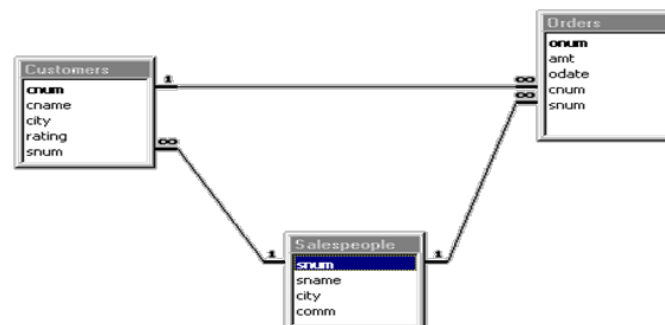


Бази даних

частина 4

Нормалізація



Навчальний курс
Валько Н.В.

SQL

Мартин Грабер

МАРТИН ГРУБЕР

Понимание SQL

Перевод Лебедева В.Н.

Под редакцией Булычева В.Н.

МОСКВА, 1993

Джеймс Р. Грофф
Пол Н. Вайнберг

SQL

Полное руководство

Второе издание,
переработанное и дополненное

Перевод с английского
под редакцией В.Р. Гинзбурга

Линн Бейли

Изучаем SQL

Приведи
в порядок
свои
отношения
с данными



Прекрати путать
термины



Освой концепцию
и синтаксис SQL
максимально
эффективно



Перестань
смушаться



Зміст

- Модель «сутність-зв'язок» (ER-модель)
- Зв'язки між таблицями
- Нормалізація
- 1НФ
- Ключові поля
- 2НФ
- Функціональна залежність
- 3НФ

Рівні абстракції при побудові моделі

Реальний об'єкт

Уявлення про
реальний
об'єкт

Модель



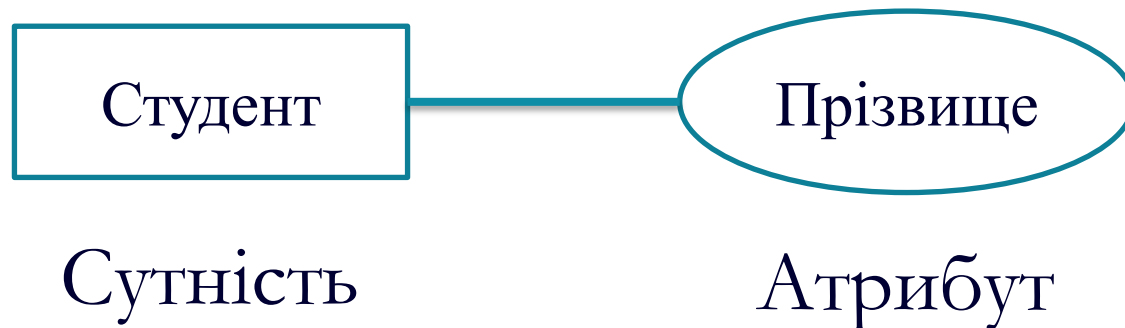
Модель «сутність-зв'язок»

Entity-relationship model (ER-модель)

- модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків
- з моделі «сутність-зв'язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об'єктна), тому вона є найзагальнішою
- Модель описує та визначає деяку предметну область (БД)

Сутність

- реальний або уявний об'єкт, що має істотне значення для аналізованої предметної області, інформація про який підлягає збереженню.
- Кожна сутність має унікальний ідентифікатор.
- Кожний екземпляр сутності однозначно ідентифікується і відрізняється від усіх інших екземплярів даного типу сутності.

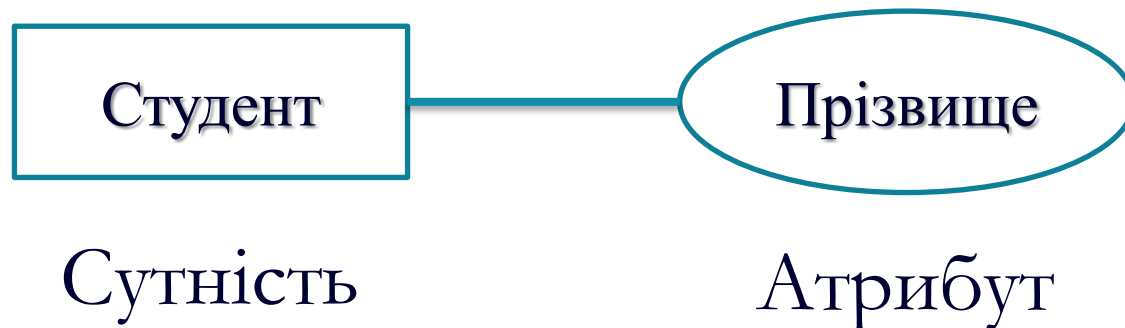


Атрибут

- будь-яка характеристика сутності, що є істотною для аналізованої предметної області та призначена для кваліфікації, ідентифікації, класифікації, кількісної характеристики або відображення стану сутності

Властивості сутності

- Кожна сутність має унікальне ім'я
- Сутність має один або декілька атрибутів, що належать їй або успадковуються через зв'язок.
- Сутність має один або декілька атрибутів, що однозначно ідентифікують кожний її екземпляр.
- Кожна сутність може мати будь-яку кількість зв'язків з іншими сутностями моделі



Види сутностей

- **Стрижнева** – незалежна від інших сутність
- **Асоціативна** (асоціація) – виражає собою зв'язок «багато до багатьох» між двома сутностями. Є цілком самостійною сутністю.



- **Характеристика** (слабка сутність) – >

Види сутностей

- **Характеристика** (слабка сутність) – пов'язана з більш сильною сутністю зв'язками «один до багатьох» і «один до одного».
 - Описує або уточнює іншу сутність.
 - Повністю залежить від неї і зникає зі зникненням останньої
 - Зарплата-працівник
 - **Позначення** це така сутність, з якої інші сутності пов'язані за принципом «багато до одного» або «один до одного». Позначення, на відміну характеристики є самостійною сутністю
 - Факультет-студент

Зв'язок

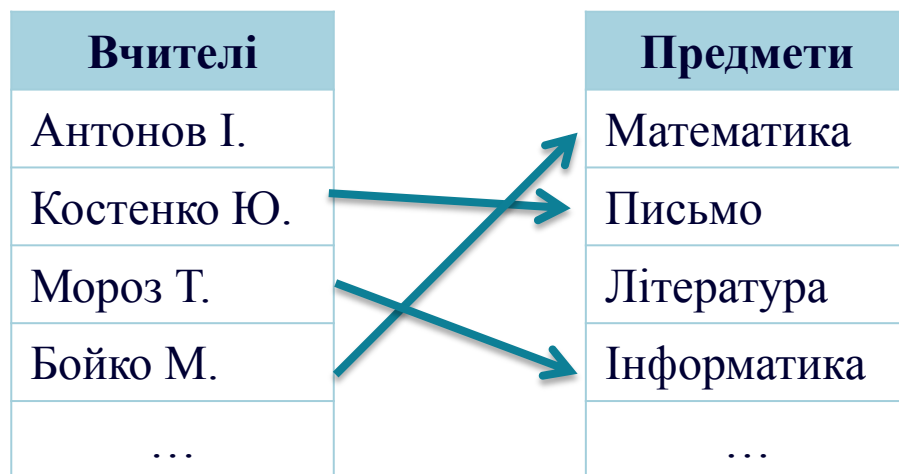
- поіменована асоціація між двома або більше сутностями, що є суттєвою для аналізованої предметної області
- асоціація між сутностями, при якій кожний екземпляр однієї сутності (сутність-предок), асоційований із довільною кількістю екземплярів іншої сутності (сутність-нащадок), а кожний екземпляр сутності-нащадка асоційований виключно з одним екземпляром сутності-предка



Зв'язки між таблицями

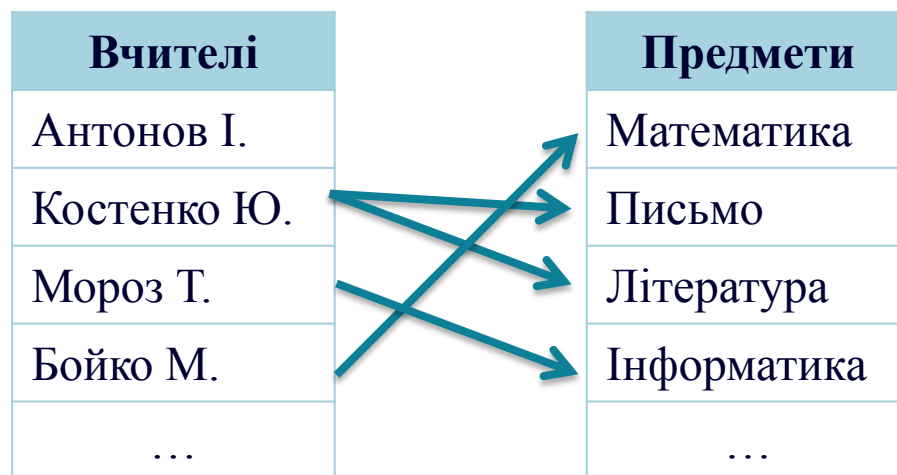
- Один до одного
- Один до багатьох, багато до одного
- Багато до багатьох

Зв'язок один-до одного



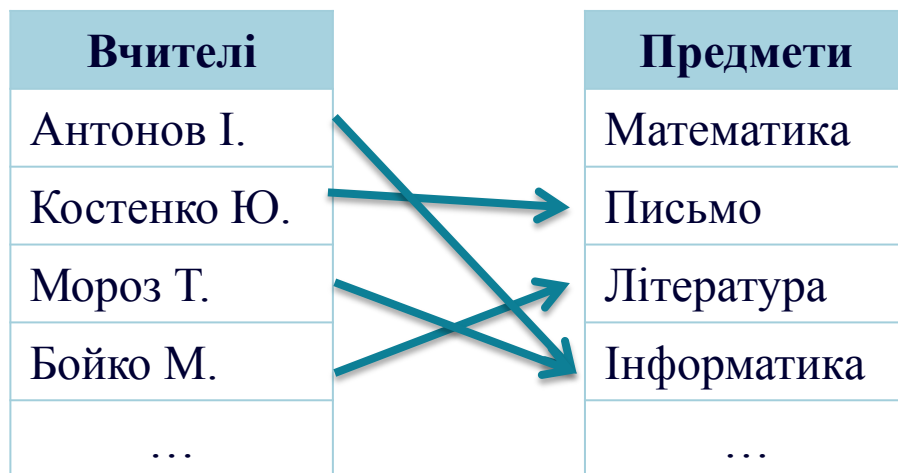
- Одному екземпляру однієї сутності відповідає один екземпляр іншої сутності

Зв'язок один-до багатьох



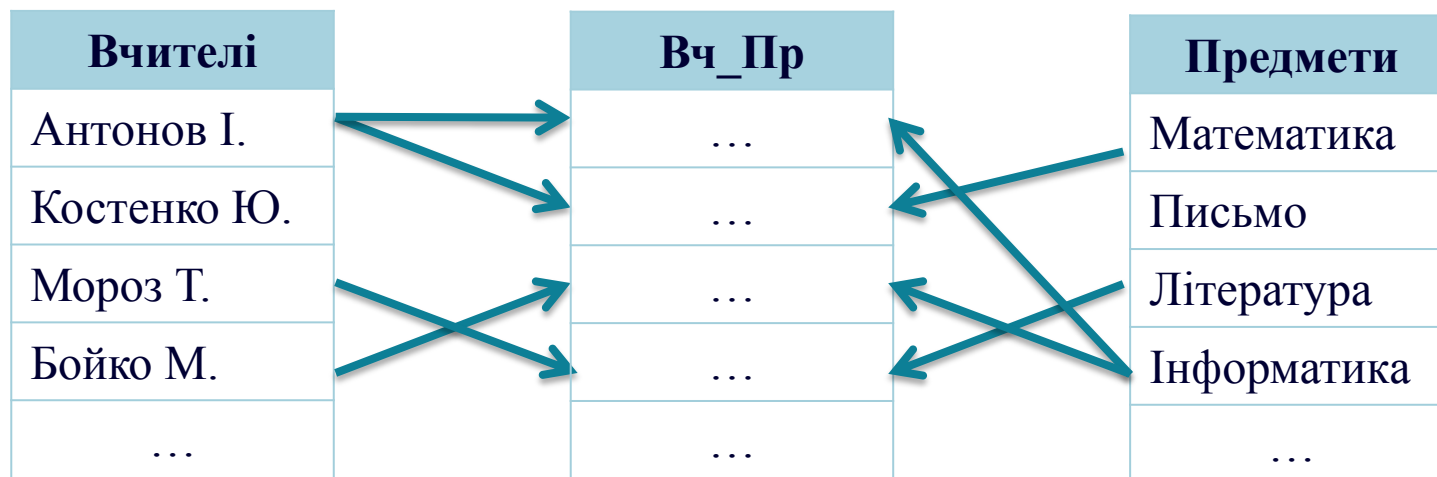
- Одному екземпляру однієї сутності може відповідати кілька екземплярів іншої сутності

Зв'язок багато-до одного



- ❑ Багатьом екземплярам однієї сутності відповідає один екземпляр іншої сутності

Зв'язок багато-до багатьох



- Багатьом екземплярам однієї сутності може відповідати кілька екземплярів іншої сутності

Нотації (Графічні діаграми)

- Сутності відображуються у вигляді прямокутників,
- Зв'язки відображуються у вигляді ромбів.
- Якщо сутність бере участь у відносинах, вони пов'язані лінією.
- Якщо відносини не є обов'язковими, то лінія пунктирна.
- Атрибути позначаються в вигляді овалів і пов'язані з однією сутністю або зв'язком.
- Овал похідних атрибутів зображується пунктирним контуром.

UML



- **UML** (англ. *Unified Modeling Language*) — уніфікована мова моделювання
- відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи

Види діаграм

Структурні діаграми:

- ☐ Класів
- ☐ Компонент
- ☐ Розгорткування
- ☐ Об'єктів
- ☐ Пакетів

Діаграми поведінки:

- ☐ Діяльності
- ☐ Станів
- ☐ Прецедентів

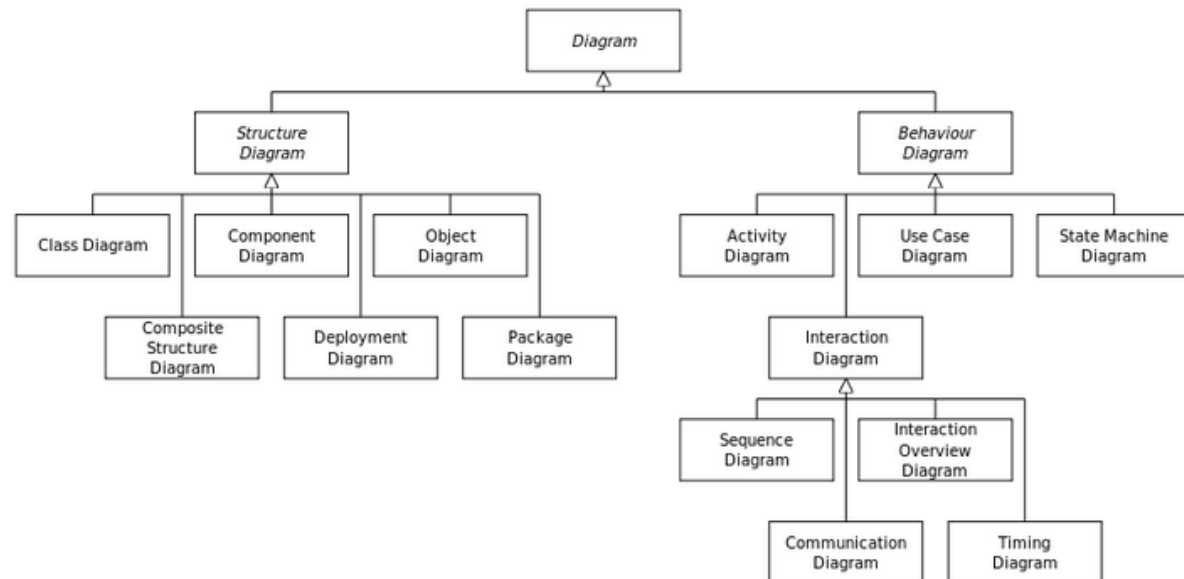
Діаграми взаємодії:

- ☐ Кооперації
- ☐ Огляду взаємодії
- ☐ Послідовності
- ☐ Синхронізації

Діаграма класів

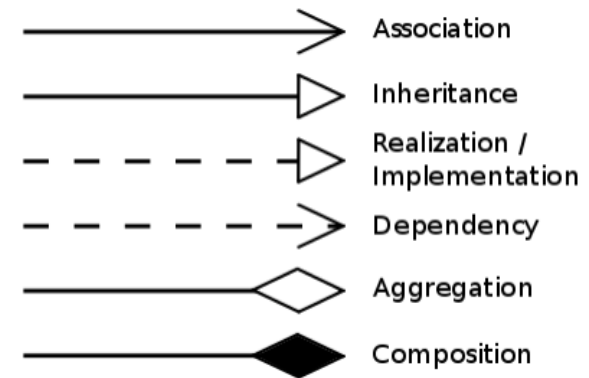
(англ. class diagram)

- ❑ Статичне представлення структури моделі
- ❑ Відображає статичні елементи, такі як: класи, типи даних, їх зміст та відношення.



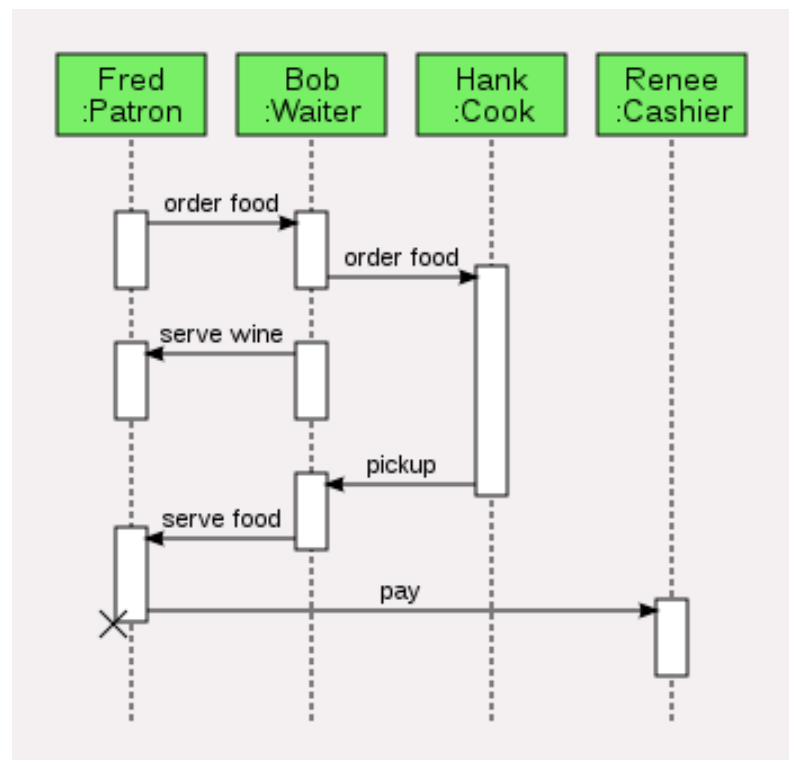
Типи зв'язків у діаграмі

- Асоціація – показує, що об'єкти однієї сутності (класу) пов'язані з об'єктами іншої сутності
- Агрегація – відображає структурне відношення між рівноправними сутностями
- Композиція – більш строгий варіант агрегації. Відома також як агрегація за значенням



Діаграма послідовності

- відображає взаємодії об'єктів впорядкованих за часом
- відображають задіяні об'єкти та послідовність відправлених повідомлень



Процес, об'єкт, повідомлення

Діаграма прецедентів (варіанти використання, use case)

- відношення між акторами та прецедентами в системі
- використовують для описання послуг, які система надає актору

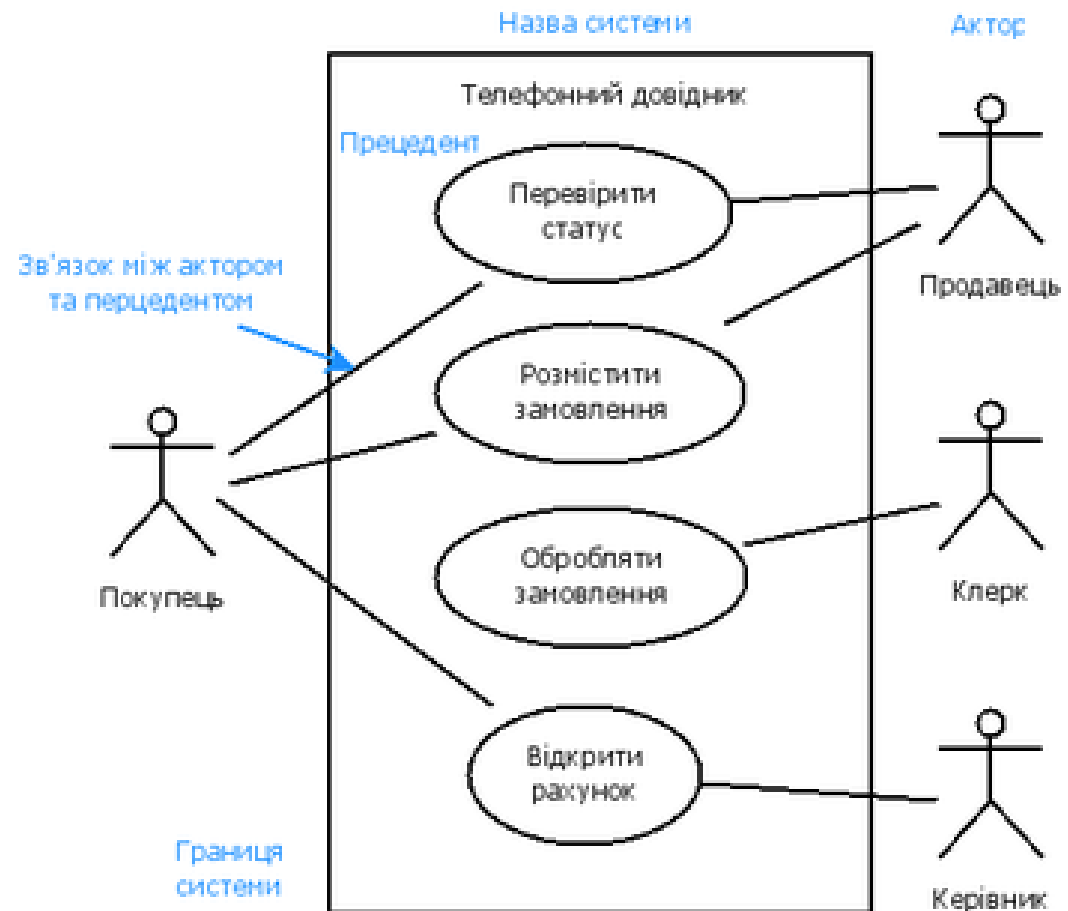


СХЕМА БД

- Опис даних (полів і таблиць) БД, включно з усіма пов'язаними об'єктами і зв'язки між ними



Нормалізація

- розбиття універсального відношення на більш дрібні
- це формальний апарат **обмежень** на формування таблиць, що **описує розбиття таблиць** на дві або більше частин і забезпечує застосування кращих методів додавання, зміни і видалення даних.

Цілі нормалізації БД:

- ❑ Можливість зберігати всі необхідні дані в БД
- ❑ Виняток надмірності даних
- ❑ Необхідність звести кількість збережених таблиць до мінімуму
- ❑ Рішення проблем, пов'язаних з оновленням і видаленням даних

НЕ нормалізована таблиця

Аномалія видалення

Покупець	Модель	Дата продажу	Ціна
Іванов І.І.	BMW	20.09.2010	90 000
Сокирко Ф.М.	Daewoo	13.10.2010	62 000
Свиров Ю.В.	Opel	5.01.2011	103 000

Проблеми не нормалізованих таблиць:
аномалія видалення, вставки, надмірність даних

НЕ нормалізована таблиця

Аномалія вставки

Покупець	Модель	Дата продажу	Ціна
Іванов І.І.	BMW	20.09.2010	96 000
Сокирко Ф.М.	Daewoo	13.10.2010	62 000
Свиров Ю.В.	Opel	5.01.2011	103 000

■Новий покупець

■RENO

Проблеми не нормалізованих таблиць:
аномалія видалення, вставки, надмірність даних

НЕ нормалізована таблиця

Надмірність даних

Покупець	Модель	Дата продажу	Ціна
Іванов І.І.	BMW	20.09.2010	96 000
Сокирко Ф.М.	Daewoo	13.10.2010	62 000
Іванов І.І.	■ Opel	24.12.2010	103 000
Свиров Ю.В.	Opel	5.01.2011	103 000

Проблеми не нормалізованих таблиць:
аномалія видалення, вставки, надмірність даних

Нормальні форми

- ❑ Перша нормальна форма 1NF
- ❑ Друга нормальна форма 2NF
- ❑ Третя нормальна форма 3NF
- ❑ Нормальна форма Бойса-Кодда BCNF
- ❑ Четверта нормальна форма 4NF
- ❑ П'ята нормальна форма 5NF

1НФ

- Таблиця знаходиться в першій нормальній формі тоді, коли вона **не містить повторюваних полів** і складових значень полів (кожне поле має містити одне значення, а не їх комбінацію)

1 НФ :

- Забезпечує зменшення бази даних
- Прискорює пошук даних, роботу запитів

1НФ

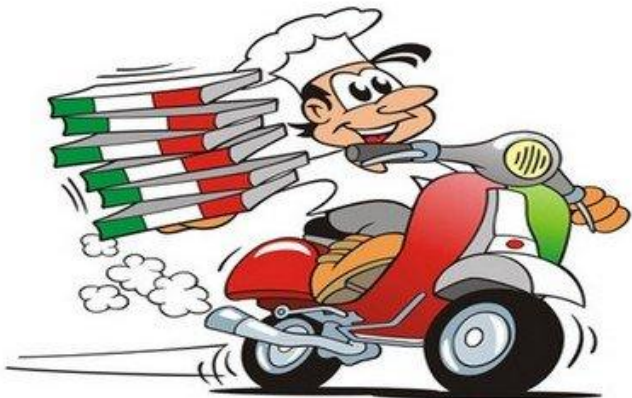
- Атомарність - маленький блок інформації, який неможливо (або небажано) розділити на складові частини меншого розміру.

Скільки потрібно полів в таблиці БД?

Вулиця, будинок

Вулиця	Будинок

!!! Структура таблиць залежить від того, як будуть використовуватись дані



Вулиця, будинок



Вулиця	Будинок

Яку таблицю побудувати?

Іграшка ↔ колір



■ М'ячі

- Зелений
- Синій
- Червоний

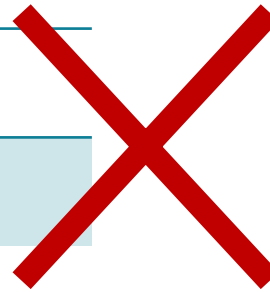


■ Йо-йо

- Червоний
- Зелений
- Жовтий

? Наприклад...

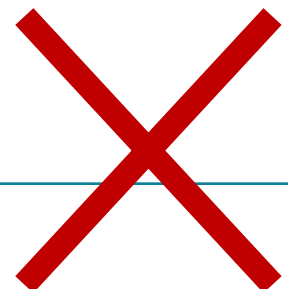
Toy_name	Colour
М'яч	Синій
М'яч	Білий
М'яч	Червоний
Йо-йо	Жовтий
Йо-йо	зелений
Йо-йо	фіолетовий



Надмірність даних

1НФ. Правило 1

- Поле, що містить атомарні значення, **не може** складатись з однотипних елементів



Toy_name	Colour
М'яч	Синій, білий, червоний, зелений
Йо-йо	Жовтий, зелений, фіолетовий

1НФ. Правило 2

- Таблиця з атомарними даними **не може** мати однотипні стовпці



Toy_name	Colour1	Colour2	Colour3
М'яч	синій	білий	червоний
Йо-йо	жовтий	зелений	фіолетовий

1НФ



- Правило 1. Поля містять тільки атомарні значення
- Правило 2. В таблиці немає повторюваних груп даних

Id_toy Toy_name		id_t Colour	
1	М'яч	1	Білий
2	Йо-йо	2	Синій
		2	Зелений
		1	Синій

1НФ



Первинний
ключ

Id_toy	Toy_name
1	М'яч
2	Йо-йо

Зовнішній
ключ

Складений
ключ

id_t	Colour
1	Білий
2	Синій
2	Зелений
1	Синій

Поняття ключа

- **Первинний ключ** це ключ за яким однозначно ідентифікуються всі записи таблиці
- **Зовнішній ключ** це ключ за яким здійснюється зв'язок з головною таблицею

1НФ

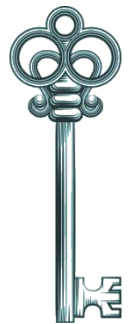
- **Складеним ключем** називається **ПЕРВИННИЙ КЛЮЧ**, що складається з декількох стовпців, комбінація яких утворює унікальні значення

Id_toy	Toy_name
1	М'яч
2	Йо-йо

id_t	Colour
1	Білий
2	Синій
2	Зелений
1	Синій

Правила первинного ключа

- ❑ Використовується для однозначного визначення запису
- ❑ НЕ може містити NULL
- ❑ Значення повинне додаватись при створенні запису (напр. автонумерація)
- ❑ Повинен бути компактним (тільки дані що забезпечують унікальність)
- ❑ Значення незмінні



Ключі. Зауваження

- Ключі служать **засобом ідентифікації об'єктів** предметної області, дані про що зберігаються у відношенні. Об'єкти предметної області повинні бути помічені.
- Ключі служать **єдиним засобом** адресації на рівні кортежів у відношенні (записів в таблиці). Точно вказати будь-який кортеж можна тільки знаючи значення його потенційного ключа

Зовнішні ключі. Зауваження

- Зовнішній ключ може бути простим і складеним
- Зовнішній ключ повинен бути визначений на тих же доменах, що і відповідний первинний ключ батьківського відношення

Первинний
ключ

Id_toy	Toy_name
1	М'яч
2	Йо-йо

Зовнішній
ключ

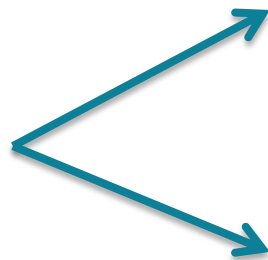
id_t	Colour
1	Білий
2	Синій
2	Зелений
1	Синій

Зовнішні ключі. Зауваження

- Зовнішній ключ, як правило, **не має** властивості **унікальності**. В дочірньому відношенні може бути кілька кортежів, що посилаються на один і той же кортеж батьківського відношення. Це дає тип відношень "**один-до-багатьох**"

UNIQUE

Id_toy	Toy_name
1	М'яч
2	Йо-йо



NOT UNIQUE

id_t	Colour
1	Білий
2	Синій
2	Зелений
1	Синій

Зовнішні ключі. Зауваження

- Якщо зовнішній ключ все-таки **має** властивість **унікальності**, то зв'язок між відносинами має тип "**один-до-одного**".
- Кожне значення зовнішнього ключа повинно збігатися зі значеннями потенційного ключа в деякому кортежі батьківського відношення, АЛЕ зворотне невірно
 - Наприклад, можуть існувати постачальники, які не постачають ніяких деталей

UNIQUE

Id_toy	Toy_name
--------	----------

1	М'яч
2	Йо-йо
3	Лялька

UNIQUE

id_t	Colour
------	--------

1	Білий
2	Синій
2	Зелений
1	Синій

Принцип цілісності

- це правила, які дають змогу уникнути введення некоректних даних у БД, а також забезпечити можливість зв'язування декількох таблиць. Ці правила можуть бути описані при створенні чи модифікації таблиці

Цілістність даних і сутностей

□ Цілістність даних

- Значення належать домену
- Перевірки на допустимість значень визначеного типу даних

□ Цілістність сутностей

- Потенційні ключі
- Первичний (головний) ключ (*primary key*)
- Зовнішні ключі (*foreign key*)

Цілісність даних

Редагування зв'язків

Таблиця/запит: Посада Пов'язана таблиця/запит: Співробітники

Код посади посади1

☒ **Забезпечення цілісності даних:**

☐ Каскадне оновлення пов'язаних полів

☐ Каскадне видалення пов'язаних полів

Тип зв'язку: Один-до-багатьох

ОК

Скасувати

Тип об'єднання...

Створити...

Зовнішній ключ таблиці
Може / НЕ може містити
значень, яких НЕ містить
ключ головної таблиці

Id_toy	Toy_name
--------	----------

1	М'яч
---	------

2	Йо-йо
---	-------

id_t	Colour
------	--------

1	Білий
---	-------

2	Синій
---	-------

3	Зелений
----------	---------

Забезпечення цілісності

- Оскільки зовнішні ключі фактично служать посиланнями на кортежі в іншому (або в тому ж самому) відношенні, то ці посилання **не повинні вказувати на неіснуючі об'єкти.**
- **Правило**
 - Зовнішні ключі не повинні бути неузгодженими, тобто для **кожного значення зовнішнього ключа** повинно існувати **відповідне значення первинного ключа** в батьківському відношенні

Id_toy	Toy_name
1	М'яч
2	Йо-йо

id_t	Colour
1	Білий
2	Синій
3	Зелений

Цілісність сутностей

- Оскільки ключі фактично служать ідентифікаторами об'єктів предметної області, то значення цих ідентифікаторів не можуть містити невідомі значення.
 - якби ідентифікатори могли містити null-значення, то ми не могли б дати відповідь "так" або "ні" на питання, збігаються чи ні два ідентифікатора.
- **Правило цілісності сутностей :**
 - Атрибути, що входять до складу деякого ключа не можуть приймати **NULL**-значень



Типи правил цілісності

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

Синтаксис установки зовнішнього ключа

[CONSTRAINT назва_обмеження]

FOREIGN KEY (ст1, ст2, ... стN)

REFERENCES головна_таблиця

(ст_головної_таблиці1, ст_головної_таблиці2,
... ст_головної_таблиціN)

[ON DELETE дія]

[ON UPDATE дія]

Створення первинного ключа

```
CREATE TABLE my_contacts  
( contact_id   Int   Not Null Auto_Increment,  
  last_name  varchar (30) default NULL,  
  first_name  varchar (20) default NULL,  
  PRIMARY KEY (contact_id)  
)
```

Додавання ключового поля у таблицю

```
ALTER TABLE my_contacts
```

```
ADD COLUMN contact_id Int Not Null Auto_Increment First,  
ADD PRIMARY KEY (contact_id)
```

```
ALTER TABLE my_contacts
```

```
ADD COLUMN contact_id Int Not Null Auto_Increment  
After first_name ,  
ADD PRIMARY KEY (contact_id)
```

FIRST - Службове слово, встановлює поле в початок таблиці

AFTER - Службове слово, встановлює поле після вказаного поля



Видалення первинного ключа

```
ALTER TABLE your_table  
DROP PRIMARY KEY;
```

Відміна автонумерації

```
ALTER TABLE your_table  
CHANGE your_id your_id INT(11) NOT NULL;
```


Друга нормальна форма

- Таблиця знаходиться в другій нормальній формі, якщо вона **задовольняє** вимогам **1НФ** і всі її поля, що не входять в первинний ключ, пов'язані повною **функціональною залежністю** з первинним ключем
- тобто **будь-яке не ключове поле однозначно ідентифікується** повним набором ключових полів

2НФ

- Включення стовпців первинних ключів в таблиці сприяє виконанню вимог 2НФ
- Друга нормальна форма **визначає зв'язок первинного ключа** таблиці з даними, що зберігаються в ній

зв'язок → функціональна залежність

Функціональна залежність

- Якщо зміна вмісту одного поля має приводити до зміни іншого, кажуть, що друге поле функціонально залежне від першого

в таблиці `super_heroes` стовпець `initials` функціонально залежить від стовпчика `name`

`super_heroes.name` → `super_heroes.initials`

↓
Складений первинний ключ



name 0+*	power 0+*	weakness	city	country	arch_enemy	initials
Супер-Мусорщик	Моментально убирает мусор	отбеливатель	Готэм	США	Неряха	СМ
Брокер	Делает деньги из ничего	NULL	Нью-Йорк	США	Налоговый Инспектор	БР
Супер-Парень	Летает	птицы	Метрополис	США	Супер-Зануда	СП
Чудо-Официант	Никогда не забывает заказы	насекомые	Париж	Франция	Обжора	ЧО
Грязнуля	Создает пыльные бури	отбеливатель	Тулза	США	Гувер	ГР
Супер-Парень	Обладает суперсилой	алюминий	Метрополис	США	Плохиш	СП

Часткова залежність

- Не ключове поле залежить від деяких, але не від усіх полів складеного первинного ключа

в разі зміни мени супергероя стовпець initials теж зміниться,
а в разі зміни power (але не name!) стовпець initials залишиться незмінним

super_heroes.power  super_heroes.initials



Складений первинний ключ

name 0+*	power 0+*	weakness	city	country	arch_enemy	initials
Супер-Мусорщик	Моментально убирает мусор	отбеливатель	Готэм	США	Неряха	СМ
Брокер	Делает деньги из ничего	NULL	Нью-Йорк	США	Налоговый Инспектор	БР
Супер-Парень	Летает	птицы	Метрополис	США	Супер-Зануда	СП
Чудо-Официант	Никогда не забывает заказы	насекомые	Париж	Франция	Обжора	ЧО
Грязнуля	Создает пыльные бури	отбеливатель	Тулза	США	Гувер	ГР
Супер-Парень	Обладает суперсилой	алюминий	Метрополис	США	Плохиш	СП

Транзитивна залежність

- Не-ключове поле пов'язане з іншим не-ключовим полем

Супергерой захотів поміняти собі закладеного ворога.

Значення `arch_enemy` при цьому зміниться, а це може привести до зміни `city`

`super_heroes.arch_enemy` → `super_heroes.city`



Складений первинний ключ

name 0+*	power 0+*	weakness	city	country	arch_enemy	initials
Супер-Мусорщик	Моментально убирает мусор	отбеливатель	Готэм	США	Неряха	СМ
Брокер	Делает деньги из ничего	NULL	Нью-Йорк	США	Налоговый Инспектор	БР
Супер-Парень	Летает	птицы	Метрополис	США	Супер-Зануда	СП
Чудо-Официант	Никогда не забывает заказы	насекомые	Париж	Франция	Обжора	ЧО
Грязнуля	Создает пыльные бури	отбеливатель	Тулза	США	Гувер	ГР
Супер-Парень	Обладает суперсилой	алюминий	Метрополис	США	Плохиш	СП

2НФ

- ❑ Правило 1. Таблиця знаходиться в 1НФ.
- ❑ Правило 2. Таблиця НЕ має часткових функціональних залежностей
- ❑ має синтетичний первинний ключ і не має складеного первинного ключа

Вся інформація о них.

Игрушки

toy_store	
toy_id	
store_id	

Вся інформація о них.

Магазины

Третя нормальна форма

- Таблиця знаходиться в третій нормальній формі, якщо вона задовольняє вимогам 2НФ і не має полів, що не входять в первинний ключ, які перебували б у транзитивній функціональній залежності від цього первинного ключа

3НФ

- Правило 1. Таблиця знаходиться в 2НФ
- Правило 2. Таблиця не має транзитивних залежностей

- При зміні `course_name` ні `instructor`, ні `instructor_phone` не змінюються.

- При зміні `instructor_phone` ні `instructor`, ні `course_name` не змінюються.

- При зміні `instructor` значення `instructor_phone` зміниться.

- Ми виявили транзитивною залежність.

courses	
→	course_id
	course_name
	instructor
	instructor_phone X

Рішення: прибрати `instructor_phone`

Нормальна форма Бойса-Кодда

- Таблиця знаходиться в нормальній формі Бойса-Кодда (НФБК), якщо і тільки якщо будь-яка функціональна залежність між його полями зводиться до повної функціональної залежності від можливого ключа

Повна декомпозиція таблиці

- така сукупність довільного числа її проєкцій, з'єднання яких повністю збігається з вмістом таблиці.

П'ята нормальна форма

- Таблиця знаходиться в п'ятій нормальній формі (5НФ) тоді і тільки тоді, коли в кожній її повній декомпозиції всі проекції містять можливий ключ. Таблиця, яка не має жодної повної декомпозиції, також знаходиться в 5НФ.

Четверта нормальна форма

- Четверта нормальна форма (4НФ) є окремим випадком 5НФ, коли повна декомпозиція повинна бути з'єднанням рівно двох проекцій.
- Вельми не просто підібрати реальну таблицю, що може бути надана в 4НФ, але не була б в 5НФ.

Створення зв'язку між таблицями


```
CREATE TABLE Colour (  
    id_t INT,  
    colour VARCHAR (10),  
    FOREIGN KEY (id_t ) REFERENCES Toy (id_toy)  
);
```

id_toy	toy_name
1	М'яч
2	Йо-йо

id_t	colour
1	Білий
2	Синій
2	Зелений
1	Синій

Видалення зв'язку

- ❑ ALTER TABLE Color
DROP FOREIGN KEY id_t;
- ❑ ALTER TABLE Color
DROP INDEX id_t;



id_toy	toy_name
1	М'яч
2	Йо-йо

id_t	colour
1	Білий
2	Синій
2	Зелений
1	Синій

Видалення таблиці зі зв'язками

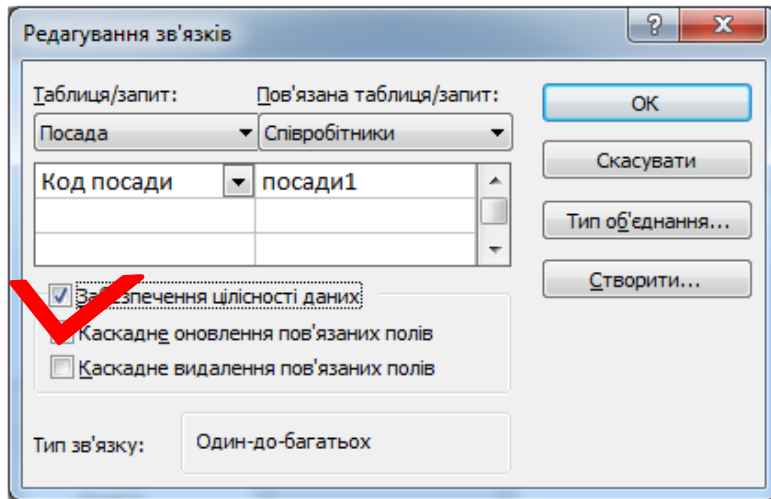
DROP TABLE name_lab
[RESTRICT | CASCADE]

Визначають долю
пов'язаних даних

- ❑ RESTRICT – забороняє операцію DROP до пов'язаних даних
- ❑ CASCADE – всі залежні об'єкти будуть видалені

Що буде видалено: таблиця, поля, дані ?

Каскадне оновлення



- Блокуються / не блокуються зміни в головній таблиці

Id_toy **Toy_name**

1	М'яч
2 5	Йо-йо

id_t **Colour**

1	Білий
2 5	Синій
2 5	Зелений
1	Синій



Каскадне видалення

Редагування зв'язків

Таблиця/запит: Посада Пов'язана таблиця/запит: Співробітники

Код посади посади1

☒ Забезпечення цілісності даних

☒ Каскадне оновлення пов'язаних полів

☒ Каскадне видалення пов'язаних полів

Тип зв'язку: Один-до-багатьох

ОК

Скасувати

Тип об'єднання...

Створити...

- Блокуються / не блокуються видалення в головній таблиці

Id_toy	Toy_name
--------	----------

1	М'яч
---	------

2	Йо-йо
--------------	------------------

id_t	Colour
------	--------

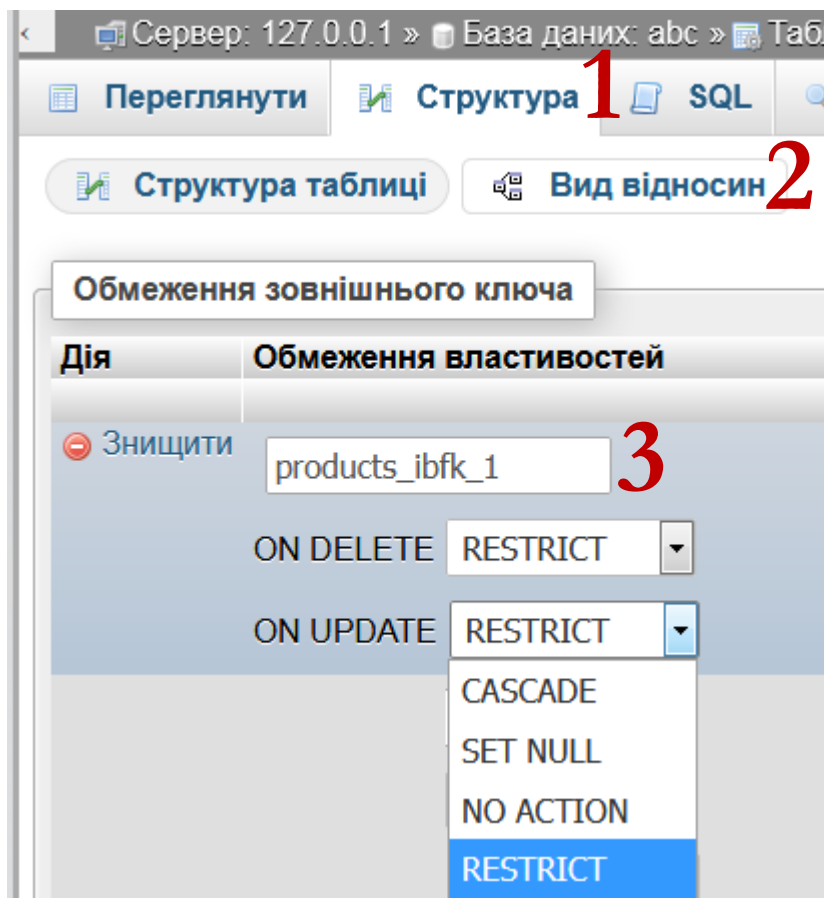
1	Білий
---	-------

2	Синій
--------------	------------------

2	Зелений
--------------	--------------------

1	Синій
---	-------

Створення зв'язку між таблицями



Питання

- ❑ Що таке атомарність?
- ❑ Коли потрібна нормалізація?
- ❑ Які властивості зовнішнього ключа?
- ❑ Чи може зовнішній ключ мати значення NULL?
- ❑ Коли НЕ потрібне каскадне видалення/оновлення?