МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"
ІНСТИТУТ ПІСЛЯДИПЛОМНОЇ ОСВІТИ
КАФЕДРА СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ

ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ №2
з курсу “ОБДЗ”
на тему:
“ Створення таблиць бази даних засобами SQL ”

Підготувала:

Студентка групи КН-209

Кульчицька Олена

Викладач:

Мельникова Н.І.

2020

Лабораторна робота №2
з курсу “ОБДЗ”
на тему:
“Створення таблиць бази даних засобами SQL”

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 (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

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*

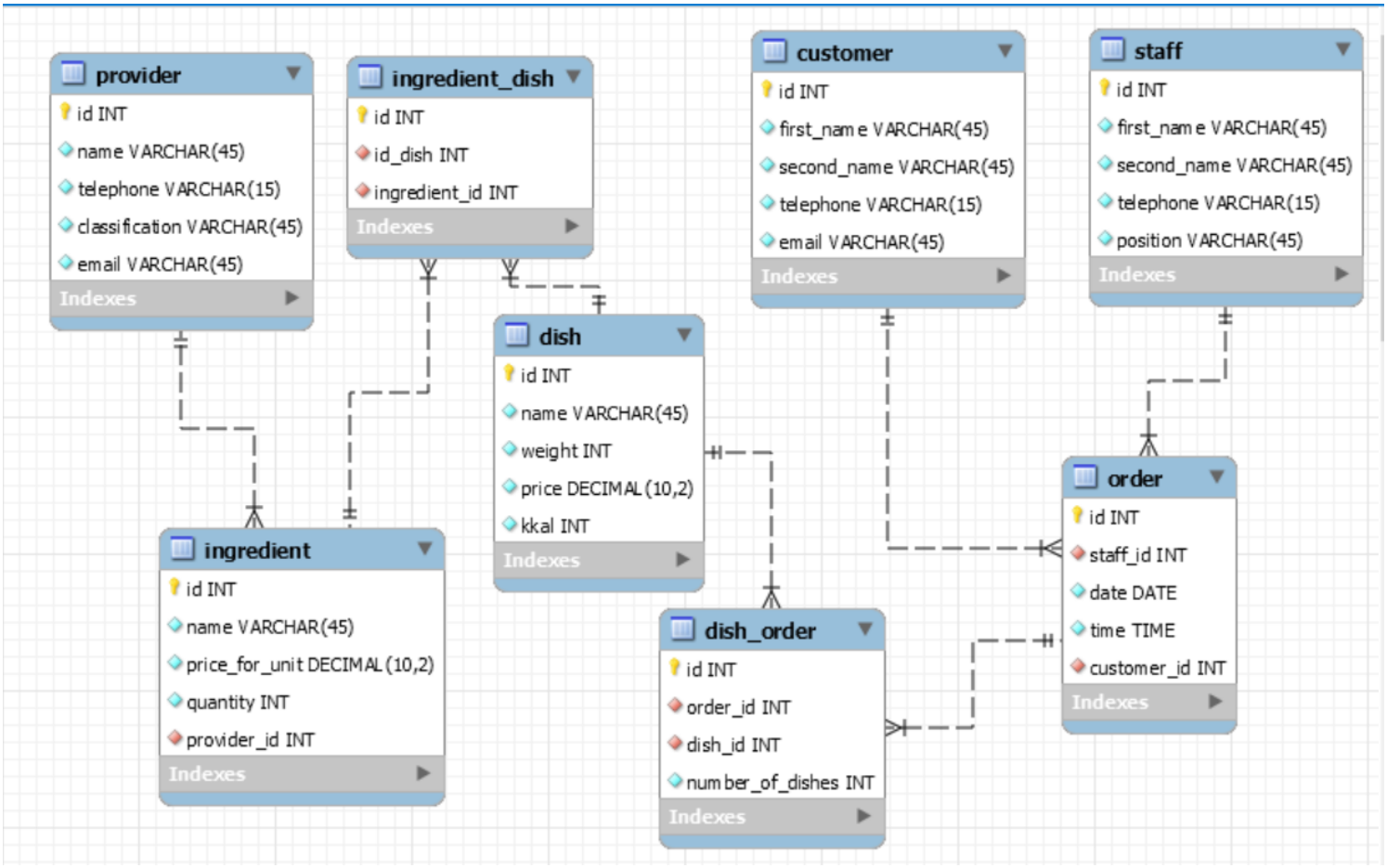
Для роботи зі схемою бази даних існують такі основні команди:
ALTER DATABASE – зміна опису бази даних; CREATE TABLE – створення нової таблиці; ALTER TABLE – зміна структури таблиці; DELETE TABLE – видалення таблиці з бази даних;
CREATE INDEX – створення нового індексу (для швидкого пошуку даних);
DROP INDEX – видалення індексу;
DROP DATABASE – видалення бази даних.

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*
| MIN_ROWS [=] *значення*
| ROW_FORMAT {DEFAULT|DYNAMIC|FIXED|COMPRESSED|REDUNDANT|COMPACT}
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
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}.
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.



Створимо нову базу даних, виконавши такі команди:

```
CREATE SCHEMA IF NOT EXISTS `Confectionary` DEFAULT CHARACTER SET utf8 ;
USE `Confectionary` ;

CREATE TABLE IF NOT EXISTS `Confectionary`.`dish` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `weight` INT NOT NULL,
  `price` DECIMAL(10,2) NOT NULL,
  `kkal` INT NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Confectionary`.`customer` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `first_name` VARCHAR(45) NOT NULL,
  `second_name` VARCHAR(45) NOT NULL,
  `telephone` VARCHAR(15) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `Confectionary`.`staff` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `first_name` VARCHAR(45) NOT NULL,
  `second_name` VARCHAR(45) NOT NULL,
```

```
  `telephone` VARCHAR(15) NOT NULL,
  `position` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `Confectionary`.`provider` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `telephone` VARCHAR(15) NOT NULL,
  `classification` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `Confectionary`.`ingredient` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `price_for_unit` DECIMAL(10,2) NOT NULL,
  `quantity` INT NOT NULL,
  `provider_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `provider_idx` (`provider_id` ASC),
  CONSTRAINT `provider`
    FOREIGN KEY (`provider_id`)
    REFERENCES `Confectionary`.`provider` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `Confectionary`.`order` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `staff_id` INT NOT NULL,
  `date` DATE NOT NULL,
  `time` TIME NOT NULL,
  `customer_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `customer_idx` (`customer_id` ASC),
  INDEX `staff_idx` (`staff_id` ASC),
  CONSTRAINT `customer`
    FOREIGN KEY (`customer_id`)
    REFERENCES `Confectionary`.`customer` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `staff`
    FOREIGN KEY (`staff_id`)
    REFERENCES `Confectionary`.`staff` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `Confectionary`.`dish_order` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `order_id` INT NOT NULL,
  `dish_id` INT NOT NULL,
  `number_of_dishes` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `dish_idx` (`dish_id` ASC),
  INDEX `order_idx` (`order_id` ASC),
  CONSTRAINT `dish`
    FOREIGN KEY (`dish_id`)
    REFERENCES `Confectionary`.`dish` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `order`
    FOREIGN KEY (`order_id`)
    REFERENCES `Confectionary`.`order` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `Confectionary`.`ingredient_dish` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `id_dish` INT NOT NULL,
  `ingredient_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `ingredient_idx` (`ingredient_id` ASC),
  INDEX `dish_idx` (`id_dish` ASC),
  CONSTRAINT `ingredientid`
    FOREIGN KEY (`ingredient_id`)
    REFERENCES `Confectionary`.`ingredient` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `dishid`
    FOREIGN KEY (`id_dish`)
    REFERENCES `Confectionary`.`dish` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

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