МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

КУРСОВА РОБОТА

з дисципліни «Системи баз даних»

на тему: «Розробка бази даних як системи для додатка-помічника по підбору одягу за прогнозом погоди»

Керівник :
ст. викл. каф. БМК,
Сердаковський В.С.

Виконав:
студент гр. БС-81, ФБМІ
Сєров О. В.
залікова книжка № БС-8122

Допущено до захисту
″___″ _____ 2020 _____
підпис

Захищено з оцінкою
_____ _____
оцінка підпис
″___″ _____ 2020

Київ-2020

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут (факультет) БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва)

Кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва)

ЗАВДАННЯ
на курсову роботу студенту

СЄРОВ ОЛЕКСАНДР ВЛАДИСЛАВОВИЧ

(прізвище, ім'я, по батькові)

1. Тема роботи Розробка бази даних як системи для додатка-помічника по підбору одягу за прогнозом погоди

2. Термін подання студентом роботи ***20 грудня 2020 року***

3. Вихідні дані до роботи Індивідуальне завдання студента

4. Зміст роботи Реалізувати реляційну базу даних для зберігання даних одягу користувачів

5. Дата видачі завдання ***20 вересня 2020 р.***

Календарний план

№ з/п	Назва етапів виконання курсової роботи	Термін виконання етапів роботи	Примітка
1	Отримати завдання на КР	20 вересня 2020р.	
	Оформлення розділу з		
	Оформлення розділу з		
	Оформлення розділу з		
	Оформлення КР		
	Подання в електронному вигляді КР та анотації до неї на сайт кафедри.	До 20 грудня 2020р.	
	Подання пакету документів по КР до захисту	20 грудня 2020р.	
	Захист КР	23-27 грудня 2020р	

Студент _____ Олександр СЄРОВ
(підпис) (ініціали, прізвище)

Керівник роботи _____ Віталій СЕРДАКОВСЬКИЙ
(підпис) (ініціали, прізвище)

ЗМІСТ

ВСТУП

За останні десятиліття бази даних стали основою інформаційних систем і докорінно змінили спосіб роботи багатьох організацій. Досягнення технологій баз даних призвели до створення програм, які є досить ефективними та простими у використанні. Вони зберігають дані, інформацію про базу даних клієнтів та іншу інформацію зручним та надійним способом, який виходить за рамки можливостей традиційних електронних таблиць.

Актуальність створення бази даних одягу користувачів зумовлена необхідністю обробки великих обсягів даних, оскільки помічники у повсякденних справах стають все більш популярними протягом останнього часу. Тема курсової роботи - розробка бази даних як системи для додатка-помічника по підбору одягу за прогнозом погоди. Об’єднавши дані у базі даних, алгоритми додатку знатимуть який одяг є у користувача та дасть пораду щодо оптимального одягу за погодою.

Метою курсу є отримання знань у галузі "Організація баз даних та знань", розробка бази даних для супровіду бази даних, як системи для додатка-помічника по підбору одягу за прогнозом погоди та автоматизації роботи з системою.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ СТУДЕНТА

Розробити базу даних для одягу користувача. База даних повинна зберігати інформацію про одяг користувача: модель одягу, назву бренду, оцінку теплозбереження, оцінку водонепроникності.

В БД є такі таблиці та дані:

Користувач (унікальний ідентифікатор користувача, ім’я, електронна пошта,);

Одяг (модель одягу, назву бренду, оцінку теплозбереження, оцінку водонепроникності);

База повинна містити необхідну кількість представлень, що надають користувачеві комфортний спосіб роботи з даними, та виконувати максимально можливу кількість завдань автоматично за допомогою тригерів, тим самим позбавляючи користувача від зайвої роботи і зменшуючи можливості допустити помилки.

Приклад завдань, які повинна вирішувати дана система:

Підбір усього одягу при поточній погоді

Підбір усього одягу за погодою на вечір / весь день / ніч

Підбір одягу на декілька днів / тиждень

Підбір одягу в поїздку - за прогнозом погоди в іншому місті

Підбір взуття при поточній погоді

і т. д.

АНОТАЦІЯ

«Курсова робота » з дисципліни «Системи баз даних» являється частиною циклу Загальної підготовки дисциплін першого (бакалаврського) рівня вищої освіти ступеня «бакалавр», зі спеціальності 122 «Комп'ютерні науки» за спеціалізацією «Комп'ютерні технології в біології та медицині»).

Загальна трудомісткість освоєння модуля становить 1 кредитів (ЕКТС), 30 годин.

Курсову роботу виконав Сєров Олександр Владиславович студент 3 курсу, гр. БС-81 кафедри Біомедичної кібернетики факультету Біомедичної інженерії НТУУ «КПІ ім. Ігоря Сікорського» .

Тема роботи : «Розробка бази даних як системи для додатка-помічника по підбору одягу за прогнозом погоди». Варіант 15.

Розглянуті питання: аналіз інформаційного наповнення бізнеспроцесів, моделювання залежностей елементів даних, побудова діаграм, розробка та тестування бази даних.

Результати по роботі:

отримано навики з проектування, розробки, організації, тестування та оптимізації SQL-баз даних.

отримано результати з області моделювання UML-діаграм для сутностей БД, створення відповідних таблиць та їх практичного застосування в області медицини.

Створено базу відповідно до індивідуального завдання.

Структура і обсяг роботи: курсова робота складається із вступу, опису завдання, 3 розділів, висновків, списку використаної літератури із 19 джерел і 6 додатків. Загальний обсяг курсової роботи становить 42 сторінки, основного тексту (без додатків) - 28 сторінок, ілюстрацій - 5, таблиць – 10.

АННОТАЦИЯ

«Курсовая работа» по дисциплине «Системы баз данных» является частью цикла Всеобщей подготовки дисциплин первого (бакалаврской) уровня высшего образования степени «бакалавр», по специальности 122 «Компьютерные науки» по специализации «Компьютерные технологи в биологии и медицине»).

Общая трудоемкость освоения модуля составляет 1 кредитов (ЕКТС), 30 часов.

Курсовую работу выполнил Серов Александр Владиславович студент 3 курса, гр. БС-81 кафедры Биомедицинской кибернетики факультета биомедицинской инженерии НТУУ «КПИ им. Игоря Сикорского».

Тема работы: «Разработка базы данных как системы для приложения-помощника по подбору одежды за прогнозом погоды». Вариант 15.

Рассматриваемые вопросы: анализ информационного наполнения бизнес-процессов, моделирование зависимостей элементов данных, построение диаграмм, разработка и тестирование базы данных.

Результаты по работе:

- получено навыки по проектированию, разработке, организации, тестирования и оптимизации SQL-баз данных.

- получены результаты из области моделирования UML-диаграмм для сущностей БД, создание соответствующих таблиц и их практического применения в области медицины.

- Создана база в соответствии с индивидуальным заданием.

Структура и объем работы: курсовая работа состоит из введения, описания задачи, 3 глав, заключения, списка использованной литературы из 19 источников и 6 приложений. Общий объем курсовой работы составляет 42 страниц основного текста (без приложений) - 28 страниц, иллюстраций – 5, таблиц - 10.

ABSTRACT

"Course work" in the discipline "Databases" is part of the cycle of General preparation of the disciplines of the first (bachelor) higher education degree "Bachelor", specialty 122 "Computer Science" in the specialization "Computer Technology in Biology and Medicine").

The total complexity of mastering the module is 1 credit (ECTS), 30 hours.

The course work was performed by Serov Alexander student of 3-year, gr. BS-81 of the Department of Biomedical Cybernetics, Faculty of Biomedical Engineering, NTUU “KPI them. Igor Sikorsky”.

R&D: " Development of a database as a system for a helper application for the selection of clothing according to the weather forecast". Option 15.

Issues considered: analysis of information content of business processes, modeling of dependencies of data elements, charting, development and testing of a database.

Results on work:

- acquired skills in designing, developing, organizing, testing and optimizing SQL databases.

- results are obtained from the field of modeling UML-diagrams for DB entities, creation of corresponding tables and their practical application in the field of medicine.

- Created base according to individual task.

Structure and scope of work: course work consists of an introduction, a description of the task, 3 sections, conclusions, a list of used literature from 19 sources and 6 appendices. The total amount of course work is 42 pages, the main text (without appendices) - 28 pages, illustrations - 5, tables - 10.

СПИСОК СКОРОЧЕНЬ

СУБД – система управління базами даних

БД - база даних

ПЗ – програмне забезпечення

Рис. – рисунок

Табл. – таблиця

VC – varchar

BI – bigint

UQ – unique (унікальний)

PK – primary key (первинний ключ)

FK – foreign key (зовнішній ключ)

NN – not null (не нульовий)

AI – auto increment (автоінкремент)

ОСНОВНА ЧАСТИНА РОБОТИ
ТЕОРЕТИЧНА ЧАСТИНА БАЗИ ДАНИХ

1.1. Бази даних

База даних (БД) - це впорядкований набір логічно-пов’язаних даних, що є спільним та призначений для задоволення інформаційних потреб користувачів. У технічному сенсі, включаючи систему управління базами даних.

Система управління базами даних (СУБД) - це сукупність програмних та мовних засобів, необхідних для створення, оновлення баз даних та організації пошуку необхідної інформації.

Централізований характер управління даними в базі даних включає необхідність того, щоб особа (група осіб) відповідала за управління даними, що зберігаються в базі даних.

Основна функція бази даних полягає у забезпеченні зберігання значної кількості інформації та наданні доступу користувачеві чи додатку. Таким чином, база даних складається з двох частин: збереженої інформації та системи управління. Для забезпечення ефективного доступу записи даних організовані як сукупність фактів (елементів даних).

Існує величезна кількість баз даних, які відрізняються за критеріями

Класифікація бази даних відповідно до моделі даних:

- ієрархічний,
- мережа,
- реляційний,
- об'єкт,
- об'єктно-орієнтований,
- об’єктно-реляційні.

1.2. Діаграма «Сутність – зв’язок»

Модель взаємозв'язку сутності (модель ER) або діаграма взаємозв'язку сутності - це модель даних, яка дозволяє описувати концептуальні діаграми за допомогою узагальнених блокових конструкцій. Модель ER - це метамодель даних, тобто інструмент для опису моделей даних. Існує багато моделей

подання знань, але одним із найзручніших інструментів уніфікованого представлення даних, незалежно від програмного забезпечення, яке його реалізує, є модель "сутність-зв'язок". Важливо те, що модель "сутність-зв'язок" здатна створити всі існуючі моделі даних (ієрархічну, мережеву, реляційну, об'єктну), тому це найпоширеніша.

Модель точка-взаємозв'язок є результатом систематичного процесу, який описує та визначає певну тематичну область. Він не визначає сам процес, він лише візуалізує його. Дані подаються у вигляді компонентів (сутностей), які пов’язані певними зв’язками, що виражають залежності та вимоги, що існують між ними, наприклад: будівля може бути розділена на нульове або багатоквартирне житло, але житло - це лише будівля. Суб'єкти можуть мати різні властивості (атрибути), що їх характеризують. Діаграми, створені для графічного представлення цих сутностей, атрибутів та зв’язків, називаються діаграмами зв’язків сутності.

Модель ER зазвичай реалізується у формі баз даних. У реляційній базі даних, в якій дані зберігаються в таблицях, кожен рядок у кожній таблиці є екземпляром сутності. Деякі поля даних у таблицях вказують на індекси в інших таблицях. Ці поля є показниками фізичної реалізації відносин між сутностями.

Визначення елементів даних - опис сутностей, визначених у сутностях в атрибутах, їх типу (текст, число, логічне значення, дата тощо) та можливих обмежень (унікальність, а не можливість нульового значення). Далі подано у вигляді таблиці (Таблиця 2).

1.3. Специфікація елементів даних

Визначення елементів даних - опис сутностей, визначених в сутності в атрибутах, їх тип (текст, число, логічне значення, дата тощо) та можливі обмеження (унікальність, а не можливість нульового значення). Далі подано у вигляді таблиці (Таблиця 2).

1.4. Тригери

Тригер - це особливий тип збереженої процедури, який не викликається явно користувачем і використання якого зумовлене виникненням певної події (операції) в реляційній базі даних:

• Додати INSERT

• видалити рядок із зазначеної таблиці ВИДАЛИТИ,

• або шляхом зміни даних у певному стовпці таблиці UPDATE.

Тригери використовуються для забезпечення цілісності даних та реалізації складної бізнес-логіки. Тригер запускається сервером автоматично при спробі змінити дані в таблиці, з якою він пов'язаний. Будь-які внесені ним зміни даних вважаються внесеними в транзакції, в якій транзакція була ініційована. Відповідно, у разі помилки або порушення цілісності даних ця транзакція може бути скасована.

Час активації визначається за ключовими словами ДО (активація ініціюється до пов'язаної події, наприклад, перед додаванням запису) або ПІСЛЯ (після події). Якщо тригер події викликається до події, він може модифікувати запис, модифікований подією (за умови, звичайно, якщо подія не є чіткою для запису). Деякі бази даних накладають обмеження на операторів, які можуть бути використані в тригері (наприклад, може бути заборонено вносити зміни до таблиці, на якій "висить" панель запуску тощо).

Крім того, тригери можуть бути пов'язані зі стадією (VIEW) замість таблиці. У цьому випадку механізм "оновленого типу" реалізується з їх допомогою. У цьому випадку ключові слова ДО і ПІСЛЯ впливають лише на порядок активаторів, оскільки сама подія (видалення, вставлення чи оновлення) не відбувається.

На деяких серверах тригери можуть викликатися не один раз для таблиці, а для кожної зміни запису. Такі тригери називаються табличними.

1.5. Представлення

Представлення (VIEW) - об’єкт бази даних, який є результатом запиту до бази даних, визначеного оператором SELECT під час запиту.

Перегляди іноді називають "віртуальними таблицями". Ця назва пов’язана з тим, що представлення створюється як таблиця для користувача, але не містить даних і при доступі витягує їх із таблиць. Якщо ви змінюєте дані в базовій таблиці, під час доступу до подання користувач отримує поточні дані, які використовують цю таблицю; кешування результатів вибірки з таблиці, якщо подання не виконуються. У цьому випадку механізм кешування запитів працює на рівні запиту користувача, незалежно від того, чи має користувач доступ до таблиць або подань.

Представлення можна створювати з таблиць та інших подань, тобто їх можна вбудовувати (до 32 рівнів вбудовування).

1.6. Реляційна модель даних

Реляційна модель даних - логічна модель даних. Вперше його запропонував британський вчений IBM Едгар Франк Кодд (E. Ф. Кодд) у 1970 році у статті під назвою "Реляційна модель даних для великих спільних баз даних". В даний час ця модель є фактичним стандартом, який зосереджений майже на всіх сучасних комерційних системах управління базами даних (СУБД).

Реляційна модель досягає вищого рівня абстракції, ніж ієрархічний або мережевий. У цій статті EF Кодд стверджує, що "реляційна модель дозволяє описувати дані лише на основі їх природної структури, тобто без необхідності введення будь-якої додаткової структури з метою машинного представлення". Іншими словами, подання даних не залежить від їх фізичної організації. Це забезпечується використанням математичного поняття відношення (сама назва «реляційний» походить від англійського relationship - «відношення»).

Реляційна модель зосереджена на упорядкуванні даних у вигляді двовимірних таблиць. Кожна реляційна таблиця є двовимірним масивом.

Модель реляційної бази даних використовує атрибути (стовпці) та записи (рядки) для зберігання та впорядкування інформації. На сьогодні модель реляційної бази даних є найбільш широко використовуваною.

Реляційна база даних складається з двовимірних таблиць, кожна таблиця містить унікальні рядки, стовпці та комірки. Кожна клітинка містить лише одне значення даних, яке є конкретним значенням атрибута відповідного запису.

Реляційна модель даних зазвичай включає теорію нормалізації. Крістофер Дат виділив три складові реляційної моделі даних: структурну, маніпулятивну, цілісну.

Структурна частина моделі визначає, що єдиною структурою даних є нормоване n-арне відношення. Зв'язок може бути представлений у вигляді таблиць, де кожен рядок є подвійним, а кожен стовпець є атрибутом, визначеним у певному діапазоні. Цей неформальний підхід до концепції відносин робить форму представлення більш відомою розробникам та користувачам, де реляційна база даних складається з кінцевих таблиць.

Маніпуляційна частина моделі визначає два основних механізми управління даними - реляційну алгебру та реляційні обчислення. Основним завданням маніпуляційної частини реляційної моделі є забезпечення метрик відносності для будь-якої конкретної мови в реляційних базах даних: мова називається реляційною, якщо вона має не менш виражальні здібності та потужність, ніж реляційна алгебра або реляційні обчислення.

Невід’ємною частиною моделі визначено вимоги до цілісності сутностей та посилань. Перша вимога полягає в тому, що будь-який двійник будь-якого відношення повинен відрізнятися від будь-якого іншого подвійного відношення, тобто кожне відношення повинно мати первинний ключ. Вимога щодо цілісності посилання або вимоги зовнішнього ключа полягає в тому, що значення кожного зовнішнього ключа, що з'являється у посиланні, має бути подвійним з однаковим значенням первинного ключа, або значення зовнішнього ключа має бути невизначеним (тобто не вказувати ні на що).

Може існувати аналогія між елементами реляційної моделі даних та елементами моделі сутність-взаємозв'язок. Реляційні відносини відповідають набору сутностей, а набори відповідають сутностям. Отже, як і в моделі сутності-

відносини, стовпці, що представляють реляційні відносини в таблиці, називаються атрибутами.

Кожен атрибут визначений на діапазоні, тому діапазон можна вважати набором допустимих значень для атрибута. Кілька атрибутів одного і того ж зв’язку або навіть атрибути різних зв’язків можуть бути визначені в одному домені.

Іменовану пару називають відношенням «ім’я атрибута - доменне ім’я». Потужність цього набору називається ступенем або "арнессією" відносин. Набір іменних схем з'єднань є схемою бази даних.

Атрибут, значення якого однозначно ідентифікує рядки, називається ключем (або просто ключем). Якщо рядки ідентифікуються лише шляхом об'єднання значень кількох атрибутів, зв'язок, як кажуть, має складний ключ. З’єднання може містити кілька ключів. Один із ключів завжди оголошується первинним, його значення не може бути оновлено. Усі інші клавіші підключення називаються можливими.

На відміну від реляційних ієрархічних та мережевих моделей даних, не існує поняття групових відносин. Копіюючи ключі, ви можете відобразити асоціації між наборами різних відносин.

1.7. Бізнес-процеси

Бізнес-процес - логічна послідовність операцій (етапів), що виконуються в рамках даної системи, метою яких є аналіз та контроль певних дій учасників процесу.

Моделювання бізнес-процесів - це процес, який відображає суб’єктивне бачення робочого процесу у формі офіційної моделі взаємопов’язаних операцій.

1.8. Елементи даних та їх залежність

Функціональна залежність (ФЗ) - це концепція, яка лежить в основі багатьох питань, пов'язаних з реляційними базами даних, включаючи, зокрема, їх дизайн. Математично це взаємозв'язок між наборами атрибутів відносин і, по

суті, відносини один до багатьох. ЗЗ забезпечує основу для наукового підходу до вирішення певних проблем, оскільки має цікаві формальні властивості.

1.9. Індекси

Індекс - це об’єкт бази даних, створений для підвищення продуктивності запитів. Таблиці в базі даних можуть містити велику кількість рядків, що зберігаються у довільному порядку, і таблиці може знадобитися багато часу, щоб переглянути їх один за одним за певним значенням. Індекс складається із значення одного або декількох стовпців у таблиці та вказівників на відповідні рядки в таблиці, що дозволяє вам знайти рядок, який потрібно для даного значення. Прискорення роботи з використанням індексів обумовлено головним чином тим, що структура індексу оптимізована для пошукової системи - наприклад, збалансоване дерево.

1.10. Таблиці

У реляційних базах даних таблиця - це набір елементів даних (значень), які сортуються за моделлю вертикальних (з різними іменами) стовпців та горизонтальних рядків. У таблиці є певна кількість стовпців, тоді як кількість рядків може змінюватися в різний час. Кожен рядок ідентифікується спеціальним набором стовпців, який називається потенційним ключем.

Нормалізація відносин - поетапний процес розбиття початкових відносин бази даних на простіші. Етапи цього процесу перетворюють схему підключення до бази даних у послідовні звичайні форми. Кожна наступна форма має кращі властивості, ніж попередня. Усі нормальні форми відповідають певним обмеженням. При перекладі структури відносин у форми вищого порядку із таблиць видаляється непотрібна описова інформація. Процес нормалізації базується на концепції функціональної залежності атрибутів.

Копіювання даних викликає проблеми при виконанні операцій з базами даних. Ці проблеми виникають під час спроби виконати дії: редагувати, додавати чи видаляти дані.

Після аналізу сутностей у предметній області та визначення їх функціональних та багатозначних залежностей була проведена нормалізація базової схеми шляхом розбиття сутностей на елементи, описані в таблицях.

1.11. Обмеження цілісності

Цілісність - взаємозв'язок внутрішньої єдності, кожної частини чогось, в єдине ціле. В інформаційній системі статус даних або інформаційної системи, в якій дані та програми використовуються за призначенням, що вимагає:

• стабільна робота системи;

• автоматичне відновлення, якщо система виявляє потенційну помилку;

• автоматичне використання альтернативних деталей замість несправних.

Цілісність бази даних - це стан бази даних, коли всі значення даних є правильними в тому сенсі, що вони відображають стан реального світу (у визначених межах точності та часової послідовності) і підпорядковуються правилам взаємної узгодженості. Підтримка цілісності бази даних включає перевірку цілісності та відновлення виявлених аномалій; це частина функції адміністратора бази даних.

Правила цілісності бази даних - це правила, які дозволяють вводити неправильні дані до бази даних і дозволяють зв’язувати кілька таблиць. Ці правила можна описати під час створення або модифікації таблиці.

Типи правил цілісності:

CHECK - Перевірка наявності дійсних значень для атрибутів.

NOT NULL / NULL - Заборонити / дозволити використання невизначених чи невизначених значень.

UNIQUE - Контролюйте унікальність значень атрибутів.

PRIMARY KEY - Первинний ключ.

FOREIGN KEY - Зовнішній ключ.

PRIMARY KEY (PK) використовується для ідентифікації рядків таблиці, він має певні характеристики:

У таблиці бази даних може бути лише один ПК.

Рядки, до яких застосовується це правило, не можуть містити невизначені або невизначені дані.

Це обмеження, призначене для кількох стовпців (складених ключів), визначає унікальність комбінацій відповідних значень, хоча унікальне значення кожного стовпця у складеному ключі не обов'язково є унікальним.

FOREIGN KEY (FG) використовується для з'єднання двох таблиць і має такі властивості:

Поле в головній таблиці, до якого буде звертатися FG, має бути оголошено як PK або UNIQUE.

Поле в головній таблиці та поле, з якого складається адреса, повинні мати однаковий тип даних

Ви можете використовувати FG для видалення та оновлення даних.

Обмеження CHECK дозволяє встановити умову, яка повинна відповідати значенню, вказаному в таблиці, перш ніж ви зможете прийняти його. Обмеження CHECK складається з ключового слова CHECK та предиката, який використовує вказане поле. Будь-яка спроба змінити або вставити значення поля, яке може призвести до недійсності цього твердження, буде відхилено. Це запобіжить введенню небажаних даних. Обмеження CHECK може використовуватися як маска введення для управління заданим форматом вхідних даних.

Обмеження UNIQUE, як обмеження PRIMARY KEY, обмежує набір значень у зазначених стовпцях унікальними значеннями. Як і обмеження PRIMARY KEY, обмеження UNIQUE може бути обмеженням таблиці, а потім визначає унікальність комбінацій значень у відповідних стовпцях.

Обмеження NULL / NOT NULL дозволяють або забороняють вводити значення NULL у поле. Очевидно, що обмеження NOT NULL повинно бути вказано для первинних ключів, інакше цілісність даних буде порушена.

Висновки з розділу 1

В результаті було опрацьовано теоретичні відомості, які допоможуть в проектуванні та реалізації бази даних для додатка-помічника по підбору одягу за прогнозом погоди.

2. МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ БАЗИ ДАНИХ

2.1. Актуальність додатку для поради одягу

На теперішній час, сфера послуг, галузь, яка надає послуги для людей, але не призводить до виробництва товарів, користується попитом та зростає у кількості охочих користуватися розумними сервісами з кожним роком. Інтернет-банкінги, інтернет-магазини, додатки для відео-конференцій, додатки для таксі, трекінг показників здоров'я, додатки для розумного будинку та міста та спектр сервісів для оптимізації часу, роботи та тощо.

Серед такої великої кількості сервісів для покращення життя прийнятно вписується додаток, який може рекомендувати поміж одягу користувача оптимальний варіант для погоди, яка очікує його при подальшому перебуванні на вулиці.

2.2. Аналіз існуючих аналогів інформаційної системи.

Перший сайт за запитом «outfit by weather» - dailydressme.com

Сайт на якому відстежується місцеположення пристрою та за погодою в цій місцевості надається 5 оптимальних жіночих аутфітів на 5 днів уперед. Явними недоліками цієї системи являється:

Повна відсутність чоловічого одягу

Відсутність можливості додати вже свій одяг, окрім рекламованого

Відсутність варіативності – лише 1 аутфіт на 1 день

Сайт являється лише рекламою для інтернет-магазина і не може використовуватися у реальних кейсах.

Найпопулярніший аналог - dressbyweather.com

Сайт на якому можна ввести дані про місцеположення, стать та вимір температури та отримати стандартні типи одягу чи аксесуари під ці дані.

Але відсутність свого одягу, рекомендацій усього одягу (відсутність штанів/джинсів) та деякі недоробки та баги роблять цей сайт не дуже зручним для повсякденного використання. Хоч і цей варіант на ступінь вище по зручності у використанні ніж перший сайт.

2.3. Бізнес процеси та елементи даних

2.3.1 Перелік бізнес-процесів

Слід враховувати, що представлена інформаційна система є лише ядром повноцінного програмного додатку, тому присутній ряд обмежень, пов’язаних із її використанням:

Представлена база даних є лише основою для програмного додатку і може виконувати не всі завдання які потрібно, наприклад давати різні рівні доступу (користувач, адміністратор). Таким чином, користувач міг би лише переглядати дані, адміністратор – редагувати інформацію про користувача, його активність.

База даних передбачає збереження та накопичення даних про одяг користувачів.

Для даної бази даних слід розробити інтуїтивно зрозумілий графічний інтерфейс як web-сторінку, або додаток для різних платформ.

Інформаційна система має давати можливість зберігати дані згідно з рівнем доступу.

На діаграмі «сутність-зв’язок» було виділено попередньо описані 3 сутностей та деякі сутності, які не входять в розробку БД, але входять в прецеденти. (рис. 2.1).

Перелік бізнес-процесів ІС наведений в таблиці (табл. 2.1).

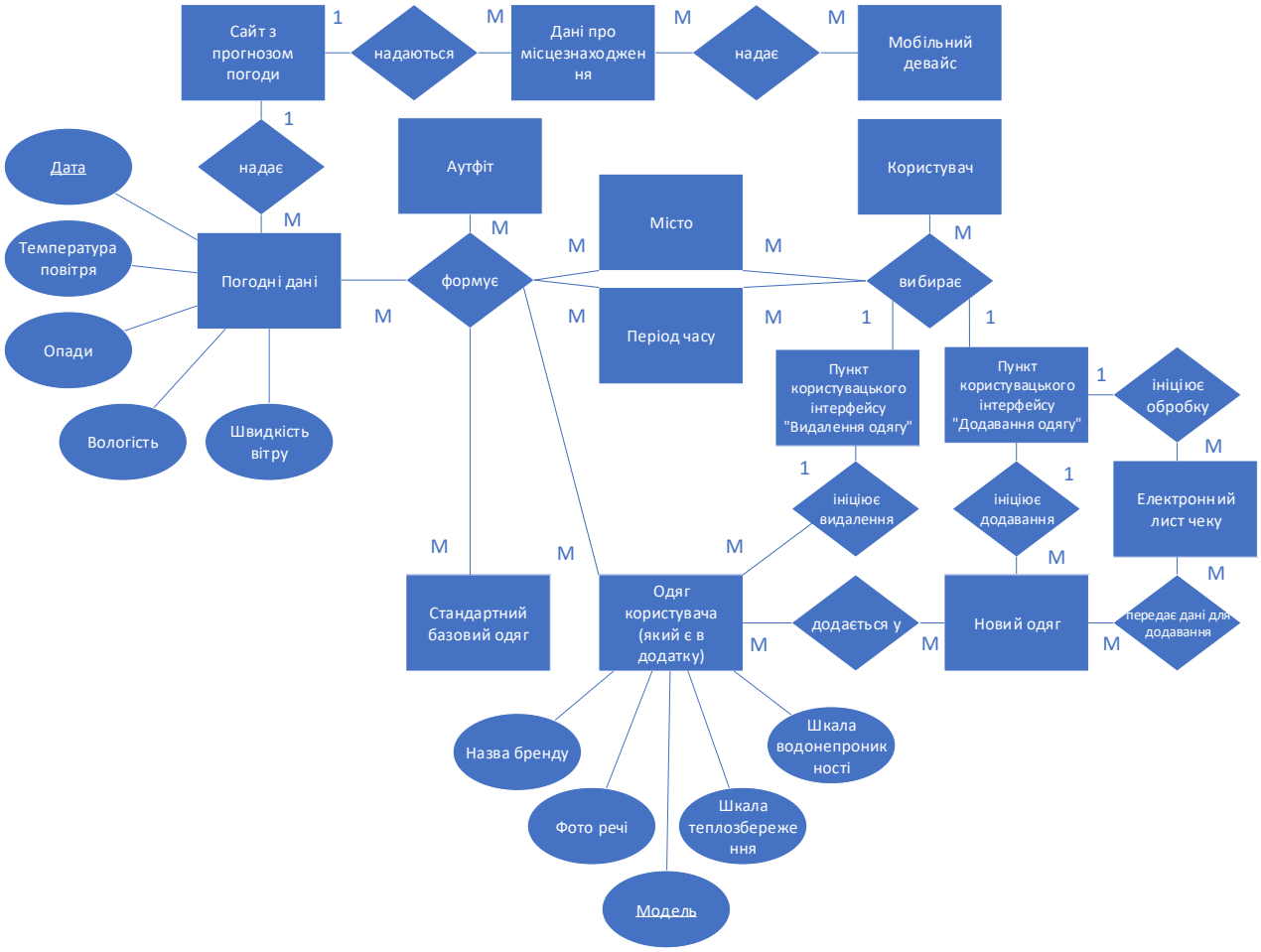


Рис. 1 Діаграма «Сутність – Зв'язок»

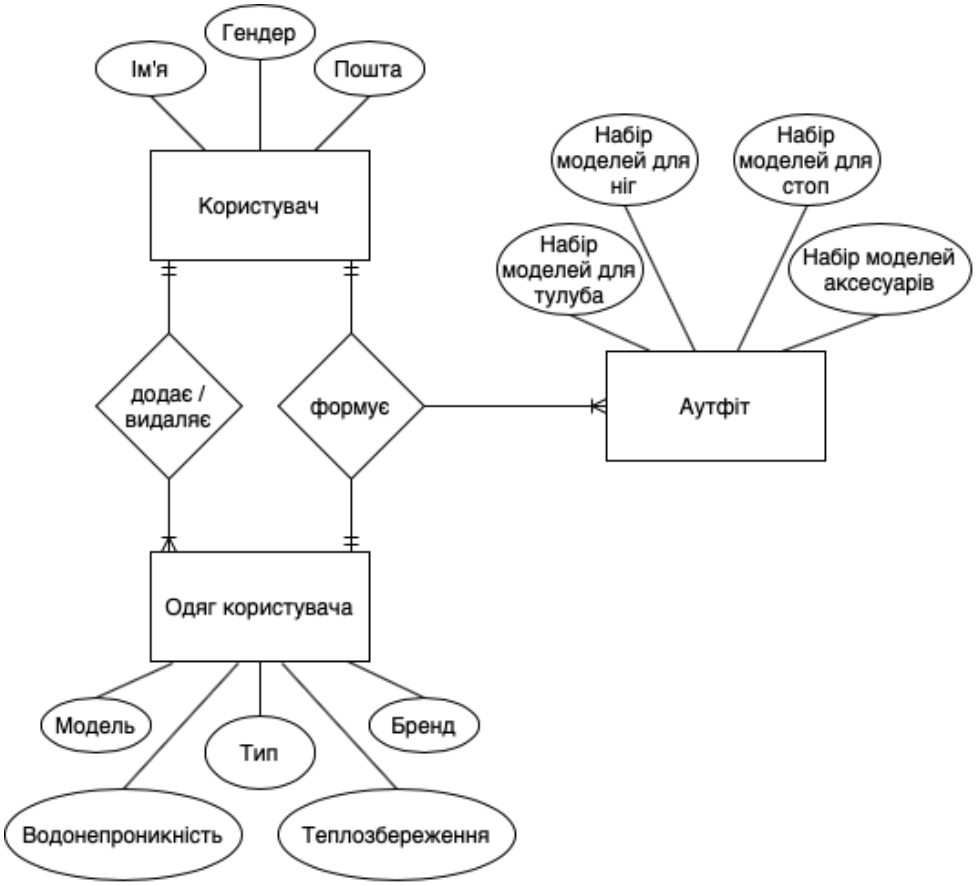


Рис. 2 Діаграма «Сутність – Зв'язок»

Таблиця 2.1. Перелік бізнес-процесів

№	Назва	Опис
1.	Завантаження нового одягу в БД	Користувач додатку в користувацькому інтерфейсі вибирає пункт додавання нового одягу. Вводить дані про новий одяг: вид одягу, назва бренду, моделі (якщо доступно), шкала теплозбереження одягу (1-5), шкала водонепроникності (1-3). При передачі моделі одягу, можлива загрузка фото з інтернету цієї речі. Додавання одягу в БД.
2.	Завантаження одягу в БД при покупці в інтернет-магазині	Користувач додатку після покупки одягу в інтернет-магазині додає у додаток електронний лист чеку з товарами, де лист обробляється на наявність одягу та додає його у БД. Якщо необхідно, користувач вручну додає відсутню інформацію щодо одягу.
3.	Підбір усього одягу при поточній погоді	Користувач додатку вибирає в користувацькому інтерфейсі пункт підбору одягу при поточній погоді. Додаток, знаючи поточне місцезнаходження девайсу, знаходить на погодному сайті з відкритим арі усі погодні дані, такі як температура повітря, опади, вологість, швидкість вітру і тд. Зпівставляючи поточну погоду та речі в наявності у користувача в БД, додаток формує аутфіт для користувача. Через недостачу персонального одягу у БД додатку, користувачу пропонується стандартні базові речі.
4.	Підбір усього одягу за погодою на вечір / весь день / ніч	Користувач додатку вибирає в користувацькому інтерфейсі пункт підбору одягу за погодою на вечір / весь день / ніч. Додаток, знаючи поточне місцезнаходження девайсу, знаходить на погодному сайті з відкритим арі усі погодні дані, такі як температура повітря, опади, вологість, швидкість вітру і тд на той період часу, що вибрав користувач (наприклад погода на всю ніч). Зпівставляючи прогноз погоди певного періоду часу та речі в наявності у користувача в БД, додаток формує аутфіт для користувача. Через недостачу персонального одягу у БД додатку,

№	Назва	Опис
		користувачу пропонується стандартні базові речі.
5.	Підбір одягу на декілька днів / тиждень	Користувач додатку вибирає в користувацькому інтерфейсі пункт підбору одягу за погодою на декілька днів / тиждень. Додаток, знаючи поточне місцезнаходження девайсу, знаходить на погодному сайті з відкритим арі усі погодні дані, такі як температура повітря, опади, вологість, швидкість вітру і тд на той період часу, що вибрав користувач (наприклад погода з п’ятниці по понеділок). Зпівставляючи прогноз погоди певного періоду часу та речі в наявності у користувача в БД, додаток формує декілька аутфітів для користувача. Через недостачу персонального одягу у БД додатку, користувачу пропонується стандартні базові речі.
6.	Підбір одягу в поїздку - за прогнозом погоди в іншому місті	Користувач додатку вибирає в користувацькому інтерфейсі пункт підбору одягу в поїздку - за прогнозом погоди в іншому місті. Вибравши місцезнаходження майбутньої поїздки, додаток, знаходить на погодному сайті з відкритим арі усі погодні дані, такі як температура повітря, опади, вологість, швидкість вітру і тд на той період часу, що вибрав користувач (наприклад погода у Мілані з 13 по 21 число). Зпівставляючи прогноз погоди у поїздці певного періоду часу та речі в наявності у користувача в БД, додаток формує декілька аутфітів для користувача. Через недостачу персонального одягу у БД додатку, користувачу пропонується стандартні базові речі.
7.	Підбір взуття при поточній погоді	Користувач додатку вибирає в користувацькому інтерфейсі пункт підбору взуття при поточній погоді. Додаток, знаючи поточне місцезнаходження девайсу, знаходить на погодному сайті з відкритим арі усі погодні дані, такі як температура повітря, опади, вологість, швидкість вітру і тд. Зпівставляючи поточну погоду та взуття в наявності у користувача в БД, додаток пропонує взуття для користувача. Через недостачу

№	Назва	Опис
		персонального взуття у БД додатку, користувачу пропонується стандартні базові варіанти взуття.
8.	Видалення одягу з БД	Користувач додатку в користувацькому інтерфейсі вибирає пункт видалення одягу. Вибирає зі своїх речей, які вже є в додатку, які необхідно видалити та підтверджує видалення. Одяг з БД видаляється.

2.3.2. Специфікація елементів даних

Специфікація елементів даних наведена у табл. 2.2 - табл. 2.5

Таблиця 2.2 - Специфікація елементів даних таблиці USER

№	Назва	Опис	Назва класу
1.	USER_ID	Унікальний ідентифікатор користувача	USER
2.	NAME	І’мя користувача	USER
3.	SEX	Гендер користувача	USER
4.	EMAIL	Електронна пошта користувача	USER

Таблиця 2.3 - Специфікація елементів даних таблиці THING

№	Назва	Опис	Назва класу
1.	USER_ID	Унікальний ідентифікатор користувача	THING
2.	TYPE	[1-4] по типу речі. 1 – одяг для тулуба користувача, 2 – одягу для ніг користувача, 3 – взуття користувача, 4 – аксесуар користувача.	THING
2.	MODEL	Модель одягу. Унікальний набір символів за яким можна знайти будь-яку річ. Це може бути як універсальна модель, як CD4216-110 S – можна	THING

№	Назва	Опис	Назва класу
		знайти в інтернеті. Так і набір символів заданий користувачем – e.g. «червона стильна куртка»	
3.	BRAND	Назва бренду речі	THING
4.	WARM	[1-5] по шкалі теплозбереження. Суб'єктивне оцінювання користувачем функції збереження тепла конкретного одягу за шкалою від 1 до 5, цифра. 1 – не зберігає тепло, 2 – зберігає тепло лише в теплу погоду, 3 - відносно зберігає тепло, 4 – зберігає тепло, 5 – значно зберігає тепло, можна носити зимою при непогоді	THING
5.	WET	[1-5] по шкалі водонепроникності. Суб'єктивне оцінювання користувачем функції непроникності води конкретного одягу за шкалою від 1 до 3, цифра. 1 – річ промокає при контакті з водою, вода затримується у речі, 2 – річ може промокати, але вода не затримується. Можна одягати при слабких опадах. 3 – вода не проникає у одяг чи одяг з особливими технологіями водонепроникності (Gore-Tex)	THING

Таблиця 2.4 - Специфікація елементів даних таблиці OUTFIT

№	Назва	Опис	Назва класу
1.	OUTFIT_ID	Унікальний ідентифікатор аутфіту під певний набір погодних параметрів	OUTFIT
2.	TOP_MODELS	Набір моделей одягу для тулуба користувача	OUTFIT
3.	BOTTOM_MODELS	Набір моделей одягу для ніг користувача	OUTFIT
4.	SHOEWEAR_MODELS	Набір моделей взуття користувача	OUTFIT

№	Назва	Опис	Назва класу
5.	ACCESSORIES_MODELS	Набір моделей аксесуарів користувача	OUTFIT

Таблиця 2.5 - Специфікація елементів даних таблиці ARCHIVE

№	Назва	Опис	Назва класу
1.	USER_ID	Унікальний ідентифікатор користувача	ARCHIVE
2.	NAME	Ім’я користувача	ARCHIVE
3.	SEX	Гендер користувача	ARCHIVE
3.	EMAIL	Електронна пошта користувача	ARCHIVE

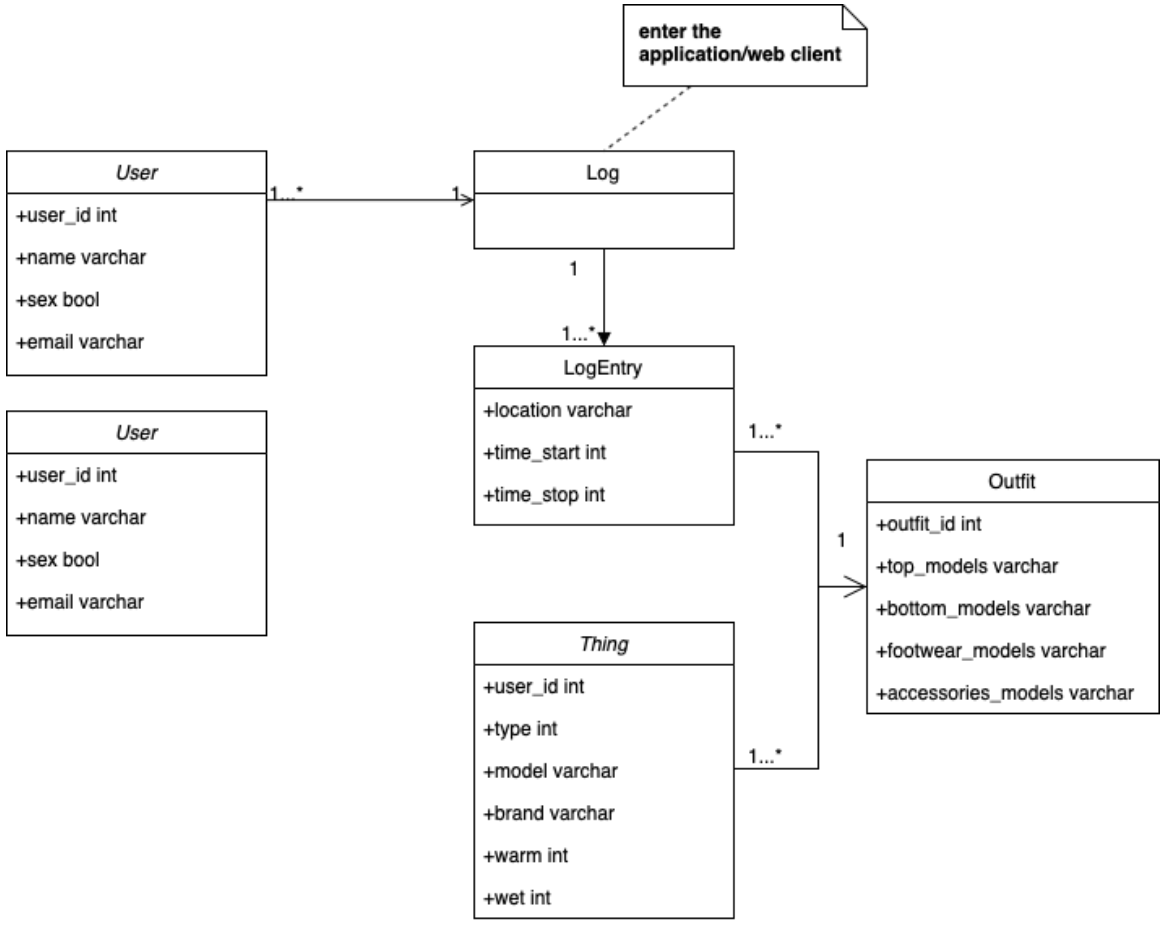


Рис. 3 Діаграма класів

2.4. Залежності елементів даних.

2.4.1. Функціональні залежності.

Перелік ФЗ надалі наводиться у вигляді таблиці (табл. 3).

Таблиця 2.6 Функціональні залежності

№	Сутність	Детермінан	Залежні атрибути	Словесне формулювання
1.	USER	USER_ID	NAME, SEX, EMAIL	Ідентифікатор користувача визначає його ім’я, гендер, адресу електронної пошти.
2.	THING	USER_ID	MODEL, TYPE, BRAND, WARM, WET	Ідентифікатор користувача визначає значення моделі, типу, бренду, теплозберігання, водонепроникності одягу.
3.	OUTFIT	OUTFIT_ID	TOP_MODELS, BOTTOM_MODELS, SHOEWEAR_MODELS, ACCESSORIES_MODELS	Ідентифікатор аутфіту визначає його набір моделей для тулуба, ніг та стоп користувача

Висновки з розділу 2

Дана інформаційна система може слугувати базою для ПЗ для порад щодо одягу. Після проведення аналізу було виділено 14 атрибутів, 3 сутності та 3 зав’язки між ними, побудовано 1 діаграму “Сутність-Зв'язок”. В результаті було створено 4 таблиці: USER, THING, OUTFIT, ARCHIVE.

3. СТВОРЕННЯ ТА ВИКОРИСТАННЯ ОБ'ЄКТІВ БАЗИ ДАНИХ

3.1. Таблиці.

Скрипти для створення та заповнення таблиць наведені у додатку (опис таблиць наведений у додатку В, а їх аргументи у додатку Г).

В результаті нормалізації було отримано 5 таблиць: «USER», «THING», «OUTFIT», «ARCHIVE».

Таблиця «USER» призначена для зберігання інформації про користувачів.

Таблиця «THING» призначена для зберігання інформації про речі користувача.

Таблиця «OUTFIT» призначена для зберігання інформації про отримані аутфіти.

Таблиця «ARCHIVE» призначена для зберігання інформації про видалених користувачів.

3.2 Обмеження цілісності

Нижче наведені приклади бізнес-правил:

1) Користувач не може не мати імені, гендеру, пошту.
2) Пошта повинна мати @ та "." у такому ж порядку.
3) Ідентифікатор користувачів не можуть повторюватись.
4) Дані гендеру, теплозберігання та водонепроникності не можуть бути від'ємні.
5) Дані теплозберігання та водонепроникності вимірюються числами від 1 до 5.
6) Дані типу одягу вимірюються числами від 1 до 4.
7) Гендер визначається як 1 – чоловік, 0 – жінка.
8) Аутфіт збирається за типами речей користувача.
9) У аутфіті повинно бути мінімум одна річ в кожному атрибуті. Виключення - це аксесуари.
10) Аксесуар не може бути у аутфіті ніде окрім у атрибуті ACCESSORIES_MODELS etc.

3.3. Тригери

Нижче наведено код створення одного з тригерів. Усі тригери будуть перелічені в Додатку Д.

1) Збереження видалених даних про користувачів (before delete)

```
DROP TRIGGER IF EXISTS DEL_SAVE;
CREATE TRIGGER DEL_SAVE BEFORE DELETE ON USER
FOR EACH ROW
BEGIN
INSERT INTO ARCHIVE (USER_ID, NAME, SEX, EMAIL)
VALUES (OLD.USER_ID, OLD.NAME, OLD.SEX, OLD.EMAIL);
END;
```

3.4. Типові вибірки. Представлення

Нижче наведено один приклад представлення. Інші знаходяться в Додатку Е.

Вивести інформацію про речі користувача (моделі, типи, бренди, теплозберігання, водонепроникності):

```
CREATE VIEW INFO_THING
AS
SELECT THING.MODEL, THING.TYPE, THING.BRAND, THING.WARM, THING.WET
FROM THING, USER
WHERE THING.USER_ID = USER.USER_ID
```

3.5. Індекси

Для підвищення продуктивності запитів, які на практиці використовувалися б найчастіше, були створені такі індекси:

```
CREATE INDEX User_Id_Index
ON USER(USER_ID);

CREATE INDEX Model_Type_Index
ON THING(MODEL, type);

CREATE INDEX Outfit_Id_Index
ON OUTFIT(OUTFIT_ID);
```

3.6. Типові оператори модифікації даних

Додавання нового користувача:

```
INSERT INTO "main"."USER"
("USER_ID", "NAME", "SEX", "EMAIL"
VALUES (12, 'Valeriy Vasiliev', 1, 'vvas@gmal.com');
```

Рис. 4 Дані користувача після додавання у БД

Зміна даних користувача:

```
UPDATE USER
SET NAME = 'Valeriy Vasiliev', SEX = 0, EMAIL = 'vvas@gmal.com'
WHERE USER_ID = 12
```

Рис. 5 Дані користувача після їх редагування

Видалення користувача:

```
DELETE FROM USER
WHERE USER_ID = 12;
```

Висновки з розділу 3

Обмеження цілісності запобігає введенню некоректних даних до таблиць та сприяє більшій стабільності системи. Створені представлення значно спрощують написання типових складних запитів. Створені в цьому розділі тригери, полегшують роботу програмного додатка, та реалізують базовий функціонал системи. Створені індекси підвищують ефективність запитів на великих обсягах даних.

ВИСНОВКИ

Було опрацьовано теоретичний матеріал для розуміння суті реляційних баз даних, їх застосування в предметній області та, безпосередньо, теоретичний матеріал за предметною областю. Було розроблено інформаційну систему для підбору одежі користувачу за погодними умовами.

В результаті створення системи були виконані поставлені задачі :

- Додати речі користувача у БД.
- Сортування речей по типам.
- Використати ці речі для побудови аутфітів.
- Вивести всі речі користувача.
- Збереження даних користувачів після видалення їх даних з основної БД.

Використано такі елементи СУБД як таблиці, представлення, тригери, індекси. Отриманий досвід роботи з СУБД SQLite та проектуванням діаграм, зокрема ERD-діаграм та діаграм «Сутність – зв’язок» та діаграм з нотації UML.

Дана система допоможе менше витрачати часу на вибір одягу перед виходом на вулицю та дозволить сфокусувати увагу людей на більш важливі речі.

На основі набутих протягом семестру знань було спроектовано та реалізовано базу даних, що складається із 4 таблиць, 5 представлень та 5 тригерів для опису сутностей.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Ульман Д., Уиндом Д. Введение в системы баз данных. — М.: Лори, 2000.

2. Вендров А. CASE-технологии. Современные методы и средства проектирования информационных систем. — М.: Финансы и статистика, 1998.

3. Конноли Т., Бегг К., Страчан А. А. Базы данных. Проектирование, реализация и сопровождение. — К.: Вильямс, 2000.

4. Дейт, К. Дж. Введение в системы баз данных. – М.: Вильямс, 2005.

5. Гаврилова и др. Базы знаний интеллектуальных систем // Учебник для вузов. — СПб.: Питер, 2000.

6. Сердаковський В.С., Корнієнко Г.А. Рекомендації студентам щодо вивчення кредитного модуля «Курсова робота» з навчальної дисципліни «Системи баз даних» (е-документ в КАМПУСІ).

7. «Мартин Грабер — Понимание SQL» [Підручник]

8. Посібник по SQLite [Електронний ресурс. –]https://www.sqlitetutorial.net/

9. "Database Concepts". [Електронний ресурс] – режим доступу: docs.oracle.com.

10. Системи бізнес-процесів даних [Електронний ресурс]. – Режим доступу до ресурсу: https://learn.ztu.edu.ua/mod/resource/view.php?id=9954

11. Целосность базы данных. [Електронний ресурс] – режим доступу: https://studfile.net/preview/6354061/page:55/

12. Язык запросов SQL [Електронний ресурс] – Режим доступу до ресурсу: https://sql-language.ru/create-view.html.

13. Кузнецов С. Д. Основы баз данных. Курс лекций. – М.: Интернетуниверситет информационных технологий, 2005.

14. [Bernstein et al., 1987] P. A. Bernstein, V. Hadzilacos, and N. Goodman. Concurrency Control and Recovery in Database Systems. – Reading, MA: Addison-Wesley, 1987.

15. Моделювання бізнес-процесів діяльності підприємства

16. [Електронний ресурс]. – 2016. – Режим доступу до ресурсу: http://www.economy.nayka.com.ua/?op=1&z=4950

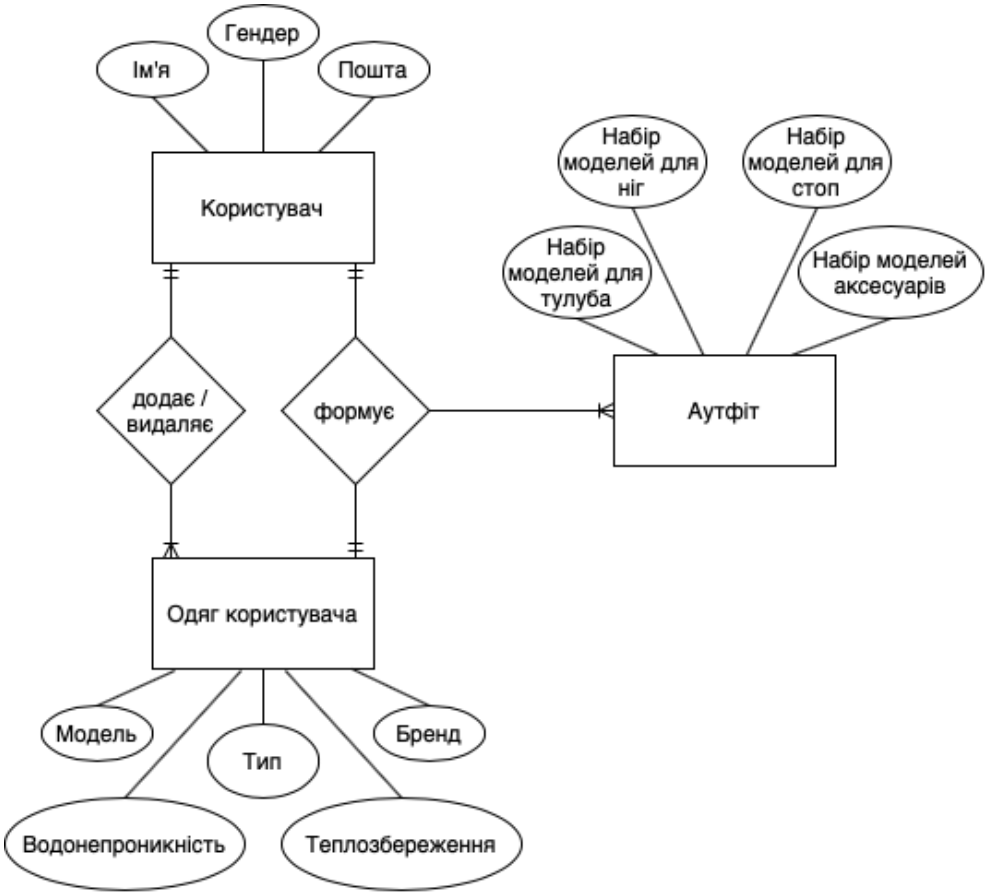
17. Что такое индексы базы данных (для начинающих)? [Електронний ресурс] – режим доступу: https://im-cloud.ru/blog/chto-takoe-indeksy-bazy-dannyh-dlja-nachinajushhih/

18. Організація баз даних та знань [Електронний ресурс]. – 2015. – Режим доступу до ресурсу: https://elearning.sumdu.edu.ua/free_content/lectured:89b3d175c06a6b137e410cb14821d0e94549ad5a/20151030211833/44700/index.html

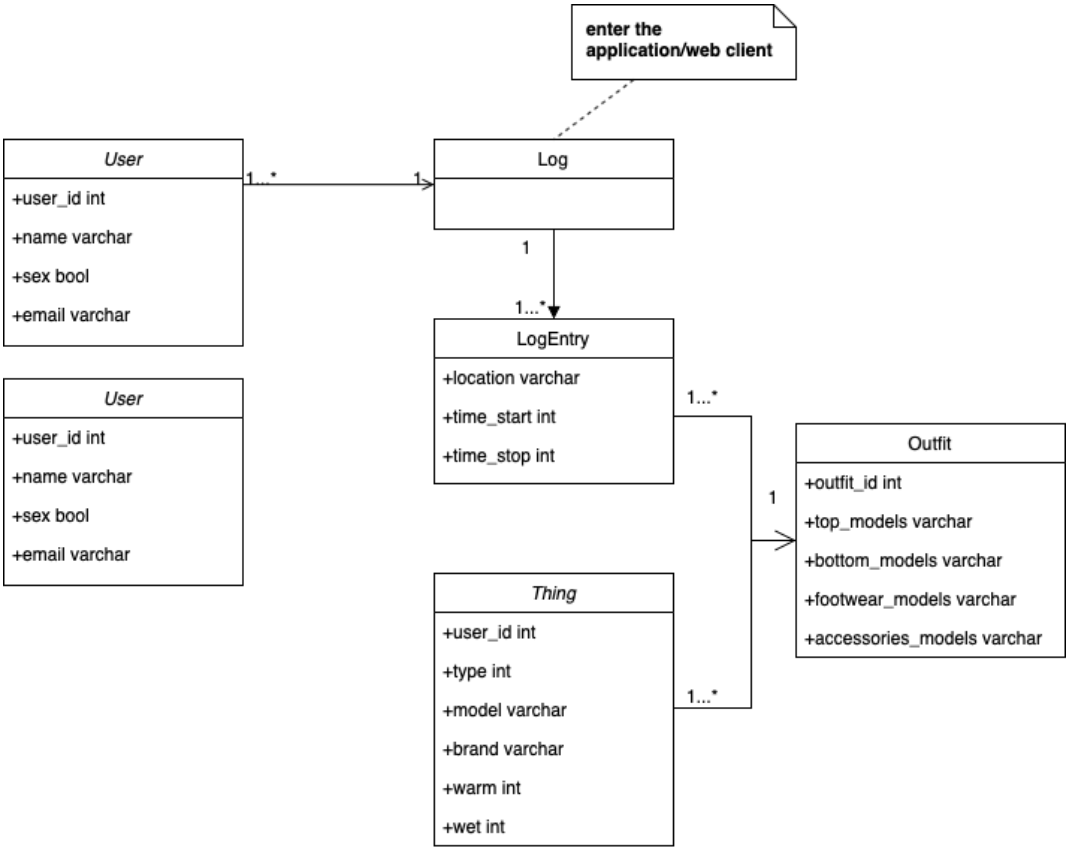
19. Кузин А. В. Базы данных / А. В. Кузин, С. В. Левонисова. – Москва, 2012. – 317 с.

ДОДАТКИ

ДОДАТОК А ДІАГРАМА «СУТНІСТЬ-ЗВ’ЯЗОК»



ДОДАТОК Б ДІАГРАМА КЛАСІВ ПРОЕКТУВАННЯ



ДОДАТОК В СТВОРЕННЯ ТАБЛИЦЬ

```
CREATE TABLE USER

(

USER_ID INT AUTO_INCREMENT PRIMARY KEY UNIQUE,

NAME VARCHAR(30) NOT NULL,

SEX BIT NOT NULL,

EMAIL VARCHAR(30) NOT NULL UNIQUE,

);

CREATE TABLE THING

(

USER_ID INT AUTO_INCREMENT,

MODEL VARCHAR(50) NOT NULL UNIQUE,

TYPE INT NOT NULL CHECK (TYPE >=1 AND TYPE <=4),

BRAND VARCHAR(30),

WARM INT NOT NULL CHECK (WARM >=1 AND WARM <=5),

WET INT NOT NULL CHECK (WET >=1 AND WET <=5),

FOREIGN KEY (USER_ID)

  REFERENCES USER(USER_ID)

  ON DELETE CASCADE

);

CREATE TABLE OUTFIT

(

OUTFIT_ID INT AUTO_INCREMENT PRIMARY KEY UNIQUE,

TOP_MODELS VARCHAR(200) NOT NULL,

BOTTOM_MODELS VARCHAR(200) NOT NULL,

SHOEWEAR_MODELS VARCHAR(200) NOT NULL,

ACCESORIES_MODELS VARCHAR(200),

);
```

```
CREATE TABLE ARCHIVE
(
USER_ID INT,
NAME VARCHAR(30),
SEX BIT,
EMAIL VARCHAR(30)
);
```

ДОДАТОК Г ЗАПОВНЕННЯ ТАБЛИЦЬ

Таблиця 2.7 - Продовження специфікації елементів даних таблиці USER

№	Назва атрибуту	Тип	Точність	Обмеження перевірки	Вхід ключа
1.	USER_ID	int	11	AI, UN	PK
2.	NAME	varchar	30	NN	-
3.	SEX	BIT	1	NN	-
4.	EMAIL	varchar	30	NN, UN	-

Таблиця 2.8 - Продовження специфікації елементів даних таблиці THING

№	Назва атрибуту	Тип	Точність	Обмеження перевірки	Вхід ключа
1.	USER_ID	int	11	AI, UN	FK
2.	MODEL	varchar	50	NN, UN	-
3.	TYPE	int	1	NN, CHECK	-
4.	BRAND	varchar	30	-	-
5.	WARM	int	1	NN, CHECK	-
6.	WET	int	1	NN, CHECK	-

Таблиця 2.9 - Продовження специфікації елементів даних таблиці DISTANCE

№	Назва атрибуту	Тип	Точність	Обмеження перевірки	Вхід ключа
1.	OUTFIT_ID	int	11	AI, UN	PK
2.	TOP_MODELS	varchar	200	NN	-
3.	BOTTOM_MODELS	varchar	200	NN	-
4.	SHOEWEAR_MODELS	varchar	200	NN	-
5.	ACCESORIES_MODELS	varchar	200	-	-

Таблиця 2.10 – Прод. Специфікації елементів даних таблиці ARCHIVE

№	Назва атрибуту	Тип	Точність	Обмеження перевірки	Вхід ключа
1.	USER_ID	int	11	-	-
2.	NAME	varchar	30	-	-
3.	SEX	BIT	1	-	-
4.	EMAIL	varchar	30	-	-

ДОДАТОК Д ТРИГЕРИ

1) Тригер на перевірку правильності введеної електронної пошти (before insert)

```
CREATE TRIGGER TRUE_EMAIL
  BEFORE INSERT ON USER
BEGIN
  SELECT
    CASE
   WHEN NEW.EMAIL NOT LIKE '%_@__%.__%' THEN
      RAISE (ABORT,'EMAIL IS NOT CORRECT')
     END;
END;
```

2) Трігер, який оновлює значення колонки NAME в таблиці USER через представлення INFO_USERS. (instead of update)

```
CREATE TRIGGER UPDATE_USER_INFO
INSTEAD OF UPDATE ON INFO_USERS
BEGIN
 UPDATE USER
 SET
  NAME = NEW.NAME
 WHERE USER_ID = OLD.USER_ID;
END;
```

3) Не можна додати або оновити ім’я, яке містить у своєму тілі цифру або символ.

```
CREATE TRIGGER OLNY_LETTERS
  BEFORE INSERT ON USER
BEGIN
  if (regexp_matches(NEW.NAME, '[a-zA-Z]*')) then return NEW;
else  raise exception 'NUMBER OR SYMBOLS ARE IN NAME'
 end if; END;
```

4) Збереження видалених даних (before delete)

```
DROP TRIGGER IF EXISTS ARCHIVE_SAVE;
CREATE TRIGGER DEL_SAVE BEFORE DELETE ON USER
```

```
FOR EACH ROW
BEGIN
INSERT INTO ARCHIVE (USER_ID, NAME, SEX, EMAIL)
VALUES (OLD.USER_ID, OLD.NAME, OLD.SEX, OLD.EMAIL);
END;
```

5) Тригер на видалення інформації про користувача (instead of delete)

```
CREATE TRIGGER DEL
INSTEAD OF DELETE
ON INFO_USERS
BEGIN
    DELETE FROM USER
        WHERE USER_ID = OLD.USER_ID;
END;
```

ДОДАТОК Е ПРЕДСТАВЛЕННЯ

1) Інформація про користувача

```
CREATE VIEW INFO_USERS
AS
SELECT USER.NAME, THING
FROM USER, THING
WHERE USER.USER_ID = THING.USER_ID
```

2) Користувачі, які вже додали одяг чи аксесуари

```
CREATE VIEW NO_THINGS
AS
SELECT USER.NAME
FROM THING, USER
WHERE USER.USER_ID = THING.USER_ID
```

3) Кількість речей кожного користувача

```
CREATE VIEW NUM_THINGS_BY_USER
AS
SELECT THING.USER_ID, THING.MODEL, COUNT(*) as THING.MODEL
FROM THING
GROUP BY MODEL
```

4) Загальна кількість користувачів

```
CREATE VIEW USERS
AS
SELECT COUNT (USER.USER_ID)
FROM USER
```

5) Весь одяг користувача

```
CREATE VIEW THINGS_BY_USER
AS
SELECT NAME
FROM USER
UNION
SELECT MODEL
FROM THING;
```