МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

КАФЕДРА СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ



Звіт до лабораторної роботи №2

з дисципліни:

“ОБДЗ”

на тему:

“**Створення таблиць бази даних засобами SQL**”

**Підготувала:**

студентка групи КН-209

Дипко Олександра

**Викладач:**

Мельникова Н.І.

Львів 2020

**Мета роботи:**

Побудувати даталогічну модель бази даних; визначити типи, розмірності та обмеження полів; визначити обмеження таблиць; розробити SQL запити для створення спроектованих таблиць.

**Короткі теоретичні відомості.**

Щоб створити нову базу даних у командному рядку клієнта MySQL (mysql.exe) слід виконати команду CREATE DATABASE, опис якої подано нижче. Тут і надалі, квадратні дужки позначають необов’язковий аргумент команди, символ "|" позначає вибір між аргументами.

**CREATE {DATABASE | SCHEMA}** [IF NOT EXISTS]*ім’я\_бази*[[DEFAULT] CHARACTER SET *кодування*] [[DEFAULT] COLLATE *набір\_правил*]

*ім’я\_бази* – назва бази даних (латинські літери і цифри без пропусків); *кодування* – набір символів і кодів (koi8u,latin1,utf8,cp1250тощо); *набір\_правил* – правила порівняння рядків символів (див. результат команди

show collation).

Нижче наведені деякі допоміжні команди для роботи в СУБД MySQL. Кожна команда і кожен запит в командному рядку повинні завершуватись розділяючим символом ";".

1. Перегляд існуючих баз даних:

SHOW DATABASES

1. Вибір бази даних для подальшої роботи:

USE DATABASE *ім’я\_бази*

1. Перегляд таблиць в базі даних:

SHOW TABLES [FOR *ім’я\_бази*]

1. Перегляд опису таблиці в базі:

DESCRIBE *ім’я\_таблиці*

1. Виконати набір команд з зовнішнього файлу:

SOURCE *назва\_файлу*

1. Вивести результати виконання подальших команд у зовнішній файл:

\T *назва\_файлу*

Для роботи зі схемою бази даних існують такі основні команди:

ALTER DATABASE – зміна опису бази даних;

CREATE TABLE – створення нової таблиці;

ALTER TABLE – зміна структури таблиці;

DELETE TABLE – видалення таблиці з бази даних;

CREATE INDEX – створення нового індексу (для швидкого пошуку даних); DROP INDEX – видалення індексу;

DROP DATABASE – видалення бази даних.

Розглянемо команду створення таблиці в MySQL та її основні аргументи.

**CREATE** [TEMPORARY] **TABLE** [IF NOT EXISTS]ім’я\_таблиці

[**(**опис\_таблиці,...**)**]

[додаткові\_параметри] ...

[вибірка\_даних]

**опис\_таблиці**:

назва\_поля опис\_поля

| [CONSTRAINT [ім’я\_обмеження]] PRIMARY KEY (назва\_поля,...) [тип\_обмеження]

| {INDEX|KEY} [ім’я\_обмеження] (назва\_поля,...)[ тип\_обмеження] | [CONSTRAINT [ім’я\_обмеження]] UNIQUE [INDEX|KEY] [ім’я\_обмеження](назва\_поля,...) [тип\_обмеження]

| {FULLTEXT|SPATIAL} [INDEX|KEY] [ім’я\_обмеження] (назва\_поля,...)

[тип\_обмеження]

| [CONSTRAINT [ім’я\_обмеження]] FOREIGN KEY

[ім’я\_обмеження] (назва\_поля,...) опис\_зв’язку | CHECK (вираз)

**опис\_поля:**

тип\_даних [NOT NULL | NULL] [DEFAULT *значення\_за\_замовчуванням*] [AUTO\_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY]

**опис\_зв’язку:**

REFERENCES *ім’я\_таблиці* (назва\_поля, ...)

[ON DELETE дія]

[ON UPDATE дія]

**дія:**

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.

**додаткові\_параметри:**

{ENGINE|TYPE} [=] тип\_таблиці

| AUTO\_INCREMENT [=] *значення\_приросту\_лічильника* | AVG\_ROW\_LENGTH [=] *значення*

| [DEFAULT] CHARACTER SET [=] кодування | CHECKSUM [=] {0 | 1}

| [DEFAULT] COLLATE [=] *набір\_правил*

| COMMENT [=] 'коментар до таблиці'

| DATA DIRECTORY [=] 'абсолютний шлях'

| DELAY\_KEY\_WRITE [=] {0 | 1}

| INDEX DIRECTORY [=] 'абсолютний шлях' | MAX\_ROWS [=] *значення*

**вибірка\_даних:**

[IGNORE | REPLACE] [AS] SELECT ... (вибір даних з інших таблиць)

**вираз:**

Логічний вираз, що повертає TRUE або FALSE.

**Опис аргументів:**

ім’я\_таблиці

Назва таблиці. Або назва\_бази.назва\_таблиці.

тип\_таблиці

* 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}.

TEMPORARY

Створення тимчасової таблиці, яка буде знищена після завершення зв’язку з 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.

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).

AVG\_ROW\_LENGTH

Приблизне значення середньої довжини рядків зі змінною довжиною.

DATA DIRECTORY

Вказує шлях, за яким таблиця має зберігатись у файловій системі.

CHECKSUM

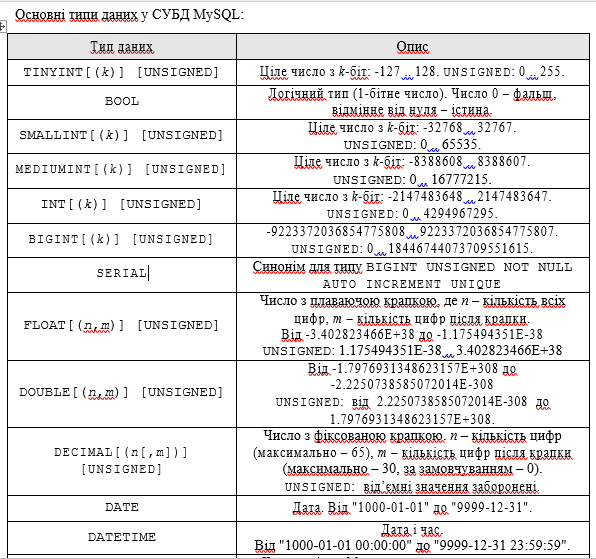
Якщо параметр = 1, то для рядків таблиці буде рахуватись контрольна сума. Це сповільнює оновлення таблиці, але робить легшим пошук пошкоджених таблиць.

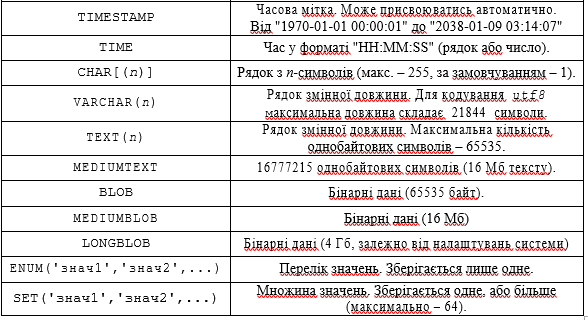
ROW\_FORMAT

Вказує на спосіб зберігання рядків таблиці (залежно від типу таблиці).

FULLTEXT|SPATIAL

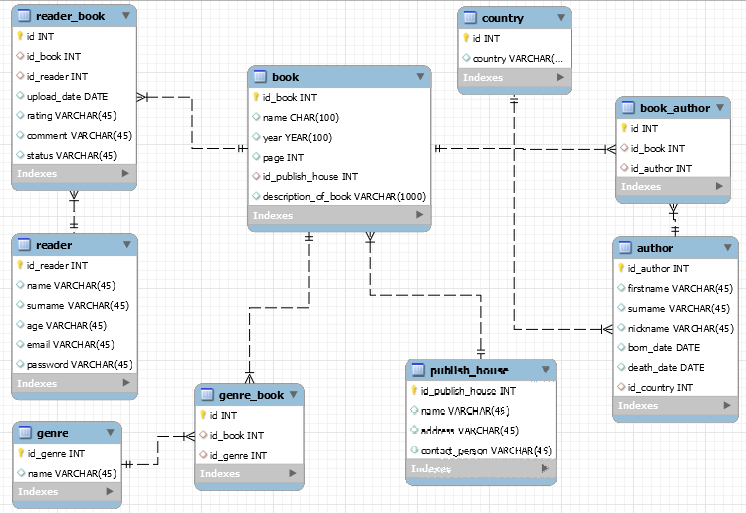
Тип індексу (повнотекстовий/просторовий; тільки для таблиць типу MyISAM).





**Хід роботи**

Даталогічна модель вимагає визначення конкретних полів бази даних, їхніх типів, обмежень на значення, тощо. На рисунку зображено даталогічну модель проектованої бази даних. Для зв’язку коментарів і повідомлень встановлено обмеження цілісності «каскадне оновлення». Для поля status у таблиці reader\_book визначено такий домен – 'not\_read', 'want\_to\_read', 'in\_progress', 'read'.



Створимо нову базу даних, виконавши такі команди:

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,

name VARCHAR(45) NULL,

address VARCHAR(45) NULL,

contact\_person VARCHAR(45) NULL,

PRIMARY KEY (`id\_publish\_house`));

CREATE TABLE IF NOT EXISTS library.book (

id\_book INT NOT NULL,

name CHAR(45) NULL,

year YEAR NULL,

page INT NULL,

id\_publish\_house INT 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,

name VARCHAR(45) NULL,

about VARCHAR(150) NULL,

PRIMARY KEY (`id\_genre`));

CREATE TABLE IF NOT EXISTS library.genre\_book (

id INT NOT NULL,

id\_book INT NULL,

id\_genre INT 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,

country VARCHAR(45) NULL,

PRIMARY KEY (`id`));

CREATE TABLE IF NOT EXISTS library.author (

id\_author INT NOT NULL,

firstname VARCHAR(45) NULL,

surname VARCHAR(45) NULL,

nickname VARCHAR(45) NULL,

born\_date DATE NULL,

death\_date DATE NULL,

id\_country INT 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,

id\_book\_author INT NULL,

id\_author INT 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,

name VARCHAR(45) NULL,

surname VARCHAR(45) NULL,

age VARCHAR(45) NULL,

email VARCHAR(45) NULL,

password VARCHAR(45) NULL,

PRIMARY KEY (`id\_reader`));

CREATE TABLE IF NOT EXISTS library.reader\_book (

id INT NOT NULL,

id\_reader\_book INT NULL,

id\_reader INT NULL,

upload\_date DATE 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:**

на цій лабораторній роботі було завершено моделювання і засобами SQL створено базу даних, що складається з дев’яти таблиць.