**МІНІСТЕРСТВО ОСВІТИ І НАУКИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

**Інститут комп'ютерних наук та інформаційних технологій**

**Кафедра систем штучного інтелекту**

****

**Звіт до лабораторної роботи №2**

з дисципліни

“ОБДЗ”

**Виконав**:

ст. гр. КН-211

Качмарик Віктор

**Викладач:**

Якимишин Х.М.

Львів – 2019

**Лабораторна робота №2**

**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.

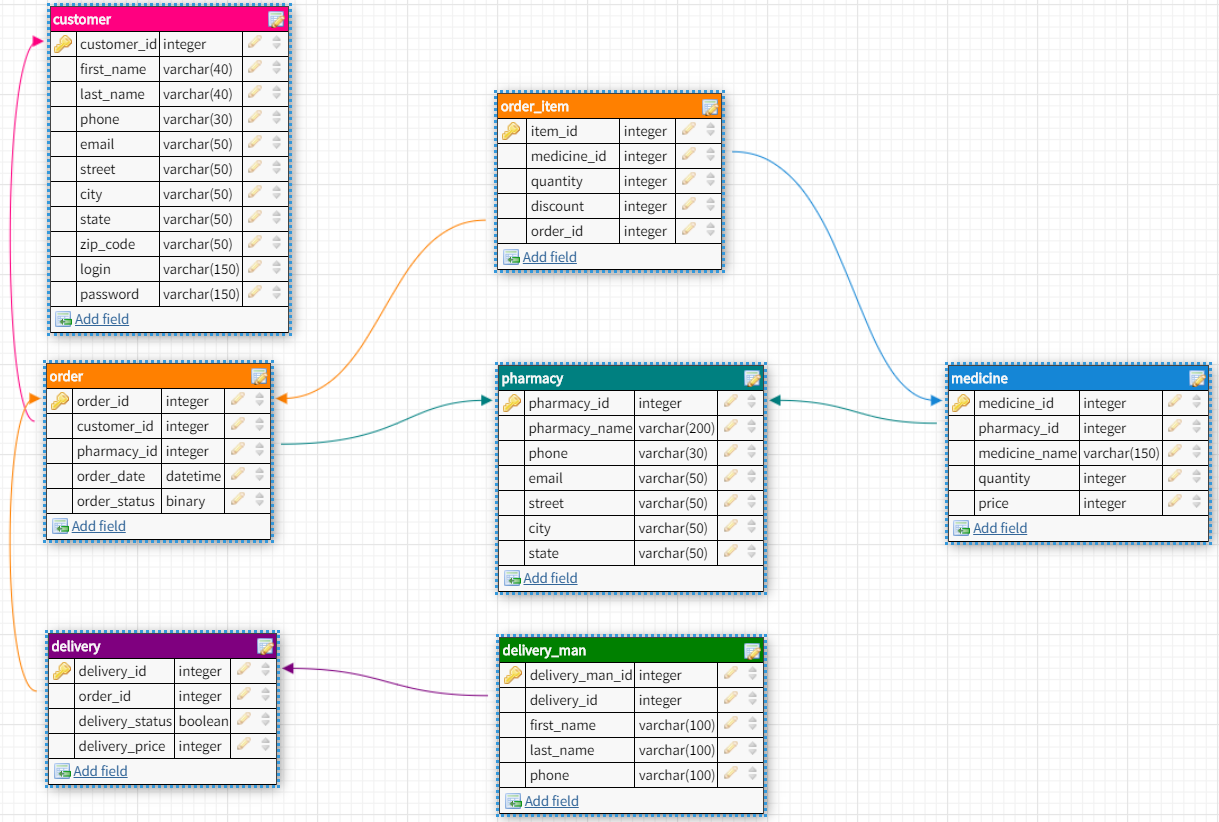
Можна дати декілька порад щодо розробки схеми бази даних і вибору типів даних. Вони дозволять уникнути повільного виконання запитів і потреби модифікації таблиць в майбутньому.

* Слід використовувати якомога менший тип даних для полів таблиць. Наприклад, для зберігання чисел від 1 до 64 краще використати тип TINYINT(6) замість SMALLINT. Це впливає на швидкість пошуку і вибірки даних.
* Слід використовувати рядки фіксованої довжини, якщо це можливо. Для цього всі поля таблиці повинні бути фіксованої довжини. Тобто, варто уникати типів VARCHAR, TEXT і BLOB. Це пришвидчить вибірку даних з середини рядків, оскільки ці дані будуть мати фіксовану адресу. При потребі використання полів з типами TEXT або BLOB, їх можна виділити в окрему таблицю.
* Якщо можливо, варто завжди використовувати поля з обмеженням NOT NULL. Хоча це може збільшувати об’єм бази на диску.
* MySQL дозволяє використовувати різні типи таблиць в одній базі даних. Слід використовувати переваги різних типів (MyISAM, INODB тощо) залежно від характеру майбутнього використання таблиці.
* Потрібно створювати індекси, які пришвидчать пошук і вибірку даних.
* В рідкісних випадках можна денормалізувати схему з метою зменшення кількості операцій з об’єднання таблиць при складних запитах. Але при цьому ускладнюється задача збереження цілісності 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.



Створимо нову базу даних, виконавши такі команди:

CREATE DATABASE pharmacy;

USE pharmacy;

CREATE TABLE `pharmacy` (

`pharmacy\_id` INT NOT NULL AUTO\_INCREMENT,

`pharmacy\_name` varchar(200) NOT NULL,

`phone` varchar(30) NOT NULL,

`email` varchar(50) NOT NULL,

`street` varchar(50) NOT NULL,

`city` varchar(50) NOT NULL,

`state` varchar(50) NOT NULL,

PRIMARY KEY (`pharmacy\_id`)

);

CREATE TABLE `customer` (

`customer\_id` INT NOT NULL AUTO\_INCREMENT,

`first\_name` varchar(40) NOT NULL,

`last\_name` varchar(40) NOT NULL,

`phone` varchar(30) NOT NULL,

`email` varchar(50),

`street` varchar(50),

`city` varchar(50) DEFAULT "Lviv",

`state` varchar(50),

`zip\_code` varchar(50),

`login` varchar(150) NOT NULL,

`password` varchar(150) NOT NULL,

PRIMARY KEY (`customer\_id`)

);

CREATE TABLE `order` (

`order\_id` INT NOT NULL,

`customer\_id` INT NOT NULL UNIQUE,

`pharmacy\_id` INT UNIQUE,

`order\_date` DATETIME NOT NULL,

`order\_status` BINARY NOT NULL,

PRIMARY KEY (`order\_id`),

CONSTRAINT `order\_fk0` FOREIGN KEY (`customer\_id`) REFERENCES `customer`(`customer\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `order\_fk1` FOREIGN KEY (`pharmacy\_id`) REFERENCES `pharmacy`(`pharmacy\_id`) ON DELETE NO ACTION ON UPDATE CASCADE

);

CREATE TABLE `all\_medicine` (

`medicine\_name` varchar(255) NOT NULL,

`medicine\_description` varchar(150) NOT NULL,

`contraindication` varchar(150),

`appliance` varchar(150),

PRIMARY KEY (`medicine\_name`)

);

CREATE TABLE `medicine` (

`medicine\_id` INT NOT NULL AUTO\_INCREMENT,

`pharmacy\_id` INT NOT NULL,

`medicine\_name` varchar(150) NOT NULL,

`quantity` INT NOT NULL,

`price` INT NOT NULL,

PRIMARY KEY (`medicine\_id`),

CONSTRAINT `medicine\_fk0` FOREIGN KEY (`pharmacy\_id`) REFERENCES `pharmacy`(`pharmacy\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `medicine\_fk1` FOREIGN KEY (`medicine\_name`) REFERENCES `all\_medicine`(`medicine\_name`) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE `order\_item` (

`item\_id` INT NOT NULL AUTO\_INCREMENT,

`medicine\_id` INT NOT NULL UNIQUE,

`quantity` INT NOT NULL,

`discount` INT,

`order\_id` INT NOT NULL,

PRIMARY KEY (`item\_id`),

CONSTRAINT `order\_item\_fk0` FOREIGN KEY (`medicine\_id`) REFERENCES `medicine`(`medicine\_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `order\_item\_fk1` FOREIGN KEY (`order\_id`) REFERENCES `order`(`order\_id`) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE `delivery` (

`delivery\_id` INT NOT NULL AUTO\_INCREMENT,

`order\_id` INT NOT NULL UNIQUE,

`delivery\_status` BOOLEAN NOT NULL,

`delivery\_price` INT NOT NULL,

PRIMARY KEY (`delivery\_id`),

CONSTRAINT `delivery\_fk0` FOREIGN KEY (`order\_id`) REFERENCES `order`(`order\_id`) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE `delivery\_man` (

`delivery\_man\_id` INT NOT NULL AUTO\_INCREMENT,

`delivery\_id` INT NOT NULL,

`first\_name` varchar(100) NOT NULL,

`last\_name` varchar(100) NOT NULL,

`phone` varchar(100) NOT NULL,

PRIMARY KEY (`delivery\_man\_id`),

CONSTRAINT `delivery\_man\_fk0` FOREIGN KEY (`delivery\_id`) REFERENCES `delivery`(`delivery\_id`) ON DELETE NO ACTION ON UPDATE NO ACTION

);

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