МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра систем штучного інтелекту

Лабораторна робота №2

із дисципліни

Бази даних

Виконав:
Ст. групи КН-207
Гуменчук Б.Р.
Прийняв:
Мельникова Н.І.

Львів – 2018 р.

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, cp1250 тощо);

набір_правил – правила порівняння рядків символів.

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
2. Вибір бази даних для подальшої роботи: USE DATABASE ім'я_бази
3. Перегляд таблиць в базі даних: SHOW TABLES [FOR ім'я_бази]
4. Перегляд опису таблиці в базі: DESCRIBE ім'я_таблиці
5. Виконати набір команд з зовнішнього файлу: SOURCE назва_файлу
6. 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.

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 полів status у таблицях MESSAGE та COMMENT визначено такий домен – ("опубліковане", "неопубліковане", "видалене").

Створимо нову базу даних, виконавши такі команди:

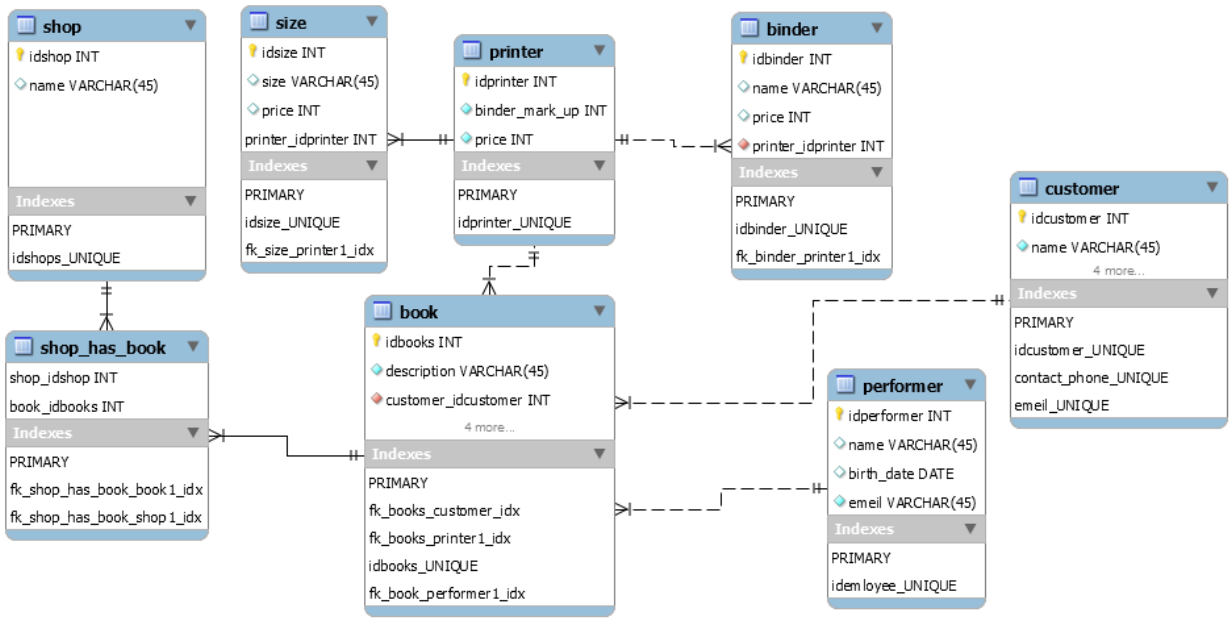
```
-- MySQL Script generated by MySQL Workbench
-- Thu Mar 21 09:03:38 2019
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

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_ZE
RO_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`.`customer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`customer` (
  `idcustomer` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `surname` VARCHAR(45) NOT NULL,
  `contact_phone` VARCHAR(45) NOT NULL,
  `emeil` VARCHAR(45) NOT NULL,
  `birth_date` DATE NULL,
  PRIMARY KEY (`idcustomer`),
  UNIQUE INDEX `idcustomer_UNIQUE` (`idcustomer` ASC) VISIBLE,
  UNIQUE INDEX `contact_phone_UNIQUE` (`contact_phone` ASC) VISIBLE,
  UNIQUE INDEX `emeil_UNIQUE` (`emeil` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`printer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`printer` (
  `idprinter` INT NOT NULL AUTO_INCREMENT,
  `binder_mark_up` INT NOT NULL,
  `price` INT NOT NULL,
  PRIMARY KEY (`idprinter`),
  UNIQUE INDEX `idprinter_UNIQUE` (`idprinter` ASC) VISIBLE)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`performer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`performer` (
  `idperformer` INT NOT NULL,
  `name` VARCHAR(45) NULL,
  `birth_date` DATE NULL,
  `emeil` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idperformer`),
  UNIQUE INDEX `idemloyee_UNIQUE` (`idperformer` ASC) VISIBLE)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`book`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`book` (
  `idbooks` INT NOT NULL AUTO_INCREMENT,
  `description` VARCHAR(45) NOT NULL,
  `customer_idcustomer` INT NOT NULL,
  `printer_idprinter` INT NOT NULL,
  `amount` INT NOT NULL,
  `employee` VARCHAR(45) NOT NULL,
  `performer_idperformer` INT NOT NULL,
  PRIMARY KEY (`idbooks`),
  INDEX `fk_books_customer_idx` (`customer_idcustomer` ASC) VISIBLE,
  INDEX `fk_books_printer1_idx` (`printer_idprinter` ASC) VISIBLE,
  UNIQUE INDEX `idbooks_UNIQUE` (`idbooks` ASC) VISIBLE,
  INDEX `fk_book_performer1_idx` (`performer_idperformer` ASC) VISIBLE,
  CONSTRAINT `fk_books_customer`
    FOREIGN KEY (`customer_idcustomer`)
    REFERENCES `mydb`.`customer` (`idcustomer`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_books_printer1`
    FOREIGN KEY (`printer_idprinter`)
    REFERENCES `mydb`.`printer` (`idprinter`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_book_performer1`
    FOREIGN KEY (`performer_idperformer`)
    REFERENCES `mydb`.`performer` (`idperformer`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`binder`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`binder` (
  `idbinder` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NULL,
  `price` INT NULL,
  `printer_idprinter` INT NOT NULL,
  PRIMARY KEY (`idbinder`),
  UNIQUE INDEX `idbinder_UNIQUE` (`idbinder` ASC) VISIBLE,
  INDEX `fk_binder_printer1_idx` (`printer_idprinter` ASC) VISIBLE,
  CONSTRAINT `fk_binder_printer1`
    FOREIGN KEY (`printer_idprinter`)
    REFERENCES `mydb`.`printer` (`idprinter`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`shop`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`shop` (
  `idshop` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NULL,
  PRIMARY KEY (`idshop`),
  UNIQUE INDEX `idshops_UNIQUE` (`idshop` ASC) VISIBLE)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`size`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`size` (
  `idsize` INT NOT NULL AUTO_INCREMENT,
  `size` VARCHAR(45) NULL,
  `price` INT NULL,
  `printer_idprinter` INT NOT NULL,
  PRIMARY KEY (`idsize`, `printer_idprinter`),
  UNIQUE INDEX `idsize_UNIQUE` (`idsize` ASC) VISIBLE,
  INDEX `fk_size_printer1_idx` (`printer_idprinter` ASC) VISIBLE,
  CONSTRAINT `fk_size_printer1`
    FOREIGN KEY (`printer_idprinter`)
    REFERENCES `mydb`.`printer` (`idprinter`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`shop_has_book`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`shop_has_book` (
  `shop_idshop` INT NOT NULL,
  `book_idbooks` INT NOT NULL,
  PRIMARY KEY (`shop_idshop`, `book_idbooks`),
  INDEX `fk_shop_has_book_book1_idx` (`book_idbooks` ASC) VISIBLE,
  INDEX `fk_shop_has_book_shop1_idx` (`shop_idshop` ASC) VISIBLE,
  CONSTRAINT `fk_shop_has_book_shop1`
    FOREIGN KEY (`shop_idshop`)
    REFERENCES `mydb`.`shop` (`idshop`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_shop_has_book_book1`
    FOREIGN KEY (`book_idbooks`)
    REFERENCES `mydb`.`book` (`idbooks`)
    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;
```



Висновок: на цій лабораторній роботі було завершено моделювання і засобами SQL створено базу даних, що складається з восьми таблиць.