ці пости мають зникнути, бо вони зв'язані зовнішнім ключем з політикою ON DELETE CASCADE.

Отже, при відкритті сторінки пошуку постів по творцю, додаток має повернути помилку 404. Але щоб проконтролювати, що пости насправді зникли, відкриємо сторінку редагування одного з них:

## Edit post

Post: e6064T6 m7rvaFOhzCNB6

```
eTaYHFOoWaAVFpi!Cq?sJNg Chi0fBrCOpFPbeJCu6M4tPA SbyOQmmQCW0vFgH 9KbYYByRUSqu1ei2nElUNCps8ZpG0luXhaT3j
F0LaGvAnxhQXXV4WytYFPA8iHweFZia7PmDHsoe9qsJtKLo6Kl?!zgtLwmnzGV0tDTibQvTmbbxNpFLYi?O1?N
7pr4YqqXzafLFaDX3Jpd5lyXY1xAHVnkxBJvCSr98QC4neTnSYN!0mk3jYcC5oNqV6H12v!109BHlrjFO0WIUbXdMxDmuzU6FEL3aj1bPrKdrQeWyhpcWXsuF
kX Ty6i5H7xeovwDVeVJ!pkM7EvSdZDMw91aDTM8J6LudN?pna?
faGUbH7FJkam4DsadZR!2WeHg6BASGLZ9k32FnDhdB6G2VkBtssHe178CNYi5jAQDbHilPrY88KznCALLqubL YeLcG?gV359xd?
PQBxGQ86hXdle3q7NI3wpB2zOoJyhyzs320X!55fAKEdUrgBawyA5seKKX jbe3i!yaoth72fRoREpR85 tewoivuzSVa2IX1sYr7aJHHi1jm UIDQA
ImDjVI4C4Ai7htSEABMoucy17wfDW39aFhqqeYoome P6i9207pPfpE3z7Ke3kfGVx9zTESUGYEqMaCqaGLkiYWvvxz2G
wUrJhx6UffXLNAql1!ZVVYjbjjOdDe cDhTnGzIJl?I8!a4J3i5HrqOpodemzCMSkQqjnOReQq3smLVgAEDPOjnlSWBxvgz?
9iucKklqE4RTToHYUjyHUIL9II3J1DRFgz6dPqpLva6aW1DVD3aYJAtz2lw09rDeEuLjH0TxFrYpGffGxGzgem5NiYA?Hf1!Ra3bvEl?
kwOKcp4XERfqhqxguXCp6LnHoCVsxsUD5o53 B? awT 7Yomzoow!CTTjwhPaorVSUKTGGN1TfD6Thoi?FSSpYD5V4zeQENAAKo0zY3yqfn?
Dbzm4q4RS!1ZhHTfo4dJ0wz?E2mlJ!BhPlqzVPLPTB69S85SVG35L?
GZHltSOWdw8kYLcjzZDxoHm7uHuK7rpzk9aFXZO1ytNRnuSKzT5bV7U!W9YpOFmBdM9kz1Zr3oMcuEPa70h
Lw6holtRYOA0rSHkNCe8i1SxJN8dhpPQqlMyEMUw8SFzR!UCe29T8mYW?JbxQPOwww0RILL5LhG9eQs6x5dMiF6PB9TdxEKOjLaxtu
cptcTrk4drorsWNNUdcCdS1Q g9 ORux3n0h0yPue gGkP5iHLtrntsZkUicOY1ZzTLMcUI?aYwPr?
EJBFFJExcN72bfNOE5Dz!zs9TrkELwoOAEPl5mOJlj56EElwDzrH1 RakzHSzf4IMwETS3jX?4okgnWtVWsQWK9Pzr
NpfILUzwRK5sUtT7WQhYoHapSsw637x85FqtD6Wf6Av96MKA2Yi?iB?QKdQdqx4EgSon60JEWVhCW FpSQ1h9r8?
d1UmMm6KNHs7lzXLIFZstyuOl7ge6kvEo6SL!09ZCHz2hXzQ8GOyiPqfOq7iQ EmMCVDsZl39ysW6G
```

UUID: 2e8e3dce-5988-4137-8bb0-718d51109992

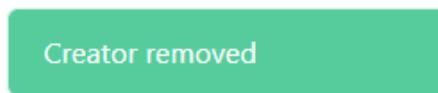
Creator: GEN1983

Actions: Save Delete

Рис 12. Один з постів

Після вилучення творця, пост має зникнути і ця сторінка, відповідно, має стати також 404.

Прибираємо користувача та отримуємо від додатка інформацію про успішне виконання операції:



## Creators

Рис 13. Запис творця вилучено успішно

Перезавантажимо сторінку з постами:

**Not Found**

php

*Рис 14. Сторінка з постами «зникла»*

Перезавантажимо сторінку поста:

**Not Found**

php

*Рис 15. Пост також зник*

Отже, зовнішні ключи коректно працюють у цьому випадку та дозволяють автоматично вилучити звітні записи у підлеглих таблицях.

Для виконання наведених вище операцій додатком були використані методи CreatorRepository::dropByNickname (видалення творця) та FeedService::fetchCreatorFeed (отримання списку постів творця):

```
public function dropByNickname(string $nickname): bool
{
    $stmt = $this->db->prepare("DELETE FROM \"Creators\" WHERE \"nickname\" = ?");
    $stmt->execute([$nickname]);

    return $stmt->rowCount() != 0;
}

public function fetchCreatorFeed(Creator $creator, int $page = 1, int $perPage = 10):
array
{
    $offset = ($page - 1) * $perPage;
    $query  = <<<EOQ
        WITH posts_sequence AS (
            SELECT
                "UUID",
                title,
                content
            FROM "Posts"
    EOQ
    $query .= "
        ORDER BY "UUID" DESC
        LIMIT $offset, $perPage
    ";
    $stmt = $this->db->prepare($query);
    $stmt->execute();
    $posts = $stmt->fetchAll(PDO::FETCH_ASSOC);

    return $posts;
}
```

```

        WHERE creator=?  

        LIMIT $perPage  

        OFFSET $offset  

    )  

SELECT  

    p."UUID",  

    p.title,  

    p.content,  

    t."UUID" AS tierId,  

    t.title  

FROM posts_sequence p  

INNER JOIN "PostTierRelation" r  

    ON r.post=p."UUID"  

INNER JOIN "Tiers" t  

    ON t."UUID" = r.tier  

EOQ;  

$stmt = $this->db->prepare($query);  

$stmt->execute([$creator->getNickname()]);  

$posts = [];  

$ids   = [];  

foreach ($stmt->fetchAll(\PDO::FETCH_NUM) as $row) {  

    $tier = new Tier;  

    $tier->setUuid($row[3]);  

    $tier->setCreator($creator->getNickname());  

    $tier->setTitle($row[4]);  

    if (array_key_exists($row[0], $posts)) {  

        $posts[$row[0]]->tiers[] = $tier;  

        continue;  

    }  

    $post = new Post;  

    $post->setUuid($row[0]);  

    $post->setTitle($row[1]);  

    $post->setContent($row[2]);  

    $post->setCreator($creator->getNickname());  

    $fp  = new FeedPost($post, $creator);  

    $fpt = new FeedPostWithTiers($fp, [$tier]);  

    $posts[$ids[] = $row[0]] = $fpt;  

}  

$results = [];  

foreach ($ids as $id)  

    $results[] = $posts[$id];  

return $results;
}

```

2) Показати виконання операції вставки запису в дочірню таблицю та виведення повідомлення про її неможливість, якщо в батьківські таблиці немає відповідного запису.

Відкриємо сторінку створення поста та спробуємо вставити запис про новий пост, але припишемо авторство творцю, якого ми щойно вилучили (GEN1983):

## New post

Post: Test message

Sample text

UUID:  
(generated)

Creator:  
GEN1983

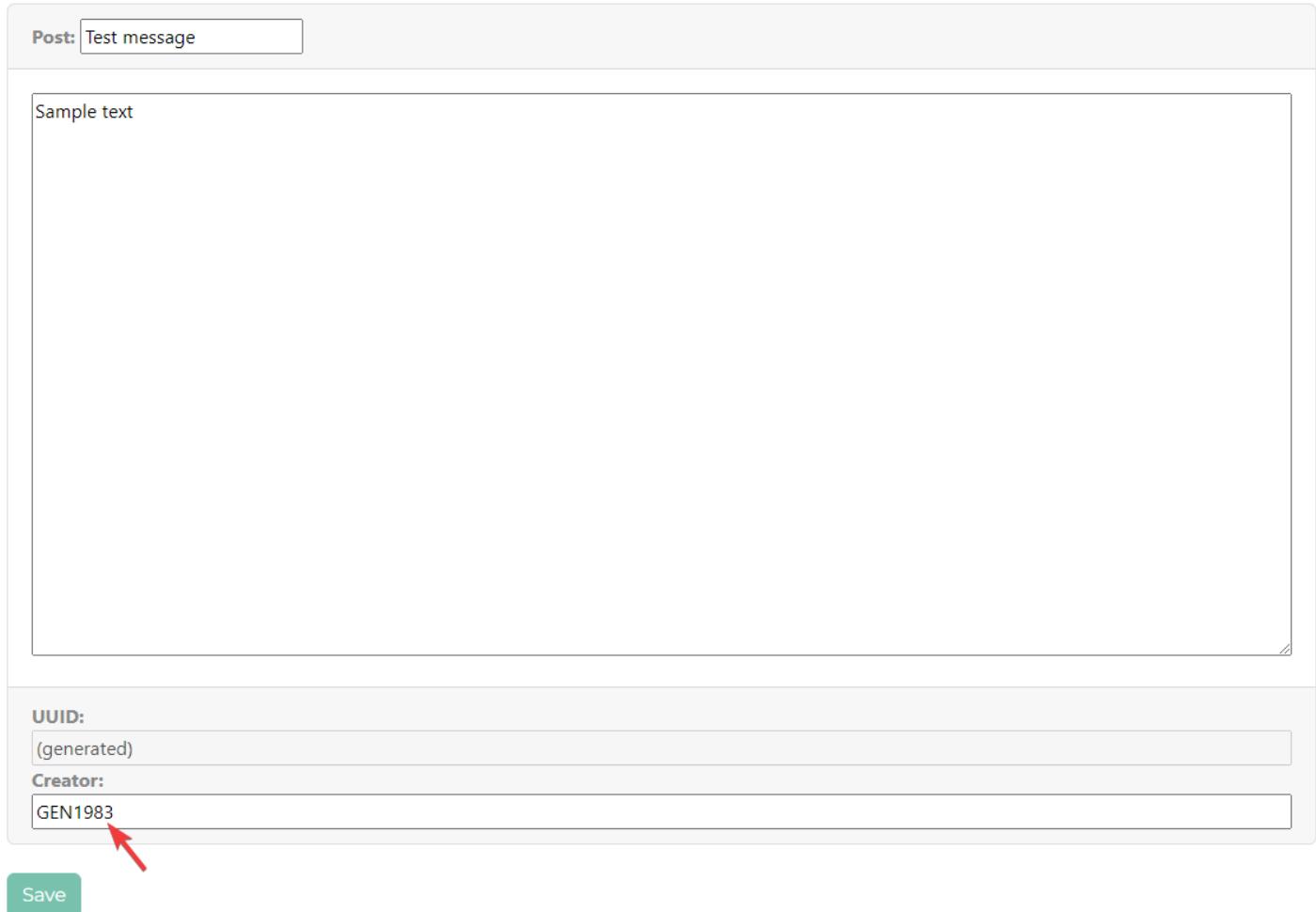


Рис 16. Створення поста, що посилається на неіснуючий запис у батьківській таблиці

Спробуємо зберегти цей запис. Додаток повідомляє про помилку:

PostgreSQL returned error: SQLSTATE[23503]: Foreign key violation: 7 ERROR: insert or update on table "Posts" violates foreign key constraint "FK\_Posts\_Creators" DETAIL: Key (creator)=(GEN1983) is not present in table "Creators".

## Рис 17. Помилка

Тепер спробуємо ввести коректний новий запис:

## New post

Post: Test message

Sample text

UUID:  
(generated)

Creator:  
GEN1000

Create

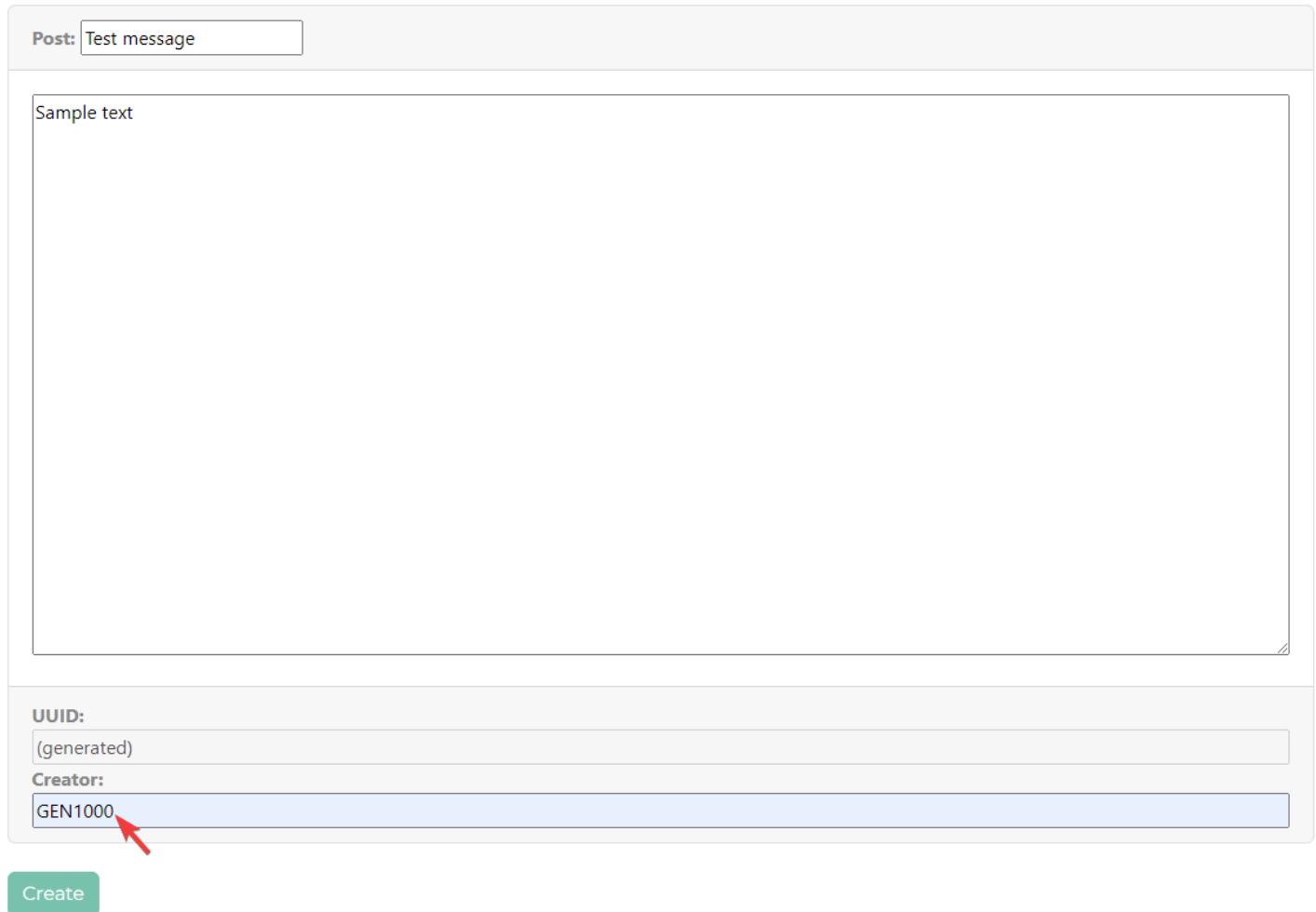


Рис 18. Створення поста

Зберігаємо, додаток має зробити перенаправлення на сторінку редагування щойно створеного поста:

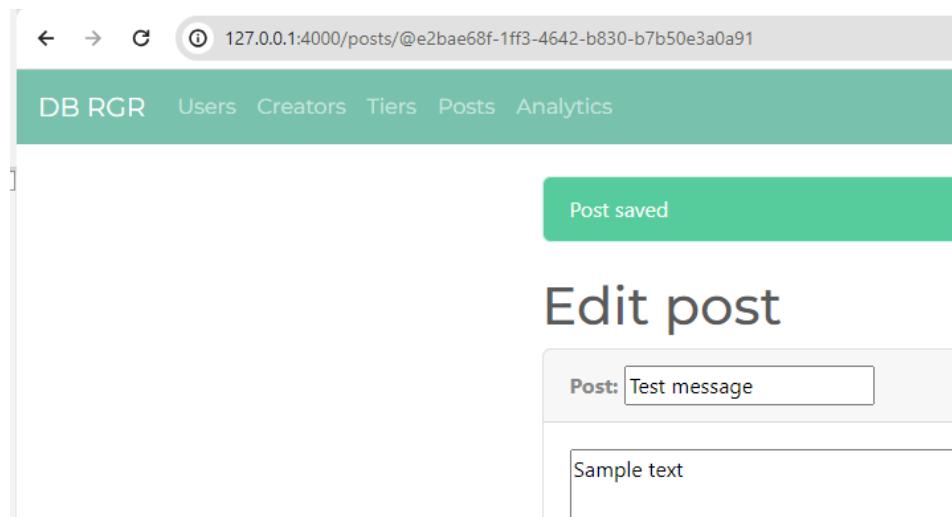


Рис 19. Пост створений успішно

Для виконання наведених вище операцій додатком були використані методи PostRepository::save -> PostRepository::insert:

```
public function insert(Post $post): string
{
    $stmt = $this->db->prepare(<<<'EOQ'
        INSERT INTO "Posts" VALUES (uuid_generate_v4(), ?, ?, ?) RETURNING "UUID"
EOQ, );
    $stmt->execute([
        $post->getCreator(),
        $post->getTitle(),
        $post->getContent(),
    ]);

    if ($stmt->rowCount() == 0)
        throw new \RuntimeException("Object was not inserted");

    return $stmt->fetchColumn();
}

public function save(Post $post): string
{
    return $this->{$post->getUuid() ? 'insert' : 'update'}($post);
}
```

## Завдання 2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі

Генерація рандомізованих даних виконується запуском файлу clean.php із терміналу. Цей скрипт викликає метод DatabaseSeeder::seed, який видаляє дані та генерує нові.

Для генерації певних текстових даних використовуються наступні кастомні збережені процедури:

```
get_random_element(varchar[] arr): varchar
BEGIN
    RETURN arr[1 + floor(random() * array_length(arr, 1))];
END;

name_generate(): varchar
DECLARE
    syllables1 VARCHAR[] := ARRAY['a', 'i', 'u', 'e', 'o', 'ka', 'ki', 'ku', 'ke', 'ko'];
    syllables2 VARCHAR[] := ARRAY['sa', 'shi', 'su', 'se', 'so', 'ta', 'chi', 'tsu',
'te', 'to'];
    syllables3 VARCHAR[] := ARRAY['na', 'ni', 'nu', 'ne', 'no', 'ha', 'hi', 'fu', 'he',
'ho'];
    syllables4 VARCHAR[] := ARRAY['ma', 'mi', 'mu', 'me', 'mo', 'ya', 'yu', 'yo'];
    syllables5 VARCHAR[] := ARRAY['ra', 'ri', 'ru', 're', 'ro', 'wa', 'wo', 'n'];
    nam VARCHAR;
BEGIN
    nam := get_random_element(syllables1) || get_random_element(syllables2) ||
get_random_element(syllables3) || get_random_element(syllables4) ||
get_random_element(syllables5);
    nam := INITCAP(nam);
    RETURN nam;
END;

string_generate(smallint min_len, smallint max_len): varchar
BEGIN
    RETURN string_generate(min_len + FLOOR(RANDOM() * (max_len - min_len + 1))::smallint);
END;

string_generate(smallint len): ResultSet<1, 1, varchar>
BEGIN
    RETURN QUERY SELECT
string_agg(substr('abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!?', ceil (RANDOM() * 65)::integer, 1), '')::varchar AS res
    FROM generate_series(1, len);
END;
```

Табл 2. Список збережених процедур

## Лістинг

```

class DatabaseSeeder
{
    private \PDO $db;
    private float $start;

    public function __construct(\PDO $db)
    {
        $this->db = $db;
    }

    private function clean(): void
    {
        $this->db->query('ALTER SEQUENCE gen_nicknames RESTART WITH 1;');
        $this->db->query('ALTER SEQUENCE gen_transactions RESTART WITH 1;');
        $this->db->query('TRUNCATE "PostTierRelation" CASCADE;');
        $this->db->query('TRUNCATE "Posts" CASCADE;');
        $this->db->query('TRUNCATE "Subscriptions" CASCADE;');
        $this->db->query('TRUNCATE "Tiers" CASCADE;');
        $this->db->query('TRUNCATE "Users" CASCADE;');
        $this->db->query('TRUNCATE "Creators" CASCADE;');

        echo (microtime(true) - $this->start), " CLEAN OK\r\n";
    }

    private function createUsers(int $count): void
    {
        $query = <<<EOQ
            DO \$do$ BEGIN
                FOR i IN 1..$count LOOP
                    INSERT INTO "Users"
                    SELECT
                        UID,
                        name_generate(),
                        ('/ava/' || UID::varchar || '.jpeg'),
                        (RANDOM() >= 0.5)::boolean
                    FROM gen_random_uuid() sub(UID);
                END LOOP;
            END \$do$;
        EOQ;
        $this->db->query($query);

        echo (microtime(true) - $this->start), " USERS OK\r\n";
    }

    private function createCreators(int $count): void
    {
        $query = <<<EOQ
            DO \$do$ BEGIN
                FOR i IN 1..$count LOOP
                    INSERT INTO "Creators"
                    SELECT
                        ('GEN' || UID),
                        name_generate(),
                        name_generate(),
                        RANDOM()::float
                    FROM gen_random_uuid() sub(UID);
                END LOOP;
            END \$do$;
        EOQ;
        $this->db->query($query);

        echo (microtime(true) - $this->start), " CREATORS OK\r\n";
    }
}

```

```

        ('/ava/' || UID || '.jpeg'),
        (RANDOM() >= 0.5)::boolean
    FROM NEXTVAL('gen_nicknames') sub(UID);
END LOOP;
END \$do$;
EOQ;
$this->db->query($query);

echo (microtime(true) - $this->start), " CREATORS OK\r\n";
}

private function createTiers(int $count): void
{
    $query = <<<EOQ
        INSERT INTO "Tiers"
        WITH creator_sequence AS (
            SELECT "nickname"
            FROM "Creators"
            CROSS JOIN GENERATE_SERIES(1, 55)
            ORDER BY RANDOM()
            LIMIT $count
        ) SELECT
            gen_random_uuid(),
            nickname,
            string_generate(10::smallint, 64::SMALLINT),
            string_generate(128::smallint, 512::smallint)::text,
            NULLIF(FLOOR(RANDOM() * 20)::numeric::money, 0::money)
            FROM generate_series(1, 1) FULL OUTER JOIN creator_sequence ON 1=1
EOQ;
$this->db->query($query);

echo (microtime(true) - $this->start), " TIERS OK\r\n";
}

private function createSubs(int $maxCount, float $staleFactor): void
{
    $fresh = $maxCount * (1.0 - $staleFactor);
    $stale = $maxCount - $fresh;
    $freshQuery = <<<EOQ
        INSERT INTO "Subscriptions"
        WITH user_sequence AS (
            SELECT "UUID" AS SUID
            FROM "Users"
            CROSS JOIN GENERATE_SERIES(1, 55)
            ORDER BY RANDOM()
            LIMIT $fresh
        ), subs_sequence AS (
            SELECT
                SUID AS uid,
                (
                    SELECT "UUID" FROM "Tiers" WHERE NOT EXISTS(
                        SELECT * FROM "Subscriptions"
                        WHERE "expires" > CURRENT_TIMESTAMP
                        AND "sponsor" = SUID
                    ) AND "UUID" IS NOT NULL ORDER BY RANDOM() LIMIT 1
                )
            )
        )
    EOQ;
    $this->db->query($freshQuery);
    $this->db->query($staleQuery);
}

```

```

        ) AS tuid
    FROM GENERATE_SERIES(1, 1)
    FULL OUTER JOIN user_sequence ON 1=1
) SELECT
    suid,
    tuid,
    CURRENT_TIMESTAMP + (CEIL(RANDOM() * 30) || ' day')::INTERVAL,
    NEXTVAL('gen_transactions')
FROM GENERATE_SERIES(1, 1)
FULL OUTER JOIN subs_sequence ON 1=1
WHERE tuid IS NOT NULL
EOQ;

$staleQuery = <<<EOQ
INSERT INTO "Subscriptions"
WITH user_sequence AS (
    SELECT "UUID" AS SUID FROM "Users" ORDER BY RANDOM() LIMIT $stale
)
SELECT
    SUID,
    (SELECT "UUID" FROM "Tiers" ORDER BY RANDOM() LIMIT 1),
    CURRENT_TIMESTAMP - (CEIL(RANDOM() * 30) || ' month')::INTERVAL,
    nextval('gen_transactions')
FROM user_sequence
EOQ;

$inserted = $this->db->query($freshQuery)->rowCount();
$inserted += $this->db->query($staleQuery)->rowCount();

echo (microtime(true) - $this->start), " SUBS OK: miss=", ($maxCount -
$inserted), "\r\n";
}

private function createPosts(int $count): void
{
    $query = <<<EOQ
INSERT INTO "Posts"
WITH creators_sequence AS (
    SELECT nickname
    FROM "Creators"
    CROSS JOIN GENERATE_SERIES(1, 55)
    ORDER BY RANDOM()
    LIMIT $count
)
SELECT
    gen_random_uuid(),
    nickname,
    string_generate(10::smallint, 64::smallint),
    string_generate(256::smallint, 4096::smallint)::text
FROM GENERATE_SERIES(1, 1)
FULL OUTER JOIN creators_sequence ON 1=1;
EOQ;
    $this->db->query($query);

    echo (microtime(true) - $this->start), " POSTS OK\r\n";
}

```

```

private function createPostRels(int $maxCount): void
{
    $query = <<<EOQ
        INSERT INTO "PostTierRelation"
        WITH post_sequence AS (
            SELECT "UUID" AS PUID, creator AS CUID
            FROM "Posts"
            CROSS JOIN GENERATE_SERIES(1, 3)
            ORDER BY RANDOM()
            LIMIT $maxCount
        ), rels_sequence AS (
            SELECT PUID, TUID
            FROM post_sequence p
            INNER JOIN (
                SELECT "UUID" AS TUID, creator AS CUID
                FROM "Tiers"
                ORDER BY RANDOM()
            ) t ON p.CUID=t.CUID
            LIMIT $maxCount
        )
        SELECT * FROM rels_sequence
        ON CONFLICT DO NOTHING
    EOQ;

    $rows = $this->db->query($query)->rowCount();

    echo microtime(true) - $this->start, " RELS OK: miss=", ($maxCount - $rows),
"\r\n";
}

public function seed(int $users = 100_000): void
{
    echo "Seeding for $users users (c=0.25, t=0.5, s=1 [sf=90%), p=1, r=0.5)\r\n";

    $this->start = microtime(true);
    $this->clean();
    $this->createUsers($users);
    $this->createCreators($users / 4);
    $this->createTiers($users / 2);
    $this->createSubs($users, 0.9);
    $this->createPosts($users);
    $this->createPostRels($users / 2);
}
}

```

## Фрагменти згенерованих даних

UUID	creator	title	content
67d20cef-75f2-4ced-9cd2-cf7868ae8af3	GEN9999	ZjPunTkv0DHl J3onURE?qE38sKnDYHMW7Nnsv1qv KivP...	kycn242lon qkgcC7OJd5wLnokHJ!QfbHsPhaps1LGBbFFF...
e1e2379b-5610-420b-aa9f-c02a3f234e72	GEN9999	!otoFx4a4qJcd!9?lKMdm52YFCwLkEdxV0NBp!RuVqLICca...	86v5y82tWukgX hMNEveSngymgVH231XRJLrToBdHcPlrT...
87b41327-1d9a-4aa4-8dd9-5a89afc7f76d	GEN9999	O6CCPwkmZCKq	5tGjYT!Mpx61PM381XOdalV1xISFvV!QOHafbFip?zXK...
a70922cb-7daf-425f-857b-07ccfd99da1	GEN9998	XMECf6a2c2vBrvJy0107	SzpUeVwRQskh62MOX6ubUt7T9hH7hNEJuwXtM40Dxu6...
53acc5a8-b343-43e4-840b-b4c3b34ebdae	GEN9998	emDrv9uH6aOQOFCDtZ5NRorJn2!tB?6	XfBSLETMzFeRJeXVGmAz9CmNH0!JsRxzsu3uqF2QW7X...
ccef8dd0-a26b-4678-a966-6c8708163e55	GEN9998	A08lrlmfldnWah1lhTOg2D8cMjmRt2NxIzchYT2ub4vwMe...	9y2DAKLmu7BAWgDfZept2XXQoLC rQApEJffMPofp2pRRI...
7bcad80c-0dfc-44be-bf63-49351d5c9619	GEN9998	nCC3UemmbO0I3cie 7UNyi	X?Fdmf2TWbbF0Q6K6E7kHa8!uPYPeoFg73GM02lia3wr...
f7e75894-857d-4b0a-91e1-46a20a784e13	GEN9997	1U8Bhv sMwLqabsEFZM 5CklsRv	6fAsrLhNI!zLGV2o d4KdgElxDW K2KzJ0OyUnMsVKYlvyt...
4778afc8-bc39-46b0-b564-191bdda1dcba	GEN9997	SK5opH5wMFuofarLapeVlrrbiUF	d2yH0JnXfhZtN!Ja!LcdzkZYAI56ul5i0f6vBTQazJ0rrnZ1E...
5b5168a4-1189-4f34-ac39-185bb195c9b3	GEN9997	1Aj?YDG ws kgFrD3?4VB3oTp5PziQLXSoRCNyke9Q	td9yGF8GCobTSOxA!dNcW8NNNSxNHkhz0O0WvQbC6Rz...
28fbcbef1-e750-4c01-827e-48bb01d643b2	GEN9996	OL!5IRcxOl2B5rlj	mp?pB5Pc0eAqm3Szfg3zwnd4fvPxfkBgjNxnJl76VNGaX...
e7fc74b3-e915-4849-93f8-474655ff0197	GEN9996	eHqV!f8PB5jqUffPK7hDuKXYpdy2FqfpNPkPgu3Mu9KYk3Z...	VRFvS3cn3RWW9jCu2aHWSh7Kr4NCqA?s ODAzqTwi...
08431453-1375-4af2-8bc8-8166ce8c0e71	GEN9995	QoY? 4EKf!JLctv zuA3L2aS9R8rxwAggV!awW0h9P0lx	8gLfdZvDcygDPLFUhmho QLyedFr8TJedq3GIumppFO r...
a134ce35-7831-4db0-80f1-64917908c99a	GEN9995	Bkig24b?GbN?AGscQmrLvorzez92ETxXF5JcIHneAOlYArT...	Y 17bKchT2iOUjhgIN wRyfw!?zbaojTEB!P?eOWG1UvEG...
1b98a14b-bb91-4bca-829b-86366e5fdcd6	GEN9995	Cu38z9oevqOHPopIu4mi	LfiwUtw!UpUuoAX4OBtmbP6d7PCmzhGzW4wZ7nIyezpv...
dee7b352-e1c2-4518-a91b-d6ea85c2af5e	GEN9995	3boMpJOnUhxh2409ZbzVLth2vFc?2A1k7NlvUQLD5e3?U...	VHrqW7KhkBuiufaFzsMQ58ekH73vhgw iyQXbs27S1oIx...
15ad6a2b-b308-421e-baeb-0e5ecc0c86bd	GEN9995	cBc48PBj4czR1QF!SssaOwtpfE	FRDSt4wXQZIuo5rXGmhvd?2a2W X!r8BDQDscyGhblusK...
4d03a5a7-8e1b-4a12-99a7-627201d3b49c	GEN9994	kPTdC0y5HrautQNVMB2br9f6VgsJgaZ8!uQQWh8xEZp...	DJQ5CPLoS0qeA2Nl0vJRJ729pBoQ16pVxwevb9BFTS NL...
3e0689e0-f52f-4ddd-ac38-afd0ac6fab04	GEN9994	fenStjyUQmkf8YjI	IsBzyJn tNuwlaeRY0U9!QT EuYIrgIYJBo6TM!h eIR6bND...
39f58658-08af-4edd-9bf6-1db398abdff4	GEN9994	oXUyPSt0KaqqMjTy?WzBn9m0z7X	D68784jZgWhWdZsWjkdUZIgsvH5VyaII2WckV9HzV6yh...
545a2aaa-4645-4525-b770-59ba6d7b8fc2	GEN9994	1o!YXhYL?opG nG7RNq 91vmG1o eMiKaxqKxvAwTT zEo...	6!p0Ek EoZoxYkzTsMBuy?es7wU2IX33bIKrIXQa9v03Y...
79bde472-c707-427a-9e81-688831ccbdd7	GEN9994	9tbICCXxHjY7hsApjaJ3JBKUjhAuHx4sdEIkD2aE8Yi6lsTf...	gNNAAFkY5W6CdHAllbcr5mhpVAdJPSZB3kJB7hmLpN2Y...
710d8c92-eb8d-4cb3-938a-e74852b1df31	GEN9994	50fMoefCrFsJr4kZAI3GHswvJERvY0ogJzzxtajlVarTh!	k 2FY5qhe2TgK0oNIT ICK!fJRartxQOh2?fle?Dkaj!RECI...
2d5035c5-3980-47ef-ae93-404e3b64097c	GEN9993	J0vLxYQySxDaTy0y6gOxQsZnB1v	SEfa?v4Fgihow?CNI YCZg6qqqPJCxOGw4hWd 1jahcQ...
234a47ec-1c15-4c31-a0ce-c870f9beb5f3	GEN9993	F52LYRG5FDn4dntcbqb6YWRwe9b?5w2bj1UFt5kRv XV...	gpNT73INT0dkQNobsZ4sIyZf0YvXDekZyqTwzonbyZifrq...
69b1a046-0d2e-4bdf-b2a4-965754e3a033	GEN9993	MEDvI71QxEVB8TbDlex?	?5l2Q0WedwIgN56VJYNBZVtMEExYIrm64Iy!fMfaoLXAcI...
d6436da1-dcb8-4f09-8ea6-4614f9d8324	GEN9993	cTf1MxdsO7rx	owikiqxDs!TuUaRnxS1c58Iy!l7eeGyIRO9xHO x6o Toa8g...
600bab11-67af-48e0-887d-02ec3ce5a34e	GEN9993	1y84TxrLyUQU!XpdmRR_8EUdRQ gLB9DiR8fzHI0Wa2yw...	TrGkD3G?Osu1N1X9GbpHx07F1zzGLG?fGJE6TuyzghJqe...
c32d070b-f992-4e2d-b957-3dc2845b36d0	GEN9993	gR!rXoc3RZO7LIMfePwlCk6YhBKpznnjuDrIqlvCsicGB1	w?AJ502khxhd1qSJrSHwZAj8plt02RTfJHDJV7zdOX2...
8f1e556b-136a-4ec2-b3a1-c20616f9aa3c	GEN9993	b oWnixfReY?jhKS2iivTDRKLStLgoVdIpDMFXUr	gJ8H5VqY 8yXQ5LNQG6Pd8K1VgvlaX2TW8rsR7St3u7h...
d38e84d1-5f2e-4470-bb3c-b6045e6261a5	GEN9993	EQ8r6q4 ZtiQ!pBkrHrkSx?hoa6pOozL aM8Ul55r1?GWNx	2o0uwskKdbQX?38NwF1EZ41tGMVRJ15BuDmdyKwSlz?Y...
770ea3ad-573f-435a-ba18-6a4e31b90079	GEN9991	dBRtZNCvO7QAVM 3CvBuVfsjDgA1g8UV!cBufKToEJn94!	Y4qdKMsCTwxGSbbp1d!9MK75kg!B5hItPc4s90aiWbG1n...
25a6fef4-9604-4681-81d4-1b2d51332ae5	GEN9991	ZA4tZ7CLfoCuVBQw2HGGo73n57Sx K	8haFYa66L9ef0lEpALb89Z22EagBfgCNRpnhXAp6jOxzY...
185bec68-7b1c-411e-99f0-ebacdd255ca4	GEN9991	y!k08T24gv1!NeFj?dt DFAQV0Ve?0akzFCGpqSCwWvMu...	fjs 5T67toN qCrPeoX7IBWbO1JQvsRgjXhv8tBVV5LURFof...

Рис 20. Згенеровані дані таблиці Posts

UUID	creator	title	description	price
2d195f1f-141f-45be-bd01-1605a922e8be	GEN13412	zfyYWUr58JGm!Pu2Ypz!70xK0Ng OXbvUU1htXC?E3Y	ad6Cnyl6H!dTsho7CB?vJSMve!P!wtg uxt6JzFACF9G4TS...	\$14.0
74abd40e-8238-4e91-82f6-d32ccca009da	GEN11398	Ztsv2Hvp1kxcf2Dn9gEC 1iyzV4 LYc 8oA IOBfvdr1cP c8w...	VN1Ts1o0vvDAT9EfhraUvjnLNHq2okm!OGvf9dc!PrQSm8i...	\$1.0
73d22042-73f1-4839-9d9b-9c9f1183c051	GEN13294	cLlgqKmthQYwqsV67?6pgOQKTvraoH6YNM8kTkZP	Ya6EU4T2iWqJd1Q99E7UjcuJveQ5RzZBxII5zUQSBAIKIL...	\$6.0
a94f832b-e158-458e-9424-72eb161d28f4	GEN20009	tc NDxsdjhTo82r0CobU73cykd9NS3yi5inqFYi8jsq	1y0QlIWZ1Pvete1HIXZ78MCVmLd4tG?ds!iNj7h45Whovs...	\$12.0
12d54fa5-d9ba-43d0-e55-1b42aa1ad5c7	GEN22235	kvcBX3h!uR386h!ydgLKlnSge3	Ix07OjqD750Qx8aGJzOzjZO5xa?7AHEMkuO1tQB8mLTF...	\$2.0
b049f5f1-c8cf-432f-9429-e9a617cdb4a1	GEN834	IO7IXIg YdUuynFcVuVcgQXec f?FGI	G5KXJnsZajhg3IkC57Os5J8FLJwDd2FTTerDzGrE0vWA...	\$12.0
c8532254-6f23-494e-8272-4df292ba8d12	GEN6569	28VqOvImWSU1JRQtp3h	28WJJngLAI?w6lq ZB tCuiqlJGfWYsFMy6lHRxvGB3z7k...	\$3.0
62702421-a9df-4b8f-bbd4-f6372b11892b	GEN20	xdywvxNWcxqc0nv?dhpnB5e	DcuWw5bL0ATdPUrzFmGz1WE12Ddh7Jlm3KdwFo2Bhmh...	\$17.0
f3cbf2da-db54-4df4-bb21-f2ac47620242	GEN130	2lgwzE7G8A1 NMZYyeyDSOuafCJwxjtjeZRXL1y	9xfcaWYstpXXIYPI6h3achUhDthfAt0LabdJ1KJuZbEB59bb...	(NULL)
d2b93018-cfd7-4faa-ae09-19a962477c59	GEN18493	uPMi9Uf1rQvZ2J7wh2M!26?kDBR94hsRsbVUpE	zbBbxpbXw8mUcc9qCQyXGp6lLv ung3DdEuctWQuDWfd...	\$9.0
75a4fe1b-91bd-4878-864d-e0010750838a	GEN375	9p0Ck?uewL7P3bz6Yq7PpMk1R2tVGW	R?7dP RHqjS85km17jkrr!uMWm9CABv9Ark9i6Tc?Sfxpk...	\$2.0
93f82a5f-e236-4780-b3c0-41ac3871c64e	GEN4599	vCYWqqlfaZJhY8Ls7?cJdX50HV035?yjPCvACj	?ju8mwBZeEoJpcPwz?N!XeVmFBVL4Sg3qrAe529FgNLUM...	\$6.0
7667b1b9-eb7d-4726-ac05-870969735360	GEN22227	p?!zQCSbjzsZ?kOIGLfhSw?EjQ8BWVYMTuVuNkZowB4vP1Eh	LXOs7VlAdh4HWb C0nWw!wZMnAaqTu3dp231dHfUE...	\$2.0
b7b94969-069d-4842-8541-2a2246e09ce5	GEN24323	bCEnRXVJULx3p	FQhg0anLgnP5k?7tOhoEzgWG4L!Q CaajabimddU81g1T8...	\$2.0
c70f3db6-9fad-417a-8a71-1186927ab0ed	GEN3737	IA4vhHPb22t!6Y0csshgFULg9opdmM1M6sgh4VLpjF3hhQ...	JFPxQLUUItOHan1WZGeSmGmFl?kUuv3WIuj29wVfoRJ...	\$10.0
704d49a1-e12e-469f-95d6-28f47a5b2a6f	GEN21286	A5hha0p mPm4 mgIckfQsBGnXrGvMfQAKWShSgmfc62e...	vZ5ndNWHeZ!s5GyU QKCd1ds!50GQAR23Ioc!1x!o h4...	\$9.0
44248aa4-a4e6-412d-9650-fa08772b4925	GEN23370	cJ6BumTaBvg	XJLG0NH6d716YIk9GX00GZIBWYk8wWtGm8AimrZWwp...	\$2.0
c637c826-f621-4258-a918-973bad255365	GEN17040	XImj!6ZyO?OXsXviFR?CFtOEADnYkU!Y1GHPh3Zdmoh85...	AvxbLr9CS2TP1btyXD 4sgN8ttMNaZPkAVD4vdG!PNB54w...	\$18.0
0e3db7a9-5969-4c68-93cd-210b2b2d4f4d	GEN16424	?jcQ!MUSDS4cMDEarar	hByyMXkbk2oJIT3O?jLeZdJrt5 VjgT1H5UQPFIUpPXJh5v...	\$6.0
e5c07cf8-c882-4af2-928a-3dd8e0fc239	GEN15619	Lar9lBYmgFUue6q	?gwG74hVhoL1Gkzew37XNAFR1EZe9fqalLgRz 2gt5KJO...	(NULL)
a69d6b18-da59-4e29-a43f-f7e42f65edf4	GEN7495	WENi0W4VDxawA8gZgB1ZyvNrNphUa879iLx419ArzsJRT...	g Ceip ZENzjZWD0m?swXhCHCKy3WHIRuFfHSc770N wlr...	\$12.0
92ce2825-589f-43e3-a91d-5c47ece351cd	GEN9923	KGKFedZ1uW4jyWRscqEAHT1HmsbaZ4tynWq0	Nmc4!Jaql3Go9ElgS6RYKDPr2qIQtkFrmMqt0YcCauE7wV...	(NULL)
005b0d3d-a7fb-4f8e-bb39-ff7fb1937f6	GEN22174	sK3RWd2lCqwKDVlIwUAQdW70eUhEdE3Jw t8	OH14SH0h4c4TR1D78ceXM6zjj9WEON rHfEsb tjwyReZK...	\$1.0
78544a62-071f-4965-b1f4-6764fc60bd3d	GEN13998	p6RlkHBIyY3po9Sc?4YcgdpajogGqt4bzOvtSwsML8CB84p...	kJNoBS3NAEvAvuL1AopzXmSoItRIkgl6kn6u4tghkRPl7b...	\$11.0
8bfff6bb4-1ce3-4522-b410-90ee183ef202	GEN13961	62?1o5rYs3!2F	bXSKz4Pmutjk5FwNsMPu008zGfwu!84EyOQ7cA1k2hEJi...	\$16.0
c2a75fa9-4398-49c9-b70a-cf04155820e3	GEN17796	nCQJA8d9k9Sdwx8f0IMsXfp0kdU5R7k	SEehE3Z?5wnfC?DNT m19ipUx8YcGcvw4!gKgwudJRP...	\$2.0
e5444edc-16eb-49f6-945d-834178053d4e	GEN1497	Lc3JH5R?wJir8w	rn3U8zhnhNhqI6g?2tXZNegrpmFeV?H61ohukK2DFZv8JU...	\$15.0
cb32ee30-0bf5-f490-9ee9-1acc1bab1cb0	GEN24949	ZZx90ZOLFM7UpaleGXxHjlw63x5qCug583h9Cd	m uOIW?F EH1lbcS7tP6rwbtvlB8vhsKrgWFoTFFZFBbFg...	\$4.0
9cd6aac1-98fe-4595-9ff0-8582134fb6d1	GEN20567	j2bQj8Xmer1tFQ1Klg2cAvmbRM4gBtg54dAA	?4X?twmoObmBS7md76MWuuh9sQD4ehHy?TFCpKPsnbjC...	\$10.0
94d05383-e091-4e07-8296-198b788db5c9	GEN21750	u0UvhYuDeE2BymVzwWTUujFhtT3Ug5AU	DeUu18v50CvQjQ4llH41j2ZJAH5bcxGemmNXxkdlVw6Xb...	\$15.0
47720398-6f4d-44aa-9e43-802c120118e9	GEN19201	3!TkcjjuT7flhlkhUyFyk11!9	qeHqSUmUw0uurzis33zUSqgPa?m2G6m52WMyIU1OxiG...	\$19.0
6253df9d-1bc0-460c-bf84-e6c6c2745260	GEN9388	plrqD4FGDpuByV3v1OLxiCNCNgIdYnefbUy7a93DsqqwWkK...	dr9rH8KnSketriBw?2 JTURW11RtS90WCANKsUO9xmy...	\$4.0
17875545-d24f-4b5a-958b-7cba336ccc47	GEN15909	YeAS3ypovmc rvy pkSGA!0akhcSLG1zVzCnRHqR9S2nE...	O1b2Mvof1tm!Tipej2u6oUOHkJOBjm80c1pvZwta8T?NlH...	\$15.0
874b1773-194b-40ec-8d12-3ecd1c6d41e3	GEN4748	x rf7NKnikyUc2yQ8fhFqxwsvUsaFAHkb!Mc1DjN JS	INDESL2Guu7DtPO9xe 4K4wyDmeeDtqA5qSw UUJqQGS...	\$9.0

Рис 21. Згенеровані дані таблиці Tiers

nickname	key	displayName	avatarUrl	isBanned
GEN1		Atonoyare	/ava/1.jpeg	false
GEN2		Katenemore	/ava/2.jpeg	true
GEN3		Osenuman	/ava/3.jpeg	false
GEN4		Kesonumiri	/ava/4.jpeg	true
GEN5		Kesunumiro	/ava/5.jpeg	true
GEN6		Usofuyuro	/ava/6.jpeg	false
GEN7		Kesenamiri	/ava/7.jpeg	true
GEN8		Isuhamiru	/ava/8.jpeg	false
GEN9		Atoniyaru	/ava/9.jpeg	true
GEN10		Katenemare	/ava/10.jpeg	false
GEN11		Ashiheyoru	/ava/11.jpeg	true
GEN12		Esuniyuwa	/ava/12.jpeg	false
GEN13		Atahoyeru	/ava/13.jpeg	false
GEN14		Esuhamire	/ava/14.jpeg	true
GEN15		Kesonuyora	/ava/15.jpeg	true
GEN16		Uchinomiro	/ava/16.jpeg	false
GEN17		Ishihiyura	/ava/17.jpeg	false
GEN18		Katehamare	/ava/18.jpeg	true
GEN19		Kosonemuru	/ava/19.jpeg	true
GEN20		Kesenomura	/ava/20.jpeg	false
GEN21		Katsunemere	/ava/21.jpeg	false
GEN22		Itonimari	/ava/22.jpeg	true
GEN23		Osanameru	/ava/23.jpeg	true
GEN24		Kosahoyuru	/ava/24.jpeg	false
GEN25		Kitohimiru	/ava/25.jpeg	true
GEN26		Kotaheyuru	/ava/26.jpeg	true
GEN27		Kosunumaru	/ava/27.jpeg	true
GEN28		Esonomuro	/ava/28.jpeg	false
GEN29		Ketsuniyuro	/ava/29.jpeg	false
GEN30		Kotsuhoyaro	/ava/30.jpeg	false
GEN31		Utanamuru	/ava/31.jpeg	false
GEN32		Kitanamire	/ava/32.jpeg	false
GEN33		Ichinumoro	/ava/33.jpeg	true
GEN34		Kisohemin	/ava/34.jpeg	false

Рис 22. Згенеровані дані таблиці *Creators*

UUID	Avatar	displayName	avatarUrl	isBanned
9f26a554-c78a-44bc-bf1e-75131c4f290f	Utonuyare	/ava/9f26a554-c78a-44bc-bf1e-75131c4f290f.jpeg	false	
970cff4-6d0e-45d6-abbe-a7c4da1e0c09	Kashinemori	/ava/970cff4-6d0e-45d6-abbe-a7c4da1e0c09.jpeg	false	
f3c48780-3195-4156-abb0-e83ea91b37f9	Kisuniyari	/ava/f3c48780-3195-4156-abb0-e83ea91b37f9.jpeg	true	
4926f7d3-f6d5-4776-8597-f93f6338fdee	Itonomin	/ava/4926f7d3-f6d5-4776-8597-f93f6338fdee.jpeg	true	
ac6bbb30-9afe-48d9-989b-98970e8ae35b	Kusonumuru	/ava/ac6bbb30-9afe-48d9-989b-98970e8ae35b.jpeg	false	
10268d76-3d70-4bbd-9731-945e3238b6ff	Osenumeru	/ava/10268d76-3d70-4bbd-9731-945e3238b6ff.jpeg	false	
aeefb7b7-7413-46a4-8529-7ba85f3e9c61	Asehemewa	/ava/aeefb7b7-7413-46a4-8529-7ba85f3e9c61.jpeg	false	
cf9d6849-32e2-4a53-bfa7-ece7918b95e6	Kasaneyuru	/ava/cf9d6849-32e2-4a53-bfa7-ece7918b95e6.jpeg	true	
91639ea5-4231-4d80-a3d2-94793ce62eb5	Kutsunumaru	/ava/91639ea5-4231-4d80-a3d2-94793ce62eb5.jpeg	true	
8bfaa6f0-3bac-480c-8d7d-4fd3dda94085	Ochihemuwo	/ava/8bfaa6f0-3bac-480c-8d7d-4fd3dda94085.jpeg	true	
71af21db-cb3e-4bdd-b1e1-f493fadca256	Asanomewo	/ava/71af21db-cb3e-4bdd-b1e1-f493fadca256.jpeg	false	
1bf6f5b3-3635-4d83-bf26-9266cc5e9f3	Osanayora	/ava/1bf6f5b3-3635-4d83-bf26-9266cc5e9f3.jpeg	false	
5c81649e-d46a-4619-9d56-301bbbf4f22bb	Kitohaman	/ava/5c81649e-d46a-4619-9d56-301bbbf4f22bb.jpeg	false	
cd18b6a9-a52e-4a9b-b3e7-657f855fa6b1	Kusoneyuwo	/ava/cd18b6a9-a52e-4a9b-b3e7-657f855fa6b1.jpeg	true	
15f4e10b-8c87-4ee1-8a60-a737064d6a3e	Kushihamuro	/ava/15f4e10b-8c87-4ee1-8a60-a737064d6a3e.jpeg	false	
245b3f9e-6996-47e8-a89b-c00dc231c2cb	Osehimura	/ava/245b3f9e-6996-47e8-a89b-c00dc231c2cb.jpeg	false	
26669ca9-f3af-4fc1-8de7-da73796e82f2	Otanuyuri	/ava/26669ca9-f3af-4fc1-8de7-da73796e82f2.jpeg	true	
37410f26-f884-4703-a03a-4458e42d45e4	Osoheyuri	/ava/37410f26-f884-4703-a03a-4458e42d45e4.jpeg	false	
1b26cd28-e8e5-4d1c-b85f-a83822e0072e	Atefumuro	/ava/1b26cd28-e8e5-4d1c-b85f-a83822e0072e.jpeg	true	
e40a9f16-ce40-4dff-a84e-f83b22ae665f	Kotefumori	/ava/e40a9f16-ce40-4dff-a84e-f83b22ae665f.jpeg	true	
5e19672a-da69-4ecc-89d6-2e3ab3aebf1e	Ketonuyora	/ava/5e19672a-da69-4ecc-89d6-2e3ab3aebf1e.jpeg	true	
5e55f2a4-34f9-4400-9f87-b7f36907c1dc	Utsuhemewa	/ava/5e55f2a4-34f9-4400-9f87-b7f36907c1dc.jpeg	false	
5cbe7f45-5159-42c4-a1e9-1c601797b26a	Isenimowa	/ava/5cbe7f45-5159-42c4-a1e9-1c601797b26a.jpeg	false	
eeaa7d0a-2997-4d1b-8aef-38bb8341a953	Esanumun	/ava/eeaa7d0a-2997-4d1b-8aef-38bb8341a953.jpeg	true	
4ae1eca8-5c36-4cc8-84e3-03366cbf7a26	Ketsunameri	/ava/4ae1eca8-5c36-4cc8-84e3-03366cbf7a26.jpeg	true	
dacde772-02bc-4dad-8586-bf787a4b041f	Kochihemeri	/ava/dacde772-02bc-4dad-8586-bf787a4b041f.jpeg	false	
d52ee7b7-f66b-4009-8b94-a28df6cad7ba	Atehimero	/ava/d52ee7b7-f66b-4009-8b94-a28df6cad7ba.jpeg	true	
1d135307-c458-4255-b223-6c0367915350	Kechifumuru	/ava/1d135307-c458-4255-b223-6c0367915350.jpeg	true	
babb4712-8c33-41df-9d16-6f6fce973fd3	Keshinumowa	/ava/babb4712-8c33-41df-9d16-6f6fce973fd3.jpeg	true	
df24ffc4-fcac-4b2d-b37f-fa62bcd2fb04	Katanemero	/ava/df24ffc4-fcac-4b2d-b37f-fa62bcd2fb04.jpeg	false	
3212f1a9-1c8e-4fa1-adeb-50da1941577d	Kesenuyora	/ava/3212f1a9-1c8e-4fa1-adeb-50da1941577d.jpeg	true	
8c00dafc-24eb-4a6d-9382-922d31986655	Atanemura	/ava/8c00dafc-24eb-4a6d-9382-922d31986655.jpeg	true	
a71dfaad-a0d5-4b71-b873-646ef1671486	Ketehemiwa	/ava/a71dfaad-a0d5-4b71-b873-646ef1671486.jpeg	false	
d20067c5-048a-45c8-b261-d7d30587631c	Ushiheyuru	/ava/d20067c5-048a-45c8-b261-d7d30587631c.jpeg	true	

Рис 23. Згенеровані дані таблиці Users

post	tier
00023eb0-b9c2-424b-9397-5581c4bf25c8	afe30174-ba39-47bd-978d-4426f240048c
00023eb0-b9c2-424b-9397-5581c4bf25c8	e03d9358-75cf-49f6-b67f-da6cff879e46
000ba5c4-c4c6-4d63-8815-401c96204674	4fc993a3-3a77-4f46-a237-ba1c5ed156b0
000ba5c4-c4c6-4d63-8815-401c96204674	5fcc8d66-a18b-4852-9767-a10423f829a0
000ba5c4-c4c6-4d63-8815-401c96204674	e082f49a-82fa-4AAF-9b0f-f60de2ac3d22
000ba880-25da-4364-9e47-e6667a693950	17e825f9-ccf3-47c2-836a-17189c7a7039
0010e094-5403-400d-ae62-fe998fbe5756	09f2b88d-7c49-4e32-b8d0-af6c91f44752
0010e094-5403-400d-ae62-fe998fbe5756	6f97a1cc-b783-4072-b9ac-86673edec1e4
00110563-06d2-4273-91d4-8b539d93233c	2b9a3602-ad28-435e-a414-bfaa3756a4a7
00110563-06d2-4273-91d4-8b539d93233c	32561e14-7d9d-4ea2-9db3-d6f0c3c59600
00110563-06d2-4273-91d4-8b539d93233c	6aa37f1d-24b9-4633-9b28-a92b158aba1e
00130573-21df-4841-9221-f5494a081fd4	658df64b-37b8-4363-a880-c0404d9e74fc
00130573-21df-4841-9221-f5494a081fd4	d8937c5c-4781-4624-83bf-eaf7741b018a
0013eada-c71c-4010-8f6c-a9ccc3b7b5d2	3dd27eaa-3133-4394-929c-6d59f7b8135b
0013eada-c71c-4010-8f6c-a9ccc3b7b5d2	5e3748e9-e338-4dfe-991d-9fc732de6ae2
0013eada-c71c-4010-8f6c-a9ccc3b7b5d2	99fb9ae8-8689-4d71-b72f-4dac51351da6
0013eada-c71c-4010-8f6c-a9ccc3b7b5d2	9b1ea0d2-31b5-4bcc-883c-8713c8364b33
0013eada-c71c-4010-8f6c-a9ccc3b7b5d2	c8893d8b-25b6-4d83-931f-89b9af081a88
0013eada-c71c-4010-8f6c-a9ccc3b7b5d2	f3c060ec-a140-44d0-91af-5ef8aad8f2d
0016a392-edb0-438e-ab57-45b9fa1b0c82	56532515-b14f-467f-ac48-a8fba9231baf
0016a392-edb0-438e-ab57-45b9fa1b0c82	73417a7c-3003-430f-a2fb-8f7e4efb7965
0016a392-edb0-438e-ab57-45b9fa1b0c82	8b3500b2-0f51-479d-9a1f-3844789359d3
0016a392-edb0-438e-ab57-45b9fa1b0c82	b094ed61-08a6-49aa-8bc1-bd588ea1fb88
0017b573-a38f-450d-90de-6703f506e5db	426053fc-0fb8-4c37-90dd-7a3759317ad0
0017b573-a38f-450d-90de-6703f506e5db	a261aed0-db4b-4042-8b29-50007f01a71e
0017ea28-e7ec-4165-a6d0-05e996a0f608	25446014-8959-4010-ba3f-5e3a6e4de30a
00181f17-f1a3-4846-96a4-5c204db82735	1ed75fac-468e-4c00-8967-5da91ccc3ad8
00181f17-f1a3-4846-96a4-5c204db82735	50ffcb79-8d2d-4000-aa9b-e4526b0a18d0
00181f17-f1a3-4846-96a4-5c204db82735	849a1c4c-8e38-4c81-b1fe-cf369564c532
00181f17-f1a3-4846-96a4-5c204db82735	9ae2e350-4ef4-4115-a2e4-08538148e9d2
001efa81-0efd-49cc-be96-5f6d6847c565	c61d331e-7294-4629-8827-f9e2405013bf
001fa321-aac9-4094-b24b-73ce72c2f961	63671371-1ab8-457b-8445-59e1336a4798
001fa321-aac9-4094-b24b-73ce72c2f961	76b41df4-40cd-40ec-bacb-f5692a20e07c
00218f06-3fee-4b55-b39e-5a99d50c6d84	0ce9c1d3-db5a-4db9-947e-cc45e51fae3c

Рис 24. Згенеровані дані таблиці PostTierRelation

sponsor	tier	expires	transaction
d489779f-beb9-4dd4-853e-3171a8d6b2f0	00e78d7b-760e-480a-a852-d0a8168de0a8	2024-01-08 16:13:22.984005	1
bc6a7a08-a74a-40e3-a14e-0aff2bc8596a	e3272ec9-723f-47b8-b512-b84f898a3982	2024-01-09 16:13:22.984005	2
77dad99d-65fb-4961-a4f9-2f914a0d7b6c	997e85e4-6fbe-4fed-b792-acf40d179a40	2024-01-21 16:13:22.984005	3
0192882d-cc31-42ab-a1eb-0d24251277ab	dd88ef75-0988-48c9-bf8b-97441595ab70	2023-12-30 16:13:22.984005	4
b786f0b8-535b-4d17-b568-860d1501e66a	f134f786-b5e7-411a-a13c-07ff72ea5a16	2024-01-09 16:13:22.984005	5
0ef4f615-d215-48c9-be76-068aacfea181	e45760d2-1965-496d-bed1-28873981b9f0	2024-01-05 16:13:22.984005	6
7f255f21-bd3b-4a88-8894-cc4530908ec8	ebac81e1-b4e8-470a-87ff-a1942a06ef4a	2024-01-08 16:13:22.984005	7
b6058c80-e825-4a22-a3eb-bd30edb44214	6a47d9af-b1ad-457a-9b1f-1fc5489c83a4	2024-01-01 16:13:22.984005	8
2dfe98af-c85c-49cb-9e36-ffb3c41ac034	9732fc04-5645-42c6-a68d-9c3a5494edf1	2024-01-10 16:13:22.984005	9
e9e04775-7830-47fc-a3b0-8ca49706143	fb59b8d6-08a0-48d2-8f49-8a7fb4817a4e	2024-01-22 16:13:22.984005	10
2221a88a-8017-4554-86f5-3fdbcca73e01	4111a17c-2dad-4ce9-81ec-021211ac9971	2024-01-17 16:13:22.984005	11
29cbbb8a-1a37-48b5-a76f-0164f30c5e64	24503bdf-64d4-4808-aa86-aa73f1dd2729	2024-01-25 16:13:22.984005	12
8e2df274-0de1-4f64-8268-b1d6008f3563	9efcf71e-3fbf-4549-b750-ea6c2142d283	2024-01-17 16:13:22.984005	13
ca7491ad-8208-4839-9bff-aa8b96fcce31	8f06b869-40b8-4894-a58f-cc60caad7a98	2023-12-28 16:13:22.984005	14
a6a8a447-1b6f-41b6-9099-b54afc3da5db	5847c71f-c85d-4b8e-95c6-7faab37137f0	2024-01-08 16:13:22.984005	15
4f7d31fc-44a8-406c-b7e7-2e7b41b08b04	71577de2-bc12-435e-a890-a2883db2d8de	2023-12-31 16:13:22.984005	16
3155e6d7-0f79-4e93-8d95-5a12c2df8148	594cb152-44a9-4aab-a8bb-ef6b65c9d856	2024-01-22 16:13:22.984005	17
f377b3dd-93c8-428e-b4f1-980a3102708e	728060b7-96c1-4592-8f16-4dd553d3d17e	2023-12-28 16:13:22.984005	18
d02346e5-6b6f-4bc5-b9e6-20777301aea2	1820de67-3c3e-4172-b807-3fd9ecaab13a	2023-12-30 16:13:22.984005	19
6a04ef6a-8554-49b2-9d12-6e2be5691be9	729f122c-0e7e-4c9c-a93e-f5efa402c7b0	2023-12-29 16:13:22.984005	20
a39c4b16-43ed-41ce-9842-eafead179c07e	33c2fc9c-89e3-447d-b313-ae35e18598ab	2024-01-13 16:13:22.984005	21
316d3430-d5ab-43e8-a0fd-df42e352bd80	b1ceec16-99d5-4abf-9de1-2a9fba76f5da	2024-01-23 16:13:22.984005	22
f97876e8-c8ba-4603-bdc2-9016ee77bf7c	c26d200d-4b65-43e5-9fde-2ef4e6286a8e	2024-01-05 16:13:22.984005	23
fa7d16e4-c9cc-434e-89de-e54296a521eb	4fb4fba-b070-4a7a-bee9-11ae327846ad	2024-01-23 16:13:22.984005	24
04426294-a239-4f17-913e-6b00ef03bed	c637c77c-d986-4b31-9e0c-ca4e849c2f8e	2024-01-25 16:13:22.984005	25
36c5bbdd-3047-42e0-a357-9ede068f7c0c	e1920dc2-bab6-4d98-a7ad-58f95d9a0df6	2024-01-06 16:13:22.984005	26
a8d913f8-8041-4b48-bb41-92092f2f5826	8fe95e43-28d5-4ce2-8e43-e327d25f5993	2024-01-18 16:13:22.984005	27
8db20a0a-a040-48bb-96d8-a3fdbac9c16b	63345498-173b-4d78-a5f7-6681e8e16691	2024-01-14 16:13:22.984005	28
69b69ea6-ad58-4f04-b3ac-0ddaf3780488	24ede474-406e-41d3-83e9-81d931969b34	2024-01-16 16:13:22.984005	29
47110e6d-e268-4da8-924f-c6957c7f96f3	06e8cd23-578a-4a8f-96c9-0172ce6944d0	2024-01-05 16:13:22.984005	30
5356bba9-6cf1-4a6f-aa98-2ef4551f3b40	5bcfdd71-0871-4c1d-a64e-44615b5dfacd	2024-01-20 16:13:22.984005	31
9a19594d-a85f-4044-8387-65fb65bf6889	6816c5bc-062b-4669-9fa8-8dc361acd761	2024-01-17 16:13:22.984005	32
c97d3c16-2b5b-47f3-94cd-0a8e405d91d7	efa10a86-593f-434f-8b89-3b4f47dec02	2024-01-25 16:13:22.984005	33
6eaba3f2-9c0a-49fb-b35d-3ff1da30996a	90af7d0b-6603-49d2-9680-347a513b1e34	2023-12-29 16:13:22.984005	34

Рис 25. Згенеровані дані таблиці Subscriptions

### Завдання 3. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сущностей одночасно

#### Octopus posts

The screenshot shows the Octopus posts application interface. On the left, there is a 'Filter' sidebar with the following settings:

- Name does not begin with:** GEN2
- Minimum length:** 3000

A green 'Apply' button is located below these filters.

The main area is titled 'Posts' and displays search results:

- Post:** DdI535 xjbH  
Is bound to **9** tiers, has length of 3060 bytes.  
tVzHwxXJ4Eys3xVqgM75MZF432  
V9iUZ9ugDa7xE3TdK0S2ibXZG6dUyJlBqgsXqpjolzPvPfwQ6LDep!rE96DsBx  
0Y9WVT?kQ1oTUlqG84C4L5Juul!Ik ngwDSmwB64ZsfEqzjrfyUC!!Q?  
veluSHY8YyaRI!Om6xq7 KxVjsplx8IAxakLW?  
bvPdapde5Sj1f6vDj3hghEzLIFecJ?flY?QqnDbFTwCKTa0?  
XNEU2lY34oFZJUIALcrv09cTfO0XlcgNWUsADwRreM qIS 5ATH  
NhFyZeBJp5yJXvZjBbTLApw66yNsajbn7FFbArb72Df3vS!!oiSjxrQQ5C3EjhC  
U2pQiKTdd8zABRzpoVkBcEcV86ooayj?hTjs?
- UUID:** b0e79c25-8938-46c6-aac8-8e563d5f9f62  
**Creator:** GEN11052
- Post:** SK0 md91reytuWYyU4uNwT d5JlptwSq11  
Is bound to **8** tiers, has length of 3961 bytes.  
ZxzIVL9J nR31URk8o5Klw?  
PGjKvOo3zPYMEL05aQ4AWNuxFuHOh!wWK9lgIRi6MYC60ZJCuvLryBCvk  
mlWuX3e1jl97oYddES4v5UmjDq?rrUBbDzMpkCcvqkCj6xzKGN

Рис 26. Результат первого запросу

Query took 108ms

## Most spending users

Filter

Only count subscriptions that expire after:

Minimum spending:  €

Users

	Kutanumawo [!] Spending: 52€ [!] Subscriptions: 4 UUID: c77b1c6b-93fe-476b-9d1c-8f8e43f68b5b Banned: Yes
	Asuhemen [!] Spending: 52€ [!] Subscriptions: 4 UUID: 3af915d-3554-487d-b804-9acc17bc5e05 Banned: No
	Katehiyara [!] Spending: 46€ [!] Subscriptions: 3 UUID: 2a00036d-d77f-42ad-90be-ce124c953885 Banned: Yes
	Etanimewa [!] Spending: 46€ [!] Subscriptions: 3 UUID: 7b8aab77-4ea8-4d95-959f-b9f9bdeb309d Banned: No

Рис 27. Результат другого запросу

Query took 40ms

## Most spending users

Filter

Only count subscriptions that expire after:

Minimum revenue:  €

Only banned

Creators

Creator: GEN35 [! Subscribers: 4 [! Revenue: 64.00€
Creator: GEN3297 [! Subscribers: 4 [! Revenue: 62.00€
Creator: GEN15909 [! Subscribers: 4 [! Revenue: 61.00€
Creator: GEN15068 [! Subscribers: 4 [! Revenue: 60.00€
Creator: GEN5694 [! Subscribers: 4 [! Revenue: 56.00€

Page 1

Рис 27. Результат третього запросу

## Лістинг

### SQL-запити

	SQL-запити
1 запит	<pre>SELECT     p.* , COUNT(*) AS cnt FROM "Posts" p INNER JOIN "PostTierRelation" r     ON r.post = p."UUID" WHERE p.creator     NOT LIKE 'GEN2%' GROUP BY p."UUID" HAVING     LENGTH(p.content) &gt; 3000 ORDER BY cnt DESC LIMIT 10 OFFSET 0</pre>
2 запит	<pre>WITH sponsors_sequence AS (     SELECT         s.sponsor,</pre>

```

        COUNT(s.sponsor) AS subs,
        SUM(t.price) AS spending
    FROM "Subscriptions" s
    INNER JOIN "Tiers" t
        ON t."UUID" = s.tier
    WHERE
        s.expires > '2014-01-27 00:00:00'
    GROUP BY s.sponsor
    HAVING
        SUM(t.price) IS NOT NULL
        AND SUM(t.price) >= 44::money
    ORDER BY spending DESC
    LIMIT 10
    OFFSET 0
) SELECT
    u.*,
    subs,
    spending
FROM sponsors_sequence x
INNER JOIN "Users" u
    ON u."UUID" = x.sponsor
ORDER BY spending DESC

```

3 запрос

```

SELECT
    cr.nickname AS "creatorNickname",
    COUNT(s.sponsor) AS subs,
    SUM(t.price) AS revenue
FROM "Creators" cr
INNER JOIN "Tiers" t
    ON t.creator = cr.nickname
    AND cr."isBanned"
INNER JOIN "Subscriptions" s
    ON s.tier = t."UUID"
WHERE
    s.expires >'2023-12-27 00:00:00'
GROUP BY cr.nickname
HAVING
    SUM(t.price) > 55::money
ORDER BY revenue DESC
LIMIT 10
OFFSET 0

```

## Код

```

class OlapService
{
    protected \PDO $db;

    public function __construct(\PDO $db)
    {
        $this->db = $db;
    }

    public function getOctopusPosts(
        string $creatorBlackListPrefix, int $minLength, int $page = 1, int $perPage = 10
    ): array
    {
        $offset = ($page - 1) * $perPage;
        $query = <<<EOQ
            SELECT
                p.*, COUNT(*) AS cnt
            FROM "Posts" p
            INNER JOIN "PostTierRelation" r

```

```

        ON r.post = p."UUID"
    WHERE p.creator
        NOT LIKE ?
    GROUP BY p."UUID"
    HAVING
        LENGTH(p.content) > ?
    ORDER BY cnt DESC
    LIMIT $perPage
    OFFSET $offset
EOQ;

$stmt = $this->db->prepare($query);
$stmt->execute(["$creatorBlackListPrefix%", $minLength]);

$results = [];
foreach ($stmt->fetchAll(\PDO::FETCH_NUM) as $row) {
    $post = (new Post)
        ->setUuid($row[0])
        ->setCreator($row[1])
        ->setTitle($row[2])
        ->setContent($row[3]);
    $results[] = [$post, $row[4]];
}

return $results;
}

public function getRichestUsers(\DateTime $expiry, int $minSpending, int $page = 1,
int $perPage = 10): array
{
    $offset = ($page - 1) * $perPage;
    $query = <<<EOQ
        WITH sponsors_sequence AS (
            SELECT
                s.sponsor,
                COUNT(s.sponsor) AS subs,
                SUM(t.price) AS spending
            FROM "Subscriptions" s
            INNER JOIN "Tiers" t
                ON t."UUID" = s.tier
            WHERE
                s.expires > ?
            GROUP BY s.sponsor
            HAVING
                SUM(t.price) IS NOT NULL
                AND SUM(t.price) >= ?
            ORDER BY spending DESC
            LIMIT $perPage
            OFFSET $offset
        ) SELECT
            u.*,
            subs,
            spending
        FROM sponsors_sequence x
        INNER JOIN "Users" u
            ON u."UUID" = x.sponsor
        ORDER BY spending DESC
EOQ;

$stmt = $this->db->prepare($query);
$stmt->execute([$expiry->format('Y-m-d H:i:s'), $minSpending]);

```

```

$results = [];
foreach ($stmt->fetchAll(\PDO::FETCH_NUM) as $row) {
    $user = (new User)
        ->setUuid($row[0])
        ->setDisplayName($row[1])
        ->setAvatarUrl($row[2])
        ->setIsBanned($row[3]);

    $results[] = new SpendingInfo($user, $row[4], (float) substr($row[5], 1));
}

return $results;
}

public function getRichestCreators(
    \DateTime $until, ?bool $banned = NULL, int $minWealth = 0, int $page = 1, int
$pagePerPage = 10
): array
{
    $banClause = "1=1";
    if (!is_null($banned))
        $banClause = ($banned ? '' : 'NOT') . ' cr."isBanned"';

    $offset = ($page - 1) * $pagePerPage;
    $query = <<<EOQ
        SELECT
            cr.nickname AS "creatorNickname",
            COUNT(s.sponsor) AS subs,
            SUM(t.price) AS revenue
        FROM "Creators" cr
        INNER JOIN "Tiers" t
            ON t.creator = cr.nickname
            AND $banClause
        INNER JOIN "Subscriptions" s
            ON s.tier = t."UUID"
        WHERE
            s.expires > ?
        GROUP BY cr.nickname
        HAVING
            SUM(t.price) > ?
        ORDER BY revenue DESC
        LIMIT $pagePerPage
        OFFSET $offset
    EOQ;
}

$stmt = $this->db->prepare($query);
$stmt->execute([$until->format('Y-m-d H:i:s'), $minWealth]);

return $stmt->fetchAll(\PDO::FETCH_CLASS, RevenueInfo::class);
}
}

```

*Табл 3. Список запитів та лістинг*

## Завдання 4. Програмний код виконати згідно шаблону MVC

### Загальні відомості

Код у цій розрахунково-графічній роботі умовно поділений на три шари: робота з даними (Model: Entities + Repositories + DTO), бізнес-логіка (Controller: Services + Controllers) та подання (View):

Model	Entities	Звичайні класи, які не містять жодної поведінки, а мають лише геттери та сеттери, повторюють структуру таблиці.	
	Repositories	Класи для роботи з сущностями (Entities), відповідають за базові операції з базою даних, що стосуються лише однієї сущності: вставка, отримання, оновлення, вилучення.	
Controller	Services	Класи, що містять у собі бізнес-логіку додатку, містять у собі дії, які може виконувати додаток, відповідають також за роботу з базою даних, здебільшого коли запит стосується більш ніж одної сущності або дуже складний.	
	Controllers	Класи, що містять у собі обробники HTTP-запитів, відповідають за виклик методів сервісів та репозиторіїв у потрібному порядку та виведення результатів користувачу через подання або збір даних від користувача.	
View	Templates	Набір файлів із шаблонами, що відповідають за перетворення структури результату, взятої з контролера, у HTML-документ.	

Табл 4. Список шарів у додатку

## Скорочений лістинг

Оскільки код програми є занадто великим та його вставка у цей звіт не є доцільною, нижче наведені лише визначення методів. Повний код доступний за посиланням на GitHub.

```
6 usages 6 inheritors
abstract class AbstractController
{
    1 usage
    protected \PDO $database;

    5 usages 5 overrides
    public function __construct(\PDO $database){...}

    1 usage
    #[NoReturn]
    protected function notFound(): void{...}

    11 usages
    protected function getPage(string $id = 'page'): int{...}

    5 usages
    protected function boolSelect(string $id, array $values, ?bool $default = NULL): ?bool{...}

    9 usages
    protected function catchNotFound(callable $cb, string $exceptionClass): mixed{...}

    27 usages
    protected function addFlash(string $kind, string $message): void{...}

    1 usage
    protected function getFlashes(bool $peek = false): array{...}

    19 usages
    protected function render(string $template, array $data = []): void{...}

    9 usages
    #[NoReturn]
    protected function redirect(string $addr, int $code = 2): void{...}

    9 usages
    #[NoReturn]
    protected function back(string $default = '/'): void{...}
}
```

```
no usages
}final class AnalyticsController extends AbstractController
{
    4 usages
    protected OlapService $olap;

    no usages
}    function __construct(\PDO $database){...}

    no usages
}    function index(): void{...}

    no usages
}    function octopusPosts(): void{...}

    no usages
}    function richestUsers(): void{...}

    no usages
}    function richestCreators(): void{...}
}

no usages
}final class CreatorsController extends AbstractController
{
    protected CreatorRepository $creators;
    protected TierRepository $tiers;

    no usages
}    function __construct(\PDO $database){...}

    no usages
}    public function enumerate(): void{...}

    public function view(string $id): void{...}

    public function edit(string $id): void{...}

    public function create(): void{...}
}
```

```
no usages
final class HomeController extends AbstractController
{
    no usages
    function home(): void{...}

    no usages
    function identicon(string $seed): void{...}
}

no usages
final class PostsController extends AbstractController
{
    protected PostRepository $posts;
    protected CreatorRepository $creators;
    6 usages
    protected FeedService $feed;

    no usages
    function __construct(\PDO $database){...}

    no usages
    public function enumerate(): void{...}

    no usages
    public function feed(): void{...}

    no usages
    public function creatorFeed(): void{...}

    public function view(string $id): void{...}

    public function edit(string $id): void{...}

    public function create(): void{...}

    no usages
    public function bind(string $id): void{...}

    no usages
    public function unbind(string $post, string $tier): void{...}
}
```

```
no usages
final class TiersController extends AbstractController
{
    protected TierRepository $tiers;

    no usages
    function __construct(\PDO $database){...}

    no usages
    function enumerate(): void{...}

    function view(string $id): void{...}

    function edit(string $id): void{...}

    public function create(): void{...}
}
```

```
no usages
}final class UsersController extends AbstractController
{
    protected UserRepository $users;
    6 usages
    protected SubscriptionService $subs;

    no usages
    function __construct(\PDO $database){...}

    no usages
    public function enumerate(): void{...}

    public function view(string $id): void{...}

    public function edit(string $id): void{...}

    public function create(): void{...}

    no usages
    public function sub(string $uid): void{...}

    no usages
    public function editSub(string $tx): void{...}
}
```

```
class Creator
{
    protected ?string $nickname;
    protected string $displayName;
    protected string $avatarUrl;
    protected bool $isBanned;

    public function getNickname(): ?string{...}

    3 usages
    public function setNickname(?string $nickname): Creator{...}

    public function getDisplayName(): string{...}

    3 usages
    public function setDisplayName(string $displayName): Creator{...}

    public function getAvatarUrl(): string{...}

    3 usages
    public function setAvatarUrl(string $avatarUrl): Creator{...}

    public function isBanned(): bool{...}

    1 usage
    public function setIsBanned(bool $isBanned): Creator{...}
}
```

```
|class Post
|{
    protected ?string $UUID = NULL;
    protected string $creator;
    protected string $title;
    protected string $content;

    public function getUuid(): ?string
    {
        return $this->UUID;
    }

    3 usages
    public function setUuid(?string $uuid): Post{...}

    public function getCreator(): string{...}

    3 usages
    public function setCreator(string $creator): Post{...}

    public function getTitle(): string{...}

    4 usages
    public function setTitle(string $title): Post{...}

    3 usages
    public function getContent(): string{...}

    4 usages
    public function setContent(string $content): Post{...}
}
```

```
class Tier
{
    protected ?string $UUID      = NULL;
    protected string $creator    = '';
    protected string $title       = '';
    protected string $description = '';
    protected ?string $price      = NULL;

    public function getUuid(): ?string{...}

    2 usages
    public function setUuid(?string $uuid): Tier{...}

    public function getCreator(): string{...}

    2 usages
    public function setCreator(string $creator): Tier{...}

    public function getTitle(): string{...}

    3 usages
    public function setTitle(string $title): Tier{...}

    public function getDescription(): string{...}

    1 usage
    public function setDescription(string $description): Tier{...}

    public function getPrice(): ?string{...}

    2 usages
    public function setPrice(?string $price): Tier{...}
}
```

```
class User
{
    protected ?string $UUID = NULL;
    protected string $displayName;
    protected string $avatarUrl;
    protected bool $isBanned;

    public function getUuid(): ?string{...}

    1 usage
    public function setUuid(?string $uuid): User{...}

    public function getDisplayName(): string{...}

    2 usages
    public function setDisplayName(string $displayName): User{...}

    public function getAvatarUrl(): string{...}

    2 usages
    public function setAvatarUrl(string $avatarUrl): User{...}

    public function isBanned(): bool{...}

    2 usages
    public function setIsBanned(bool $isBanned): User{...}
}
```

```
4 usages 4 inheritors
└─ abstract class AbstractRepository
{
    protected \PDO $db;

    6 usages
    public function __construct(\PDO $db){...}
}
```

```
6 usages
class CreatorRepository extends AbstractRepository
{
    3 usages
    public function getByNickname(string $nickname): Creator{...}

    1 usage
    public function dropByNickname(string $nickname): bool{...}

    1 usage
    public function fetch(int $page = 1, ?bool $isBanned = NULL, $perPage = 10): array{...}

    public function save(Creator $user): string{...}
}

3 usages
class PostRepository extends AbstractRepository
{
    2 usages
    public function getByUUID(string $uuid): Post{...}

    1 usage
    public function dropByUUID(string $uuid): bool{...}

    1 usage
    public function fetch(int $page = 1, $perPage = 10): array{...}

    no usages
    public function update(Post $post): string{...}

    no usages
    public function insert(Post $post): string{...}

    public function save(Post $post): string{...}
}
```

```
7 usages
class TierRepository extends AbstractRepository
{
    2 usages
    public function getByUUID(string $uuid): Tier{...}

    1 usage
    public function dropByUUID(string $uuid): bool{...}

    2 usages
    public function fetch(
        int $page = 1, ?string $creator = NULL, ?float $price = NULL, ?bool $isFree = NULL, $perPage = 10
    ): array{...}

    no usages
    public function update(Tier $tier): string{...}

    no usages
    public function insert(Tier $tier): string{...}

    public function save(Tier $tier): string{...}
}

3 usages
class UserRepository extends AbstractRepository
{
    2 usages
    public function getByUUID(string $uuid): User{...}

    1 usage
    public function dropByUUID(string $uuid): bool{...}

    1 usage
    public function fetch(int $page = 1, ?bool $isBanned = NULL, $perPage = 10): array{...}

    no usages
    public function update(User $user): string{...}

    no usages
    public function insert(User $user): string{...}

    public function save(User $user): string{...}
}
```

```
3 usages
|class FeedService
{
    protected \PDO $db;

    1 usage
    | public function __construct(\PDO $db){...}

    1 usage
    | public function getTiersForPost(string $post): array{...}

    1 usage
    | public function unbindPost(string $post, string $tier): bool{...}

    1 usage
    | public function bindPost(string $post, string $tier): bool{...}

    1 usage
    | public function fetchPostsForUser(string $user, int $page = 1, int $perPage = 10): array{...}

    1 usage
    | public function fetchCreatorFeed(Creator $creator, int $page = 1, int $perPage = 10): array{...}
}

3 usages
|class OlapService
{
    protected \PDO $db;

    1 usage
    | public function __construct(\PDO $db){...}

    1 usage
    | public function getOctopusPosts(
        |     string $creatorBlackListPrefix, int $minLength, int $page = 1, int $perPage = 10
    ): array{...}

    1 usage
    | public function getRichestUsers(\DateTime $expiry, int $minSpending, int $page = 1, int $perPage = 10): array{...}

    1 usage
    | public function getRichestCreators(
        |     \DateTime $until, ?bool $banned = NULL, int $minWealth = 0, int $page = 1, int $perPage = 10
    ): array{...}
}
```

```
3 usages
class SubscriptionService
{
    protected \PDO $db;

    1 usage
    public function __construct(\PDO $db){...}

    no usages
    public function getByTxId(int $tx): Subscription{...}

    1 usage
    public function dropByTxId(int $tx): bool{...}

    1 usage
    public function updateExpirationByTxId(int $tx, \DateTime $dt): bool{...}

    2 usages
    public function subscribe(string $user, string $tier, \DateTime $expires, int $tx): bool{...}

    1 usage
    public function getSubscriptions(
        string $uid, ?bool $expired = NULL, int $page = 1, int $perPage = 10
    ): array{...}
}
```

4 usages

```
function createCrudRoutes(RouteCollector $r, string $controller): void
{
    $r->addRoute( httpMethod: 'GET', route: '/list', handler: "$controller:enumerate");
    $r->addRoute( httpMethod: 'GET', route: '/@{uuid}', handler: "$controller:view");
    $r->addRoute( httpMethod: 'POST', route: '/@{uuid}', handler: "$controller:edit"); # should also handle removal & creation
    $r->addRoute( httpMethod: 'GET', route: '/new', handler: "$controller:create"); # only form for edit thingy
}

return function (RouteCollector $r): void {
    $r->addRoute( httpMethod: 'GET', route: '/', handler: 'Home:home');
    $r->addRoute( httpMethod: 'GET', route: '/ava/{amogus}.jpeg', handler: 'Home:identicon');

    $r->addGroup( prefix: '/users', fn ($r) => createCrudRoutes($r, controller: 'Users'));
    $r->addGroup( prefix: '/creators', fn ($r) => createCrudRoutes($r, controller: 'Creators'));
    $r->addGroup( prefix: '/posts', fn ($r) => createCrudRoutes($r, controller: 'Posts'));
    $r->addGroup( prefix: '/tiers', fn ($r) => createCrudRoutes($r, controller: 'Tiers'));

    $r->addRoute( httpMethod: 'GET', route: '/posts/feed', handler: 'Posts:feed');
    $r->addRoute( httpMethod: 'GET', route: '/posts/feed/by-creator', handler: 'Posts:creatorFeed');
    $r->addRoute( httpMethod: 'POST', route: '/posts/@{post}/bind', handler: 'Posts:bind');
    $r->addRoute( httpMethod: 'POST', route: '/posts/@{post}/unbind/{tier}', handler: 'Posts:unbind');

    $r->addRoute( httpMethod: 'POST', route: '/users/@{user}/sub', handler: 'Users:sub');
    $r->addRoute( httpMethod: 'POST', route: '/subs/@{tx}', handler: 'Users:editSub');

    $r->addGroup( prefix: '/olap', function (RouteCollector $r): void {
        $r->addRoute( httpMethod: 'GET', route: '', handler: 'Analytics:index');
        $r->addRoute(['GET', 'POST'], route: '/octopus', handler: 'Analytics:octopusPosts');
        $r->addRoute(['GET', 'POST'], route: '/richest/users', handler: 'Analytics:richestUsers');
        $r->addRoute(['GET', 'POST'], route: '/richest/creators', handler: 'Analytics:richestCreators');
    });
};

});
```