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”*

Виконала:

студентка ІІI курсу

групи КВ-34   
Понамарьова Сніжана

Перевірив:

Павловский В. І.

Київ – 2025

**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**

Електронний журнал для батьків та вчителів — предметна галузь, що описує структуру зберігання й обміну даними навчального процесу: записи про оцінки, відвідуваність, предмети. У центрі моделі — зв’язок кожного запису з конкретним учнем і його контактами (батьками), а також прив’язка записів до відповідних вчителів і предметів. Така предметна галузь формалізує, які саме дані потрібно зберігати і як вони взаємопов’язані, щоб забезпечити коректне відображення інформації про успішність і відвідування.

**Опис сутностей**

1. Учень (Student)

Атрибути: ім’я, прізвище, дата народження, клас, електронна пошта.

**Призначення:** зберігає основну інформацію про учня — персональні дані та належність до класу.

1. Батьки (Parents)

Атрибути: ім’я, прізвище, мобільний телефон, електронна пошта.

**Призначення:** зберігає контактні дані батьків і як доступ до інформації про успішність та відвідуваність їхніх дітей.

1. Вчитель (Teacher)

Атрибути: ім’я, прізвище, електронна пошта.

**Призначення:** зберігає дані викладача.

1. Предмет (Subject)

Атрибути: назва.

**Призначення:** зберігає дані предмета.

1. Журнал (Journal)

Атрибути: дата запису, оцінка, статус відвідування.

**Призначення:** зберігає дані про навчальні події: оцінки та відвідуваність і зв’язують їх із конкретними учнями.

**Опис зв’язків між сутностями**

**Зв’язок-сутність: «Батьки-Учень». Зв’язок 1:N.** Батьки можуть бути відповідальним за кількох учнів (наприклад, мати/тато декількох дітей), а один учень може мати тільки одних батьків.

Зв’язок-сутність: «Учень-Журнал». Зв’язок 1:N. Для кожного учня є багато записів у журналі (різні дати, предмети, оцінки, відвідування). Кожен запис журналу прив’язаний до одного учня.

Зв’язок-сутність: «Вчитель-Журнал». Зв’язок 1:N. Кожен запис прив’язаний до одного вчителя, але вчитель має багато записів.

Зв’язок-сутність: «Предмет-Журнал». Зв’язок 1:N. Один предмет може мати багато записів у журналі (різні дати, різні учні, різні вчителі), а кожен запис журналу відноситься до одного предмета.

У зв’язку «Журнал» реалізовано 3-хмістне відношення: Вчитель–Учень–Предмет. Тобто журнал не просто зв’язує учня з вчителем, а також фіксує, з яким предметом пов’язана ця взаємодія.

Изображение выглядит как зарисовка, рисунок, диаграмма, Штриховая графика

Содержимое, созданное искусственным интеллектом, может быть неверным.

*Рисунок 1 – ER-діаграма, побудована за нотацією Чена*

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

Сутність **"Учень"** перетворено в таблицю **Student** з первинним ключем student\_id та атрибутами first\_name, last\_name, birth\_date, class, email. У таблиці є зовнішній ключ parent\_id, який пов'язаний з таблицею **Parents.**

Сутність **"Батьки"** перетворено в таблицю **Parents** з первинним ключем parents\_id та атрибутами first\_name, last\_name, phone, email.

Сутність **"Вчитель"** перетворено в таблицю **Teacher** з первинним ключем teacher\_id та атрибутами first\_name, last\_name, email.

Сутність **"Предмет"** перетворено в таблицю **Subject** з первинним ключем subject\_id та атрибутом name.

Сутність **"Журнал"** перетворено в таблицю **Journal** з первинним ключем journal\_id та атрибутами entry\_date, grade, attendance\_status. У таблиці є зовнішній ключ teacher\_id, який пов'язаний з таблицею **Teacher,** student\_id, який пов’язаний з таблицею Student та subject\_id, який пов’язаний з таблицею Subject

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

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

*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 |
| *Student −* містить дані про учня | *student\_id* − унікальний ідентифікатор учня | integer (числовий) |
| *parents\_id* − ідентифікатор бітьків | integer (числовий) |
| *first\_name* − ім’я учня | character varying (рядок) |
| *last\_name* − прізвище учня | character varying (рядок) |
| *birth\_date* − дата народження учня | date (дата) |
| *class* − клас в якому навчається учень | character varying (рядок) |
| *email* − електронна пошта учня | character varying (рядок) |
| *Parents* − містить дані про батьків | *parents\_id* − унікальний ідентифікатор батьків | integer (числовий) |
| *first\_name* − ім’я батьків | character varying (рядок) |
| *last\_name* − прізвище батьків | character varying (рядок) |
| *phone* − мобільний телефон батьків | character varying (рядок) |
| *email* − електронна пошта батьків | character varying (рядок) |
| *Teacher* − містить дані про вчителя | *teacher\_id* − унікальний ідентифікатор вчителя | integer (числовий) |
| *first\_name* − ім’я вчителя | character varying (рядок) |
| *last\_name* − прізвище вчителя | character varying (рядок) |
| *email* − електронна пошта вчителя | character varying (рядок) |
| *Subject* − містить дані про предмет | *subject\_id* − унікальний ідентифікатор предмету | integer (числовий) |
| *name* − назва предмету | character varying (рядок) |
| *Journal* − містить дані про записи в журналі | *journal\_id* − унікальний ідентифікатор запису журналу | integer (числовий) |
| *student\_id* − ідентифікатор учня | integer (числовий) |
| *teacher\_id* − ідентифікатор вчителя | integer (числовий) |
| *subject\_id* − ідентифікатор предмету | integer (числовий) |
| *entry\_date* − дата запису | date (дата) |
| *grade* − оцінка | integer (числовий) |
| *attendance\_status* − статус відвідуваності | character varying (рядок) |

**Функціональні залежності для кожної таблиці**

1. Parents

parents\_id => {first\_name, last\_name, phone, email}

1. Student

student\_id => {parents\_id, first\_name, last\_name, birth\_date, class, email}

parents\_id => {first\_name, last\_name, phone, email}

1. Teacher

teacher\_id => {first\_name, last\_name, email}

1. Subject

subject\_id=> {name}

1. Journal

journal\_id => {student\_id, teacher\_id, subject\_id, entry\_date, grade, attendance\_status}

student\_id => {first\_name, last\_name, birth\_date, class, email}

teacher\_id => {first\_name, last\_name, email}

subject\_id=> {name}

У моделях реляційної бази даних функціональні залежності показують, які атрибути в кожній таблиці однозначно визначаються іншими атрибутами. Це важливо для нормалізації — коректно встановлені залежності допомагають уникати дублювання даних і зберігати їх цілісність. У практичному сенсі PK кожної таблиці функціонально визначає її власні атрибути (наприклад student\_id → {first\_name, birth\_date, class, email}), а зовнішні ключі показують, як записи пов’язані між таблицями.

Транзитивна функціональна залежність виникає тоді, коли значення одного атрибута визначає інший через проміжний атрибут. Формально: якщо A → B і B → C, то A транзитивно визначає C. Такі зв’язки можуть свідчити про надлишковість даних, якщо результуючі атрибути зберігаються у тій самій таблиці. В межах кожної окремої таблиці транзитивних функціональних залежностей не виявлено — зв’язки між сутностями реалізовано через зовнішні ключі.

**Відповідність схеми нормальним формам**

1. 1NF вимагає:

* щоб кожен атрибут таблиці був атомарним (клітинка містить одне значення);
* кожен запис у таблиці був унікальним.

Усі таблиці моделі відповідають вимогам першої нормальної форми: кожне поле містить атомарне значення, повторних полів нема, а унікальність записів забезпечується первинними ключами. Отже, схема задовольняє умови 1NF.

1. 2NF вимагає:

* що схема повинна бути в в 1NF;
* Кожний непервинний атрибут повинен функціонально залежати від усього первинного ключа, а не тільки від його частини (тобто відсутні часткові залежності).

Модель перебуває в першій нормальній формі та задовольняє вимоги другої нормальної форми: у всіх таблицях первинні ключі є одиничними (відповідно parent\_id, student\_id, teacher\_id, subject\_id, journal\_id) і кожний непервинний атрибут функціонально залежить від усього свого первинного ключа. Таким чином, схема відповідає вимогам 2NF.

1. 3NF вимагає:

* що схема повинна бути в 2NF;
* жоден непервинний атрибут не повинен бути транзитивно залежним від первинного ключа (тобто не повинна існувати ситуація PK => X і X => Y, де Y — непервинний атрибут тієї ж таблиці).

Модель відповідає третій нормальній формі: вона перебуває в 2NF, і у всіх таблицях відсутні транзитивні функціональні залежності між непервинними атрибутами. У кожній таблиці непервинні поля функціонально залежать напряму від її первинного ключа, а інформація, що стосується інших сутностей (наприклад — дані про батьків або вчителів), зберігається у відповідних таблицях і доступна через FK.

Приклад: у таблиці journal: journal\_id => student\_id, teacher\_id, subject\_id, entry\_date, grade, attendance\_status Жоден неключовий атрибут (наприклад, attendance\_status чи grade) не визначає інший неключовий. Отже, транзитивних залежностей немає, схема відповідає вимогам 3НФ. Отже, схема відповідає вимогам 3NF.

Отже схема бази даних відповідає вимогам 1NF, 2NF і 3NF.

Схема бази даних у pgAdmin 4 зображено на рисунку 3.

**Изображение выглядит как текст, снимок экрана, число, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.**

*Рисунок 3 − Схема бази даних у pgAdmin 4*

**Таблиці бази даних у pgAdmin4**

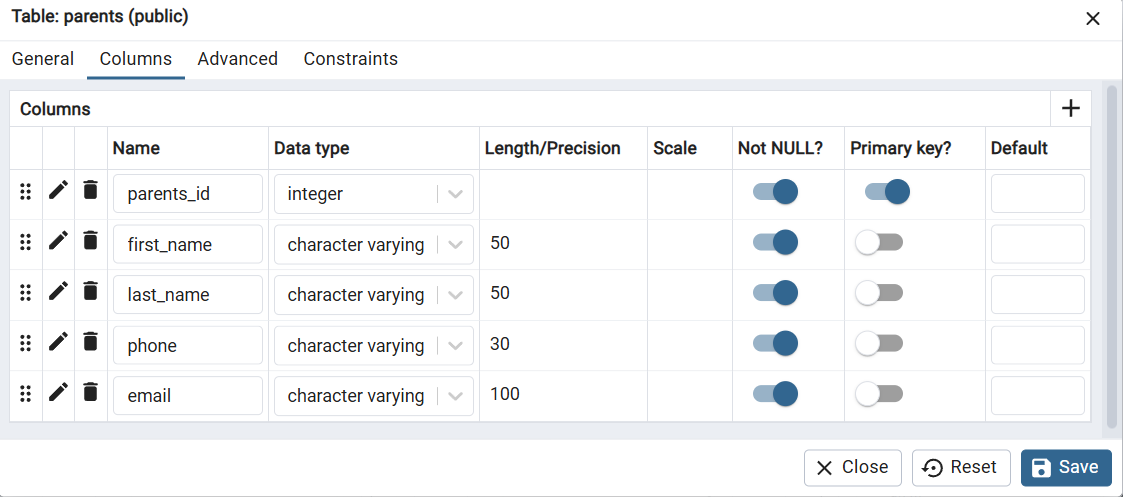
Створені у pgAdmin4 таблиці наведено на рисунку 4.

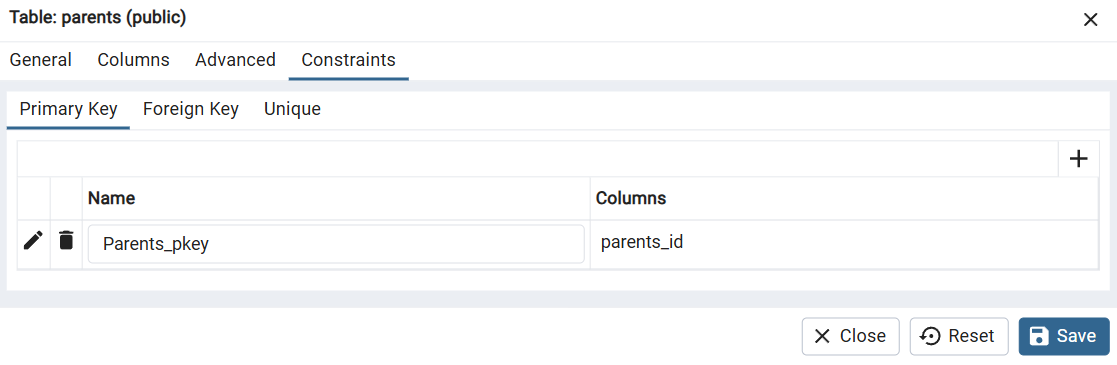
*Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Содержимое, созданное искусственным интеллектом, может быть неверным.*

*Рисунок 6 – Список таблиць у pgAdmin4*

Таблиця Parents





Изображение выглядит как текст, Шрифт, снимок экрана, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Таблиця Student

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, Шрифт, линия, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, Шрифт, число, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Таблиця Teacher

Изображение выглядит как текст, число, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, линия, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Таблиця Subject

Изображение выглядит как текст, Шрифт, число, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, линия, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, Шрифт, число, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Таблиця Journal

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, линия, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Таблиці в коді SQL**

BEGIN;

CREATE TABLE IF NOT EXISTS public.student

(

student\_id integer NOT NULL,

parents\_id integer NOT NULL,

first\_name character varying(50) NOT NULL,

last\_name character varying(50) NOT NULL,

birth\_date date NOT NULL,

class character varying(20) NOT NULL,

email character varying(100) NOT NULL,

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

);

CREATE TABLE IF NOT EXISTS public.teacher

(

teacher\_id integer NOT NULL,

first\_name character varying(50) NOT NULL,

last\_name character varying(50) NOT NULL,

email character varying(100) NOT NULL,

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

);

CREATE TABLE IF NOT EXISTS public.parents

(

parents\_id integer NOT NULL,

first\_name character varying(50) NOT NULL,

last\_name character varying(50) NOT NULL,

phone character varying(30) NOT NULL,

email character varying(100) NOT NULL,

CONSTRAINT "Parents\_pkey" PRIMARY KEY (parents\_id)

);

CREATE TABLE IF NOT EXISTS public.subject

(

subject\_id integer NOT NULL,

name character varying(50) NOT NULL,

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

);

CREATE TABLE IF NOT EXISTS public.journal

(

journal\_id integer NOT NULL,

student\_id integer NOT NULL,

teacher\_id integer NOT NULL,

subject\_id integer NOT NULL,

entry\_date date NOT NULL,

grade integer,

attendance\_status character varying(30) NOT NULL,

CONSTRAINT "Journal\_pkey" PRIMARY KEY (journal\_id)

);

ALTER TABLE IF EXISTS public.student

ADD FOREIGN KEY (parents\_id)

REFERENCES public.parents (parents\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.journal

ADD FOREIGN KEY (student\_id)

REFERENCES public.student (student\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.journal

ADD FOREIGN KEY (teacher\_id)

REFERENCES public.teacher (teacher\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.journal

ADD FOREIGN KEY (subject\_id)

REFERENCES public.subject (subject\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

END;

**Висновки**

У ході виконання лабораторної роботи було створено базу даних електронного журналу, що містить інформацію про учнів, батьків, учителів та результати навчання.

Побудовано ER-діаграму для графічного відображення структури та взаємозв’язків між сутностями. Схему бази даних перевірено на відповідність нормалізаційним формам (1НФ, 2НФ, 3НФ), що гарантує ефективне та надійне зберігання даних.

Також у середовищі pgAdmin4 було створено таблиці, визначено ключі та зв’язки між ними, а також здійснено перевірку правильності структури.

У результаті отримана база даних відповідає поставленим вимогам та може бути використана як основа для подальшої розробки системи електронного журналу.

**Контакти:**

Репозиторій Git: <https://github.com/snizhana-ponamarova/databases>