**Лекція №3. Моделі даних: інфологічна модель даних, дата-логічна модель даних, фізична модель даних.**

Основою бази даних є ***модель даних*** — фіксована система понять і правил для представлення даних структури, стану і динаміки проблемної області в базі даних. У різний час послідовне застосування одержували ***ієрархічна, мережна і реляційна*** моделі даних. У наш час усе більшого поширення набуває ***об'єктно-орієнтований*** підхід до організації баз даних ГІС.

Термін *база даних* почав застосовуватися з 1963р. і записувався англійською як *data base*. З розвитком обчислювальної техніки ці два слова були з’єднані (*database*). Один із розробників теорії баз даних, Інгліс, у 1972р. дав таке визначення: *база даних* - сукупність збережених операційних даних, що використовуються прикладними системами деякого підприємства.

Інший класик теорії баз даних К. Дейт у своїх працях навів більш точне визначення бази даних як сукупності даних, що зберігаються у вторинній пам’яті (на дисках). *База даних* (БД) - це пойменована сукупність даних, організованих за певними правилами, що передбачають загальні принципи опису, зберігання і маніпулювання даними, не залежно від прикладних програм.

Як і у випадку будь-якого файлу даних, БД складається із записів; записи діляться на поля. *Запис* є найменшою одиницею обміну даними між оперативною й зовнішньою пам’яттю, *поле* - найменшою одиницею обробки даних.

**Види моделей даних**

**Модель «сутність – зв’язок». Основні поняття моделі «сутність – зв’язок»: сутності, зв’язки, атрибути та їх класифікація.**

***Модель «сутність-зв'язок» (ER-модель) (англ. Entity-relationship model або entity-relationship diagram)*** — модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків. ER-модель — це мета-модель даних, тобто засіб опису моделей даних. Існує ряд моделей для представлення знань, але одним з найзручніших інструментів уніфікованого представлення даних, незалежного від програмного забезпечення що його реалізує, є модель «сутність-зв'язок». Важливим є той факт, що з моделі «сутність-зв'язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об'єктна), тому вона є найзагальнішою.

Модель сутність-зв'язок є результатом систематичного процесу, який описує та визначає деяку предметну область. Вона не визначає сам процес, а лише візуалізує його. Дані представлені у вигляді компонентів (сутностей), які пов'язані між собою певними зв'язками, які виражають залежності і вимоги між ними, такі як: одна будівля може бути розділена на нуль або більше квартир, але одна квартира може бути розташована лише в одній будівлі. Сутності можуть мати різні властивості (атрибути), які характеризують їх. Діаграми, створені для представлення цих сутностей, атрибутів і зв'язків графічно, називають сутність-зв'язок діаграмами.

ER-модель зазвичай реалізується в вигляді баз даних. У разі реляційної бази даних, в якій зберігаються дані в таблицях, кожен рядок кожної таблиці являє собою один екземпляр сутності. Деякі поля даних в цих таблицях вказують на індекси в інших таблицях. Такі поля є покажчиками фізичної реалізації зв'язків між сутностями.

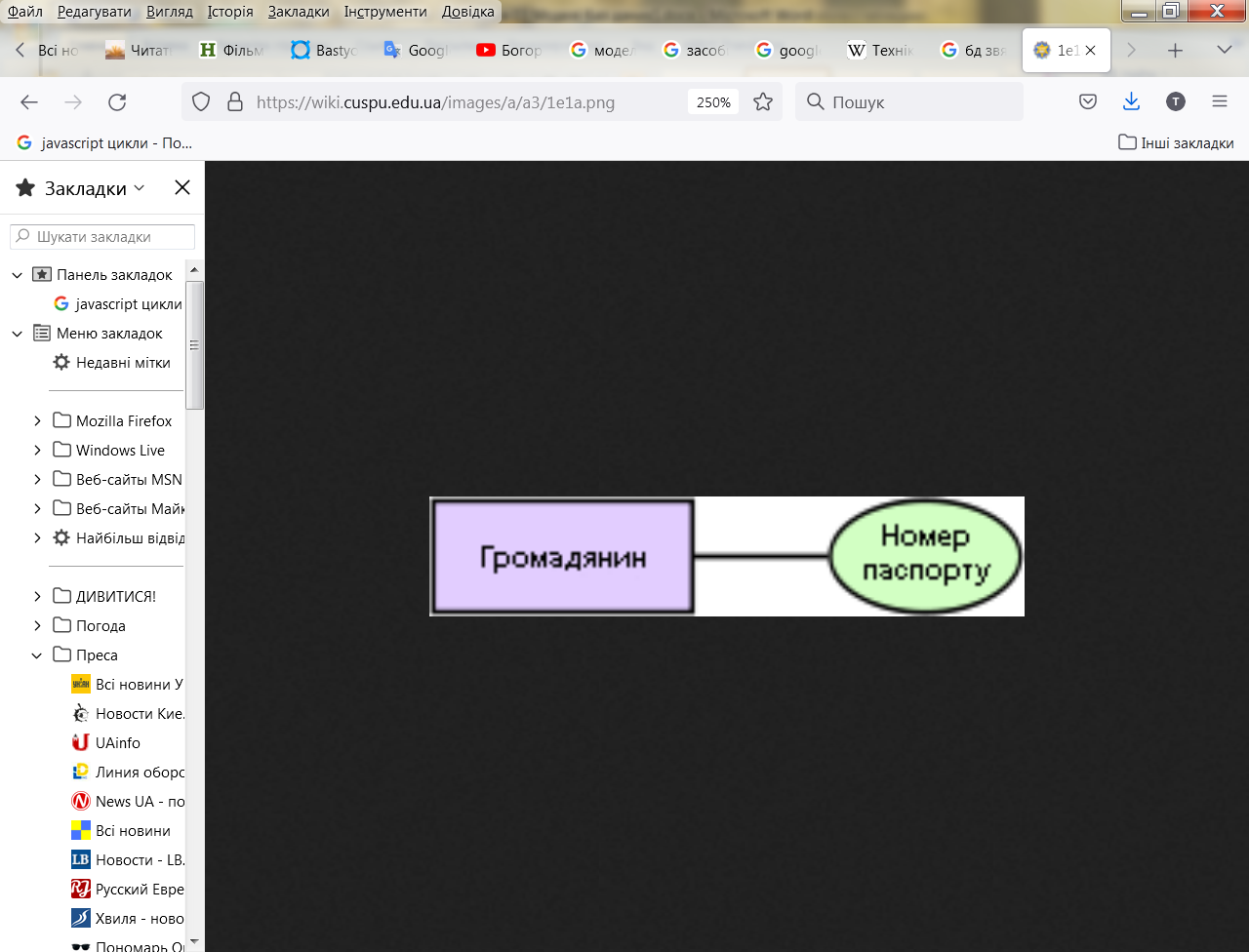
Модель сутність-зв'язок була запропонована в 1976 Пітером Пін-Шен Ченом, американським професором комп'ютерних наук в університеті штату Луїзіана. Насправді, Чен не придумував цієї моделі самотужки, він взяв ідеї з попередніх робіт наприклад, А. Брауна та інших. Але Чен зробив дуже багато для формалізації та популярності ERM, а також для її проникнення в літературу.

**Модель «сутність-зв'язок»**

Коли ми говоримо про сутність, ми зазвичай говоримо про деякий аспект реального світу, який можна виділити поміж інших аспектів. *Сутність — це збірне поняття, деяка абстракція реально існуючого об'єкта, процесу, явища чи деякого уявлення про об'єкт.* Хоча термін сутність найбільш вживаний, потрібно розрізняти поняття типу сутності та екземпляру сутності. Поняття тип сутності відноситься до набору однорідних особистостей, предметів, подій або ідей, виступаючих як ціле. Екземпляр сутності відноситься до конкретної речі в наборі. Наприклад, типом сутності може бути МІСТО, а екземпляром — Київ, Львів і т. д.

**Сутність**

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.



**Сутність та її атрибут**

*Властивості сутності* Кожна сутність має унікальне ім’я, і до того самого імені повинна завжди застосовуватися та сама інтерпретація. Одна й та сама інтерпретація не може застосовуватися до різних імен, якщо тільки вони не є псевдонімами. 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.

*Виділяють три види сутностей:* стрижнева, асоціативна (асоціація) і характеристична (характеристика):

*Стрижнева (сильна) сутність* — незалежна від інших сутність. Стрижнева сутність не може бути асоціацією, характеристикою чи позначенням.

*Асоціативна сутність (або асоціація)* виражає собою зв'язок «багато до багатьох» між двома сутностями. Є цілком самостійною сутністю. Наприклад, між сутностями ЧОЛОВІК і ЖІНКА існує асоціативний зв'язок, висловлюваний асоціативної сутністю ШЛЮБ.

*Характеристичну сутність* ще називають слабкою сутністю. Вона пов'язана з більш сильною сутністю зв'язками «один до багатьох» і «один до одного». Характеристична сутність описує або уточнює іншу сутність. Вона повністю залежить від неї і зникає зі зникненням останньої. Наприклад, сутність Зарплата є характеристикою конкретних працівників підприємства і не може в такому контексті існувати самостійно — при видаленні екземпляра сутності Працівника повинні бути видалені і екземпляри сутності Зарплата, пов'язані з видаленим працівником. 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. Наприклад, сутність Факультет позначає приналежність студента до даного підрозділу інституту, але є цілком самостійною. При моделюванні прийнято виражати (іменувати) сутність іменником або іменником з прикметником, що характеризує його, а зв'язок дієсловом, що поєднує два чи більше іменників.

**Зв’язок**

Зв’язок — це поіменована асоціація між двома або більше сутностями, що є суттєвою для аналізованої предметної області. Зв’язок — це асоціація між сутностями, при якій, як правило, кожний екземпляр однієї сутності, що називається сутністю-предком, асоційований із довільною кількістю екземплярів іншої сутності, що називається сутністю-нащадком, а кожний екземпляр сутності-нащадка асоційований виключно з одним екземпляром сутності-предка.

***Зв'язок та його атрибут***

*Класифікація зв’язків:*

— за множинністю;

— за повнотою.

*Типи зв’язків між сутностями за множинністю:* Один — до одного 1 : 1 Одному екземпляру однієї сутності відповідає один екземпляр іншої сутності Кожний учитель (екземпляр сутності «Викладачі») працює в окремому кабінеті (екземпляр сутності «Кабінети»)

*Один — до багатьох* 1 : ∞ або 1 : М Одному екземпляру однієї сутності може відповідати кілька екземплярів іншої сутності Кожний учитель (екземпляр сутності «Викладачі») викладає в декількох класах (екземпляри сутності «Класи»)

*Багато — до одного* ∞ : 1 або М : 1 Багатьом екземплярам однієї сутності відповідає один екземпляр іншої сутності Декілька викладачів (екземпляри сутності «Викладачі») викладають один предмет (екземпляри сутності «Предмети»)

*Багато — до багатьох* ∞ : ∞ або М : М Багатьом екземплярам однієї сутності може відповідати кілька екземплярів іншої сутності Декілька викладачів (екземпляри сутності «Викладачі») викладають у декількох класах (екземпляри сутності «Класи»)

*Типи зв’язків між сутностями за повнотою:* Приклад Кожен екземпляр однієї сутності обов’язково пов’язаний з одним чи кількома екземплярами іншої сутності Зв’язок між сутностями «Викладачі» та «Предмети» передбачає, що викладач повинен викладати хоча б один предмет Кожний екземпляр однієї сутності не обов’язково пов’язаний з одним чи кількома екземплярами іншої сутності Зв’язок між сутностями «Викладачі» та «Класи» передбачає, що є клас, в якому не викладає жоден з викладачів Розглянуту модель предметної області називають моделлю «сутність-зв’язок», або ER-моделлю. Назва «реляційна» (relational) пов’язана з тим, що кожен запис у таблиці даних містить інформацію, яка стосується якогось конкретного об’єкта. Крім того, пов’язані між собою (тобто перебувають у певних відношеннях — relations) дані навіть різних типів у моделі можуть розглядатися як одне ціле.

Сутності та зв'язки можуть мати свої атрибути. Наприклад, сутність громадянин має атрибут номер паспорту, а зв'язок має між сутностями гравець та аккаунт володіє атрибутом останній вхід.

**Атрибут**

Атрибут — будь-яка характеристика сутності, що є істотною для аналізованої предметної області та призначена для кваліфікації, ідентифікації, класифікації, кількісної характеристики або відображення стану сутності. Атрибут подає тип характеристик або властивостей, асоційованих із множиною реальних чи абстрактних об’єктів (людей, місць, подій, станів, ідей, пар предметів тощо). Екземпляр сутності повинен мати єдине визначене значення для асоційованого атрибута.

Кожна сутність (якщо це не слабка сутність) має мати мінімальний набір унікальних атрибутів, що зветься первинним ключем.

***Нотації (Графічні діаграми)***

***Нотація Пітера Чена***

Сутності відображуються у вигляді прямокутнків, зв'язки у вигляді ромбів. Якщо сутність бере участь у відносинах, вони пов'язані линією. 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.

***Crow's Foot***

Нотація «вороняча лапка» Дана нотація була запропонована Гордоном Еверестом (англ. Gordon Everest) під назвою Inverted Arrow («обернена стрілка»), однак частіше за все цю нотацію називають Crow's Foot («вороняча лапка»), або ж Fork («виделка»).

Згідно даної нотації, *сутність* подається у вигляді прямокутника, де сутність виражається іменником. Ім'я сутності має бути унікальним в рамках однієї моделі. Зв'язок *зображується лінією, яка пов'язує дві сутності, що беруть участь у відношенні. Ступінь кінця зв'язку вказується графічно, множинність зв'язку зображується у вигляді «виделки» на кінці зв'язку. Модальність зв'язку так само зображується графічно — необов'язковість зв'язку позначається кружком на кінці зв'язку.*

*Атрибути* сутності записуються усередині прямокутника, що зображує сутність і виражаються іменником в однині (можливо, з уточнюючими словами). Серед атрибутів виділяється ключ сутності — ненадлишковий набір атрибутів, значення яких в сукупності є унікальними для кожного екземпляра сутності.

**Ієрархічна модель**

***Ієрархічна структура*** (модель) будується у вигляді ієрархічної деревоподібної структури, у якій для кожного головного об’єкта існує кілька підлеглих, а для кожного підлеглого об’єкта може бути тільки один головний. На найвищому рівні ієрархії перебуває кореневий об’єкт. Прикладом ієрархічної структури даних може бути організація каталогів на диску, різного роду класифікації, структура державної влади тощо.

Концептуальна схема ієрархічної моделі являє собою сукупність типів записів, пов'язаних типами зв'язків в одним чи кількома деревами. Усі типи зв'язків цієї моделі належать до виду ***«один до декількох»*** і зображуються у вигляді стрілок.

Таким чином, взаємозв'язки між об'єктами нагадують взаємозв'язки в генеалогічному дереві, за єдиним винятком: для кожного породженого (підлеглого) типу об'єкта може бути тільки один вхідний (головний) тип об'єкта. Тобто ієрархічна модель даних допускає тільки два типи зв'язків між об'єктами: «один до одного» і «один до декількох». Ієрархічні бази даних є навігаційними, тобто доступ можливий тільки за допомогою заздалегідь визначених зв'язків.

При моделюванні подій, як правило, необхідні зв'язки типу ***«багато до декількох».*** Як одне з можливих рішень зняття цього обмеження можна запропонувати дублювання об'єктів. Однак дублювання об'єктів створює можливості неузгодженості даних.

Достоїнство ієрархічної бази даних полягає в тому, що її навігаційна природа забезпечує швидкий доступ при проходженні вздовж заздалегідь визначених зв'язків. Однак негнучкість моделі даних і, зокрема , неможливість наявності в об'єкта декількох батьків, а також відсутність прямого доступу до даних роблять її непридатною в умовах частого виконання запитів, не запланованих заздалегідь. Ще одним недоліком ієрархічної моделі даних є те, що інформаційний пошук з нижніх рівнів ієрархії не можна спрямувати по вище розміщених вузлах.

**Мережна модель**

*У* ***мережній моделі*** один і той же об’єкт може одночасно виступати як у ролі головного, так і підлеглого елемента. Це означає, що кожний об’єкт може брати участь у довільній кількості зв’язків. Зв’язок у цьому випадку може встановлюватися явно, коли значення деяких полів є посилання на дані, що містяться в іншому файлі. Прикладом мережної структури БД може бути структура автобусних маршрутів (із будь-якого населеного пункту існують маршрути в інші).

Подібно до ієрархічної, мережну модель також можна подати у вигляді орієнтованого графа. Але в цьому випадку граф може містити цикли, тобто вершина може мати кілька батьківських вершин.

Така структура набагато гнучкіша і виразніша від попередньої і придатна для моделювання більш ширшого класу завдань.

Ієрархічні і мережні бази даних часто називають базами даних з навігацією. Ця назва відбиває технологію доступу до даних, використовувану при написанні програм обробки мовою маніпулювання даними. При цьому доступ до даних по шляхах, не передбачених при створенні бази даних, може потребувати нерозумно тривалого часу.

Підвищуючи ефективність доступу до даних і скорочуючи таким чином час відповіді на запит, принцип навігації разом з цим підвищує і ступінь залежності програм і даних. Програми обробки даних виявляються жорстко прив'язаними до поточного стану структури бази даних і повинні бути переписані при її змінах. Операції модифікації і видалення даних вимагають переустановлення покажчиків, а маніпулювання даними залишається записоорієнтованим. Крім того, принцип навігації не дозволяє істотно підвищувати рівень мови маніпулювання даними, щоб зробити його доступним користувачу-непрограмісту чи навіть програмісту-непрофесіоналу. Для пошуку запису-мети в ієрархічній або мережній структурі програміст повинен спочатку визначити шлях доступу, а потім — крок за кроком переглянути всі записи, що трапляються на цьому шляху.

Наскільки складними є схеми представлення ієрархічних і мережних баз даних, настільки і трудомістким є проектування конкретних прикладних систем на їхній основі. Як показує досвід, тривалі терміни розроблення прикладних систем нерідко призводять до того, що вони постійно перебувають на стадії розроблення і доопрацювання. Складність практичної реалізації баз даних на основі ієрархічної і мережної моделей визначила створення реляційної моделі даних.

**Реляційна модель**

*У* ***реляційній моделі*** дані й взаємозв’язки між ними подаються за допомогою прямокутних таблиць. Рядки в реляційній базі даних називають *записами*, а стовпці — *полями*. Модель реляційної бази даних була вперше розроблена доктором Е.Ф. Коддом на початку 70-х років XX ст. як більш зручний засіб збереження, вибірки й маніпулювання даними, ніж ієрархічні й мережні бази даних. Модель двовимірної таблиці дозволяє звертатися до даних як по рядках, так і по стовпцях, що є значною перевагою.

Назва «реляційна» (relational) пов'язана з тим, що кожен запис у таблиці даних містить інформацію, яка стосується (related) якогось конкретного об'єкта. Крім того, зв'язані між собою (тобто такі, що знаходяться в певних відношеннях — relations) дані навіть різних типів в моделі можуть розглядатися як одне ціле.

Таблиця має такі властивості:

* кожний елемент таблиці являє собою один елемент даних;
* повторювані групи відсутні;
* усі стовпці в таблиці однорідні; це означає, що елементи стовпця мають однакову природу;
* стовпцям присвоєні унікальні імена;
* у таблиці немає двох однакових рядків

Порядок розміщення рядків і стовпців у таблиці довільний; таблиця такого типу називається відношенням. У сучасній практиці для рядка використовується термін ***«запис»,*** а для стовпця термін ***«поле».***

*Основною відмінністю пошуку даних в ієрархічних, мережних і реляційних базах даних є те, що ієрархічні і мережні моделі даних здійснюють зв'язок і пошук між різними об'єктами за структурою, а реляційні — за значенням ключових атрибутів* (наприклад, можна знайти всі записи, значення яких у полі «номер будинку» дорівнює 3, але не можна знайти 3-й рядок).

Оскільки реляційна структура концептуально проста, вона дозволяє реалізовувати невеликі і прості (і тому легкі для створення) бази даних, навіть персональні, сама можливість реалізації яких ніколи навіть і не розглядалася в системах з ієрархічною чи мережною моделлю.

Недоліком реляційної моделі даних є надмірність по полях (для створення зв'язків між різними об'єктами бази даних).

Практично всі існуючі на сьогоднішній день комерційні бази даних і програмні продукти для їх створення використовують реляційну модель даних.

**Інфологічна модель даних “Сутність-зв’язок”**

Базу даних можна представити на трьох рівнях: інфологічному, даталогічному та фізичному.

Мета інфологічного моделювання – забезпечення найбільш природних для людини способів збору та представлення тієї інформації, яку передбачається зберігати в базі даних. Тому інфологічну модель даних намагаються будувати по аналогії з природною мовою.

На етапі даталогічного проектування будується логічна структура БД. При цьому **даталогічна модель** розробляється з урахуванням конкретної реалізації СУБД основі її інфологічної моделі. Кінцевим результатом даталогічного проектування є опис структури БД на мові опису даних конкретних СУБД.

Не всі види зв’язку, що існують в інфологійній моделі наочній області, можна відобразити у даталогічній моделі. Так більшість СУБД не забезпечують підтримку зв’язку типу М:М. В цьому випадку вводиться допоміжний елемент, тобто M:N розбивається на два відношення (1:M, 1:N).

**Фізична модель** – прив’язка даталогічної моделі БД до середовища зберігання. Використовуються можливості даної конкретної СУБД. Приховано від розробника.

Основними конструктивними елементами інфологічної моделі є сутність, зв’язки між ними та їх властивості (атрибути), які ми розглядали вище.

**Сутність**– будь-який помітний об’єкт (об’єкт, який можна відрізнити від іншого), інформацію про який необхідно зберігати в базі даних.

**Атрибут**– пойменована характеристика сутності. Його найменування повинне бути унікальним для конкретного типа сутностів, але може бути однаковим для різного типа

**Ключ**– мінімальний набір атрибутів, по значеннях яких можна однозначно знайти необхідний екземпляр сутності. Мінімальність означає, що виключення з набору будь-якого атрибуту не дозволяє ідентифікувати сутність по тих, що залишилися.

**Зв’язок**– асоціювання двох або більш сутностей. Якби призначенням бази даних було тільки зберігання окремих, не зв’язаних між собою даних, то її структура могла б бути дуже простою. Проте одна з основних вимог до організації реляційної бази даних – це забезпечення можливості відшукання однієї сутності за значеннями інших, для чого необхідно встановити між ними певні зв’язки. А оскільки в реальних базах даних нерідко містяться сотні сутностей, то теоретично між ними може бути встановлено більше тисячі зв’язків. Наявність такої безлічі зв’язків і визначає складність інфологічної моделей.

**Характеристика зв’язків, мова моделювання**

При побудові інфологічних моделей можна використовувати мову *ER-діаграм* (від англ. Entity-Relationship, тобто сутність-зв’язок). У них є певні позначення:

- сутність

- ассоциация

- атрибут

Зв’язки позначаються лініями, над якими може проставлятися ступінь зв’язку (1 або буква, яка замінює слово “багато”) і необхідне пояснення.

Між двома сутностями, наприклад, А і В, можливі чотири види зв’язків.

1. Зв’язок ОДИН-ДО-ОДНОГО (1:1): у кожен момент часу кожному екземпляру сутності А відповідає 1 або 0 екземпляр сутності В.

PIC2-2-1

Студент може не одержувати стипендію, одержати звичайну або одну з підвищених стипендій.

2. Зв’язок ОДИН-ДО-БАГАТЬОХ (1:М): одному представнику сутності А відповідають 0, 1 або декілька представників сутності В.

PIC2-2-2

Квартира може бути порожньою, в ній може жити один або декілька мешканців.

Оскільки між двома сутностями можливі зв’язки в обох напрямах, то існує ще два типу зв’язку: БАГАТО-ДО-ОДНОГО (М:1) і БАГАТО-ДО-БАГАТЬОХ (М:N).

**Приклад.** Якщо зв’язок між сутностями ЧОЛОВІК і ЖІНКА називається ШЛЮБ, то існує чотири можливі представлення такого зв'язку:



Характер зв’язків між сутностями не обмежується перерахованими. Існують і складніші зв’язки, наприклад “безліч зв’язків між однією і тією ж суттю” (пацієнт, маючи одного лікуючого лікаря, може мати також декілька лікарів-консультантів; лікар може бути лікуючим лікарем декількох пацієнтів і може одночасно консультувати дещо інших пацієнтів);

PIC2-2-4

У приведених прикладах для підвищення ілюстративності даних зв’язків не показані атрибути сутності та асоціацій у всіх ER-діаграмах. Так, введення лише декількох основних атрибутів в опис шлюбних зв’язків значно ускладнить ER-діаграму. У зв’язку з цим мова ER-діаграм використовується для побудові невеликих моделей та ілюстрації окремих фрагментів великих.

Для виявлення зв’язків між сутностями необхідно, як мінімум, визначити саму сутність. Але це не просте завдання, оскільки в різних наочних областях один і той же об'єкт може бути сутністю або атрибутом.

***Класифікація сутності***

*Стрижньова* сутність (стрижень) – це незалежна сутність.

*Асоціативна сутність* (*асоціація*) – це зв’язок виду “БАГАТО-ДО-БАГАТЬОХ” між двома або більше сутностями або екземплярами сутності. Асоціації розглядаються як повноправна сутність:

* вони можуть брати участь в інших асоціаціях і позначеннях точно так, як і стрижньова сутність;
* можуть володіти властивостями, тобто мати не тільки набір ключових атрибутів, необхідних для вказівки зв’язків, але і будь-яке число інших атрибутів, що характеризують зв’язок.

*Характеристична сутність* (*характеристика*) – це зв’язок виду “БАГАТО-ДО-ОДНОГО” або “ОДИН-ДО-ОДНОГО” між двома сутностями. Єдина мета характеристики в рамках даної наочної області полягає в описі або уточненні деякої іншої сутності. Необхідність в них виникає у зв’язку з тим, що сутність реального світу має іноді багатозначні властивості. Чоловік може мати декілька дружин, книга – декілька характеристик перевидання (виправлене, доповнене, перероблене, ...) тощо.

Для опису характеристики використовується нова пропозиція МІМ, що має в загальному випадку вигляд:

ХАРАКТЕРИСТИКА (атрибут 1, атрибут 2, ...)

{СУТНІСТЬ, ЩО ХАРАКТЕРИЗУЄТЬСЯ}.

Розширимо мову ER-діаграм, ввівши для зображення характеристики трапецію (мал. 2.).



*Рис. 2. Елементи розширеної мови ER-діаграм*

*Позначення* – це зв’язок виду “БАГАТО-ДО-ОДНОГО” або “ОДИН-ДО-ОДНОГО” між двома сутностями, відрізняється від характеристики тим, що не залежить від сутності, що позначається.

Розглянемо приклад побудови інфологічної моделі бази даних “Живлення”, де повинна зберігатися інформація про блюда, їх щоденне споживання, продукти, з яких готуються ці блюда, постачальників цих продуктів. Інформація використовуватиметься кухарем і керівником невеликого підприємства громадського харчування, а також його відвідувачами.

|  |
| --- |
| Лобіо по грузински  Ламану очищену квасолю, нашаткований лук посолити, посипати перцем і припустити в маслі з невеликою кількістю бульйону; додати кинзу, зелень петрушки, рейган (базилік) і довести до готовності. Потім запекти в духовці.  Квасоля стручкова (свіжа або консервована) 200  Лук зелений 40, Масло вершкове 30, Зелень 10.  Вихід 210. Калорій 725. |

За допомогою вказаних користувачів виділені наступні об’єкти та характеристики майбутньої бази даних:

1. Блюда, для опису яких потрібні дані, що входять в їх кулінарні рецепти: номер блюда (наприклад, з книги кулінарних рецептів), назва блюда, вид блюда (закуска, суп, гаряче), рецепт (технологія приготування блюда), вихід (вага порції), назва, калорійність і вага кожного продукту, що входить в блюдо.
2. Для кожного постачальника продуктів: найменування, адреса, назва продукту, що поставляється, дата постачання і ціна на момент постачання.
3. Щоденне споживання блюд (витрата): блюдо, кількість порцій, дата.

Аналіз об’єктів дозволяє виділити:

* + стрижні Блюда, Продукти і Міста;
  + асоціації Склад (пов'язує Блюда з Продуктами) і Постачання (пов’язує Постачальників з Продуктами);
  + позначення Постачальники;
  + характеристики Рецепти і Витрата.

ER-діаграма моделі показана на мал. 3.

У цих моделях Блюдо, Продукт і Постачальник – найменування, а БЛ, ПР і ПОС – цифрові коди блюд, продуктів і організацій, що поставляють ці продукти.

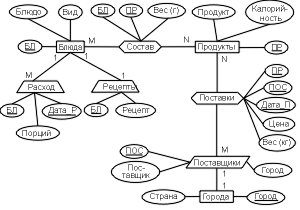


Рис. 3. Інфологічна модель бази даних "Живлення"

**Первинні та зовнішні ключі**

Нагадаємо, що *ключ* або *можливий ключ* – це мінімальний набір атрибутів, по значеннях яких можна однозначно знайти необхідний екземпляр сутності. Мінімальність означає, що виключення з набору будь-якого атрибуту не дозволяє ідентифікувати сутність по тих, що залишилися.

Кожна сутність володіє хоч би одним можливим ключем. Один з них береться за *первинний ключ*. При виборі первинного ключа слід віддавати перевагу нескладеним ключам або ключам, складеним з мінімального числа атрибутів. Недоцільно також використовувати ключі з довгими текстовими значеннями (переважно використовувати цілочисельні атрибути). Так, для ідентифікації студента можна використовувати або унікальний номер залікової книжки, або набір з прізвища, імені, по батькові, номера групи і може бути додаткових атрибутів, оскільки не виключено появи в групі двох студентів з однаковими прізвищами та іменами. Погано також використовувати як ключ назву, наприклад “Закуска з плавлених сирків "Дружба" з шинкою”.

Не допускається, щоб первинний ключ стрижньової сутності (будь-який атрибут, що бере участь в первинному ключі) приймав невизначене значення. Інакше виникне суперечлива ситуація: з’явиться не існуючий екземпляр стрижньової сутності, що не володіє індивідуальністю. З тих же причин необхідно забезпечити унікальність *первинного ключа*.

Тепер про *зовнішні ключі*:

* Якщо сутність зв’язує сутність А і В, то вона повинна включати зовнішні ключі, відповідні первинним ключам сутності А і В.
* Якщо сутність В позначає сутність А, то вона повинна включати зовнішній ключ, відповідний первинному ключу суті А.

**Обмеження цілісності**

Хоча поняття реляційної моделі даних першим ввів основоположник реляційного підходу Едгар Кодд, найбільш поширене трактування реляційної моделі даних, мабуть, належить відомому популяризатору ідей Кодда Крістоферу Дейту, який відтворює її (з різними уточненнями) практично у всіх своїх книгах. Згідно трактуванню Дейта, реляційна модель складається з трьох частин, що описують різні аспекти реляційного підходу: структурної, маніпуляційної та цілісної частини.

1. У структурній частині моделі фіксується, що єдиною родовою структурою даних, яка використовується у реляційних БД, є нормалізовані n-арні відношення (таблиця). Визначаються поняття доменів, атрибутів, кортежів, заголовка, тіла і змінної відношення.

2. У маніпуляційній частині моделі визначаються два фундаментальні механізми маніпулювання реляційними БД – реляційна алгебра і реляційне числення. Основною функцією маніпуляційної частини реляційної моделі є забезпечення міри реляційності будь-якої конкретної мови реляційних БД: мова називається реляційною, якщо він володіє не меншою виразністю і потужністю, чим реляційна алгебра або реляційне числення.

3. У цілісній частині реляційної моделі даних фіксуються дві базові вимоги цілісності, які повинні підтримуватися в будь-якій реляційній СУБД. Перша вимога називається вимогою цілісності сутності (entity integrity). Об’єкту або сутності реального світу в реляційних БД відповідають кортежі відношень. Вимога цілісності сутності повністю звучить таким чином: *у будь-якої змінної відношення повинен існувати первинний ключ, і ніяке значення первинного ключа в у записах таблиці не повинне містити невизначених значень*. Щоб це формулювання було повністю зрозуміле, потрібно обговорити поняття невизначеного значення (NULL).

Теоретично будь-який запис, що заноситься у таблицю, повинен містити всі характеристики модельованої їм сутності, які потрібно зберегти в базі даних. Проте на практиці не всі ці характеристики можуть бути відомі до того моменту, коли потрібно зафіксувати сутність в базі даних. Простим прикладом може бути процедура ухвалення на роботу людини, розмір заробітної платні якого ще не визначений.

Едгар Кодд запропонував використовувати в таких випадках невизначені значення. Невизначене значення не належить ніякому типу даних і може бути присутнім серед значень будь-якого атрибуту, визначеного на будь-якому типі даних (якщо це явно не заборонено при визначенні атрибуту). Якщо а – це значення деякого типу даних або NULL, op – будь-яка двомісна «арифметична» операція цього типу даних (наприклад +), а lop – операція порівняння значень цього типа (наприклад =), то за визначенням:

а op NULL = NULL

NULL op а = NULL

а lop NULL = unknown

NULL lop а = unknown

Тут unknown – це третє значення логічного, або булевого, типу, що володіє наступними властивостями:

NOT unknown = unknown

true AND unknown = unknown

true OR unknown = true

false AND unknown = false

false OR unknown = unknown

Друга вимога, яка називається вимогою цілісності по посиланнях (referential integrity), є складнішою. Очевидно, що при дотриманні нормалізованості відношень складна сутність реального світу представляється в реляційній БД у вигляді декількох кортежів (записів) декількох відношень (таблиць). Вимога цілісності по посиланнях, або вимога *цілісності зовнішнього ключа, полягає в тому, що для кожного значення зовнішнього ключа повинен знайтися запис з таким же значенням первинного ключа, або значення зовнішнього ключа повинне бути повністю невизначеним* (тобто ні на що не указувати).

**Обмеження цілісності сутності і по посиланнях, які повинні підтримуватися СУБД.**

Для дотримання цілісності сутності досить гарантувати відсутність в будь-якій таблиці записів з одним і тим же значенням первинного ключа (і забороняти входження в значення первинного ключа невизначених значень).

З цілісністю по посиланнях справа йде дещо складніше. Зрозуміло, що при оновленні підпорядкованої таблиці (вставці нових записів або модифікації значення зовнішнього ключа в існуючих записах) досить стежити за тим, щоб не з’являлися некоректні значення зовнішнього ключа.

Але як бути при видаленні запису з таблиці, на яке існує посилання? Тут існують три підходи, кожний з яких підтримує цілісність по посиланнях. Перший підхід полягає в тому, що взагалі забороняється проводити видалення запису, для якого існують посилання (тобто спочатку потрібне або видалити записи, що посилаються, або відповідним чином змінити значення їх зовнішнього ключа). При другому підході при видаленні запису, на який є посилання, у всіх записах, що посилаються, значення зовнішнього ключа автоматично стає повністю невизначеним. Третій підхід (каскадне видалення) полягає в тому, що при видаленні запису з таблиці, на яке веде посилання, з таблиці, що посилається, автоматично віддаляються записи, що посилаються.

**Даталогічна модель бази даних**

Інфологічні моделі даних використовуються на ранніх стадіях проектування для опису структур даних у процесі розробки додатка, а даталогічні моделі вже підтримуються конкретними СУБД.

Даталогічна модель є моделлю логічного рівня і являє собою відображення логічних зв'язків між елементами даних безвідносно до їхнього змісту й середовищу зберігання. Ця модель будується в термінах інформаційних одиниць, припустимих у тієї конкретної СУБД, у середовищі якої ми проектуємо базу даних. Етап створення даталогічної моделі називається даталогічним проектуванням. Опис логічної структури бази даних мовою СУБД називається схемою.

Хоча даталогічне проектування є логічною структурою бази даних, на нього впливають можливості фізичної організації даних, що представляються конкретної СУБД. Тому знання особливостей фізичної організації даних є корисним при проектуванні логічної структури.

Логічна структура бази даних, а також сама заповнена даними база даних є відображенням реальної предметної області. Тому на вибір проектних розв'язків найбезпосередніший вплив виявляє специфіка відображуваної предметної області, відбита в інфологічній моделі.

Для реляційної бази даних проектування логічної структури полягає в тому, щоб розбити всю інформацію з файлів (відносинам), а також визначити состав полів (атрибутів) для кожного із цих файлів.

Одне з вимог -- реляційна база даних повинна бути нормалізована. Процес нормалізації має своєю метою усунення надмірності даних і полягає в приведенні до третьої нормальної форми (або до нормальної форми Бойса-Кодда).

Нормалізацією називається формальна процедура, у ході якої атрибути даних групуються в таблиці, а таблиці групуються в базу даних (БД).

Аналіз предметної області звичайно здійснюється на підставі відомих відомостей про неї з урахуванням мети проектування програмної системи.

Базами даних називають електронні сховища інформації, доступ до яких здійснюється за допомогою одного або декількох комп'ютерів.

Звичайно БД створюється для зберігання й доступу до даних, що містять відомості про деяку предметну область, тобто деякої області людської діяльності або області реального миру.

Для реляційної БД проектування логічної структури полягає в тому, щоб розбити всю інформацію з файлів ( по відносинах), а також визначити состав полів (атрибутів) для кожного із цих файлів

Одиницею, що зберігається в БД інформації, є таблиця. Кожна таблиця являє собою сукупність рядків і стовпців, де рядка відповідають екземпляру, а стовпці - атрибутам (ознакам, характеристикам, параметрам об'єкта, події, явища).

**Фізична модель** даних визначає всі компоненти логічної бази даних та послуги, необхідні для створення бази даних, або можуть бути компонуванням існуючої бази даних.

Фізична модель даних складається з структури таблиці, назв і значень стовпців, зовнішніх та первинних ключів та зв’язків між таблицями.

Фізична модель даних насамперед визначає всі моделі реляційних даних та об'єкти бази даних. Він створюється з використанням рідної мови бази даних системи управління базами даних (СУБД). Він також може бути створений шляхом перетворення логічної моделі. Фізична модель даних використовується адміністраторами баз даних для оцінки розміру систем баз даних та для планування потужностей. Обмеження моделі фізичних даних, такі як розмір, конфігурація та безпека, можуть змінюватися залежно від основної системи баз даних.