Міністерство освіти і науки України

Відокремлений підрозділ Національного університету біоресурсів і природокористування України

Бережанський агротехнічний інститут

Факультет енергетики та електротехніки

**Кафедра інформаційних технологій та вищої математики**

напрям підготовки 122

«Комп’ютерні науки»

**КУРСОВА РОБОТА**

з освітньої компоненти: «Організація бази даних»

на тему:

**«Створення бази даних для автосервісу»**

Студента І курсу групи Кн-21Б

\_Мартинишина РОМАНА

Керівник:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Національна шкала: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_

Оцінка ECTS: \_\_\_\_\_\_\_\_\_\_\_

Члени комісії:

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

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали)

м. Бережани – 2023 рік

**Вступ**

**Завдання згідно варіанту**

1 **Теоретична частина.**

1.1Аналіз предметної області.

1.2Побудова моделі бази даних.

1.3Нормалізація бази даних

**2 Технічна реалізація MySQL**

2.1 Команди MySQL.

2.2 Створення бази даних

2.3 Вставка записів.

2.4 Побудова ERR-діаграми.

**3 Вибірка даних**

3.1Прості запити.

3.2 Складні запити.

3.3 Застосування агрегатних фунцій.

**4 Висновки**

5 Список літератури

**Вступ**

**База даних (БД)** — це організована структура, яка призначена для зберігання, зміни та обробки взаємозалежної інформації, переважно великих обсягів. Бази даних можуть зберігати відомості про людей, продукти, замовлення тощо.[1]

**Призначення**

Бази даних дають змогу зберігати інформацію, ця інформація виглядає як зв'язані між собою таблиці задопомогою певних ключів(звязків). Саме в БД зберігаються вся необхідна та корисна інформація для функціонування сайтів, а саме: клієнтські дані, прайс-лист, список товарів, дані замовлень.

Щоб створити запит до бази даних часто використовують Structured Query Language, скорочено: SQL. Це мова програмування яка дає змогу додавати, редагувати та видаляти інформацію, що міститься у таблицях.

Для взаємодії з базою даних використовують **СУБД**(система управління базами даних). Це програмне забезпечення яке спеціально використовується для створення, управління і оптимізації баз даних. СУБД включає в себе різноманітні функції такі як оновлення та аналіз даних, зберігання, пошук даних певного типу, тим самим ці програми забезпечують високий рівень надійності та безпеки опрацювання інформації. Прикладами найбільш популярних СУБД є MySQL, Oracle Database, Microsoft SQL Server та PostgreSQL.

**Детальніше про одну із СУБД, а саме: MySQL**

MySQL — це система керування реляційною базою даних з відкритим кодом, заснована на SQL. Він був розроблений і оптимізований для веб-додатків і може працювати на будь-якій платформі. Оскільки в Інтернеті з’явилися нові й інші вимоги, MySQL стала платформою вибору для веб-розробників і веб-додатків. Оскільки MySQL розроблено для обробки мільйонів запитів і тисяч транзакцій, це популярний вибір для компаній електронної комерції, яким потрібно керувати кількома грошовими переказами. Гнучкість на вимогу є основною особливістю MySQL.

MySQL — це СУБД, що стоїть за деякими з найкращих веб-сайтів і веб-додатків у світі, включаючи Airbnb, Uber, LinkedIn, Facebook, Twitter і YouTube.[2]

Висновок:

Бази даних представляють собою систематизовані структури, створені для зберігання, управління та обробки великих обсягів інформації. Їхнє важливе призначення полягає в забезпеченні ефективного та структурованого зберігання даних, які взаємодіють між собою. Бази даних є невід'ємною частиною сучасних інформаційних систем, і вони відіграють ключову роль у багатьох сферах, включаючи бізнес, науку, технології та інші галузі.

Важливість баз даних полягає в їхній здатності забезпечити надійне зберігання даних, швидкий доступ до інформації, забезпечення консистентності та цілісності даних. Вони сприяють покращенню ефективності бізнес-процесів, а також дозволяють вести аналітичні дослідження та формувати стратегічні рішення. Існування мови SQL та систем управління базами даних (СУБД) робить взаємодію з базами даних зручною та ефективною.

Отже, бази даних визнаються важливим інструментом для ефективного управління інформацією, надаючи потрібну структуру та функціональність для оптимізації роботи організацій та полегшення прийняття рішень на основі доступних даних.[3]

**Завдання згідно варіанту**

Створити базу даних для автосервісу, в якій будуть зберігатися дані про покупців, а саме(ПІБ, номер телефону, адреса електроної пошти і id транспорту) їхні транспортні засоби, і причину ремонту. Також треба зберігати дані про те, який ремонтний бокс використовувався при ремонті певного транспортного засобу, список деталей на складі, та інформація про працівників(ПІБ, адреса проживання, банківські дані, номер телефону). Дана інформація повина формувати чек таблицю в якій зберігається вся вище перерахована інформація і підбивається підсумок по ціні проробленого ремонту.  
Інформаційна система, що проектується, повинна вирішувати такі задачі:

-кількість замовлень за певний період часу

-які замовлення робили клієнти

-загальна ціна всії ремонтних боксів

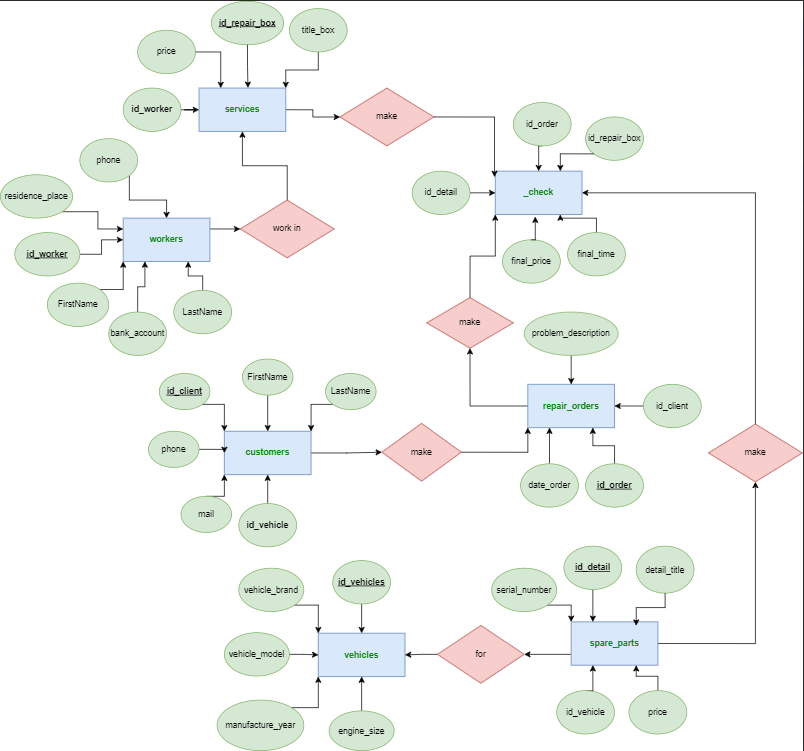
**Розділ 1. Теоретична частина.**

**1. Аналіз предметної області.**

Автосервіси праціють з транспортом, але як і інше підриємство їм доводиться вести певний облік, тобто працювати з великим обсягом однотипної інформації. Колись для цього використовувлось блокноти та записники, але з приходом компютерних технологій зявилась можливість оформляти це в електроні списки з якими зручніше працювати, ці списки це таблиці, а таблиці в свою чергу формують бази даних. Тому в нашому випадку автосервіса теж було би корисно оформити дані в таблиці бази даних, але які дані потрібно вносити? Для початку зробимо аналіз вже наявних даних. Коли до автосервісу приїжджає новий клієнт, нам потрібно дізнатись контактні дані цього клієнта, марку-модель його автомобіля і з якою проблемою він приїхав. Так в нас вже є дані для таблиці **customers.** Дані про причину поломки і дату створення запиту ми винесемо в окрему таблицю **repair\_orders,** також в цій таблиці зберігатиметьсяid\_клієнта на попередню таблицю. Для роботи з замовленями автосервісу потрібні працівники, інформація про яких зберігатиметься у таблиці **workers**. Ця теблиця зберігатиме інформацію про ПІБ працівника, контактні дані, бінківські рахунки. Для роботи працівників виділені спеціальні робочі бокси **services** інформація про ці бокси включа в себе назву боксу, ціна роботи в цьому боксі, і id закріпленого за цим боксом робочого. Інформація про те які машини обслуговує даний авосервіс винесена в окрему таблицю **vehicles** в цій таблиці інформація про марку-модель авто, рік випуску, кубатуру мотора. Також є таблиця **spare\_parts** в якій зберігається інформація про деталі на складі,а саме опис деталі, до якої машини, ціна. І вся ця інформація формує таблицю **\_check** в цій таблиці зберігаються id замовлення, використаних деталей, боксу, час закінчення робіт і фінальна ціна.

**2 Побудова моделі бази даних.**

Для побудови даної бази даних було використано програмне забезпечення для створення ER діаграми braw.io з ресурсу diagrams.net

****

**Опис Таблиць**

Таблиця “**customer”** містить інформацію про власників автомобілів які робили замовлення в автосервісі. В цій таблиці зберігається інформація про Ім’я і прізвище власника, його контактну інформацію, а саме номер телефону, і електрону пошту для звязку з особою.

Таблиця “**vehicles”** містить інформацію протранспортні засоби якіякими займається даний автосервіс. В даній таблиці зберігається марка і модель автомобіля, також є інформація про рік випуску даної моделі і об’єм мотора.

Таблиця “**spare\_parts**” зберігає інформацію про наявні на складі деталі для певних авто. В таблиці описано назву деталі, серійний номер, ціна, і за допомогою id аовязано з таблицею vehicles для розуміння до якого транспорту ця деталь.

Таблиця “**repair\_orders”** містить інформацію про замовлення на ремонт, в якому описується причина поломки, дата отримання замовлення, і є посилання на клієнта з таблиці customer за допомогою id.

Таблиця “**services”** містить інформацію про ремонтні бокси в яких проводиться кваліфікований ремонт в певній сфері. В цій таблиці є стовбець назви ремонтного боску, ціна за оренду цього боксу, і id на робочого який працює в цему боксі.

Таблиця “**workers**” містить інформацію про робочих які працюють в ремонтних боксах. В цій таблиці зберігається інформація про ім’я і прізвище робочого, місце проживання, номер телефону, і банківський рахунок.

Таблиця “**\_check”** це фінальна таблиця в якій зберігається певна інформація з попередніх таблиць за допомогою id, а саме з таблиці spare\_parts, repair\_orders і services. Також в цій таблиці зберігається інформація про дату і час закінчення робіт, і фінальна ціна робіт.

**3 Нормалізація бази даних.**

**Перша нормальна форма**

Перша нормальна форма утворює ґрунт для структурованої схеми бази даних:

Кожна таблиця повинна мати основний ключ. В даній базі даних це колонка id для кожного елемента таблиці

Уникнення повторень груп. Було створено окрему таблицю **vehicles** для уникнення повторення назв автомобілів в таблиці деталей і замовлень.

Атомарність: кожен атрибут повинен мати лише одне значення, а не множину значень.

**Друга нормальна форма**

Друга нормальна форма вимагає, аби дані, що зберігаються в таблицях із композитним ключем, не залежали лише від частини ключа:

Схема бази даних повинна відповідати вимогам першої нормальної форми.

Дані, що повторно з'являються в декількох рядках, виносяться в окремі таблиці.

**Третя нормальна форма**

Третя нормальна форма вимагає, аби дані в таблиці залежали винятково від основного ключа:

Схема бази даних повинна відповідати всім вимогам другої нормальної форми.

Будь-яке поле, що залежить від основного ключа та від будь-якого іншого поля, має виноситись в окрему таблицю

**Розділ 2. Технічна реалізація MySQL**

**1. Команди MySQL.**

При створені даної бази даних сикористовувалось таке СУБД як mySQL Workbench 8.0CE. Дане СУБД використовує мову програмування SQL, тому команди відповідні цій мові.

1. Створення бази даних:  
CREATE DATABASE назва\_бази\_даних;

Сттворює нову базу яка зявляється в лівому вікні огляду. Нова база адних повністю пуста.

2.Вибір активної бази даних:

USE назва\_бази\_даних;

Вибирає як основну при роботі з командами ту базу даних назву якої вказано в команді, вибирає з вже створених.

3.Створення таблиці:

CREATE TABLE назва\_таблиці(

Колонка 1 ; тип даних;

Колонка 2 ; тип даних;

);

Створює таблиці в поточній базі даних з перерахованими в душках колонками та іхніми типами даних.

4.Додавання первинного ключа:

ALTER TABLE назва\_таблиці

ADD PRIMARY KEY (назва колонки);

Додає первиний ключ до вказаної колонки в певній таблиці, використовується для зв’язків.

5.Додавання зовнішнього ключа:

ALTER TABLE таблиця1

ADD CONSTRAINT fk\_назва\_констрейнта

FOREIGN KEY (назва\_колонки\_таблиці1)

REFERENCES таблиця2(назва\_колонки\_таблиці2);

Додає зовнішній ключ до колонки тадлиці, теж використовується для встановлення зв’язків

6.Заповнення таблиць:

INSERT INTO таблиця(дані,,,,)

VALUES (‘’,’’,’’,’’,’’);

Додає введені дані до таблиці.

7. Видалення таблиці:

DROP TABLE назва\_таблиці;

Повністю видаляє таблицю, використовувати обережно!

8.Очищення таблиць:

DELETE FROM назва\_таблиці;

Видаляє всі дані з таблиці.

9. Зміна інформації в стовпцях:

UPDATE назва\_таблиці

SET назва\_стовпця = нове\_значення

WHERE умова;

Використовується для зміни даних в таблиці.

10.Видалення ствопців:

ALTER TABLE назва\_таблиці

DROP COLUMN назва\_стовпця,

Видаляє певний стовбець в вказаній таблиці.

11. Додавання нових стовпців:

ALTER TABLE назва\_таблиці

ADD COLUMN новий\_стовпець, тип\_даних,

Додає нові стовбці до таблиці.

**2. Створення бази даних**

Для створення бази даних використовувались вище згадані команди, але з даними які потрібні були для цієї бази даних. Ось декілька з команд які використовувались.

Створення бази даних car\_service:

create database car\_service;

Використання Бази даних як поточну:

USE car\_service;

Створення таблиці customers:

CREATE TABLE customers (

id\_client int primary key auto\_increment,

FirstName varchar(55),

LastName varchar(55),

phone varchar(20),

mail varchar(155)

);

Створення таблиці workers:

CREATE TABLE workers(

id\_worker int primary key auto\_increment,

FirstName varchar(55),

LastName varchar(55),

phone varchar(20),

residence\_place varchar(355),

bank\_account int

);

Створення таблиці vehicles:

CREATE TABLE vehicles (

id\_vehicles int primary key auto\_increment,

vehicle\_brand varchar(255),

vehicle\_model varchar(255),

manufacture\_year tinyint,

engine\_size tinyint

);

Створення підсумкової таблиці \_check:

CREATE TABLE \_check(

id\_order int,

id\_repair\_box int,

id\_detail int,

final\_time datetime,

final\_price varchar(150)

);

Внесення змін до таблиці vehicles:

ALTER TABLE vehicles

MODIFY COLUMN engine\_size int;

Створення зв’язка між таблицями repair\_orders(зовнішній ключ) і customers (первиний ключ):

ALTER TABLE repair\_orders

ADD FOREIGN KEY (id\_client) REFERENCES customers(id\_client);

Створення зв’язка між таблицями \_check(зовнішній ключ) і repair\_orders(первинний ключ)

ALTER TABLE \_check

ADD FOREIGN KEY (id\_order) REFERENCES repair\_orders(id\_order);

Всі команди створення бази даних винесені в окремий файл в додатку.

**3. Вставка записів**

Для вставки

Приклад вставки записів в таблицю customers за допомогою команди SQL insert into.

INSERT INTO customers (FirstName, LastName, phone, mail, id\_vehicle)

VALUES (

"Kosta",

"Bagry",

"+380976054777",

"Kosta228@gmail.com",

4

);

Вставка записів в таблицю vehicles:

INSERT INTO vehicles (vehicle\_brand, vehicle\_model, manufacture\_year, engine\_size)

VALUES (

"Porsche",

"911",

1963,

3800

);

Вставка записів в таблицю services:

INSERT INTO services (title\_box, price, id\_worker)

VALUES (

"бокс встановлення тюнінгу",

"2300грн",

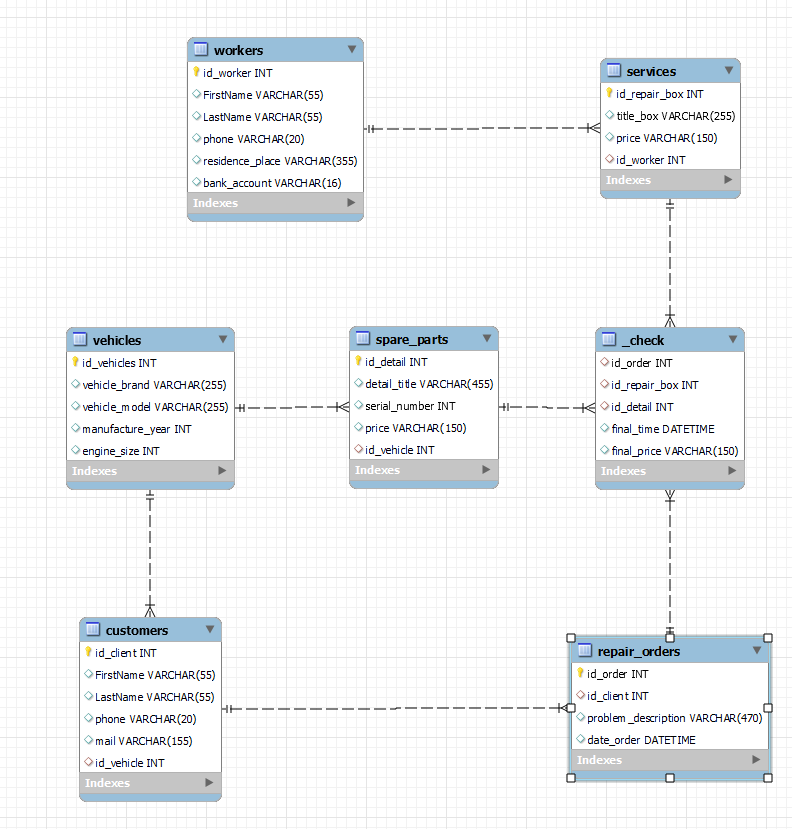
7

);

Приклади вставки записів в базу даних винесені в окремий файл в додатку.

**4 Побудова ERR-діаграми.**

Для побудови EER діаграми можна скористатися наявними в mySQL Workbench засобами, для цього достатньо в верхньому меню перейти на вкладку Database, далі шукаємо Reverse Engineer. Після цього відкриється нове вікно де ми нажимаємо Next, далі вписуєм свій пароль доступу до користувача в програмі, і нажимаємо Next. Вибираємо потрібну базу даних і нажимаємо Next. Погоджуємось зі всіма умовами, продовжуєм і далі ми зможемо отримати EER діаграму в програмі з якою ми працюємо.

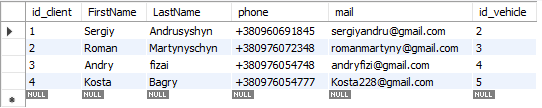


**Розділ 3. Вибірка даних**

**1 Прості запити.**

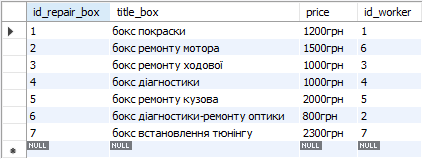
Вибірка всіх даних з таблиці **customers.**

SELECT \* FROM car\_service.customers;



Вибірка всіх даних з таблиці **services.**

SELECT \* FROM car\_service.services;

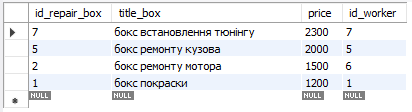
****

Вибірка даних з таблиці **services** ціна яких більша, рівна 1200грн і відсортована за зменшенням ціни.

SELECT \* FROM car\_service.services

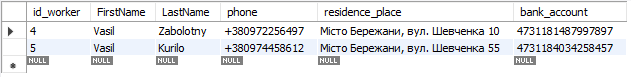
WHERE price >= 1200

ORDER BY price DESC;



Вибірка даних з таблиці workers про працівників які проживають за вулицею Шевченка.

SELECT \* FROM workers WHERE residence\_place LIKE '%вул. Шевченка%';



**2 Складні запити.**

Виведення інформації з таблиці services з зміною значень в стовпці id\_worker на дані на які посилається це id.

SELECT

services.\*,

workers.\*

FROM

services

JOIN workers ON services.id\_worker = workers.id\_worker;

****

Виведення інформації з таблиці \_check з обєднанням даних за значеням id на значення на які вони посилаються

SELECT

\_check.final\_time,

\_check.final\_price,

repair\_orders.problem\_description,

repair\_orders.id\_client,

services.title\_box,

spare\_parts.detail\_title

FROM

\_check

JOIN

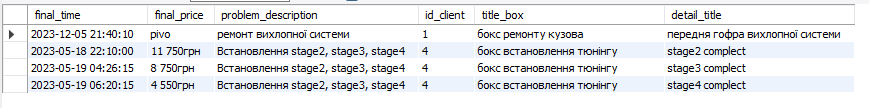
repair\_orders ON \_check.id\_order = repair\_orders.id\_order

JOIN

services ON \_check.id\_repair\_box = services.id\_repair\_box

JOIN

spare\_parts ON \_check.id\_detail = spare\_parts.id\_detail;

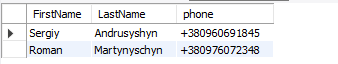
****

Виведення інформації про клієнтів з таблиці customers які робили замовлення після '2023-05-16'.

SELECT FirstName, LastName, phone

FROM customers

WHERE id\_client IN (SELECT id\_client FROM repair\_orders WHERE date\_order > '2023-05-16');

****

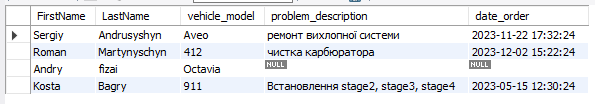
Виведення імя і прізвища про всіх клієнтів з таблиці customers, модель їхнього авто з таблиці vehicles і їхні замовлення, також виведена інформація про клієнтів які ще не зробили замовлення.

SELECT customers.FirstName, customers.LastName, vehicles.vehicle\_model, repair\_orders.problem\_description, repair\_orders.date\_order

FROM customers

LEFT JOIN repair\_orders ON customers.id\_client = repair\_orders.id\_client

LEFT JOIN vehicles ON customers.id\_vehicle = vehicles.id\_vehicles;



**3 Застосування агрегатних фунцій.**

Виведення кількості рядків в таблиці

SELECT COUNT(\*) FROM services;

****

Виведення загальної ціни за всі ремонтні бокси

SELECT SUM(price) FROM services;

****

Знаходження ремонтного боксу з мінімальною ціною за допомогою функції MIN().

SELECT title\_box, price

FROM services

WHERE price = (SELECT MIN(price) FROM services);

****

Знаходження ремонтного боксу з найбільшою ціною за допомогою функції MAX().

SELECT title\_box, price

FROM services

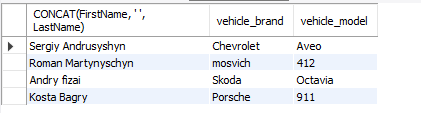
WHERE price = (SELECT MAX(price) FROM services);



Обєднання даних з двох стовпців в один за допомогою функції CONCAT().

SELECT CONCAT(FirstName, ' ', LastName), vehicles.vehicle\_brand, vehicles.vehicle\_model

FROM customers

JOIN vehicles ON customers.id\_vehicle = vehicles.id\_vehicles;****

**4 Висновки**

**1 Структура**

Згідно завдання була спроектовано план дій створення бази даних. Для початку була створена структура бази даних, за допомогою онлайн ресурса diagrams.net побудавана ER діаграма бази даних, спроектовано які таблиці потрібно створити і які типи даних надати стовпцям таблиць.

**2 використане ПО.**

У процесі вивчення та розробки бази даних для автосервісу використовувалися сучасні інструменти та програмні засоби, які значно полегшили весь процес проектування та розгортання. Основними інструментами, використаними для створення та управління базою даних, стали MySQL Workbench і diagrams.net.

MySQL Workbench надавав зручний інтерфейс для проектування схеми бази даних, моделювання таблиць, встановлення зв'язків між ними та визначення обмежень. Його інтеграція з сервером MySQL дозволила легко та ефективно розгортати структуру бази даних та управляти нею.

Diagrams.net (раніше відомий як draw.io) використовувався для створення діаграм та схем, що дозволяли візуалізувати взаємозв'язки між різними складовими системи, такими як таблиці, сутності та атрибути. Його функціональність сприяла наглядному поданню структури бази даних, що полегшувало сприйняття та спілкування з іншими учасниками проекту.

Загальний використаний підхід до проектування бази даних забезпечив не лише ефективну організацію даних для автосервісу, а й дозволив врахувати потреби користувачів, забезпечивши надійність та швидкий доступ до інформації. Використання таких інструментів, як MySQL Workbench і diagrams.net, сприяло процесу створення ефективної та добре структурованої бази даних для оптимізації роботи автосервісу.[3]

**Використані Джерела**

1 https://apeps.kpi.ua/shco-take-basa-danykh

Що таке бази даних

2 https://www.oracle.com/database/what-is-database/

# What Is a Database?(мова оригіналу:англійська)

3 https://chat.openai.com/  
 Чат на основі штучного інтелекту від OpenAI

**Додаток**

Данна База даних доступна для перегляду на github за посиланням: https://github.com/RomanMartynyshyn/kursova-bd