



Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»

Лабораторна робота №1
**«Проектування бази даних та ознайомлення з
базовими операціями СУБД PostgreSQL»**

Виконав студент групи: КВ-32

ПІБ: Клубук Максим Віталійович

Перевірив: _____

Київ 2025

Мета роботи

здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL

Варіант

Електронний журнал для ведення особистого щоденника.

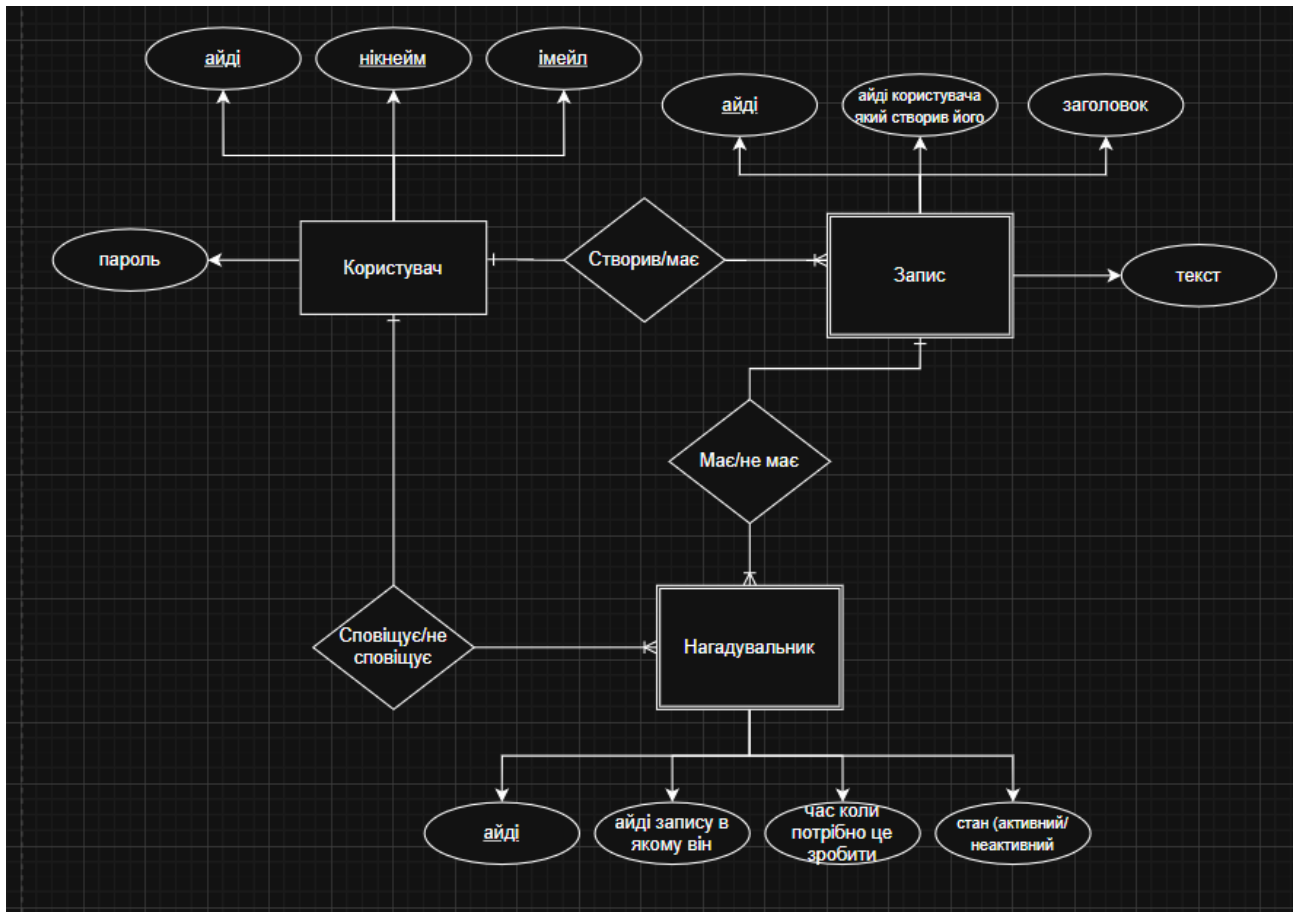
Хід роботи

1. Розробити модель «сутність-зв'язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».

Для початку роботи розберемо всю ситуацію. В нас є користувач, який веде щоденник, тобто може робити багато записів в нього. Це вже дві сутності, сутність «Користувач» та сутність «Запис». Також непогано було б додати нагадувальник для виконання певних записів, наприклад я, як користувач, зробив запис «купити молоко в магазині» і хочу додати нагадування на певний час, коли це зробити, якщо забуду. Тому, додаємо також сутність «Нагадувальник», яка буде прив'язана до певного запису та сповіщати користувача про його обраний запис.

Тепер розберемо атрибути, «користувач» буде складатися з: ідентифікатор (id), ім'я (username), поштової скриньки (email), пароля (password). Сутність «запис» буде складатися з: ідентифікатора (entry_id), ідентифікатор користувача до якого він буде прив'язаний (user_id), заголовка (title), контексту або вмісту запису (text). І ще в нас є «Нагадувальник», тому він буде мати: номер (reminder_id), номер запису в якому він знаходиться (entry_id), час коли сповістити (remind_at), та його стан тобто увімкнений чи вимкнений (active).

Загалом, графічно це виглядатиме ось так:



Скріншот з сайту для розробки діаграм

В овалах описані атрибути, в прямокутниках сутності а ромби – відносини між сутностями. Один користувач має скільки завгодно записів, тобто тут в нас 1:N. Один запис може мати багато нагадувальників, тому також 1:N. Так само і з нагадувальником.

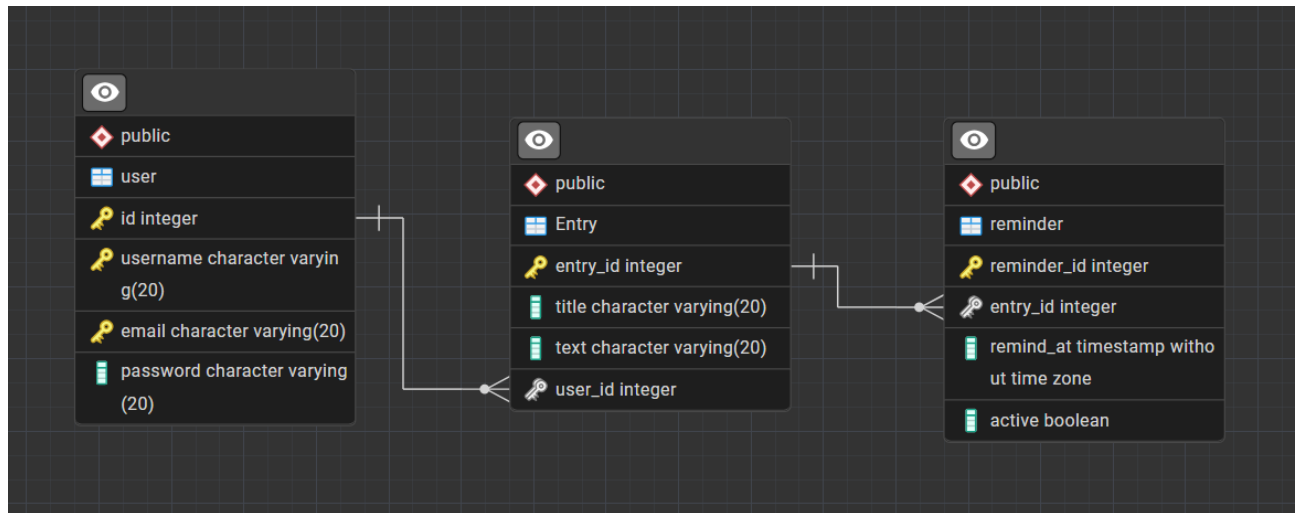
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.

Перейдемо до середовища pgAdmin4. Відкриємо Tools -> ERD Tool та почнемо створювати таблиці. Почнемо з користувача, будемо писати все латиницею для зручності в користуванні в подальшому. Тому створюємо:

Сутність «Користувач» було перетворено в таблицю «user», де атрибутами стали: ідентифікатор (id, integer, PK, UNIQUE, NOT NULL), псевдонім (username, var char, PK, UNIQUE, NOT NULL), поштова скринька (email, var char, PK, UNIQUE, NOT NULL), пароль (password, var char, NOT NULL).

Сутність «Запис» було перетворено в таблицю «entry», де атрибутами стали: Ідентифікатор (entry_id, integer, PK, UNIQUE, NOT NULL), заголовок (title, var char), текст (text, var char), посилання на ідентифікатор користувача (user_id, integer, FK, NOT NULL).

Сутність «Нагадувальник» було перетворено в таблицю «reminder», де атрибутами стали: ідентифікатор (reminder_id, integer, PK, UNIQUE, NOT NULL), посилання на ідентифікатор запису (entry_id, integer, FK, NOT NULL), час коли потрібно нагадати (remind_at, timestamp with time zone), активний чи деактивований стан (active, Boolean, NOT NULL).



3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).

Функціональні залежності:

User: id -> username, email, password.

id -> username.

id -> email.

id -> password.

Entry: entry_id -> title, text, user_id.

entry_id -> title.

entry_id -> text.

entry_id -> user_id.

Reminder: reminder_id -> entry_id, remind_at, active.

reminder_id -> entry_id.

reminder_id -> remind_at.

reminder_id -> active.

Перевірка на 1НФ:

кожне поле містить **атомарні значення** (нема масивів чи списків),

рядки унікальні,

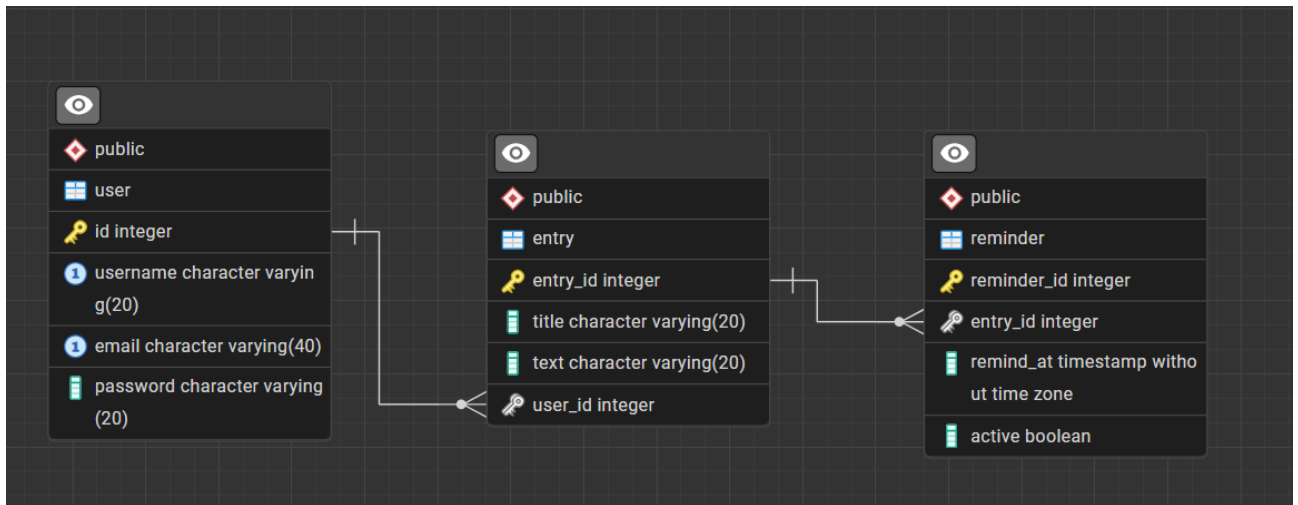
є первинний ключ.

В створених таблицях (user, entry, reminder) є прості поля типу varchar, integer, timestamp, boolean, також кожна таблиця має первинний ключ, та рядки унікальні, що означає, що 1НФ виконується.

Перевірка на 2НФ:

нема часткової залежності атрибутів від частини складного ключа.

У таблиці user був **PRIMARY KEY (username, email, id)**. Це погано, бо складний ключ створює ризик часткової залежності. Тому переробимо user, знявши галочку на PRIMARY KEY в атрибутах email та username.



Перевірка на 3НФ:

нема транзитивних залежностей (атрибут залежить від РК через інший атрибут).

Перевіримо:

user: id -> username, email, password. Нормально.

entry: entry_id -> title, text, user_id. Все ок.





reminder: reminder_id -> entry_id, remind_at, active. Все ок.

4. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4 та внести декілька рядків даних у кожен з таблиць засобами pgAdmin 4.

user

GeneralColumnsAdvancedConstraintsParametersSecuritySQL

Primary KeyForeign KeyCheckUniqueExclude

	Name	Columns
 	email	email
 	username	username

Close

Reset









Save

entry

GeneralColumnsAdvancedConstraintsPartitionsParametersSecuritySQL

Inherited from table(s)

Select to inherit from...

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
 	entry_id	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
 	title	character varying	20		<input type="checkbox"/>	<input type="checkbox"/>	'No title':
 	text	character varying	20		<input type="checkbox"/>	<input type="checkbox"/>	'No text':
 	user_id	integer			<input type="checkbox"/>	<input type="checkbox"/>	

Close

Reset

Save

Посилання на репозиторій Github:

<https://github.com/reinlaught/data-bases-kpi>