A black and white drawing of a building

Description automatically generated

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського» ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота №1**

з дисципліни **Бази даних і засоби управління**

*на тему: “Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL”*

Виконав:

студент ІІI курсу

групи КВ-12

Зимовець Є.О.

Перевірив:

Павловский В. І.

Київ – 2023

**Мета:** здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

**Завдання:**

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 кожну з таблиць засобами pgAdmin 4.

**Опис предметної області**

Дана предметна область – електронний журнал для батьків та вчителів. Електронний журнал для батьків та вчителів - це інформаційна система або програмний продукт, який призначений для спрощення та покращення комунікації між батьками та вчителями щодо навчання і успішності дітей.

**Опис сутностей**

Для побудови бази даних обраної області, були виділені такі сутності:

1. Предмет(Subject)

Атрибути: subject\_id,teacher\_id,name

Призначення: зберігає інформацію про предмет

1. Вчителі(Teacher)

Атрибути: teacher\_id,name,surname,number

Призначення: зберігає інформацію про вчителів

1. Журнал(Classbook)

Атрибути: teacher\_id,student\_id,date,mark,subject\_id,note\_id

Призначення: зберігати інформацію про успішність учнів

1. Член сім’ї(Family\_member)

Атрибути:family\_member\_id,name,surname,number,student\_id

Призначення: зберігає інформацію про людтну відповідальну за учня

1. Учень(Student)

Атрибути: student\_id, name,surname,class

Призначення: зберігає інформацію про учня

**Опис зв’язків між сутностями**

Зв’язок “Family\_member” - “Classbook” 1:N,тому що журнал один,але в ньому багато членів різних сімей.

Зв’язок “Classbook” – “Teacher” 1:N,тому що журнал один,але багато вчителів вносять в нього зміни.

Зв’язок “Teacher” – “Subject” 1:1,тому що один викладач викладає один предмет.

Зв’язок “Subject” – “Student” 1:N,тому що один учень вчить багато предметів.

Зв’язок “Family\_member” – “Student” 1:N,тому що один член сім’ї має одного або декількох дітей.

Графічне подання концептуальної моделі «Сутність-зв’язок» зображено на рисунку 1.

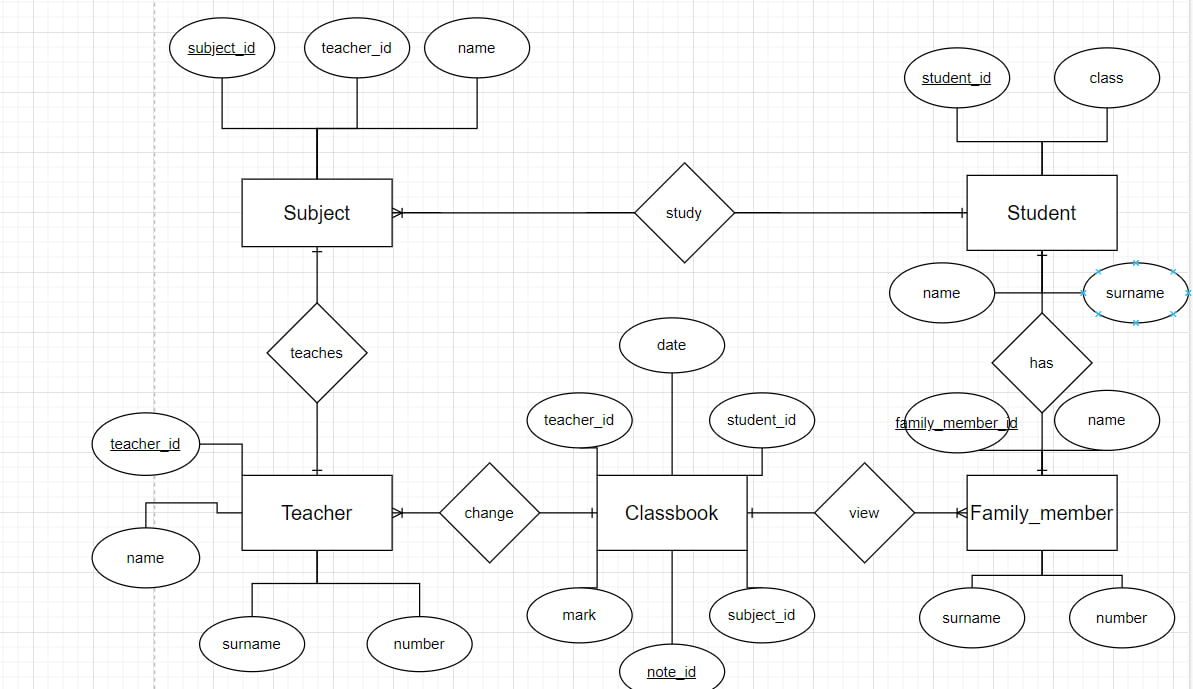


Рисунок 1 – ER-діаграма, побудована за нотацією Чена

**Перетворення концептуальної моделі у логічну модель та схему бази даних**

Сутність Subject перетворено в таблицю Subject з первинним ключем subject\_id та атрибутами name та зовнішнім ключем teacher\_id

Сутність Teacher перетворено в таблицю Teacher з первинним ключем teacher\_id та атрибутами name,surname,number.

Сутність Classbook перетворено в таблицю Classbook з первинним ключем note\_id та атрибутами mark, date та зовнішніми ключами teacher\_id,student\_id, subject\_id.

Сутність Family\_member перетворено в таблицю Family\_member з первинним ключем family\_member\_id та атрибутами name,surname,number,та зовнішнім ключем student\_id.

Сутність Student перетворено в таблицю Student з первинним ключем student\_id та атрибутами class,name,surname.

Графічне подання логічної моделі «Сутність-зв’язок» зображено на рисунку 2.

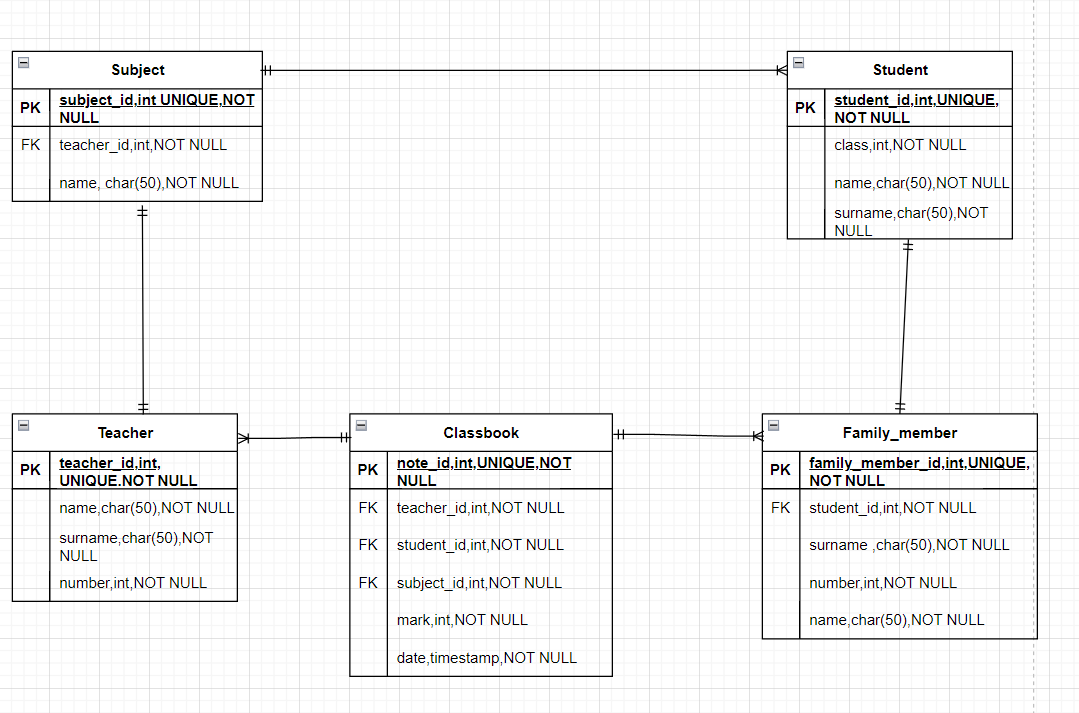


Рисунок 2 – Схема бази даних

Таблиця 1 ілюструє детальний перехід від однієї моделі до іншої.

Таблиця 1 – Опис об’єктів бази даних

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| Subject - містить інформацію про предмет | subject\_id – унікальний ідентифікатор предмету | integer |
| teacher\_id – ідентифікатор вчителя що викладає предмет | integer |
| name – назва предмету | character |
| Teacher - містить інформацію про вчителів | teacher\_id – унікальний ідентифікатор вчителя | integer |
| name – ім’я вчителя | character |
| surname -прізвище вчителя | character |
| number – номер телефону вчителя | integer |
| Classbook – містить інформацію про успішність учнів | note\_id – унікальний ідентифікатор запису | integer |
| student\_id – ідентифікатор учня | integer |
| teacher\_id – ідентифікатор вчителя | integer |
| subject\_id – ідентифікатор предмету | integer |
| mark – оцінка учня по певному предмету | integer |
| date – дата коли буда виставленаоцінка | data |
| Family\_member – містить інформацію про члена сім’ї | family\_member\_id – унікальний ідентифікатор члена сім’ї | integer |
| name – ім’я члена сім’їч | character |
| surname – прізвище члена сім’ї | character |
| number – номер телефону члена сім’ї | integer |
| student\_id – унікальний id учня,чиїм родичем є цей член сім’ї | character |
| Student – містить інформацію про учня | student\_id – унікальний ідентифікатор учня | integer |
| class – клас в якому навчається учень | integer |
| name – ім'я учня | character |
| surname – прізвище учня | character |

**Функціональні залежності для кожної таблиці**

1. Subject

subject\_id -> {teacher\_id,name}

teacher\_id -> { subject\_id , name }

1. Teacher

teacher\_id -> { name, surname,number}

1. Classbook

note\_id -> { teacher\_id,student\_id, subject\_id,mark,date}

teacher\_id, subject\_id,student\_id -> {note\_id, date, mark}

1. Family\_memeber

family\_member\_id -> {name,surname,number,student\_id}

student\_id -> {name,surname,number, family\_member\_id }

1. Student

student\_id -> {name,surname,class}

Ці функціональні залежності вказують на те, які атрибути в кожній таблиці визначаються від інших атрибутів. Це важливо для нормалізації та управління базою даних.

**Відповідність схеми нормальним формам**

1. Щоб задовільнити умови 1НФ кожен атрибут в таблиці має бути атомарним, тобто:

* Кожна клітинка містить єдине значення;
* Кожен запис є унікальним.

Дана схема відповідає 1НФ.

1. Щоб схема відповідала 2НФ повинні виконуватись умови:

* Схема перебуває в 1НФ;
* Кожний неключовий атрибут функціонально залежить від цілого ключа.

У даній схемі кожна таблиця має власний унікальний ідентифікатор (ключ). Кожний атрибут у кожній таблиці залежить від цього унікального ідентифікатора. Тобто, схема також в НФ2.

1. Щоб схема відповідала 3НФ повинні виконуватись умови:

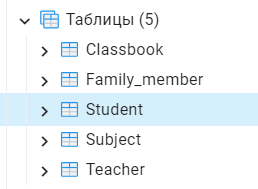
* Схема перебуває в 2НФ;
* Кожен неключовий атрибут нетранзитивно залежить від кожного ключа.

Оскільки дана схема в НФ2 та неключові атрибути не транзитивно залежать від інших неключових атрибутів, схема також в НФ3.

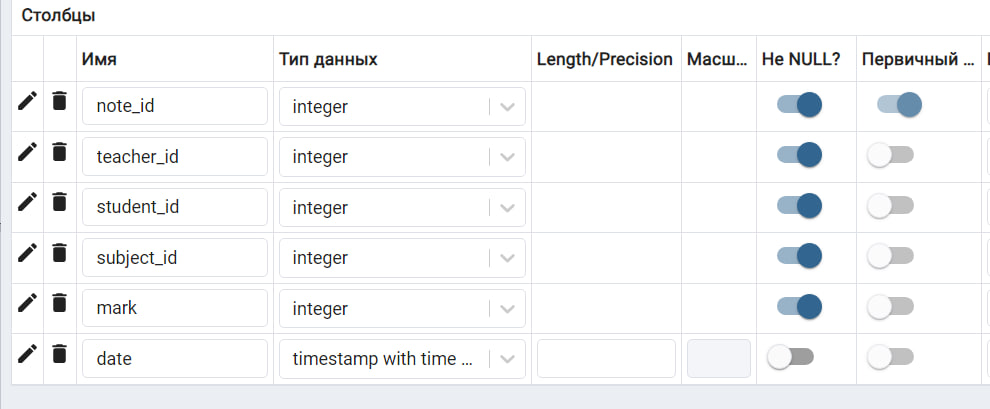
Висновок:

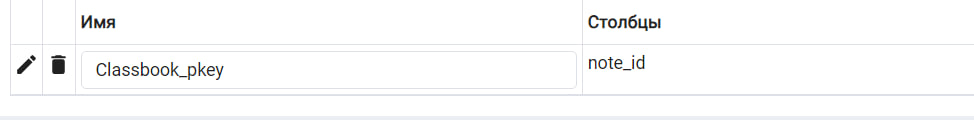
Схема бази даних відповідає нормальним формам НФ1, НФ2 та НФ3. Вона добре структурована і нормалізована, що сприяє ефективному та надійному зберіганню та обробці даних.

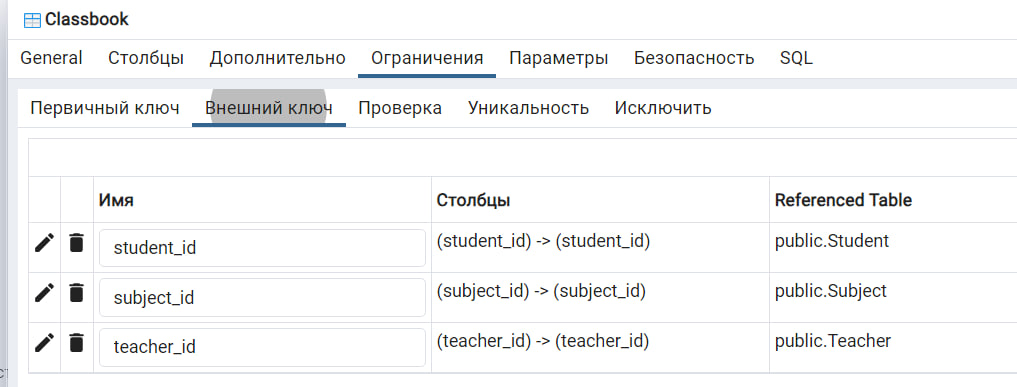
**Таблиці бази даних у pgAdmin4**

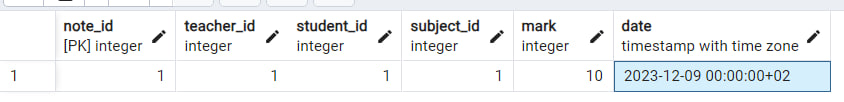


**Classbook**

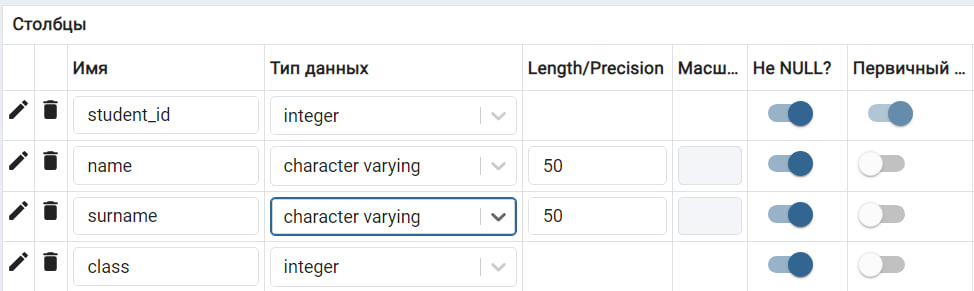


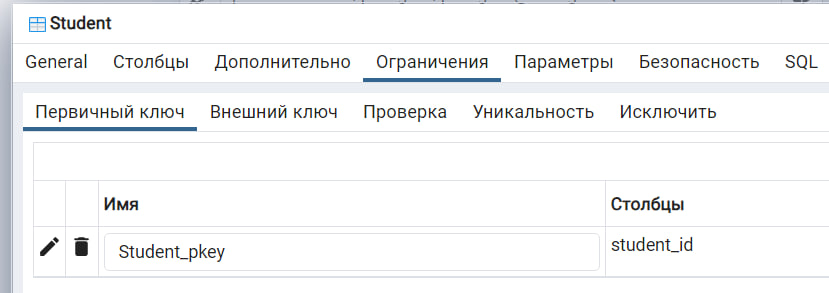


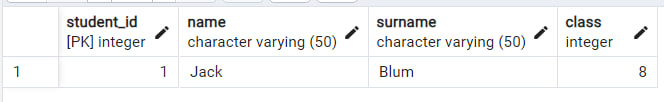




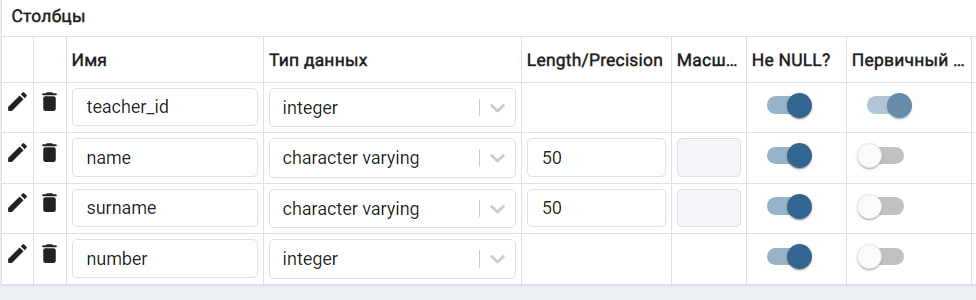
**Student**

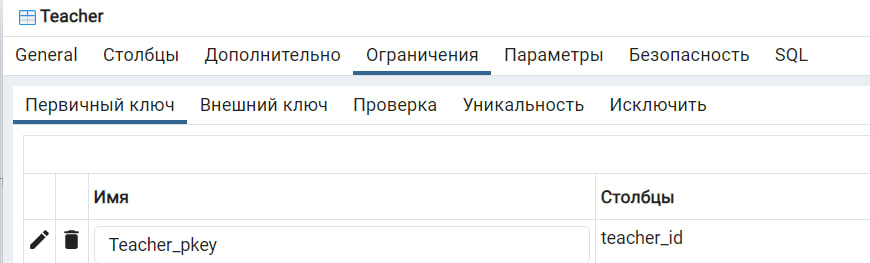


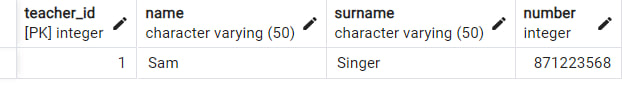




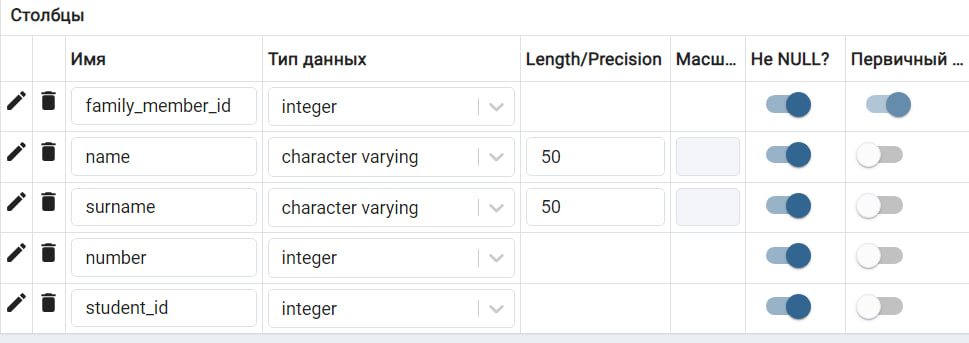
**Teacher**

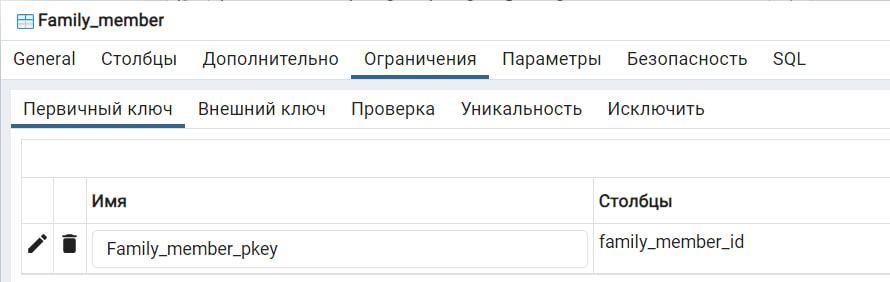


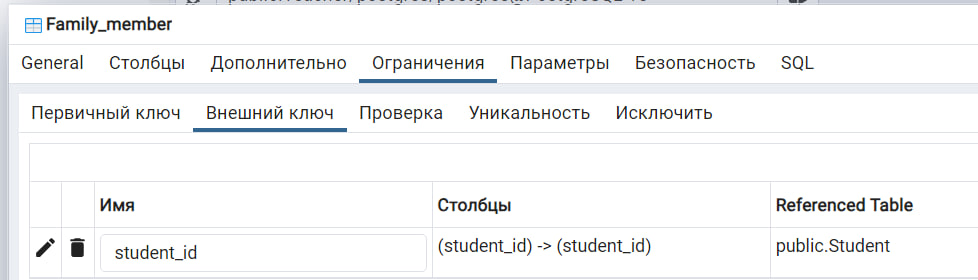


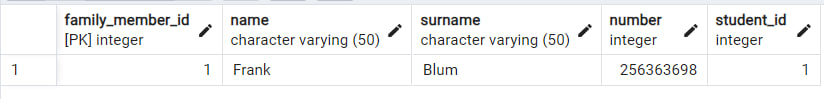


**Family\_member**

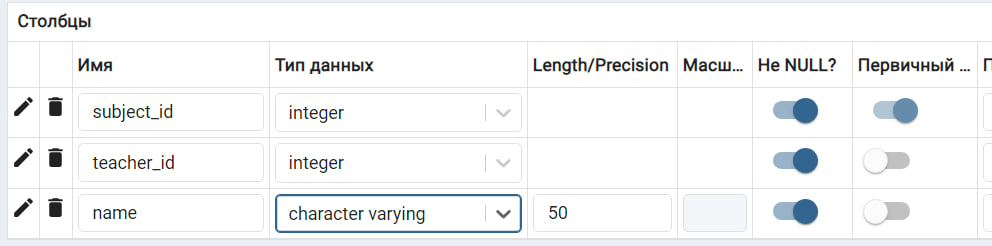


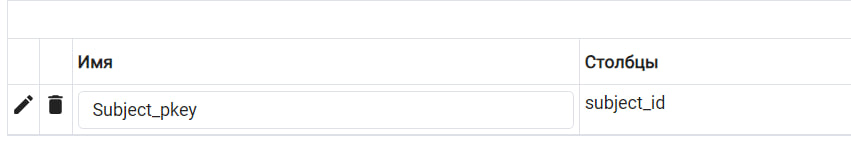


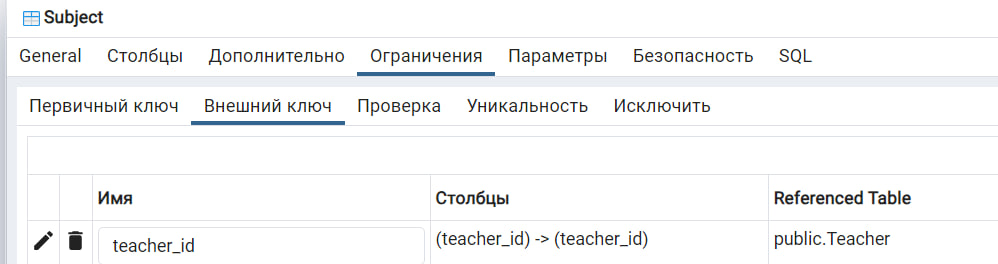




**Subject**









**Таблиці в коді SQL**

-- Table: public.Classbook

-- DROP TABLE IF EXISTS public."Classbook";

CREATE TABLE IF NOT EXISTS public."Classbook"

(

note\_id integer NOT NULL,

teacher\_id integer NOT NULL,

student\_id integer NOT NULL,

subject\_id integer NOT NULL,

mark integer NOT NULL,

date timestamp with time zone,

CONSTRAINT "Classbook\_pkey" PRIMARY KEY (note\_id),

CONSTRAINT student\_id FOREIGN KEY (student\_id)

REFERENCES public."Student" (student\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID,

CONSTRAINT subject\_id FOREIGN KEY (subject\_id)

REFERENCES public."Subject" (subject\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID,

CONSTRAINT teacher\_id FOREIGN KEY (teacher\_id)

REFERENCES public."Teacher" (teacher\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Classbook"

OWNER to postgres;

-- Table: public.Family\_member

-- DROP TABLE IF EXISTS public."Family\_member";

CREATE TABLE IF NOT EXISTS public."Family\_member"

(

family\_member\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

surname character varying(50) COLLATE pg\_catalog."default" NOT NULL,

"number" integer NOT NULL,

student\_id integer NOT NULL,

CONSTRAINT "Family\_member\_pkey" PRIMARY KEY (family\_member\_id),

CONSTRAINT student\_id FOREIGN KEY (student\_id)

REFERENCES public."Student" (student\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Family\_member"

OWNER to postgres;

-- Table: public.Student

-- DROP TABLE IF EXISTS public."Student";

CREATE TABLE IF NOT EXISTS public."Student"

(

student\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

surname character varying(50) COLLATE pg\_catalog."default" NOT NULL,

class integer NOT NULL,

CONSTRAINT "Student\_pkey" PRIMARY KEY (student\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Student"

OWNER to postgres;

-- Table: public.Subject

-- DROP TABLE IF EXISTS public."Subject";

CREATE TABLE IF NOT EXISTS public."Subject"

(

subject\_id integer NOT NULL,

teacher\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT "Subject\_pkey" PRIMARY KEY (subject\_id),

CONSTRAINT teacher\_id FOREIGN KEY (teacher\_id)

REFERENCES public."Teacher" (teacher\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Subject"

OWNER to postgres;

-- Table: public.Teacher

-- DROP TABLE IF EXISTS public."Teacher";

CREATE TABLE IF NOT EXISTS public."Teacher"

(

teacher\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

surname character varying(50) COLLATE pg\_catalog."default" NOT NULL,

"number" integer NOT NULL,

CONSTRAINT "Teacher\_pkey" PRIMARY KEY (teacher\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Teacher"

OWNER to postgres;

**Висновки**

У ході виконання лабораторної роботи була розроблена база даних для електронного журналу для батьків та вчителів. База даних включає такі сутності: "Classbook", "Student", "Teacher", "Subject" та "Family\_member". Кожна з цих сутностей має відповідні атрибути, які дозволяють зберігати та керувати інформацією про регіони, індикатори, наглядачами і звітами.

ER-діаграма бази даних була підготовлена для візуального відображення взаємозв'язків між сутностями. Нотація Чена використана для позначення зв'язків та атрибутів.

Схема бази даних пройшла аналіз на відповідність нормальним формам НФ1, НФ2 та НФ3. Всі таблиці відповідають цим нормальним формам, що гарантує надійне та ефективне зберігання даних.

На останньому етапі було надано копії екранів з pgAdmin4, що демонструють властивості стовпців та обмеження, а також вміст таблиць бази даних у PostgreSQL.

В цілому, розроблена база даних відповідає поставленим завданням та вимогам.