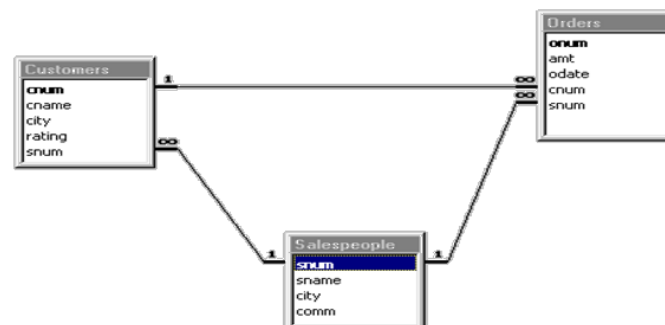


# Бази даних

---

## Загальні відомості частина 1



Навчальний курс  
Валько Н.В.

# SQL

**Мартин Грабер**

МАРТИН ГРУБЕР

# Понимание SQL

Перевод Лебедева В.Н.

Под редакцией Булычева В.Н.

МОСКВА, 1993

Джеймс Р. Грофф  
Пол Н. Вайнберг

# SQL

*Полное руководство*

Второе издание,  
переработанное и дополненное

Перевод с английского  
под редакцией В.Р. Гинзбурга

Линн Бейли

# Изучаем SQL

Приведи  
в порядок  
свои  
отношения  
с данными



Прекрати путать  
термины



Освой концепцию  
и синтаксис SQL  
максимально  
эффективно



Перестань  
смушаться



# Зміст

---

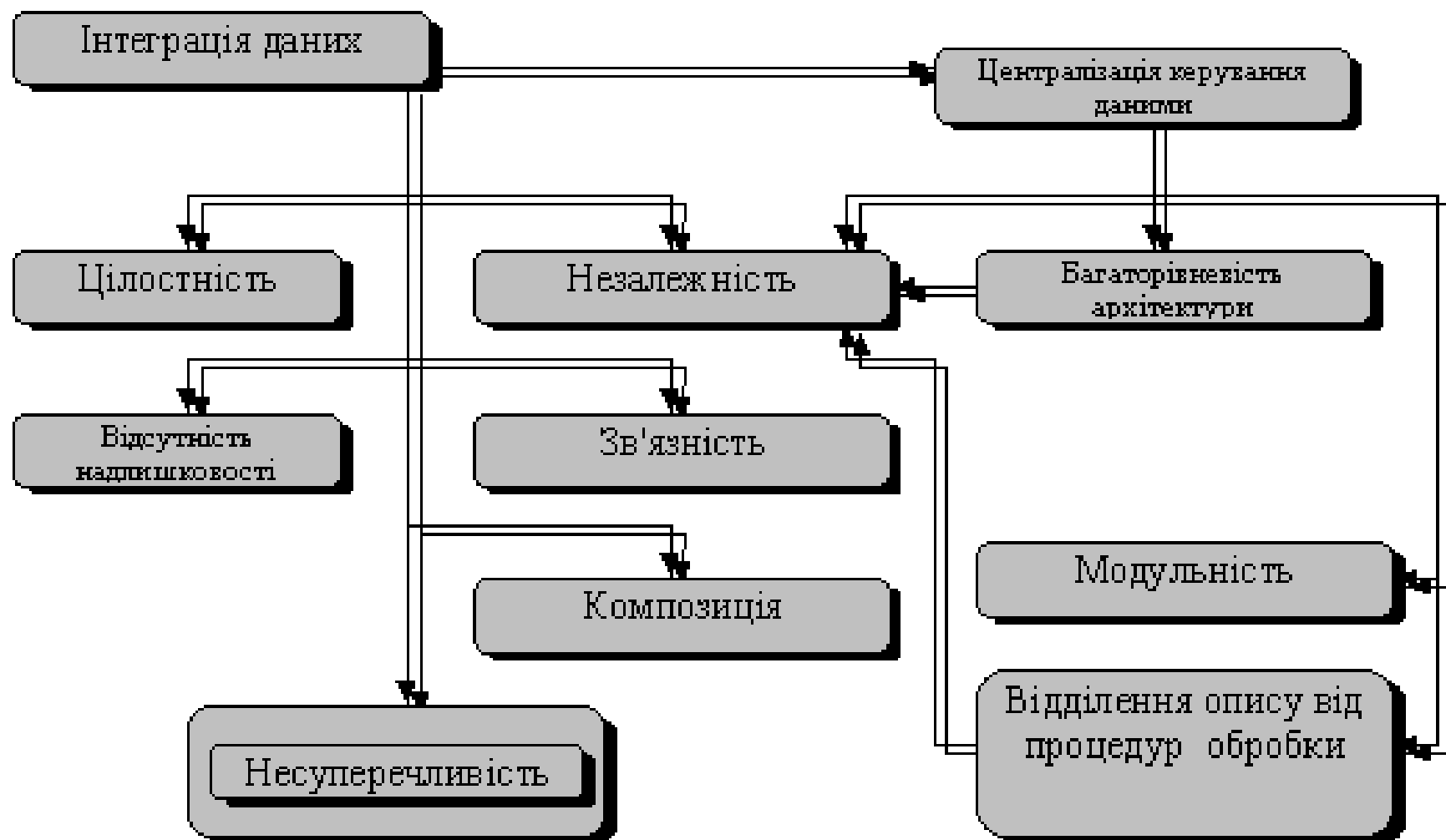
- Поняття БД
- Поняття реляційної БД
- Реляційна модель БД (відношення, атрибути, кортежі, властивості)
- РБД, ключі
- SQL, phpMyAdmin
- DDL, дії з БД
  - створення/видалення
- DDL, дії з таблицями
  - створення
  - видалення
  - зміна
  - перейменування
  - очищення
  - індексація

# Система баз даних

---

- Компьютеризована система зберігання записів призначення якої – зберігати інформацію надаючи засоби для її знаходження і модифікації
- 4 компоненти системи
  - Дані
  - Апаратне забезпечення
  - Програмне забезпечення
  - Користувачі

# Принципи побудови БД



# Принцип інтеграції даних

---

- Об'єднання окремих, взаємно не зв'язаних даних у єдине ціле, в ролі якого виступає база даних. В результаті користувачу і його прикладним програмам всі дані представляються єдиним інформаційним масивом. При цьому полегшуються пошук взаємозалежних даних і їхня спільна обробка, зменшується надлишковість даних, спрощується процес ведення БД.

# Принцип цілісності

---

- це правила, які дають змогу уникнути введення некоректних даних у БД, а також забезпечити можливість зв'язування декількох таблиць. Ці правила можуть бути описані при створенні чи модифікації таблиці



# Принцип централізації керування

---

- полягає в передачі усіх функцій керування даними єдиному комплексу керуючих програм - системі керування базами даних



# БД за типом структури

---

- Структуровані
  - ієрархічна,
  - мережна
  - реляційна
- Не структуровані
  - повнотекстові БД (Вікіпедія)

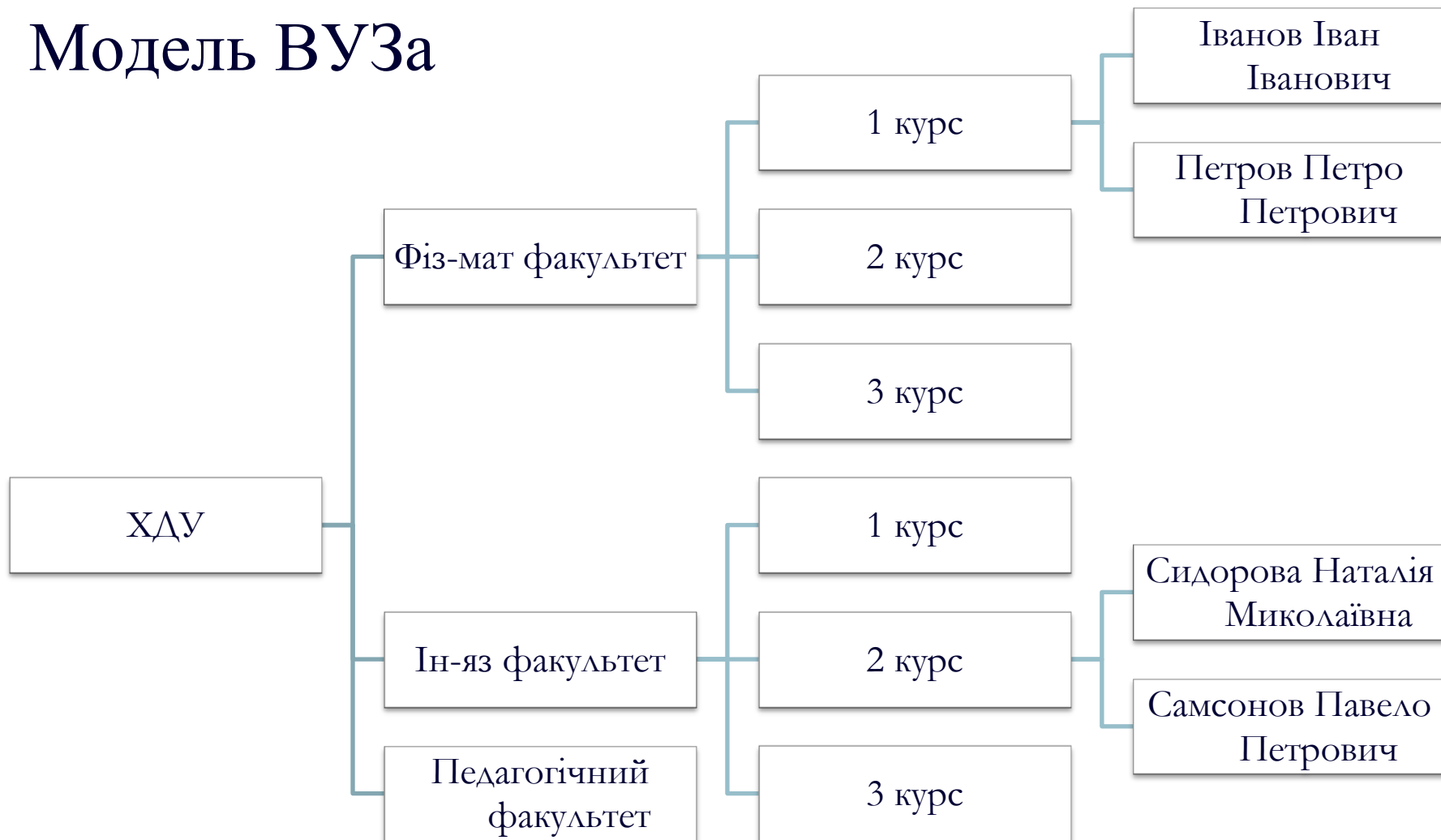
# БД за організацією даних

---

- **Ієрархічна.** Ієрархічна база даних може бути представлена як дерево. Між об'єктами існують зв'язки типу «предок-нащадок».
- **Мережна.** Така база даних подібна до ієрархічної, за винятком того, що кожен об'єкт може мати більше одного предка.
- **Реляційна.** Реляційна база даних зберігає дані у вигляді таблиць.
  - SQL
- **Об'єктно-орієнтована.** У базі даних цього виду дані оформляють у вигляді моделей об'єктів.
  - Графові (NoSQL)

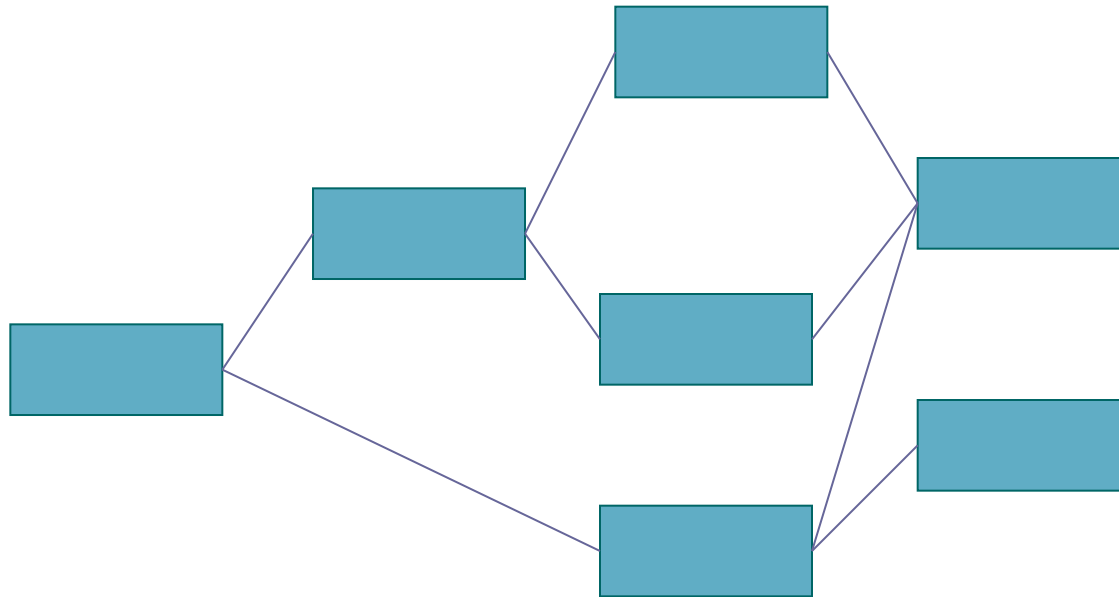
# Ієрархічна модель

## Модель ВУЗа



# Мережева модель

---



# Реляційна модель

| Код  | ППБ            | Дата народження | Стать |
|------|----------------|-----------------|-------|
| 123  | Іванов І.П.    | 10.05.1980      | Ч.    |
| 321  | Петров П.Н.    | 06.12.1976      | Ч.    |
| 4556 | Орлова Г.В.    | 11.11.1962      | Ж.    |
| 111  | Миколаєва Н.Н. | 03.06.1988      | Ж.    |

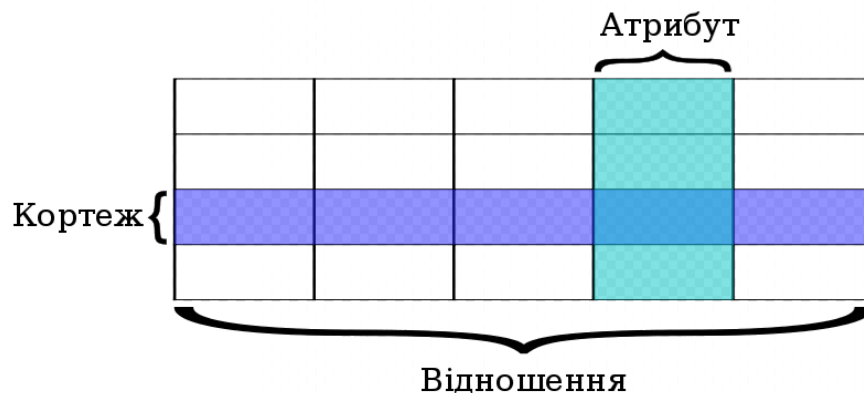
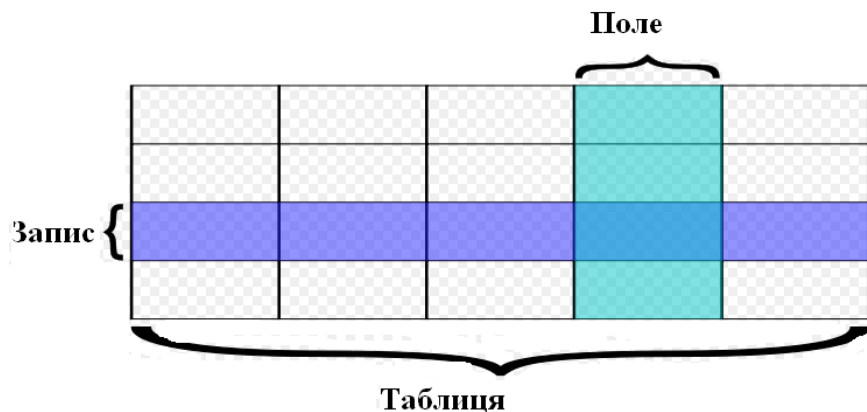
# Реляційна база даних

---

- Дані представлені за допомогою рядків в таблицях і інтерпретуються як істинні висловлювання
- Для обробки рядків надаються оператори які безпосередньо підтримують процес логічного отримання додаткових істинних висловлювань з існуючих висловлювань
- Реляційна алгебра -> SQL

# Реляційна база даних

- це сукупність відношень, що містять всю інформацію, яка повинна зберігатися в базі даних



# Реляційна модель термінологія

| Реляційний термін             | Відповідний "табличний" термін |
|-------------------------------|--------------------------------|
| База даних                    | Набір таблиць                  |
| Схема бази даних              | Набір заголовків таблиць       |
| Відношення                    | Таблиця                        |
| Заголовок відношення          | Заголовок таблиці              |
| Тіло відношення               | Тіло таблиці                   |
| Атрибут відношення            | Назва стовпця таблиці          |
| Кортеж відношення             | Рядок таблиці                  |
| Степінь (-арність) відношення | Кількість стовпців таблиці     |
| Потужність відношення         | Кількість рядків таблиці       |
| Домени і типи даних           | Типи даних в комірках таблиці  |



# Термінологія

---

- На рівні теорії множин ми говоримо "множина", "підмножина декартового добутку", "кортеж".
- На рівні реляційної моделі використовуємо терміни "домен", "відношення", "кортеж".
- На рівні стандарту SQL і конкретних реалізацій використовуємо терміни "тип даних", "таблиця", "рядок таблиці".

# Реляційна модель

## типи даних

---

- Тип даних повинен бути простим - в реляційних операціях не повинна враховуватися внутрішня структура даних
  - Числові
    - З фіксованою точністю і довжиною **decimal**[ (p[ , s] )] и **numeric**[ (p[ , s] )]
    - **int**, **smallint**, **tinyint**, **float**, **real**, **money** (currency), **smallmoney**
    - **bit**
  - Рядки
    - **char(n)**, **varchar(n)**, **nchar(n)**, **nvarchar(n)**
  - Дата і час
    - **datetime**, **smalldatetime**
    - **timestamp**
  - Великі масиви даних і тексти
    - **binary**, **varbinary**, **image**, **text**

# Реляційна модель

## **null** – значення

---

- в ситуації, коли можлива поява невідомих або неповних даних
  - замість невідомих даних вводити або нульові значення, або значення спеціального виду
    - наприклад, домовитися, що рядок "АДРЕСА НЕВІДОМА" і є ті дані, які потрібно вводити замість невідомої адреси
  - Використовувати null-значення замість невідомих даних
    - Мати функції визначення рівності поля null

# Реляційна модель

## відношення, атрибути, кортежі

---

- Заголовок відношення описує декартовий добуток доменів, на якому задано відношення. Заголовок є статичним, він не змінюється під час роботи з базою даних.
  - Якщо у відношенні змінені, додані або видалені атрибути, то в результаті отримаємо вже інше відношення (хай навіть з колишнім ім'ям).
- Тіло відношення є набір кортежів, тобто підмножина декартового добутку доменів. Таким чином, тіло відношення власне і є відношенням в математичному сенсі слова.
  - Тіло відношення може змінюватися під час роботи з базою даних - кортежі можуть змінюватися, додаватися і віддалятися

# Реляційна модель

## Приклад

---

- Заголовок відношення має вигляд:  
`book ( id_book, title, author, price)`
- Відношення містить три кортежі:
  1. (1, Мотивація , Іванов, 100)
  2. (2, Термінал , Петров, 130)
  3. (3, Сьогодні , Сидоров, 85)

# Реляційна модель

## Приклад

| <b>id_book</b> | <b>title</b> | <b>author</b> | <b>price</b> |
|----------------|--------------|---------------|--------------|
| 1              | Мотивація    | Іванов        | 100          |
| 2              | Термінал     | Петров        | 130          |
| 3              | Сьогодні     | Сидоров       | 85           |

# Реляційна модель відношення, атрибути, кортежі

---

*Визначення 1.* **Реляційної базою даних** називається набір відношень.

*Визначення 2.* **Схемою** реляційної бази даних називається набір заголовків відношень, що входять в базу даних.

**відношення не є таблицею!**

# Реляційна модель

## властивості відношень

---

- *У відношенні немає однакових кортежів.*
  - Тілом у відношенні є безліч кортежів. Таблиці на відміну від відношень можуть містити однакові рядки.
- *Кортежі не впорядковані (зверху вниз).*
  - Тілом відношення є множина, а множина не впорядкована. Не можна ототожнити відношення і таблиці - рядки в таблицях впорядковані. Одне і те ж відношення може бути зображено різними таблицями, в яких рядки йдуть в різному порядку.



# Реляційна модель властивості відношень

---

- *Атрибути не впорядковані (зліва направо).*
  - Оскільки кожен атрибут має унікальне ім'я в межах відношення, то порядок атрибутів не має значення. Не можна ототожнити відношення і таблиці - стовпці в таблиці впорядковані. Одне і те ж відношення може бути зображене різними таблицями, в яких стовпці йдуть в різному порядку.
- *Всі значення атрибутів атомарні.*
  - Відмінність відношень від таблиць - в комірки таблиць можна помістити що завгодно - масиви, структури, і навіть інші таблиці.

# Реляційна модель властивості відношень

---

- *Не кожна* таблиця може задавати відношення.
- Для того, щоб деяка таблиця задавала відношення, необхідно
  - щоб таблиця мала просту структуру (містила б тільки рядки і стовпці, причому, в кожному рядку було б однакова кількість полів),
  - в таблиці не повинно бути однакових рядків,
  - будь-який стовпець таблиці повинен містити дані тільки одного типу,
  - всі використовувані типи даних повинні бути простими
- Такі таблиці
  - називаються "плоскими" (plane-tables)
  - **Знаходяться в 1-й нормальній формі**



# DB $\leftrightarrow$ SQL $\leftrightarrow$ Project

---

- Система управління базами даних (СУБД)
- Набір: база даних + програми для доступу до цих даних

# SQL

## □ **DDL Data Definition Language** (*Мова опису даних*)

---

У випадку з SQL ці дієслова:

- Create (Створити)
- Alter (Змінити)
- Drop (Видалити)

## □ **DML Data Manipulation Language** (*Мова маніпулювання даними*)

У випадку з SQL ці дієслова:

- Select (Вибрати)
- Insert (Вставити)
- Update (Оновити/Модифікувати)
- Delete (Видалити)

## □ **DCL Data Control Language** (*Мова контролю даних*)

- GRANT (надати) — дозволити визначеним користувачам виконувати визначені маніпуляції
- REVOKE (скасувати) — скасувати надані права

<https://dev.mysql.com/doc/>



The world's most popular open source database



[MYSQL.COM](#)

[DOWNLOADS](#)

[DOCUMENTATION](#)

[DEVELOPER ZONE](#)

[MySQL Server](#)

[MySQL Enterprise](#)

[Workbench](#)

[InnoDB Cluster](#)

[MySQL NDB Cluster](#)

[Connectors](#)

[More](#)

# MySQL Documentation

MySQL 8.0

[Reference Manual](#)

MySQL 8.0

[Release Notes](#)



[HTML](#) [CSS](#) [JAVASCRIPT](#) [SQL](#) [PYTHON](#) [PHP](#) [BOOTSTRAP](#) [HOW TO](#) [MORE ▾](#) [REFERENCES ▾](#) [EXERCISES ▾](#) [🔔](#) [🌐](#) [🔍](#)

[SQL Full Join](#)  
[SQL Self Join](#)  
[SQL Union](#)  
[SQL Group By](#)  
[SQL Having](#)  
[SQL Exists](#)  
[SQL Any, All](#)  
[SQL Select Into](#)  
[SQL Insert Into Select](#)  
[SQL Case](#)  
[SQL Null Functions](#)  
[SQL Stored Procedures](#)  
[SQL Comments](#)  
  
[SQL Database](#)  
[SQL Create DB](#)  
[SQL Drop DB](#)  
[SQL Backup DB](#)  
[SQL Create Table](#)  
[SQL Drop Table](#)  
[SQL Alter Table](#)

## SQL CREATE DATABASE Statement

[< Previous](#)

[Next >](#)

### The SQL CREATE DATABASE Statement

The CREATE DATABASE statement is used to create a new SQL database.

#### Syntax

```
CREATE DATABASE databasename;
```

# MySQL

одобрено лучшими  
российскими программистами

- [Новости](#)
- [Документация](#)
- [Download](#)
- [Webboard](#)
- [Поиск](#)
- [FAQ / ЧаВо](#)
- [E-mail](#)

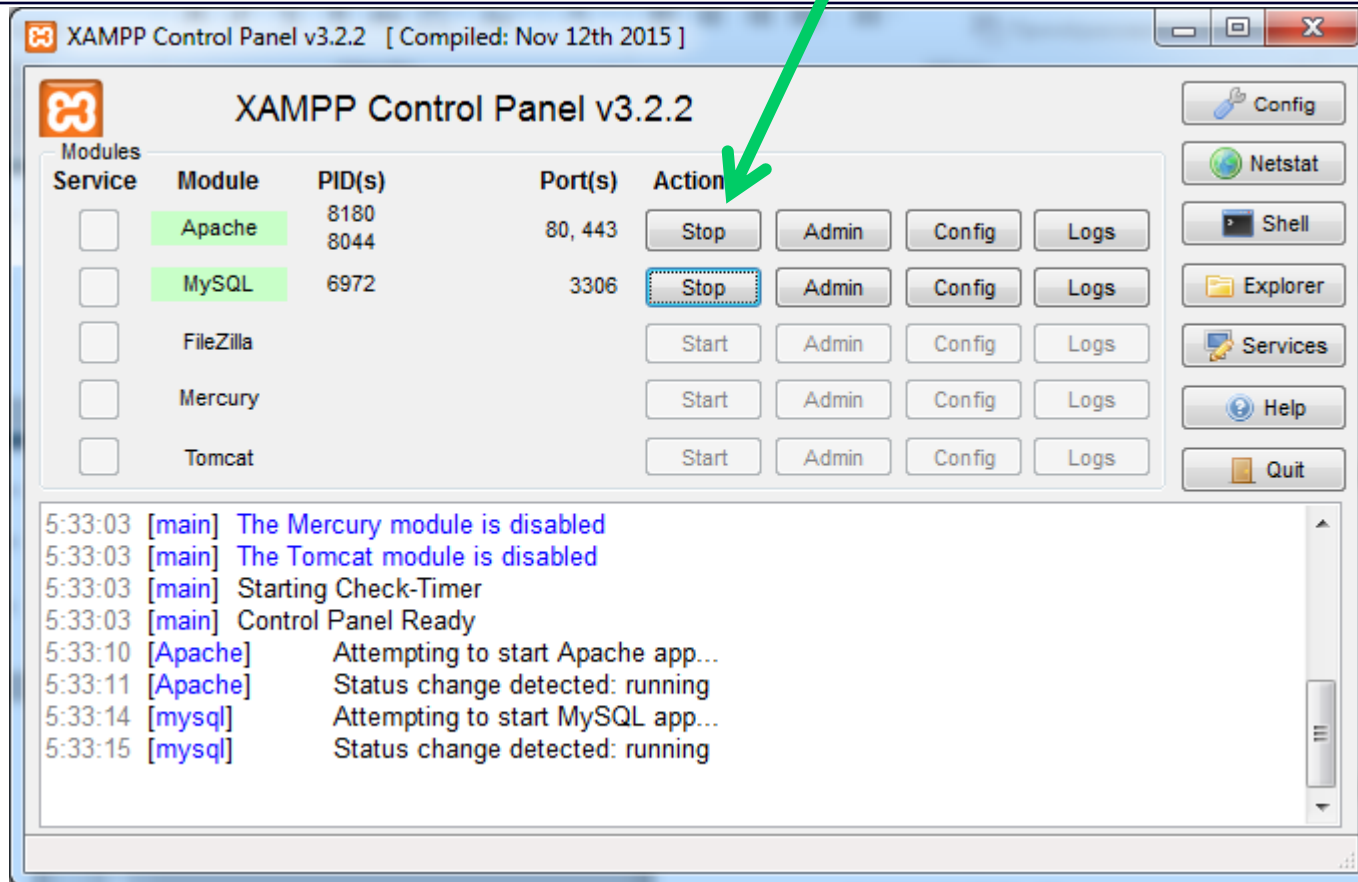
## Справочное руководство по MySQL

### 3.3 Создание и использование базы данных

- 3 Учебное пособие по MySQL
  - 3.1 Подсоединение к серверу и отсоединение от него
  - 3.2 Ввод запросов
  - 3.3 Создание и использование базы данных
    - 3.3.1 Создание и выбор базы данных
    - 3.3.2 Создание таблицы
    - 3.3.3 Загрузка данных в таблицу
    - 3.3.4 Выборка информации из таблицы
  - 3.4 Получение информации о базах данных и таблицах
  - 3.5 Примеры стандартных запросов
  - 3.6 Использование mysql в пакетном режиме
  - 3.7 Запросы проекта "Близнецы" (Twin Project)
  - 3.8 Использование MySQL совместно с Apache

[Previous](#) / [Next](#) / [Up](#) / [Table of Contents](#)

# XAMPP



Ліцензія GNU General Public License

# phpMyAdmin

localhost



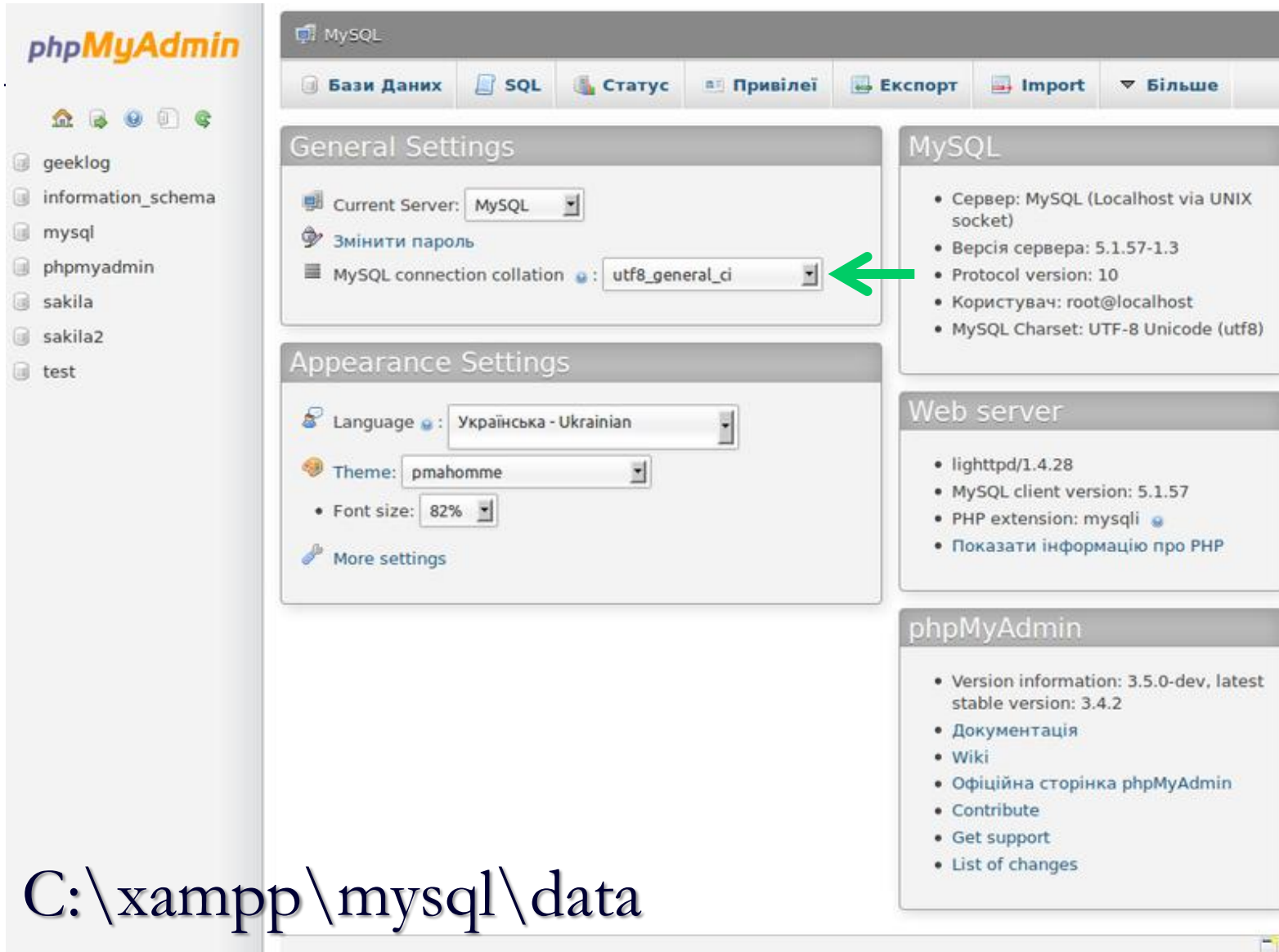
The screenshot shows a web browser window with the address bar set to `localhost/dashboard/`. The browser's bookmark bar includes links to 'Додатки', 'Telegram', 'KSU Online', 'Google Академія', 'Sci-Hub', '3D modeling | Sket...', 'PKP Journal Мелітополь', 'Online C++', 'Строки в языке C+...', 'SpringerJournals', and 'Інші закладки'. The page header features a dark blue navigation bar with links: 'Apache Friends', 'Applications', 'FAQs', 'HOW-TO Guides', 'PHPInfo', and 'phpMyAdmin'. A green arrow points from the word 'localhost' above to the browser's address bar. Another green arrow points from the word 'phpMyAdmin' above to the 'phpMyAdmin' link in the navigation bar. The main content area has a light beige background and displays the XAMPP logo (an orange square with a white 'X') followed by the text 'XAMPP Apache + MariaDB + PHP + Perl'. Below this, the heading 'Welcome to XAMPP for Windows 7.3.1' is shown. The text explains that XAMPP is installed successfully and provides links to 'FAQs' and 'HOW-TO Guides'. It also includes a warning about security for production use, suggesting 'WAMP', 'MAMP', or 'LAMP' as alternatives. At the bottom, it instructs to 'Start the XAMPP Control Panel to check the server status.' In the bottom right corner, there is a logo for 'phpMyAdmin' featuring a stylized sailboat.

Community

Ліцензія GNU General Public License



# XAMPP



The screenshot displays the phpMyAdmin web interface. On the left is a sidebar with the phpMyAdmin logo and a list of databases: geeklog, information\_schema, mysql, phpmyadmin, sakila, sakila2, and test. The main content area is titled 'MySQL' and contains several tabs: Бази Даних, SQL, Статус, Привілеї, Експорт, Import, and Більше. The 'General Settings' tab is active, showing 'Current Server' as 'MySQL' and 'MySQL connection collation' as 'utf8\_general\_ci', with a green arrow pointing to the latter. Below this is the 'Appearance Settings' section, which includes 'Language' set to 'Українська - Ukrainian', 'Theme' set to 'pmahomme', and 'Font size' set to '82%'. To the right of the settings are three summary panels: 'MySQL' (listing server details like version 5.1.57-1.3 and user root@localhost), 'Web server' (listing httpd/1.4.28 and MySQL client version 5.1.57), and 'phpMyAdmin' (listing version 3.5.0-dev and links to documentation and support).

phpMyAdmin

Бази Даних SQL Статус Привілеї Експорт Import Більше

General Settings

Current Server: MySQL

Змінити пароль

MySQL connection collation: utf8\_general\_ci

Appearance Settings

Language: Українська - Ukrainian

Theme: pmahomme

Font size: 82%

More settings

MySQL

- Сервер: MySQL (Localhost via UNIX socket)
- Версія сервера: 5.1.57-1.3
- Protocol version: 10
- Користувач: root@localhost
- MySQL Charset: UTF-8 Unicode (utf8)

Web server

- lighttpd/1.4.28
- MySQL client version: 5.1.57
- PHP extension: mysqli
- Показати інформацію про PHP

phpMyAdmin

- Version information: 3.5.0-dev, latest stable version: 3.4.2
- Документація
- Wiki
- Офіційна сторінка phpMyAdmin
- Contribute
- Get support
- List of changes

C:\xampp\mysql\data



# C:\xampp\mysql\data

---

## Файл

tbl\_name.frm

tbl\_name.ibd

tbl\_name.myi

bd\_name.opt

## Призначення

Файл структури таблиці

Файл значень таблиці

Файл індексів

Файл підтримки БД

# Визначення даних (DDL)

---

## □ DB

- CREATE DATABASE
- DELETE DATABASE

## □ TABLE

- CREATE TABLE
- DROP TABLE
- ALTER TABLE
- RENAME TABLE
- TRUNCATE
- CREATE INDEX
- DROP INDEX

# Створення БД

---

**CREATE DATABASE [IF NOT EXISTS]**  
database\_name

## Приклад

**CREATE DATABASE** library;  
**CREATE DATABASE IF NOT EXISTS** library;

■ **SHOW DATABASES;**      ■ **USE** database\_name;

# Видалення БД

---

**DROP DATABASE [IF EXISTS]**

database\_name

## Приклад

**DROP DATABASE** library;

**DROP DATABASE IF EXISTS** library;

# Типи даних

# DATATYPE

---

|                                |                                    |
|--------------------------------|------------------------------------|
| Рядки символів фікс. Довжини   | CHAR(довжина) / CHARACTER(довжина) |
| Рядки символів змінної довжини | VARCHAR(довжина)                   |
| Довгий текст                   | LONG VARCHAR / TEXT / MEMO         |
| Цілі числа різних розмірів     | TINYINT / SMALLINT / INT(INTEGER)  |
| Масштабовані цілі              | DECIMAL(p,s) / NUMERIC(p,s)        |
| Грошові величини               | MONEY / CURRENCY                   |
| Числа з плаваючою комою        | FLOAT / REAL / DOUBLE PRECISION    |
| Дата/час                       | DATE / TIME/ TIMESTAMP(DATETIME)   |
| Логічні (булеві) значення      | LOGICAL / BOOLEAN                  |
| Потік байтів, об'єкти, графіка | BINARY/ IMAGE                      |

---



# Створення таблиці

---

```
CREATE TABLE table_name  
(  
    column1 DATATYPE,  
    column2 DATATYPE,  
    column3 DATATYPE,  
    ....  
);
```

# Приклад створення таблиці

| id_book | title |
|---------|-------|
|---------|-------|

```
CREATE TABLE book  
( id_book INT(20),  
  title VARCHAR(10)  
)
```

```
1 CREATE TABLE book  
2 ( id_book INT(20),  
3   title VARCHAR(10)  
4 )
```

```
1 CREATE TABLE book ( id_book INT(20), title VARCHAR(10) )
```

|                          | # | Ім'я    | Тип         | Зіставлення     | Атрибути | Нуль | За замовчуванням | Коментарі |
|--------------------------|---|---------|-------------|-----------------|----------|------|------------------|-----------|
| <input type="checkbox"/> | 1 | id_book | int(20)     |                 |          | Так  | NULL             |           |
| <input type="checkbox"/> | 2 | title   | varchar(10) | utf8_general_ci |          | Так  | NULL             |           |



# Приклад створення таблиці

id\_book title

```
CREATE TABLE library.book  
( id_book INT(20),  
  title VARCHAR(10)  
)
```

|                          | # | Ім'я    | Тип         | Зіставлення     | Атрибути | Нуль | За замовчуванням | Коментарі |
|--------------------------|---|---------|-------------|-----------------|----------|------|------------------|-----------|
| <input type="checkbox"/> | 1 | id_book | int(20)     |                 |          | Так  | NULL             |           |
| <input type="checkbox"/> | 2 | title   | varchar(10) | utf8_general_ci |          | Так  | NULL             |           |

# Створення таблиці (цілісність)

---

CREATE TABLE *table name*

(*column\_name* DATATYPE (size)

[<colcnstrnt> ...], ...

[<tabcnstrnt>] , ... );

NOT NULL (не нульовий),

UNIQUE (унікальний),

PRIMARY KEY (первинний ключ),

CHECK(<predicate>) (перевірка предиката),

DEFAULT = <value expression> (за змовчуванням = виразу)

REFERENCES <table name> [(<column name> ,... )]

(посилання на назва таблиці [(назва стовпця) ] )

# Створення таблиці (цілісність)

---

CREATE TABLE *table name*

( *column\_name* DATATYPE (size)

[<colcnstrnt> ...] , ... ,

[<tabcnstrnt>] , ... );

UNIQUE (унікальний),

PRIMARY KEY (первинний ключ),

CHECK (перевірка предиката)

FOREIGN KEY(<column name>) (зовнішній ключ)

REFERENCES <table name> [(<column name> ,... )]

(посилання на назва таблиці [(назва стовпця) ] ) ].

# Типи правил цілісності

---

- ❑ **CHECK** - Контроль допустимих значень атрибутів.
- ❑ **NOT NULL/NULL** - Заборона/ дозвіл на використання не заданих або не визначених значень.
- ❑ **UNIQUE** - Контроль унікальності значень атрибутів.
- ❑ **PRIMARY KEY** - Первинний ключ.
- ❑ **FOREIGN KEY** - Зовнішній ключ.

# Цілістність даних і сутностей

---

## □ Цілістність даних

- Значення належать домену
- Перевірки на допустимість значень визначеного типу даних

## □ Цілістність сутностей

- Потенціальні ключі
- Первичний (головний) ключ (*primary key*)
- Зовнішні ключі (*foreign key*)

# Ключі

---

## Визначення

- Потенційний ключ, що складається з одного атрибута, називається **простим**.
- Потенційний ключ, що складається з декількох атрибутів, називається **складеним**
- Один з потенційних ключів оголошується **первинним**

## Зауваження

- Потенційні ключі служать **засобом ідентифікації об'єктів** предметної області, дані про яких зберігаються у відношенні. Об'єкти предметної області повинні бути помітні.
- Потенційні ключі служать **єдиним засобом** адресації на рівні кортежів у відношенні (записів в таблиці). Точно вказати будь-який кортеж можна тільки знаючи значення його потенційного ключа

# Приклад створення таблиці

|         |       |
|---------|-------|
| id_book | title |
|---------|-------|

```
CREATE TABLE library.book  
(id_book INT(20) NOT NULL AUTO_INCREMENT ,  
title VARCHAR(10) NOT NULL ,  
PRIMARY KEY (id_book)  
)  
ENGINE = InnoDB;
```

обробник таблиць, що забезпечує  
безпечні транзакції

або ...

| # | Ім'я    | Тип      | Зіставлення     | Атрибути | Нуль | За замовчуванням | Коментарі | Додатково | Дія                      |
|---|---------|----------|-----------------|----------|------|------------------|-----------|-----------|--------------------------|
| 1 | id_book | int(20)  |                 |          | Так  | NULL             |           |           | Змінити  Знищити  Більше |
| 2 | title   | char(20) | utf8_general_ci |          | Так  | NULL             |           |           | Змінити  Знищити  Більше |

# Приклад створення таблиці








|         |       |
|---------|-------|
| id_book | title |
|---------|-------|

або

```
CREATE TABLE library.book
```

```
(code INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
title VARCHAR(10) NOT NULL ,  
)
```

```
ENGINE = InnoDB;
```

| # | Ім'я  | Тип      | Зіставлення     | Атрибути | Нуль | За замовчуванням | Коментарі | Додатково | Дія  |
|---|---|----------|-----------------|----------|------|------------------|-----------|-----------|--|
| 1 | id_book   | int(20)  |                 |          | Так  | NULL             |           |           |  Змінити  Знищити  Більше |
| 2 | title  | char(20) | utf8_general_ci |          | Так  | NULL             |           |           |  Змінити  Знищити  Більше |



# Створення таблиці

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name [(create_definition,...)]
[table_options] [select_statement]
```

create\_definition:

```
col_name type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT]
[PRIMARY KEY] [reference_definition]
```

или PRIMARY KEY (index\_col\_name,...)  
или KEY [index\_name] (index\_col\_name,...)  
или INDEX [index\_name] (index\_col\_name,...)  
или UNIQUE [INDEX] [index\_name] (index\_col\_name,...)  
или FULLTEXT [INDEX] [index\_name] (index\_col\_name,...)  
или [CONSTRAINT symbol] FOREIGN KEY [index\_name] (index\_col\_name,...)  
[reference\_definition]  
или CHECK (expr)

type:

TINYINT[(length)] [UNSIGNED] [ZEROFILL]  
или SMALLINT[(length)] [UNSIGNED] [ZEROFILL]  
или MEDIUMINT[(length)] [UNSIGNED] [ZEROFILL]  
или INT[(length)] [UNSIGNED] [ZEROFILL]  
или INTEGER[(length)] [UNSIGNED] [ZEROFILL]  
или BIGINT[(length)] [UNSIGNED] [ZEROFILL]  
или REAL[(length,decimals)] [UNSIGNED] [ZEROFILL]  
или DOUBLE[(length,decimals)] [UNSIGNED] [ZEROFILL]  
или FLOAT[(length,decimals)] [UNSIGNED] [ZEROFILL]  
или DECIMAL(length,decimals) [UNSIGNED] [ZEROFILL]  
или NUMERIC(length,decimals) [UNSIGNED] [ZEROFILL]  
или CHAR(length) [BINARY]  
или VARCHAR(length) [BINARY]  
или DATE  
или TIME  
или TIMESTAMP  
или DATETIME  
или TINYBLOB  
или BLOB  
или MEDIUMBLOB  
или LONGBLOB  
или TINYTEXT  
или TEXT  
или MEDIUMTEXT  
или LONGTEXT  
или ENUM(value1,value2,value3,...)  
или SET(value1,value2,value3,...)

index\_col\_name:  
col\_name [(length)]

reference\_definition:

```
REFERENCES tbl_name [(index_col_name,...)]
[MATCH FULL | MATCH PARTIAL]
[ON DELETE reference_option]
[ON UPDATE reference_option]
```

reference\_option:

```
RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT
```

table\_options:

TYPE = {BDB | HEAP | ISAM | InnoDB | MERGE | MRG\_MYISAM | MYISAM }  
или AUTO\_INCREMENT = #  
или AVG\_ROW\_LENGTH = #  
или CHECKSUM = {0 | 1}  
или COMMENT = "string"  
или MAX\_ROWS = #  
или MIN\_ROWS = #  
или PACK\_KEYS = {0 | 1 | DEFAULT}  
или PASSWORD = "string"  
или DELAY\_KEY\_WRITE = {0 | 1}  
или ROW\_FORMAT= { default | dynamic | fixed | compressed }  
или RAID\_TYPE= {1 | STRIPED | RAID0 } RAID\_CHUNKS=# RAID\_CHUNKSIZE=#  
или UNION = (table\_name,[table\_name...])  
или INSERT\_METHOD= {NO | FIRST | LAST }  
или DATA DIRECTORY="абсолютный путь к каталогу"  
или INDEX DIRECTORY="абсолютный путь к каталогу"

select\_statement:

```
[IGNORE | REPLACE] SELECT ... (любое корректное выражение SELECT)
```

# Заповнення рядків таблиці

---

```
INSERT INTO table_name (col1, col2, ...)  
VALUES (dat_1, dat_2, ...);
```

## □ Приклад

```
INSERT INTO book  
    (id_book, title, datt)  
VALUES (45, 'Tom Sawyer', '2020-02-24');
```

# Заповнення рядків таблиці

---

```
INSERT INTO table_name (col1, col2, ...)  
    VALUES (dat_11, dat_12, ...) , ... ,  
            (dat_21, dat_22, ...)
```

## □ Приклад

```
INSERT INTO book (title, author)  
    VALUES ('Tom Sawyer', 'Mark Twain') ,  
            ('ТАОСР' , ' Donald Knuth')
```

**Можна вказати не всі поля?**



# Очищення таблиці

---

**TRUNCATE TABLE** *table name*;

Приклад

**TRUNCATE TABLE** book;

**Таблиця залишиться?**

# Видалення таблиці

---

**DROP TABLE [IF EXISTS] *table name*;**

Приклад

**DROP TABLE book;**

**DROP TABLE IF EXISTS book;**

**Чи можна видалити НЕ пусту таблицю?**



# Перейменування таблиці

---

```
RENAME TABLE tbl_name TO new_tbl_name  
[, tbl_name2 TO new_tbl_name2,...]
```

Приклад

```
RENAME TABLE book TO my_book
```

# Зміна структури таблиці

---

□ **ALTER TABLE** *tab\_name* **XXX** *col\_name datatype*;

**CHANGE** – зміна імені або типу даних полів

**MODIFY** – зміна типу даних або позиції полів

**ADD** – додавання полів в таблицю (тип даних)

**DROP** – видалення полів з таблиці

**RENAME TO** – перейменування

- В одній команді **ALTER TABLE** можна використовувати декілька однотипних команд (через кому)

- Може привести до втрати даних !!!

# Зміна структури таблиці

---

|         |       |        |
|---------|-------|--------|
| id_book | title | Author |
|---------|-------|--------|

## Приклади

- ❑ ALTER TABLE book **ADD** Author varchar(255);
- ❑ ALTER TABLE book  
**MODIFY COLUMN** Author var(20);
- ❑ ALTER TABLE book **DROP COLUMN** Author;



# Зміна структури таблиці

---

```
ALTER [IGNORE] TABLE tbl_name alter_spec [, alter_spec ...]
```

alter\_specification:

```
    ADD [COLUMN] create_definition [FIRST | AFTER column_name ]
или   ADD [COLUMN] (create_definition, create_definition,...)
или   ADD INDEX [index_name] (index_col_name,...)
или   ADD PRIMARY KEY (index_col_name,...)
или   ADD UNIQUE [index_name] (index_col_name,...)
или   ADD FULLTEXT [index_name] (index_col_name,...)
или   ADD [CONSTRAINT symbol] FOREIGN KEY index_name (index_col_name,...)
        [reference_definition]
или   ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}
или   CHANGE [COLUMN] old_col_name create_definition
        [FIRST | AFTER column_name]
или   MODIFY [COLUMN] create_definition [FIRST | AFTER column_name]
или   DROP [COLUMN] col_name
или   DROP PRIMARY KEY
или   DROP INDEX index_name
или   DISABLE KEYS
или   ENABLE KEYS
или   RENAME [TO] new_tbl_name
или   ORDER BY col
или   table_options
```

# Індексація в Базах Даних

---

- Впорядкування записів за ознакою
  - Створення окремої таблиці індексів полів для
    - Primary key (Foreign key, Unique)
    - Поля, які ЧАСТО використовуються у SELECT
      - Where column=
      - Order by
      - Group by
      - On
  - НЕ рекомендується індекс для полів
    - Like
    - Update, Insert
    - Having

# Індексація в Базах Даних

---

- HashMap – для великої кількості індексів (текст)
- BitMap – для малої кількості індексів
- Бінарне дерево – для унікальних значень
- В-дерево – індексація за частиною значення

# Створення індексу

---

```
CREATE INDEX index_name  
ON table_name (column1, column2, ...);
```

```
CREATE UNIQUE INDEX index_name  
ON table_name (column1, column2, ...);
```

## Приклад

```
CREATE INDEX idx_title ON book (title)
```

# Питання

---

- ❑ БД і СУБД одне й те ж саме?
- ❑ Які існують види БД?
- ❑ Чим таблиця відрізняється від відношення?
- ❑ SQL це мова програмування?
- ❑ Що може статися якщо змінити тип поля?
- ❑ Для чого створюють індексоване поле?



Далі буде...

---