

**Національний технічний університет України  
“Київський політехнічний інститут імені Ігоря  
Сікорського”**

**Факультет прикладної математики  
Кафедра системного програмування і спеціалізованих  
комп’ютерних систем**

**ЛАБОРАТОРНА РОБОТА № 1**

***з дисципліни***

***“Бази даних і засоби управління”***

**Група: KB-03**

**Виконав: Палажченко Максим**

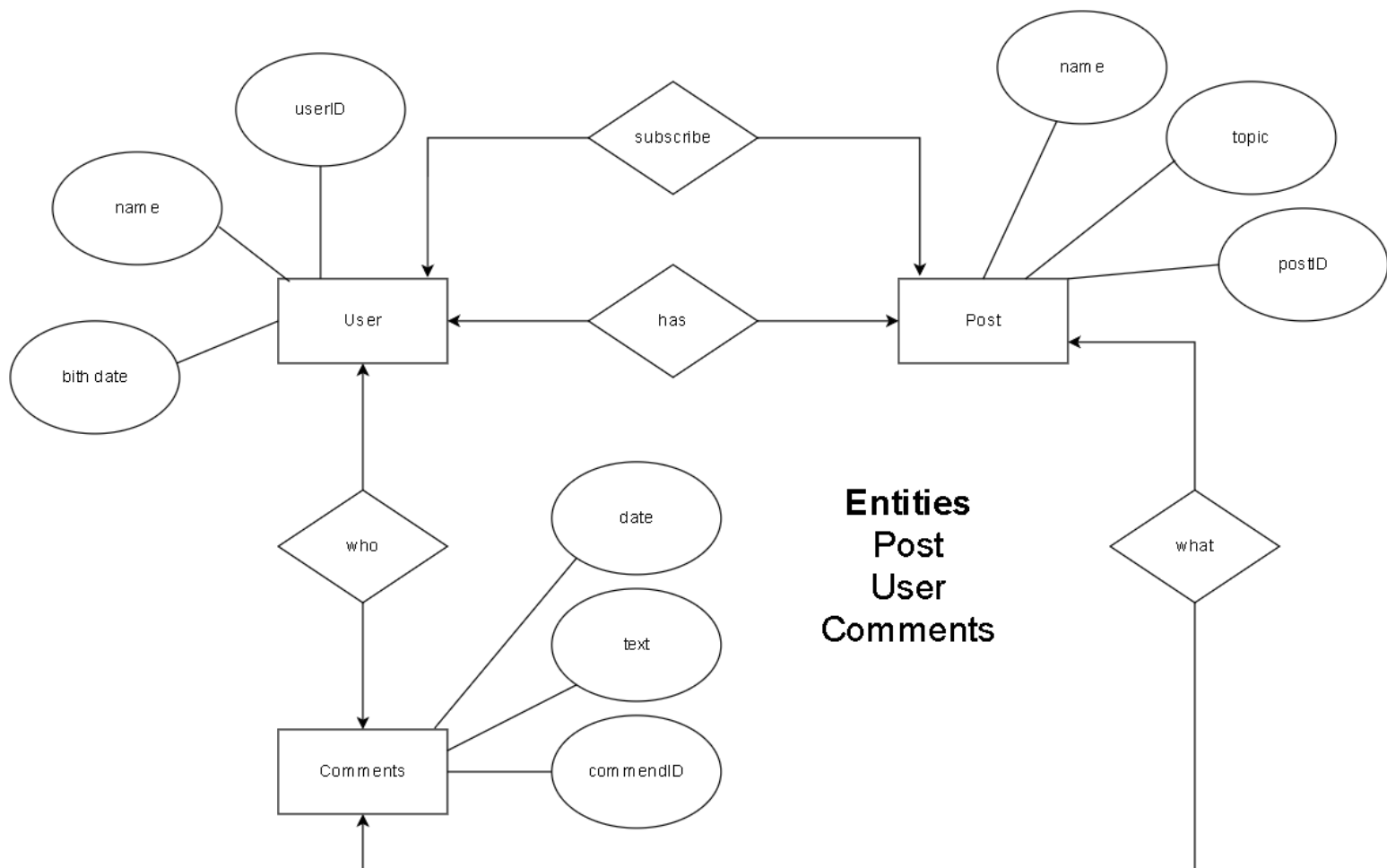
**Оцінка:**

*Метою роботи є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.*

*Завдання роботи полягає у наступному:*

1. Розробити модель «сутність-зв'язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.
3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).
4. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4 та внести декілька рядків даних у кожен з таблиць засобами pgAdmin 4

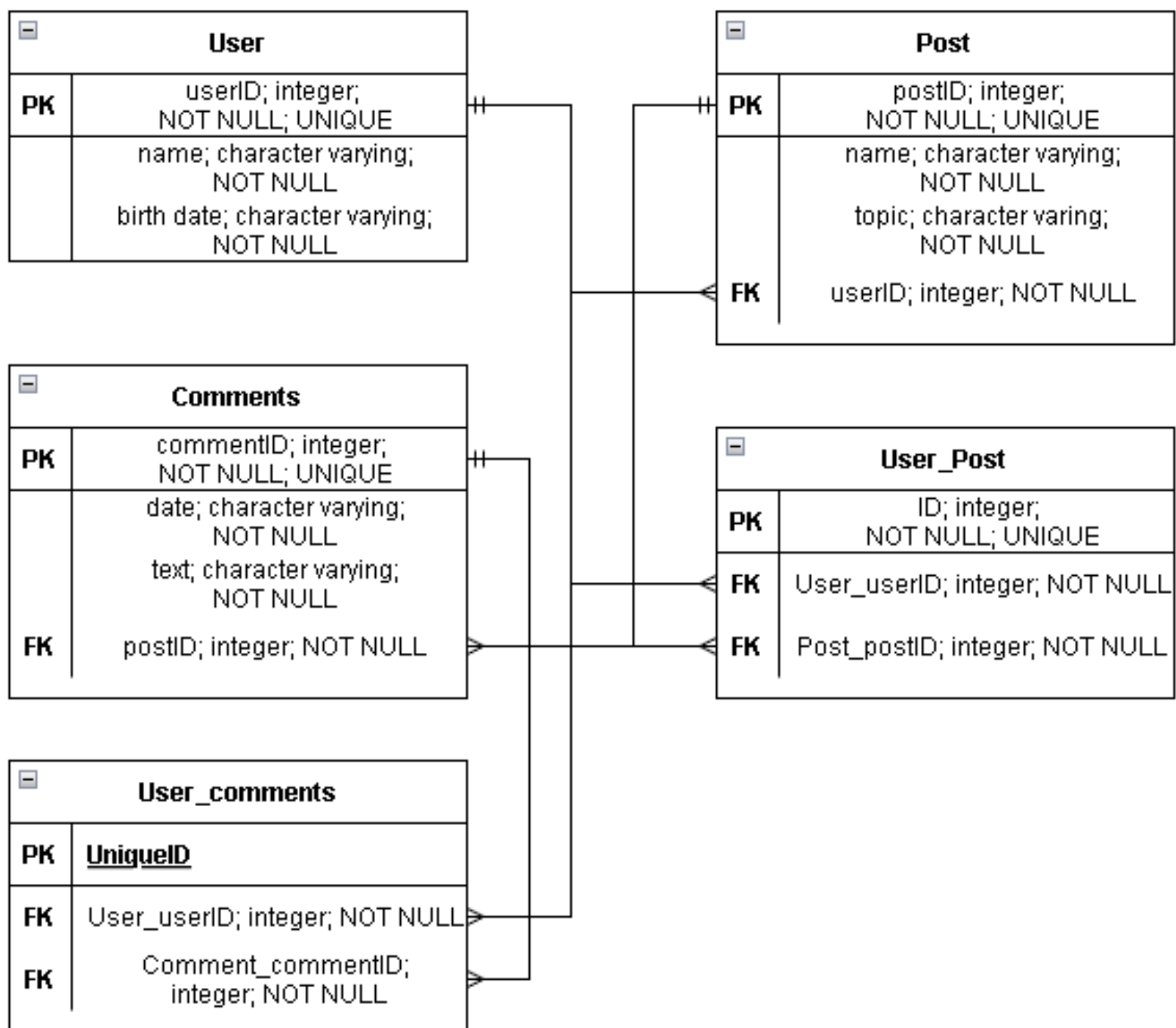
## Завдання 1 - модель «сутність-зв'язок» галузі постів соц. мережі



### Сутності, їх призначення та опис в'язків:

Маємо три сутності: User, Post, Comments. Сутність User описує користувачів соціальної мережі. Кожен користувач має ім'я, дату народження та свій ID. Сутність Post – це пости цих самих користувачів. Тобто кожен окремий користувач може мати скільки завгодно постів в своєму блозі, але скільки завгодно користувачів можуть бути підписаними на ці пости. Кожний пост також має свій ID, назву поста та тему, на яку написано пост. Третя сутність – це Comments. Це коментарі під постами. Коментар має два зв'язки: чий коментар (користувача) та що це за коментар (тобто під яким саме постом). Багато користувачів можуть написати скільки завгодно коментарів, а також під одним постом може бути багато коментарів. Коментар має свій ID, свою дату та сам текст.

## Завдання 2 – перетворення графічного вигляду бази даних у схему бази даних PostgreSQL



## Опис процесу перетворення

Сутність User було перетворено в таблицю User. Сутність Post було перетворено в таблицю Post, зв'язок has (1:N) із сутністю User зумовив появу у ній зовнішнього ключа userID. Сутність Comments було перетворено в таблицю Comments, зв'язок what (1:N) із сутністю Post зумовив появу у ній зовнішнього ключа postID. Зв'язок subscribe (N:M) зумовив появу додаткової таблиці User\_Post. Зв'язок who (N:M) зумовив появу додаткової таблиці User\_Comments.

**Завдання 3** - нормалізацію схеми бази даних до третьої нормальної форми

**Функціональні залежності:**

User (*userID*, name, birth date):

$userID \rightarrow name$

$userID \rightarrow birth\ date$

$userID \rightarrow name, birth\ date$

Post (*postID*, userID, name, topic):

$postID \rightarrow userID$

$postID \rightarrow name$

$postID \rightarrow topic$

$postID \rightarrow userID, name, topic$

Comments (*commentID*, postID, date, text):

$commentID \rightarrow postID$

$commentID \rightarrow date$

$commentID \rightarrow text$

$commentID \rightarrow postID, date, text$

User\_Post (User\_userID, Post\_postID, *ID*):

$ID \rightarrow Post\_postID$

$ID \rightarrow User\_userID$

$ID \rightarrow Post\_postID, User\_userID$

User\_Comments (User\_userID, Comment\_commentID, *ID*):

$ID \rightarrow Comment\_commentID$

$ID \rightarrow User\_userID$

$ID \rightarrow Comment\_commentID, User\_userID$

Схема бази даних відповідає нормальній формі НФ1, тому що всі атрибути таблиці є атомарними, кожна таблиця має primary key та кожна таблиця має мінімальний набір атрибутів.

Схема бази даних відповідає нормальній формі НФ2, тому що вона відповідає нормальній формі НФ1 і кожен неключовий атрибут функціонально залежить від цілого ключа, а не від його частини. Схема бази даних відповідає нормальній формі НФ3, тому що вона відповідає нормальній формі НФ2 і дані в таблиці залежать від primary key.

## Завдання 4 – таблиці бази даних у pgAdmin 4

User:

> Indexes		<b>userID</b> [PK] integer	<b>name</b> character varying	<b>birth date</b> character varying
> RLS Policies	1	1	Oleg	27th of March 20...
> Rules	2	2	Max	12th of March 20...
> Triggers	3	3	Ivan	14 of April 2002

User

General Columns Advanced Constraints Parameters Security SQL

Inherited from table(s)

Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
	userID	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
	name	character varying			<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	birth date	character varying			<input checked="" type="checkbox"/>	<input type="checkbox"/>	

User

General Columns Advanced Constraints Parameters Security SQL

Primary Key Foreign Key Check Unique Exclude

Name	Columns
User_pkey	userID

Post:

> Indexes		<b>postID</b> [PK] integer	<b>name</b> character varying	<b>topic</b> character varying	<b>userID</b> integer
> RLS Policies	1	1	sport	about the gold b...	2
> Rules	2	2	war in Ukraine	with drawal of tro...	2
> Triggers	3	3	films	about the serial A...	1

Post

General Columns Advanced Constraints Parameters Security SQL

Inherited from table(s)

Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
	postID	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
	name	character varying			<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	topic	character varying			<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	userID	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Post

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

		Name	Columns
		Post_pkey	postID

Post

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

		Name	Columns	Referenced Table
		userID	(userID) -> (userID)	public.User

## Comments:

Operators

Procedures

Sequences

Tables (5)

Comments

	commentID [PK] integer	date character varying	text character varying	postID integer
1	1	4th of September...	Where is Messi?	1
2	2	1th of October 20...	Putin huilo	2
3	3	15of November 2...	wasd	3











Comments

General
Columns
Advanced
Constraints
Parameters
Security
SQL

Inherited from table(s)
Select to inherit from...



Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
 	commentID	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
 	date	character varying			<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	text	character varying			<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	postID	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Comments

General
Columns
Advanced
Constraints
Parameters
Security
SQL



Primary Key
Foreign Key
Check
Unique
Exclude

	Name	Columns
 	Comments_pkey	commentID

Comments

General
Columns
Advanced
Constraints
Parameters
Security
SQL

Primary Key
Foreign Key
Check
Unique
Exclude

	Name	Columns	Referenced Table
 	postID	(postID) -> (postID)	public.Post

## User\_Post:

- Indexes
- RLS Policies
- Rules
- Triggers
- User\_Post







	ID [PK]	Post_postID	User_userID
1	1	1	3
2	2	2	2
3	3	3	2

User\_Post

General
Columns
Advanced
Constraints
Parameters
Security
SQL

Inherited from table(s)
Select to inherit from...

Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
 	ID	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
 	Post_postID	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	User_userID	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	



General Columns Advanced Constraints Parameters Security SQLPrimary Key Foreign Key Check Unique Exclude

				+	
		Name	Columns	Referenced Table	
		Comment_commentID	(Comment_commentID) -> (commentID)	public.Comments	
		User_userID	(User_userID) -> (userID)	public.User	