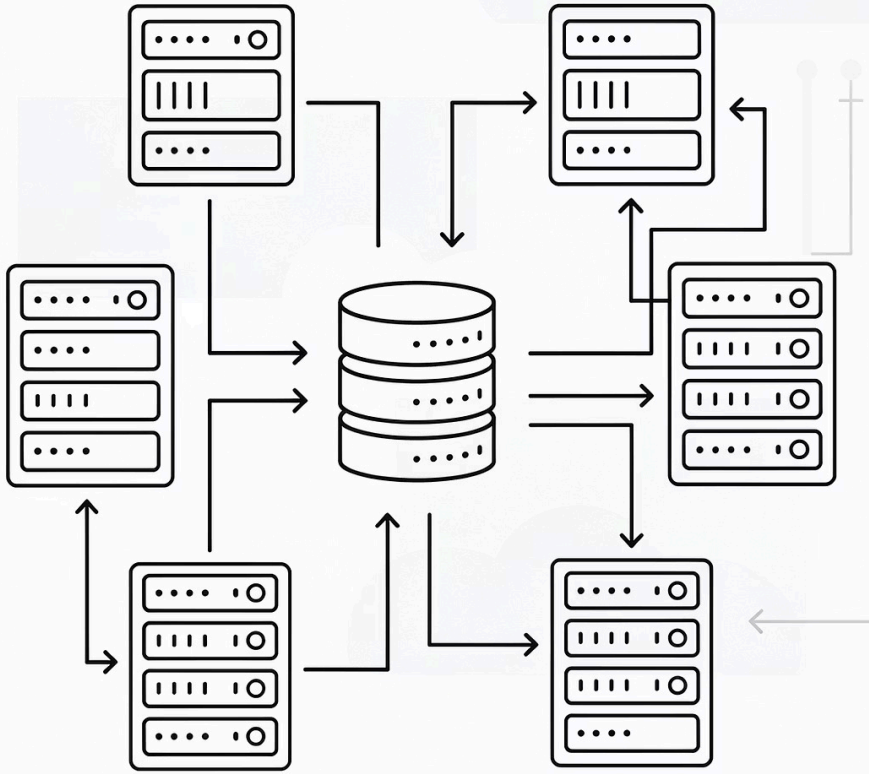
Лекція 6

Моделювання даних та предметної області



Золоте правило IT

Code rots

Код застаріває. Фронтенд переписується з Angular на React, бекенд змінюється з Java на Go.

Data remains

Дані залишаються. База клієнтів, історія транзакцій живуть 10-20 років і переживають будь-які технологічні зміни.

Міграція даних — найдорожчий і найризикованіший процес в IT. Помилка в проєктуванні даних коштує в 100 разів дорожче, ніж помилка в алгоритмі. Моделювання даних — це процес створення "креслення" для зберігання інформації.

Три рівні моделювання даних

Індустрія (стандарт ANSI/SPARC) розділяє моделювання на три рівні. Ми рухаємось від абстракції до технічної реалізації.

Концептуальна модель

Для бізнесу та стейкхолдерів. Основні сутності та зв'язки без технічних деталей.

Логічна модель

Для архітекторів та розробників. Всі атрибути, ключі, нормалізація без прив'язки до СУБД.

Фізична модель

Для DBA та DevOps. Оптимізація під конкретну СУБД з індексами та тригерами.

РІВЕНЬ 1

Концептуальна модель

Для кого

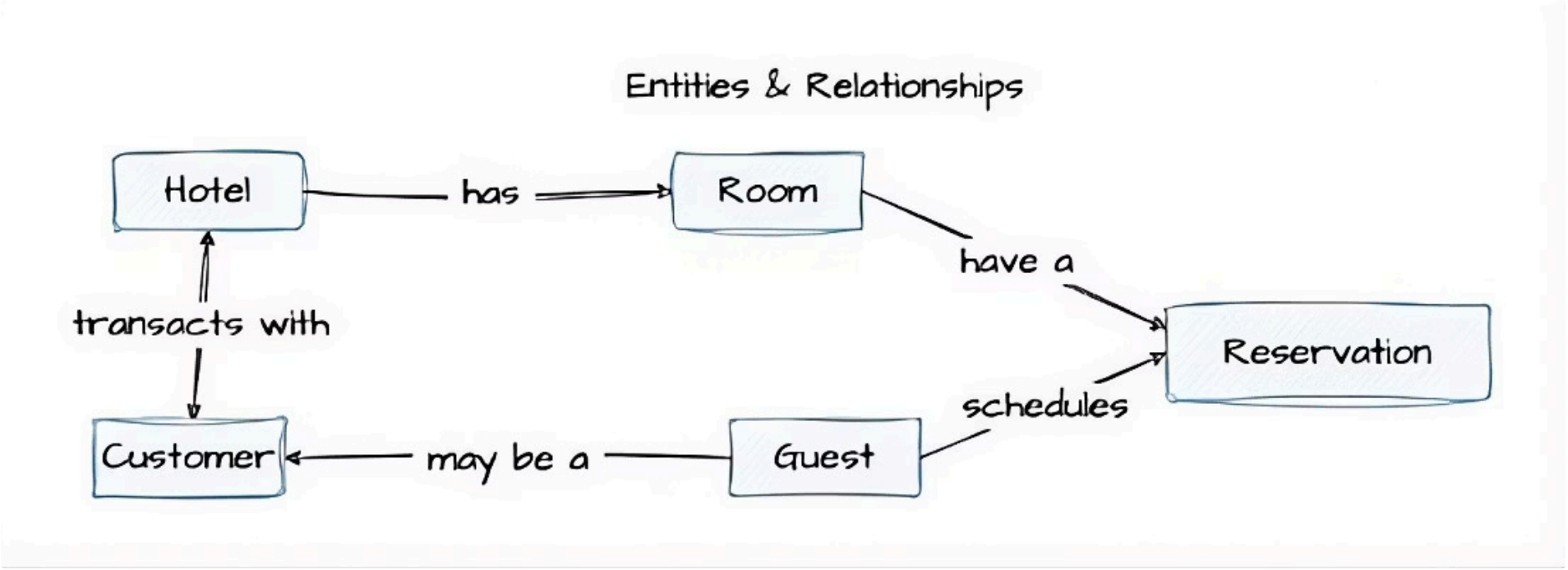
Бізнес, Стейкхолдери, Аналітики

Мета

Домовитися про терміни і зрозуміти бізнес-сутності

Що містить

Лише основні сутності та зв'язки



РІВЕНЬ 2

Логічна модель

Для кого

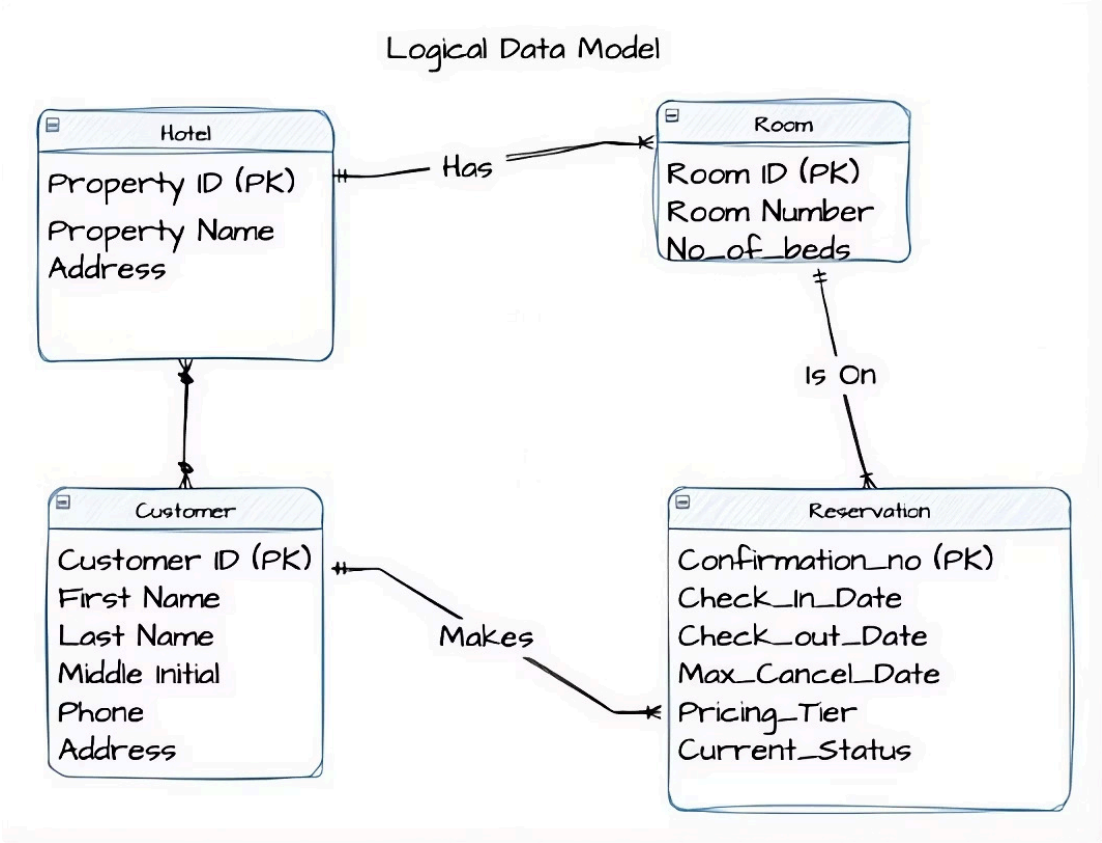
Архітектори, розробники

Мета

Описати структуру без прив'язки до конкретної СУБД (PostgreSQL чи Oracle)

Що містить

Всі атрибути, первинні ключі (PK), зовнішні ключі (FK), нормалізацію



РІВЕНЬ 3

Фізична модель

Для кого

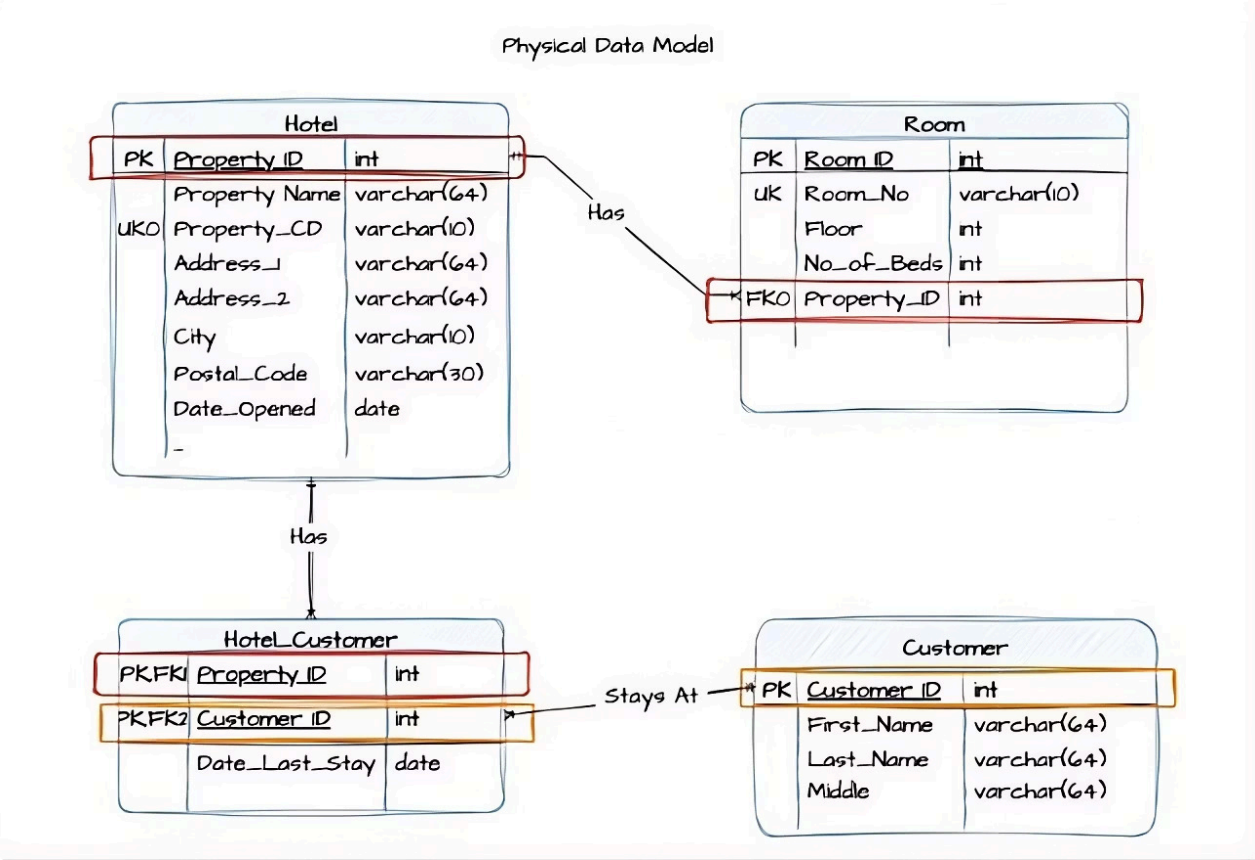
DBA (Адміністратори баз даних), DevOps

Мета

Оптимізація під конкретну СУБД

Що містить

Типи даних (varchar(255), jsonb), індекси, партиціонування, тригери



Суть концептуального моделювання

Концептуальна модель — це словник вашого проєкту. Вона описує предметну область (Domain), а не базу даних. У сучасній розробці (особливо в Domain-Driven Design — DDD) це етап формування Єдиної мови (Ubiquitous Language).

Головна проблема

Розробник

Чує слово "Клієнт" і створює таблицю users

1

Бухгалтер

Чує слово "Клієнт" і думає про "Юридичну особу з реквізитами"

3

2

Маркетолог

Чує слово "Клієнт" і думає про стратегію продажу

Якщо ви пропустите етап концептуального моделювання, ви створите систему, де в таблиці users буде каша з реальних покупців, анонімних відвідувачів і компаній.

Основні елементи концептуальної моделі

На цьому етапі ми оперуємо лише трьома поняттями:

- **Сутність (Entity)**

  Реальний об'єкт або подія, про яку важливо зберігати інформацію. Це іменники в описі задачі ("Студент", "Товар", "Замовлення"). Сутність повинна мати унікальність.

- **Атрибут (Attribute)**

  Характеристика сутності. Це прикметники або деталі ("Ім'я", "Ціна", "Колір", "Дата народження"). На цьому етапі не пишемо тип даних.

- **Зв'язок (Relationship)**

  Асоціація між сутностями. Це дієслова ("Купує", "Відвідує", "Складається з"). На концептуальному рівні дозволяємо зв'язки "Багато-до-багатьох".

Від малюнків до креслень

ERD (Entity-Relationship Diagram) — це архітектурне креслення. Це стандарт, за яким DBA може створити реальну базу даних, навіть не знаючи суті проєкту.

Нотація Пітера Чена

Ромбики для зв'язків, овали для атрибутів. Класика академічної науки 70-х. Виглядає красиво, але займає купу місця. На практиці майже не використовується.

Нотація "Вороняча лапка"

Індустріальний стандарт. Компактна, чітка, використовується у всіх сучасних інструментах (MySQL Workbench, MS SQL Management Studio).

> Ми будемо вивчати саме Crow's Foot — "Воронячу лапку".

Анатомія "Воронячої лапки"

А. Сутність (Entity)

Малюється як прямокутник, розділений на дві частини: Заголовок (назва сутності) та Тіло (список атрибутів).

PK (Primary Key)

Первинний ключ. Унікальний ідентифікатор рядка (наприклад, id, passport_num).

FK (Foreign Key)

Зовнішній ключ. Посилання на PK іншої таблиці. Це "місток", який з'єднує таблиці.

Б. Зв'язок (Relationship)

Це лінія між двома таблицями. Найважливіше — її кінці, які показують Кардинальність (відповідь на питання "Скільки?").

Символи "Воронячої лапки"

У нотації використовуються 4 символи на кінцях ліній, які комбінуються:

Риска

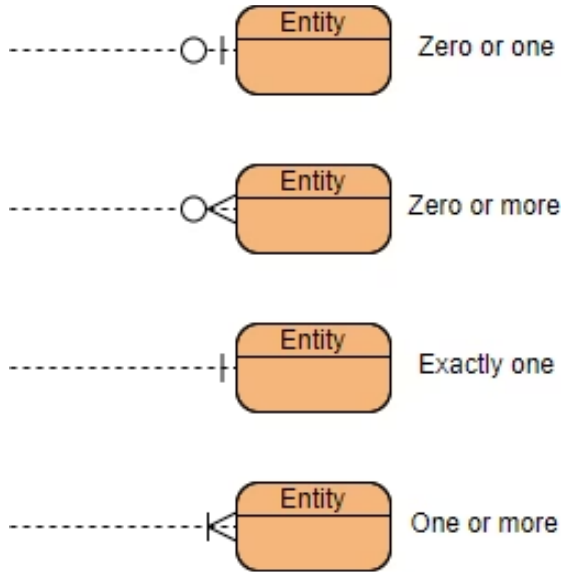
Означає "Один" (One) або "Обов'язково" (Mandatory)

Кільце

Означає "Нуль" (Zero) або "Опціонально" (Optional)

Вороняча лапка

Означає "Багато" (Many)



Типи кардинальності

Існує три базових типи відносин між даними. Розуміння цього — це 90% успіху в проєктуванні БД.

Один-до-Одного (1:1)

Один запис у таблиці А відповідає рівно одному запису в таблиці Б. Приклад: Громадянин — Паспорт. Зустрічається рідко.

Один-до-Багатьох (1:N)

"Золотий стандарт" реляційних баз даних. Приклад: Курс — Студенти. FK завжди на стороні "Багато".

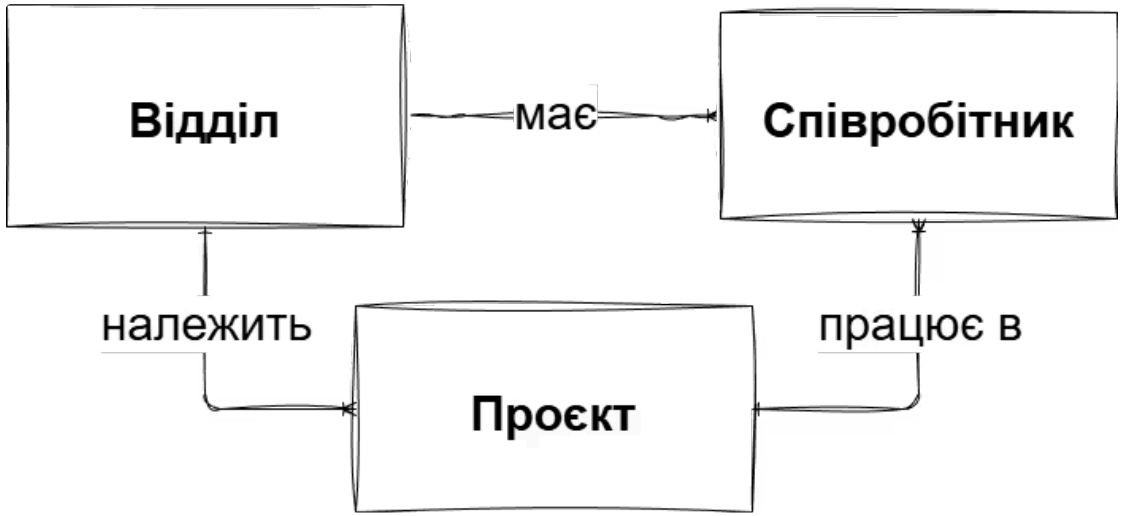
Багато-до-Багатьох (M:N)

Найскладніший тип. Приклад: Студент — Дисципліна. Вимагає проміжної таблиці (Junction Table).

Типова помилка: Циклічні зв'язки

Уявіть зв'язки:

- Відділ має Співробітників
- Співробітник працює в Проєкті
- Проєкт належить Відділу



Це створює надлишковість і ризик суперечностей. Співробітник може працювати в Проєкті відділу А, але сам числитися у відділі Б. Уникайте замкнених циклів без нагальної потреби.

1NF

Перша нормальна форма (1NF): Атомарність

> Правило: Кожна комірка таблиці повинна містити лише одне (атомарне) значення. Таблиця повинна мати Первинний ключ (PK).

Порушення 1NF

ID	Student	Phones
1	Ivan	050-111, 067-222

Поле Phones містить список — це порушення 1NF.

Результат у 1NF

ID	Student	Phone
1	Ivan	050-111
1	Ivan	067-222

Примітка: ID перестав бути унікальним — потрібен складений ключ або нова таблиця Student_Phones.

Вирішення: Розділити на рядки або винести телефони в окрему таблицю.

2NF

Друга нормальна форма (2NF): Повна функціональна залежність

> Вимога: Таблиця вже в 1NF.

> Правило: Усі неключові атрибути повинні залежати від усього первинного ключа, а не від його частини. Це стосується лише таблиць зі Складеним ключем (Composite Key).

Порушення 2NF

Таблиця Grades. Складений ключ: Student_ID + Course_ID

Student_ID	Course_ID	Grade	Professor_Name
101	Math	A	Mr. Smith
102	Math	B	Mr. Smith

- Grade залежить від обох ключів (Student_ID + Course_ID). ✓ ОК
- Professor_Name залежить ТІЛЬКИ від Course_ID — часткова залежність. *X* Порушення 2NF

Вирішення: Розбити на дві таблиці

Таблиця Grades

Student_ID	Course_ID	Grade
101	Math	A
102	Math	B

Таблиця Courses

Course_ID	Professor_Name
Math	Mr. Smith

3NF

Третя нормальна форма (3NF): Транзитивна залежність

> Вимога: Таблиця вже в 2NF.

> Правило: Жоден неключовий атрибут не повинен залежати від іншого неключового атрибута.

Порушення 3NF

Таблиця Students. PK = Student_ID

Student_ID	City	Zip_Code
1	Kyiv	01001
2	Lviv	79000
3	Kyiv	01001

- City залежить від Student_ID? Так, студент живе в місті.
- Але City залежить від Zip_Code — якщо знаємо 01001, знаємо Київ.
- Ланцюжок (транзитивність): Student → Zip → City. *Х* Порушення 3NF

Вирішення: Виносимо адресу в довідник

Таблиця Students

Student_ID	Zip_Code
1	01001
2	79000
3	01001

Таблиця Locations

Zip_Code	City
01001	Kyiv
79000	Lviv

Денормалізація: Коли правила треба порушувати?

Не потрібно дробити базу на сотні мікроскопічних таблиць. Денормалізація — це свідоме повернення дублювання заради швидкості читання.

Кейс: Інтернет-магазин

У таблиці Orders зберігаємо Product_Price, хоча ціна є в таблиці Products. Причина: ціна в Products змінюється. Якщо не зберегти «історичну ціну» в Orders на момент покупки — через місяць сума замовлення у звіті зміниться. Тут дублювання є архітектурно необхідним.

1NF: Атомарність

Немає списків у комірках

2NF: Повна залежність

Все залежить від всього ключа (вирішуємо проблеми складених ключів)

3NF: Без транзитивності

Немає залежності між звичайними полями (виносимо довідники)

Нормалізація

Пришвидшує запис (INSERT/UPDATE) і економить місце

Денормалізація

Пришвидшує читання (SELECT), але ускладнює підтримку

Висновки

Концептуальна модель Відповідає на питання "ЩО зберігаємо", фіксує бізнес-термінологію	**ERD (Crow's Foot)** Структурує дані для технічної реалізації. Правильне визначення 1:1, 1:N, M:N є критичним
Нормалізація 1NF, 2NF, 3NF пришвидшує запис і забезпечує цілісність даних	**Денормалізація** Застосовується свідомо для оптимізації швидкості читання

Зв'язок Багато-до-Багатьох (M:N) є проблемою для баз даних. Також проблемою є дублювання даних. Щоб вирішити ці проблеми, застосовується математичний процес — Нормалізація.

> Завдання на лабораторну: Створити ER-діаграму системи (Crow's Foot) та нормалізувати її до 3NF.