

# **Бази даних**

## **Лекція 11**

---

# Тематика лекції

- Функціональні залежності
- Нормальні форми

# Функціональні залежності

---

У заданому відношенні, атрибут  $Y$  функціонально залежний від атрибуту  $X$  тоді та тільки тоді, коли кожне значення  $X$  відповідає рівно одному значенню  $Y$ .

Позначення:  $X \rightarrow Y$

Також можна сказати “ $X$  визначає  $Y$ ”, або “ $Y$  залежить від  $X$ ”, або “якщо відоме  $X$  - можна знайти рівно одне відповідне значення  $Y$ ”.

# Приклад

---

Student	
PK	student_id
	name
	surname
	contact_data

# Приклад

---

Student	
PK	student_id
	name
	surname
	contact_data

**student\_id → name**  
**student\_id → surname**  
**student\_id → contact\_data**

# Приклад

---

Enrolment	
	grade
PK	course_id
PK	student_id

# Приклад

---

Enrolment	
	grade
PK	course_id
PK	student_id

**(student\_id, course\_id) → grade**

# Повні функціональні залежності

---

Повними є такі функціональні залежності, де видалення будь-якого з атрибутів з  $X$  означає, що залежність більше не існує.

Наприклад:

$(\text{student\_id}, \text{course\_id}) \rightarrow \text{grade}$



# Транзитивні залежності

---

Якщо  $A \rightarrow B$  та  $B \rightarrow C$ , то залежність  $A \rightarrow C$  є транзитивною.

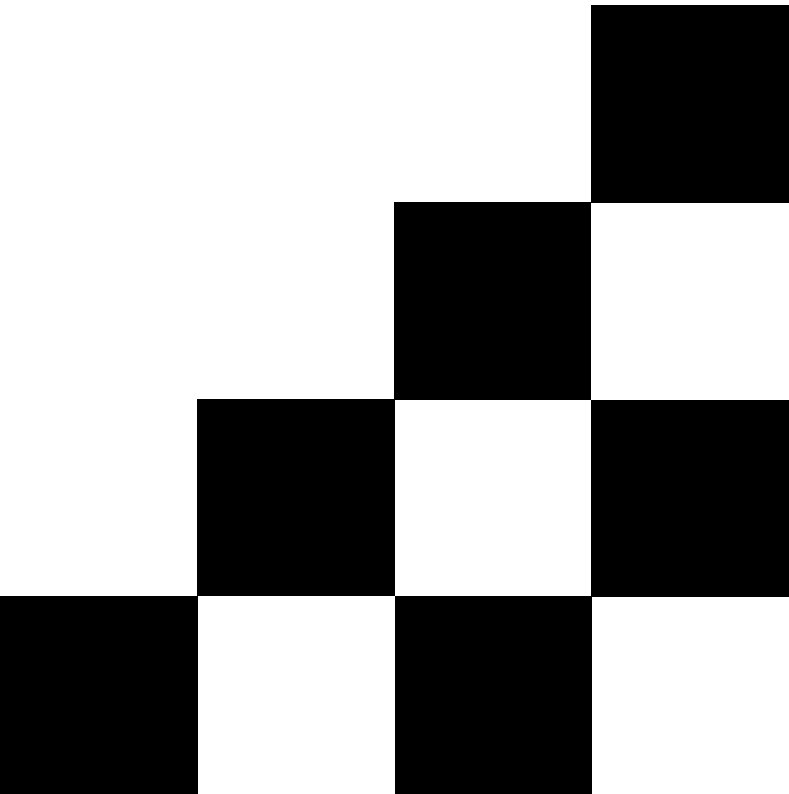
Наприклад:

$\text{student\_id} \rightarrow \text{department\_id}$

$\text{department\_id} \rightarrow \text{department\_name}$

$\text{student\_id} \rightarrow \text{department\_name}$  - транзитивна залежність

Book Order	
order_id	
customer_id	
customer_name	
book_isbn	
book_title	
author	
price	
delivery_city	



## Book Order

order\_id

customer\_id

customer\_name

book\_isbn

book\_title

author

price

delivery\_city

**order\_id → customer\_id → customer\_name**

**order\_id → delivery\_city**

**book\_isbn → book\_title, author**

**(order\_id, book\_isbn) → price**

**(order\_id, book\_isbn) - PK**

# Нормалізація

---

**Нормалізація - процес декомпозиції таблиць для уникнення надлишковості та небажаних аномалій даних.**

**Основні цілі нормалізації:**

- **простота модифікації даних**
- **консистентність даних**
- **простота підтримки бази даних**
- **мінімізація дублювань даних**

# **Основні принципи нормалізації**

---

- **Збереження одних і тих же даних лише один раз**
- **Організація даних на основі логічних залежностей**
- **Підчас нормалізації не можна втрачати дані**
- **Всі функціональні залежності повинні бути збережені**

# Нормальні форми

---

Не нормалізовані дані → 1NF → 2NF → 3NF → BCNF → 4NF → 5NF

- Всього є 6 нормальних форм
- Кожна з наступна нормальна форма базується на попередній
- В реальному житті зазвичай зупиняються на 3NF або на BCNF.

# Перша Нормальна Форма (1NF)

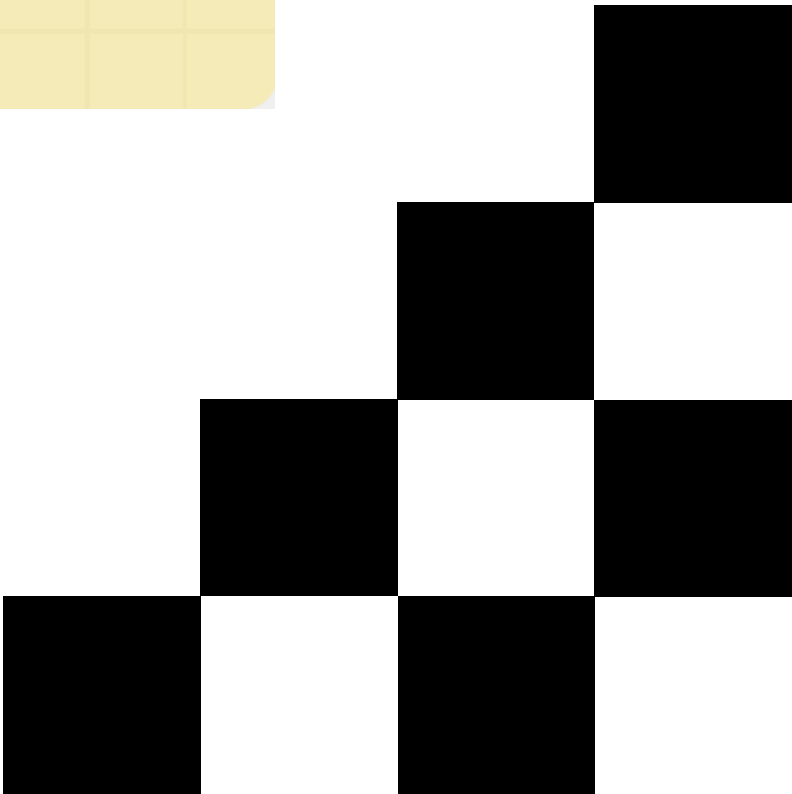
---

Таблиця перебуває у 1NF, якщо:

- Всі атрибути таблиці є неподільними (містять лише одне значення).
- Кожен рядок є унікальним (має РК).
- Порядок рядків у таблиці не важливий.
- Кожна колонка містить значення лише одного типу.

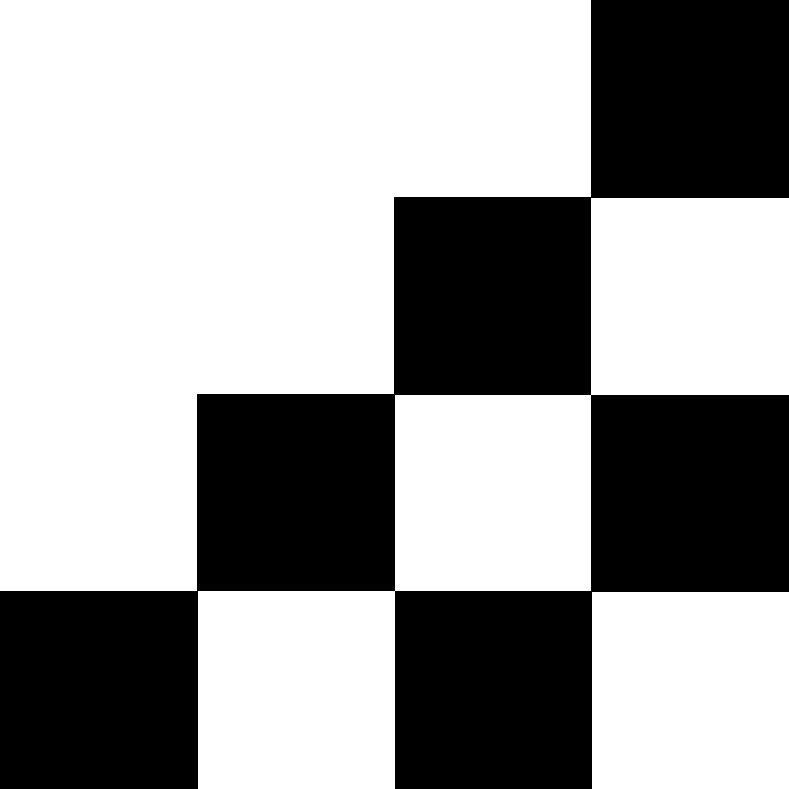
Student	
PK	student_id
	name
	phone_numbers

Student	
PK	student_id
	name
	phone_number1
	phone_number2





Student	
PK	student_id
	name
	phone_numbers TEXT[]



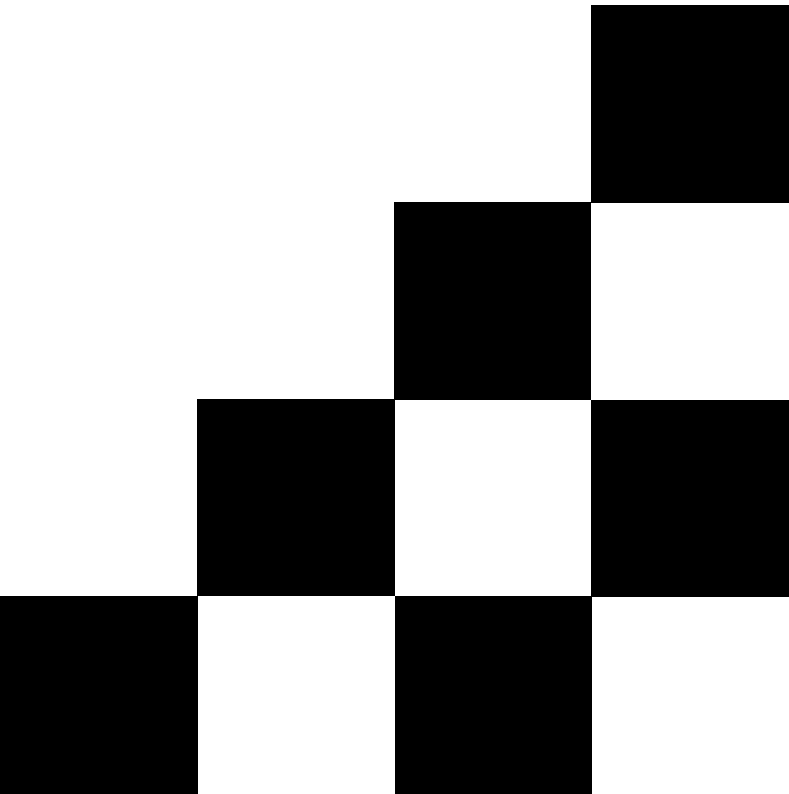
# Друга Нормальна Форма (2NF)

---

Таблиця перебуває у 2NF, якщо:

- Задовільняються всі вимоги 1NF (перебуває в першій нормальній формі).
- Усі неключові атрибути повністю функціонально залежні від первинного ключа.

Book Order	
PK	order_id
	customer_id
	customer_name
PK	book_isbn
	book_title
	author
	price
	delivery_city



# Перетворення до 2NF

---

- 1. Знайти всі часткові залежності.**
- 2. Створити окремі таблиці для кожної з частин первинного ключа, які мають часткові залежності.**
- 3. Перенести атрибути, що мають часткові залежності у відповідні створені таблиці.**
- 4. В початковій таблиці залишити лише атрибути, які перебувають у повній функціональній залежності від ключа.**

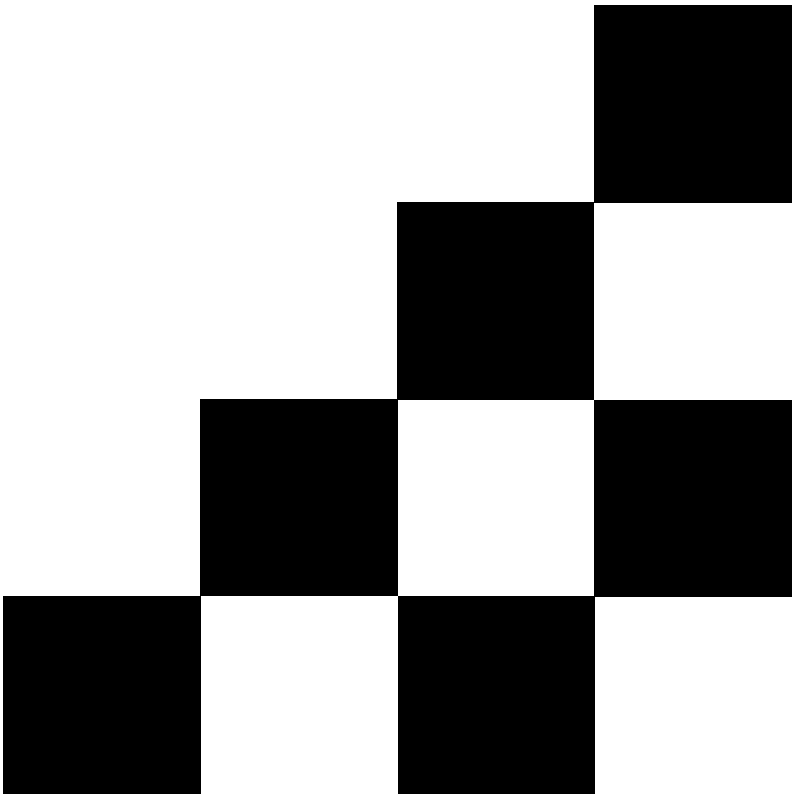
# Третя Нормальна Форма (3NF)

---

Таблиця перебуває у 3NF, якщо:

- Перебуває у 2NF.
- Відсутні транзитивні залежності - жоден з неключових атрибутів не залежить від іншого неключового атрибуту.

Student	
PK	student_id
	name
	department_id
	department_name

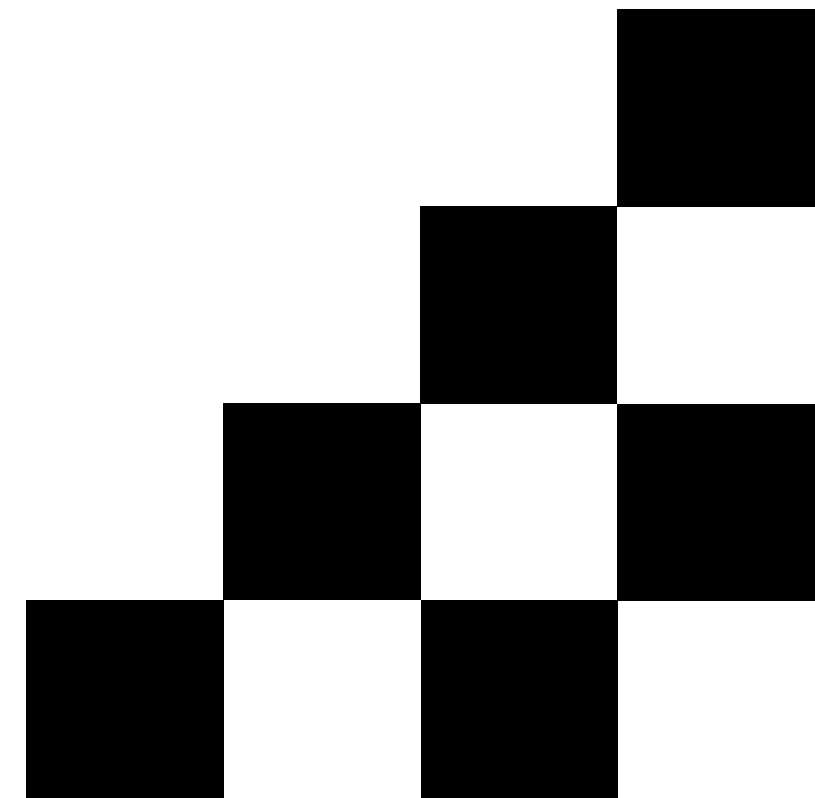


# Перетворення до 3NF

---

1. Знайти всі транзитивні залежності.
2. Створити окремі таблиці для кожної з окремих транзитивних залежностей.
3. Перенести неключові атрибути, що мають на інші неключові атрибути у відповідні створені таблиці.
4. В початковій таблиці залишити лише атрибути, які прямо залежать від повного ключа.

Order	
PK	order_id
	customer_id
	customer_name
	customer_city
	customer_zip
	customer_country
	country_tax
	total_amount





# Нормальна Форма Бойса-Кодда (BCNF)

---

Таблиця перебуває у BCNF, якщо:

- Перебуває у 3NF.
- Жоден з елементів первинного ключа не залежить від неключових атрибутів (для усіх залежностей  $X \rightarrow Y$ ,  $X$  є первинним ключем).

Ситуації, коли задовольняється 3NF, проте не BCNF є доволі рідкісними на практиці.

Certification	
PK	employee_id
PK	skill_name
	certifier_name
	certification_date

### Умови:

- у співробітника може бути багато сертифікацій
- один “сертифікатор” сертифікує лише одну навичку

# Багатозначні залежності

---

Залежність є багатозначною, якщо для кожного значення  $X$  існує набір значень  $Y$ , які є незалежними від інших атрибутів.

Позначення:  $X \twoheadrightarrow Y$

Також можна сказати “ $X$  визначає багато значень  $Y$ ”.

Дана залежність являє собою кілька one-to-many залежностей в межах одної таблиці.

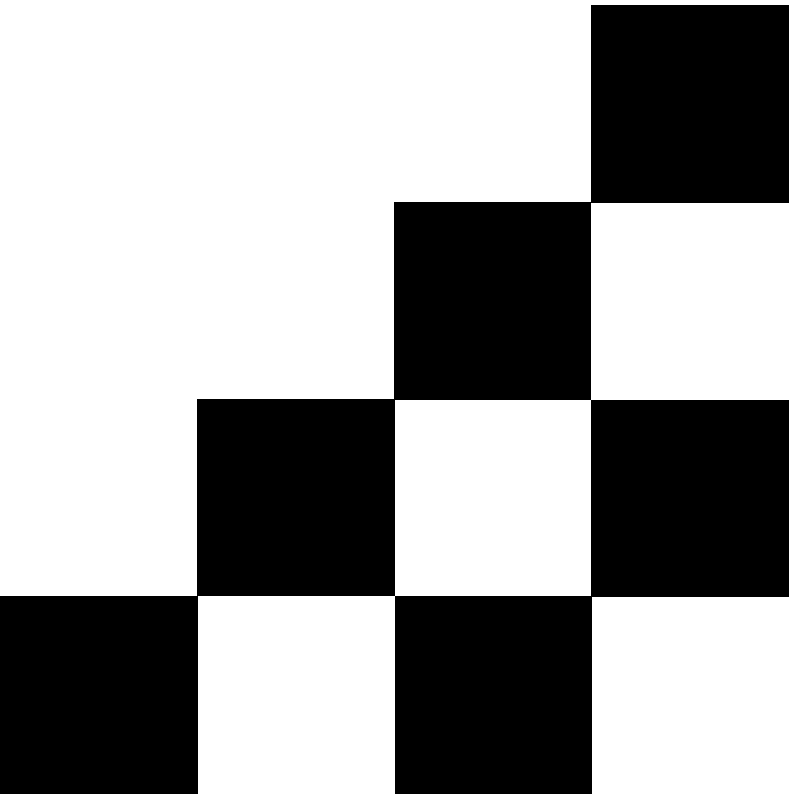
# Четверта Нормальна Форма (4NF)

---

Таблиця перебуває у 4NF, якщо:

- Перебуває у BCNF.
- Має не більше однієї багатозначної залежності.

Qualification	
PK	employee_id
PK	hard_skill
PK	spoken_language



# Join залежності

---

Таблиця має Join залежності, якщо вона може бути декомпозована на кілька менших таблиць, які в результаті join операцій можуть сформувати початкову таблицю, проте лише для певного набору комбінацій.

# П'ята Нормальна Форма (5NF)

---

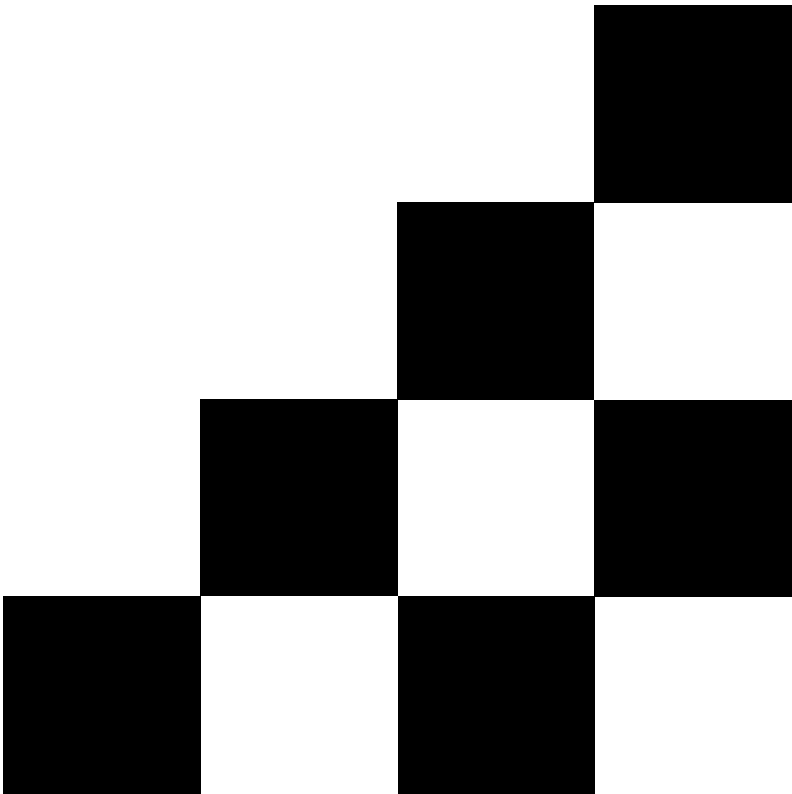
Таблиця перебуває у 5NF, якщо:

- Перебуває у 4NF.
- Не може бути розкладена на менші таблиці без втрати даних.

5NF унеможливорює ситуації, коли необхідно зберігати усі комбінації 3 і більше атрибутів, якщо частина з них неможлива.

Agent Company Product		
PK	agent_id	
PK	company	
PK	product	

Alice	Apple	iPhone
Alice	Apple	iPad
Alice	Samsung	Galaxy
Bob	Apple	iPhone
Bob	Samsung	Galaxy





---

# Переваги нормалізації

- Простіше оновлення даних
- Простіша підтримка
- Простіші запити
- Оптимізація об'єму даних

---

# Недоліки нормалізації

- **Складніші запити - більше join-ів**
- **Більше таблиць - складніша логіка**
- **Нижча швидкодія читання**

# ПРАКТИЧНІ ПОРАДИ

---

- Нормалізація це компроміс між структурованістю даних та швидкодією - чим вищий рівень нормалізації - тим більше join-ів потрібно для читання - тим нижча швидкодія.
- Варто тримати баланс між рівнем нормалізації та швидкодією - у більшості випадків достатньо 3NF. Вищі рівні нормалізації необхідні переважно у специфічних випадках. 5NF майже не використовується на практиці.

**Запитання**