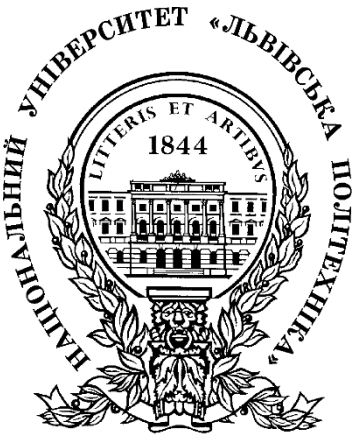
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра систем штучного інтелекту



ЗВІТ 2

з курсу “ОБДЗ”

на тему:

“ Створення таблиць бази даних засобами SQL”

Виконала:

студентка групи КН-211

Досяк Ірина

Викладач:

Якимишин Х.М.

Лабораторна №2

Мета роботи: створення таблиць бази даних засобами SQL

Короткі теоретичні відомості

Щоб створити нову базу даних, використовуємо **CREATE DATABASE**. Щоб створити структуру таблиці, використовуємо **CREATE TABLE**, який визначає ім’я таблиці а також її стовпчики із типами та розмірами.

Використовуємо такі типи даних:

ім’я_таблиці	Назва таблиці. Або назва_бази.назва_таблиці.
тип_таблиці	В MySQL крім типів таблиць MyISAM та InnoDB існують типи MEMORY, BDB, ARCHIVE тощо.
тип_обмеження	Задає тип індексу для ключового поля: USING {BTREE | HASH | RTREE}.
PRIMARY KEY	Вказує, що дане поле буде первинним ключем в таблиці.
UNIQUE	Вказує на те, що в даному полі будуть зберігатися унікальні значення.
FOREIGN KEY ... REFERENCES	Створює зовнішній ключ, зв’язаний із вказаним полем (полями).
TEMPORARY	Створення тимчасову таблицю, яка буде знищена після завершення зв’язку із сервером.
CONSTRAINT	Вказує на початок оголошення PRIMARY KEY, UNIQUE, або FOREIGN KEY обмеження.

NULL | NOT NULL	Директива, що дозволяє/забороняє null-значення для даного поля.
FULLTEXT|SPATIAL	Тип індексу (повнотекстовий/просторовий; тільки для таблиць типу MyISAM).
AVG_ROW_LENGTH	Приблизне значення середньої довжини рядків зі змінною довжиною.
DATA DIRECTORY	Вказує шлях, за яким таблиця має зберігатись у файловій системі.
CHECKSUM	Якщо параметр = 1, то для рядків таблиці буде рахуватись контрольна сума. Це сповільнює оновлення таблиці, але робить легшим пошук пошкоджених таблиць.
ROW_FORMAT	Вказує на спосіб зберігання рядків таблиці (залежно від типу таблиці).

CHAR(size)	Містить рядок фіксованої довжини (може містити букви, цифри, та інші символи). Фіксована довжина задається в дужках. Може зберігати до 255 символів.
VARCHAR(size)	Містить рядок змінної довжини. Найбільша довжина задається в дужках. Може зберігати до 255 символів. Примітка: Якщо ви покладете туди значення більше за 255, тип буде перетворений на TEXT.
TINYTEXT	Рядок з найбільшою довжиною 255 символів
TEXT	Зберігає рядок з найдовшою довжиною 65,535 символів
BLOB	Великий двійковий об'єкт (Binary Large OBject). Зберігає до 65,535 байт даних
MEDIUMBLOB	Великий двійковий об'єкт. 16 Мегабайт даних
LONGTEXT	Рядок з найбільшою довжиною в 4,294,967,295 символів.
LONGBLOB	Великий двійковий об'єкт. 4 Гігабайти даних

SET	Подібно до ENUM окрім того, що SET може містити до 64 значень списку, і не може зберігати більше одного вибору.

TINYINT(size)	Цілий від -128 до 127 . Від 0 до 255 UNSIGNED[1]. Максимальне число цифр задається в дужках.
SMALLINT(size)	Від -32768 до 32767. Від 0 до 65535 UNSIGNED[1]. Максимальне число цифр задається в дужках.
MEDIUMINT(size)	Від -8388608 до 8388607. Від 0 до 16777215 UNSIGNED[1]. Максимальне число цифр задається в дужках.
INT(size)	Від -2147483648 до 2147483647. Від 0 до 4294967295 UNSIGNED]. Максимальне число цифр задається в дужках.
FLOAT(size,d)	Число з плаваючою крапкою. Максимальне число цифр задається в параметрі size. Максимальне число цифр після десяткової крапки задається в параметрі d.
DOUBLE(size,d)	Точніше число з плаваючою крапкою. Максимальне число цифр задається в параметрі size. Максимальне число цифр після десяткової крапки задається в параметрі d.
DECIMAL(size,d)	DOUBLE, що зберігається як рядок з фіксованою крапкою.. Максимальне число цифр задається в параметрі size.

	Максимальне число цифр після десяткової крапки задається в параметрі d.

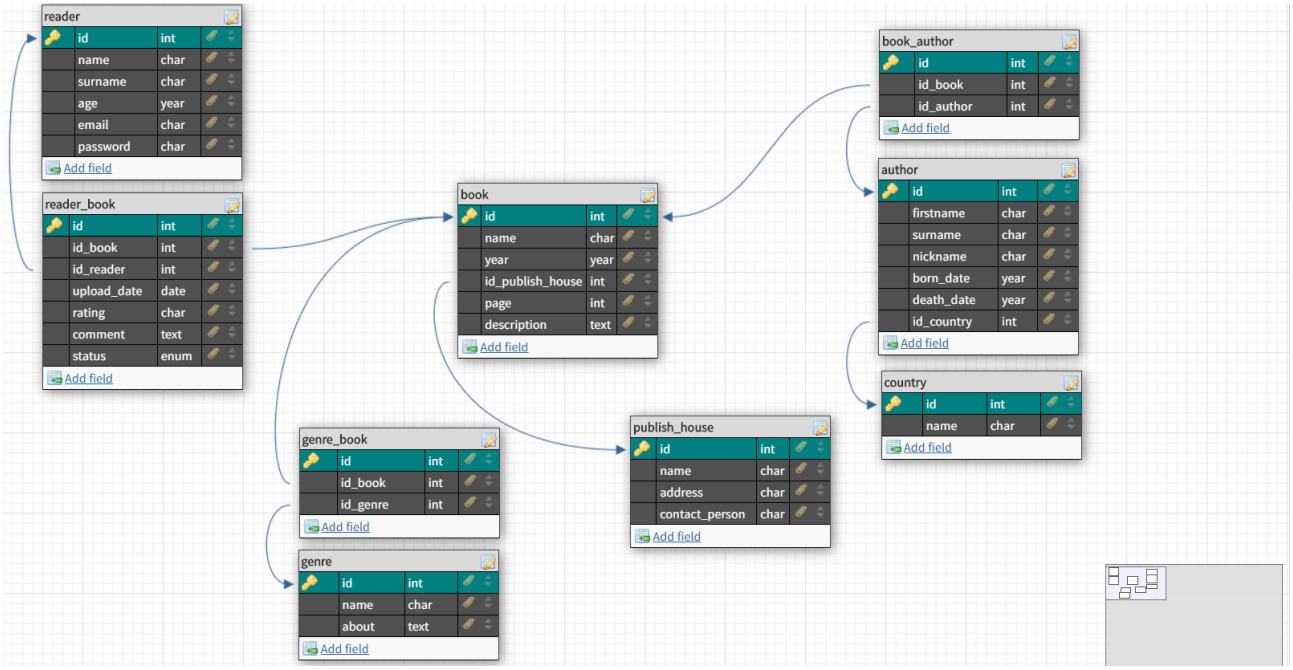
DATE()	Дата. Формат: YYYY-MM-DD Зауваження: Підтримується діапазон від '1000-01-01' до '9999-12-31'
DATETIME()	Формат: YYYY-MM-DD HH:MM:SS [2]. Зауваження: Підтримується діапазон від '1000-01-01 00:00:00' до '9999-12-31 23:59:59'
TIME()	Час. Формат: HH:MM:SS Зауваження: Підтримується діапазон від '-838:59:59' до '838:59:59'
YEAR()	Рік в двоцифровому, або чотирицифровому форматі. Зауваження: Значення, що дозволені в чотирицифровому форматі: від 1901 до 2155. Значення дозволені в двоцифровому форматі: від 70 до 69, що відповідає 1970 та 2069.

Для дії використовуємо:

- CASCADE - одночасне видалення, або оновлення відповідного значення у зовнішній таблиці.
- RESTRICT Аналог NO ACTION. Дія над значенням поля ігнорується, якщо існує відповідне йому значення у зовнішній таблиці. Опція задана за замовчуванням.
- SET NULL При дії над значенням у первинній таблиці, відповідне значення у зовнішній таблиці замінюється на NULL.

Виконання:

Будуємо даталогічну модель бази даних, визначивши типи, розмірності та обмеження полів:



Для зв'язку читача і книги встановлено обмеження цілісності «каскадне оновлення». Для поля «status» в таблиці «book» визначили такий домен: ('not_read', 'want_to_read', 'in_progress', 'read')

Створимо спроектовану базу даних:

```
CREATE SCHEMA IF NOT EXISTS library DEFAULT CHARACTER SET utf8 ;
USE library ;
CREATE TABLE IF NOT EXISTS library.publish_house (
  id_publish_house INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(45) NOT NULL,
  address VARCHAR(70) NULL,
  contact VARCHAR(45) NULL,
  PRIMARY KEY (`id_publish_house`));
CREATE TABLE IF NOT EXISTS library.book (
  id_book INT NOT NULL AUTO_INCREMENT,
  name CHAR(45) NOT NULL,
  year char(4) NULL,
```

```
  page INT NULL,
  id_publish_house INT NOT NULL,
  description_of_book VARCHAR(1000) NULL,
  PRIMARY KEY (`id_book`),
  INDEX id_publish_house_idx (`id_publish_house` ASC) VISIBLE,
  CONSTRAINT id_publish_house
    FOREIGN KEY (`id_publish_house`)
    REFERENCES library.publish_house (`id_publish_house`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
CREATE TABLE IF NOT EXISTS library.genre (
  id_genre INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(45) NOT NULL,
  about VARCHAR(150) NULL,
  PRIMARY KEY (`id_genre`));
CREATE TABLE IF NOT EXISTS library.genre_book (
  id INT NOT NULL AUTO_INCREMENT,
  id_book INT NOT NULL,
  id_genre INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX id_book_idx (`id_book` ASC) VISIBLE,
  INDEX id_genre_idx (`id_genre` ASC) VISIBLE,
  CONSTRAINT id_book
    FOREIGN KEY (`id_book`)
    REFERENCES library.book (`id_book`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT id_genre
    FOREIGN KEY (`id_genre`)
    REFERENCES library.genre (`id_genre`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

```
CREATE TABLE IF NOT EXISTS library.country (
  id INT NOT NULL AUTO_INCREMENT,
  country VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`));
CREATE TABLE IF NOT EXISTS library.author (
  id_author INT NOT NULL AUTO_INCREMENT,
  firstname VARCHAR(45) NOT NULL,
  surname VARCHAR(45) NULL,
  nickname VARCHAR(45) NULL,
  born_date DATE NOT NULL,
  death_date DATE NULL,
  id_country INT NOT NULL,
  PRIMARY KEY (`id_author`),
  INDEX id_country_idx (`id_country` ASC) VISIBLE,
  CONSTRAINT id_country
    FOREIGN KEY (`id_country`)
    REFERENCES library.country (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
CREATE TABLE IF NOT EXISTS library.book_author (
  id INT NOT NULL AUTO_INCREMENT,
  id_book_author INT NOT NULL,
  id_author INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX id_author_idx (`id_author` ASC) VISIBLE,
  INDEX id_book_author_idx (`id_book_author` ASC) VISIBLE,
  CONSTRAINT id_author
    FOREIGN KEY (`id_author`)
    REFERENCES library.author (`id_author`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT id_book_author
```

```
    FOREIGN KEY (`id_book_author`)
    REFERENCES library.book (`id_book`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
CREATE TABLE IF NOT EXISTS library.reader (
  id_reader INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(45) NOT NULL,
  surname VARCHAR(45) NOT NULL,
  age VARCHAR(45) NOT NULL,
  email VARCHAR(45) NOT NULL,
  password VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id_reader`));
CREATE TABLE IF NOT EXISTS library.reader_book (
  id INT NOT NULL AUTO_INCREMENT,
  id_reader_book INT NOT NULL,
  id_reader INT NOT NULL,
  upload_date DATE NOT NULL,
  rating VARCHAR(45) NULL,
  comment VARCHAR(45) NULL,
  status ENUM ('not_read', 'want_to_read', 'in_progress', 'read') DEFAULT 'not_read',
  PRIMARY KEY (`id`),
  INDEX id_reader_book_idx (`id_reader_book` ASC) VISIBLE,
  INDEX id_reader_idx (`id_reader` ASC) VISIBLE,
  CONSTRAINT id_reader_book
    FOREIGN KEY (`id_reader_book`)
    REFERENCES library.book (`id_book`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT id_reader
    FOREIGN KEY (`id_reader`)
    REFERENCES library.reader (`id_reader`)
    ON DELETE CASCADE
```

```
ON UPDATE CASCADE);
```

Висновок : під час виконання даної лабораторної роботи я побудувала даталогічну модель бази даних , визначила типи і обмеження таблиць й полів.