 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Лабораторна робота №3-4**

з дисципліни «Системи баз даних»

на тему: «Реляційна логічна модель ІС»

Варіант 7

**Виконав:**

студент гр. БС-03

Затуловський Г. А.

**Перевірив:**

ст.вик. [Сердаковський В. С.](https://schedule.kpi.ua/lecturers?lecturerId=f5dbaf77-ed72-47dd-bf2c-a135a78a44c8)

доц. [Лісневський Р. О.](https://schedule.kpi.ua/lecturers?lecturerId=78dc4be6-c994-41fb-bdf5-c5283e4a6467)

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

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

(підпис викладача)

Київ-2022

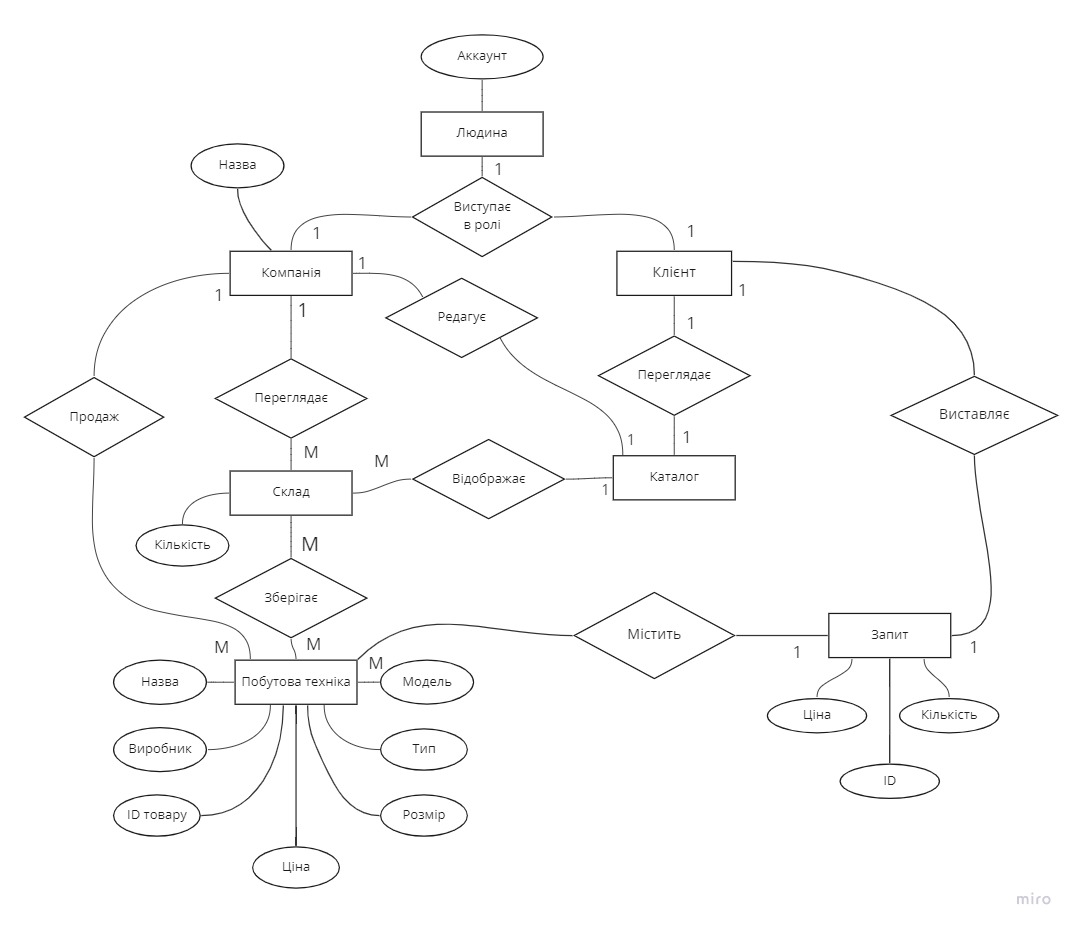
**Мета:** навчитися будувати реляційну модель та діаграму класів UML етапу логічного проектування інформаційної системи згідно обраної тематики

**Завдання:**

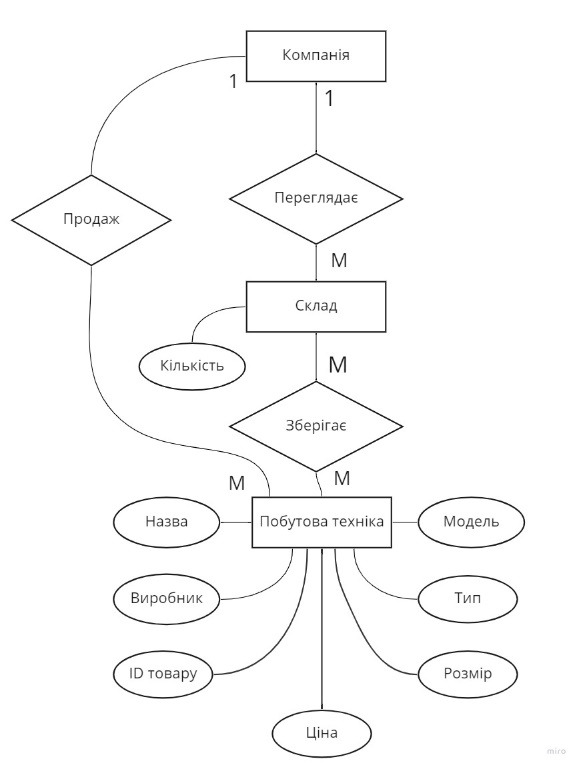
1. Згідно Індивідуальне завдання ("Розробити реляційну логічну модель ІС ...") відредагувати Діаграму "сутність-зв'язок" етапу концептуального проектування;
2. Діаграма "сутність-зв'язок" етапу концептуального проектування (фрагмент для прикладу процесу декомпозиції ДОДАТОК А);
3. Перелік та опис атрибутів;
4. Проведення декомпозиції (опис процесу);
5. Діаграма класів UML етапу логічного проектування; 6. Висновок до роботи.
6. Відповіді на контрольні питання.
7. Додайте файл з протоколом роботи в форматі будь-якого текстового процесора або PDF.

**Порядок виконання роботи**

**Відредагована діаграма "сутність-зв'язок" етапу концептуального проектування:**



**Відредагована діаграма "сутність-зв'язок" етапу концептуального проектування (фрагмент для прикладу процесу декомпозиції):**



**Перелік атрибутів:**

userName – ім’я аккаунту

userPassword – пароль аккаунту

userType – тип аккаунту

productName – назва продукту

productManufacture – виробник продукту

productID – ID продукту

productPrice – ціна продукту

productModel – модель продукту

productType – тип продукту

productSize – розмір продукту

quantityProduct – кількість продукту

orderPrice – ціна заказу

orderQuantity – кількість заказаного товару

orderID – ID заказу

**Опис атрибутів:**

Продукт, який має свою назву, виробника, ціну, модель, тип, розмір та кількість

productID → productName(1), productManufacture(2) , productPrice(3), productModel(4) , productType(5), productSize(6), quantityProduct(7)

Аккаунт має свій пароль та тип доступу

userName → userPassword(8), userType(9)

Кожен заказ містить в собі загальну ціну ордеру, номер товару та кількість покупляємої продукції т

orderID – productID(10), orderPrice(11), orderQuantity(12), productID(13)

**Проведення декомпозиції:**

R**(**userName, userType, productName, productManufacture, productID, productPrice, productModel, productType, productSize, quantityProduct, orderID, productID, orderPrice, orderQuantity**)**

Ключ userName= {userPassword, userType, productID, productName, productManufacture, productPrice, productModel, productType, productSize, quantityProduct, orderID , orderPrice, orderQuantity }

ФЗ 1-2 на R1

R1(userPassword, userType)

R2(productID, productName, productManufacture, productPrice, productModel, productType, productSize, quantityProduct, orderID , orderPrice, orderQuantity)

Ключ на R1: userName ФЗ 1-2

Ключ на R2: productName ФЗ 3-14

Ф3 3-11 на R2

R3(productName, productManufacture, productPrice, productModel, productType, productSize, quantityProduct)

R4(orderID, orderPrice, orderQuantity)

Ключ на R3: productID ФЗ 3-11

Ключ на R4: orderID ФЗ 12-14

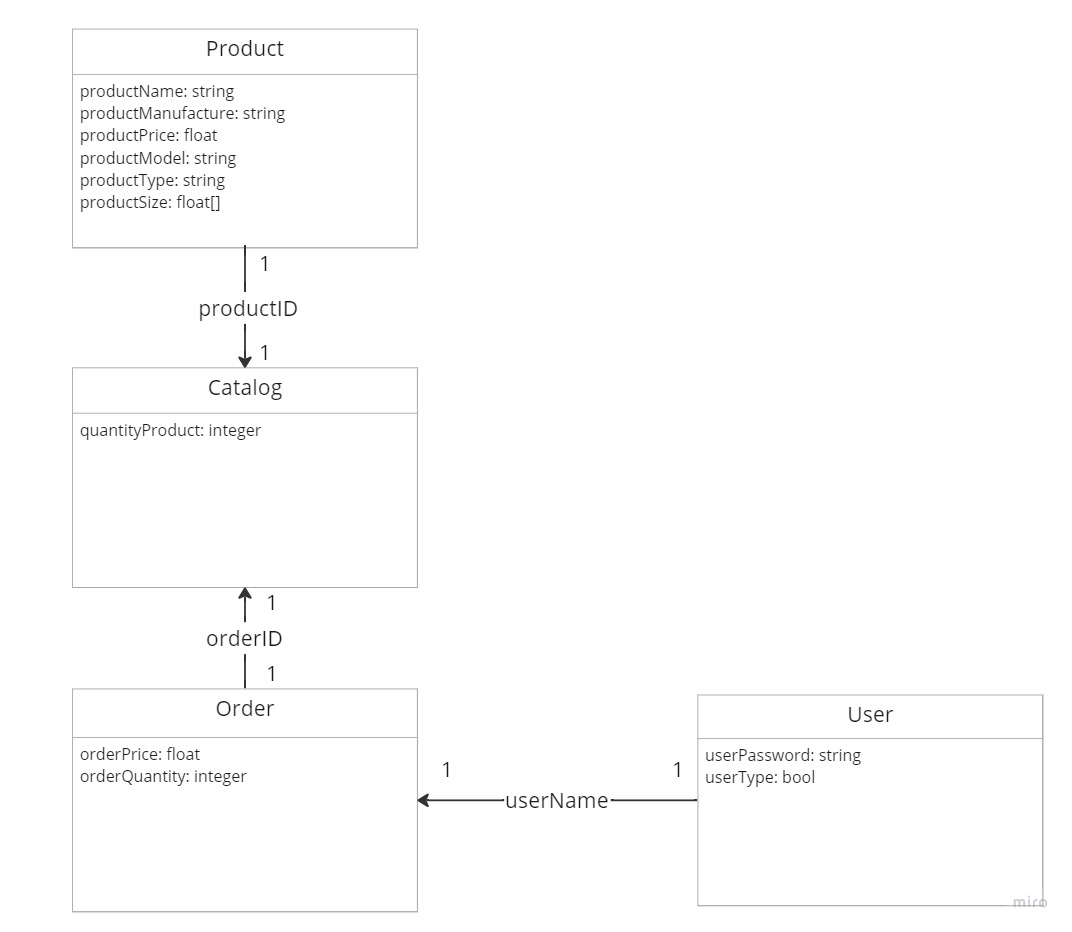
**Результати декомпозиції**

User (userName, userPassword, userType) – R1

Product(productID, productName, productManufacture, productPrice, productModel, productType, productSize, quantityProduct) – R3

OrderID(orderID, orderPrice, orderQuantity) – R5

**Діаграма класів UML етапу логічного проектування:**



**Висновок:** Ми навчилися будувати реляційну модель та діаграму класів UML етапу логічного проектування інформаційної системи згідно обраної тематики

**Відповіді на питання:**

1. **Відомості про функціональну залежність(ФЗ).**

ФЗ – бінарне відношення між множинами атрибутів даного відношення і є, по суті, зв’язком «один-до-багатьох». Функціональна залежність виконує роль основи для наукового підходу до розв’язання проблем.

1. **Відомості про багатозначну залежність(БЗ).**

БЗ – повне обмеження між двома множинами атрибутів у відношенні. На відміну від функціональної залежності, багатозначна залежність вимагає наявність певних кортежів у відношенні. Отже, багатозначна залежність це особливий випадок кортеж-твірної залежності. Поняття багатозначної залежності використовується при визначенні четвертої нормальної форми.

1. **Відомості про нормалізацію до – нормальної форми Бойса-Кодда (НФБК)**

НФБК – нормальна форма Бойса-Кодда використовуэться в нормалізації баз даних та являється трохи сильнішою версією третьої нормальної форми. Таблиця знаходиться в НФБК тоді і тілько тоді, коли для кожної її нетривіальної функціональної залежності X→Y, X це суперключ, тобто Х, або потенційний ключ, або його надмножина.

1. **Відомості про 1-2НФ.**

Хаотично створені таблиці даних не завжди відповідають першій нормальній формі (1НФ). Як відомо, приведення таблиці до першої нормальної форми є необхідною умовою нормалізації. Більш детально про необхідність застосування нормалізації в базі даних описується тут.

Відношення відповідає другій нормальній формі, якщо:

* таблиця приведена до першої нормальної форми 1НФ;
* у таблиці відсутні часткові залежності. Це означає, що кожен неключовий атрибут (поле) таблиці неприводимо залежить від первинного ключа таблиці (ключа відношення).

Друга нормальна форма має актуальність тільки для тих відношень, в яких є складені (композитні) ключі.

1. **Відомості про 3-4НФ.**

3НФ — нормальна форма використовна в [нормалізації баз даних](https://uk.wikipedia.org/wiki/%D0%9D%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%B1%D0%B0%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85" \o "Нормалізація баз даних).  За Коддом таблиця знаходиться в 3НФ [тоді й лише тоді](https://uk.wikipedia.org/wiki/%D0%A2%D0%BE%D0%B4%D1%96_%D0%B9_%D0%BB%D0%B8%D1%88%D0%B5_%D1%82%D0%BE%D0%B4%D1%96" \o "Тоді й лише тоді), коли виконуються наступні умови:

* [Відношення](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%BD%D1%8F_(%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C)) R (таблиця) знаходиться в [2НФ](https://uk.wikipedia.org/wiki/%D0%94%D1%80%D1%83%D0%B3%D0%B0_%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%84%D0%BE%D1%80%D0%BC%D0%B0)
* Кожен неключовий атрибут відношення R нетранзитивно (безпосередньо) залежить від кожного [потенційного ключа](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%B5%D0%BD%D1%86%D1%96%D0%B9%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BB%D1%8E%D1%87" \o ") в R.

4НФ – нормальна форма застосовна в нормалізації баз даних. Наступний рівень нормалізації після нормальної форми Бойса — Кодда. Тоді як друга, третя, і нормальна форма Бойса — Кодда опікуються функціональною залежністю, 4НФ опікується загальнішим типом залежності, відомим як багатозначна залежність. Таблиця перебуває в 4НФ тоді і тільки тоді, коли для кожної її багатозначної залежності X →→ Y, X — суперключ, тобто X або потенційний ключ, або його надмножина.

1. **Види ключів у реляційних базах даних ?**
2. первинний ключ
3. зовнішній ключ
4. простий та складений ключі
5. відношення, типи відношень
6. штучний та природній ключі
7. головна (master) та підлегла (detail) таблиці
8. **Умови цілісності даних у реляційних БД?**

У реляційній БД цілісність відношень визначається за допомогою первинного ключа, для якого мають виконуватися такі обмеження цілісності: атрибути первинного ключа не можуть містити NULL-значень; значення первинного ключа (як окремого атрибута або сукупності атрибутів) не можуть дублюватися в межах відношення.

1. **Сценарії цілісності даних у БД?**

Сценарії цілісності бази даних - це сценарії, які дають змогу уникнути введення некоректних даних у [БД](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), а також забезпечити можливість зв'язування декількох [таблиць](https://uk.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D1%8F_(%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)" \o "Таблиця (бази даних)). Ці правила можуть бути описані при створенні чи модифікації таблиці.

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

1. **Розкрити поняття методу нормалізації аналізу таблиць?**

**Метод нормалізації аналізу таблиць** – це процес (процедура) приведення таблиць бази даних до ряду нормальних форм (НФ) з метою уникнення надлишковості в базі даних, аномалій вставки, редагування та видалення даних. Таблиці можуть мати неефективну або не підходячу структуру, яку потрібно нормалізувати. Нормалізація передбачає розбиття вихідної таблиці (відношення) на декілька нових таблиць (відношень).

1. **Правила Кодда для реляційних систем БД?**

**Фундаментальне правило** (Foundation Rule)

Реляційна СКБД має бути здатною повністю керувати базою даних, використовуючи зв'язки між даними

**Інформаційне правило** (Information Rule)

Інформація має бути представлена у вигляді даних, що зберігаються в комірках. Дані, що зберігаються у комірках, мають бути атомарними. Порядок рядків у реляційній таблиці не повинен впливати на зміст даних і їх обробку.

**Правило гарантованого доступу** (Guaranteed Access Rule)

Доступ до даних має бути вільним від двозначності. До кожного елементу даних має бути гарантований доступ за допомогою комбінації імені таблиці, первинного ключа рядку й імені стовпця.

**Систематична обробка Null-значень** (Systematic Treatment of Null Values)

Невідомі значення NULL, відмінні від будь-якого відомого значення, мають підтримуватись для всіх типів даних при виконанні будь-яких операцій. Наприклад, для числових даних невідомі значення не повинні розглядатись як нулі, а для символьних даних — як порожні рядки.

**Правило доступу до системного каталогу на основі реляційної моделі** (Dynamic On-line Catalog Based on the Relational Model)

Словник даних має зберігатись у формі реляційних таблиць, і СКБД повинна підтримувати доступ до нього за допомогою стандартних мовних засобів, тих самих, що використовуються для роботи з реляційними таблицями, які містять дані користувача.

**Правило повноти підмови маніпулювання даними** (Comprehensive Data Sublanguage Rule)

Система управління реляційними базами даних має підтримувати хоча б одну реляційну мову, яка

а) має лінійний синтаксис,

б) може використовуватись інтерактивно і в прикладних програмах,

в) підтримує операції визначення даних, визначення уявлень, маніпулювання даними (інтерактивні та програмні), обмежувачі цілісності, управління доступом та операції управління транзакціями (begin, commit і rollback).

**Правило модифікації [розрізів](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D1%80%D1%96%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85" \o "Розріз даних)** (View Updating Rule)

Кожне подання має підтримувати усі операції маніпулювання даними, які підтримують реляційні таблиці: операції вибірки, вставки, модифікації і видалення даних.

**Правило високорівневих операцій модифікації даних** (High-level Insert, Update, and Delete)

Операції вставки, модифікації і видалення даних мають підтримуватись не тільки щодо одного рядку реляційної таблиці, але й щодо будь-якої безлічі рядків.

**Правило фізичної незалежності даних** (Physical Data Independence)

Додатки не повинні залежати від використовуваних способів зберігання даних на носіях, від апаратного забезпечення комп'ютерів, на яких знаходиться реляційна база даних.

**Правило логічної незалежності даних** (Logical Data Independence)

Представлення даних в додатку не повинно залежати від структури реляційних таблиць. Якщо в процесі нормалізації одна реляційна таблиця розділяється на дві, подання має забезпечити об'єднання цих даних, щоб зміна структури реляційних таблиць не позначалась на роботі додатків.

**Правило незалежності контролю цілісності** (Integrity Independence)

Вся інформація, необхідна для підтримки цілісності, має бути у словнику даних. Мова для роботи з даними має виконувати перевірку вхідних даних і автоматично підтримувати цілісність даних.

**Правило незалежності від розміщення** (Distribution Independence)

База даних може бути розподіленою, може перебувати на кількох комп'ютерах, і це не повинно впливати на додатки, як і перенесення бази даних на інший комп'ютер.

**Правило узгодженості мовних рівнів** (The Nonsubversion Rule)

Якщо використовується низькорівнева мова доступу до даних, вона не повинна ігнорувати правила безпеки і правила цілісності, які підтримуються мовою більш високого рівня.