**ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ**

**ОДЕСЬКИЙ КОЛЕДЖ КОМП’ЮТЕРНИХ ТЕХНОЛОГІЙ**

ЦК комп’ютерних систем

**Пояснювальна записка**

до курсового проекту

з дисципліни «Бази даних»

на тему “Розробка інформаційної системи для туристичної фірми Камчатка”

Виконав: студент III курсу, групи 16ПР1

галузі знань (спеціальності)

12 – Інформаційні технології

121 – Інженерія програмного забезпечення

Разводовський Владислав Русланович

Керівник Імерідзе Максим Борисович

Оцінка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Одеса - 2019

**ЗМІСТ**

[РОЗДІЛ 1. База даних 4](#_Toc10564630)

[1.1. Визначення 4](#_Toc10564631)

[1.2. Типи БД 4](#_Toc10564632)

[1.3. Реляційна БД 5](#_Toc10564633)

[1.4. SQL 5](#_Toc10564634)

[1.5. DDL 6](#_Toc10564635)

[1.6. DML 6](#_Toc10564636)

[1.7. СКБД 7](#_Toc10564637)

[РОЗДІЛ 2. Використані інструменти 9](#_Toc10564638)

[2.1. HTML 9](#_Toc10564639)

[2.2. CSS 10](#_Toc10564640)

[2.4. JavaScript (JS) 11](#_Toc10564641)

[2.5. PHP 12](#_Toc10564642)

[2.6. MySQL 12](#_Toc10564643)

[2.7. phpMyAdmin 13](#_Toc10564644)

[2.8. Локальний сервер Open Server Panel 13](#_Toc10564645)

[2.9. TCPDF 13](#_Toc10564646)

[РОЗДІЛ 3. Створення веб-сторінки 14](#_Toc10564647)

[3.1. Створення бази даних 14](#_Toc10564648)

[3.1.1. Створення таблиці 15](#_Toc10564649)

[3.1.2. Створення зв'язків між таблицями 18](#_Toc10564650)

[3.1.3. Створення БД для роботи зі сайтом 22](#_Toc10564651)

[3.2. Створення головної сторінки для звичайного користувача 24](#_Toc10564652)

[3.2.1. Сторінка оплати 26](#_Toc10564653)

[3.2.2. Сторінка входу 31](#_Toc10564654)

[3.3. Сторінка адміністратора 33](#_Toc10564655)

[3.3.1. Редагування таблиць 33](#_Toc10564656)

[3.3.1.1. Вставка 33](#_Toc10564657)

[3.3.1.2. Збереження 35](#_Toc10564658)

[3.3.1.3. Видалення 36](#_Toc10564659)

[3.3.2. Звіти 37](#_Toc10564660)

[3.3.2.1. Дохід фірми від кожного туру і за всіма турам в цілому 38](#_Toc10564661)

[3.3.2.2. Список клієнтів по одному туру і всім турам 40](#_Toc10564662)

[3.3.2.3. Тривалість кожного туру. Середня тривалість туру 42](#_Toc10564663)

[3.3.2.4. Список турів за кожен місяць та рік 43](#_Toc10564664)

[СПИСОК ДЖЕРЕЛ 45](#_Toc10564665)

# **РОЗДІЛ 1. База даних**

* 1. **Визначення**

Базою даних (БД) називається організована відповідно до визначеними правилами і підтримувана в пам'яті комп'ютера сукупність відомостей про об'єкти, процеси, події чи явища, що відносяться до недоторою предметної області, темі або завданню. Вона організована таким чином, щоб забезпечити інформаційні потреби користувачів, а також удобне зберігання цієї сукупності даних, як в цілому, так і її частини.

* 1. **Типи БД**

За моделлю організації даних розрізняють такі бази даних:

* Ієрархічна. Ієрархічна база даних може бути представлена як дерево, що складається з об'єктів різних рівнів. Між об'єктами існують зв'язки типу «предок-нащадок». При цьому можлива ситуація, коли об'єкт не має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок.
* 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. Найвживаніші [СКБД](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) використовують [реляційну модель даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).
* Об'єктно-орієнтована. У базі даних цього виду дані оформляють у вигляді моделей об'єктів.
  1. **Реляційна БД**

Реляційна база даних являє собою безліч взаімосвязанних таблиць, кожна з яких містить інформацію про об'єкти певного виду. Кожен рядок таблиці містить дані про один об'єкт (наприклад, автомобілі, комп'ютері, клієнті), а стовпці таблиці містять различ- ні характеристики цих об'єктів - атрибути (наприклад, номер двигуна, марка процесора, телефони фірм або клієнтів).

Рядки таблиці називаються записами. Всі записи таблиці мають одинаковую структуру - вони складаються з полів (елементів даних), в яких зберігаються атрибути об'єкта. Кожне поле запису містить одну характеристику об'єкта і являє собою заданий тип даних (наприклад, текстовий рядок, число, дата). Для ідентифікації записів використовується первинний ключ. Первинним ключем називається набір полів таблиці, комбінація значен ний яких однозначно визначає кожний запис у таблиці.

* 1. **SQL**

SQL — [декларативна мова програмування](https://uk.wikipedia.org/wiki/%D0%94%D0%B5%D0%BA%D0%BB%D0%B0%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) для взаємодії користувача з [базами даних](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), що застосовується для формування запитів, оновлення і керування [реляційними БД](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних. Сама по собі SQL не є ані системою керування базами даних, ані окремим програмним продуктом. На відміну від дійсних мов програмування ([C](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_C) або [Pascal](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_Pascal)), SQL може формувати інтерактивні запити або, бувши вбудованою в прикладні програми, виступати як інструкції для керування даними. Окрім цього, стандарт SQL містить функції для визначення зміни, перевірки та захисту даних.

Це діалогова мова програмування для здійснення [запиту](https://uk.wikipedia.org/wiki/%D0%9F%D1%96%D0%B4%D0%B7%D0%B0%D0%BF%D0%B8%D1%82) і внесення змін до бази даних, а також керування базами даних. Багато баз даних підтримує SQL з розширеннями до стандартної мови. Ядро SQL формує командна мова, яка дозволяє здійснювати пошук, вставку, оновлення і вилучення даних за допомогою використання системи керування і адміністративних функцій. SQL також включає CLI (Call Level Interface) для доступу і керування базами даних дистанційно.

* 1. **DDL**

Data Definition Language (DDL) (Мова опису даних) — це сімейство комп'ютерних мов, що використовуються в комп'ютерних програмах або користувачами баз даних для опису структури даних. Вперше термін було використано в моделі бази даних CODASYL, де структура бази була описана Data Definition Language та визначала записи, поля, та «набори» («sets»), що являють собою користувацьку модель даних.

DDL мають свою функціональну здатність, організовану за початковим словом в заяві (запит), яке майже завжди є дієсловом. У випадку з SQL ці дієслова:

* [Create (Створити)](https://uk.wikipedia.org/wiki/Create_(SQL))
* [Alter (Змінити)](https://uk.wikipedia.org/wiki/Alter_(SQL))
* [Drop (Видалити)](https://uk.wikipedia.org/wiki/Drop_(SQL))
  1. **DML**

Data Manipulation Language (DML) (Мова маніпулювання даними) — це сімейство комп'ютерних мов, що використовуються в комп'ютерних програмах або користувачами баз даних для отримання, вставки, видалення або зміни даних в базах даних.

Тепер найпопулярнішою мовою DML є [SQL](https://uk.wikipedia.org/wiki/SQL), що використовується для отримання і маніпулювання даними в реляційній базі даних. Мови DML спочатку використовувалися лише комп'ютерними програмами, але з появою SQL вони стали доступними і для людей.

DML мають свою функціональну здатність, організовану за початковим словом в заяві (запит), яке майже завжди є дієсловом.

У випадку з SQL ці дієслова:

* [Select (Вибрати)](https://uk.wikipedia.org/wiki/Select_(SQL))
* [Insert (Вставити)](https://uk.wikipedia.org/wiki/Insert_(SQL))
* [Update (Оновити/Модифікувати)](https://uk.wikipedia.org/wiki/Update_(SQL))
* [Delete (Видалити)](https://uk.wikipedia.org/wiki/Delete_(SQL))
  1. **СКБД**

Система керування ба́зами да́них— набір взаємопов'язаних даних ([база даних](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)) і програм для доступу до цих даних. Надає можливості створення, збереження, оновлення та пошуку інформації в [базах даних](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) з контролем доступу до даних.

Основні характеристики СКБД:

* Контроль за надлишковістю даних
* Несуперечливість даних
* Підтримка цілісності бази даних (коректність та несуперечливість)
* Цілісність описується за допомогою обмежень
* Незалежність прикладних програм від даних
* Спільне використання даних
* Підвищений рівень безпеки

Можливості СКБД:

* Дозволяється створювати БД (здійснюється за допомогою мови визначення даних DDL)
* Дозволяється додавання, оновлення, видалення та читання інформації з БД (за допомогою мови маніпулювання даними [DML](https://uk.wikipedia.org/wiki/DML), яку часто називають мовою запитів).

# **РОЗДІЛ 2. Використані інструменти**

* 1. **HTML**

Hypertext Markup Language (HTML) — стандартна [мова розмітки](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D1%80%D0%BE%D0%B7%D0%BC%D1%96%D1%82%D0%BA%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) для створення [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0) і [веб-додатків](https://uk.wikipedia.org/w/index.php?title=%D0%92%D0%B5%D0%B1-%D0%B4%D0%BE%D0%B4%D0%B0%D1%82%D0%BA%D1%96%D0%B2&action=edit&redlink=1) . З [Cascading Style Sheets](https://uk.wikipedia.org/wiki/CSS) (CSS) і [JavaScript](https://uk.wikipedia.org/wiki/JavaScript), вона утворює тріаду основних технологій для [World Wide Web](https://uk.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D1%81%D0%B2%D1%96%D1%82%D0%BD%D1%94_%D0%BF%D0%B0%D0%B2%D1%83%D1%82%D0%B8%D0%BD%D0%BD%D1%8F).

[Веб-браузери](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B1%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80%D0%B8) отримують HTML-документи з [веб-сервера](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) або з локальної пам'яті і передають документи в мультимедійні веб-сторінки. HTML описує структуру веб-сторінки [семантично](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D0%BC%D0%B0%D0%BD%D1%82%D0%B8%D1%87%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0) і спочатку включені сигнали для зовнішнього вигляду документа.

[Елементи HTML](https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%B8_HTML) є будівельними блоками сторінок HTML. За допомогою конструкцій HTML, зображення та інші об'єкти, такі як [інтерактивні форми](https://uk.wikipedia.org/w/index.php?title=%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D1%96_%D1%84%D0%BE%D1%80%D0%BC%D0%B8&action=edit&redlink=1), можуть бути вбудовані у візуалізовану сторінку. HTML надає засоби для створення [структурованих документів](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%85_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D1%96%D0%B2&action=edit&redlink=1), позначаючи структурну [семантику](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D0%BC%D0%B0%D0%BD%D1%82%D0%B8%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) тексту, наприклад заголовки, абзаци, списки, [посилання](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%B8%D0%BB%D0%B0%D0%BD%D0%BD%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), цитати та інші елементи. Елементи HTML окреслені тегами, написаними з використанням кутових дужок . Теги, такі як і безпосередньо вводять вміст на сторінку. Інші теги, такі як <img /> <input /> <p>оточують і надають інформацію про текст документа і можуть включати інші теги як під-елементи. Браузери не показують теги HTML, але використовують їх для інтерпретації вмісту сторінки.

Розмітка в HTML складається з чотирьох основних компонентів: елементів (та їхніх атрибутів), базових типів даних, символьних мнемонік та декларації типу документа.

Документ HTML складається з трьох частин:

1. Декларація типу документа, на початку документа, в якій визначається тип документа ([DTD](https://uk.wikipedia.org/wiki/Document_Type_Definition)).
2. Шапка документа (знаходиться в межах елемента head), в якій записано загальні технічні відомості або додаткова інформація про документ, яка не відтворюється безпосередньо в [браузері](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80);
3. Тіло документа (може знаходитися в елементі body), в якому міститься основна інформація документа.

Приклад загальної структури HTML-документа:

<! DOCTYPE html>

<html>

<head>

<title>Назва</title>

</head>

<body>

<p> Hello world!</p>

</body>

</html>

* 1. **CSS**

CSS ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Cascading Style Sheets, [укр.](https://uk.wikipedia.org/wiki/%D0%A3%D0%BA%D1%80%D0%B0%D1%97%D0%BD%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Каскадні таблиці стилів) — спеціальна [мова](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), що використовується для опису зовнішнього вигляду сторінок, написаних [мовами розмітки даних](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D1%80%D0%BE%D0%B7%D0%BC%D1%96%D1%82%D0%BA%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

Найчастіше CSS використовують для візуальної презентації сторінок, написаних [HTML](https://uk.wikipedia.org/wiki/HTML) та [XHTML](https://uk.wikipedia.org/wiki/XHTML), але формат CSS може застосовуватися до інших видів [XML](https://uk.wikipedia.org/wiki/XML)-документів.

Специфікації CSS були створені та розвиваються [Консорціумом Всесвітньої мережі](https://uk.wikipedia.org/wiki/W3C).

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3. Профілі — сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або [інтерфейсів](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81). Наприклад, існують профілі CSS для [принтерів](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BD%D1%82%D0%B5%D1%80), мобільних пристроїв тощо.

CSS (каскадна або блочна верстка) прийшла на заміну табличній верстці [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0). Головна перевага блочної верстки — розділення змісту сторінки (даних) та їхньої візуальної презентації.

CSS має спеціальні позначення (селектори, від [англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) selector) для визначення діапазону застосування правила.

* 1. **JavaScript (JS)**

JavaScript (JS) — динамічна, [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [прототипна](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). Реалізація стандарту [ECMAScript](https://uk.wikipedia.org/wiki/ECMAScript). Найчастіше використовується для створення сценаріїв [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0), що надає можливість на стороні [клієнта](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0) (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, [асинхронно](https://uk.wikipedia.org/wiki/AJAX) обмінюватися даними з [сервером](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0), змінювати [структуру](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0) та [зовнішній вигляд](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B4%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD) [веб-сторінки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0).

JavaScript класифікують як [прототипну](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) (підмножина [об'єктно-орієнтованої](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), [скриптову мову](https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування ([імперативну](https://uk.wikipedia.org/wiki/%D0%86%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) та частково [функціональну](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) і деякі відповідні архітектурні властивості, зокрема: [динамічна](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D1%96%D1%87%D0%BD%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F) та слабка [типізація](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97), автоматичне керування пам'яттю, прототипне [наслідування](https://uk.wikipedia.org/wiki/%D0%9D%D0%B0%D1%81%D0%BB%D1%96%D0%B4%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), [функції](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) як [об'єкти першого класу](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82_%D0%BF%D0%B5%D1%80%D1%88%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BB%D0%B0%D1%81%D1%83).

* 1. **PHP**

PHP ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) PHP: Hypertext Preprocessor — PHP: [гіпертекстовий](https://uk.wikipedia.org/wiki/%D0%93%D1%96%D0%BF%D0%B5%D1%80%D1%82%D0%B5%D0%BA%D1%81%D1%82) [препроцесор](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80)), попередня назва: Personal Home Page Tools — [скриптова](https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), була створена для генерації [HTML](https://uk.wikipedia.org/wiki/HTML)-сторінок на стороні [веб-сервера](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). PHP є однією з найпоширеніших мов, що використовуються у сфері веб-розробок (разом із [Java](https://uk.wikipedia.org/wiki/Java), [.NET](https://uk.wikipedia.org/wiki/.NET), [Perl](https://uk.wikipedia.org/wiki/Perl), [Python](https://uk.wikipedia.org/wiki/Python), [Ruby](https://uk.wikipedia.org/wiki/Ruby)). PHP підтримується переважною більшістю [хостинг-провайдерів](https://uk.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3%D0%BE%D0%B2%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D1%96%D1%8F). PHP — проект [відкритого програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F).

PHP інтерпретується веб-сервером у HTML-код, який передається на сторону клієнта. На відміну від скриптової мови [JavaScript](https://uk.wikipedia.org/wiki/JavaScript), користувач не бачить PHP-коду, бо браузер отримує готовий html-код. Це є перевагою з точки зору безпеки, але погіршує інтерактивність сторінок.

* 1. **MySQL**

MySQL — [вільна](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [система керування реляційними базами даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B8%D0%BC%D0%B8_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85). MySQL був розроблений компанією «ТсХ» для підвищення швидкодії обробки великих баз даних. Ця система керування базами даних ([СКБД](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)) з відкритим кодом була створена як альтернатива комерційним системам. MySQL з самого початку була дуже схожою на [mSQL](https://uk.wikipedia.org/w/index.php?title=MSQL&action=edit&redlink=1), проте з часом вона все розширювалася і зараз MySQL — одна з найпоширеніших систем керування базами даних. Вона використовується, в першу чергу, для створення динамічних [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0), оскільки має чудову підтримку з боку різноманітних [мов програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F).

* 1. **phpMyAdmin**

phpMyAdmin — веб-додаток з відкритим кодом на мові PHP із графічним веб-інтерфейсом для адміністрування бази даних MySQL або MariaDB. phpMyAdmin дозволяє через браузер здійснювати адміністрування сервера MySQL, запускати запити SQL, переглядати та редагувати вміст таблиць баз даних. Ця програма користується великою популярністю у веб-розробників, оскільки дозволяє керувати базу даних MySQL без вводу SQL команд через дружній інтерфейс і з будь-якого комп'ютера під'єднаного до інтернету без необхідності встановлення додаткового програмного забезпечення.

* 1. **Локальний сервер Open Server Panel**

Open Server Panel - це портативна серверна платформа і програмне середовище, створена спеціально для веб-розробників з урахуванням їх рекомендацій і побажань.

Програмний комплекс має багатий набір серверного програмного забезпечення, зручний, багатофункціональний продуманий інтерфейс, має потужні можливості з адміністрування та налаштування компонентів. Платформа широко використовується з метою розробки, налагодження і тестування веб-проектів, а так само для надання веб-сервісів в локальних мережах.

* 1. **TCPDF**

TCPDF - це безкоштовний PHP-клас з відкритим вихідним кодом для генерації PDF-документів. TCPDF - єдина заснована на PHP бібліотека, яка включає повну підтримку Unicode UTF-8 і мов з написанням справа наліво.

# **РОЗДІЛ 3. Створення веб-сторінки**

У цьому розділі ми розглянемо основи роботи зі phpMyAdmin. Також створемо сайт для роботи з БД.

* 1. **Створення бази даних**

Для створення та керуванням базою даних будемо використовувати phpMyAdmin. Спочатку перейдемо на головну сторінку phpMyAdmin (рис. 3.1).



Рисунок 3.1 - Головна сторінка phpMyAdmin

Далі вибираємо рядок "Создать БД" зліва (рис. 3.2).

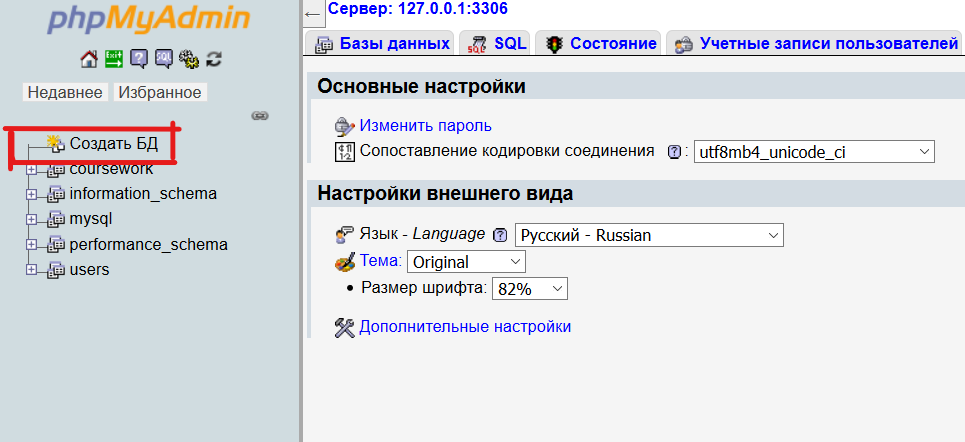


Рисунок 3.2 - Рядок для створення БД

На сторінці, що з'явилася можна побачити БД, які були створені раніше. Також є поле для введення імені нової БД.

Вводимо ім'я новий БД і натискаємо на кнопку “Создать”. Для прикладу створення БД введемо ім'я “test” (рис 3.3).

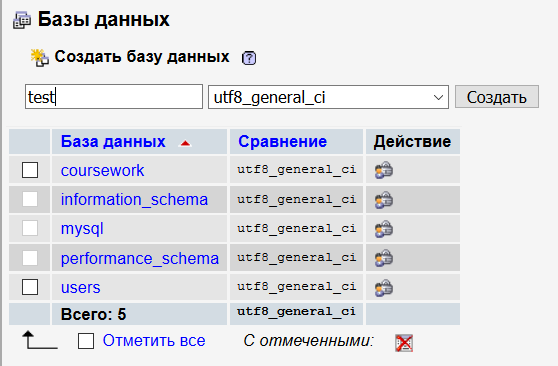


Рисунок 3.3 – Створення БД

Після натискання на кнопку нам пропонують створити нову таблицю.

* + 1. **Створення таблиці**

Після створення БД нам пропонують створити нову таблицю (рис. 3.4).

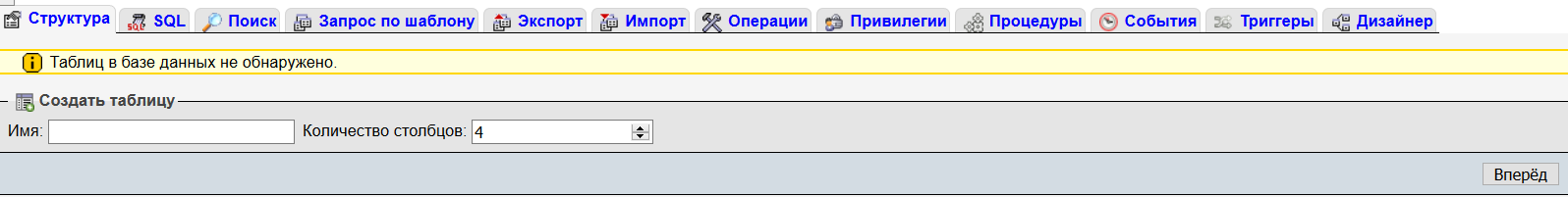


Рисунок 3.4 – Створення таблиці

В рамках прикладу введемо ім'я таблиці "A" і натиснемо “Вперед”. Тепер нам запропоновані набір осередків для установки параметрів стовпців.

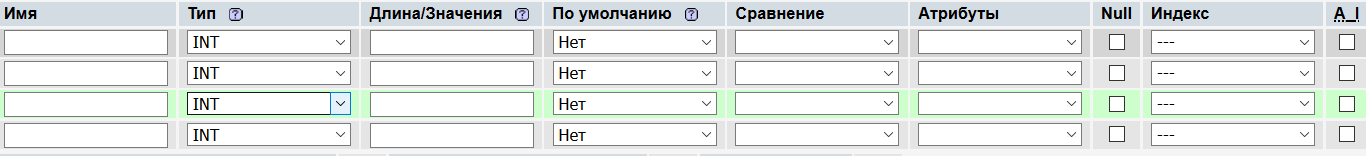


Рисунок 3.5 –Вікно створення таблиці

Розглянемо основні параметри для осередків рядка.  
“Имя” - задає ім'я стовпчика  
“Тип” - вказує типу значення, яке буде зберігається в осередку таблиці. Основними типами є:  
INT - осередок буде зберігати числа в діапазоні від -2 147 483 648 до 2 147 483 647  
VARCHAR - рядок змінної довжини від 0 до 65535 символів. Для цього типу потрібно вказати параметр " Далина/Значення ".  
TEXT - рядок з максимальною довжиною 65535  
DATE - являє собою дату, діапазон від 1000-01-01 до 9999-12-31  
“Длина/Значення” - вказує скільки пам'яті виділити для зберігання даних.  
“Індекс” - використовується для ідентифікування запису у таблиці. Дозволяє встановити первинний ключ для даної таблиці.  
“А\_І” (auto\_increment) - генерація унікального ідентифікатора для нових рядків.

Встановивши такі значення як на рисунку 3.6 для полів, рядок ID буде первинним ключем для даної таблиці. Первинний ключ буде типу INT та, для запобігання помилок, включимо “A\_I”. Рядок “L1” буде зовнішнім ключем.

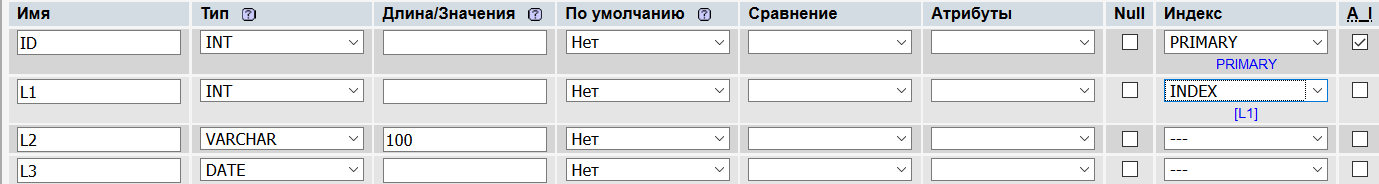


Рисунок 3.6 – Таблиця А

Зовнішній ключ - це стовпець або комбінація стовпців, значення яких відповідають первинного ключа в іншій таблиці. Зв'язок між двома таблицями задається через відповідність первинного ключа в одній з таблиць зовнішньому ключу в другій. Зовнішній ключ має мати такий самий тип, як і первинний ключ.

Натискаємо “Сохранить”. Тепер в нашій БД є одна таблиця.

Щоб подивитися на всі існуючі таблиці, нам потрібно натиснути на ім'я нашої БД. У вкладці “Структура” будуть представлені всі таблиці для даної БД (рис. 3.7).

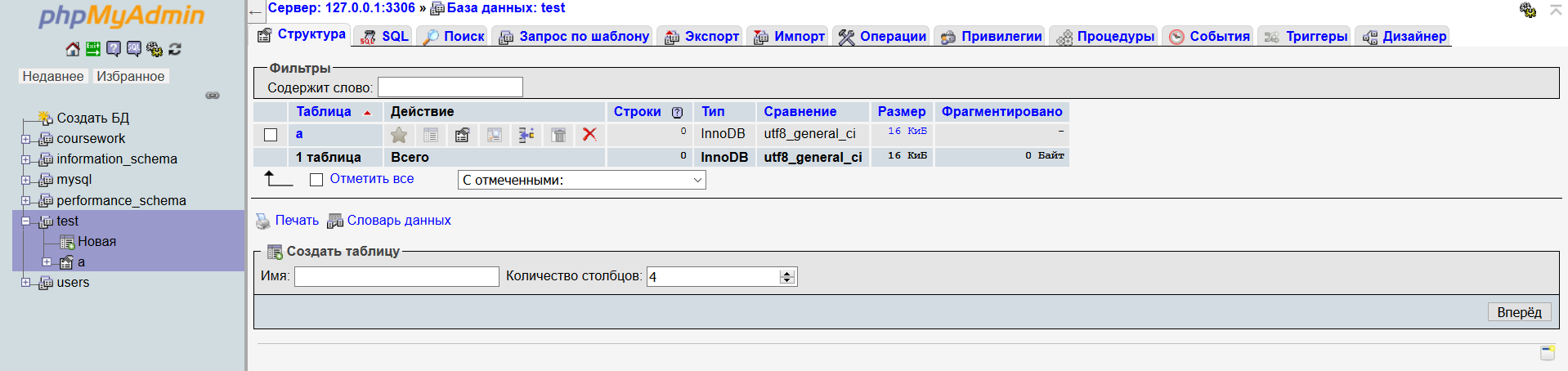


Рисунок 3.7 – Сторінка з всіма таблицями

* + 1. **Створення зв'язків між таблицями**

Для створення зв'язків нам потрібно мати хоча б дві таблиці, але спочатку розглянемо які існують види зв'язків таблиць.

Види зв'язків:

1. Один до багатьох - при цьому виді зв'язку для рядка однієї таблиці може відповідати безліч рядків іншої таблиці (рис. 3.8).
2. Багато до багатьох - при цьому виді зв'язку для кожного рядка однієї таблиці може відповідати безліч рядків іншої таблиці і навпаки (рис. 3.9).
3. Один до одного - при цьому виді зв'язку для кожного рядка однієї таблиці може відповідати один рядок іншої таблиці і навпаки (рис. 3.10).

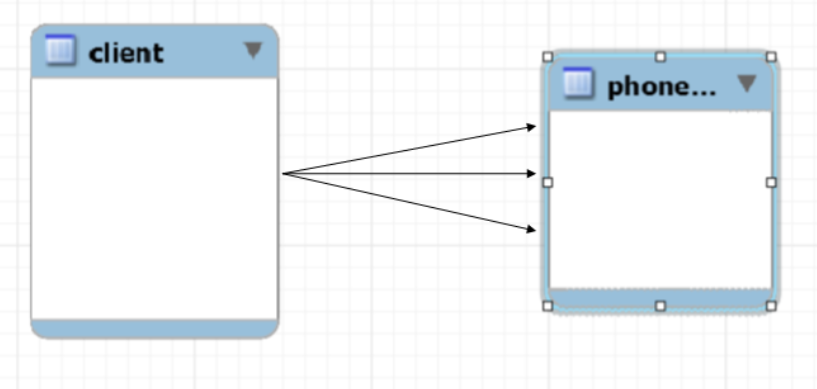


Рисунок 3.8 – Зв’язок один до багатьох

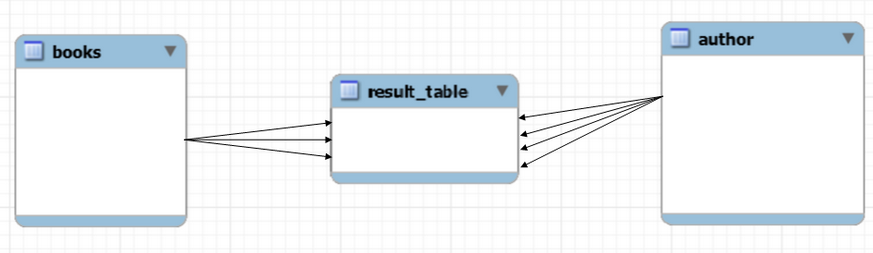


Рисунок 3.9 - Зв’язок багато до багатьох

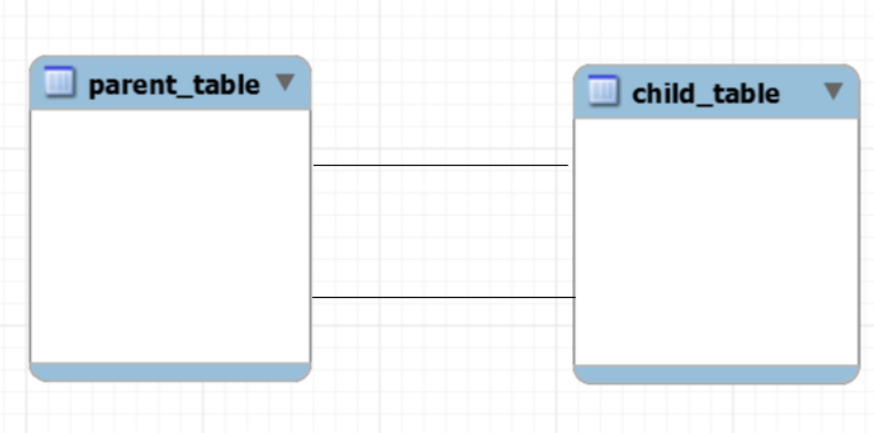


Рисунок 3.10 - Зв’язок один до одного

Створимо ще одну таблицю, щоб розглянути приклад створення простого зв'язку між таблицями використовуючи phpMyAdmin. Для цього натиснемо на ім'я нашої БД зліва. Далі у вкладці “Структура” під списком існуючих таблиць запропоновано ввести ім'я для нової таблиці і кількість стовпців (рис. 3.11). Введемо ім'я В, а кількість стовпців вкажемо 2. Натискаємо “Вперед”.

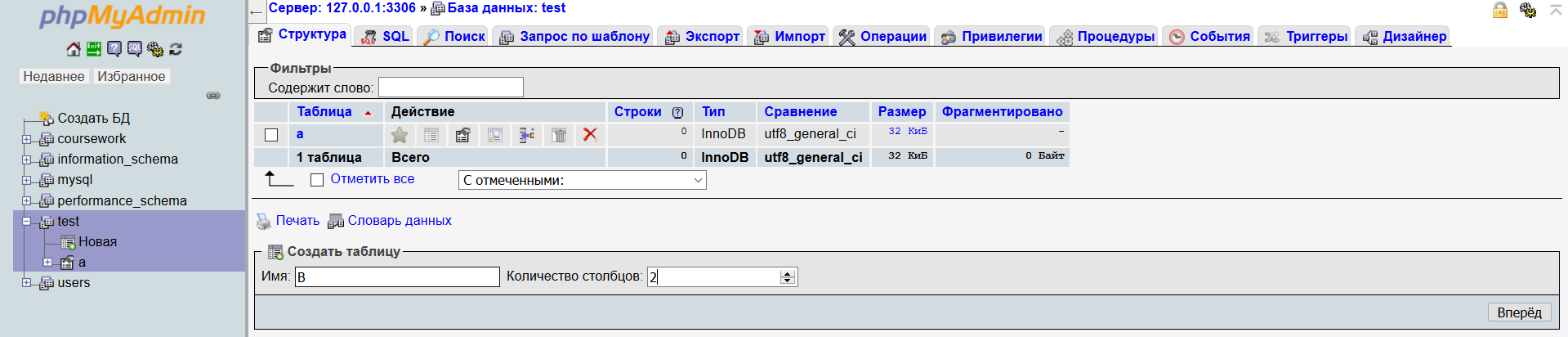


Рисунок 3.11 – Створення таблиці B

Задамо імена В\_ID і В\_NAME для стовпів. Стовпець В\_ID буде первинним для даної таблиці (рис. 3.12).

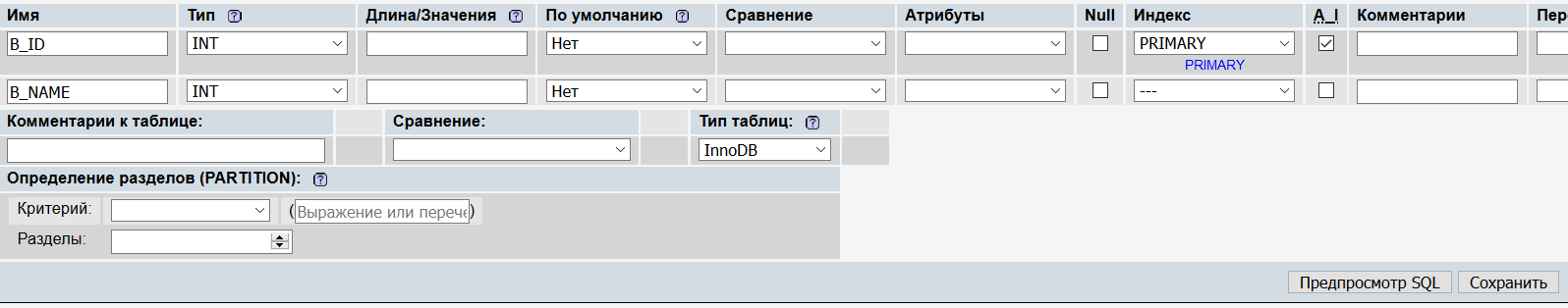


Рисунок 3.12 – Таблиця В

Далі натискаємо “Сохранить”.

Тепер вибираємо таблицю “А”. Переходимо у вкладку “Структура”, відразу під ім'я вкладки є кнопка “Связи”. Натискаємо на неї. Після цього ми опинимося на сторінці для створення зв'язків (рис. 3.13).

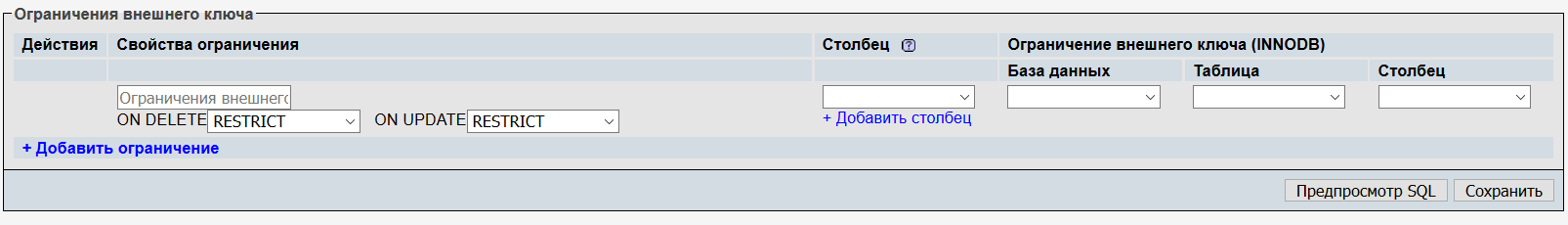


Рисунок 3.13 - Створення зв’язків між таблицями

Вирази ON DELETE і ON UPDATE зовнішніх ключів використовуються для вказівки дій, які будуть виконуватися при видаленні рядків батьківської таблиці (ON DELETE) або зміні батьківського ключа (ON UPDATE). Установим значение “RESTRICT” для “ON DELETE” и для “ON UPDATE”. Значення “RESTRICT” вказує на те, що забороняється видаляти або змінювати батьківський ключ, коли існує один або кілька посилань на нього дочірніх ключів. Далі вибираємо наш зовнішній ключ, який являє собою стовпець "L 1", таблиці "A". І пов'язуємо його з первинним ключем "В\_ID" таблиці "B" (рис. 3.14).

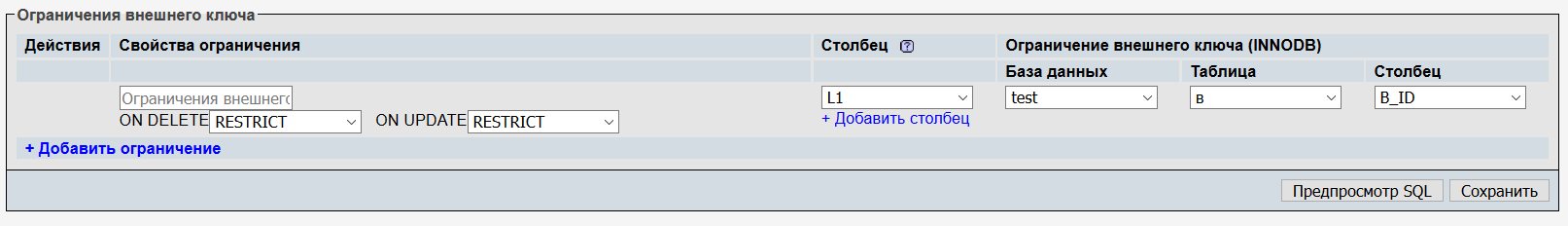


Рисунок 3.14 – Створення зв’язка поміж таблицями А та В

Після цього натиснувши на ім'я нашої БД зліва, у вкладці "Дизайнер" можна буде подивитися графічне представлення зв'язків між таблицями в нашій БД (рис. 3.15).

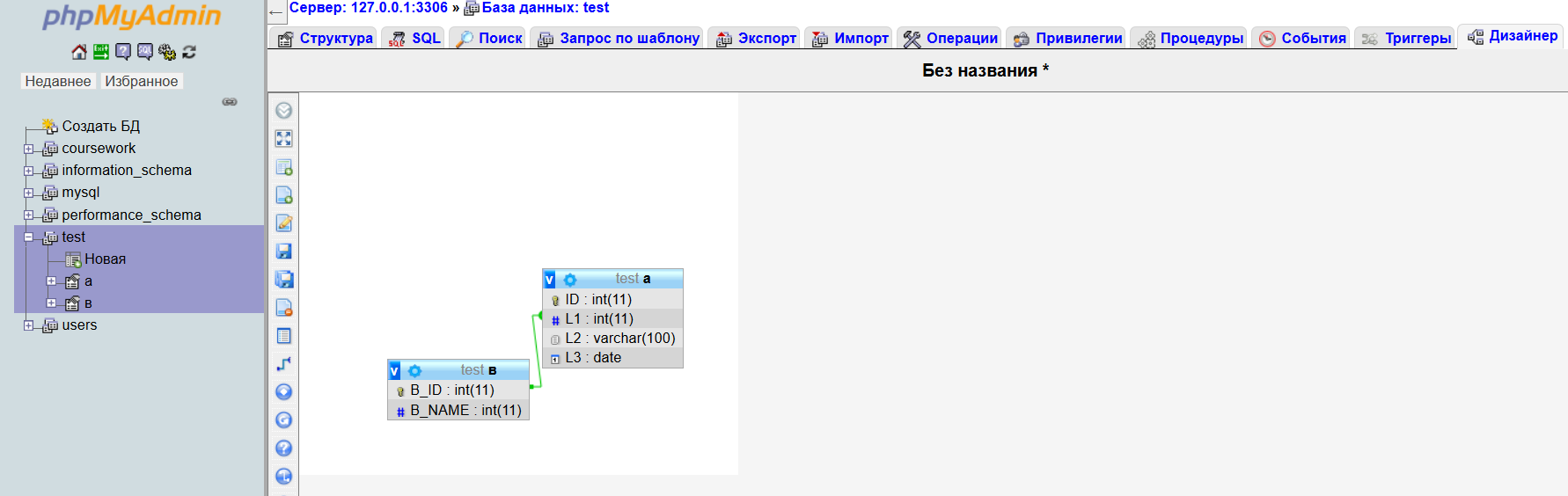


Рисунок 3.15 – Графічне зображення зв’язків поміж таблицями

* + 1. **Створення БД для роботи зі сайтом**

Використовуючи методи представлені в розділах 3.1.1, 3.1.2, 3.1.3, ми створимо нову БД з ім'ям "coursework". Після додамо таблиці “cities” (рис. 3.16), “countries” (рис. 3.17), “tours” (рис. 3.18), “clients” (рис. 3.19), “vehicles” (рис. 3.20). Після встановимо зв'язок між таблицями (рис. 3.21).

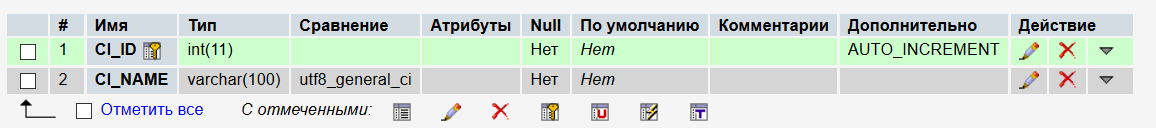


Рисунок 3.16 – Таблиця cities

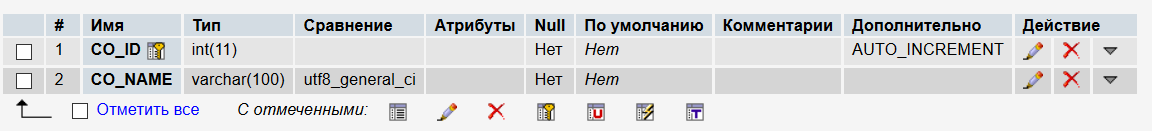


Рисунок 3.17 – Таблиця counties

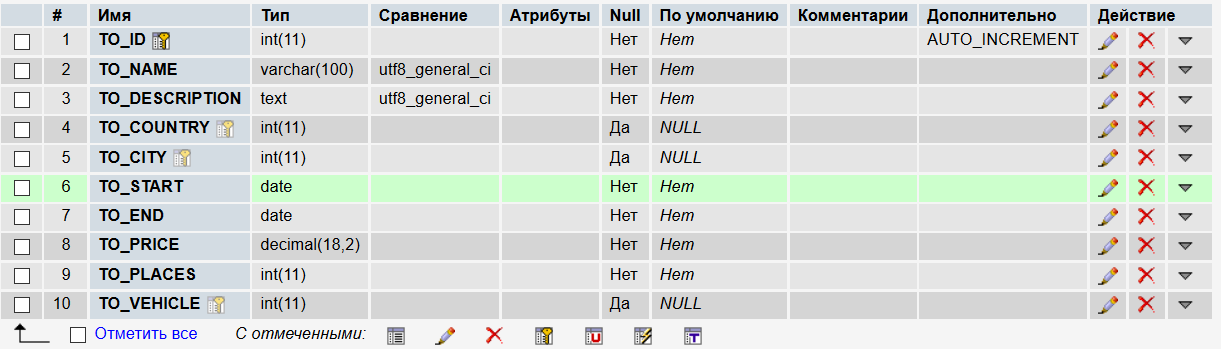


Рисунок 3.18 – Таблиця tours

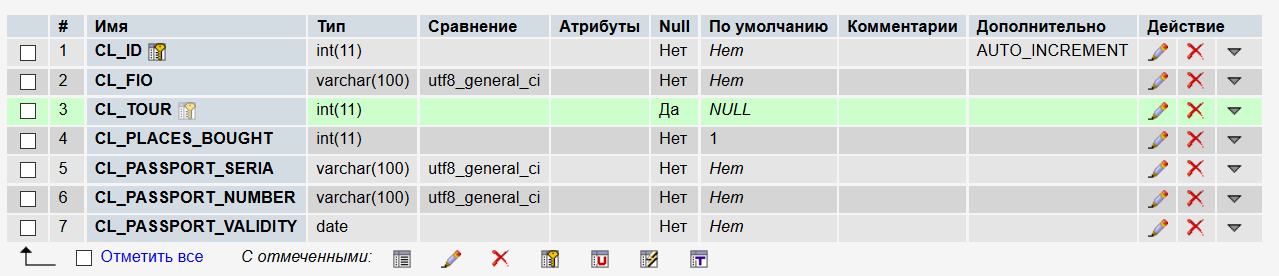


Рисунок 3.19 – Таблиця clients

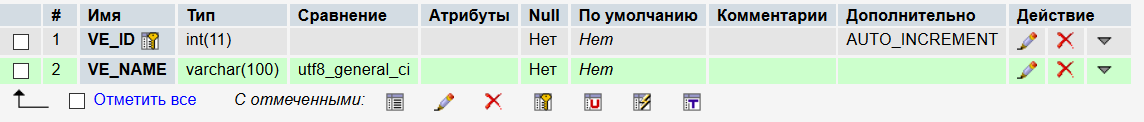


Рисунок 3.20 – Таблиця vehicles

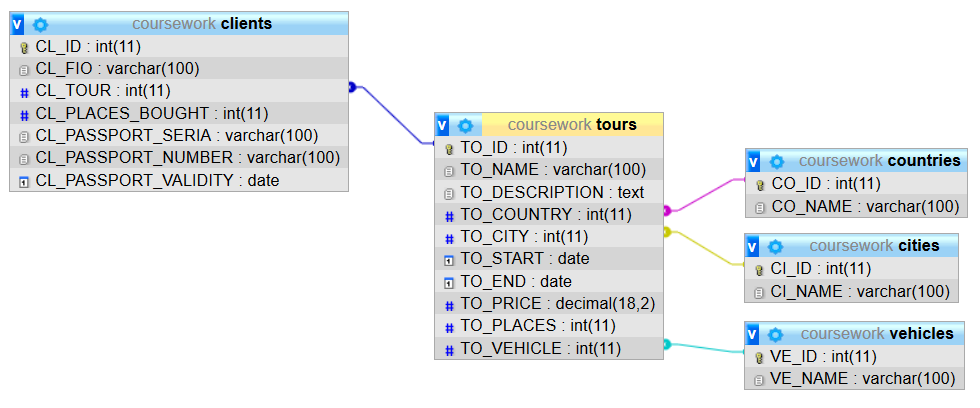


Рисунок 3.21 – Графічне зображення зв’язків між таблицями

* 1. **Створення головної сторінки для звичайного користувача**

Сторінка для незареєстрованого користувача буде відрізнятися від сторінки користувача admin. На сторінці незареєстрованого користувача йому буде пропозиція вибрати тур, який на даний момент є в нашій БД. Для цього нам потрібно буде підключитися до нашої БД і виконати MySQL запит, щоб отримати доступні тури з таблиці “tours”. Все це можна успішно зробити, використовуючи PHP. Отже, щоб підключитися до БД використовуючи PHP можна використовувати функцію mysqi\_connect. Ця функція приймає 4 аргументу:

* Адреса, де розташований MySQL сервер.
* Користувач, який буде використовуватися для виконання запитів.
* Пароль для даного користувача.
* Останній параметр це ім'я БД, яку будемо використовувати.

Так як цю функцію ми будемо використовувати досить часто, створимо окрему функцію, яка буде повертати підключення до БД (рис. 3.22).

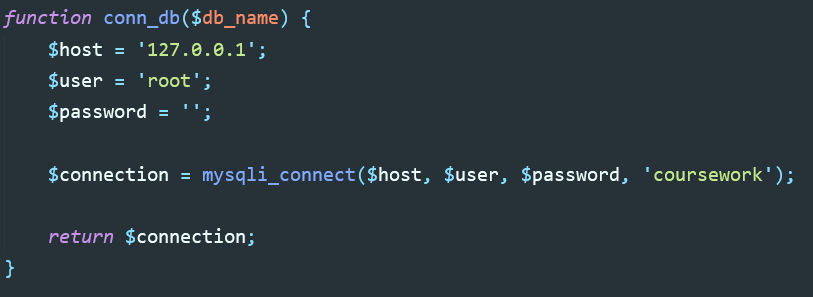


Рисунок 3.22 – Функція conn\_db

Далі створимо функцію get\_goods (рис. 3.23). Вона буде виконуватися запит до БД, яка буде передана в якості аргументу і повертати асоціативний масив рядків результуючої таблиці.

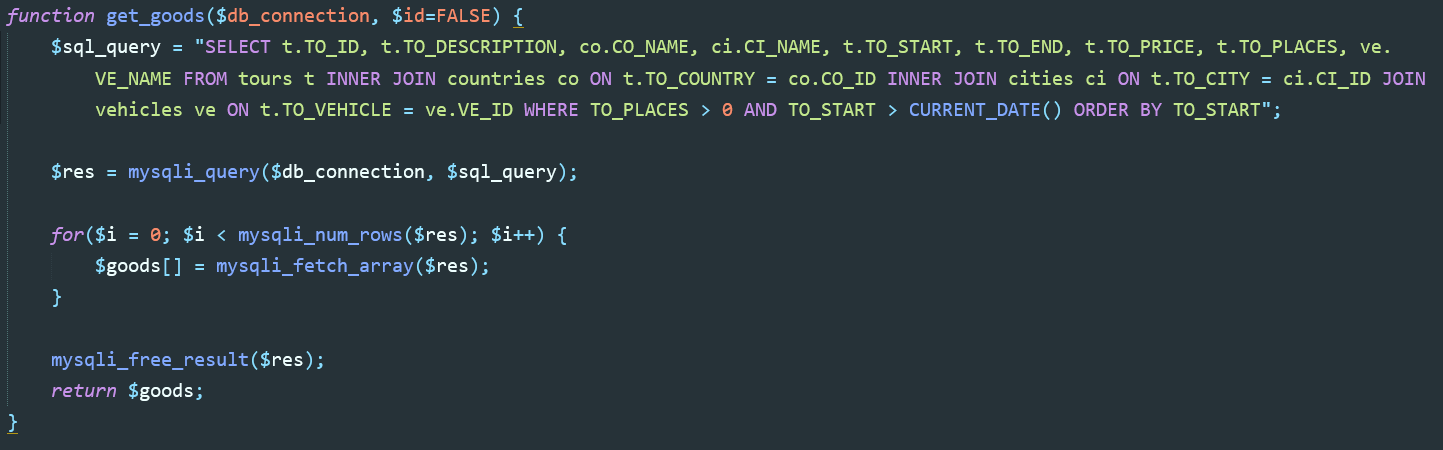


Рисунок 3.23 – Функція get\_goods

Функція є mysqli\_query функцією PHP, яка виконує SQL запит до БД. Результатом виконання даної функції є об'єкт, який представляє собою результуючу таблицю.

Розглянь запит детальніше: Оператор запиту SELECT повертає значення зазначених стовпців з таблиць указвних після оператора FROM або ж інших операторів, наприклад JOIN, як в нашому запиті. Оператор JOIN виконує об'єднання двох таблиць, по стовпцю зазначеного після ON. Оператор WHERE вказує умови, за яким оператор SELECT вибирає значення зазначених йому стовпців.

Результат функції get\_goods буде використаний для створення і відображення існуючих турів на сайті. Відображення на сайті буде мати наступний вигляд (рис. 3.24).



Рисунок 3.24 – Зовнішній вигляд турів на сайті для звичайного користувача

* + 1. **Сторінка оплати**

Після того, як користувач вибере тур йому потрібно буде його оплатити. Для цього створимо окрему сторінку для оплати. Вона матиме наступний вид (рис. 3.25).

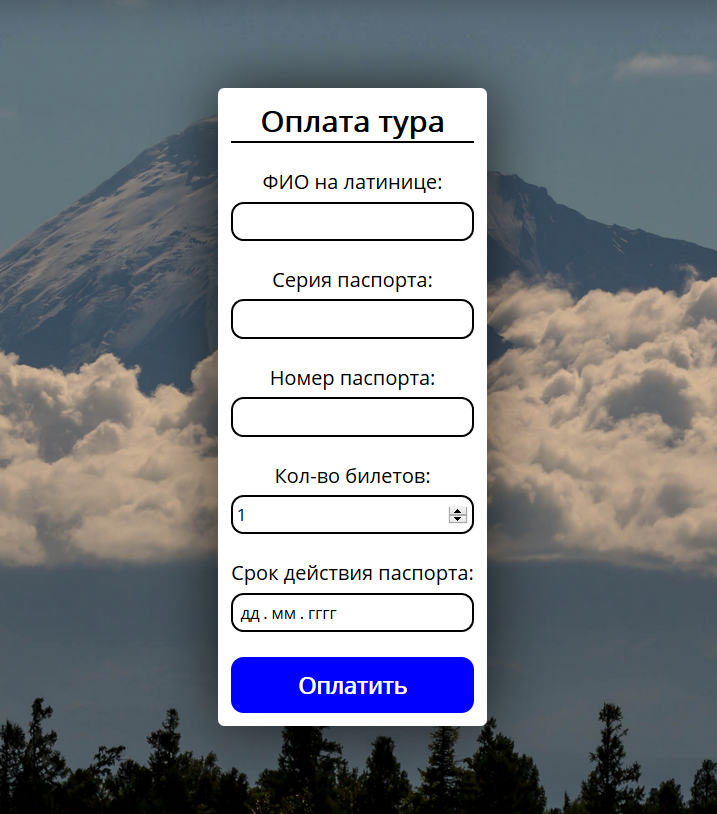


Рисунок 3.25 – Сторінка оплати

Для роботи з даною сторінкою створимо дві фунції insert і update. Перша буде додавати нового клієнта в таблицю "clients", а друга буде оновлювати значення кількості доступних квитків в таблиці "tours".

function insert($table, $data) {

$connection = conn\_db('coursework') or die("Conn error.");

$query = 'SELECT \* FROM ' . $table;

$to\_count\_columns = mysqli\_query($connection, $query) or die("Columns error.");

$columns\_num = mysqli\_num\_fields($to\_count\_columns)-1;

if($columns\_num != count($data)) {

die("Not equal colomns number and data.");

}

$formar\_str = "INSERT INTO " . $table . " VALUES (NULL, ";

for ($i=0; $i < $columns\_num; $i++) {

$formar\_str = $formar\_str."?";

if($i != $columns\_num-1) {

$formar\_str = $formar\_str . ", ";

}

}

$formar\_str = $formar\_str .")";

$types = '';

for ($i=0; $i < $columns\_num; $i++) {

$var = $data[$i];

if(is\_string($var)) {

$types = $types . 's';

} else if(is\_integer($var)) {

$types = $types . 'i';

} else {

die ("Unknown type.");

}

}

$stmt = mysqli\_prepare($connection, $formar\_str);

mysqli\_stmt\_bind\_param($stmt, $types, ...$data);

mysqli\_stmt\_execute($stmt);

mysqli\_close($connection);

}

Функція insert приймаємо два аргументи. Перший аргумент це назва таблиці, в яку будуть вноситися дані, а другим аргументом будуть дані, які будуть вноситися в таблицю. Функція працює наступним чином: обчислює кількість стовпців в таблиці, а потім стравнівает з кількість переданих даних. Якщо воно нерівно то видає помилку, інакше створює форматіруюзую рядок запиту. Після цього підставляє значення з другого аргументу і виконує SQL запит. Розглянемо SQL запит для вставки даних в таблицю. Після інструкції INSERT INTO вказується ім'я таблиці, після цього вказується ім'я таблиці куди буде вставлятися значення. Після VALUES в дужках зазначаються бажані значення для вставки.

Тепер розглянемо другу функцію update для поновлення даних.



Рисунок 3.26 – Функція update

Ця функція приймає три аргументи. Перший це таблиця, де буде встановлюватися значення. Другий це значення, які потрібно встановити. Третій аргумент умови, для того, щоб змінити значення певного рядки. Тож розглянемо сам запит. Він приймає наступний вигляд UPDATE ім’я таблиці SET ім’я стовпчика = нове значення WHERE умови для вибору певного стовпчика.

Зараз ми подивимося, що відбувається з базою даних після покупки клієнтом квитка. Для цього збережемо початковий стан таблиці “clients” (рис. 3.27).

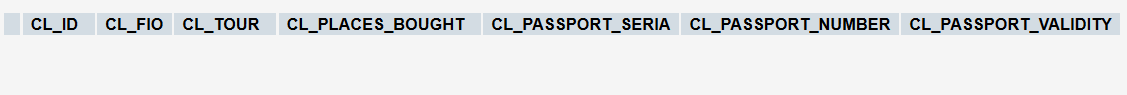


Рисунок 3.27 – Початковий стан таблиці clients

Та таблиці “tours” (рис. 3.28).



Рисунок 3.28 - Початковий стан таблиці tours

Купувати ми будемо квиток з TO\_ID 48, у якого на даний момент 50 вільних квитків. Дані про клієнта, який купує квиток представлені на малюнку 29.

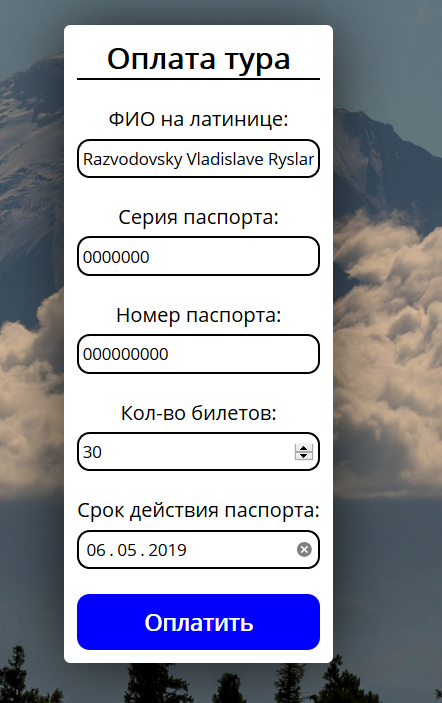


Рисунок 3.29 - Приклад покупки квитка

Давайте тепер поглянемо на наші таблиці після оплати. Таблиця “clients” (рис. 3.30).

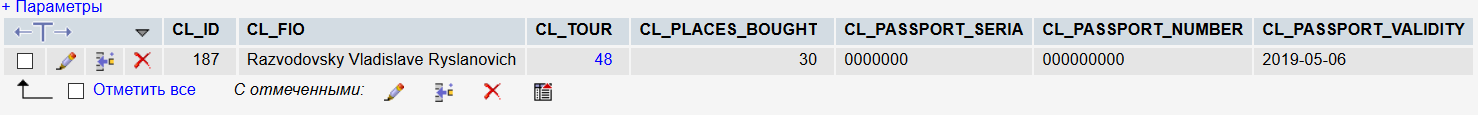


Рисунок 3.30 – Стан таблиці clients після покупки квитка

Таблиця “tours” (рис. 3.31).



Рисунок 3.31 – Стан таблиці tours після покупки квитка

Як можна помітити в таблиці “clients” з'явився новий запис з даними, які ми ввели при покупці, а в таблиці “tours” оновилося значення осередка “TO\_PLACES” тура з ім’ям “FUTURE”.

* + 1. **Сторінка входу**

На сайті є можливість авторизації (рис. 3.32). Увійти під обліковим записом адмініістратора можно, використовуючи логін і пароль “admin”, для управління бази даних з меншими можливостями, ніж через phpMyAdmin. Можливості у адміністратора будуть наступні: створення, зміна, видалення даних з таблиць. Так само адміністратор може отримати звіти(рис. 3.33).

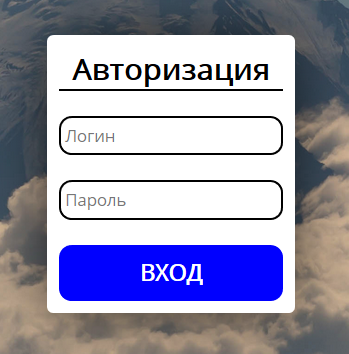
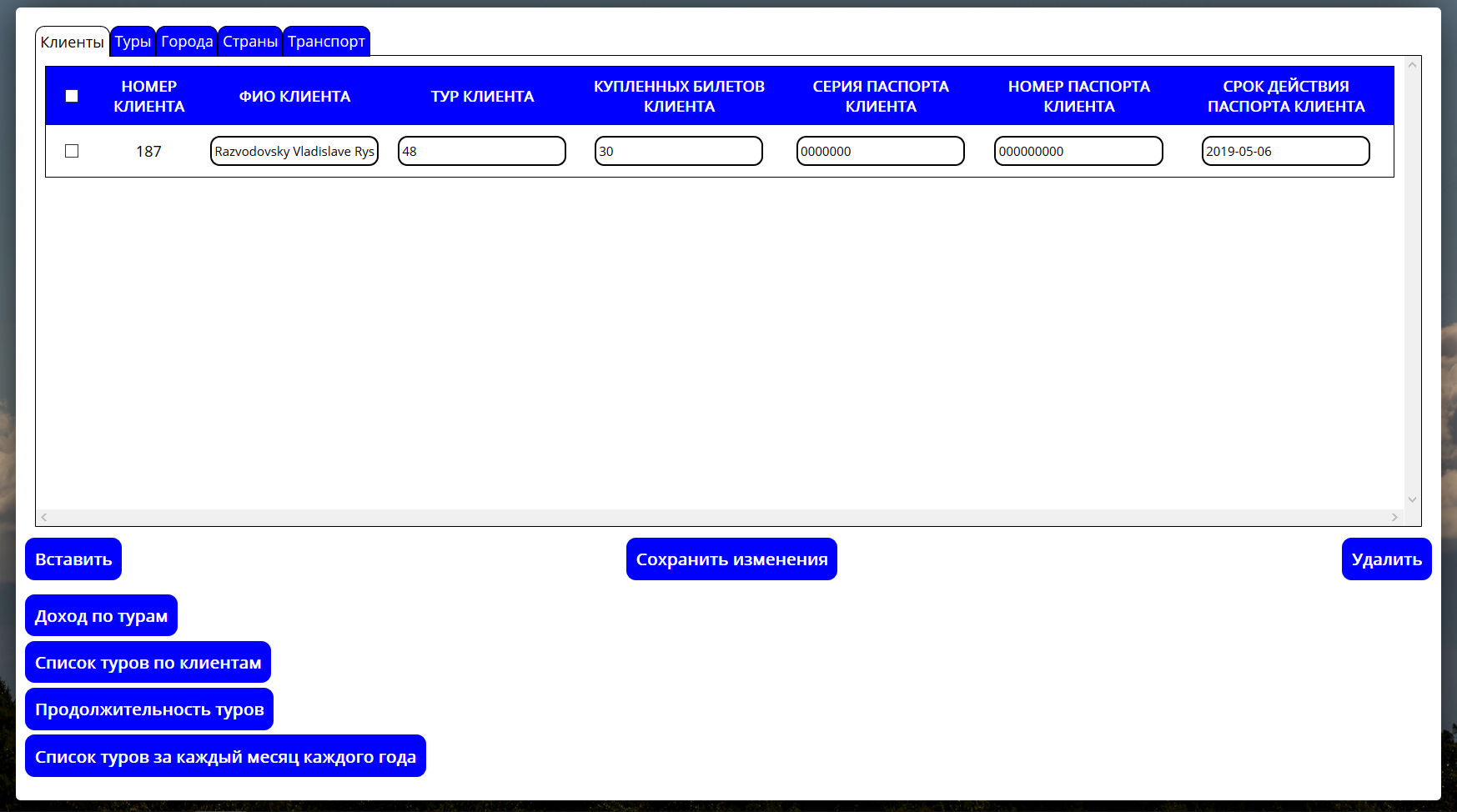


Рисунок 3.32 – Сторінка авторизації



(рис. 3.33)

* 1. **Сторінка адміністратора**

Сторінку адміністратора можна умовно поділити на дві частини. Перша частина відповідає за взаємодію з БД і таблицями. Так само вона складається з кнопок “Вставить”, “Сохранить”, “Удалить” і області для відображення записів таблиць.

Друга частина це кнопки для отримання звітів.

* + 1. **Редагування таблиць**

Зміна таблиць буде відбуватися з використання трьох кнопок “Вставить”, “Сохранить”, “Удалить”, які будуть описано нижче докладніше. Перехід між таблицями буде відбуватися натисканням на ім'я потрібної вкладки. Ім'я вкладок розташовані вгорі змісту сторінки (рис. 30). При натискання буде відбувається запит до потрібної таблиці, що означає втрату незбережених даних.

* + - 1. **Вставка**

Розглянємо кнопку “Вставить” для роботи з таблицями. Отже, після натискання на кнопку, JavaScript обробляє натискання (рис. 3.34) . Все що виконує даний скрипт - використовує написану нами функцію set\_content (рис. 3.35), що асинхронні змінює частина документ HTML відповіддю від сервера (рис. 3.36).



Рисунок 3.34 - Код, що відповідає за обробку натискання

Функція get\_cells повертає масив значень для кожного рядка відкритої в даний момент таблиці.



Рисунок 3.35 - Функція, що змінює частина HTML документа

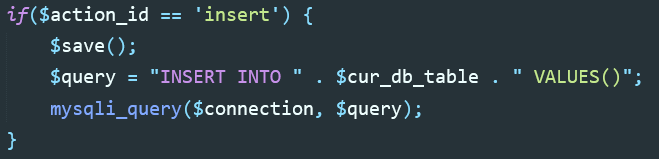


Рисунок 3.36 - Частина коду, що відповідає за формування відповіді сервера

Функція save, яку ми використовували, відповідає за збереження даних перед додавання нових записів у таблицю.

Використання VALUES без вказівки вмісту для кожного стовпчика створює запис зі значеннями за замовчуванням.

* + - 1. **Збереження**

Збереження працює майже так само як вставка, але на стороні сервера виконуються інші дії. А саме виконання інструкції UPDATE для кожного осередку в поточній таблиці. Фрагмент коду, який це виконує представлений нижче в зображенні.



Рисунок 3.37 – Обновлення для осередків

data це масив значень осередків з поточний таблиця, яку повертає get\_cells. У циклі для кожної рядку виконується інсткруція UPDATE cur\_db\_table SET columns = set\_data WHERE id\_column = id, де cur\_db\_table – поточна таблиця, columns – масив стовців для поточної таблиці, set\_data – значення певного осередка, id\_column – стовпец, що зберігає унікальний ідентифікатор запису, id - унікальний ідентифікатор запису для поточного рядка.

* + - 1. **Видалення**

Для видалення рядка з таблиці потрібно натиснути на квадрат в найпершому лівому стовпчику навпроти необхідного рядка. Оброблювач натискання такий же як і у інших. Відмінність в тому, що одним з аргументів функції set\_content виступає результат роботи функції get\_checked. Ця функція повертає масив рядків, які повинні бути видалені.

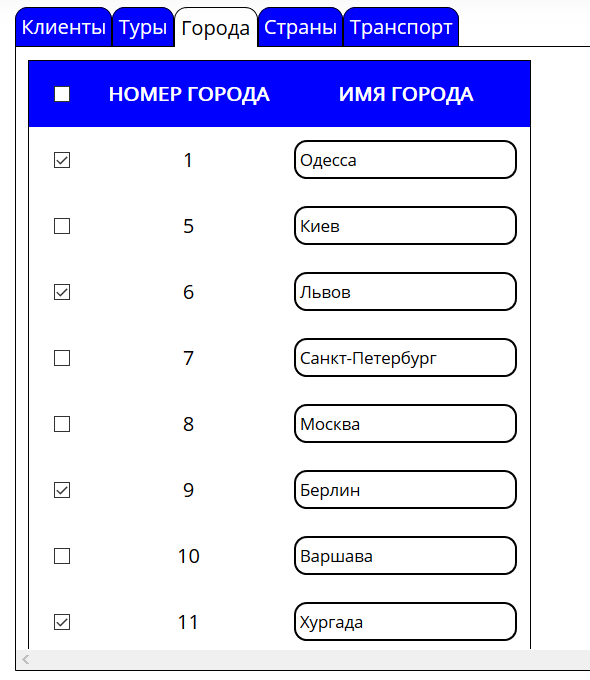


Рисунок 3.38 - Приклад видалення

Розглянемо що виконує сервер при натисканні кнопки видалення (рис. 3.39).

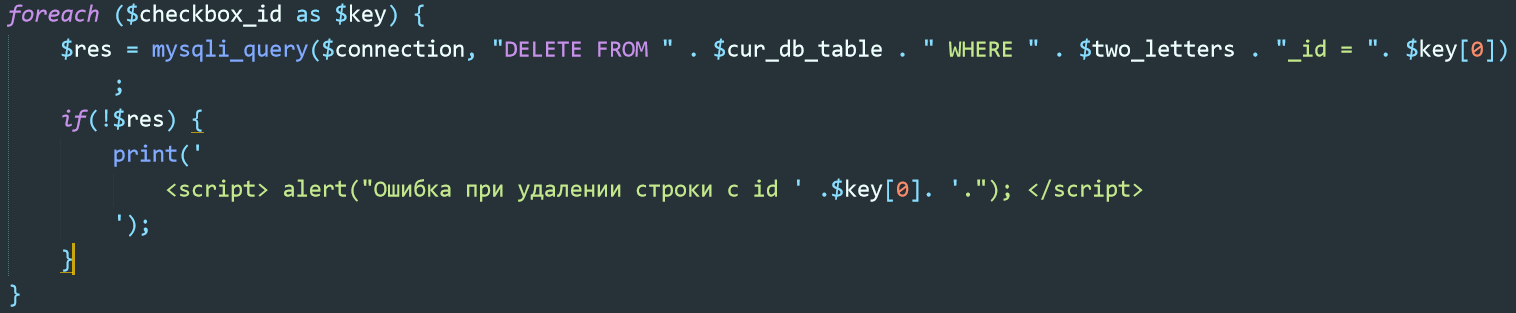


Рисунок 3.39 – Код, для видалення рядків з таблиці

При натискання на клавішу, сервер виконує SQL запит на видалення рядків, які повернула функція get\_checked. Виконується це все інструкцією DELETE. Інструкція має наступний вигляд: DELET FROM ім'я таблиці WHERE умови, яким повинна відповідати рядок для видалення. У нашому випадки cur\_db\_table - ім'я поточної таблиці, two\_letters - це перші дві букви назви таблиці, які потім з'єднуються з рядком “\_id”, key - це уявлення checkbox\_id, яке зберігає в собі id рядків для видалення.

* + 1. **Звіти**

Всі звіти будуть в форматі .pdf. Самі дані будуть представлені у вигляді таблиці. Для формування файлу звіту буде використовувати бібліотека TCPDF. Є два варіанти для формування PDF документа по URL або ж по HTML документу. Ми виберемо варіант по HTML документу, тому що ми використовуємо локальний сервер, через це ми не зможемо отримати доступ до потрібної нам сторінки через URL.

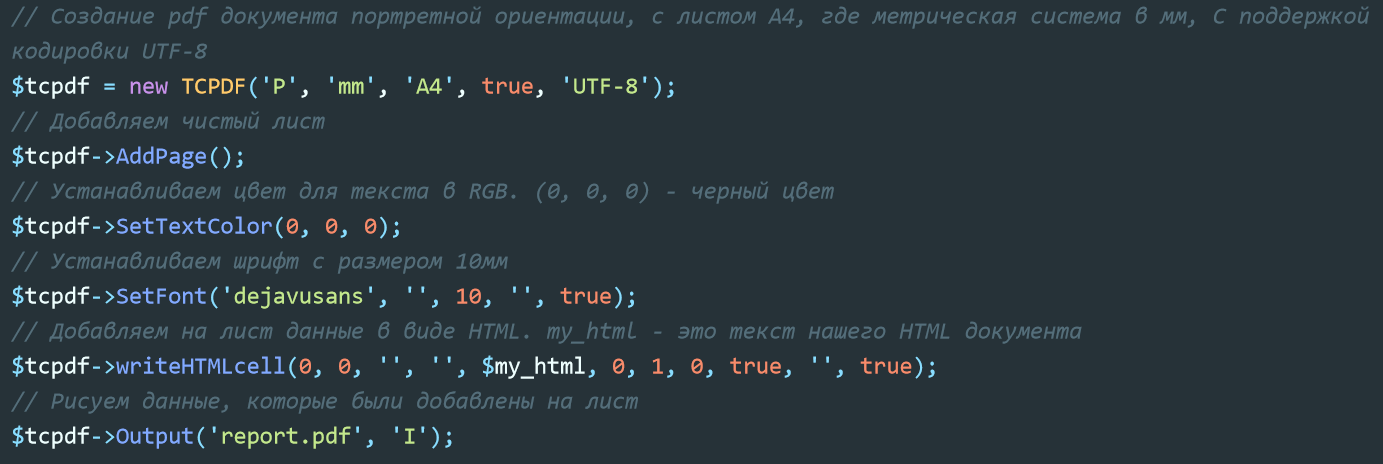


Рисунок 3.40 - Приклад створення .pdf документа за допомогою TCPDF

Так само створимо функцію draw, яка буде виводить таблицю у вигляді HTML документа. Ця функція буде приймати один аргумент - SQL запит. Потім вона буде виконувати цей запит і виводити назви стовпців, а так само дані кожного рядка результуючий таблиці (рис. 3.41).

Умовно, для надання кожного зі звітів, їх можна поділити на дві частини.



Рисунок 3.41 Приклад таблиці у .pdf форматі

* + - 1. **Дохід фірми від кожного туру і за всіма турам в цілому**

Загальний вигляд для даного звіту повинен мати вигляд як на рисунку 3.42 нижче.



Рисунок 3.42 - Загальний вигляд звіту

Для того, щоб отримати бажаний результат для початку використовуємо нашу функцію draw передавши в якості аргументу наступний запит: SELECT TO\_NAME AS 'Название тура', co\_name AS 'Страна', ci\_name AS 'Город', to\_start AS 'Начало тура', CL\_PLACES\_BOUGHT \* tours.TO\_PRICE AS 'Доход' FROM clients INNER JOIN tours ON CL\_TOUR = tours.TO\_ID INNER JOIN countries ON TO\_COUNTRY = countries.CO\_ID INNER JOIN cities ON TO\_CITY = cities.CI\_ID.

Цей запит вибере необхідні для нас стовпчики і задасть їм потрібні імена, які ми вказали після AS.

В результаті ми отримаємо тільки таблицю. Крім таблиці, нам необхідно вивести загальний дохід за всіма турам. Це і буде другою частиною нашого звіту. Щоб зробити це виконаємо ще один запит, який має наступний вигляд: SELECT SUM(t.TO\_PRICE\*c.CL\_PLACES\_BOUGHT) FROM clients c INNER JOIN tours t ON c.CL\_TOUR = t.TO\_ID.

Цей запит поверне лише один рядок, який буде містити суму добутка ціни квитка на кількість куплених квитків.

Далі нам залишається вивести це значення.

В результаті ми отримаємо те, що і потрібно було отримати.

* + - 1. **Список клієнтів по одному туру і всім турам**

Даний звіт повинен мати таблицю для кожного туру зі списком даних клієнта, які купили цей тур, це буду перша частина звіту. Так само повинна бути таблиця зі списком даних всіх клієнтів, які купили хоча б один тур, а це буду друга частина звіту.

Перше, що ми виконаємо буде наступний запит: SELECT t.TO\_NAME, countries.CO\_NAME, cities.CI\_NAME, t.TO\_START, t.TO\_ID FROM clients c INNER JOIN tours t ON c.CL\_TOUR = t.TO\_ID INNER JOIN countries ON t.TO\_COUNTRY = countries.CO\_ID INNER JOIN cities ON t.TO\_CITY = cities.CI\_ID. Він отримає ім'я туру, країну туру, місто туру, дату початку туру і те, що нам знадобиться надалі, унікальний ідентифікатор туру.

Далі в циклі будемо витягувати рядок результуючої таблиці і кожну ітерацію виводити дані про тур (рис. 3.43).

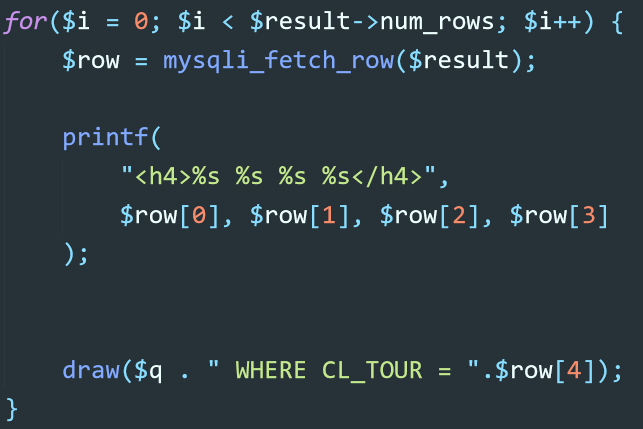


Рисунок 3.43 - Виводим дані про тур

А потім скористаємося іншим запитом, а саме: SELECT CL\_FIO AS 'ФИО', CL\_PASSPORT\_SERIA AS 'Серия паспорта', CL\_PASSPORT\_NUMBER AS 'Номер паспорта', CL\_PASSPORT\_VALIDITY AS 'Срок действия' FROM clients і додамо до нього інструкцію WHERE, яка буде шукати клієнтів з потрібним id туром. Row – це масив значень для поточного рядка результуючої таблиці, де row[0] – ім’я туру, row[1] – країна туру, row[2] – місто туру, row[3] – дата початку туру, row[4] – id туру. Цей запит буде передаватися як аргумент нашої функції draw.

Таким чином ми виведемо ім'я туру, країну туру, місто туру і початку туру, а поле цього буде таблиця зі списком даних клієнтів для потрібно туру.

Щоб вивести другу частину звіту, досить просто передати в функцію draw другий запит з першої частини без додавання інструкції WHERE. Тобто просто передати як аргумент наступний запит: SELECT CL\_FIO AS 'ФИО', CL\_PASSPORT\_SERIA AS 'Серия паспорта', CL\_PASSPORT\_NUMBER AS 'Номер паспорта', CL\_PASSPORT\_VALIDITY AS 'Срок действия' FROM clients. Таким чином отримаємо потрібну нам таблицю з даними про всіх клієнтів, які придбали тур.

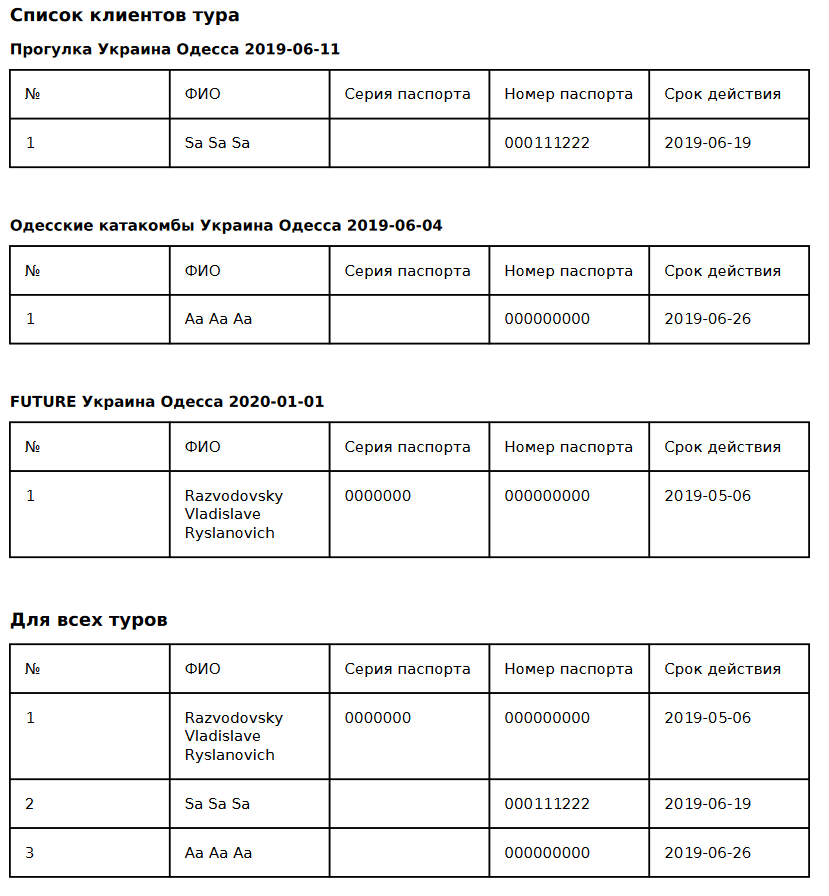


Рисунок 3.44 - Приклад звіту

* + - 1. **Тривалість кожного туру. Середня тривалість туру**

Для цього звіту перша частина – це таблиця з ім’я туру та його тривалістю. Друга частина – це середня тривалість турів.

Для того, щоб обчислити тривалість туру, потрібно буде відняти з дати кінця туру дату початку туру. Для цього потрібно буде використовувати функцію MySQL DATEDIFF. Ця функція приймає два аргументи, які повинні представляти собою дати. Результатом цієї функції буде різниця цих дат в днях.

Для виконання другої частини, а саме обчислення середньої тривалості туру, будемо використовувати ще одну функцію MySQL AVG, яка обчислює середнє значення. Вона приймає стовпчик таблиці з чисельним значенням. Використання цієї функції в комбінації з DATEDIFF дозволить дізнатися середню тривалість туру.

Так як перша частина повинна являти собою таблицю, то використовуємо функцію draw, а аргументом буде довльно простий запит: SELECT TO\_NAME AS 'Имя тура', DATEDIFF(TO\_END, TO\_START) AS 'Длительность тура(дней)' FROM tours. Тепер у нас є таблиця з тривалістю для кожного туру.

Вид запиту для виконання другої частини звіт: SELECT AVG(DATEDIFF(TO\_END, TO\_START)) FROM tours. Так як друга частина не повинна бути таблицею, то просто винесемо однин єдиний рядок з результуючої таблиці, використовуючи вже відому нам функцію PHP mysqli\_fetch\_row. І виведемо значення єдиного стовпчика.



Рисунок 3.45 - Приклад звіту

* + - 1. **Список турів за кожен місяць та рік**

Перш ніж почати створювати звіт, розглянемо функції і оператори MySQL, які будемо використовувати для цього.

Оператор DISTINCT - вказує SELECT, що потрібно вибирати унікальні значення. Це означає, що в результуючій таблиці не буде рядків, відповідні стовпчики якої повністю збігаються.

Функція YEAR - витягує з дати, яка передається як аргумент, рік.

Функція MONTH - схожа на YEAR, але витягує місяць.

Функція COUNT - повертає кількість записів SELECT запиту.

Отже приступимо до створення самого звіту.

Спочатку виконаємо наступний запит: SELECT TO\_NAME AS 'Имя тура', DATEDIFF(TO\_END, TO\_START) AS 'Длительность тура(дней)' FROM tours, щоб отримати всі роки і місяці без повторень. Потім для кожного місяця в році будемо створювати таблицю з турами в цей самий місяць (рис. 3.46).

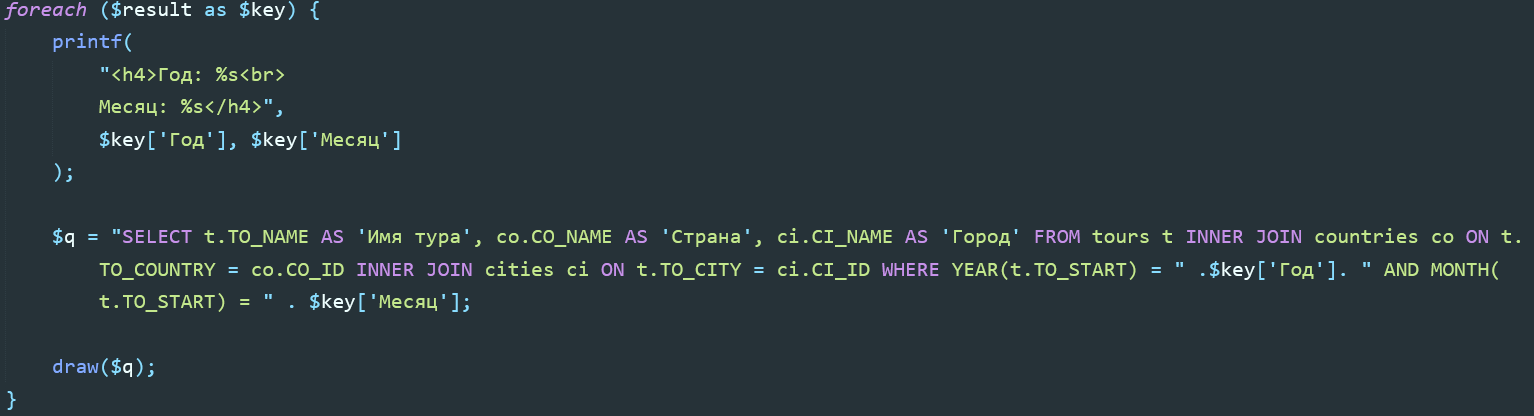


Рисунок 3.46 – Виводимо таблиці з даними

Зробити це можна використавши нашу функцію з наступним аргументом: SELECT t.TO\_NAME AS 'Имя тура', co.CO\_NAME AS 'Страна', ci.CI\_NAME AS 'Город' FROM tours t INNER JOIN countries co ON t.TO\_COUNTRY = co.CO\_ID INNER JOIN cities ci ON t.TO\_CITY = ci.CI\_ID WHERE YEAR(t.TO\_START) = " .$key['Год']. " AND MONTH(t.TO\_START) = " . $key['Месяц'], де key – це уявлення рядка результуючої таблиці. Виконавши це, ми отримаємо таблиці для кожного місяця і року.

Крім таблиць ми повинні вивести кількість турів в кожному році. Щоб виконати це, для початку отримаємо список всіх років запитом: SELECT DISTINCT YEAR(TO\_START) FROM tours. Далі все просто, в циклі для кожного року будемо виконувати запит: SELECT COUNT(TO\_ID) FROM tours WHERE YEAR(TO\_START) = " . $year\_row[0], де year\_row[0] – це поточний рік для у циклі. Цу дозволить для кожного року зі списку отриманого раніше порахувати кількість турів і вивести їх.

# **СПИСОК ДЖЕРЕЛ**

1. Wikipedia [Електроний ресурс – режим доступу: <https://uk.wikipedia.org>/]
2. PHPmanual [Електроний ресурс – режим доступу: <https://www.php.net/manual/ru/intro-whatis.php>]
3. TCPDF [Електроний ресурс – режим доступу: <https://tcpdf.org/>]
4. JavaScript [Електроний ресурс – режим доступу: <https://learn.javascript.ru/>]
5. ZametkiNaPolyah [Електроний ресурс – режим доступу: https://zametkinapolyah.ru/zametki-o-mysql/chast-3-2-vidy-svyazej-mezhdu-tablicami-v-baze-dannyx-svyazi-v-relyacionnyx-bazax-dannyx-otnosheniya-kortezhi-atributy.html]