МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДВНЗ «УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ

В. О. Нелюбов
Ю. Ю. Білак
А. М. Реблян

ПРОЄКТУВАННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Навчальний посібник в електронному вигляді

для студентів спеціальностей:
121 "Інженерія програмного забезпечення"
122 "Комп’ютерні науки та інформаційні технології"

Ужгород - 2023

УДК – 004.43

Проєктування реляційних баз даних: навчальний посібник в електронному вигляді / Укладачі: В.О. Нелюбов, Ю.Ю. Білак. А.М. Реблян. Ужгород: ДВНЗ «УжНУ», 2023. 60 с.

У посібнику досліджено аномалії, що є наслідком надлишкового дублювання даних та функціональних залежностей між атрибутами в таблицях. Ґрунтовно розглянуто проєктування баз даних методом нормальних форм та методом сутність-зв’язок. На прикладі досліджено технологію проєктування баз даних.

Рецензенти:

Мулеса Оксана Юріївна – доктор технічних наук, професор кафедри програмного забезпечення систем ДВНЗ «Ужгородський національний університет»;

Головач Йожеф Ігнацович – доктор технічних наук, професор кафедри математики та інформатики Закарпатського угорського інституту ім. Ференца Ракоці.

Затверджено на засіданні кафедри програмного забезпечення систем

як посібник з курсу «Проєктування баз даних і експертних систем», протокол №2 від 21 вересня 2023 р.

Рекомендовано до публікації Вченою радою факультету інформаційних технологій ДВНЗ «УжНУ», протокол №2 від 25 вересня 2023 р.

ЗМІСТ

Нелюбов

Володимир Олександрович

Кандидат технічних наук, заслужений професор УжНУ, професор кафедри програмного забезпечення систем.
Понад 130 наукових і методичних праць.
Наукова спеціалізація:
– мат. моделювання технічних систем;
– використання IT в навчанні.

Білак

Юрій Юрійович

Кандидат фізико-математичних наук, доцент, Завідувач кафедри програмного забезпечення систем,
Понад 90 наукових і методичних праць.
Сфери наукових інтересів: сучасні IT, фізика, техніко-дидактичні аспекти використання ІКТ у навчанні, дослідження ефективності сучасних методів навчання.

Реблян
Антонина Муратівна

Викладач кафедри програмного забезпечення систем.
Сфери наукових інтересів: сучасні IT, вища математика, проектування інформаційних систем, дослідження ефективності сучасних методів навчання.

ВСТУП

У загальному випадку база даних будь-якої предметної області являє собою сукупність зв’язаних між собою елементів: таблиць, запитів, макросів тощо. Зв’язки між елементами складні і приховані від користувача, тому йому важко керувати базою даних: вводити, модифікувати і вилучати дані, отримувати необхідну інформацію за допомогою запитів, виводити на друк звіти. Для ефективного керування базою даних створюється інтерфейс користувача, основу якого складають кнопкові та звичайні екранні форми. Поєднання бази даних з інтерфейсом користувача утворює додаток бази даних. Кнопкові і звичайні екранні форми – це об’єкти баз даних, на яких розміщуються елементи керування (кнопки, списки, перемикачі) і підписи до них. До елементів керування приєднують макроси або модулі VBA, що обумовлюють виконання певних дій: відкриття інших форм, друк звітів тощо.

В лабораторному практикумі на прикладі створення бази даних «Моя бібліотека» покроково розглядається технологія побудови інтерфейсу користувача в середовищі СУБД MS Access. Надаються практичні рекомендації щодо використання ефективних прийомів роботи. Лабораторні роботи є обов’язковою частиною виконання навчального плану з дисциплін «Проєктування баз даних і експертних систем» та «Організація баз даних і баз знань» для спеціальностей: 121 «Інженерія програмного забезпечення», 122 «Комп’ютерні науки та інформаційні технології».

ПРОЄКТУВАННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Розглядаються загальні питання проєктування реляційних баз даних. Дається визначення поняттю – проєкт реляційної БД. Наводяться етапи проєктування БД. Аналізуються види надмірностей у таблицях. Обговорюється надлишкове дублювання даних у таблицях та типи аномалій, до яких вони приводять. Наводиться приклад формування вихідної таблиці та її аналіз на наявність надмірностей.

Поняття проєкту реляційних баз даних

Будь-які штучні об’єкти неможливо якісно створити без проєкту. Безумовно це стосується також баз даних.

Проєкт реляційної бази даних – це набір взаємозалежних таблиць, у яких визначені всі поля, задані первинні ключі та інші властивості, які підтримують цілісність даних.

Відповідно до визначення, таблиці повинні бути взаємозалежними, тому що при виконанні запитів відбувається їхнє об'єднання за певними стовпцями. Тип даних по цих стовпцях повинен бути однаковим. Наприклад, якщо в одній таблиці оцінка позначається цифрами, а в іншій словами, наприклад, 5 і "відмінно", то неможливо об'єднати ці таблиці по стовпцю *Оцінка*.

Процес проєктування БД здійснюється у декілька етапів та є послідовністю переходів від неформального словесного опису інформаційної структури предметної області до формалізованого опису об'єктів предметної області в термінах деякої моделі.

На етапі системного аналізу проводиться докладний словесний опис предметної області і реальних зв'язків, що існують між об'єктами. Використовують два підходи:

1. ***Функціональний –*** реалізує принцип руху "від задачі". Застосовується, коли відомі функції певної групи осіб і комплекс задач, для інформаційних потреб яких створюється БД. У цьому випадку можна чітко виділити мінімальний набір об'єктів предметної області, яку необхідно описати.

2. ***Предметний –*** коли інформаційні потреби конкретних користувачів БД не фіксовані. Неможливо виділити весь набір об'єктів предметної області, які необхідно описати. У цьому випадку в опис включають найбільш характерні об'єкти і взаємозв'язки. Така БД може використовуватися для вирішення різноманітних, заздалегідь невизначених задач.

Рекомендується використовувати компромісний варіант, який орієнтований на конкретні задачі і потреби користувачів, а також дозволяє нарощувати новий функціонал.

Системний аналіз повинен закінчуватися докладним описом інформації про об'єкти, яка потрібна для вирішення конкретних задач і повинна зберігатися в БД. Формулюються задачі, які вирішуються за допомогою БД і даються короткі алгоритми їхнього вирішення. Описуються вихідні документи, що будуть генеруватися в системі, і вхідні документи для наповнення БД.

Проєктування баз даних здійснюється на фізичному і логічному рівнях. ***Проєктування на фізичному рівні*** багато в чому залежить від використовуваної СУБД, звичайно, автоматизовано і приховано від користувача.

Логічне проєктування полягає у визначенні числа і структури таблиць, формуванні запитів до БД, визначенні типів звітних документів, розробці алгоритмів обробки даних, створенні форм для введення і редагування даних у базі тощо.

При логічному проєктуванні БД найбільш важлива проблема – структуризація даних.

Класичний підхід до проєктування ***структур даних*** полягає в зборі даних про об'єкти предметної області в одній таблиці і подальшій декомпозиції її на кілька взаємозалежних таблиць з використанням процедури нормалізації. Надалі ми докладно будемо розглядати саме такий підхід.

Можливі й інші підходи до проєктування БД, наприклад CASE-системи (системи автоматизації проєктування і розробки БД).

Надлишкове дублювання даних і аномалії

Розрізняють просте (ненадлишкове) і надлишкове дублювання даних в таблицях. Просте дублювання допускається в БД. Надлишкове дублювання даних може приводити до проблем при їхній обробці.

R1

Співробітник	Телефон
Шпак	3003
Голуб	4004
Синиця	4004
Орлик	4004

Рисунок 1. Ненадлишкове дублювання.

Приклад ***ненадлишкового дублювання*** даних наведено на рис. 1 у таблиці **R1** з атрибутами *Співробітник* і *Телефон*. Для співробітників, що знаходяться в одному приміщенні, номери телефонів збігаються. Номер телефону 4004 зустрічається кілька разів, хоча для кожного співробітника номер телефону унікальний. Тому жоден з номерів не є надлишковим. При видаленні одного з номерів телефонів будуть загублені дані про те, за яким номером можна додзвонитися до конкретного службовця.

Приклад ***надлишкового дублювання (надмірності)*** наведений на рис. 2 у таблиці **R2**, яка доповнена атрибутом *№_кімн* (номер кімнати співробітника). Можна припустити, що всі співробітники в одній кімнаті мають однаковий номер телефону. Отже, у таблиці наявне надлишкове дублювання даних. Так, у зв'язку з тим, що *Синиця* та *Орлик* знаходяться в тій же кімнаті, що і *Голуб*, їх номер можна довідатися із запису даних про *Голуб*.

R2

Співробітник	Телефон	№_кімнати
Шпак	3003	109
Голуб	4004	111
Синиця	4004	111
Орлик	4004	111

R3

Співробітник	Телефон	№_комнати
Шпак	3003	109
Голуб	4004	111
Синиця	-	111
Орлик	-	111

Рисунок 2. Надлишкове дублювання.

На рис. 2 наведено приклад невдалої таблиці **R3**, у якій замість телефонів *Синиці* та *Орлика* поставлені прочерки (невизначені значення). По-перше, при програмуванні прийдеться перед-

бачити механізм обробки даних для прочерків у таблиці. По-друге, пам'ять усе рівно виділяється під значення поля з прочерками, хоча дублювання даних і виключено. По-третє, при звільненні *Голуба* запис з даними про нього буде виключений з таблиці і буде знищено номер телефону в 111-й кімнаті, що неприпустимо.

R4

Телефон	№_кімнати
3003	109
4004	111

R5

Співробітник	№_кімнати
Шпак	109
Голуб	111
Синиця	111
Орлик	111

Рисунок 3. Виключення надлишкового дублювання.

Вихід з цієї ситуації наведено на рис. 3, де показані дві таблиці: **R4** (містить дані про номери телефонів у кожній з кімнат) і **R5** (містить дані про номери кімнат, у яких розташовуються співробітники). Ці таблиці отримані шляхом декомпозиції вихідної таблиці **R3**. Якщо Голуба звільнять і видалять дані про нього з БД, це не приведе до втрати даних про номер телефону в 111-й кімнаті.

Процедура декомпозиції однієї таблиці на декілька таблиць є основною процедурою нормалізації таблиць при проєктуванні БД.

Надлишкове дублювання даних створює проблеми при обробці полів таблиць, які називають аномаліями відновлення таблиць. Ці проблеми виникають при спробі вилучення, додавання чи редагування стовпців.

Аномаліями називається така ситуація в таблицях, що приводить до протиріч у базі даних або істотно ускладнює обробку даних.

Є три види аномалій: *аномалії модифікації (чи редагування), аномалії вилучення та аномалії додавання.*

Аномалії модифікації проявляються в тому, що зміна значення одного даного може викликати перегляд усієї таблиці та відповідну зміну деяких інших записів таблиці.

Наприклад, зміна номера телефону в кімнаті 111 потребує перегляду всієї таблиці **R2** і зміни стовпця *Телефон* у записах, що відносяться до Голуба, Синиці та Орлика.

Аномалії вилучення полягають у тому, що при вилученні якого-небудь даного з таблиці може зникнути й інша інформація, що не зв'язана прямо з вилученим даним.

Наприклад, у таблиці **R2** вилучення запису про Шпака приводить до зникнення даних про номер телефону в 109-й кімнаті.

Аномалії додавання виникають у випадках, коли дані в таблицю не можна помістити доти, поки вона неповна, або вставка нового запису вимагає додаткового перегляду таблиці.

Наприклад, додавання нового співробітника в таблицю **R2**. Очевидно, буде протиприродним збереження даних у таблиці тільки про кімнату і номер телефону в ній, поки ніхто із співробітників не поміщений у ній. Якщо в таблиці **R2** поле *Співробітник* є ключовим, то збереження в ній неповних записів з відсутнім прізвищем службовця просто неприпустимо через невизначеність значення ключового поля.

Формування вихідної таблиці

Проєктування БД починається з визначення всіх об'єктів, зведення про які будуть включені в базу, та визначення їхніх атрибутів. Потім атрибути зводяться в одну вихідну таблицю.

Приклад. Формування вихідної таблиці.

Для навчальної частини факультету створюється БД. На першому етапі проєктування БД необхідно визначити зведення, які будуть зберігатися в базі і про те, як вона буде використовуватися, а також яку інформацію замовник (декан) хоче отримувати в процесі її експлуатації. У результаті встановлюються стовпці (поля), які повинні міститися в таблицях БД, і зв'язки між ними.

Імена полів (атрибутів) і їх короткі характеристики:

ПІБ – прізвище викладача, ім'я та по батькові;

Посада - займана посада;

Оклад - оклад викладача;

Стаж - викладацький стаж;

НСтаж - надбавка за стаж;

Каф - назва кафедри, на якій працює викладач;

Предм - назва предмета (дисципліни), що читається викладачем;

Група - номер групи, у якій викладач проводить заняття;

ВидЗан - вид занять, що проводяться викладачем у навчальній групі.

Усі поля таблиці повинні мати прості значення. Приклад вихідної таблиці **ДЕКАНАТ** наведений на рис. 4.

ТаблДЕКАНАТ

▼ПІБ	Посада	Оклад	Стаж	НСтаж	Кафедра	▼Предмет	▼Група	ВидЗанять
Шпак І. М.	викладач	10000	5	1000	ПЗС	СУБД	256	практика
Шпак І. М.	викладач	10000	5	1000	ПЗС	ПЛ/1	123	практика
Голуб М. І.	ст. викладач	12000	7	1200	ПЗС	СУБД	256	лекція
Голуб М. І.	ст. викладач	12000	7	1200	ПЗС	Паскаль	256	практика
Синиця Н. Г.	викладач	10000	10	1500	ПЗС	ПЛ/1	123	лекція
Синиця Н. Г.	викладач	10000	10	1500	ПЗС	Паскаль	256	лекція
Орлик В. В.	викладач	10000	5	1000	ІПЗ	ПЭВМ	244	лекція

Рисунок 4. Вихідна таблиця ДЕКАНАТ.

Таблиця має схему **ТаблДЕКАНАТ** (*ПІБ, Посада, Оклад, Стаж, НСтаж, Кафедра, Предмет, Група, ВидЗанять*). Ключ таблиці складовий (*ПІБ, Предмет, Група*). Атрибути, що входять до ключа, позначені трикутником та виділені кольором.

ТаблДЕКАНАТ містить надлишкове (надмірне) дублювання даних, яке буде причиною аномалій

редагування. Надмірність буває явною і неявною.

Явна надмірність - у таблиці **ТаблДЕКАНАТ** записи з даними про викладачів, які проводять заняття в різних групах, повторюються декілька разів. Наприклад, дані по Шпаку повторюються двічі. Якщо Шпак стане старшим викладачем, то його посаду необхідно змінити в обох рядках. Інакше буде протиріччя в даних, що є прикладом аномалії редагування, яка обумовлена явною надмірністю даних у таблиці.

Неявна надмірність у таблиці **ТаблДЕКАНАТ** виявляється в однакових окладах у всіх викладачів і в однакових добавках до окладу за однаковий стаж. Тому, якщо при зміні окладів з 10000 на 10500 це значення змінити у всіх викладачів, крім, наприклад, Орлика, то база стане суперечливою. Це приклад аномалії редагування для варіанту з неявною надмірністю.

Засобом виключення надмірності і, як наслідок, аномалій є нормалізація таблиць.

Висновки

1. База даних обов’язково має бути запроєктованою.

2. Проєкт реляційної БД є сукупність зв’язаних між собою таблиць.

3. Основною причиною необхідності проєктування БД є надмірне дублювання даних у вихідних таблицях, які обумовлені природою речей.

4. Надмірне дублювання даних приводить до виникнення аномалій, які викликають протиріччя у базі даних або істотно ускладнюють обробку даних.

5. Аномалії проявляються: при додаванні даних до таблиці, модифікації даних або видаленні даних з таблиці.

6. Усунення аномалій здійснюються в процесі нормалізації таблиць при проєктуванні БД.

7. Процедура нормалізації передбачає декомпозицію однієї таблиці з надлишковим дублюванням даних на декілька таблиць, в яких відсутнє надлишкове дублювання.

ДАТАЛОГІЧНЕ ПРОЄКТУВАННЯ

Розглядається даталогічне проєктування реляційних баз даних. Аналізуються функціональні залежності між атрибутами таблиць. На прикладі розглядається виявлення функціональних залежностей в таблицях, яке потрібне для проєктування БД.

В реляційних БД даталогічне (логічне проєктування) приводить до розробки схеми БД як сукупності таблиць, що моделюють об'єкти предметної області і семантичні зв'язки між ними.

Основна задача, що вирішується при проєктуванні БД - нормалізація таблиць. Класичним методом проєктування реляційних БД є метод нормальних форм, що заснований на понятті функціональної залежності (ФЗ) між атрибутами таблиць.

Залежності між атрибутами

Залежності визначають стійкі відносини між об'єктами і їх властивостями в певній предметній області.

Види залежностей: *функціональні, транзитивні і багатозначні*.

Функціональні залежності

Атрибут *B* ***функціонально залежить*** від атрибута *A*, якщо кожному значенню *A* відповідає значення *B*, позначається *A→B*. Це означає, що значення атрибута *A* в запису однозначно визначає значення атрибута *B* у цьому ж запису. *A* та *B* можуть бути складовими та складатися з двох і більше атрибутів.

У таблиці на рис. 4 можна виділити ФЗ між атрибутами *ПІБ→Каф*, *ПІБ→Посада*, *Посада→Оклад* та інші. *Наявність ФЗ у таблицях визначається природою речей*.

ПІБ	*НомерПаспорта*
Деркач П. І.	ПР 456344
Златко Т. П.	ПР 235872
Данько В. В.	ПР 562234
Квітка І. М.	КЕ 798865
Ніколенко К. Д.	НП 456783
Петричко М. І.	КЕ 367815
Сірко Н. Г.	НП 775681

Рисунок 5.

Якщо є ФЗ виду *A→B* и *B→A*, то між *A* і *B* існує ***функціональна взаємозалежність*** (взаємно однозначна відповідність), позначається *A↔B* або *B↔A*.

Приклад. Є таблиця з атрибутами, що функціонально залежать один від одного (рис. 5). Це номер паспорта (*НомерПаспорта*) і прізвище власника (*ПІБ*). Наявність ФЗ поля *ПІБ* від *НомерПаспорта* означає, що значення поля *НомерПаспорта* однозначно визначає значення поля *ПІБ*. У даному випадку діє і зворотна ФЗ: кожному значенню поля *ПІБ* відповідає тільки одне значення поля *НомерПаспорта*.

Часткова функціональна залежність - залежність неключового атрибута від частини складового ключа. У таблиці **ТаблДЕКАНАТ** атрибут *Посада* знаходиться у ФЗ від атрибута ПІБ, який є частиною ключа. Через це атрибут *Посада* знаходиться в частковій залежності від ключа таблиці.

Повна функціональна залежність – це залежність неключового атрибута від усього складеного ключа. Атрибут *ВидЗанять* знаходиться в повній ФЗ від складеного ключа.

Транзитивна залежність. Атрибут *C* залежить від атрибута *A* ***транзитивно,*** якщо для атрибутів *A, B, C* виконуються умови *A→B* і *B→C*, але зворотна залежність відсутня. У прикладі транзитивною залежністю зв'язані атрибути: *ПІБ→Посада→Оклад*.

Багатозначна залежність може існувати між атрибутами. *B* таблиці **R** атрибут *B* багатозначно залежить від атрибута *A*, якщо кожному значенню *A* відповідає безліч значень *B*, не зв'язаних з іншими атрибутами **R**.

Багатозначні залежності можуть бути "*один до багатьох*" (*1 : Б*), "*багато до одного*" (*Б : 1*) чи "*багато до багатьох*" (*Б : Б*), позначаються відповідно: *A⇒B, A⇐B* і *A⇔B*.

Приклад. Нехай викладач веде кілька предметів, а кожен предмет може вестися декількома викладачами, тоді має місце залежність *ПІБ⇔Предмет*. У прикладі викладач Шпак веде заняття з двох предметів (СУБД, ПЛ/1), а дисципліна СУБД читається двома викладачами (Шпак і Голуб).

Між двома атрибутами однієї таблиці можуть існувати залежності: 1:1, 1 : Б, Б : 1 і Б : Б. Оскільки залежність між атрибутами є причиною аномалій, то такі таблиці розбивають на декілька таблиць зі зв'язками виду 1:1, 1 : Б, Б : 1 і Б : Б. Зв'язки між таблицями відбивають залежності між атрибутами різних таблиць.

Атрибути називаються ***взаємно незалежними***, якщо кожний з них функціонально незалежний від іншого атрибута. Відсутність залежності атрибута *A* від атрибута *B* позначається *A¬→B*. Коли *A¬→*B і *B¬→A* то позначення *A¬=B*.

Виявлення залежностей між атрибутами

Виявлення залежностей між атрибутами необхідно для проєктування БД методом нормальних форм.

Основний спосіб установлення наявності ФЗ - уважний аналіз семантики атрибутів. Як правило, у таблиці існує декілька функціональних залежностей між атрибутами. Причому, якщо існує одна чи декілька функціональних залежностей, то можна вивести й інші функціональні залежності, існуючі в цій таблиці.

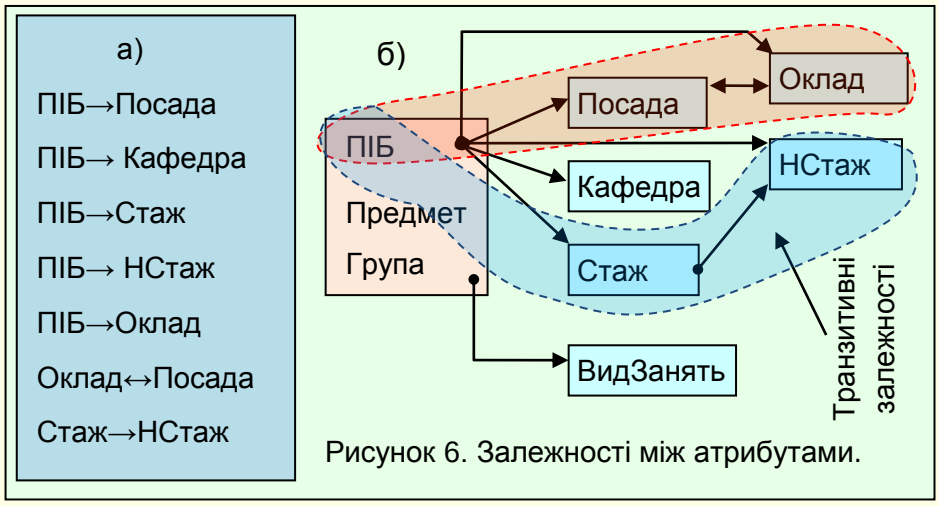


Рисунок 6. Залежності між атрибутами.

Виявимо залежності між атрибутами **ТаблДЕКАНАТ**. Врахуємо умову: один викладач в одній групі може проводити один вид занять (лекції чи практичні заняття).

У результаті аналізу таблиці одержуємо залежності між атрибутами (рис. 6).

До виділення цих ФЗ приводять наступні міркування:

- ПІБ у викладачів різні;
- кожен викладач має посаду (*викладач, ст. викладач*), але однакову посаду можуть займати декілька викладачів, тобто має місце ФЗ *ПІБ→Посада*. Зворотна ФЗ відсутня, тому що однакову посаду можуть займати різні викладачі;
- кожен викладач працює тільки на одній кафедрі - наявна ФЗ *ПІБ→Кафедра*. З іншого боку, на одній кафедрі багато викладачів, тому зворотної ФЗ немає;
- кожному викладачу відповідає його стаж, тобто має місце ФЗ *ПІБ→Стаж*. Зворотне невірно - однаковий стаж може бути в різних викладачів;
- кожен *викладач* має надбавку за стаж, тобто є ФЗ *ПІБ→НСтаж*. Зворотна ФЗ відсутня - однакову надбавку можуть мати кілька викладачів;
- кожному викладачу відповідає оклад, однаковий для всіх викладачів з однаковими посадами. Це враховується залежностями *ПІБ→Оклад* і *Посада→Оклад*. Немає однакових окладів для різних посад - ФЗ *Оклад→Посада*, тому існує функціональна взаємозалежність *Оклад↔Посада*;
- кожному стажу відповідає певна надбавка, тобто є ФЗ *Стаж→НСтаж*. Зворотне неправильно - одній надбавці може відповідати різний стаж роботи;
- можна виділити дві транзитивні залежності: *ПІБ→Посада→Оклад* і *ПІБ→ Стаж→НСтаж*;
- один викладач в одній групі з різних предметів може проводити різні види занять. Визначення виду занять, що проводить викладач, неможливе без вказівки на предмет і групу, тому має місце функціональна залежність ПІБ, Предмет, Група→ВидЗанять. Дійсно, викладач Голуб у 256-й групі читає лекції і проводить практичні заняття. Але лекції він читає з СУБД, а практику проводить з Паскалю.

Не були виділені залежності між атрибутами *ПІБ, Предмет* і *Група*, тому що вони утворюють складений ключ і не враховуються в процесі нормалізації вихідної таблиці.

Після виділення усіх ФЗ варто перевірити їхню погодженість з даними вихідної таблиці. Наприклад, *Посада=викладач* і Оклад=*10000* завжди відповідають один одному у всіх кортежах, тобто підтверджується функціональна залежність *Посада→Оклад*. Так само варто перевірити й інші ФЗ.

Висновки

1. В таблицях, що моделюють об’єкти предметної області, зазвичай наявні функціональні залежності.

2. Функціональні залежності бувають повними, частковими та транзитивними.

3. Виявлення функціональних залежностей необхідне для проєктування баз даних.

МЕТОД НОРМАЛЬНИХ ФОРМ

Розглядається проєктування реляційних баз даних методом нормальних форм. На прикладах аналізується переведення вихідної таблиці з надлишковим дублюванням даних у таблиці без надлишкового дублювання. Цей перевід здійснюється шляхом перетворення вихідної таблиці у першій нормальній формі в таблиці наступних нормальних форм.

Метод нормальних форм є класичним методом проєктування БД. Процес проєктування БД з використанням методу нормальних форм полягає в послідовному переведенні таблиць з першої нормальної форми в нормальні форми більш високого рівня. Кожна наступна нормальна форма обмежує визначений тип функціональних залежностей, усуває відповідні аномалії і зберігає властивості попередніх нормальних форм, тобто має кращі властивості.

Метод декомпозиції без втрат

Переведення таблиці в наступну нормальну форму здійснюється методом "декомпозиції без втрат". Така декомпозиція забезпечує те, що запити до вихідної таблиці і до таблиць, отриманих у результаті декомпозиції, дають однаковий результат.

Декомпозиція здійснюється за допомогою операції проекції реляційної алгебри. Наприклад, у таблиці **R**(*A, B, C, D, E*), *A, C* – ключ, існує часткова ФЗ *A*→*B*, яка є наслідком надлишкового дублювання даних (рис. 7). Усунення часткової ФЗ *A*→*B* дозволяє перевести її в наступну нормальну форму та позбавитися аномалій.

▼*A*	*B*	▼*C*	*D*	*E*
Шпак І. М.	викладач	СУБД	256	практика
Шпак І. М.	викладач	ПЛ/1	123	практика
Голуб М. І.	ст. викладач	СУБД	256	лекція
Голуб М. І.	ст. викладач	Паскаль	256	практика
Синиця Н. Г.	доцент	ПЛ/1	123	лекція
Синиця Н. Г.	доцент	Паскаль	256	лекція
Орлик В. В.	професор	ПЕОМ	244	лекція

Рисунок 7. Таблиця R. A → B часткова ФЗ.

Виконаємо декомпозицію таблиці **R** (Рис. 7) на дві нові таблиці **R1**(A, C, D, E. Рис. 8) і **R2**(A, B. Рис. 9). Таблиця **R2** є проекцією таблиці **R** на атрибути **A** і **B** (рис. 9). Таблиці **R1** та **R2** зв'язані по полю *A*.

▼*A*	▼*C*	*D*	*E*
Шпак І. М.	СУБД	256	практика
Шпак І. М.	ПЛ/1	123	практика
Голуб М. І.	СУБД	256	лекція
Голуб М. І.	Паскаль	256	практика
Синиця Н. Г.	ПЛ/1	123	лекція
Синиця Н. Г.	Паскаль	256	лекція
Орлик В. В.	ПЕОМ	244	лекція

Рисунок 8. Таблиця R1.

▼*A*	*B*
Шпак І. М.	викладач
Шпак І. М.	викладач
Голуб М. І.	ст. викладач
Голуб М. І.	ст. викладач
Синиця Н. Г.	доцент
Синиця Н. Г.	доцент
Орлик В. В.	професор

Рисунок 9. Таблиця R2.

Об'єднання даних з таблиць **R1** та **R2** дозволяє без втрат отримати такі ж дані, як, і у вихідній таблиці **R**. Але на відміну від таблиці **R,** в таблицях **R1** та **R2** відсутнє надмірне дублювання даних та пов'язані з ним аномалії.

Перша нормальна форма

Таблиця знаходиться в ***Першій нормальній формі*** (***1НФ***), якщо всі її атрибути є простими (мають єдине значення).

Вихідна таблиця будується так, щоб вона була в 1НФ (рис. 10).

ТаблДЕКАНАТ

▼ПІБ	Посада	Оклад	Стаж	НСтаж	Кафедра	▼Предмет	▼Група	ВидЗа
Шпак І. М.	викладач	10000	5	1000	ПЗС	СУБД	256	прак-
Шпак І. М.	викладач	10000	5	1000	ПЗС	ПЛ/1	123	прак-
Голуб М. І.	ст. викладач	12000	7	1200	ПЗС	СУБД	256	лекція
Голуб М. І.	ст. викладач	12000	7	1200	ПЗС	Паскаль	256	прак-
Синиця Н. Г.	викладач	10000	10	1500	ПЗС	ПЛ/1	123	лекція
Синиця Н. Г.	викладач	10000	10	1500	ПЗС	Паскаль	256	лекція
Орлик В. В.	викладач	10000	5	1000	ІПЗ	ПЭВМ	244	лекція

Рисунок 10. Вихідна таблиця.

Вихідна таблиця **ТаблДЕКАНАТ** має складений ключ (*ПІБ, Предмет, Група*) і знаходиться в 1НФ – всі атрибути прості. Відповідно до рис. 10 можна виділити часткову залежність атрибутів *ПІБ → Посада, ПІБ → Оклад ПІБ → Стаж, ПІБ → НСтаж, ПІБ → Кафедра* від частини ключа *ПІБ*.

Ця часткова залежність від ключа приводить:

1. до явного і неявного надлишкового дублювання даних, наприклад:
   - повторення зведень про стаж, посаду й оклад викладачів, які проводять заняття в декількох групах і / чи з різних предметів;
   - повторення зведень про оклади для однієї і тієї ж посади або про надбавки за однаковий стаж;
2. до проблем редагування даних; наприклад, зміна посади у викладача Шпака І. М. вимагає перегляд всіх кортежів відношення і внесення змін у ті з них, що містять зведення про цього викладача.

Частина надмірності усувається при переведенні відношення в Другу нормальну форму (2НФ).

Друга нормальна форма

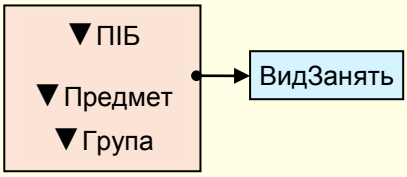
*Таблиця знаходиться в **другій нормальній формі** (**2НФ**), якщо вона знаходиться в 1НФ, і кожен неключовий атрибут функціонально повністю залежить від первинного ключа.*

Для усунення часткової залежності і переведення таблиці в 2НФ необхідно розкласти її на кілька таблиць так:

- побудувати проекцію без атрибутів, що знаходяться в частковій функціональній залежності від первинного ключа;
- побудувати проекції на частині складеного первинного ключа й атрибути, які залежать від цих частин.

ТаблНАВЧАННЯ

▼ПІБ	▼Предмет	▼Група	ВидЗанять
Шпак І. М.	СУБД	256	Практика
Шпак І. М.	ПЛ/1	123	Практика
Голуб М. І.	СУБД	256	Лекція
Голуб М. І.	Паскаль	256	Практика
Синиця Н. Г.	ПЛ/1	123	Лекція
Синиця Н. Г.	Паскаль	256	Лекція
Орлик В. В.	ПЭВМ	244	Лекція



Таблиця БД в 2НФ

R2

▼ПІБ	Посада	Оклад	Стаж	НСтаж	Каф
Шпак І. М.	викладач	10000	5	1000	ПЗС
Голуб М. І.	ст. викладач	12000	7	1200	ПЗС
Синиця Н.	викладач	10000	10	1500	ПЗС
Орлик В.	викладач	10000	5	1000	ІПЗ

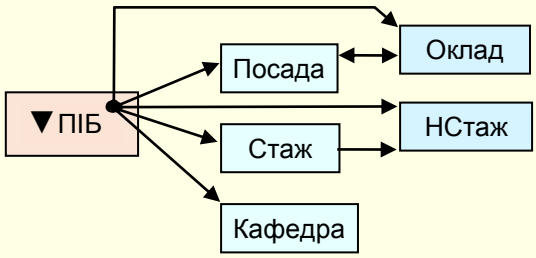


Рисунок 11.

Отримаємо дві таблиці **ТаблНАВЧАННЯ** і **R2**. Таблицю **ТаблНАВЧАННЯ** приведено до 2НФ (рис. 11). Первинний ключ таблиці **ТаблНАВЧАННЯ** є складовим і складається з атрибутів *ПІБ, Предмет, Група*. Цей ключ у таблиці **ТаблНАВЧАННЯ** отримано у припущенні, що кожен викладач в одній групі з одного предмета може або читати лекції, або проводити практичні заняття. У таблиці *R2* ключ атрибут *ПІБ* є частиною ключа.

Дослідження таблиць **ТаблНАВЧАННЯ** і **R2** показує, що приведення таблиці **ТаблНАВЧАННЯ** до 2НФ дозволив виключити явну надмірність даних - повторення рядків зі зведеннями про викладачів. У таблиці **R2,** як і раніше, існує неявне дублювання даних.

Для подальшого удосконалення таблиць **R2** необхідно перевести в Третю нормальну форму (3НФ).

Третя нормальна форма

Визначення 1. Таблиця знаходиться в ***третій нормальній формі*** (***3НФ***), якщо вона знаходиться в 2НФ і кожен неключовий атрибут нетранзитивно залежить від первинного ключа.

Визначення 2. Таблиця знаходиться в *третій нормальній формі* (***3НФ***) у тому випадку, якщо всі неключові атрибути таблиці взаємно незалежні і цілком залежать від первинного ключа.

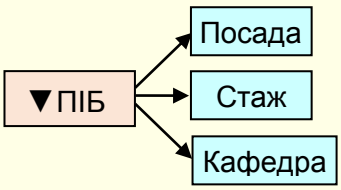
Те, що неключові атрибути цілком залежать від первинного ключа, означає, що таблиця знаходиться в 2НФ. Взаємна незалежність атрибутів означає відсутність усякої залежності між атрибутами таблиці, у тому числі і транзитивної. Отже, друге визначення 3НФ зводиться до першого визначення.

Якщо в таблиці **ТаблНАВЧАННЯ** (Рис. 11) транзитивні залежності відсутні, то в таблиці **R2** вони є: *ПІБ→Посада→Оклад*, *ПІБ→Оклад→Посада*, *ПІБ→Стаж→НСтаж*.

Транзитивні залежності також породжують надлишкове дублювання даних. Усунемо їх, використовуючи операцію проекції на атрибути, що є причиною транзитивних залежностей. Одержимо таблиці **ТаблВИКЛАДАЧ**, **ТаблПОСАДА** і **ТаблСТАЖ**, які знаходяться в 3НФ (Рис. 12, а). Графічно ці таблиці представлені на рис. 12, б.

б)

▼ПІБ	Посада	Стаж	Кафедра
Шпак І. М.	викладач	5	ПЗС
Голуб М. І.	ст. викладач	7	ПЗС
Синиця Н. Г.	викладач	10	ПЗС
Орлик В. В.	викладач	5	ІПЗ

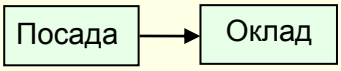


ТаблВИКЛАДАЧ

Таблиця БД у 3НФ

ТаблПОСАДА

▼Посада	Оклад
викладач	10000
ст. викл	12000



ТаблСТАЖ

▼Стаж	НСтаж
5	1000
7	1200
10	1500

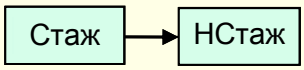


Рисунок. 12.

На практиці побудова 3НФ схем таблиць у більшості випадків є достатнім, і на цьому процес проєктування реляційної БД закінчується. Дійсно, приведення таблиць до 3НФ у нашому прикладі привело до усунення надлишкового дублювання.

Якщо у таблиці є залежність атрибутів складового ключа від неключових атрибутів, то необхідно перейти до посиленої 3НФ.

Посилена 3НФ чи нормальна форма Бойса-Кодда (БКНФ)

Таблиця знаходиться в ***нормальній формі Бойса-Кодда*** (БКНФ), якщо вона знаходиться в 3НФ і в ній відсутні залежності ключів (атрибутів складеного ключа) від неключових атрибутів.

У прикладі, що розглядається, подібної залежності немає, тому процес проєктування на цьому закінчується. Результатом проєктування є схема БД (Рис. 13), яка складається зі зв'язаних таблиць: **ТаблНАВЧАННЯ**, **ТаблВИКЛАДАЧ**, **ТаблПОСАДА**, **ТаблСТАЖ**.

В отриманої БД є необхідне дублювання даних, але відсутнє надлишкове.

Таблиця **ТаблНАВЧАННЯ** отримана в результаті переведення вихідної таблиці **ТаблДЕКАНАТ** у другу нормальну форму.

Таблиця **ТаблВИКЛАДАЧ** отримана в результаті переведення таблиці **R2** у третю нормальну

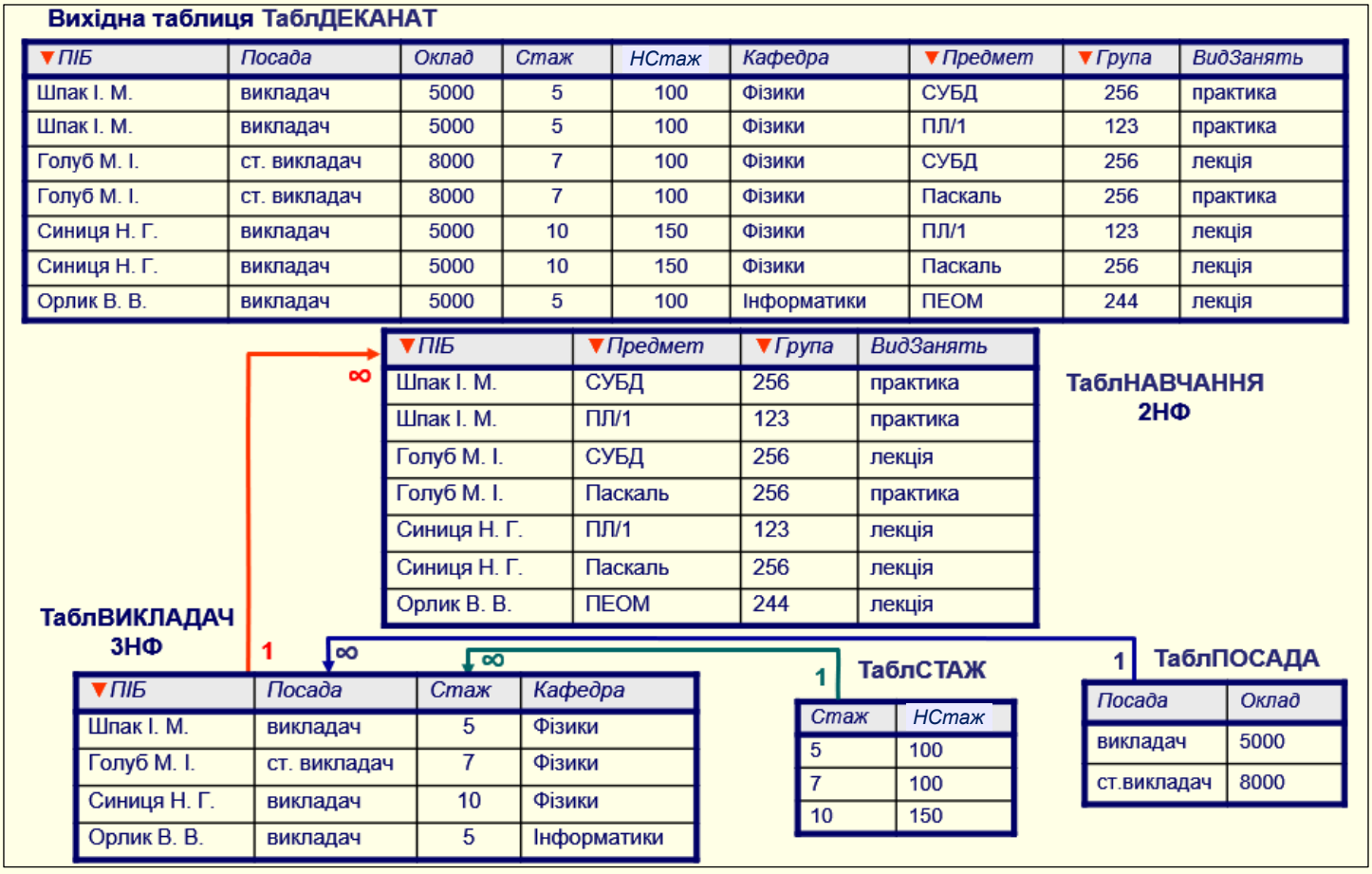
форму.

Таблиці **ТаблПОСАДА** та **ТаблСТАЖ** утворилися в результаті нормалізації таблиці **R2**. У них відсутнє надлишкове дублювання даних і, як наслідок, аномалії. Тому вони не потребують нормалізації.

У всіх таблицях мають бути ключові поля, які є ідентифікаторами рядків та за допомогою яких таблиці зв'язуються між собою. Тому в таблиці **ТаблПОСАДА** ключовим визначено поле *Посада*, а в таблиці **ТаблСТАЖ** – поле *Стаж.*

На Рис. 13 наведено схему бази даних для предметної області ДЕКАНАТ, яка отримана в результаті проєктування методом нормальних форм. Проєкт БД – це сукупність зв'язаних між собою нормалізованих таблиць. У цих таблицях, у порівнянні з вихідною таблицею **ТаблДЕКАНАТ**, відсутнє надлишкове дублювання даних та притаманні їм аномалії.

Звертаємо увагу на те, що кожна з нормалізованих таблиць утримує дані про певний об'єкт предметної області.



Вихідна таблиця ТаблДЕКАНАТ

ПІБ	Посада	Оклад	Стаж	НСтаж	Кафедра	Предмет	Група	ВидЗанять
Шпак І. М.	викладач	5000	5	100	Фізики	СУБД	256	практика
Шпак І. М.	викладач	5000	5	100	Фізики	ПЛ/1	123	практика
Голуб М. І.	ст. викладач	8000	7	100	Фізики	СУБД	256	лекція
Голуб М. І.	ст. викладач	8000	7	100	Фізики	Паскаль	256	практика
Синиця Н. Г.	викладач	5000	10	150	Фізики	ПЛ/1	123	лекція
Синиця Н. Г.	викладач	5000	10	150	Фізики	Паскаль	256	лекція
Орлик В. В.	викладач	5000	5	100	Інформатики	ПЕОМ	244	лекція

ТаблНАВЧАННЯ 2НФ

ПІБ	Предмет	Група	ВидЗанять
Шпак І. М.	СУБД	256	практика
Шпак І. М.	ПЛ/1	123	практика
Голуб М. І.	СУБД	256	лекція
Голуб М. І.	Паскаль	256	практика
Синиця Н. Г.	ПЛ/1	123	лекція
Синиця Н. Г.	Паскаль	256	лекція
Орлик В. В.	ПЕОМ	244	лекція

ТаблВИКЛАДАЧ 3НФ

ПІБ	Посада	Стаж	Кафедра
Шпак І. М.	викладач	5	Фізики
Голуб М. І.	ст. викладач	7	Фізики
Синиця Н. Г.	викладач	10	Фізики
Орлик В. В.	викладач	5	Інформатики

ТаблСТАЖ

Стаж	НСтаж
5	100
7	100
10	150

ТаблПОСАДА

Посада	Оклад
викладач	5000
ст.викладач	8000

Рисунок 13. Нормалізовані таблиці.

У більшості випадків для нормалізації таблиць з надлишковим дублюванням даних достатньо перевести їх у третю нормальну форм або у нормальну форму Бойса-Кодда. Але бувають випадки, коли цього недостатньо.

НомерПроєкту	КодСпівробітника	Задання
001	05	1
001	05	2
001	05	3
004	02	1
004	02	2
004	03	1
004	03	2
004	05	1
004	05	2
007	06	1

Четверта нормальна форма

Таблиця **ПРОЄКТИ** має схему: **ПРОЄКТИ** (*НомерПроєкту, КодСпівробітника, Завдання*). У таблиці містяться номери проєктів, для кожного проєкту - список кодів співробітників-виконавців і список завдань, які передбачені кожним проєктом. Співробітники можуть працювати в декількох проєктах, і різні проєкти можуть містити однакові завдання. Передбачається, що кожен співробітник, який бере участь у певному проєкті, виконує всі завдання.

Отже, єдиним ключем таблиці є складений атрибут *НомерПроєкту, КодСпівробітника, Завдання*. Він є первинним ключем таблиці. Звідси випливає, що таблиця **ПРОЄКТИ** знаходиться у формі БКНФ.

Головний недолік таблиці **ПРОЄКТИ** в тому, що при підключенні/відстороненні від проєкту співробітника приходиться додавати/виключати з таблиці стільки рядків, скільки завдань є в проєкті. Внесення чи виключення у таблицю одного факту про певного співробітника вимагає серії операцій через дублювання значень у записах.

Непряма ознака аномалії - дублювання інформації в таблиці. Припустимо, що причиною аномалії є наявність деякої залежності між атрибутами таблиці.

Дійсно, у відношенні **ПРОЄКТИ** існують дві багатозначні залежності:

НомерПроєкту⇒КодСпівробітника; НомерПроєкту⇒Завдання.

У довільній таблиці **R**(*A, B, C*) може одночасно існувати багатозначна залежність *A⇒B* і *A⇒C*. Цю обставину позначимо як *A⇒B|C*.

Подальша нормалізація таблиць, схожих з таблицею **ПРОЄКТИ**, ґрунтується на теоремі Фейджіна.

Таблицю R(A, B, C) можна спроєктувати без втрат у таблицю R1(A, B) і R2(A, C) у тому випадку, якщо існує залежність A⇒B | C.

Під проєктуванням без втрат розуміється такий спосіб декомпозиції таблиць, при якому вихідна таблиця цілком і без надмірності відновлюється шляхом природного з'єднання отриманих таблиць.

R

A	B	C
К	15	1
К	15	2
Л	10	1
М	20	1
М	20	2
М	20	3

R1

A	B
К	15
Л	10
М	20

R2

A	C
К	1
К	2
Л	1
М	1
М	2
М	3

Приклад. Є таблиця **R**(*A, B, C*). Побудуємо проєкції **R1** і **R2** на атрибути *A*, *B* і *A*, *C* відповідно.

Результатом операції з'єднання таблиць **R1**(*A, B*) і **R2**(*A, C*) за атрибутом *A* є таблиця з атрибутами *A*, *B* і *C*, рядки якої виходять у результаті зв'язування таблиць **R1** і **R2** по типу 1:Б на основі збігу значень атрибута *A*.

Зв'язування рядків (*к, 15*) і {(*к, 1*), (*к, 2*)} дає рядки {(*к, 15,1*), (*к, 15,2*)}.

Зв'язування **R1**(*A, B*) і **R2**(*A, C*) породжує вихідну таблицю **R**(*A, B, C*). У таблиці **R** немає зайвих рядків та втрат.

Визначення четвертої нормальної форми.

Таблиця R знаходиться ***в четвертій нормальній формі*** *(4НФ)* у тому і тільки в тому випадку, коли існує багатозначна залежність A⇒B, а всі інші атрибути R функціонально залежать від A.

ПРОЄКТИ-СПІВРОБІТНИКИ

НомерПроєкту	КодСпівробітника
001	05
004	02
004	03
004	05
007	06

ПРОЄКТИ-ЗАВДАННЯ

НомерПроєкту	Завдання
001	1
001	2
001	3
004	1
004	2
007	1

Таблицю **ПРОЄКТИ** можна представити у вигляді двох таблиць: **ПРОЄКТИ-СПІВРОБІТНИКИ** (*НомерПроєкту, КодСпівробітника*). Первинний ключ таблиці: *НомерПроєкту*, КодСпівробітника; ПРОЄКТИ-ЗАВДАННЯ.

СПІВРОБІТНИКИ-ВІДДІЛИ-ПРОЄКТИ

Код Співробітника	Код Відділу	Номер Проєкту
01	РД	036
02	АД	004
03	УП	004
04	АД	019
05	ЛС	001
05	ЛС	004
06	УП	007
08	ВЦ	013
09	ВЦ	014
10	СЖ	013

Ці таблиці знаходяться в 4НФ, і дублювання значень атрибутів кодів співробітників зникло. Можна з'єднати ці таблиці і переконатися, що утвориться таблиця **ПРОЄКТИ**.

Не всі таблиці можна відновити до вихідної. У даному випадку відновлення можливе тому, що кожен співробітник, який бере участь у певному проєкті, виконує усі завдання з цього проєкту (саме це укладається в принцип 1: Б з'єднання таблиць). Самі ж співробітники брали участь в декількох проєктах, і різні проєкти могли містити однакові завдання.

П’ята нормальна форма

Результатом нормалізації попередніх схем таблиць були дві нові таблиці. Іноді це зробити не вдається, або таблиці, що виходять, мають небажані властивості. У цьому випадку виконується декомпозиція вихідної таблиці більш ніж на дві таблиці.

Розглянемо таблицю **СПІВРОБІТНИКИ-ВІДДІЛИ-ПРОЄКТИ** (*КодСпівробітника, КодВідділу, НомерПроєкту*). Первинний ключ складений і включає всі атрибути: *КодСпівробітника, КодВідділу і НомерПроєкту*. Припустимо, що один співробітник може працювати в декількох відділах, причому в кожному відділі він може брати участь у кількох проєктах. В одному відділі можуть працювати кілька співробітників, але кожний проєкт виконує тільки один співробітник. Функціональних і багатозначних залежностей між атрибутами не існує.

За структурою таблиці можна констатувати, що вона знаходиться у формі 4НФ. Проте у таблиці можуть бути аномалії, зв'язані з можливістю повторення значень атрибутів у декількох рядках. Наприклад, співробітник може працювати в декількох відділах, при його звільненні це вимагає пошуку і наступного вилучення з вихідної таблиці кількох записів.

Визначення залежності з'єднання.

Таблиця R(X, Y, ... , Z) задовольняє ***залежності з'єднання***; її позначимо *(X, Y,..., Z), у тому випадку, якщо R відновлюється без втрат шляхом з'єднання своїх проекцій на X, Y, ... , Z. Залежність з'єднання є узагальненням функціональної і багатозначної залежностей.

Визначення п'ятої нормальної форми.

Таблиця R знаходиться ***в п'ятій нормальній формі*** *(5НФ або нормальній формі проекції-з'єднання - PJ/NF) у тому випадку, коли будь-яка залежність з'єднання в R випливає з існування певного можливого ключа в R.*

Утворимо складені атрибути таблиці **СПІВРОБІТНИКИ-ВІДДІЛИ-ПРОЄКТИ**:

СО={ *КодСпівробітника, КодВідділу*};

СП={ *КодСпівробітника, НомерПроєкту*};

ОП={ КодВідділу, НомерПроєкту}.

Покажемо, що якщо таблицю **СПІВРОБІТНИКИ-ВІДДІЛИ-ПРОЄКТИ** спроєктувати на складені атрибути *З, СП* і *ОП*, то з'єднання цих проєкцій дає вихідну таблицю. Це означає, що в таблиці існувала залежність з'єднання *(*З, СП, ОП*). Проєкції на складені атрибути назвемо *СПІВРОБІТНИКИ-ВІДДІЛИ*, *СПІВРОБІТНИКИ-ПРОЄКТИ* і *ВІДДІЛИ-ПРОЄКТИ.*

СПІВРОБІТНИКИ-ВІДДІЛИ

Код_ Співробітника	Код_ Відділу
01	РД
02	АД
03	УП
04	АД
05	ЛС
05	ЛС
06	УП
08	ВЦ
09	ВЦ
10	СЖ

СПІВРОБІТНИКИ-ПРОЄКТИ

Код_ Співробітника	Номер_ Проєкту
01	036
02	004
03	004
04	019
05	001
05	004
06	007
08	013
09	014
10	013

ВІДДІЛИ-ПРОЄКТИ

Код_ Відділу	Номер_ проєкту
РД	036
АД	004
УП	004
АД	019
ЛС	001
ЛС	004
УП	007
ВЦ	013
ВЦ	014
СЖ	013

Раніше виконувалося з'єднання двох проєкцій і відразу виходив шуканий результат. Для відновлення таблиці з трьох або більш проєкцій потрібно отримати усі попарні з'єднання, над якими потім виконати операцію перетинання. Попарні з'єднання трьох наведених вище таблиць мають вигляд, який наведений нижче.

Неважко побачити, що перетинання усіх таблиць дає вихідну таблицю **СПІВРОБІТНИКИ-ВІДДІЛИ-ПРОЄКТИ**.

Таблиці **СПІВРОБІТНИКИ-ВІДДІЛИ**, **СПІВРОБІТНИКИ-ПРОЄКТИ** і **ВІДДІЛИ-ПРОЄКТИ** знаходяться в 5НФ. Вона є останньою відомою формою. Умови її одержання складні, і тому вона майже не використовується на практиці. Крім цього, вона має визначені недоліки.

Припустимо, що необхідно довідатися, де і які проєкти виконує співробітник з кодом 02. Для цього у таблиці СПІВРОБІТНИКИ-ВІДДІЛИ знайдемо, що співробітник з кодом 02 працює у відділі АД, а з таблиці ВІДДІЛИ-ПРОЄКТИ знайдемо, що у відділі АД виконуються проєкти 004 і 019. А це значить, що співробітник 02 повинний виконувати проєкти 004 і 019. На жаль, інформація про те, що співробітник з кодом 02 виконує проєкт 019, помилкова (див. вихідну таблицю СПІВРОБІТНИКИ-ВІДДІЛИ-ПРОЄКТИ).

Такі протиріччя можна усунути тільки шляхом спільного розгляду всіх проекцій основної таблиці. Саме тому після з'єднання проекцій і виконувалася операція їхнього перетинання. Безумовно, це недолік. Відзначимо, що наявність недоліків не вимагає обов'язкового відмовлення від визначених видів нормальних форм. Треба враховувати недоліки й умови їхнього прояву. У деяких постановках задач недоліки не виявляються.

***(СО, СП)**

Код_Співробітника	Код_Відділу	Номер Проєкту
01	РД	036
02	АД	004
03	УП	004
04	АД	019
05	ЛС	001
05	ЛС	004
06	УП	007
08	ВЦ	013
09	ВЦ	014
10	СЖ	013

***(СО, ОП)**

Код_Співробітника	Код_Відділу	Номер_Проєкту
01	РД	036
02	АД	004
02	АД	019
03	УП	004
03	УП	007
04	АД	004
04	АД	019
05	ЛС	001
05	ЛС	004
06	УП	004
06	УП	007
08	ВЦ	013
08	ВЦ	014
09	ВЦ	013
09	ВЦ	014
10	СЖ	013

***(СП, ОП)**

Код_Співробітника	Код_Відділу	Номер Проєкту
01	РД	036
02	АД	004
02	ЛС	004
02	УП	004
03	АД	004
03	ЛС	004
03	УП	004
04	АД	019
05	ЛС	001
05	АД	004
05	ЛС	004
05	УП	004
06	УП	004
06	УП	007
08	ВЦ	013
08	ВЦ	014
09	ВЦ	013
09	ВЦ	014
10	СЖ	013

На практиці звичайно обмежуються структурою БД, що відповідає 3НФ чи БКНФ. Тому процес нормалізації таблиць методом нормальних форм передбачає послідовне усунення з вихідної таблиці міжатрибутних залежностей:

- часткових залежностей неключових полів від ключа (вимоги 2НФ);
- транзитивних залежностей неключових полів від ключа (вимоги 3НФ);
- залежності ключів (полів складених ключів) від неключових полів (вимоги БКНФ).

Метод нормальних форм застосовують для проєктування невеликих БД. При проєктуванні великих БД застосовують метод ER-діаграм (метод "Сутність-зв'язок"). Однак на останньому етапі методу ER-діаграм таблиці перевіряються на приналежність їх до БКНФ. Цей етап виконується з використанням методу нормальних форм. Після завершення проєктування створюється БД за допомогою СУБД.

Рекомендації з розробки структури баз даних

При проєктуванні бази даних методом нормальних форм спочатку створюється загальна вихідна таблиця, в яку вносяться дані про предметну область. Розмір таблиці (кількість стовпців) залежить від розміру предметної області та детальності, з якою її необхідно змоделювати.

У більшості випадків вихідна таблиця утримує надлишкове дублювання даних і притаманні їм аномалії. Тому вихідна таблиця потребує нормалізації. Для цього в таблиці необхідно виявити функціональні залежності між атрибутами (стовпцями). Зрозуміло, що чим більше атрибутів у таблиці, тим складніше виявити залежності між ними. Тому метод нормальних форм зазвичай використовують для проєктування невеликих баз даних.

У результаті проєктування утворюється сукупність зв’язаних нормалізованих таблиць (Рис.13). Кожна з цих таблиць утримує дані для моделювання певних об’єктів (сутностей) предметної області. Тому проєктування можна почати не зі створення загальної таблиці, а відразу таблиць для кожного об’єкта БД. Зрозуміло, що таблиці для окремих об’єктів будуть значно меншими, ніж загальна вихідна таблиця, тому нормалізувати їх буде значно простіше. Суттєво підвищити ефективність такого підходу можна, дотримуючись певних правил і рекомендацій, які наводяться нижче.

Якими повинні бути таблиці сутностей?

Під сутністю розуміється будь-який об’єкт предметної області, для якої створюється база даних.

Основне правило: ***кожній сутності - окрему таблицю.***

Стовпці таблиць сутностей можуть бути ключовими і неключовими. Введення ключів у таблицю дозволяє забезпечити унікальність значень у записах таблиці за ключем, прискорити обробку записів і виконати автоматичне сортування записів за значеннями в ключових полях.

Звичайно досить визначити простий ключ, рідше - вводять складений ключ. Наприклад, таблиця збереження списку співробітників (прізвище, ім'я та по-батькові), у якій зустрічаються однофамільці, повинна мати складений ключ. У деяких СУБД користувачам пропонується визначити автоматично створюване ключове поле нумерації (у Access - це поле типу "лічильник"), що спрощує рішення проблеми унікальності записів таблиці.

Іноді в таблицях сутностей є поля опису властивостей або характеристик об'єктів. Якщо в таблиці є значне число повторень і ці дані мають великий обсяг, то краще їх виділити в окрему таблицю. Тим більше, варто утворити додаткову таблицю, якщо властивості взаємозалежні.

Рекомендація: дані про сутності варто представити таким чином, щоб неключові поля в таблицях були взаємно незалежними і цілком залежали від ключа (див. визначення третьої нормальної форми).

При обробці таблиць сутностей треба мати на увазі, що нову сутність легко додати і змінити, але при її вилученні варто знищити всі посилання на неї з таблиць зв'язків, інакше таблиці зв'язків будуть некоректними. Багато сучасних СУБД блокують некоректні дії в подібних випадках.

Організація зв'язку сутностей

Записи *таблиці зв'язків* призначені для відображення зв'язків між сутностями, дані про які знаходяться у відповідних таблицях сутностей. Таблиці зв’язків створюють для зв’язування таблиць, які не мають однакових стовпців (полів), за якими їх зв’язують у звичайний спосіб.

Найчастіше одна *таблиця зв'язків* описує взаємозв'язок двох сутностей. Звичайно таблиці сутностей мають по одному ключовому полю, тоді *таблиця зв'язків* двох таблиць для забезпечення унікальності записів про зв'язки повинна мати складений ключ, який складається з ключів цих двох таблиць. Можна створити *таблицю зв'язків* і без ключів, але тоді функції контролю унікальності записів лягають на користувача.

Складні зв'язки (не бінарні) варто зводити до бінарних. Для опису взаємозв'язків **N** об'єктів потрібно **N -1** таблиць зв'язків. Транзитивних зв'язків не повинно бути. Надлишок зв'язків приводить до протиріч.

Не слід включати в *таблиці зв'язків* характеристики сутностей, інакше неминучі аномалії, їх краще зберігати в окремих таблицях. При роботі з *таблицями зв'язків* варто мати на увазі, що будь-який запис з *таблиці зв'язків* може бути вилучений, оскільки сутності можуть обійтися і без зв'язків. При додаванні чи зміні вмісту записів *таблиці зв'язків* треба контролювати правильність посилань на існуючі об'єкти, тому що зв'язок без об'єктів існувати не може. Більшість сучасних СУБД контролюють правильність посилань на об'єкти.

Забезпечення цілісності

Під цілісністю розуміють властивість бази даних містити повну, несуперечливу й адекватно відбиваючу предметну область інформацію.

Розрізняють *фізичну і логічну цілісність*.

Фізична цілісність означає наявність фізичного доступу до даних і те, що дані не втрачені.

Логічна цілісність означає відсутність логічних помилок у базі даних, до яких відносяться порушення структури БД чи її об'єктів, вилучення або зміна зв'язків між об'єктами тощо.

Підтримка логічної цілісності БД включає контроль цілісності і її відновлення при виявленні протиріч у базі. Цілісний стан БД задається за допомогою обмежень цілісності у вигляді умов, які повинні задовольняти дані в базі.

Серед обмежень цілісності можна виділити два основних типи обмежень: обмеження значень полів таблиць і структурні обмеження на записи таблиць.

Прикладом ***обмежень значень*** полів таблиць є вимога неприпустимості порожніх чи повторюваних значень у полях, а також контроль приналежності значень полів заданому діапазону. Так наприклад, у записах таблиці про кадри значення поля *Дата_Народження* не можуть перевищувати значення поля *Дата_Пийому*.

Найбільш гнучким засобом реалізації контролю значень полів є збережені процедури і тригери, наявні в деяких СУБД.

Структурні обмеження визначають вимоги цілісності сутностей і цілісності посилань. Кожному екземпляру сутності, представленому в таблиці, відповідає тільки один запис. Вимога цілісності сутностей полягає в тому, що кожен запис таблиці повинен відрізнятися від іншого запису цієї таблиці, тобто будь-яка таблиця повинна мати первинний ключ.

Вимога цілісності посилань пов'язана з поняттям зовнішнього ключа. Зовнішні ключі служать для зв'язку таблиць між собою. Поле однієї таблиці (головної, батьківської) називається зовнішнім ключем, якщо воно є первинним ключем іншої таблиці (підлеглої, дочірньої). Кажуть, що таблиця, у якій визначений зовнішній ключ, посилається на таблицю, у якій це ж поле є первинним ключем.

Вимога *цілісності посилань* полягає в тому, що для кожного значення зовнішнього ключа дочірньої таблиці повинен знайтися рядок у батьківській таблиці з таким же значенням первинного ключа. Наприклад, якщо у таблиці **R1** (рис. 14) наявні дані про співробітників кафедри, а атрибут цієї таблиці *Посада* є первинним ключем таблиці **R2**, то в цій таблиці для кожної посади з **R1** повинен бути рядок з відповідним їй окладом.

У більшості сучасних СУБД наявні засоби забезпечення контролю цілісності БД.

R1 Дочірня таблиця

Зовнішній ключ

ПІБ	Посада	Кафедра	Стаж
Шпак І. М.	викладач	ПЗС	5
Голуб М. І.	ст. викладач	ПЗС	7
Синиця Н. Г.	викладач	ПЗС	10
Орлик В. В.	викладач	ІПЗ	5

R2 Батьківська таблиця

Первинний ключ

Посада	Оклад
викладач	10000
ст.викладач	12000

Рисунок. 14. Зв'язок таблиць за допомогою зовнішнього ключа.

Висновки

1. Класичним методом проєктування БД є *Метод Нормальних форм*.

2. При проєктуванні бази даних методом нормальних форм спочатку створюється загальна вихідна таблиця, в яку вносяться дані про предметну область. Зазвичай вихідна таблиця утримує надлишкове дублювання даних і притаманні їм аномалії. Тому вихідна таблиця потребує нормалізації.

3. Для цього спочатку в вихідній таблиці необхідно виявити функціональні залежності між атрибутами.

2. Процес проєктування БД з використанням методу нормальних форм полягає в послідовному переведенні вихідної таблиці з першої нормальної форми в нормальні форми більш високого рівня.

3. Кожна наступна нормальна форма обмежує визначений тип функціональних залежностей, усуває відповідні аномалії і зберігає властивості попередніх нормальних форм, тобто має кращі властивості.

4. Всього існує п’ять нормальних форм. Але на практиці для нормалізації таблиць достатньо перевести їх у третю нормальну форму або у нормальну форму Бойса-Кодда.

5. У результаті проєктування утворюється сукупність зв’язаних нормалізованих таблиць. Кожна з цих таблиць утримує дані для моделювання певних об’єктів предметної області.

6. Метод нормальних форм відносять до класичних, тому що він завжди приводить до нормалізованих таблиць, в яких відсутнє надлишкове дублювання даних та притаманні їм аномалії.

7. Однак, при проєктуванні баз даних для великої предметної області, коли вихідна таблиця складається з десятків або сотень атрибутів, застосування методу нормальних форм суттєво ускладнюється. Це пов’язано з виявленням великої кількості функціональних залежностей між атрибутами вихідної таблиці. Тому метод нормальних форм доцільно використовувати для проєктування невеликих баз даних.

8. Підвищити ефективність застосування методу нормальних форм можна, якщо почати не із створення загальної великої вихідної таблиці, а відразу із створення таблиць для кожного

об’єкта предметної області. Зрозуміло, що таблиці для окремих об’єктів будуть значно меншими, ніж загальна вихідна таблиця, тому нормалізувати такі таблиці буде значно простіше.

9. Після проєктування бази даних можна приступити до її реалізації за допомогою певної СУБД.

ПРОЄКТУВАННЯ БАЗ ДАНИХ МЕТОДОМ СУТНІСТЬ-ЗВ'ЯЗОК

Розглядається проєктування реляційних баз даних методом сутність-зв'язок. Наводяться основні поняття методу та надаються приклади діаграм ER-екземплярів і діаграмам ER-типу. Аналізуються вихідні документи предметної області для виявлення сутностей та їх зв'язків. Відповідно до варіантів діаграм ER-екземплярів і ER-типу наводяться правила формування таблиць баз даних.

Основні поняття методу

Метод сутність-зв'язок називають також методом "ER-діаграм": по-перше, ER - абревіатура від слів Essence (сутність) і Relation (зв'язок), по-друге, метод заснований на використанні діаграм, які називаються відповідно діаграмами ER-екземплярів і діаграмами ER-типу.

Основними поняттями методу сутність-зв'язок є:

- сутність;
- атрибут сутності;
- ключ сутності;
- зв'язок між сутностями;
- ступінь зв'язку;
- клас приналежності екземплярів сутності;
- діаграми ER-екземплярів;
- діаграми ER-типу.

Сутність – це об'єкт предметної області, дані про який зберігаються в базі даних. Екземпляри сутності відрізняються один від одного та однозначно ідентифікуються. Назвами сутностей є, як правило, іменники, наприклад: ВИКЛАДАЧ, ДИСЦИПЛІНА, КАФЕДРА, ГРУПА.

Атрибут є властивістю сутності. Це поняття аналогічне поняттю атрибута у таблиці. Так, атрибутами сутності ВИКЛАДАЧ може бути його Прізвище, Посада, Стаж тощо.

Ключ сутності - атрибут або набір атрибутів, які використовуються для ідентифікації екземпляра сутності. Поняття ключа сутності аналогічне поняттю ключа таблиці.

Зв'язок двох чи більш сутностей припускає залежність між атрибутами цих сутностей. Назва зв'язку звичайно представляється дієсловом. Прикладами зв'язків між сутностями: ВИКЛАДАЧ ***ВЕДЕ*** ДИСЦИПЛІНУ (Орлик ВЕДЕ "Бази даних"), ВИКЛАДАЧ ***ВИКЛАДАЄ*** В ГРУПІ (Орлик ВИКЛАДАЄ в 256 групі).

Метод сутність зв'язок не цілком формалізований, але прийнятний для практичного використання. Зазвичай його використовують для проєктування БД для предметних областей, що складаються з великої кількості об'єктів. Варто мати на увазі, що в результаті проєктування можуть бути отримані кілька варіантів тієї ж БД. Так, два різних проєктувальники, розглядаючи одну і ту саму проблему з різних точок зору, можуть одержати різноманітні набори сутностей і зв'язків між ними. При цьому обидва варіанти можуть бути робочими, а вибір кращого з них буде результатом особистих переваг.

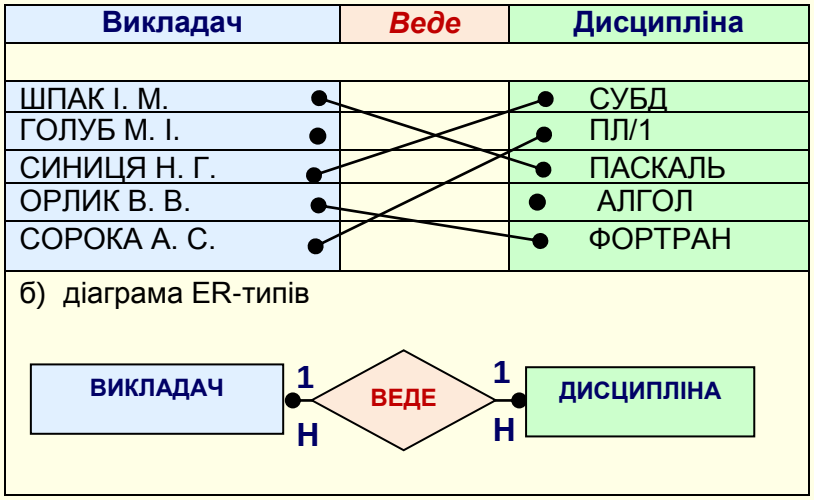
З метою підвищення наочності і зручності проєктування для представлення сутностей, екземплярів сутностей і зв'язків між ними використовуються наступні графічні засоби:

- ***діаграми ER-екземплярів*** - показують, як зв'язані між собою екземпляри сутностей;

– ***діаграми ER-типів чи ER-діаграми*** - показують тип зв'язку між сутностями.

На рис. 15 а) наведено приклад діаграми ER-екземплярів для сутностей **Викладач** і **Дисципліна** зі зв'язком ***Веде***.

а) діаграма ER-екземплярів



б) діаграма ER-типів

Рисунок 15. Діаграми ER-екземплярів і ER-типів.

Діаграма ER-екземплярів показує, яку конкретно дисципліну (СУБД, ПЛ/1 тощо) веде кожний з викладачів.

На рис. 15, б) наведено приклад діаграми ER-типу, що відповідає розглянутій діаграмі ER-екземплярів.

На початковому етапі проєктування БД виділяються атрибути, що складають ключі сутностей.

На основі аналізу діаграм ER-типу формуються таблиці проєктованої БД. При цьому враховується ***ступінь зв'язку сутностей*** і ***клас їхньої приналежності***, що, у свою чергу, визначаються на основі аналізу діаграм ER-екземплярів відповідних сутностей.

Ступінь зв'язку є характеристикою зв'язку між сутностями і може бути типу: 1:1, 1 : Б, Б : 1, Б : Б.

Клас приналежності (КП) сутності може бути: ***обов'язковим*** і ***необов'язковим***.

Клас приналежності сутності є обов'язковим, якщо всі екземпляри цієї сутності обов'язково беруть участь у розглянутому зв'язку, у протилежному випадку клас приналежності сутності є необов'язковим.

У залежності від класу приналежності сутностей для кожного з названих типів зв'язку можна одержати кілька варіантів діаграм ER-типу. Розглянемо приклади деяких з них.

Приклад 1. Зв'язок типу 1:1 та необов'язковий клас приналежності.

У наведеній на рис. 15 діаграмі ступінь зв'язку між сутностями 1:1, а клас приналежності обох сутностей необов'язковий. Дійсно, з рисунку видно наступне:

– кожен викладач веде не більше однієї дисципліни, а кожна дисципліна ведеться не більше, ніж одним викладачем (ступінь зв'язку 1:1);

– деякі викладачі не ведуть ні однієї дисципліни, і є дисципліни, що не веде жоден з викладачів (клас приналежності обох сутностей необов'язковий).

Приклад 2. Зв'язок типу 1:1 і обов'язковий клас приналежності. На рис. 16 наведені діаграми, у яких ступінь зв'язку між сутностями 1:1, а клас приналежності обох сутностей обов'язковий.

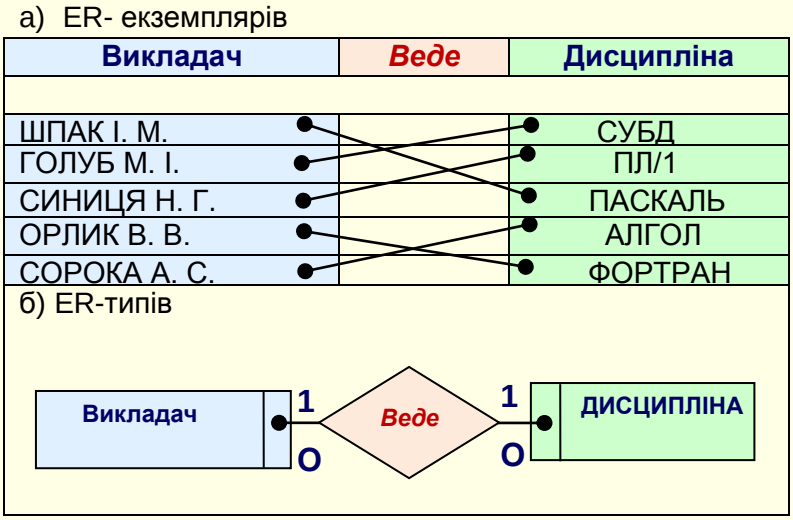


Рисунок 16. Діаграми для зв'язку 1:1 та обов'язковим КП обох сутностей.

У цьому випадку кожен викладач веде одну дисципліну, і кожна дисципліна ведеться одним викладачем.

Зауваження.

- На діаграмах ER-типу обов'язкова участь у зв'язку екземплярів сутності позначається блоком із крапкою усередині, суміжним із блоком цієї сутності. Під лінією зв'язку вказується буква **О** (рис. 16, б).
- При необов'язковій участі екземплярів сутності в зв'язку додатковий блок до блоку сутності не пристроюється, а крапка розміщається на лінії зв'язку. Під лінією зв'язку вказується буква **Н** (рис. 16, б).
- Символи на лінії зв'язку вказують на ступінь зв'язку. Під кожним блоком, що відповідає певній сутності, вказується її ключ, який виділяється підкресленням. Крапки за ключовими атрибутами означають, що можливі інші атрибути сутності, але жоден з них не може бути частиною її ключа. Ці атрибути виявляються після формування таблиць.
- На практиці ступінь зв'язку і клас приналежності сутностей при проєктуванні БД визначаються специфікою предметної області. Розглянемо приклади варіантів зі ступенем зв'язку 1 : Б або Б : 1.

Приклад 3. Зв'язок типу 1 : Б. Кожен викладач може вести кілька дисциплін, але кожна дисципліна ведеться одним викладачем.

Приклад 4. Зв'язок типу Б : 1. Кожен викладач може вести одну дисципліну, але кожну дисципліну можуть вести декілька викладачів.

Приклади з типом зв'язку 1 : Б або Б : 1 можуть мати ряд варіантів, що відрізняються класом приналежності однієї чи обох сутностей. Позначимо обов'язковий клас приналежності символом **О**, а необов'язковий - символом **Н**, тоді варіанти для зв'язку типу 1 : Б умовно можна представити як: О - О, О - Н, Н - О, Н - Н. Для зв'язку типу Б : 1 також наявні 4 аналогічні варіанти.

Приклад 5. Зв'язок типу 1 : Б варіант Н - О. Кожен викладач може вести кілька дисциплін або ні однієї, але кожна дисципліна ведеться одним викладачем (рис. 17).

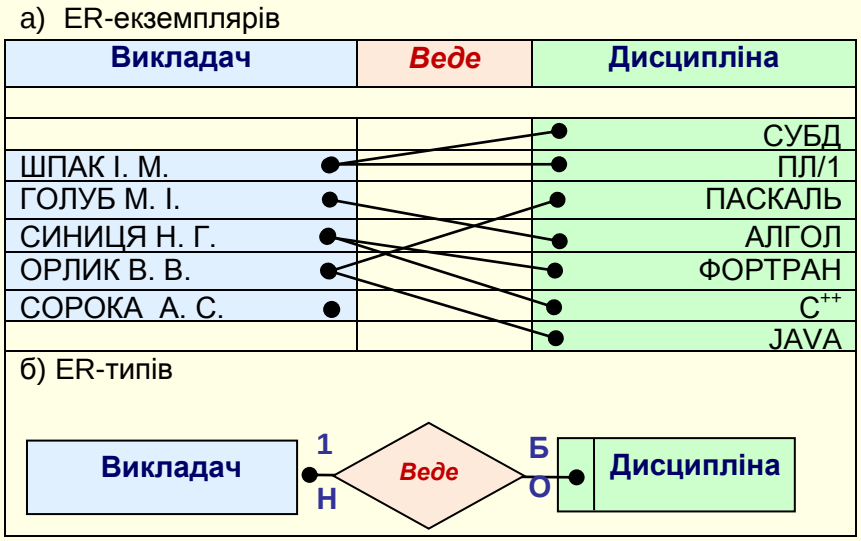


Рисунок 17. Діаграми для зв'язку типу 1 : Б варіанта Н – О.

За аналогією легко побудувати діаграми і для інших варіантів.

Приклад 6. Зв'язку типу Б : Б.

Кожен викладач може вести кілька дисциплін, а кожна дисципліна може вестися декількома викладачами.

Як і у випадку з іншими типами зв'язку, для зв'язку типу Б : Б можливі 4 варіанти, що відрізняються класом приналежності сутностей.

Приклад 7. Зв'язку типу Б : Б і варіант класу приналежності О - Н. Допустимо, що кожен викладач веде не менше однієї дисципліни, а дисципліна може вестися більше, ніж одним викладачем, є і такі дисципліни, що ніхто не веде. Відповідні до цього випадку діаграми наведені на рис. 18.

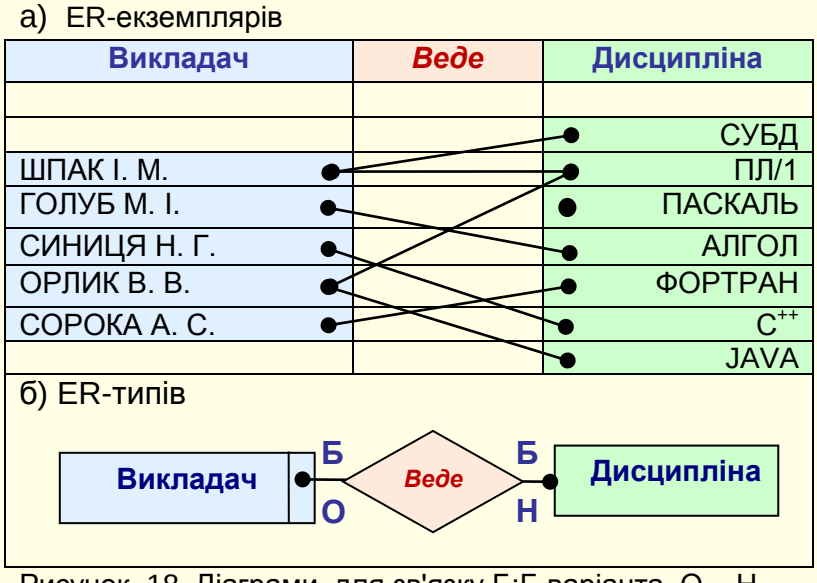


Рисунок. 18. Діаграми для зв'язку Б:Б варіанта О – Н.

Виявлення сутностей і зв'язків між ними, а також формування на їхній основі діаграм ER-типу виконуються на початкових етапах методу сутність-зв'язок. Розглянемо етапи реалізації методу.

Етапи проєктування БД методом сутність-зв’язок

Процес проєктування бази даних є ітераційним, який допускає повернення до попередніх етапів для перегляду раніше прийнятих рішень і включає наступні етапи:

1. аналіз предметної області, виділення сутностей і зв'язків між ними;
2. побудова діаграм ER-типу з урахуванням усіх сутностей і їхніх зв'язків;
3. формування набору попередніх таблиць із вказівкою передбачуваного первинного ключа для кожної таблиці та використанням діаграм ER-типу;
4. додавання неключових атрибутів у таблиці;
5. приведення попередніх таблиць до третьої нормальної форми або форми Бойса-Кодда методом нормальних форм;
6. перегляд ER-діаграм у наступних випадках:
   - деякі таблиці не приводяться до нормальної форми Бойса-Кодда;
   - деяким атрибутам не знаходиться логічно-обґрунтованих місць у попередніх таблицях.

Після перетворення ER-діаграм здійснюється повторне виконання попередніх етапів проєктування (повернення до етапу 1).

Одним з вузлових етапів проєктування є етап формування таблиць. Розглянемо процес формування попередніх таблиць, які утворюють первинний варіант схеми БД.

У наведених вище прикладах зв'язок ВЕДЕ завжди з'єднує дві сутності і тому є бінарним. Сформульовані нижче правила формування таблиць з діаграм ER-типу поширюються саме на бінарні зв'язки.

Правила формування таблиць

Правила формування таблиць засновані на урахуванні:

- ступеня зв'язку між сутностями (1:1, 1 : Б, Б : 1, Б : Б);
- класу приналежності екземплярів сутностей (обов'язковий і необов'язковий).

Розглянемо формулювання шести правил формування таблиць на основі діаграм ER-типу.

Формування таблиць для зв'язку 1:1

Правило 1. Якщо ступінь бінарного зв'язку 1:1 і клас приналежності обох сутностей обов'язковий, то формується одна таблиця. Первинним ключем цієї таблиці може бути ключ кожної з двох сутностей.

На рис. 19 наведено діаграму ER-типу і таблиця, сформована за правилом 1 на її основі. На рисунку використовуються наступні позначення: С1, С2 - сутності 1 і 2; К1, К2 - ключі першої і другої сутностей відповідно; R1 - таблиця, сформована на основі першої і другої сутностей; К1 v К2,... означає, що ключем сформованої таблиці може бути або К1, або К2.

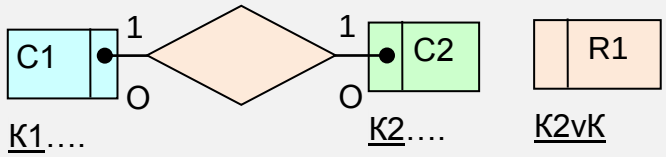


Рисунок. 19. Діаграма і таблиця для правила 1.

Викладач_Дисципліна

ПІБ	*Посада*	*Назва*	*Години*
Шпак І.М.	асистент	СУБД	60
Голуб М.І.	викладач	ПЛ/1	70
Синиця Н.Г.	доцент	ПАСКАЛЬ	100
Орлик В. В.	доцент	АЛГОЛ	80
Сорока С. А.	професор	ФОРТРАН	90

Рисунок. 20. Схема і таблиця за правилом 1.

Це й інші правила будемо перевіряти, розглядаючи різні варіанти зв'язку **Викладач *Веде* Дисципліну**. Нехай сутність **Викладач** характеризується атрибутами *ПІБ* (прізвище, ім'я та по-батькові), *Посада* (посада викладача). Сутність **Дисципліна** характеризується атрибутами *Назва* (найменування дисципліни), *Години* (години, що відводяться на дисципліну). Тоді схема таблиці, що містить дані про обидві сутності, і саме таблиця для випадку, коли ступінь зв'язку дорівнює 1:1, а КП обов'язковий для всіх сутностей, може мати вигляд, наведений на рис. 20. **Викладач_Дисципліна** (*ПІБ, Посада, Найменування, Години*).

Таблиця містить повну інформацію про викладачів, дисципліни і про те, як вони зв'язані між собою. Так, викладач Шпак веде тільки дисципліну СУБД, а дисципліна СУБД ведеться тільки Шпаком (зв'язок 1:1). У цій таблиці відсутні порожні поля (КП обов'язковий для всіх сутностей), тому що немає викладачів, які б щось не викладали, і немає дисциплін, які ніхто не викладає. Таким чином, однієї таблиці в даному випадку досить. Як первинний може бути обраний ключ першої таблиці – *ПІБ* або ключ другої таблиці – *Назва*.

Правило 2. Якщо ступінь зв'язку 1:1 і клас приналежності однієї сутності обов'язковий, а другої - необов'язковий, то під кожну із сутностей формується по таблиці з первинними ключами, що є ключами відповідних сутностей. Далі до таблиці, сутність якої має обов'язковий КП, додається як атрибут ключ сутності з необов'язковим КП.

На рис. 21 наведена діаграма ER-типу і таблиця, сформовані за правилом 2 на її основі.

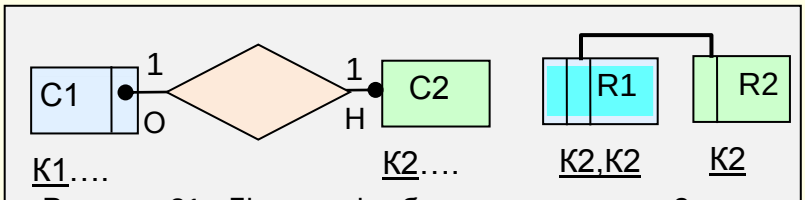


Рисунок. 21. Діаграма і таблиця для правила 2.

Щоб переконатися в справедливості правила, розглянемо наступний приклад. На рис. 22 наведена вихідна таблиця, яка містить дані про викладачів і дисципліни. Вона представляє варіант, у якому клас сутності **ВИКЛАДАЧ** є обов'язковим, а сутності **ДИСЦИПЛІНА** – необов'язковим. При цьому проміжки "–" (порожні поля) присутні у всіх кортежах з даними про дисципліни, що не викладаються жодним з викладачів.

Викладач

ПІБ▼	*Посада*	*Назва*	*Години*
Шпак І.М.	асистент	ПАСКАЛЬ	100
Голуб М.І.	викладач	СУБД	60
Синиця Н.Г.	доцент	ПЛ/1	70
Орлик В.В.	доцент	АЛГОЛ	80
...	---	ФОРТРАН	90

Рисунок. 22. Вихідна таблиця.

Дисципліна

ПІБ▼	*Посада*	*Назва*
Шпак І.М.	асистент	ПАСКАЛЬ
Голуб М.І.	викладач	СУБД
Синиця Н.Г.	доцент	ПЛ/1
Орлик В.В.	доцент	АЛГОЛ

Назва▼	*Години*
ПАСКАЛЬ	100
СУБД	60
ПЛ/1	70
АЛГОЛ	80
ФОРТРАН	90

Рисунок. 23. Таблиці, отримані за правилом 2.

Уникнути цієї ситуації можна, застосувавши правило 2, відповідно до якого утворюються дві таблиці, наведені на рис. 23.

У результаті ми уникнули порожніх полів у таблицях, не втративши даних. Додавши атрибут *Назва* - ключ сутності **Дисципліна** (з необов'язковим КП) як зовнішній ключ у таблицю, що відповідає сутності **Викладач** (з обов'язковим КП), ми зв'язали таблиці по полю *Назва* (рис. 24).

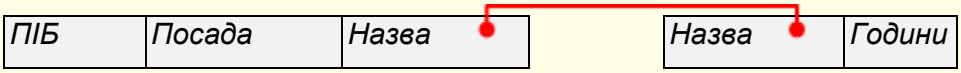


Рисунок. 24. Зв'язок таблиць по зовнішньому ключу.

Точніше кажучи, ми створили умови для зв'язування таблиць. Це дозволяє одержувати одночасно дані про викладачів і про дисципліни, які вони викладають (години).

Правило 3. Якщо ступінь зв'язку 1:1 і клас приналежності обох сутностей є необов'язковим, то необхідно використовувати три таблиці. Дві таблиці відповідають сутностям, що зв'язуються, ключі яких є первинними в цих таблицях. Третя таблиця є зв'язковою між першими двома, тому її ключ поєднує ключові атрибути таблиць, які зв'язуються.

На рис. 25 наведена діаграма ER-типу і таблиці, сформовані за правилом 3 на її основі.

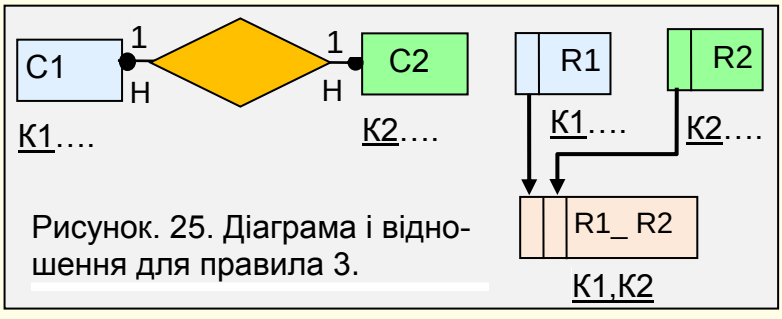


Рисунок. 25. Діаграма і відношення для правила 3.

На рис. 26 наведені приклади таблиць, які підтверджують необхідність використання трьох таблиць при наявності необов'язкового класу приналежності для обох сутностей, що зв'язуються.

Використання однієї таблиці в даному випадку приводить до наявності небажаних порожніх полів (рис. 26, а). При використанні двох таблиць (рис. 26, б) доведеться додати ключі кожної із сутностей в таблицю, що відповідає іншій сутності, щоб не втратити зведення про те, яку дисципліну викладає кожен викладач і навпаки. При цьому також з'явилися порожні поля.

а) одна таблиця

Викладач_Дисципліна

ПІБ	*Посада*	*Назва*	*Години*
Шпак І.М.	асистент	ПАСКАЛЬ	100
Голуб М.І.	викладач	СУБД	60
Синиця Н.Г.	доцент	ПЛ/1	70
…	…	АЛГОЛ	80
Орлик В.В.	доцент	ФОРТРАН	90
Сорока С.А.	професор	…	…

б) дві таблиці

Викладач

ПІБ	*Посада*	*Назва*
Шпак І.М.	асистент	ПАСКАЛЬ
Голуб М.І.	викладач	СУБД
Синиця Н.Г.	доцент	ПЛ/1
Орлик В.В.	доцент	ФОРТРАН
Сорока С.А.	професор	…

Дисципліна

Назва	*Години*	*ПІБ*
ПАСКАЛЬ	100	Шпак І.М.
СУБД	60	Голуб М.І.
ПЛ/1	70	Синиця Н.Г.
АЛГОЛ	80	
ФОРТРАН	90	Орлик В.В.

в) три таблиці

Викладач

ПІБ	*Посада*
Шпак І.М.	асистент
Голуб М.И.	викладач
Синиця Н.Г.	доцент
Орлик В.В.	доцент
Сорока С.А.	професор

Веде

ПІБ	*Назва*
Шпак І.М.	ПАСКАЛЬ
Голуб М.І.	СУБД
Сидоров Н.Г.	ПЛ/1
Орлик В.В.	ФОРТРАН

Дисципліна

Назва	*Години*
ПАСКАЛЬ	100
СУБД	60
ПЛ/1	70
АЛГОЛ	80
ФОРТРАН	90

Рис. 26. Варіанти таблиць для правила 3.

Вихід полягає у використанні трьох таблиць, сформованих за правилом 3 (рис. 26, в). Об'єктні таблиці (з атрибутами сутностей) містять дані про всіх викладачів і дисципліни відповідно. Зв'язна таблиця **Веде** містить дані про викладачів, які ведуть дисципліни і про дисципліни, що ведуться викладачами. При цьому в ній є тільки одне згадування про кожного викладача і дисципліну відповідно до зв'язку 1:1. Ця таблиця містить тільки ключові атрибути обох сутностей, але може мати й інші атрибути, що характеризують цей зв'язок. Наприклад, номер семестру, у якому викладач веде дисципліну.

Отже, сформульовані три правила, що дозволяють формувати відношення на основі ER-діаграм для варіантів зі ступенем зв'язку типу 1:1.

Формування таблиць для зв'язку 1:Б

Якщо дві сутності **С1** і **С2** зв'язані як 1:Б, то сутність **С1** будемо називати однозв'язною (1-зв'язна), а сутність **С2** - багатозв'язною (Б-зв'язна). Визначальним фактором при формуванні таблиць, зв'язаних цим видом зв'язку, є клас приналежності Б-зв'язної сутності. Так, якщо клас приналежності Б-зв'язної сутності обов'язковий, то в результаті застосування правила одержимо дві таблиці, якщо необов'язковий - три таблиці. Клас приналежності однозв'язної сутності не впливає на результат.

Викладач_Дисципліна

ПІБ	*Посада*	*Назва*	*Години*
Шпак І.М.	асистент	СУБД	60
Шпак І.М.	асистент	ПЛ/1	70
Голуб М.І.	викладач	АЛГОЛ	80
Синиця Н.Г.	доцент	ФОРТРАН	90
Синиця Н.Г.	доцент	С++	85
Орлик В. В.	доцент	ПАСКАЛЬ	100
Орлик В. В.	доцент	JAVA	65
Сорока А. С.	професор	….	….

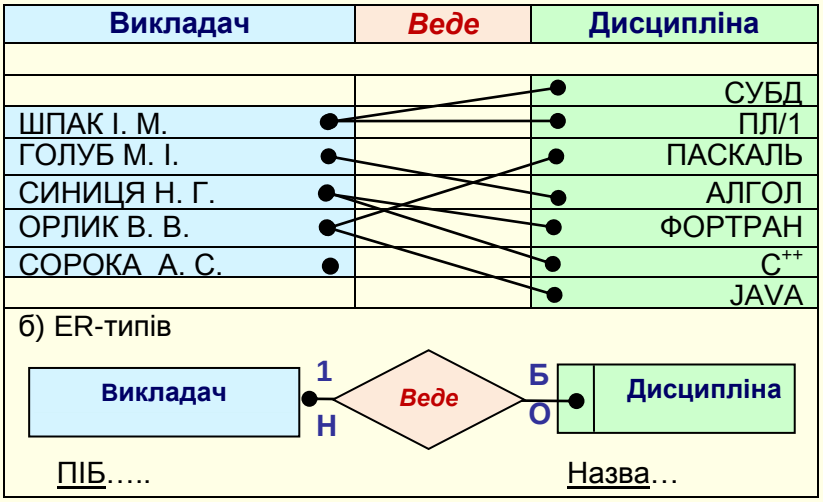
Рисунок. 27. Вихідна таблиця.

Щоб переконатися в цьому, розглянемо таблицю **Викладач_Дисципліна** (рис. 27), якому відповідають діаграми, наведені на рис. 28, тобто випадку, коли: зв'язок типу 1 : Б, клас приналежності Б-зв'язної сутності обов'язковий, 1 - зв'язної - необов'язковий.

З таблицею **Викладач_ Дисципліна** (рис. 27) пов'язані наступні проблеми:

- наявні кортежі з порожніми полями (викладач не веде дисципліни);
- надлишкове дублювання даних (повторюється посада викладача) у кортежах зі зведеннями про викладачів, які ведуть кілька дисциплін.

а) ER-екземплярів



б) ER-типів

Рисунок 28. Діаграми для зв'язку типу 1 : Б варіанта Н – О.

Якби клас приналежності 1-зв'язної сутності був обов'язковим (немає викладача без дисципліни), то зникли б порожні поля, але повторювані дані в атрибутах викладача збереглися б.

Для усунення названих проблем таблиці повинні бути сформовані за наступним правилом.

Правило 4. Якщо ступінь зв'язку між сутностями 1:Б (або Б:1) і клас приналежності Б-зв'язної сутності обов'язковий, то достатньо формування двох таблиць (по таблиці на кожну із сутностей). При цьому первинними ключами цих таблиць є ключі їхніх сутностей. Крім того, ключ 1-зв'язної сутності додається як атрибут (зовнішній ключ) у таблицю, що відповідає Б-зв'язній сутності.

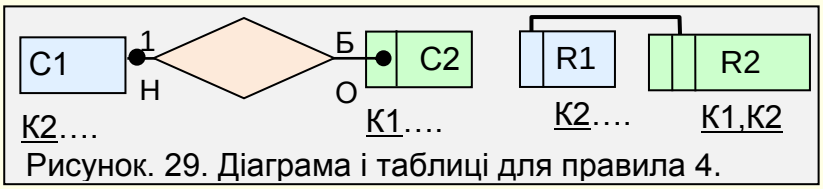


Рисунок. 29. Діаграма і таблиці для правила 4.

На рис. 29 наведена діаграма ER-типу і таблиці, сформовані за правилом 4.

Відповідно до правила 4 перетворимо таблицю на рис. 27 на дві таблиці (рис. 30).

Викладач

ПІБ	*Посада*
Шпак І.М.	асистент
Голуб М.І.	викладач
Синиця Н.Г.	доцент
Орлик В.В.	доцент
Сорока А.С.	професор

Дисципліна_ Викладач

Назва	*Години*	*ПІБ*
СУБД	60	Шпак І.М.
ПЛ/1	70	Шпак І.М.
АЛГОЛ	80	Голуб М.І.
ФОРТРАН	90	Синиця Н.Г.
С++	85	Синиця Н.Г.
ПАСКАЛЬ	100	Орлик В.В.
JAVA	65	Орлик В.В.

Рисунок. 30. Таблиці, що отримані за правилом 4.

З рис. 30 видно, що порожні поля і дублювання даних вдалося усунути. Утрати зведень про те, хто з викладачів веде яку дисципліну, не відбулося завдяки введенню ключа *ПІБ* сутності **Викладач** як зовнішнього ключа у таблицю **Дисципліна_ Викладач**.

Приклад. Зв'язок між сутностями 1 : Б, а клас приналежності Б-зв'язної сутності необов'язковий.

Викладач_Дисципліна

ПІБ	*Посада*	*Назва*	*Години*
Іванов І.М.	асистент	СУБД	60
Іванов І.М.	асистент	ПЛ/1	70
Петров М.І.	викладач	АЛГОЛ	80
Сідоров Н.Г.	доцент	ФОРТРАН	90
…	…	С++	85
Єгоров В. В.	доцент	ПАСКАЛЬ	100
Єгоров В. В.	доцент	JAVA	65
Козлов А.С.	професор	…	…

Рис. 31. Вихідна таблиця.

Нехай клас приналежності 1-зв'язної сутності також необов'язковий, хоча це і не принципово, тому що визначальним є клас приналежності Б-зв'язної сутності. Подивимося, до чого може привести використання однієї таблиці в цьому випадку (рис. 31).

З наведеною таблицею пов'язані проблеми:

1. Є порожні поля в кортежах, які містять наступне:

   а) дані про викладачів, які не ведуть дисциплін;

   б) дані про дисципліни, які не ведуться викладачами.

2. Надлишкове дублювання даних про викладачів, які ведуть більше однієї дисципліни.

У випадку обов'язкового класу приналежності 1-зв'язної сутності зникають проблеми 1 а). Для усунення всіх проблем потрібно перейти до трьох відношень у відповідності з наступним правилом.

Правило 5. Якщо ступінь зв'язку 1 : Б (Б : 1) і клас приналежності Б-зв'язної сутності є необов'язковим, то необхідно формування трьох таблиць (рис. 32). Дві таблиці відповідають сутностям що зв'язуються, ключі яких є первинними в цих таблицях. Третя таблиця є сполучною між першими двома (його ключ поєднує ключові атрибути таблиць, що зв'язуються)

У результаті застосування правила 5 до таблиці (рис. 32) її дані розподіляються за трьома таблицями (рис. 33).

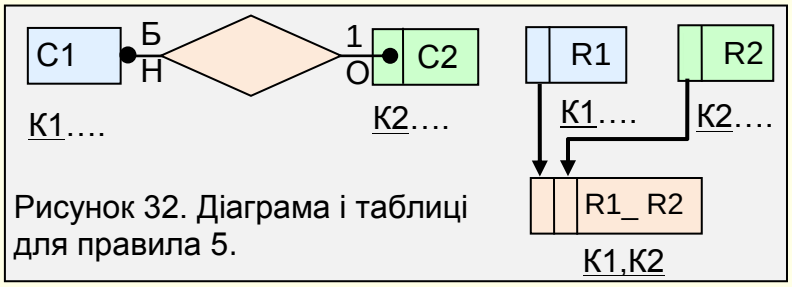


Рисунок 32. Діаграма і таблиці для правила 5.

Таким чином, зазначені проблеми вдалося уникнути. Ключ у зв'язаній таблиці ВЕДЕ є складовим і містить у собі ключові атрибути обох таблиць, які зв'язуються. Підкреслимо, що визначальним фактором при виборі між 4-м або 5-м правилом є клас приналежності Б-зв'язної сутності.

Викладач

ПІБ	*Посада*
Шпак І.М.	асистент
Голуб М.І.	викладач
Сидоров Н.Г.	доцент
Орлик В.В.	доцент
Сорока А.С.	професор

Веде

ПІБ	*Назва*
Шпак І.М.	СУБД
Шпак І.М.	ПЛ/1
Голуб М.І.	АЛГОЛ
Синиця Н.Г.	ФОРТРАН
Орлик В. В.	ПАСКАЛЬ
Орлик В. В.	JAVA

Дисципліна

Назва	*Години*
СУБД	60
ПЛ/1	70
АЛГОЛ	80
ФОРТРАН	90
С++	85
ПАСКАЛЬ	100
JAVA	65

Рисунок 33. Таблиці, що отримані за правилом 5.

Формування таблиць для зв'язку Б:Б

При наявності зв'язку Б : Б між двома сутностями необхідно сформувати три таблиці незалежно від класу приналежності кожної із сутностей. Використання однієї чи двох таблиць у цьому випадку не рятує від порожніх полів або полів, в яких дані надлишково дублюються.

Правило 6. Якщо ступінь зв'язку сутностей Б : Б, то незалежно від класу приналежності сутностей формуються три таблиці. Дві таблиці відповідають сутностям, що зв'язуються, і їхні ключі є первинними ключами цих сутностей. Третя таблиця є сполучною між першими двома, а її ключ поєднує ключові атрибути таблиць, що зв'язуються.

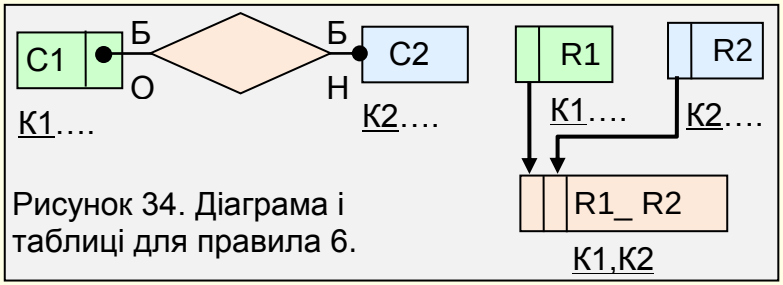


Рисунок 34. Діаграма і таблиці для правила 6.

На рис. 34 наведено діаграму ER-типу і таблиці, сформовані за правилом 6. По-

казано варіант із класом приналежності сутностей О-Н, хоча, відповідно до правила 6, він може бути довільним.

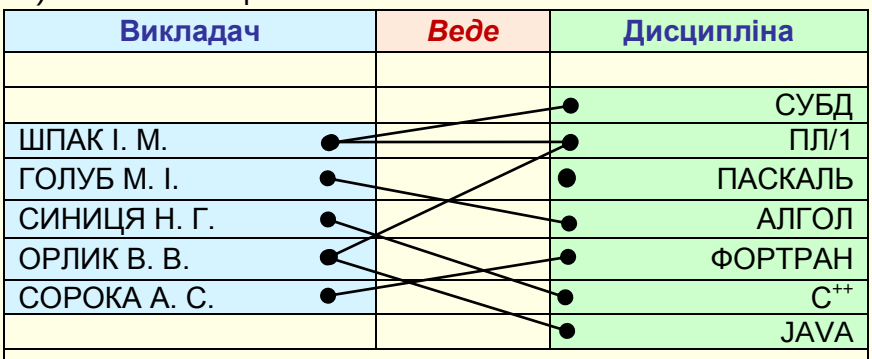
ВИКЛАДАЧ_ДИСЦИПЛІНА

ПІБ	*Посада*	*Назва*	*Години*
Шпак І.М.	асистент	СУБД	60
Шпак І.М.	асистент	ПЛ/1	70
Голуб М.І.	викладач	АЛГОЛ	80
Сидоров Н.Г.	доцент	ФОРТРАН	90
…	…	С++	85
Орлик В. В.	доцент	ПАСКАЛЬ	85
Орлик В. В.	доцент	JAVA	100
Сорока А.С.	професор	ФОРТРАН	65

Рисунок 35. Вихідна таблиця.

Застосуємо правило 6 до вихідної таблиці, наведеної на рис. 35. У ній ступінь зв'язку Б : Б, клас приналежності для сутності **Викладач** обов'язковий, а для сутності **Дисципліна** - необов'язковий. Відповідні цьому прикладу діаграми наведені на рис. 36.

а) ER-екземплярів



б) ER-типів

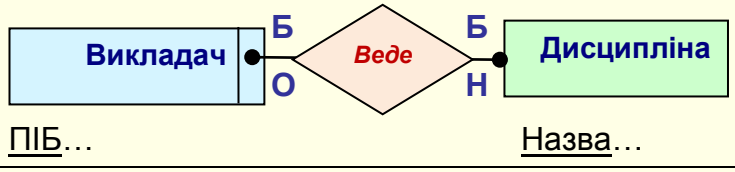


Рис. 36. Діаграми для зв'язку типу Б:Б варіанта О – Н.

У результаті застосування правила 6 отримуємо три таблиці (рис. 37). Аналогічні результати можна отримати і для трьох інших варіантів, що розрізняються класами приналежності їхніх сутностей.

Викладач

ПІБ	*Посада*
Шпак І.М.	асистент
Голуб М.І.	викладач
Синиця Н.Г.	доцент
Орлик В. В.	доцент
Сорока А.С.	професор

Веде

ПІБ	*Назва*
Шпак І.М.	СУБД
Шпак І.М.	ПЛ/1
Голуб М.І.	АЛГОЛ
Сидоров Н.Г.	ФОРТРАН
Орлик В. В.	ПАСКАЛЬ
Орлик В. В.	JAVA
Сорока А.С.	ФОРТРАН

Дисципліна

Назва	*Години*
СУБД	60
ПЛ/1	70
АЛГОЛ	80
ФОРТРАН	90
С++	85
ПАСКАЛЬ	85
JAVA	100
ФОРТРАН	65

Рисунок 37. Таблиці, які отримані згідно правила 6.

Висновки

1. Для проєктування баз даних будь-якого розміру використовується метод сутність-зв’язок.

2. Метод сутність-зв’язок не до кінця формалізований. Для отримання працездатного проєкту бази даних потрібно зробити кілька ітерацій.

3. Таблиці, отримані методом сутність-зв’язок, обов’язково перевіряються на відповідність нормальним формам.

4. Для застосування методу сутність-зв’язок спочатку аналізується предметна область для виявлення сутностей та зв’язків між ними.

5. Для кожної пари зв’язаних сутностей будуються ER-діаграми:

– діаграми ER-екземплярів – показують, як зв'язані між собою екземпляри сутностей або ступінь зв'язку між сутностями (1 : 1, 1 : Б, Б : Б);

– діаграми ER-типів – показують тип зв'язку між сутностями або клас приналежності екземплярів сутностей (обов’язковий чи необов’язковий).

6. В залежності від ступеню зв’язку між сутностями та класу приналежності для формування таблиць використовують шість правил.

ПРОЄКТУВАННЯ БАЗИ ДАНИХ МЕТОДОМ СУТНІСТЬ-ЗВ'ЯЗОК

На прикладі розглядається проєктування реляційної бази даних для предметної області Деканат методом сутність-зв'язок. Аналізуються вихідні документи для виявлення сутностей та зв'язків між ними. Будуються діаграми ER-екземплярів і ER-типу. Визначаються таблиці та зв'язки між ними. Всі таблиці перевіряються на відповідність нормальним формам.

Проєкт реляційної бази даних **–** це набір взаємозалежних таблиць, у яких визначені всі поля, задані первинні ключі і інші властивості, які підтримують цілісність даних.

Етапи проєктування баз даних методом сутність-зв'язок.

1. Опис предметної області, пошук і впорядкування необхідних відомостей;
2. Виділення сутностей і зв'язків між ними;
3. Побудова діаграм ER-екземплярів і ER-типу з урахуванням усіх сутностей і їхніх зв'язків;
4. З використанням діаграм ER-екземплярів і ER-типу та правил методу сутність-зв'язок визначення:
   – сукупності таблиць з вказівкою ключових та неключових полів;
   – зв'язків між таблицями;
5. Перевірка таблиць на відповідність третьої нормальної форми або форми Бойса-Кодда;
6. Перегляд ER-діаграм і побудова нових таблиць у наступних випадках:
   – деякі таблиці не приводяться до третьої нормальної форми або форми Бойса-Кодда;
   – деяким атрибутам не знаходиться логічне обґрунтованих місць у попередніх таблицях.
7. Оформлення проєкту у вигляді схеми БД;
8. Реалізація проєкту в середовищі СУБД.

Опис предметної області, пошук і впорядкування необхідних відомостей

Формулювання призначення бази даних

Необхідно чітко сформулювати призначення БД певної предметної області, хто і як планує її використати. По суті це є формулюванням задачі БД. Формулювання призначення БД має бути коротким і зазвичай складатися з декількох речень. До уточнення формулювання призначення БД можна повернутися під час процесу її проєктування. Чітке розуміння призначення БД допомагає зосередитися на визначених цілях під час прийняття рішень. Наприклад, це дозволяє зосередитися на зборі тільки тих вхідних і вихідних документів, які пов'язані з поставленою задачею.

Завдання: розробка застосування бази даних факультету Інформатики, яке призначено для отримання інформації про навчальний процес поточного семестру.

Опис предметної області.

Зазвичай це неформальний (словесний) опис предметної області та сукупність вхідних і вихідних документів з даними, що вони утримують, а також інша інформація, яка надається замовником БД.

Документи мають бути заповненими, наприклад, списки навчальних груп з прізвищами студентів, екзаменаційні відомості з прізвищами студентів, назвами предметів і оцінками.

Крім цього, від замовника необхідно одержати відомості про додаткову інформацію, яку б він хотів одержати з бази даних. Наприклад, середній бал за сесію певного студента або предмету, прізвища студентів, що не склали сесію, тощо.

Сукупність цих документів і відомостей утворюють словесну характеристику предметної області, для якої створюється БД. Зауважимо, що до уваги беруться тільки ті документи, які мають відношення до призначення БД. Наприклад, у випадку, що розглядається, не є визначальною кількість комп'ютерів на факультеті та їх розташування в класах, тому ці дані можна не брати до уваги.

Предметна область характеризується даними щодо студентів, викладачів і навчальних предметів та дозволяє отримувати інформацію про результати екзаменаційної сесії. Наповнення бази даними здійснюється методистом факультету. Перегляд бази доступний студентам і викладачам факультету. До складу бази даних входять:

– списки студентів в групах;

– перелік предметів, які вивчаються;

– викладацький склад кафедр;

– відомості про лекційні і практичні заняття в кожній з груп;

– екзаменаційні і залікові відомості у всіх групах з усіх предметів.

Форми документів, що утримують вхідну довідкову інформацію.

Спеціальності.

На факультеті навчання проводиться за спеціальностями:

1. Інформаційні системи і технології (ІСТ).
2. Комп'ютерні науки (КН);
3. Комп'ютерні науки (Інформатика);
4. Інженерія програмного забезпечення (ІПЗ);
5. Інженерія програмного забезпечення, Молодший спеціаліст (МС).

Групи.

На одному курсі навчається по одній групі кожної спеціальності. Групи позначаються: КН-1, ІСТ-1 ,ІПЗ-1, МС-1 ...ІСТ-n, ІПЗ-n, МС-n, де цифри від 1 до n позначають номер групи, які зростають. Групи містять списки студентів: прізвище та ім'я, номер залікової книжки; електронна пошта; телефон.

Дані щодо студентських груп утримуються в таблицях. Для прикладу в таблиці на рис. 38 наведено склад групи №3, спеціальності Молодший спеціаліст (МС-3).

Список студентів

Спеціальність: Інженерія програмного забезпечення, Молодший спеціаліст

Курс: 3 Навчальна група: МС-03

Прізвище та ім’я	Залікова книжка	Електронна пошта	Телефон
Баб'як Олег	МС-0001	babyak.oleh1@student.uzhnu.edu.ua	0502659288
Гранич Домініка	МС-0002	hranych.dominika@student.uzhnu.edu.ua	0509128495
Гулан Михайло	МС-0003	hulan.mykhailo@student.uzhnu.edu.ua	0995312905
Гутич Владислав	МС-0004	hutych.vladyslav@student.uzhnu.edu.ua	0684367648
Єрмолаєв Віктор	МС-0005	yermolaiev.viktor@student.uzhnu.edu.ua	0951369669
…			
Чіжмар Крістіна	МС-0017	chizhmar.kristina@student.uzhnu.edu.ua	0660884340
Чума Дмитро	МС-0018	chuma.dmytro@student.uzhnu.edu.ua	0958752405
Шикула Юрій	МС-0019	shykula.yurii@student.uzhnu.edu.ua	0991388997
Кількість студентів в групі 19			

Рисунок 38. Дані щодо студентських груп.

Навчальний процес здійснюється викладачами кафедр. Дані щодо кафедр наявні в таблиці на рис. 39.

Склад кафедри програмного забезпечення систем

Прізвище, ім’я та по батькові	Посада	Вчене звання
Білак Юрій Юрійович	Зав. Кафедри, доцент	Кандидат наук, доцент
Нелюбов Володимир Олександрович	Професор	Кандидат наук, професор
…		
Легеза Андрій Васильович	Викладач	Без звання
Кількість викладачів на кафедрі 24		

Рисунок 39. Дані щодо кафедр.

Предмети читаються викладачами, які мають атрибути: прізвище, ім'я і по батькові; науковий ступінь; вчене звання; кафедра. Причому викладачі кафедри читають тільки ті предмети, які закріплені за цією кафедрою.

Відповідно до навчального плану всі студенти групи в поточному семестрі вивчають певні предмети. Дані щодо планів проведення занять знаходяться в таблиці, форма якої наведена на рис. 40.

Всі предмети мають назви, закріплені за певними кафедрами і викладачами, за ними проводяться певні види занять (лекції, практичні заняття) і здійснюється підсумковий контроль (іспит, або залік).

План проведення занять

Номер групи	Назва предмета	Назва кафедри	ПІБ викладача	Вид занять	Вид контролю
МС-003	Проєктування БД і експертних систем	ПЗС	Нелюбов Володимир Олександрович	Лекції	Іспит
МС-003	Проєктування БД і експертних систем	ПЗС	Нелюбов Володимир Олександрович	Лаб. роботи	Залік
ІПЗ-003	Проєктування БД і експертних систем	ПЗС	Білак Юрій Юрійович	Лаб. роботи	Залік
МС-003	Операційні системи	ПЗС	Левчук Олександр Миколайович	Лекції	Іспит
МС-003	Операційні системи	ПЗС	Клименко Михайло Володимирович	Лаб. роботи	Залік
…					

Рисунок 40. Плани проведення занять.

Екзаменаційна відомість

Спеціальність: Інженерія програмного забезпечення, Молодший спеціаліст

Курс: 3 Група: № МС-003

Назва предмета: Проєктування баз даних і експертних систем

Викладач: Нелюбов Володимир Олександрович

Вид контролю: Залік

Дата: 15.12.2023 р.

Номер залікової книжки	Прізвище та ім’я	Оцінка	Підпис викладача
ІУСТ\0020	Дурдинець Василь		
ІУСТ\0021	Куль Микита		
ІУСТ\0022	Попович Віктор		
…			

Кількість студентів в групі ___
Здавало іспит ___
Отримало оцінку: відмінно ___
добре ___
задовільно ___
незадовільно ___

Не з’явилось

Рисунок 41. Екзаменаційна відомість.

Вивчення предметів закінчується підсумковим контролем знань у вигляді іспиту або заліку. Дані про результати контролю зберігаються в екзаменаційній відомості, форма якої наведена на рис. 41.

База даних має надати можливість отримати наступну інформацію:

1. Склад навчальних груп (списки).

2. Загальна інформація щодо кожного студента (прізвище та ім’я, номер залікової книжки, адреса).

3. Інформація про предмети, які повинна вивчати кожна студентська група (назва предмету, форма контролю, викладач).

4. Інформація про предмети, які має повинен вивчати кожний студент (назва предмету, форма контролю, викладач).

5. Інформація щодо кафедр факультету (назва, викладацький склад).

6. Інформація щодо викладачів (прізвище, посада, наукове звання, предмет, група, вид занять.

7. Екзаменаційні і залікові відомості по предметах і групах (номер групи, назва предмета, посада, прізвище, ім’я та по-батькові викладача, номер залікової книжки, прізвище та ім’я студента, оцінка, дата, кількість студентів у групі, кількість студентів, які проходили контроль, кількість студентів, що отримали певну оцінку, середній бал з предмета в групі, вид контролю).

Значення оцінок за результатами іспиту визначаються чотирьох-бальною шкалою (від 2 до 5). Результати заліку визначаються текстом: *зараховано* або *не зараховано*.

Визначення сутностей та їх атрибутів

Це перший крок у напрямку формалізації предметної області. Завданням етапу є виділення основних абстракцій (сутність, атрибут, зв'язок) у предметній області і визначення їх параметрів.

Будь-яка предметна область утворюється як сукупність взаємопов’язаних об’єктів, кожний з яких є також сукупністю дрібніших об’єктів. Еквівалентом будь-якого об’єкту в проєкті БД є сутність. На даному етапі необхідно визначити сутності (об’єкти), які б характеризували предметну область в обсязі, достатньому для вирішення задачі БД. Цей процес не формалізовано, тому його результат залежить від досвіду проєктувальника. Одночасно доцільно виявити характеристики кожної сутності. Наприклад, характеристиками (атрибутами) сутності **Студент** можуть бути: прізвище та ім’я, номер залікової книжки, адреса електронної пошти тощо. Оскільки номер залікової книжки є унікальним, то за ним можна ідентифікувати кожного студента. Тут також важливо обрати тільки ті атрибути, які є визначальними для вирішення поставленої задачі. Наприклад, ріст студента або колір його очей у даному випадку не мають ніякого значення.

При визначені сутностей та їх характеристик можна дотримуватися наступної рекомендації: уважно розглянути кожний вхідний і вихідний документ і в окрему сутність виділяти такі відомості, що повторюються в документі кілька разів, або сукупність деяких відомостей утворює декілька однотипних екземплярів.

У загальному випадку документ складається із назви, шапки, тіла і підсумків (рис. 42).

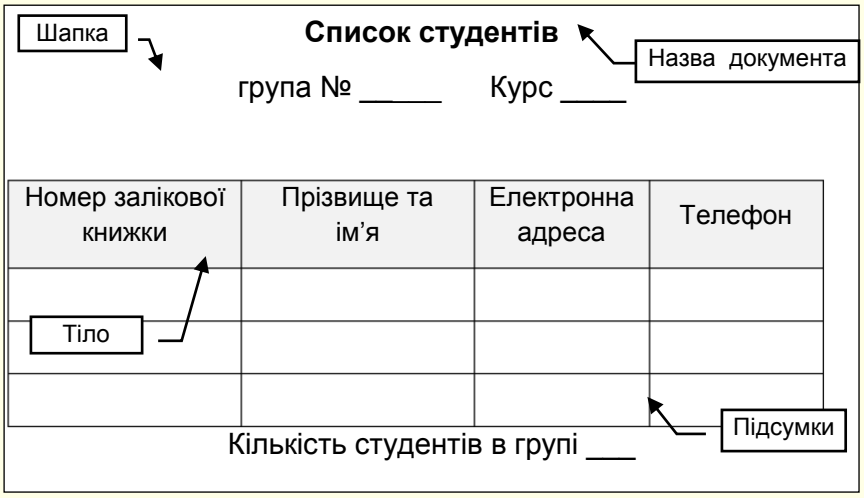


Рисунок 42. Приклад документа.

Простіше за все було б розглядати кожен документ як певну окрему сутність. Але це не зовсім правильно. Уважний перегляд документа дозволяє зробити висновок, що він складається з декількох об'єктів.

Ознакою наявності окремого об'єкта є те, що відомості про нього повторюються в документі багато разів. Зазвичай такі відомості виносяться в шапку і тіло документа. Дійсно, якщо б номер групи був внесений до тіла документа, то він повторювався б для всіх студентів однієї групи. Крім цього, в окрему сутність слід виділяти характеристики певного об'єкта. Дані, що відображаються в розділі ***Підсумки*****,** зазвичай в БД не зберігаються і переважно є результатом запитів, які поновлюються при кожному звернені до бази даних.

Виходячи з аналізу форм вхідних документів, можна виділити наступні сутності та їх зв'язки.

Документ Список студентів (рис. 38) утримує дані про ***сутності***:

1. Сутність **Група** характеризується атрибутами: номер групи (***Група***); номер курсу (***Курс***). Ключем сутності можна обрати атрибут ***Група***, який є унікальним.

Сутність_Група
Група
Курс

2. Сутність **Студент** характеризується атрибутами: прізвище та ім'я (***ПІ_Студент***), електронна пошта (***е-пошта***), номер телефону (***Телефон***). Ключем сутності можна обрати атрибут (***ПІ_Студент*****)**, який є унікальним.

Сутність_Студент
ПІ_Студент
Пошта
Телефон

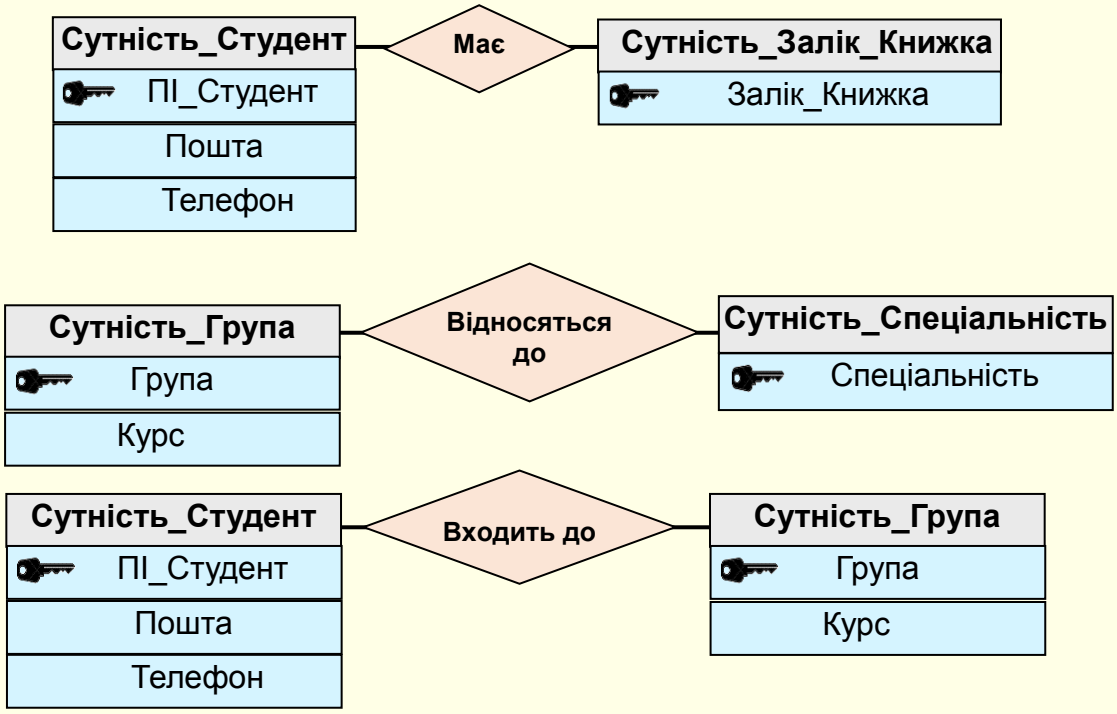
3. Сутність **Залік_Книжка** характеризується атрибутом номер залікової книжки (**Залік_Книжка**). Цей атрибут є ключем.

Сутність
Залік_Книжка

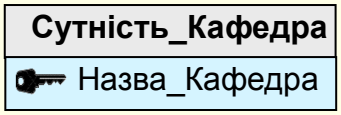
4. Сутність **Спеціальність** з атрибутом **Спеціальність**, який є атрибутом сутності.

Сутність_Спеціальність
Спеціальність

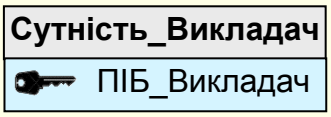
Ці сутності зв’язані між собою:



Документ ***Склад кафедри*** (рис. 39) утримує дані про сутності:

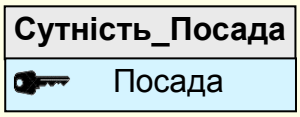


5. **Сутність_Кафедра** характеризується атрибутом: назва кафедри (**Назва_Кафедра**), яку можна прийняти ключем сутності.

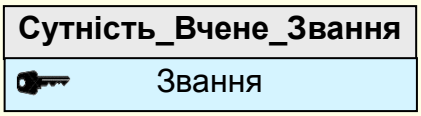


6. **Сутність_Викладач** характеризується атрибутами: прізвище, ім’я і по- батькові викладача (**ПІБ_Викладач**), який є унікальним, тому його можна призначити ключем.

Оскільки однакову посаду або вчене звання на кафедрі можуть мати декілька викладачів (наприклад, старший викладач або кандидат наук, доцент), то такі відомості обов’язково будуть повторюватися. Тому доцільно відокремити їх в самостійні сутності.

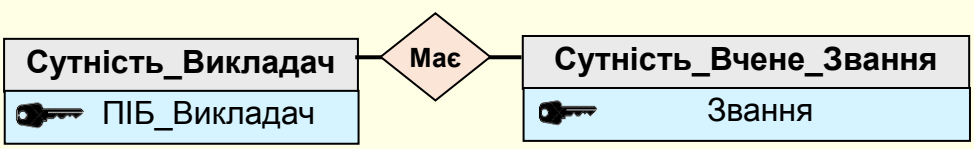


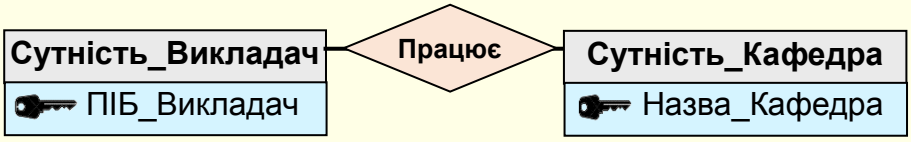
7. **Сутність_Посада** характеризується атрибутом назва посади **Посада**, який є ключем сутності.



8. **Сутність_Вчене_Звання** характеризується атрибутом **Звання**, який є ключем сутності.

Ці сутності зв’язані між собою:



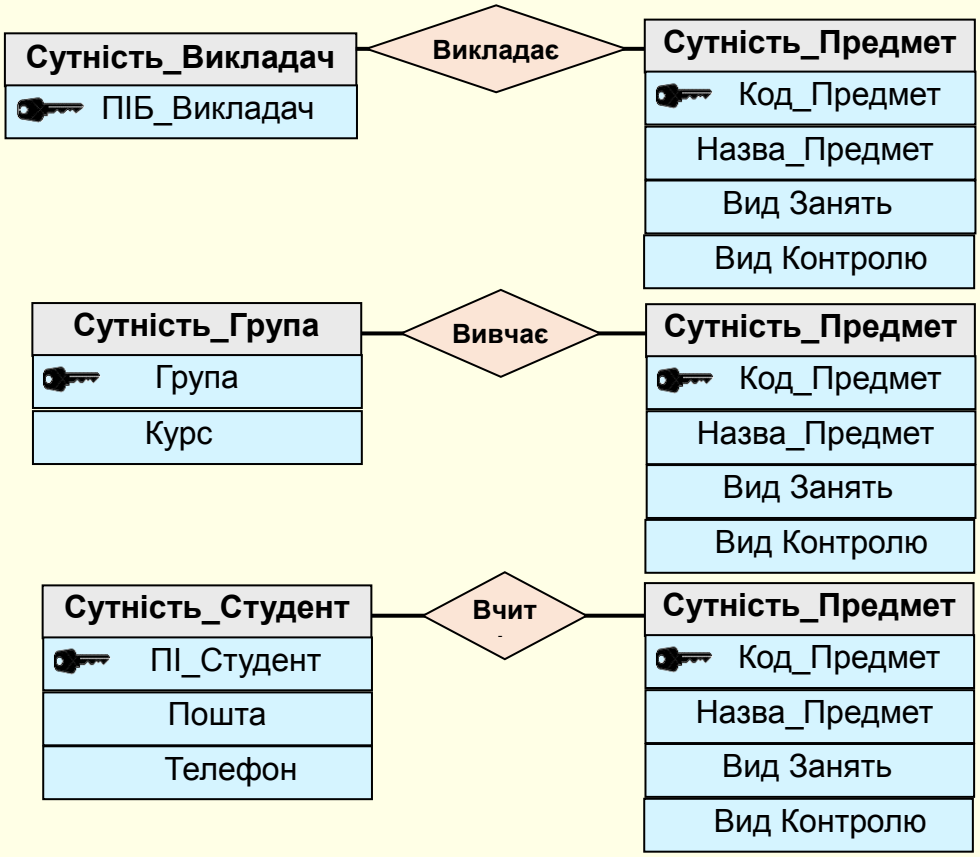


Документ ***План проведення занять*** має дані про сутності (рис. 40).

9. **Сутність_Предмет** характеризується атрибутами: назва предмету (**Назва_Предмет**); вид занять (**Вид_Занять**); вид контролю (**Вид_Контролю**). По одному предмету можуть проводитися кілька видів занять (і лекції, і практичні заняття), крім цього, назва предмету може бути довгою, тому для спрощення і прискорення ідентифікації доцільно ввести атрибут код предмету (**Код_Предмет**), який буде ключем сутності.

Документ ***Екзаменаційна відомість*** утримує дані про сутності, які вже існують (рис. 41).

Ці сутності зв’язані між собою:



Побудова діаграм ER-типу з урахуванням усіх сутностей та їхніх зв'язків

Метод сутність-зв’язок розглядає зв’язки між сутностями і полягає в побудові діаграм ER-екземплярів, які графічно показують, як зв’язані між собою окремі екземпляри різних сутностей. На основі діаграм ER-екземплярів будуються діаграми ER-типів, які показують клас приналежності екземплярів сутності у зв’язку (обов’язковий або необов’язковий) і ступінь зв’язку між

екземплярами сутностей (1:1, 1:Б, Б:Б). Діаграма ER-типів дозволяє застосувати певні формальні правила для визначення кількості таблиць, їх зв'язків і ключових атрибутів для відображення в БД сутностей, зв'язок між якими розглядається.

Зв'язок між сутностями **Студент** і **Залікова_Книжка.**

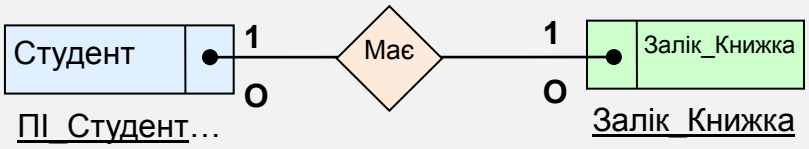
Всі студенти мають залікову книжку, тому між сутностями **Студент** і **Залікова_Книжка** є зв'язок ***Має***.

Для аналізу цього зв'язку побудовані ER-діаграми. Для спрощення на діаграмах показані тільки ключові атрибути.

Діаграма ER-екземплярів

Студент	***Має***	**Залік_Книжка**
Ведмідь		МС-0001
Вовк		МС-0002
Лось		МС-0003
Лелека		МС-0004
Ластівка		МС-0005
Синиця		МС-0006

Діаграма ER-типів



З **ER–** діаграм видно, що всі студенти обов'язково мають залікові книжки і не існує залікових книжок, до яких би не відносився жоден студент, тому клас приналежності обох сутностей обов'язковий.

Певний студент може мати тільки одну залікову книжку, так само як кожна залікова книжка може належати лише одному студенту, тому ступінь зв'язку між цими сутностями 1:1. Цей випадок підпадає під дію правила 1.

Правило 1.

Якщо ступінь зв'язку між сутностями 1:1 і клас приналежності обох сутностей обов'язковий, то формується одна таблиця. Первинним ключем таблиці може бути ключ будь-якої з двох сутностей.

Необхідно сформувати одну таблицю **Табл_Студент** з атрибутами:

– ПІ_Студент – тому можна призначити ключем.
– **Пошта, Телефон** – неключові атрибути.

Атрибут ***Залік_Книжка*** з сутності **Залікова_Книжка** додано як атрибут до таблиці **Табл_Студент**.

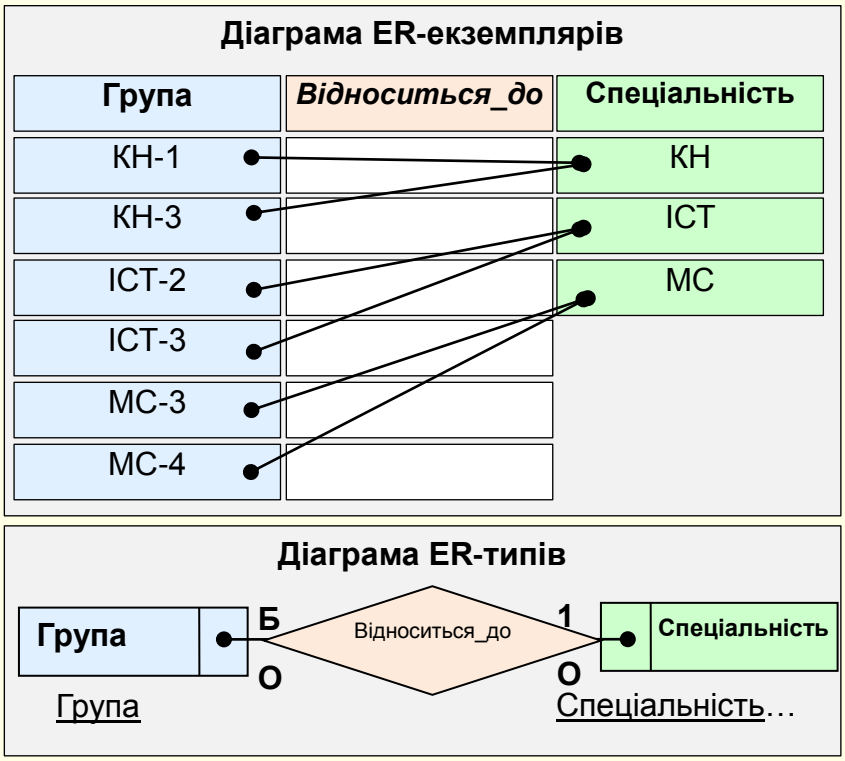
Атрибут ***ПІ_Студент*** обрано ключем **Табл_Студент**.

Табл_Студент
ПІ_Студент
Залік_Книжка
Пошта
Телефон

Зв'язок між сутностями **Група** і **Спеціальність.**

Всі групи відносяться до певної спеціальності, тому сутності **Група** і **Спеціальність** мають зв'язок ***Відноситься_до***.

Для аналізу цього зв'язку побудовані ER-діаграми. Для спрощення на діаграмах показані тільки ключові атрибути.



З **ER–** діаграм видно, що всі групи обов'язково відносяться до певної спеціальності і не існує спеціальностей, до яких не відноситься жодна група, тому клас приналежності обох сутностей обов'язковий.

До певної спеціальності може відноситися одна або декілька груп, але кожна група може входити тільки до однієї спеціальності, тому ступінь зв'язку між цими сутностями **Б:1**. Цей випадок підпадає під дію правила 4.

Правило 4.

Якщо ступінь зв'язку між сутностями 1:Б і клас приналежності Б-зв'язної сутності обов'язковий, то достатньо формування двох таблиць (по одній на кожну із сутностей). При цьому первинними ключами цих таблиць є ключі їхніх сутностей. Крім цього, ключ однозв'язної сутності додається як атрибут (зовнішній ключ) у таблицю, що відповідає Б-зв'язної сутності.

Для сутності **Спеціальність** - **Табл_Спеціальність** з атрибутом**:**

– Спеціальність – не повторюється, тому можна обрати ключем.

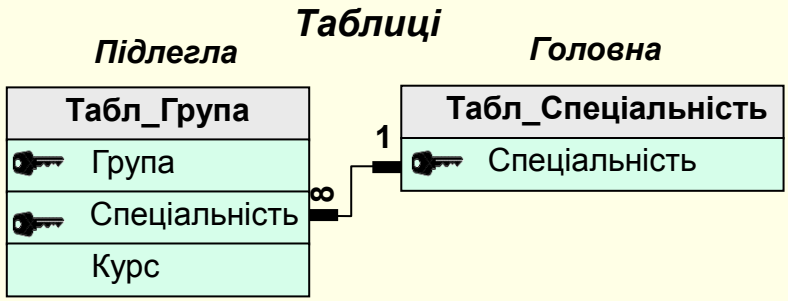
Табл_Спеціальность є головною таблицею.

Для сутності **ГРУПА** - **Табл_Група** з атрибутами**:**

- **Група, *Спеціальність*** (складений ключ);
- **Курс** (неключові атрибути);

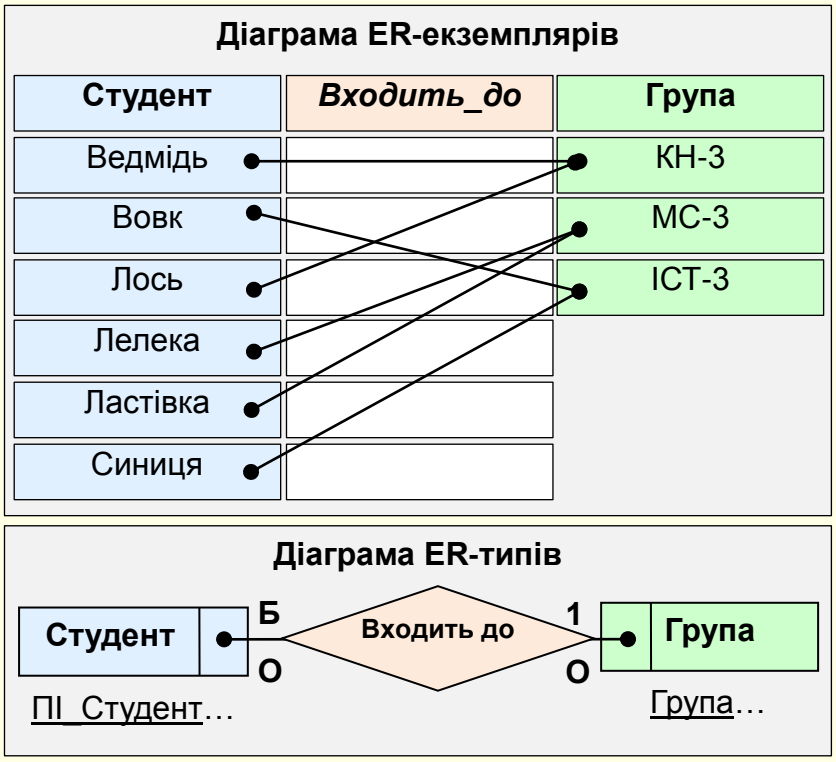
Табл_Група є залежною таблицею.

Атрибут ***Спеціальність*** з таблиці **Табл_Спеціальність** додано як частину складеного ключа до таблиці **Табл_Група**.



Зв'язок між сутностями **Студент** і **Група**

Всі студенти входять до складу певної групи, тому сутності **Студент** і **Група** мають зв'язок між собою ***Входить до***. Для аналізу цього зв'язку побудуємо ER-діаграми. Для спрощення на діаграмах показані тільки ключові атрибути.



З діаграм видно, що всі студенти обов'язково входять до складу певної групи, та не існує груп, до складу яких не входить жоден студент, тому клас приналежності обох сутностей обов'язковий.

До складу певної групи входить декілька студентів, але кожний студент може входити тільки до складу однієї групи, тому ступінь зв'язку між цими сутностями **Б:1**. Цей випадок підпадає під дію правила 4, згідно з яким необхідно сформувати дві таблиці:

Для сутності **Група** – **Табл_Група** з атрибутами:

– **Група, Спеціальність** (складений ключ);

– **Курс** (неключовий атрибут);

Табл_Група є основною таблицею.

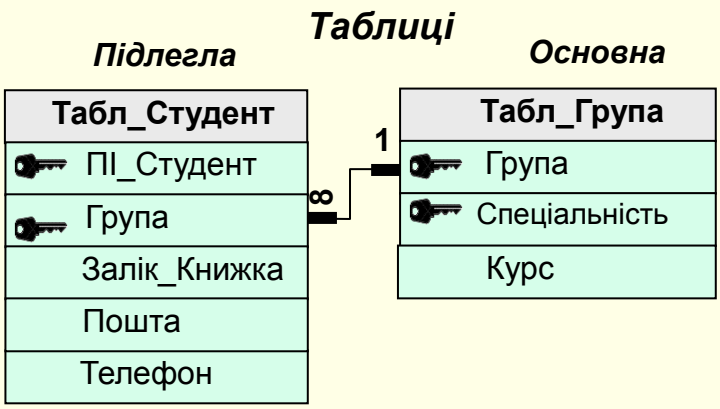
Для сутності **Студент** - **Табл_Студент** з атрибутами**:**

– **ПІ_Студент, Група** (складений ключ)**;**

– **Залік_Книжка, Пошта, Телефон** (неключові атрибути);

Табл_Студент є залежною таблицею.

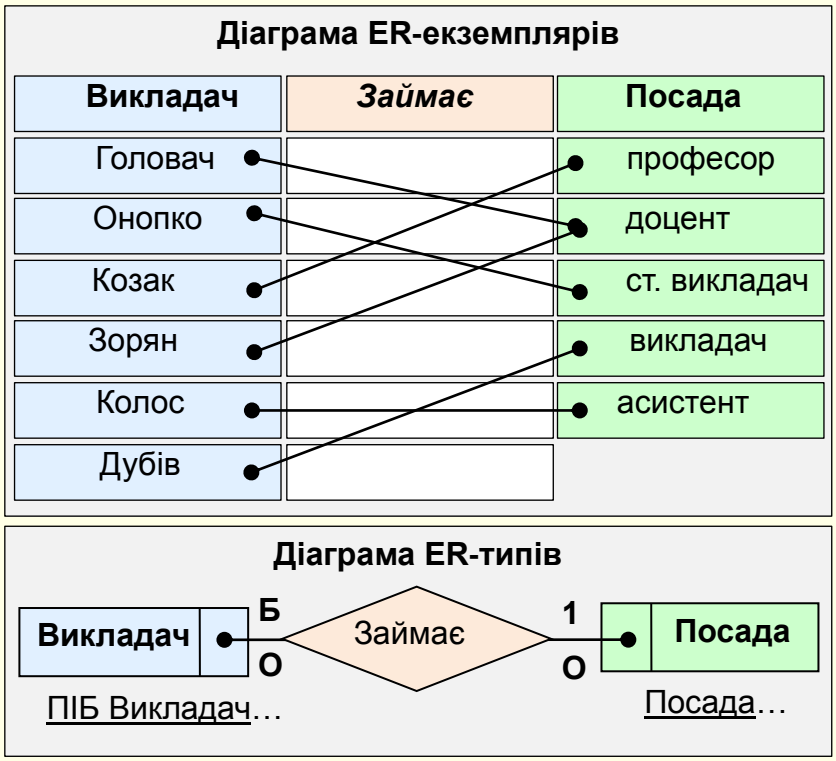
Атрибут **Група** з таблиці **Табл_Група** додано як атрибут до таблиці **Табл_Студент**. Тут він є зовнішнім ключем та за умови забезпечення цілісності даних має входити до складеного ключа. Крім того, дані для поля **Група** таблиці **Табл_Студент** необхідно обирати з відповідного поля **Група** таблиці **Табл_Група.**



Табл_Група і **Табл_Студент** зв'язані по полю **Номер_Група**.

Зв'язок між сутностями **Викладач** і **Посада.**

Всі викладачі займають певні посади, тому сутності **Викладач** і **Посада** мають зв'язок між собою ***Займає***. Для аналізу цього зв'язку побудуємо ER-діаграми. Для спрощення на діаграмах показані тільки ключові атрибути.



З діаграм видно, що всі викладачі обов'язково займають певну посаду і не існує посад, які не займані жодним викладачем, тому клас приналежності обох сутностей обов'язковий.

Однакову посаду можуть займати декілька викладачів, але кожний викладач може займати тільки одну посаду, тому ступінь зв'язку між цими сутностями **Б:1**. Цей випадок підпадає під дію правила 4, згідно з яким необхідно сформувати дві таблиці:

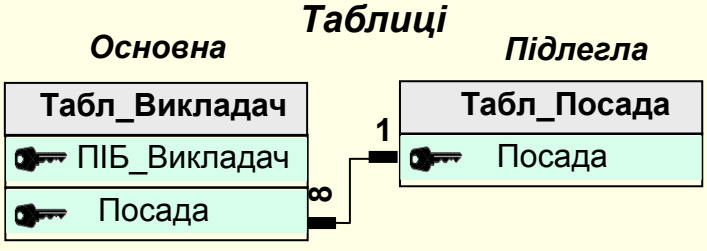
Для сутності **Посада** – **Табл_Посада** з атрибутом **Посада** (ключ). **Табл_Посада** є основною таблицею.

Для сутності **Викладач** - **Табл_Викладач** з атрибутами**:**

– **ПІБ_Викладач, Посада** (складений ключ);

Табл_Викладач є залежною таблицею.

Атрибут **Посада** з таблиці **Табл_Посада** додано як атрибут до таблиці **Табл_Викладач**. Тут він є зовнішнім ключем та за умови забезпечення цілісності даних має входити до складеного ключа. Крім цього, дані для поля **Посада** таблиці **Табл_Викладач** необхідно обирати з відповідного поля **Посада** таблиці **Табл_Посада.**

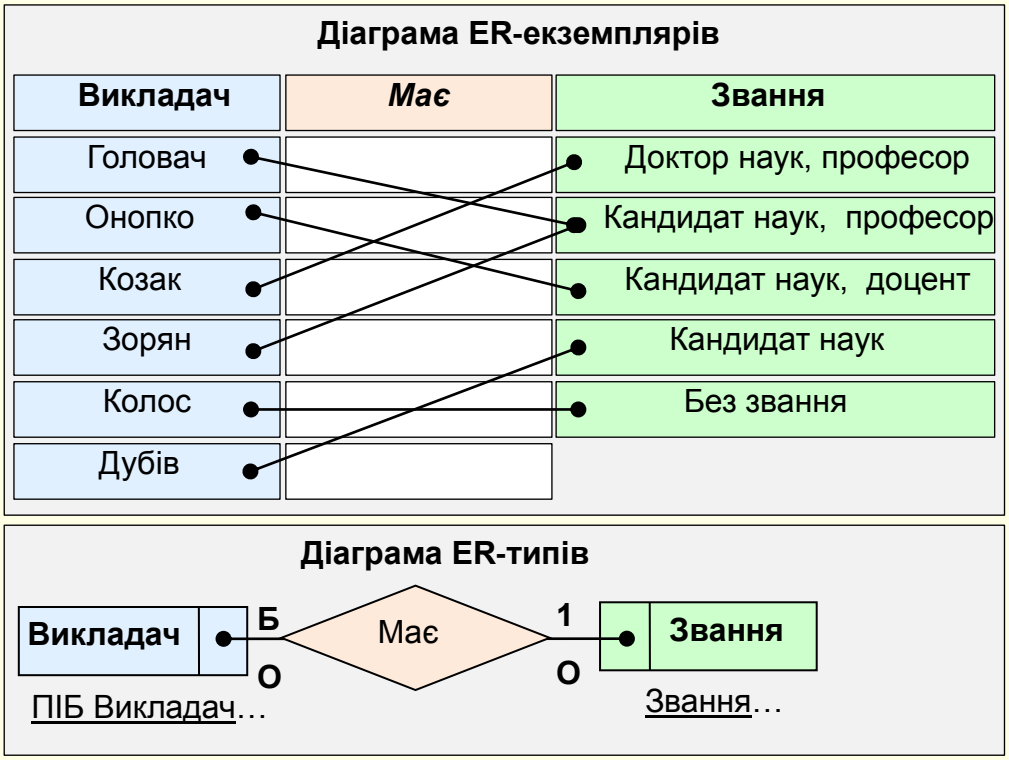


Табл_Посада і **Табл_Викладач** зв'язані по полю **Посада.**

Зв'язок між сутностями **Викладач** і **Звання.**

Всі викладачі мають певне вчене звання, тому сутності **Викладач** і **Звання** мають зв'язок між собою ***Має***. Для аналізу цього зв'язку побудуємо ER-діаграми. Для спрощення на діаграмах показані тільки ключові атрибути.

З діаграм видно, що всі викладачі обов'язково мають певне вчене звання і не існує вчених звань, які б не належали жодному викладачу, тому клас приналежності обох сутностей обов'язковий.



Однакове вчене звання можуть мати декілька викладачів, але кожний викладач може мати тільки одне вчене звання, тому ступінь зв'язку між цими сутностями **Б:1**. Цей випадок підпадає під дію правила 4.

Необхідно сформувати дві таблиці:

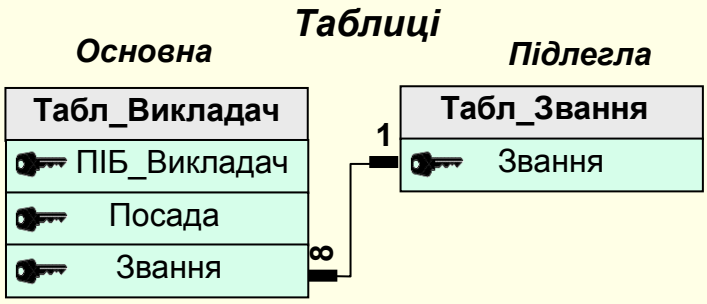
Для сутності **Вчене_Звання** – **Табл_Звання** з атрибутом **Звання** (ключ). **Табл_Звання** є основною таблицею.

Для сутності **ВИКЛАДАЧ** - **Табл_Викладач** з атрибутами**:**

– **ПІБ_Викладач, Посада, Звання** (складений ключ).

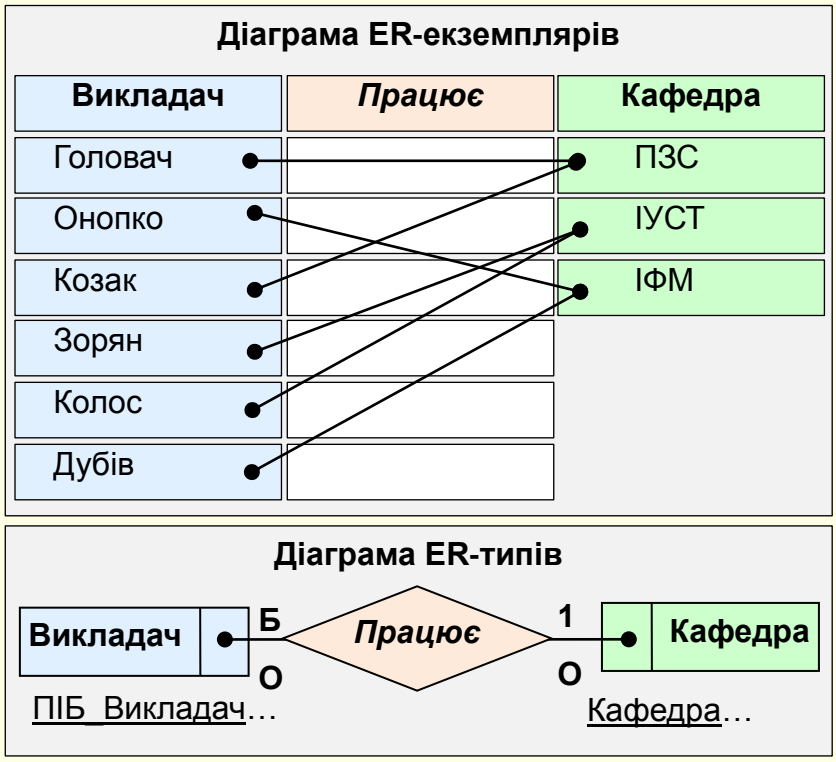
Табл_Викладач є залежною таблицею.

Атрибут **Звання** з таблиці **Табл_Звання** додано як атрибут до таблиці **Табл_Викладач**. Тут він є зовнішнім ключем та за умови забезпечення цілісності даних має входити до складеного ключа. Крім цього, дані для поля **Звання** таблиці **Табл_Викладач** необхідно обирати з відповідного поля **Звання** таблиці **Табл_Звання.**



Зв'язок між сутностями **Викладач** і **Кафедра**

Всі викладачі працюють на певних кафедрах, тому сутності **Викладач** і **Кафедра** мають зв'язок між собою ***ПРАЦЮЄ***. Для аналізу цього зв'язку побудуємо ER-діаграми. Для спрощення на діаграмах показані тільки ключові атрибути, а прізвища викладачів замінені табельними номерами**.**



З діаграм видно, що всі викладачі обов'язково працюють на певній кафедрі і не існує кафедр, на яких би не працювали викладачі, тому клас приналежності обох сутностей обов'язковий.

На певній кафедрі може працювати декілька викладачів, але кожний викладач може працювати тільки на одній кафедрі, тому ступінь зв'язку між цими сутностями **Б:1**. Цей випадок підпадає під дію правила 4.

Необхідно сформувати дві таблиці:

1. Для сутності **Кафедра** – **Табл_Кафедра** з атрибутами:

– **Кафедра** (ключ);

Табл_Кафедра є основною таблицею.

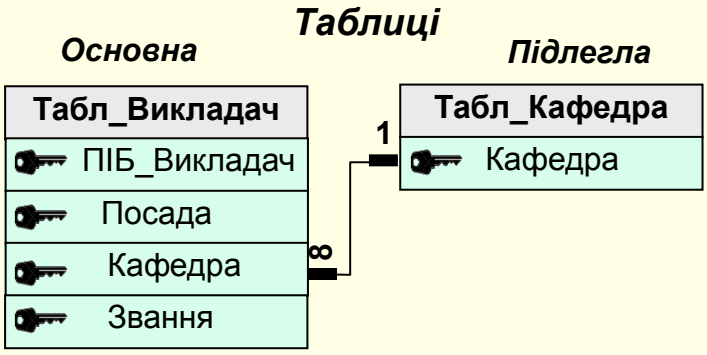
2. Для сутності **Викладач** - **Табл_Викладач** з атрибутами**:**

– **ПІБ_Викладач, Посада, Звання, Кафедра** (складений ключ).

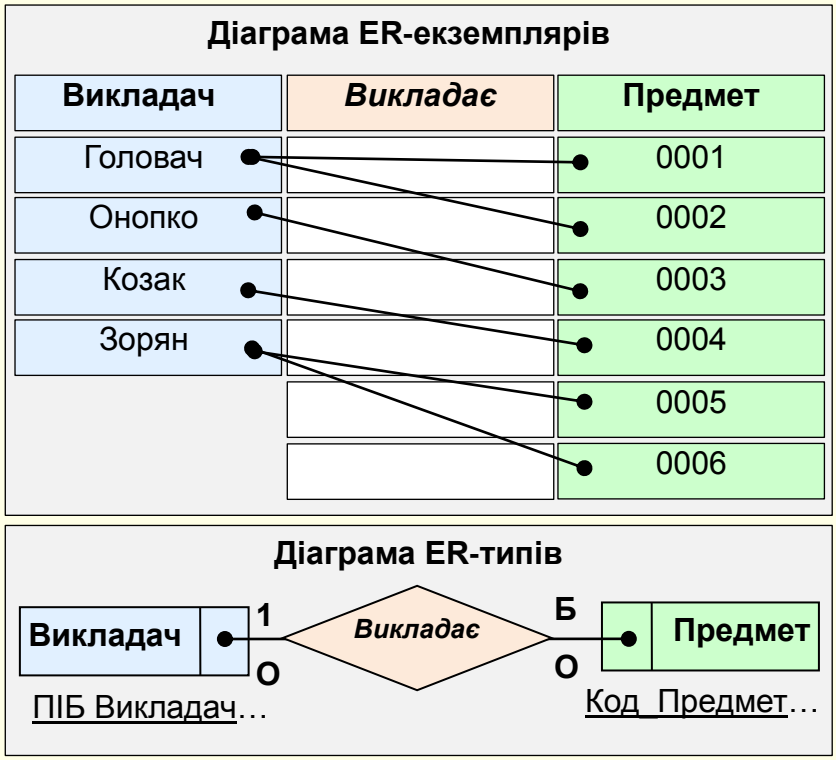
Табл_Викладач є залежною таблицею.

Атрибут **Код_Кафедра** з таблиці **Табл_Кафедра** додано як атрибут до таблиці **Табл_Викладач**. Тут він є зовнішнім ключем та за умови забезпечення цілісності даних має входити до складеного ключа. Крім цього, дані для поля **Кафедра** таблиці **Табл_Викладач** необхідно обирати з відповідного поля **Кафедра** таблиці **Табл_Кафедра.**

Табл_Кафедра і **Табл_Викладач** зв'язані по полю **Код_Кафедра.**



Зв'язок між сутностями **Викладач** і **Предмет.** Всі викладачі викладають певні предмети, тому сутності **Викладач** і **Предмет** мають між собою зв'язок ***Викладає***. Для аналізу цього зв'язку побудуємо ER-діаграми.



Для спрощення на діаграмах показані тільки ключові атрибути.

З діаграм видно, що всі викладачі обов'язково викладають певний предмет і не існує предметів, які не викладаються жодним викладачем, тому клас приналежності обох сутностей обов'язковий.

Кожний викладач може викладати декілька предметів, але кожний вид занять (лекції, лабораторні, практичні, предмет і вид занять мають певний код) може проводити тільки один викладач, тому ступінь зв'язку між цими сутностями **Б:1**. Випадок підпадає під дію правила 4.

Необхідно сформувати дві таблиці:

1. Для сутності **ВИКЛАДАЧ** - **Табл_Викладач** з атрибутами**:**

– **ПІБ_Викладач, Кафедра, Посада, Звання** (складений ключ);

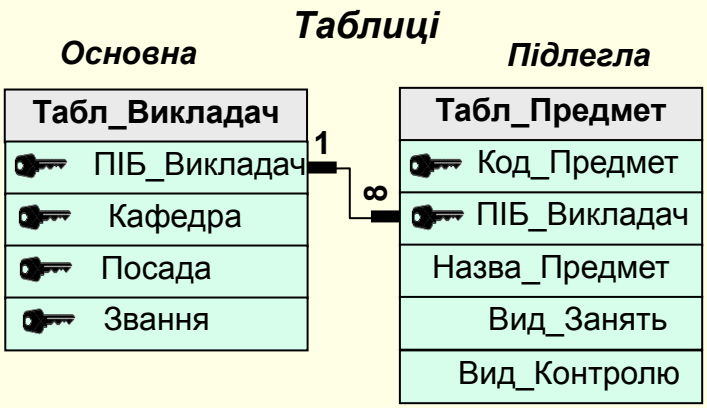
Табл_Викладач є головною таблицею.

У відповідності з правилом ключ однозв'язної сутності **Викладач** (**ПІБ_Викладач**) необхідно додати у таблицю, що відповідає багатозв'язній сутності.

2. Для сутності **Предмет** – **Табл_Предмет** з атрибутами:

– **Код_Предмет, ПІБ_Викладач** (складений ключ);

– **Назва_Предмет, Вид_Занять, Вид_Контролю** (неключові атрибути).



Введення поля **Код_Предмет** пояснюється тим, що з певного предмету проводяться різні види занять (лекції, лабораторні, практичні тощо) та різні види контролю (залік, іспит). Тому обрати у якості ключа тільки назву предмета не можна.

Табл_Викладач є головною таблицею, а **Табл_Предмет** – залежною, тому дані для поля **ПІБ_Викладач** таблиці **Табл_Предмет** необхідно обирати з відповідного поля **ПІБ_Викладач** таблиці **Табл_Викладач.** Тут ключ є зовнішнім та за умови забезпечення цілісності даних має входити до складеного ключа. Таблиці зв'язані по полю **ПІБ_Викладач**.

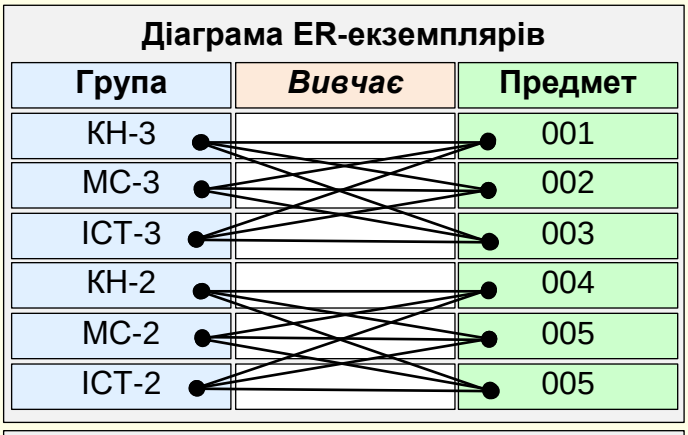
Зв'язок між сутностями **Група** і **Предмет**

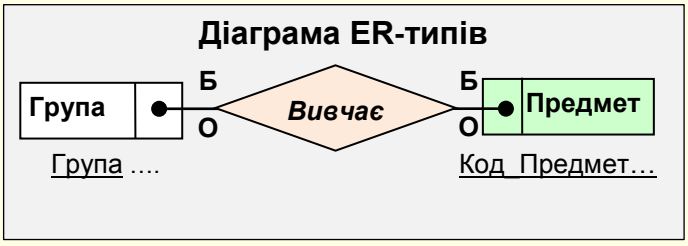
Студенти кожної групи вивчають певні предмети відповідно до обраної спеціальності. Тому сутності **Група** і **Предмет** мають зв'язок ***Вивчає***. Цей зв'язок відображено і проаналізовано на ER-діаграмах. Для спрощення на діаграмах показані тільки ключові атрибути.

З діаграми видно, що студенти кожної групи вивчають багато предметів, так само як один і той же предмет також можуть вивчати в декількох групах багато студентів, наприклад, англійську мову в групах різних спеціальностей. Тобто клас приналежності обох сутностей обов'язковий, і ступінь зв'язку між цими сутностями **Б:Б**. Цей випадок підпадає під дію правила 6.

Правило 6.

Якщо ступінь зв'язку Б : Б, то незалежно від класу приналежності сутностей формуються три таблиці. Дві таблиці відповідають сутностям, що зв'язуються, і їхні ключі є первинними ключами цих сутностей. Третя таблиця є зв'язковою, а її ключ поєднує ключові атрибути таблиць, що зв'язуються.





Відповідно до правила необхідно сформувати три таблиці.

1) для сутності **Група** - **Табл_Група** з атрибутами:

– **Група, Спеціальність** (складений ключ);

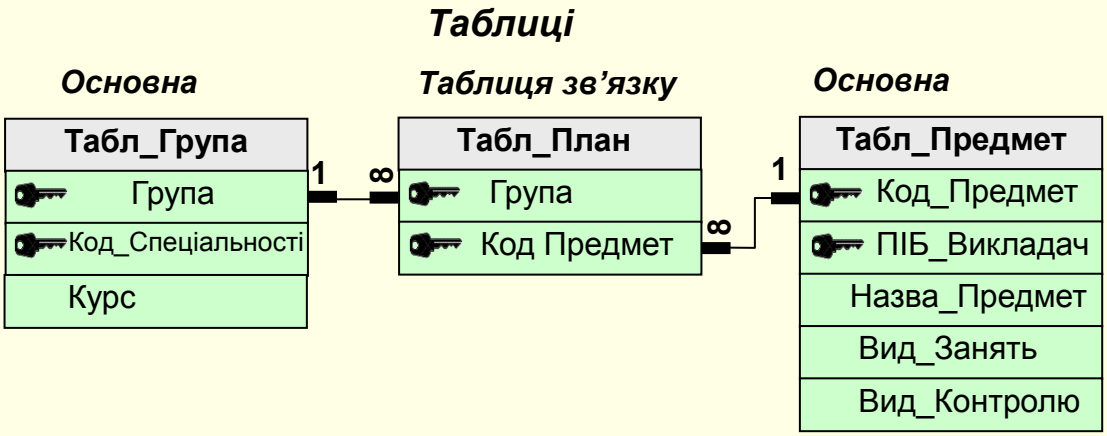
– **Курс** (неключовий атрибут);

2) для сутності **Предмет** – **Табл_Предмет** з атрибутами:

– **Код_Предмет, ПІБ_Викдадач** (складений ключ);

– **Назва_Предмет, Вид_Занять, Вид_Контролю** (неключові атрибути);

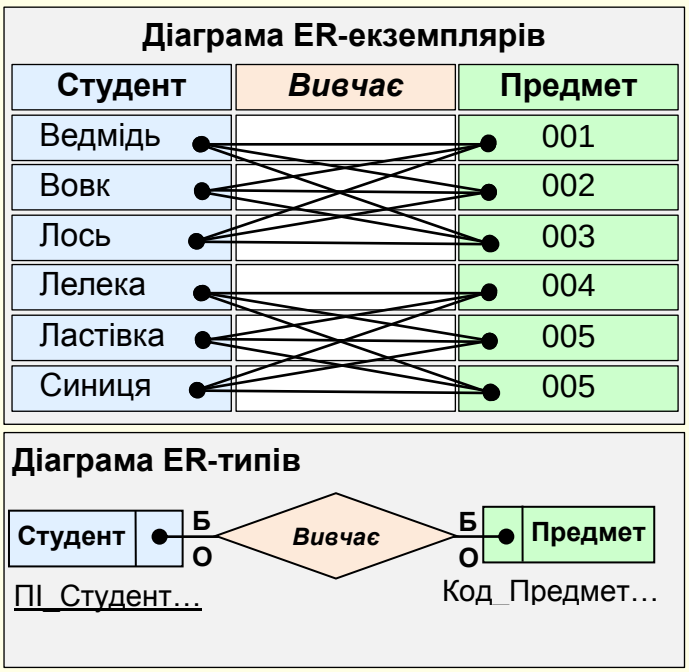
3) таблиця зв'язку **Табл_План** з атрибутами: **Група, Код_Предмет**. Сукупність цих атрибутів утворює складений ключ. Дані для поля **Група** таблиці **Табл_План** необхідно обирати з відповідного поля **Група** таблиці **Табл_Група**, а дані для поля **Код_Предмет** таблиці **Табл_План** необхідно обирати з відповідного поля **Код_Предмет** таблиці **Табл_Предмет** Сукупність цих атрибутів утворює зовнішній ключ.



Зв'язок між сутностями **Студент** і **Предмет**

Всі студенти вивчають певні предмети. Тому сутності **Студент** і **Предмет** зв'язані між собою зв'язком ***Вивчає.*** На ER-діаграмі відображено та проаналізовано цей зв'язок. Для спрощення на діаграмах показані тільки ключові атрибути.

З діаграми видно, що кожний студент обов'язково вивчає багато предметів, так само як той самий предмет також обов'язково вивчають багато студентів. Тобто клас приналежності обох сутностей обов'язковий, і ступінь зв'язку між цими сутностями **Б:Б**. Цей випадок підпадає під дію правила 6, відповідно до якого необхідне формування трьох таблиць. Дві таблиці відповідають сутностям, що зв'язуються, і їхні ключі є первинними ключами цих сутностей. Третя таблиця є зв'язковою між першими двома, а її ключ поєднує ключові атрибути таблиць, що зв'язуються. Таким чином, остаточно для збереження інформації про ці сутності необхідно сформувати три таблиці:



1) для сутності **Студент** - **Табл_Студент** з атрибутами:

– **ПІ_Студент, Група** (складений ключ)**;**

– **Залік_Книжка, Пошта, Телефон** (неключові атрибути);

2) для сутності **ПРЕДМЕТ** – **Табл_Предмет** з атрибутами:

– **Код_Предмет, ПІБ_Викдадач** (складений ключ);

– **Назва_Предмет, Вид_Занять, Вид_Контролю** (неключові атрибути).

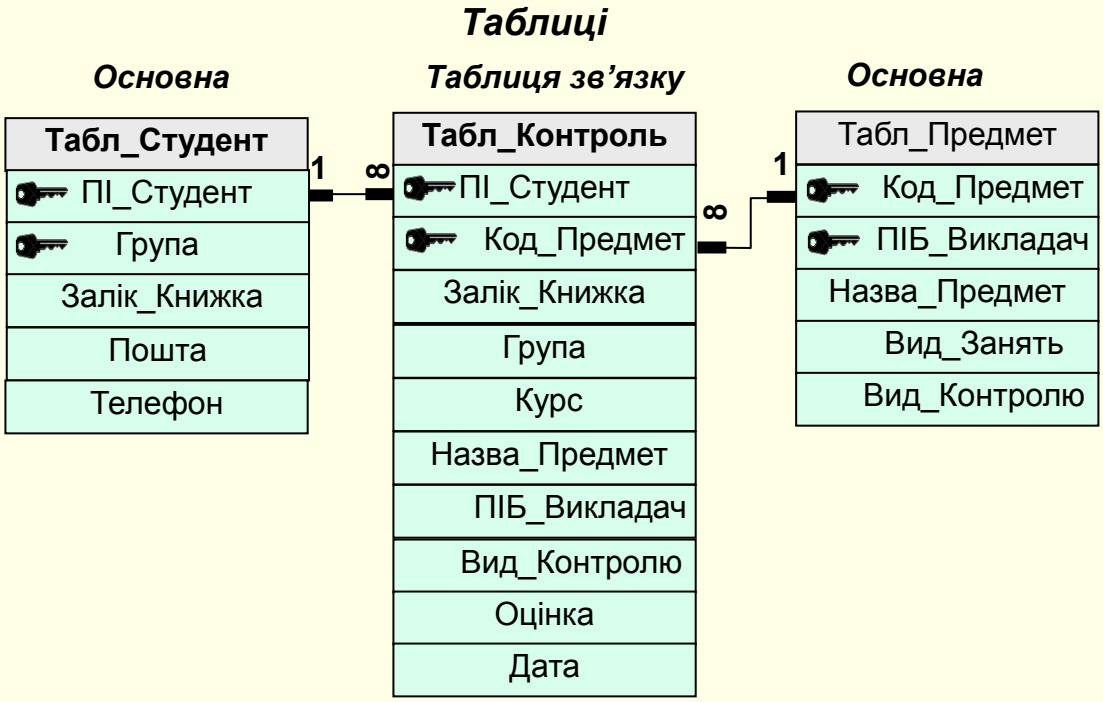
3) таблиця зв'язку **Табл_Контроль** з атрибутами:

– **ПІ_Студент, Код_Предмет**. Сукупність цих

атрибутів утворює складений ключ.

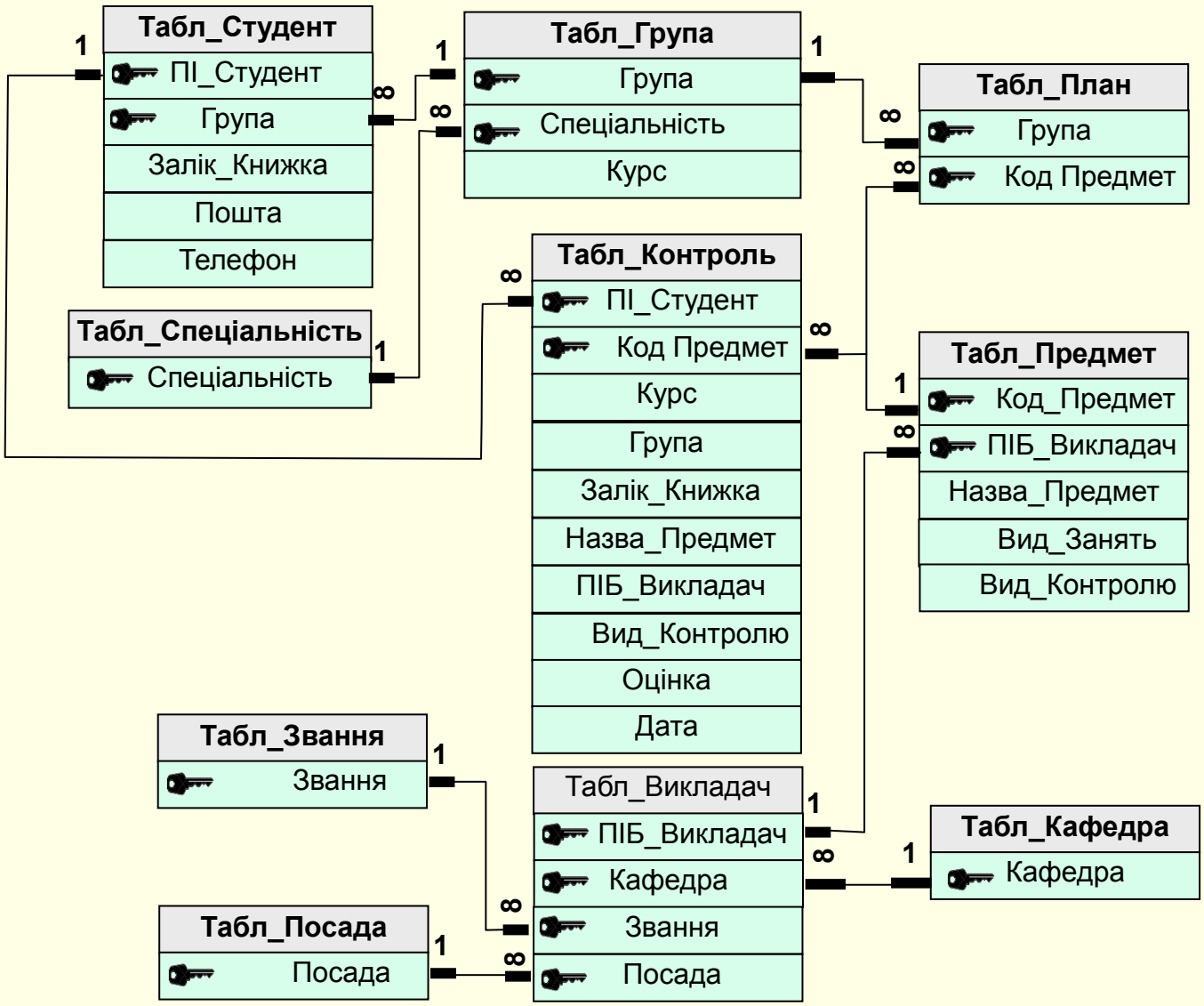
– Якщо додати до цієї таблиці атрибути **Залік_Книжка, Оцінка**, **Дата**, **Курс**, **Група**, **Назва_Предмет**, **ПІБ_Викладач**, **Вид_Контролю,** то її можна використовувати як екзаменаційну відомість.

Дані для поля **ПІ_Студент** таблиці **Табл_Контроль** необхідно обирати з відповідного поля **ПІ_Студент** таблиці **Табл_Студент**, а дані для поля **Код_Предмет** таблиці **Табл_Контроль** необхідно обирати з відповідного поля **Код_Предмет** таблиці **Табл_Предмет.**



Інформаційно-логічна модель бази даних

Проєкт БД розроблено методом сутність-зв'язок. У результаті отримана Інформаційно-логічна модель БД, яка наочно відображає всі таблиці, їх ключі та інші поля, а також зв'язки між ними таблицями. У загальному випадку на боці зв'язку 1 відображаються поля головних таблиць, а на боці зв'язку Б (∞) – поля залежних таблиць.



Перевірка таблиць на відповідність нормальним формам

Перевірка таблиць БД на відповідність нормальним формам є важливим етапом проєктування, який дозволяє переконатися у тому, що запроєктована база даних має правильну структуру. Якщо деякі таблиці не відповідають нормальним формам, то необхідно повернутися до початкових етапів проєктування для внесення в базу відповідних змін.

У загальному випадку таблиці мають відповідати третій нормальній формі та формі Бойса-Кодда, які вимагають щоб:

– всі неключові атрибути таблиці були взаємно незалежні і цілком залежали від первинного ключа.

– в таблиці були відсутні залежності ключів (атрибутів складеного ключа) від неключових атрибутів.

Головні таблиці

Спочатку розглянемо головні таблиці ті що, знаходяться на стороні зв'язку 1. Поле на боці такого зв'язку є первинним ключем.

Таблиця **Табл_Посада**, атрибут **Посада** (ключ). Оскільки неключові атрибути відсутні, то можна стверджувати, що ключ не залежить від неключового атрибута, що відповідає нормальній формі Бойса-Кодда.

Табл_Посада
Посада

Таблиця **Табл_Звання**, атрибут **Звання** (ключ). Оскільки неключові атрибути відсутні, то можна стверджувати, що ключ не залежить від не ключового атрибута, що відповідає нормальній формі Бойса-Кодда.

Табл_Звання
Звання

Таблиця **Табл_Спеціальність**, атрибут **Спеціальність** (ключ). Оскільки неключові атрибути відсутні, то можна стверджувати, що ключ не залежить від неключового атрибута, що відповідає нормальній формі Бойса-Кодда

Табл_Спеціальність
Спеціальність

Таблиця **Табл_Кафедра**, атрибут **Кафедра** (ключ). Оскільки неключові атрибути відсутні, то можна стверджувати, що ключ не залежить від неключового атрибута, що відповідає нормальній формі Бойса-Кодда.

Табл_Кафедра
Кафедра

Залежні таблиці

Залежні таблиці, які знаходяться на стороні зв'язку ∞. Поле на боці такого зв'язку є зовнішнім ключем. Виходячи з умов забезпечення цілісності даних, воно має входити до складеного ключа.

Таблиця **Табл_Група**, атрибути:

Група, **Спеціальність** – складений ключ, **Курс** – неключовий атрибут.

Оскільки неключовий атрибут **Курс** тільки один, то можна стверджувати, що неключові атрибути не залежать один від одного. Складений ключ (Група, Спеціальність) може бути будь-яким незалежно від Курсу. Тобто ключ не залежить від неключового атрибута, що відповідає нормальній формі Бойса-Кодда.

Табл_Група
Група
Код_Спеціальності
Курс

Таблиця **Табл_Студент**, атрибути:

ПІ_Студент і **Група** (складений ключ);

Залік_Книжка, Пошта, **Телефон** – неключові атрибути.

Можна стверджувати, що неключові атрибути не залежать один від одного. У той же час всі вони відповідають прізвищу студента. Тобто можна стверджувати, що неключові атрибути залежать від ключа або його частини. Це відповідає третій нормальній формі.

Табл_Студент
ПІ_Студент
Група
Залік_Книжка
Пошта
Телефон

Таблиця **Табл_Викладач**, атрибути: **ПІБ_Викладач**, **Код_Кафедра**, **Посада**, **Звання** – сукупність цих атрибутів утворює складений ключ.

Оскільки неключові атрибути відсутні, то таблиця відповідає третій нормальній формі.

Табл_Викладач
ПІБ_Викладач
Посада
Код_Кафедра
Звання

Таблиця **Табл_Предмет**, атрибути: **Код_Предмет, Таб_Номер** (складений ключ);

– **Назва_Предмет, Вид_Занять, Вид_Контролю** (неключові атрибути). Можна стверджувати, що вони не залежать один від одного і цілком залежать від частини складеного ключа **Код_Предмет**, що відповідає третій нормальній формі.

Табл_Предмет
Код_Предмет
Таб_Номер
Назва_Предмет
Вид_Занять
Вид_Контролю

Таблиця **Табл_План**, атрибути **Номер_Група**, **Код_Предмет** - сукупність цих атрибутів утворює складений ключ. Оскільки неключові атрибути відсутні, то можна стверджувати, що ключ не залежить від неключового атрибута, що відповідає нормальній формі Бойса-Кодда.

Табл_План
Група
Код Предмет

Таблиця **Табл_Контроль**, атрибути: **ПІ_Студент**, **Код_Предмет** – сукупність цих атрибутів утворює складений ключ; **Залік_Книжка, Назва_Предмет, Вид_Контролю, ПІБ_Викладач, Група, Курс, Оцінка, Дата** – неключові атрибути. Можна стверджувати, що всі неключові атрибути не залежать один від одного. У той же час певна оцінка ставиться у певну залікову книжку з певного предмету у певну дату, тому можна стверджувати, що неключові атрибути залежить від ключа, що відповідає третій нормальній формі.

Таким чином, всі таблиці, які входять до запроєктованої БД, відповідають вимогам третьої нормальної форми або посиленої нормальної форми Бойса-Кодда. Це дозволяє стверджувати, що в БД відсутні аномалії, які можуть привести до помилок або суттєво ускладнити роботу з нею.

Табл_Контроль
ПІ_Студент
Код_Предмет
Залік_Книжка
Група
Курс
Назва_Предмет
ПІБ_Викладач
Вид_Контролю
Оцінка
Дата

Навчальне видання

ПРОЄКТУВАННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Навчальний посібник в електронному вигляді

Комп'ютерний набір і дизайн
В.О. Нелюбов
Коректура і технічна редакція
О.І. Дудаш

Навчальний посібник в електронному вигляді створено
на кафедрі програмного забезпечення систем ДВНЗ «УжНУ»
88015, м. Ужгород, вул. Заньковецької, 89
E-mail: it-center@uzhnu.edu.ua

Відредаговано у редакційно-видавничому відділі ДВНЗ «УжНУ»
88015, м. Ужгород, вул. Заньковецької, 89
E-mail: dep-editors@uzhnu.edu.ua