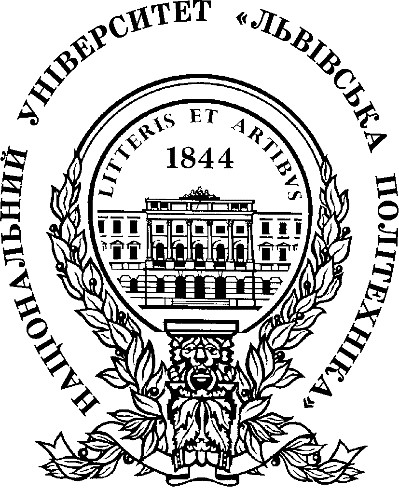
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра систем штучного інтелекту



ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ № 2 **з курсу “ОБДЗ”** на тему:

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

**Виконав:**

студент групи КН-208

Фіняк М.В.

**Викладач:**

Якимишин Х.М.

**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 потрібно використати команду **CREATE DATABASE**. Тут і надалі, квадратні дужки позначають необов’язковий аргумент команди, символ "|" позначає вибір між аргументами.

***Синтаксис :***

CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] ім’я\_бази

[[DEFAULT] CHARACTER SET кодування]

[[DEFAULT] COLLATE набір\_правил]

***Пояснення :***

* 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).

Щоб створити структуру таблиці,потрібно використати оператор **CREATE TABLE**, який визначає ім’я таблиці а також її стовпчики із типами та розмірами. Таблиця повинна містити хоча б один стовпчик.

***Синтаксис :***

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) ✓ **вираз:**

Логічний вираз, що повертає TRUE або FALSE.

* **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)

***Пояснення деяких 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}. |
| **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). |
| **TEMPORARY** | Створення тимчасову таблицю, яка буде знищена після завершення зв’язку із сервером. |
| **CONSTRAINT** | Вказує на початок оголошення PRIMARY KEY, UNIQUE, або FOREIGN KEY обмеження. |
| **NULL | NOT NULL** | Директива, що дозволяє/забороняє nullзначення для даного поля. |
| **FULLTEXT|SPATIAL** | Тип індексу (повнотекстовий/просторовий; тільки для таблиць типу MyISAM). |
| **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). |

***Типи даних (MySQL):***

**Текстові**

|  |  |
| --- | --- |
| **CHAR(size)** | Містить рядок фіксованої довжини (може містити букви, цифри, та інші символи). Фіксована довжина задається в дужках. Може зберігати до 255 символів. |
| **VARCHAR(size)** | Містить рядок змінної довжини. Найбільша довжина задається в дужках. Може зберігати до 255 символів. Примітка: Якщо ви покладете туди значення більше за 255, тип буде перетворений на TEXT. |
| **TINYTEXT** | Рядок з найбільшою довжиною 255 символів |
| **TEXT** | Зберігає рядок з найдовшою довжиною 65,535 символів |
| **BLOB** | Великий двійковий об'єкт (Binary Large OBject).  Зберігає до 65,535 байт даних |
| **MEDIUMBLOB** | Великий двійковий об'єкт. 16 Мегабайт даних |
| **LONGTEXT** | Рядок з найбільшою довжиною в 4,294,967,295 символів. |
| **LONGBLOB** | Великий двійковий об'єкт. 4 Гігабайти даних |
| **ENUM(x,y,z,і т.д.)** | Дозволяє ввести список можливих значень.  Можна перелічити до 65535 різних значень типу. Якщо значення що вставляють в поле не буде належати списку, вставиться порожнє значення. Зауваження: Значення будуть відсортовані в тому порядку в якому ви їх запишете.  Можливі значення вводяться в такому форматі:  ENUM('X','Y','Z') |
| **SET** | Подібно до ENUM окрім того, що SET може містити до 64 значень списку, і не може зберігати більше одного вибору. |

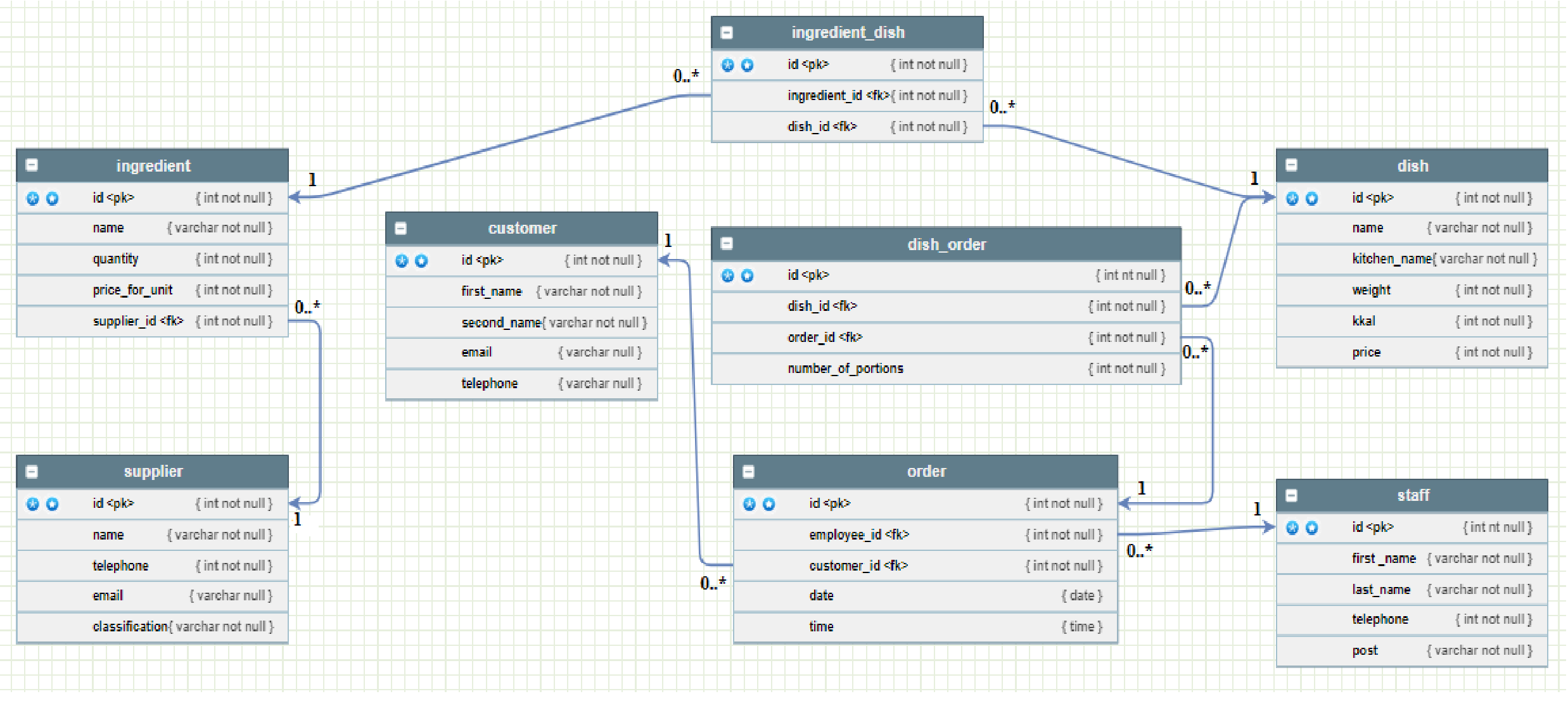
**Числові**

|  |  |
| --- | --- |
| **TINYINT(size)** | Цілий від -128 до 127 . Від 0 до 255 UNSIGNED[[1].](https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_MySQL#cite_note-un-1) Максимальне число цифр задається в дужках. |
| **SMALLINT(size)** | Від -32768 до 32767. Від 0 до 65535  UNSIGNED[[1].](https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_MySQL#cite_note-un-1) Максимальне число цифр задається в дужках. |
| **MEDIUMINT(size)** | Від -8388608 до 8388607. Від 0 до 16777215 UNSIGNED[[1].](https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_MySQL#cite_note-un-1) Максимальне число цифр задається в дужках. |
| **INT(size)** | Від -2147483648 до 2147483647. Від 0 до 4294967295 UNSIGNED[].](https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_MySQL#cite_note-un-1) Максимальне число цифр задається в дужках. |
| **FLOAT(size,d)** | Число з плаваючою крапкою. Максимальне число цифр задається в параметрі size. Максимальне число цифр після десяткової крапки задається в параметрі d. |
| **DOUBLE(size,d)** | Точніше число з плаваючою крапкою. Максимальне число цифр задається в параметрі size. Максимальне число цифр після десяткової крапки задається в параметрі d. |
| **DECIMAL(size,d)** | DOUBLE, що зберігається як рядок з фіксованою крапкою.. Максимальне число цифр задається в параметрі size.  Максимальне число цифр після десяткової крапки задається в параметрі d. |

**Дата і час**

|  |  |
| --- | --- |
| **DATE()** | Дата. Формат: YYYY-MM-DD  Зауваження: Підтримується діапазон від  '1000-01-01' до '9999-12-31' |
| **DATETIME()** | Формат: YYYY-MM-DD HH:MM:SS [[2].](https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_MySQL#cite_note-dt-2)  Зауваження: Підтримується діапазон від  '1000-01-01 00:00:00' до '9999-12-31 23:59:59' |
| **TIME()** | Час. Формат: HH:MM:SS  Зауваження: Підтримується діапазон від '-  838:59:59' до '838:59:59' |
| **YEAR()** | Рік в двоцифровому, або чотирицифровому форматі.  Зауваження: Значення, що дозволені в чотирицифровому форматі: від 1901 до 2155. Значення дозволені в двоцифровому форматі: від 70 до 69, що відповідає 1970 та 2069. |

**Хід роботи**

Будуємо даталогічну модель бази даних,визначивши типи, розмірності та обмеження полів:

Створимо спроектовану базу даних за допомогою таких команд :

**CREATE DATABASE** IF NOT EXISTS `Restaurant` DEFAULT CHARACTER SET utf8 ;

-- Table `supplier`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`supplier` (

`id` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(100) NOT NULL,

`telephone` VARCHAR(45) NOT NULL,

`email` VARCHAR(100) NULL,

`classification` VARCHAR(100) NOT NULL,

PRIMARY KEY (`id`));

-- Table `customer`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`customer` (

`id` INT NOT NULL AUTO\_INCREMENT,

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

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

`email` VARCHAR(100) NULL,

`telephone` VARCHAR(45) NULL,

PRIMARY KEY (`id`));

-- Table `dish`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`dish` (

`id` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(200) NOT NULL,

`kitchen\_name` VARCHAR(100) NOT NULL,

`weight` VARCHAR(45) NOT NULL,

`kkal` INT NOT NULL,

`price` DECIMAL(10,2) NOT NULL,

PRIMARY KEY (`id`));

-- Table `staff`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`staff` (

`id` INT NOT NULL AUTO\_INCREMENT,

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

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

`telephone` VARCHAR(45) NOT NULL,

`post` VARCHAR(100) NOT NULL,

PRIMARY KEY (`id`)

);

-- Table `order`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`order` (

`id` INT NOT NULL AUTO\_INCREMENT,

`employee\_id` INT NOT NULL,

`customer\_id` INT NOT NULL,

`date` DATE NOT NULL,

`time` TIME NOT NULL,

PRIMARY KEY (`id`),

CONSTRAINT `customer\_id`

FOREIGN KEY (`customer\_id`)

REFERENCES `Restaurant`.`customer` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `employee\_id`

FOREIGN KEY (`employee\_id`)

REFERENCES `Restaurant`.`staff` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- Table `dish\_order`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`dish\_order` (

`id` INT NOT NULL AUTO\_INCREMENT,

`dish\_id` INT NOT NULL,

`order\_id` INT NOT NULL,

`number\_of\_portions` INT NOT NULL,

PRIMARY KEY (`id`),

CONSTRAINT `dish\_id`

FOREIGN KEY (`dish\_id`)

REFERENCES `Restaurant`.`dish` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `order\_id`

FOREIGN KEY (`order\_id`)

REFERENCES `Restaurant`.`order` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE)

;

-- Table `ingredient`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`ingredient` (

`id` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(100) NOT NULL,

`quantity` VARCHAR(45) NOT NULL,

`price\_for\_unit` DECIMAL(10,2) NOT NULL,

`supplier\_id` INT NOT NULL,

PRIMARY KEY (`id`),

CONSTRAINT `supplier\_id`

FOREIGN KEY (`supplier\_id`)

REFERENCES `Restaurant`.`supplier` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- Table `ingredient\_dish`

**CREATE TABLE** IF NOT EXISTS `Restaurant`.`ingredient\_dish` (

`id` INT NOT NULL AUTO\_INCREMENT,

`ingredient\_id` INT NOT NULL,

`dish\_id1` INT NOT NULL,

PRIMARY KEY (`id`),

CONSTRAINT `ingredient\_id`

FOREIGN KEY (`ingredient\_id`)

REFERENCES `Restaurant`.`ingredient` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `dish\_id1`

FOREIGN KEY (`dish\_id1`)

REFERENCES `Restaurant`.`dish` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE);

**Висновок :** під час виконання даної лабораторної роботи я побудував даталогічну модель бази даних . При цьому визначив типи, розмірності та обмеження полів і таблиць. Також вивчив декілька SQL команд за допомогою яких створив спроектовані таблиці.