Міністерство освіти і науки України

Тернопільський національний економічний університет

Факультет комп’ютерних інформаційних технологій

Кафедра комп’ютерних наук

**ЛАБОРАТОРНА РОБОТА № 4**

З дисципліни:

«Web-програмування»

**Виконав:**

студент групи ІПЗ-32

Фатюк В.І

**Перевірив:**

викл. Дарморост І.А.

Тернопіль 2018

Мета роботи: отримання навиків роботи з реляційними базами даних на прикладі СУБД MySQL.

**Предметна область- «Бібліотека»**

Створення БД, таблиць та зв’язків між ними

-- MySQL Script generated by MySQL Workbench

-- Sun Nov 04 12:07:08 2018

-- Model: New Model Version: 1.0

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;

USE `mydb` ;

-- -----------------------------------------------------

-- Table `mydb`.`Book`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Book` (

`idBook` INT NOT NULL,

`Name` VARCHAR(35) NULL,

`Author` VARCHAR(15) NULL,

`Publisher` VARCHAR(25) NULL,

`Year\_publication` DATE NULL,

`Volume` VARCHAR(45) NULL,

`Price` FLOAT NULL,

PRIMARY KEY (`idBook`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`IssueBooks`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`IssueBooks` (

`idIssueBooks` INT NOT NULL,

`Date` DATE NULL,

`Return\_date` DATE NULL,

PRIMARY KEY (`idIssueBooks`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`Booking`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Booking` (

`idBooking` INT NOT NULL,

`Ticket\_number` INT NULL,

`Order\_date` DATE NULL,

`Book\_idBook` INT NOT NULL,

`IssueBooks\_idIssueBooks` INT NOT NULL,

PRIMARY KEY (`idBooking`, `Book\_idBook`, `IssueBooks\_idIssueBooks`),

INDEX `fk\_Booking\_Book\_idx` (`Book\_idBook` ASC) VISIBLE,

INDEX `fk\_Booking\_IssueBooks1\_idx` (`IssueBooks\_idIssueBooks` ASC) VISIBLE,

CONSTRAINT `fk\_Booking\_Book`

FOREIGN KEY (`Book\_idBook`)

REFERENCES `mydb`.`Book` (`idBook`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Booking\_IssueBooks1`

FOREIGN KEY (`IssueBooks\_idIssueBooks`)

REFERENCES `mydb`.`IssueBooks` (`idIssueBooks`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`Reader`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Reader` (

`idReader` INT NOT NULL,

`Name` VARCHAR(15) NULL,

`Surname` VARCHAR(15) NULL,

`Last\_name` VARCHAR(15) NULL,

`Phone` INT NULL,

`Address` VARCHAR(45) NULL,

`Booking\_idBooking` INT NOT NULL,

`Booking\_Book\_idBook` INT NOT NULL,

`Booking\_IssueBooks\_idIssueBooks` INT NOT NULL,

PRIMARY KEY (`idReader`, `Booking\_idBooking`, `Booking\_Book\_idBook`, `Booking\_IssueBooks\_idIssueBooks`),

INDEX `fk\_Reader\_Booking1\_idx` (`Booking\_idBooking` ASC, `Booking\_Book\_idBook` ASC, `Booking\_IssueBooks\_idIssueBooks` ASC) VISIBLE,

CONSTRAINT `fk\_Reader\_Booking1`

FOREIGN KEY (`Booking\_idBooking` , `Booking\_Book\_idBook` , `Booking\_IssueBooks\_idIssueBooks`)

REFERENCES `mydb`.`Booking` (`idBooking` , `Book\_idBook` , `IssueBooks\_idIssueBooks`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

**SQL-запити**

**1.**SQL-запити, які дозволяють вивести таку інформацію з БД Бібліотека:

INSERT INTO `mydb`.`Book` (`idBook`, `Name`, `Author`, `Publisher`, `Year\_publication`, `Volume`, `Price`) VALUES ('1', 'Путь программиста', 'Джон Сонмез', 'м.Тернополь', '2015-10-10', '502ст.', '234');

INSERT INTO `mydb`.`Book` (`idBook`, `Name`, `Author`, `Publisher`, `Year\_publication`, `Volume`, `Price`) VALUES ('2', 'Лайфхакинг из первых рук', 'Игорь Савчук', 'м.Киев', '2016-11-08', '607ст.', '654');

INSERT INTO `mydb`.`Book` (`idBook`, `Name`, `Author`, `Publisher`, `Year\_publication`, `Volume`, `Price`) VALUES ('3', 'Факты и заблуждения ', 'Роберт Гласс', 'м.Тернополь', '2018-10-10', '892ст.', '356');

INSERT INTO `mydb`.`Book` (`idBook`, `Name`, `Author`, `Publisher`, `Year\_publication`, `Volume`, `Price`) VALUES ('4', 'Вальсируя с Медведями', 'Том ДеМарко', 'м.Львов', '2018-09-10', '345ст.', '325');

INSERT INTO `mydb`.`Book` (`idBook`, `Name`, `Author`, `Publisher`, `Year\_publication`, `Volume`, `Price`) VALUES ('5', 'Джоэл о программировании', 'Джоэл Спольски', 'м.Тернополь', '2018-10-04', '444ст.', '123');

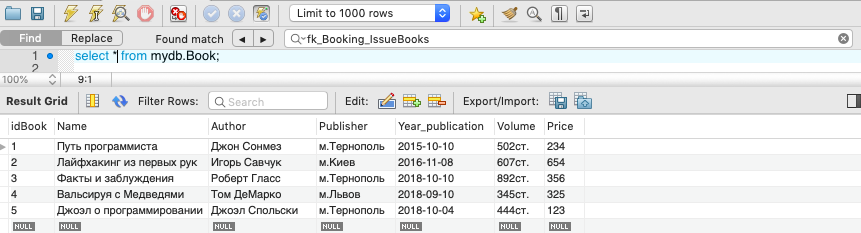


Рисунок 1-Вивід БД

**2.** Виводимо список книг, та їх автора.

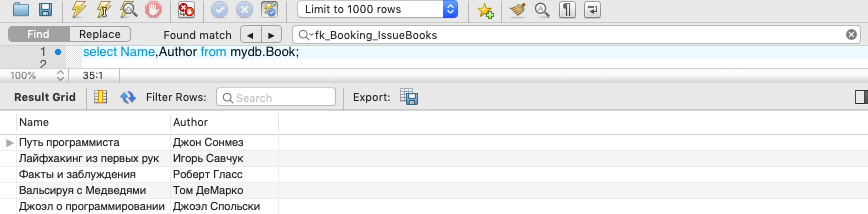


Рисунок 2- Список книг та авторів

**3.** Виводимо книги з найбільшою та найменшою кількістю сторінок.

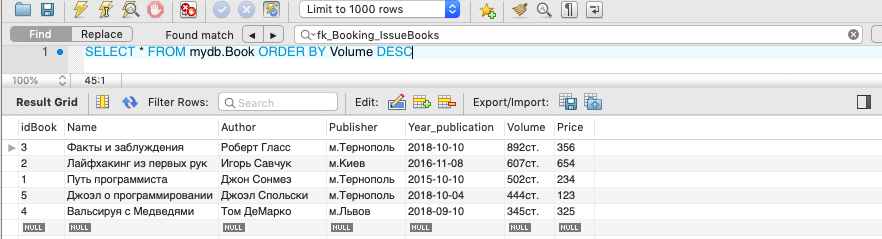


Рисунок 3- Книги з найбільшою кількістю сторінок

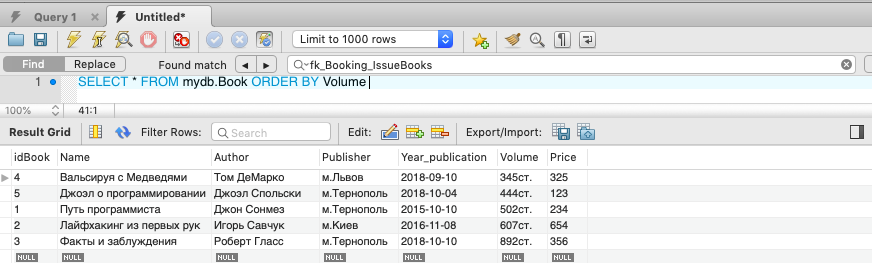


Рисунок 4- Книги з найменшою кількістю сторінок

**4.** Запит, який виводить книгу, в назві якої найбільше знаків.

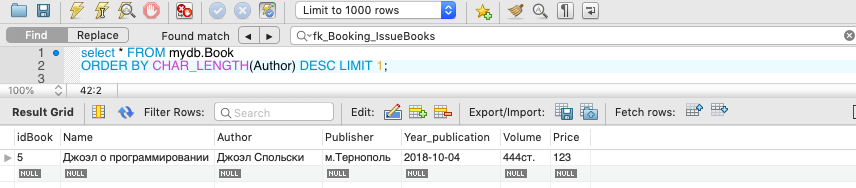


Рисунок 5- Книга з найбільшою назвою

**5.** Виведемо звіт про те, скільки книг є кожного автора.

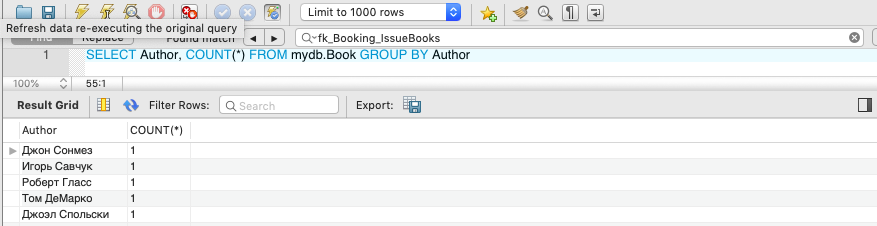


Рисунок 6- Кількість книг у кожного автора

**6.** Визначимо на яку букву алфавіту припадає найбільше прізвищ авторів книг.

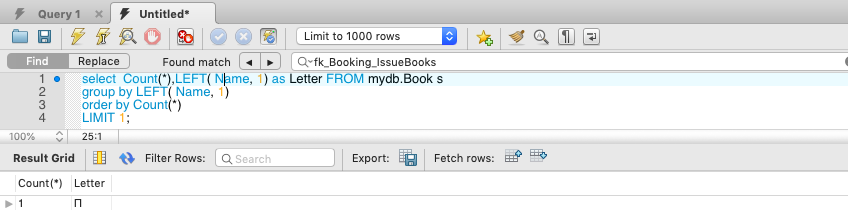


Рисунок 7- Більшість прізвищ авторів

**7.** Знайдемо книги, місто публікацій яких закінчується на «ль».

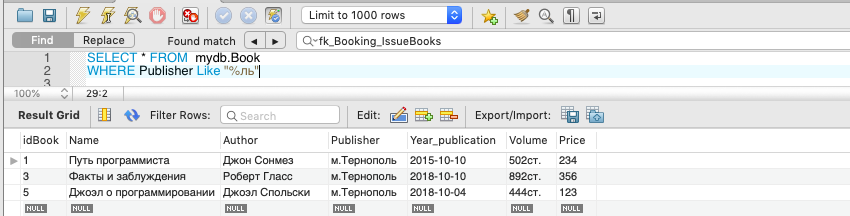


Рисунок 8- Місто публікацій має закінчення «ль»

**8.** Виводимо список авторів, в яких сьогодні день народження.

select \* from mydb.Book s

where Day(Year\_publication)+Month(Year\_publication)=Day(CURDATE())+Month(CURDATE())

**Функції маніпулювання даними**

**9.** Запит, який повертає книги, що були додані в каталог більш як 4 роки з дати їх створення.

SELECT \* FROM mydb.Book s

WHERE TIMESTAMPDIFF(Year,Year\_publication, CURDATE()) >4

**10.** Виводимо з БД список книг, які були написані у високосному році.

SELECT \* FROM mydb.Book s

WHERE Year(Year\_publication)%4=0

**11.** Запит до БД , результатом якого є таблиця з двома полями.

Select TIMESTAMPDIFF(Year,Year\_publication, CURDATE()),s.Name from mydb.Book s

**Запити-дії**

**12.** Вставимо за допомогою SQL-запиту в таблицю запис автора нової книги (себе).

INSERT INTO `mydb`.`Book` (`idBook`, `Name`, `Author`, `Publisher`, `Year\_publication`, `Volume`, `Price`) VALUES ('1', 'Путь программиста', Фатюк Василь, 'м.Тернополь', '2015-10-10', '999ст.', '532');

**13.** Запит, який добавляє до таблиці 3 нові поля.

ALTER TABLE mydb.Book

ADD COLUMN ss VARCHAR(100) NULL,

ADD COLUMN hh decimal(10,2) NULL;

ADD COLUMN dd decimal(10,2) NULL;

**14.** Запит, який видаляє книги певного автора.

DELETE FROM `mydb`.`Book` WHERE (`idBook` = '5');

**Об’єднання таблиць**

**15.** Виводимо, скільки книг представлено в кожній рубриці каталогу (жанру).

select \* from mydb.Book

inner join mydb.Booking on (mydb.Booking.idBooking=mydb.Book.idBook)

**16.** Виводимо список всіх книг.

select \* from mydb.Book ;

**Views**

**17.** Створити представлення для таблиці автор книги, в якому замість поля автор буде два поля: адрес та телефон.

CREATE VIEW v1 AS SELECT Name, Author (Adress+" "+Phone+" ") as Reader from mydb.Reader

**18.** Створити представления для БД Бібліотека, яке показує, скільки книг є кожного автора.

CREATE VIEW v2 AS SELECT Name,Count(\*) FROM mydb.Book g inner join mydb.Reader s on g.Id = idReader Group by idReader

**Процедури**

**19.** Процедура, яка відображає інформацію про книги.

CREATE PROCEDURE job\_Book\_info() SELECT Name from mydb.Book

1. Процедура, яка повертає кількість книг певного автора.

CREATE PROCEDURE job\_Book\_info2(IN info varchar(50))

begin

select Name,Count(\*) FROM mydb.Book g inner join mydb.Reader s on g.Id = idReader Group by idReader

**21.** Процедура, яка відображає книги в межах заданого діапазону.

CREATE PROCEDURE job\_Book\_info3 ()) BEGIN SELECT \* FROM mydb.Reader t where Adress >2 and Adress<10; END

**Транзакції**

**22.** Розпочнемо транзакцію транзакцію. Зробимо зміни в кількох рядках таблиці.

START TRANSACTION;

SELECT\* FROM mydb.Reader t WHERE idReader=10;

UPDATE mydb.Reader t SET t.FirstName = "SomeFirst" WHERE idReader=10;

UPDATE mydb.Reader t SET t.FirstName = "SomeFirst" WHERE idReader=11;

COMMIT;

**Тригери**

**23.** Створимо тригер, який би при виконанні команди DELETE здійснював запис про користувача, що цю команду виконав.

CREATE TRIGGER log\_Reader\_delete AFTER DELETE on mydb.Reader

FOR EACH ROW

BEGIN

Select Name()

END

24. Створити тригер, який при здійснені команди INSERT,підраховував кількість записів в таблиці.

CREATE TRIGGER log\_Bokking

BEFORE INSERT ON mydb.Booking

FOR EACH ROW

BEGIN

select count(\*) from mydb.Booking

END;

**Висновок:** виконуючи дану лабораторну роботу було отримано навички роботи з реляційними базами даних на прикладі СУБД MySQL.