МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ТЕХНІЧНИЙ КОЛЕДЖ  
НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Відділення «інформаційних технологій»  
  
  
  
**КУРСОВА РОБОТА**З дисципліни «Бази даних»  
На тему:  
«Проектування та застосування бази даних інформаційної системи автосервісу»

Виконав студент групи 33-ПЗ Кравчук Н.В

Залікова книжка № 8629

|  |  |  |
| --- | --- | --- |
| Оцінка | Балів | Дата |
|  |  |  |

Керівник роботи Заяць М.М.

Львів 2015

ЗАВДАННЯ

на курсову роботу з дисципліни «Бази даних»

студент групи 33-ПЗ Кравчук Н.В

**Тема: «Проектування бази даних інформаційної системи**

**автосервісу»**

**Завдання:** автоматизувати процес формування та опрацювання бази даних «автосервісу».

**Зміст завдання та календарний план його виконання**

|  |  |  |
| --- | --- | --- |
| **№**  **з/п** | **Зміст завдання** | **Дата** |
| 1 | Здійснити аналітичний огляд літератури за заданою темою та обґрунтувати вибір інструментальних засобів проектування. | До 10.10 |
| 2 | Визначення та опис предметної області | До 11.11 |
| 3 | Побудова моделі «сутність-зв’язок» | До 13.11 |
| 4 | Побудова логічної структури бази даних | До 17.11 |
| 5 | Побудова обмежень відношень бази даних. Нормалізація відношень бази даних. | До 21.11 |
| 6 | Виконання над відношеннями операцій реляційної алгебри. | До 25.11 |
| 7 | Оформити записку до курсової роботи згідно з вимогами Міжнародних стандартів, дотримуючись такого змісту :   * Вступ; * Теоретична частина; * Визначення та опис предметної області; * Концептуальна модель «сутність-зв’язок»; * Логічна структура бази даних; * Нормалізація бази даних; * Виконання операцій реляційної алгебри; * Висновки; * Списки використаних джерел; * Додатки. | До 01.11 |

Завдання прийнято до виконання: (Кравчук Н.В)

(підпис студента)

Керівник роботи : /Заяць М.М./

Зміст

[1.Вступ. 4](#_Toc439006058)

[2.Теоретична частина. Бази даних в інтернеті. 5](#_Toc439006059)

[3. Визначення та опис предметної області 13](#_Toc439006060)

[4.Концептуальна модель «Сутність - Зв’язок 14](#_Toc439006061)

[5.Логічна структура бази даних. 15](#_Toc439006062)

[6.Нормалізація бази даних. 17](#_Toc439006063)

[7.Виконання операцій реляційної алгебри. 18](#_Toc439006064)

[8.Висновки. 20](#_Toc439006065)

[9.Список використаних джерел. 21](#_Toc439006066)

[10.Додатки. 22](#_Toc439006067)

[Додаток А. 22](#_Toc439006068)

[Додаток Б. 24](#_Toc439006069)

[Додаток В. 25](#_Toc439006070)

# 1.Вступ.

Бази даних виконують функцію систематизації знань. На основі цієї систематизації можуть створюватися нові знання. Так чи інакше, будь-яка база даних служить людині саме для опису подій, що відбулися у минулому, і на основі знання цих подій допомагає ухвалити те або інше рішення на майбутнє. База знань може бути побудована як мультимедійний довідник або як набір текстів і файлів іншого формату, проіндексованих за певними ознаками в базі даних.

**База даних** - це, перш за все, сховище об'єктів даних, тобто набору можливих понять або подій, що описуються базою даних, з можливістю пошуку цих об'єктів за ознаками. Невід'ємною межею бази даних є можливість скріплення об'єктів між собою. Базою даних можна вважати не тільки таблиці, що індексують файли із знаннями різних форматів, але і самі ці файли, тому, що вони є сховищами знань, що не типізуються, в такій базі даних.

Основною ціллю курсового проекту є закріплення, систематизація та поглиблення знань, отриманих під час вивчення дисципліни «Бази даних», а також розвинення практичних навичок з аналізу та проектування баз даних, розробки та налагодження програмного забезпечення для організації роботи зі спроектованою базою даних.

В даній курсовій роботі необхідно спроектувати базу даних «**Автосервіс**». Для реалізації цього завдання використовується програмне забезпечення Microsoft Office Access, яке містить в собі потрібний набір інструментів і засобів для роботи з реляційними базами даних.   
 Першим етапом буде створення концептуальної моделі «Сутність - Зв’язок». В ній буде визначено ключові атрибути сутностей,розроблено обмеження унікальності цілісності посилань, домену та загального вигляду для множини сутностей, атрибутів і зв‘язків.  
 Потім буде виконана нормалізація бази даних,а також деякі операції реляційної алгебри,а саме:

* Перетин,об’єднання,різниця відношення.
* Декартів добуток відношення.
* Селекція,проекція одного відношення.
* Натуральне та умовне з’єднання двох відношень.
* Включення,вилучення кортежів,зміна значень атрибутів в одному з відношень.
* Визначення нового атрибута,вилучення атрибута,зміна параметрів атрибута у одному з відношень.

# 2.Теоретична частина. Бази даних в інтернеті.

**Мова XML**   
Мова XML (extensible Markup Language - розширювана мова розмітки) була розроблена і підтримується консорціумом W3C. Вона розроблялася як мова розмітки документів, а не як мова опису баз даних. Розширюваність є головною відмінністю XML від іншої популярної мови розмітки — HTML. Спочатку фахівці вважали, що ця мова замінить HTML як мову публікації веб-документів, проте наявність у мові засобів визначення нових тегів, а також можливість створювати вкладені структури тегів дозволила використовувати XML для зображення даних складної структури, а не тільки документів. У зв'язку з цим мова стала інтенсивно використовуватися у додатках, що здійснюють обмін даними, а не просто як замінник HTML. Завдяки відкритості та розширюваності XML стала основою для нового покоління форматів збереження даних в Інтернеті.

**Атрибути**Елементи можуть мати атрибути, значення яких вказуються всередині початкового тегу елементу згідно з таким форматом: назва\_атрибута=значення. Елемент мо­же мати кілька атрибутів, але кожна назва атрибута має бути унікальним у межах елементу. Наведемо приклад означення атрибутів:

<факультет корпус=”5/1” назва="інформатики">

Зазначимо, що атрибути і вкладені елементи взаємозамінні. Наприклад, наведена щойно конструкція рівнозначна такій:

<факультет>

<корпус>5/1</корпус>

<назва>інформатики</назва>

</факультет>

**Розділи CDATA**Щоб символи текстового рядка, який визначає вміст елементу, не розпізнавалися як символи розмітки, вони записуються в так званому CDATA-розділі. Він може розміщуватися всюди, де допускається записувати рядкові дані. CDATA-розділ починається із запису <![CDATA] і закінчується символами ]]>. Всередині можуть вказуватись довільні символи, разом із тими, які використовуються для розмітки.

**Опис структури документа**У схемі бази даних описується, яка саме інформація може бути збережена в базі даних, як вона структурована і якими є типи значень, що зберігаються. У доку­ментах XML описувати структуру даних не обов'язково. Проте, щоб XML-дані міг автоматично інтерпретувати той, кому вони призначені, між ним і тим, хто ці дані надсилає, має існувати попередня домовленість щодо способу тлумачення даних. У зв'язку з цим у мові XML передбачена можливість описувати структуру даних у вигляді схем XML-даних. Є два механізми (мови) опису таких схем:

DTD (Document Type Definition — визначення типу документа), який сьогодні використовується дуже широко;

XML Schema — досить новий механізм, який наразі використовується рідше.  
Мова DTD.

Тип XML-документа може бути визначений за допомогою мови DTD, що дає змогу вказати:

* які саме елементи можуть зустрічатися в документі;
* які атрибути може або повинен мати елемент;
* які вкладені елементи може або повинен містити елемент і якою є їхня кіль­кість.

Загальний синтаксис DTD-опису є таким:

<!DOCTYPE назва\_документа [

<!ELEMENT назва\_елементу (специфікація\_вмісту)>

<!ATTLIST назва\_елементу специфікація\_атрибутів>

…

]>

**Специфікація вмісту елементу**Структура елементу документа XML специфікується означенням типу елементу і списку його атрибутів. Головна мета специфікації вмісту елементу полягає в то­му, аби вказати, елементи яких типів можуть бути вкладені до нього.

Ця специфікація може містити:

* cписок розділених пробілами назв елементів, які може або повинен містити даний елемент. Порядок елементів у переліку не суттєвий;
* взятий у круглі дужки список назв вкладених елементів, що розділяються комами. Порядок запису назв елементів у списку відповідає порядку запису вкладених елементів у XML-документі;
* взятий у круглі дужки список назв вкладених елементів, що розділяються символами |. Ці елементи є альтернативними, тобто один з елементів списку може або повинен бути вкладений у елемент, вміст якого специфікується;
* ключове слово #PCDATA (Parsed Character data), яке вказує, що елемент може містити лише рядки символів, але не вкладені елементи. В ієрархії елементів розділи PCDATA розташовані на найнижчому рівні;
* ключове слово EMPTY, яке означає, що елемент є порожнім;
* ключове слово ANY, яке вказує на те, що вміст елементу може бути довільним.

Зазначимо, що власне вміст елементів розглядається в XML лише як рядки символів і не вказується в DTD.

Є два різновиди БД із вбудованою підтримкою XML:

* орієнтовані на зберігання тексту (текстові XML-БД);
* орієнтовані на модель (модельні XML-БД).

У деяких XML-БД моделі документів зберігаються в реляційних чи об'єкт­но-орієнтованих базах даних. Наприклад, збереження моделі DOM у реляційній базі даних може привести до побудови таких таблиць, як Elements, Attributes, PCDATA, Entities та EntityReferences. В інших орієнтованих на модель базах даних із вбудованою підтримкою XML використовуються фізичні структури, спеціаль­но розроблені для оптимального збереження і маніпулювання моделлю докумен­та XML. Модельні XML-БД, побудовані на основі баз даних іншого типу, мають гірші характеристики продуктивності, особливо щодо пошуку даних, аніж ті, які використовують власні, спеціально розроблені під вибрану модель, формати збе­реження даних.

БД із вбудованою підтримкою XML мають підтримувати принаймні одну спеціалізовану мову запитів. Найбільш популярними мовами є XPath (з можливим розширенням для побудови запитів за багатьма документами) і XQuery. Деякі БД із вбудованою підтримкою XML оснащені власними спеціалізованими мова­ми запитів. Напевно, в майбутньому всі БД із вбудованою підтрим­кою XML будуть підтримувати мову XQuery.

**Оновлення й видалення документів**У БД із вбудованою підтримкою XML застосовується багато різних стратегій оновлення і видалення документів:

* просте видалення або заміна наявних документів;
* модифікація документів з використанням певної моделі DOM;
* використання спеціалізованих мов з метою модифікації документів або їхніх фрагментів.

Багато з цих методів маніпулювання документами є специфічними для конкретних XML-БД. Проте існують кілька стандартних мов маніпулювання документами XML. Наприклад, мова XUpdate, в якій для ідентифікації вершин використовуються вирази мови XPath, призначена для оновлення XML-документів. Крім того, членами робочої групи W3C XQuery було запропоновано кілька розширень мови XQuery засобами оновлення і видалення документів XML та їхніх фрагментів. Незабаром мова XQuery має бути офіційно розширена можливостя­ми з оновлення документів.

**Індексування**У БД із вбудованою підтримкою XML індекси використовуються для прискорення пошуку. Існують три типи індексів:

* індекси значень дають можливість індексу­вати значення атрибутів або тексту. Вони дозволяють здійснювати швидкий по­шук за запитами типу «Знайти всі елементи або атрибути, що містять значення бази даних ХМL»;
* індекси структури дають можливість індексувати розташування елементів і атрибутів та прискорювати пошук за запитами на зразок «Знайти всі елементи <заголовок>». Разом індекси значень і структури дають змогу відповіда­ти на запити типу «Знайти всі елементи <заголовок>, що містять значення бази даних XML»;
* повнотекстові індекси дають можливість індексувати всі слова в тексті та значеннях атрибутів і використовуються при реалізації запитів типу «Знайти документи, що містять усі слова з фрази бази даних ХМL», або спільно зі структурними індексами для обробки запитів типу «Знайти документи, що містять усі слова з фрази бази даних XML в елементах <заголовок>».

Як правило, в базах даних із вбудованою підтримкою XML застосовуються індекси значень і структу­ри, а інколи й повнотекстові індекси.

**Мови запитів**Один з найпростіших підходів до відображення даних, що зберігаються в базі, полягає в їхній безпосередній інтерпретації відповідно до тієї моделі, за правилами якої вони були введені. Проте структура документа XML, як правило, відрізня­ється від структури бази даних, тому під час передавання даних з бази в документи XML доцільно скористатися спеціальною проміжною мовою, такою як XSLT - мова перетворення таблиць стилів.

Проблему формування документів на основі даних, що зберігаються в базі, мо­же вирішити реалізація мови запитів, вирази якої повертатимуть документи XML. Як правило, такі вирази базуються на використанні фрази SELECT, вкладеної в шаблони. Очікується, що питання вибору документів з бази даних буде вирішено з реалізацією мов XQuery і SQL/XML. На жаль, майже всі XML-мови запитів ма­ють лише засоби вибору документів (зокрема XQuery 1.0 і перша версія SQL/ XML) і в них відсутні засоби вставки, заміни і видалення даних.

**Мови запитів на основі шаблонів**

Багато мов запитів, що будують XML-документи на основі даних з реляційних баз, використовують шаблони. У цих мовах відсутній будь-який наперед визначе­ний спосіб відображення структури бази даних у структурі документа. Замість цього команди вибирання даних вбудовуються в шаблон, що обробляється про­міжним програмним забезпеченням, призначеним для передавання даних. На­приклад у шаблоні, що наводиться далі, елемент <SelectStmt> використовується для включення в нього фрази SELECT, а записи вигляду $назва\_стовпця визначають, де мають бути розміщені результати застосування шаблону:

<?xml version="1.0"?>

<FlightInfo>

<Introduction>C місця на такі рейси:</Introduction

<SelectStmt>SELECT Airline. FltNumber. Depart. Arrive FROM Flights</SelectStmt> <Flight>

<Airline>$Airline</Airline>

<FltNumber>$FltNumber</FltNumber>

<Depart>$Depart</Depart>

<Arrive>$Arrive</Arrive>

</Flight>

<Conclusion> Ми сподіваємося. Ви знайдете те, що Вам необхідно.</ Conclusion>

</FlightInfo>

Після застосування цього шаблону можна отримати такий XML-документ:

<?xml version="1.0"?>

<FlightInfo>

<Introduction> Є місця на такі рейси:</Introduction>

<Flights>

<Flight>

< Airline>Авіалінії України</Airline>

<FltNumber>123</FltNumber>

<Depart>17 серпня 2005: 16:05</Depart>

<Arrive>18 серпня 2005 01:21</Arrive>

</Flight>

</Flights>

<Conclusion> Ми сподіваємося. Ви знайдете те, що Вам необхідно.</Conclusion>

</FlightInfo>

Мови запитів на основі шаблонів відзначаються високою гнучкістю. Вони надають такі можливості:

* набори значень, що вибираються, можна розмістити в будь-якому місці документа, котрий формується, зокрема їх можна використовувати як параметри послідовних фраз SELECT, не обмежуючись лише форматуванням результатів, як показано в наведеному вище прикладі;
* можна використовувати змінні й означувати функції;
* підтримуються цикли і конструкції іf-then;
* підтримується параметризація речень SELECT, наприклад за допомогою параметрів HTTP.
* Мови запитів на основі шаблонів використовуються переважно для переда­вання даних з реляційної бази у XML-документ.

**Мови запитів на основі SQL**У мовах запитів, призначених для формування XML-документів, але розробле­них на основі SQL, використовується модифікований варіант фрази SELECT, ре­зультат виконання якої перекладається на XML. Є кілька мов, що були розробле­ні й реалізовані в конкретних XML-БД. У найпростішому випадку в таких мовах використовуються вкладені фрази SELECT, що обробляються згідно з принципами об'єктно-реляційного відображення. Існують також мови, що перетворюють об'єк­ти SQL 3 безпосередньо на елементи XML.

У 2000 році кілька компаній об'єдналися з метою розробки стандарту, що інтегруватиме XML і SQL. Нині загальновизнаним є стандарт ISO, що одержав на­зву XML/SQL. У мові XML/SQL введено тип даних XML і реалізовано функції, що дають змогу будувати елементи й атрибути XML на основі даних із реляцій­них баз. Наприклад, нижче мовою XML/SQL записано запит, результатом якого є таблиця з двох стовпців:

SELECT Orders.SONumber.

XMLELEMENT(NAME "Order",

XMLATTRIBUTES(Orders.SONumber AS SONumber).

XMLELEMENT(NAME "Date", Orders.Date).

XMLELEMENT(NAME "Customer", Orders.Customer)) AS xmldocument FROM Orders

Перший стовпець містить номер замовлення, другий — документ XML. Кожен рядок містить один документ XML, що створюється з даних відповідного рядка таблиці замовлень.

**Робота з базами даних через мережу Інтернет.**У міру зростання популярності Інтернету як базового середовища для різноманітних інформаційних систем висуваються все вищі вимоги до механізмів, що реалізують взаємодію користувачів і додатків з базами даних через мережу Інтернет. Розробники сучасних СКБД прагнуть задовольнити вимоги користувачів, надаючи різні механізми пошуку й обробки даних в Інтернеті.

Користувач за допомогою браузера формує запит на звернення до бази даних. Цей запит містить:

* назву комп'ютера, на якому розташований веб-сервер;
* назву CGI-сценарія, якому має бути передано запит для його подальшого передавання СКБД;
* власне текст запиту, що обробляється СКБД.

Запит надходить на вказаний веб-сервер за протоколом передавання гіпертексту HTTP. Сервер, розпізнавши в запиті звернення до CGI-сценарію, надсилає за­пит йому. CGI (Common Gateway Interface — загальний шлюзовий інтерфейс) специфікує правила взаємодії веб-серверів з прикладними програмами. У нашому випадку CGI-сценарій, прийнявши запит, перетворить його на звернення до бази даних і передасть серверу додатків, серверу баз даних або СКБД для подаль­шої обробки. Це може бути запит на оновлення або вибирання даних. Якщо це був вибірковий запит, то отримавши дані з бази, CGI-сценарій передає їх веб-серверу (відповідно до інтерфейсу CGI), який, у свою чергу, пересилає їх браузеру.

**Метод доступу до даних**Доступ до даних, що містяться у вказаній користувачем базі, здійснюється одним з таких методів:

* за запитом з URL;
* за допомогою шаблонів запитів;
* за допомогою запитів XPath.

# 3. Визначення та опис предметної області

Предметна область «Автосервіс».

Автосервіс займається діагностикою і ремонтом легкових автомобілів. Для прискорення процесу обслуговування клієнтів інформація про замовлення. Процес виконання ремонтних робіт починається з надходження замовлення від клієнта, у якості якого можуть виступати фізичні і юридичні особи. Замовлення має ідентифікаційний номер і дату. особисті дані клієнтів повинні зберігатися в базі даних.

Автомобіль клієнта обов'язково проходить попередню обробку: мийку, огляд і визначення виду ремонту. Після, згідно виду ремонту, автомобіль направляється при необхідності на діагностику і далі відбувається сам процес ремонту. Наприкінці обслуговування здійснюється перевірка якості ремонту автомобіля і усунення можливих недоліків. Коли все готове клієнт оплачує замовлення. Для представлення даних предметної області в реляційної базі даних зручніше спочатку побудувати модель «сутність-зв'язок».

База даних буде складатися з таких таблиць:

1. Клієнт.
2. Автомобілі.
3. Види робіт.
4. Замовлення.

База даних буде складатися з таких полів:

1. Клієнт: (Код власника, ПІБ, Паспортні данні, Телефон, Адреса).
2. Автомобілі: (Код, Марка, ТехПаспорт, Колір, Рік випуску).
3. Види робіт: (Код, Назва роботи, Ціна роботи, Час виконання, Гарантія).
4. Замовлення: (Код, Код авто, Код роботи, Код власника, Дата поступлення).

# 4.Концептуальна модель «Сутність - Зв’язок

Для побудови реляційної бази даних виберемо такі сутності:

* Клієнт
* Автомобілі
* Види робіт
* Замовлення

Кожній сутності буде відповідати множина атрибутів.

Сутність «Клієнт» буде складатись з такої множини атрибутів:

* Код власника
* ПІБ
* Паспортні данні
* Телефон
* Адреса

Сутність «Автомобілі» буде складатись з такої множини атрибутів:

* Код
* Марка
* ТехПаспорт
* Колір
* Рік випуску

Сутність «Види робіт» буде складатись з такої множини атрибутів:

* Код
* Назва роботи
* Ціна роботи
* Час виконання
* Гарантія

Сутність «Замовлення» буде складатись з такої множини атрибутів:

* Код
* Код авто
* Код роботи
* Код власника
* Дата поступлення

# 5.Логічна структура бази даних.

Тут ми детальніше розглянемо таблиці нашої бази даних,а саме їхні поля,типи полів,розмірності полів і деякі властивості полів.  
**Клієнт –** інформація про клієнта.  
**Автомобілі** – інформація про автомобілі.  
**Види робіт –** інформація про види робіт які може виконувати автосервіс.  
**Замовлення** – інформація про замовлення які виконував або виконує автосервіс.

**Структура таблиць.**

|  |  |  |
| --- | --- | --- |
| **Назва поля** | **Тип поля** | **Розмірність** |
| Код | Числовий | Довге ціле |
| Марка | Текстовий | 30 |
| ТехПаспорт | Текстовий | 10 |
| Колір | Текстовий | 15 |
| Рік випуску | Числовий | ціле |

Таблиця Клієнт:

|  |  |  |
| --- | --- | --- |
| **Назва поля** | **Тип поля** | **Розмірність** |
| Код власника | Числовий | Довге ціле |
| ПІБ | Текстовий | 40 |
| Паспортні данні | Текстовий | 8 |
| Телефон | Текстовий | 20 |
| Адреса | Текстовий | 50 |

Таблиця Автомобілі:

Таблиця Види робіт:

|  |  |  |
| --- | --- | --- |
| **Назва поля** | **Тип поля** | **Розмірність** |
| Код | Числовий | Довге ціле |
| Назва роботи | Текстовий | 30 |
| Ціна роботи | Числовий | Довге ціле |
| Час виконання | Дата/Час | - |
| Гарантія | Числовий | Довге ціле |

Таблиця Замовлення:

|  |  |  |
| --- | --- | --- |
| **Назва поля** | **Тип поля** | **Розмірність** |
| Код | Лічильник | Довге ціле |
| Код авто | Числовий | Довге ціле |
| Код роботи | Числовий | Довге ціле |
| Код власника | Числовий | Довге ціле |
| Дата поступлення | Дата/Час | - |

Для деяких полів встановимо обмеження та додаткові властивості:

**Клієнт**Код власника – ключове поле.  
Паспортні данні – маска LL999999.  
Телефон – маска "+(380)("00\)000\-00\-00.

**Автомобілі**Код - ключове поле.  
ТехПаспорт – маска LL999999LL.  
Рік випуску – маска 0000.

**Види робіт**Код – ключове поле.  
Час виконання - маска 99:00:00;0;

**Замовлення**Код – ключове поле.

У додатку А зображено таблиці.  
У додатку Б зображено схему бази даних.

# 6.Нормалізація бази даних.

**Нормалізація схеми бази даних** - покроковий процес розбиття одного відношення (на практиці: таблиці) відповідно до алгоритму нормалізації на декілька відношень на базі функціональних залежностей. Отримані відношення необхідно перевірити на відповідність трьом нормальним формам або нормальну форму Бойса-Кодда.

**Ненормалізована форма (ННФ)** - таблиця, що містить одну або кілька повторюваних груп даних.

**Перша нормальна форма (1 НФ)** - відношення, в якому на перетині кожного рядка і кожного стовпця міститься одне й тільки одне значення.

**Друга нормальна форма (2НФ)** - відношення, яке знаходиться в першій нормальній формі і кожен атрибут якого, не входить до складу первинного ключа, характеризується повною функціональною залежністю від цього первинного ключа.

**Третя нормальна форма (ЗНФ)** - відношення, яке знаходиться в першій і в другій нормальних формах і не має атрибутів, що не входять в первинний ключ атрибутів, які перебували б у транзитивній функціональній залежності від цього первинного ключа.

Таблиці Клієнт, Автомобілі, Види робіт та Замовлення мають 3 нормальну форму.

# 7.Виконання операцій реляційної алгебри.

Реляційна алгебра - замкнута система операцій над відношеннями в  
реляційній моделі даних.   
Операції реляційної алгебри також називають реляційними операціями.   
Всі операції над відношеннями за їх змістом, особливостями та способом використання поділяють на групи:

* Теоретико - множинні;
* Спеціальні реляційні;
* Операції над схемами відношень;
* Операції над станами відношень.

Теоретико-множинні операції грунтуються на тому, що кожне відношення бази даних може розглядатись як множина однотипних кортежів , і , відповідно , до них можуть бути застосовані операції , запозичені з теорії множин.

Теоретико-множинні операції поділяють на:

* Операції об'єднання;
* Операції перетину;
* Операції різниці;
* Операції декартового добутку;

Спеціальні реляційні операції поділяють на:

* Операція проекції відношень
* Операція селекції відношень
* Операція натурального з’єднання відношень
* Операція умовного з’єднання

 Операції над станом відношення:

* Операція додавання кортежів
* Операція вилучення кортежів
* Операція зміни значень атрибутів у кортежах
* Операції над схемами відношень
* Операція визначення нового атрибуту у відношенні
* Операція вилучення атрибуту з відношення
  + Операція зміни атрибуту у відношенні

Загальна інтерпретація реляційних операцій:

* При виконанні операції об'єднання двох відношень отримується відношення, що включає всі кортежі, які входять хоча б в один з відношень;
* операція перетину двох відношень отримуємо відношення, що включає всі кортежі , які входять в обидва відношення;
* При виконанні операції різниці двох відношень отримуємо відношення, що включає всі кортежі які є різні у двох відношеннях;
* При виконанні операції декартового добутку двох відношень отримуємо відношення, що вилучає об’єднання кожного коржа першого відношення з другим;
* При виконанні операції проекції відношення отримуємо відношення кортежів яких ми вказали;
* При виконанні операції селекції відношення отримуємо відношення на вибір даних за певним критерієм;
* При виконанні операції натурального з’єднання двох відношень отримуємо відношення, що з’єднує дві таблиці по спільному кортежу за допомогою операції INNERJOIN;
* При виконанні операції умовного з’єднання двох відношень отримуємо відношення, що з’єднює дві таблиці по спільному кортежу;
* При операції додавання кортежу у відношення ми додаємо новий кортеж у відношення за допомогою операції INSERTINTO () VALUES();
* При операції зміни значення атрибута у кортежі ми оновлюємо кортеж за допомогою операції UPDATE;
* При операції вилучення кортежу з відношення вилучається кортеж за допомогою операції DELETE;
* При операції визначення нового атрибута у відношенні, у відношення додається новий атрибут за допомогою команди ADDCOLUMN(назва атрибуту);
* При операції вилучення атрибуту з відношення, з відношення видаляється визначений атрибут за допомогою команди DROPCOLUMN(назва атрибуту);
* При операції зміни значення атрибута змінюється значення атрибута за допомогою команди ALTERCOLUMN назва атрибуту (значення);

Результати всіх операцій зображені у Додатку В.

# 8.Висновки.

В даній курсовій роботі я навчився застосовувати методи побудови реляційної бази даних для конкретної предметної області,а саме:

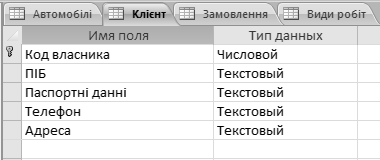
* Побудувати модель типу «Сутність - Зв'язок» для предметної області  
   «Автосервіс»;
* побудували структуру бази даних і визначили атрибути, відношення і зв'язки для даної бази даних;
* за допомогою таблиць баз даних СУБД MS Access, спроектував базу даних для предметної області «Автосервіс»;
* навчився виконувати операції реляційної алгебри над відношеннями даної бази даних.

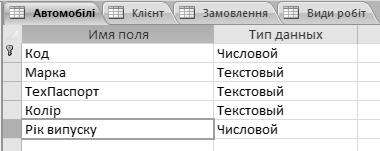
# 9.Список використаних джерел.

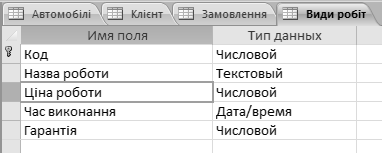
1. **http://ukrbukva.net/print:page,1,65177-Razrabotka-avtomatizirovannoiy-sistemy-uchieta-raboty-avtoservisa.html**
2. **http://bibliofond.ru/view.aspx?id=445279#1**
3. **http://www.slideshare.net/vika\_kopoty/ss-21247372**
4. **http://uk.wikipedia.org/**
5. **http://bridge.tneu.edu.ua/web\_info/lectures/part8/08-05.html**

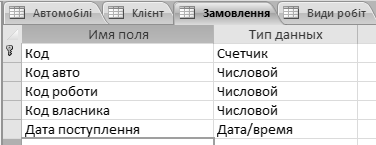
# 10.Додатки.

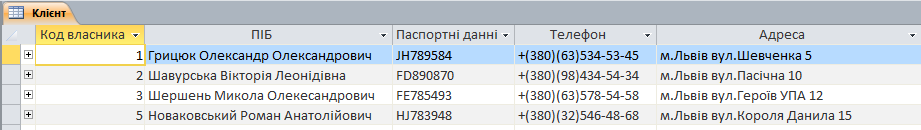
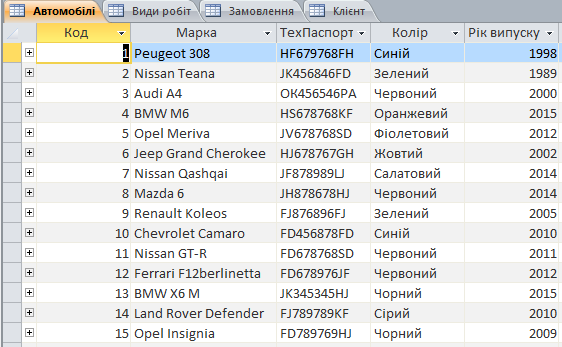
## Додаток А.

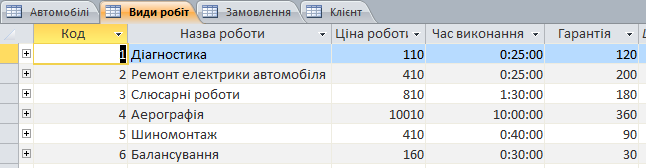
  
Таблиця *Клієнт* в режимі конструктора.

  
Таблиця *Автомобілі* в режимі конструктора.

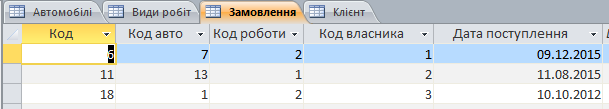
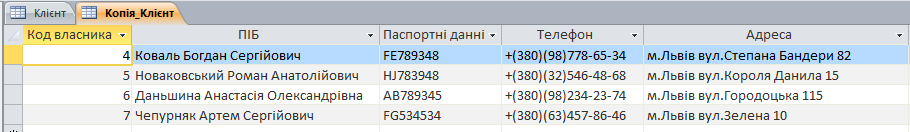
  
Таблиця *Види робіт*  в режимі конструктора.

  
Таблиця *Замовлення*  в режимі конструктора.

  
Початковий стан таблиці *Клієнт.*

Початковий стан таблиці *Автомобілі*

Початковий стан таблиці *Види робіт*  *.*

  
Початковий стан таблиці *Замовлення*  *.*  Початковий стан таблиці *Копія\_Клієнт.*

## Додаток Б.

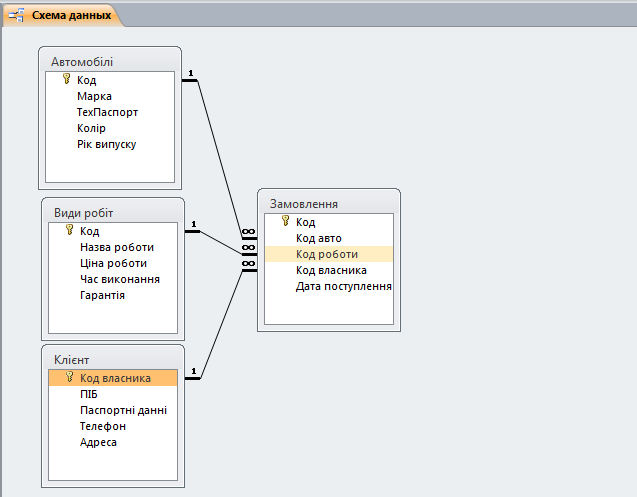
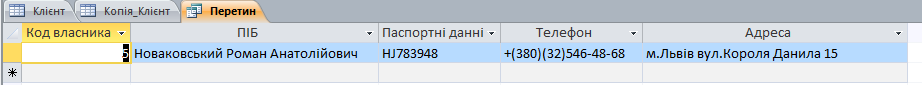
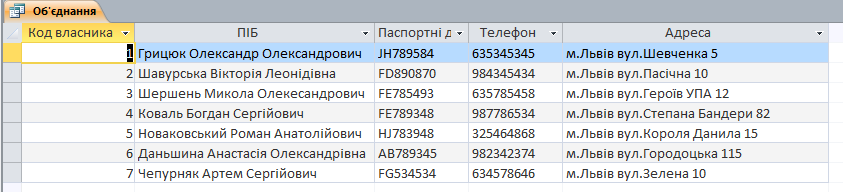
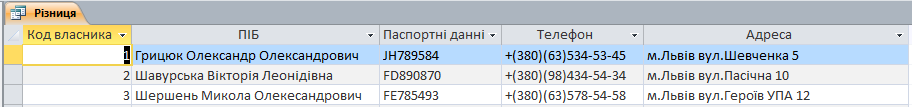


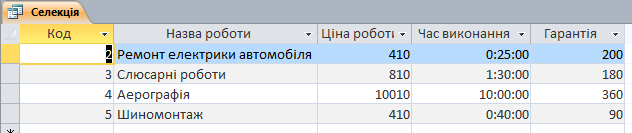
Схема бази даних «**Автосервіс**».

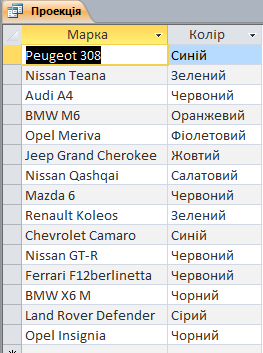
## Додаток В.

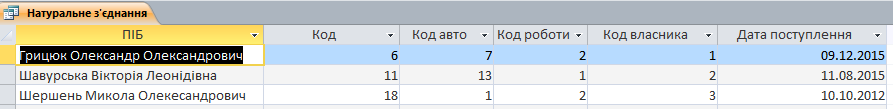
**Запит SQL на перетин таблиць *Клієнт і Копія\_Клієнт.***Операція перетину отримує таблицю, до складу якої входять лише такі рядки, які зустрічаються одночасно і в таблиці ***Клієнт***, і в таблиці ***Копія\_Клієнт***.

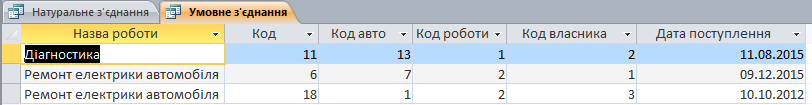
SELECT \*FROM Клієнт WHERE (((Клієнт.[Код власника])   
In (SELECT Копія\_Клієнт.[Код власника] FROM [Копія\_Клієнт])));  
**Результат виконання запиту:  
Запит SQL на об’єднання таблиць *Клієнт і Копія\_Клієнт.***Операція об'єднання дозволяє об'єднувати результати окремих запитів по декількох таблицях в єдину результуючу таблицю. Таким чином, команда UNION об'єднує виведення двох або більше SQL-запитів в єдиний набір рядків і стовпцівSELECT \* FROM Клієнт UNION SELECT \*FROM Копія\_Клієнт;  
**Результат виконання запиту:**  
 **Запит SQL на різницю таблиць *Клієнт і Копія\_Клієнт.***Операція різниці отримує таблицю, до складу якої входять лише ті рядки з таблиці ***Клієнт***, яких нема в таблиці ***Копія\_Клієнт***.SELECT \*FROM Клієнт WHERE (((Клієнт.[Код власника])  
 Not In (SELECT Копія\_Клієнт.[Код власника] FROM [Копія\_Клієнт])));   
**Результат виконання запиту:**

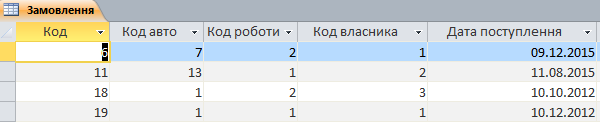
**Запит SQL на декартів добуток таблиць *Автомобілі і Клієнт.***Операція декартового добутку отримує таблицю, що складається з рядків, які є конкатенацією (або злиттям) одного рядку першої таблиці та одного рядку другої таблиціSELECT \* FROM [Види робіт], Клієнт;  
 **Результат виконання запиту:**

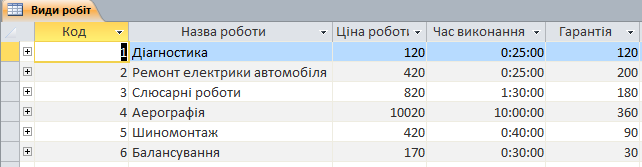
**Запит SQL на селекцію в таблиці *Види робіт.***Операція селекції дозволяє отримати всі рядки (записи) або частину рядків однієї таблиці за допомогою певної умови.SELECT [Види робіт].\*FROM [Види робіт] WHERE ((([Види робіт].[Ціна роботи])>400));   
**Результат виконання запиту:**  


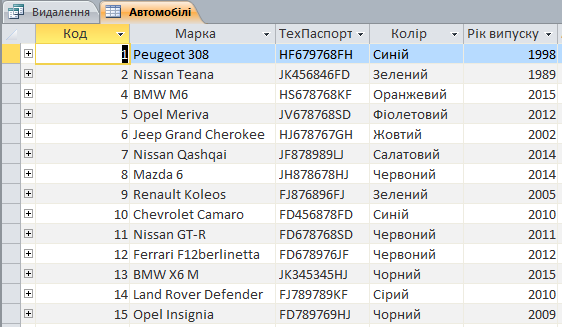
**Запит SQL на проекцію таблиці *Автомобілі.***Операція проекції дозволяє виділити підмножину стовпців таблиці.SELECT Автомобілі.Марка, Автомобілі.Колір FROM Автомобілі; **Результат виконання запиту:**  


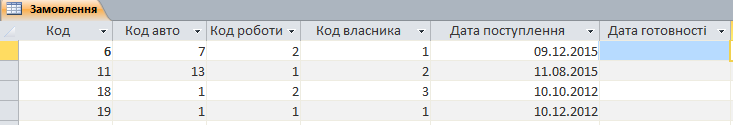
**Запит SQL на натуральне з’єднання таблиць *Клієнт і Замовлення.***Операція натуральне з’єднання дозволяє з'єднувати рядки з більш ніж однієї таблиці (за деякою умовою) для утворення нових рядків даних.SELECT Клієнт.ПІБ, Замовлення.\* FROM Клієнт INNER JOIN Замовлення ON Клієнт.[Код власника] = Замовлення.[Код власника]; **Результат виконання запиту:**

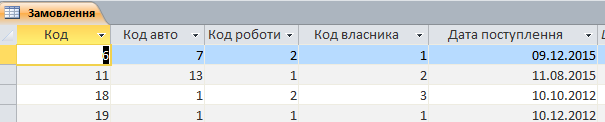
**Запит SQL на умовне з’єднання таблиць Види робіт *і Замовлення.***Ця операція використовується, коли необхідно з'єднати два відношення на основі деяких умов, відмінних від еквівалентності.SELECT [Види робіт].[Назва роботи], Замовлення.\*FROM [Види робіт], Замовлення WHERE [Види робіт].Код= [Замовлення].[Код роботи]; **Результат виконання запиту:**

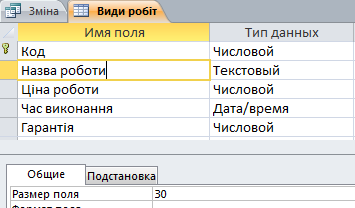
**Операція на додавання кортежу у таблицю *Замовлення.***INSERT INTO Замовлення ( [Код авто], [Код роботи], [Код власника], [Дата поступлення] ) VALUES (Код, Тип, Назва, Дата); **Результат виконання запиту:**

**Операція на оновлення кортежу у таблиці *Види робіт.***UPDATE [Види робіт] SET [Види робіт].[Ціна роботи] = [Види робіт].[Ціна роботи]+10; **Результат виконання запиту:**

**Операція на видалення кортежу з таблиці *Автомобілі.***DELETE Автомобілі.[Рік випуску], \*FROM Автомобілі  
WHERE (((Автомобілі.[Рік випуску])=2000)); **Результат виконання запиту:**

**Операція додавання нового атрибута у таблицю Замовлення*.***ALTER TABLE Замовлення ADD COLUMN [Дата готовності] DATETIME ;  
**Результат виконання запиту:**

**Операція видалення атрибута з таблиці *Замовлення.***ALTER TABLE Замовлення DROP COLUMN [Дата готовності];  
**Результат виконання запиту:**

**Операція зміни значення атрибута у таблиці *Види робіт.***Початковий стан таблиці Види робіт.  
  
ALTER TABLE [Види робіт] ALTER COLUMN [Назва роботи] VARCHAR(50);  
**Результат виконання запиту:**  
