Проектування баз даних: міфи та методичні принципи

У статті описано найважливіші методичні принципи навчання моделюванню предметних областей та реалізації моделей у середовищі систем керування базами даних у межах шкільного курсу інформатики. Розглянуто також типові помилки у викладанні цієї теми.

Шкільний курс інформатики поєднує в собі дві складові: прикладну (набуття компетенцій, необхідних для повноцінного життя в інформаційному суспільстві) та розвивальну (розвиток розумових здібностей, певних типів мислення). Перше, що спадає на думку стосовно завдань розвивальної складової, – це розвиток алгоритмічного мислення, який, безперечно, є одним із найважливіших завдань курсу інформатики загалом. Однак досить часто він неправомірно проголошується єдиним завданням, самоціллю, а прикладні програми розглядаються як чужинці, що тільки заважають «правильному та інтелектуальному» навчанню програмуванню. Найприкріше, коли у категорію «вигнанців» потрапляє така технологія, як бази даних, хоча насправді її вивчення є не менш розвивальним, ніж програмування. Цього не помічають тому, що вона дає змогу розвинути інший, ніж алгоритмічний, тип мислення, який ми назвемо мисленням *структурним*. До цього типу мислення належать такі вміння:

- визначати параметри об'єктів та їх можливі значення;
- класифікувати явища та об'єкти;
- знаходити структурні та ієрархічні зв'язки між класами об'єктів, класифікувати зв’язки;
- розв’язувати задачі з обробки структур даних (передусім – формалізовувати вимоги щодо відбору даних за певними критеріями).

Образно кажучи, мова йде про вміння «розкладати дані по поличках», яке в сьогоднішньому переповненому інформацією світі надзвичайно важливе. Технологію створення й обробки баз даних категорично не можна перелічувати через кому в ряді «офісних» технологій. Вона незрівнянно фундаментальніша, ніж, скажімо, обробка текстів чи створення презентацій. СКБД по суті не є офісною програмою, MS Access включено в пакет Office лише з маркетингових міркувань.

В основі будь-якої бази даних лежить модель певної предметної області, яку також називають моделлю «сутність-зв’язок». Побудувати таку модель означає:

- виявити у предметній області сутності;
- з’ясувати, які параметри мають об’єкти сутностей;
- визначити, які між сутностями існують зв’язки.

Сутності – це множини однотипних об’єктів. Наприклад, сутність «Учні» складається з таких об’єктів, як учень Петров Владислав, учениця Петренко Інна та ін. Об’єкти сутності є однотипними в тому розумінні, що вони мають однаковий набір параметрів, наприклад, кожен учень характеризується віком, зростом, середньою успішністю тощо. Таким чином, виявляти сутності – це все одно, що класифікувати об’єкти, а загалом конструювання моделей «сутність-зв’язок» розвиває три перших з перелічених вище чотирьох компонентів структурного мислення. Уміння проектувати такі моделі стане міцним фундаментом для подальшого вивчення і СКБД, і об’єктно-орієнтованого програмування; це один з найважливіших розвивальних аспектів курсу інформатики загалом. Саме методиці навчання проектуванню моделей «сутність-зв’язок» та їх реалізації у середовищі СКБД присвячено цю статтю. Четвертий компонент структурного мислення розвиває інша ключова підтема курсу баз даних – конструювання запитів, методиці її викладання буде присвячено одну з наступних статей.

Не слід плутати поняття моделі предметної області з поняттям моделі даних. Модель даних – це система правил формального опису структур даних. Відомі такі моделі даних, як реляційна, ієрархічна, мережева, об'єктно-орієнтована, дедуктивна та ін. Модель предметної області не прив'язана до конкретної моделі даних, вона може бути реалізована в СКБД, що підтримують різні моделі даних.

На жаль, в українській шкільній інформатиці склалися негарні традиції переплутування та хибної інтерпретації деяких ключових понять, що належать власне моделюванню предметних областей та реляційній моделі даних, які погіршуються несвідомим наслідуванням певних концепцій застарілої ієрархічної моделі. Тому перш ніж розглядати методичні принципи навчання моделюванню предметних областей, спробуємо розвіяти деякі найпоширеніші міфи, що побутують у шкільній інформатичній освіті стосовно цієї теми.

Міф 1, кумедний. «Основною ознакою реляційної моделі даних є наявність зв'язків (relations) між таблицями. Звідси походить і назва моделі».

Насправді «зв'язок» англійською буде «relationship», а «relation» означає «відношення» – математичний об'єкт, підмножину декартового добутку кількох множин. Відношення зручно зображувати у вигляді таблиць і в теорії реляційних баз даних терміни «таблиця» і «відношення» майже еквівалентні. Тому правильніше було б назвати цю модель даних «табличною», а не калькувати її назву з англійської. Подання структур даних у вигляді таблиць і є визначальною ознакою реляційної моделі даних.

А кумедність міфу полягає в тому, що реляційна модель – єдина серед усіх моделей даних, у якій не передбачено жодних механізмів безпосереднього зв'язування об'єктів. В ієрархічній, мережевій та об'єктно-орієнтованій моделях даних зв'язки створюються за допомогою *вказівників* – полів, що належать одним об'єктам та вказують на місця в пам'яті, де розташовано інші об'єкти. У реляційній моделі зв'язки моделюються непрямо, за допомогою *зовнішніх ключів* – додаткових, штучно створених полів, які безпосередньо нікуди не вказують, а лише містять значення, за якими можна знайти зв'язані записи в іншій таблиці. Тому однією з характерних ознак реляційної моделі є якраз *відсутність* безпосередніх зв'язків між об'єктами даних.

Міф 2, абсурдний. «Є 4 типи зв'язків: один-до-одного, один-до-багатьох, багато-до-одного та багато-до-багатьох».

Цілком очевидно, що «один-до-багатьох» і «багато-до-одного» – це один той самий тип зв'язку. «Відмінність» між цими «типами зв'язків» суто лінгвістична: якщо сказати «учень вчиться у школі», вийде зв'язок «багато-до-одного», а якщо «у школі вчаться учні» – «один-до-багатьох», хоча в дійсності йдеться про один той самий зв'язок.

Ця помилка походить від переплутання концепцій реляційної та ієрархічної моделей даних, адже в реляційній моделі даних зв'язки симетричні (якщо школа 17 зв'язана з учнем Петровим, то і учень Петров – зі школою 17), а в ієрархічній моделі – спрямовані (з того, що школу 17 зв'язано з учнем Петровим ще не випливає, що учень Петров зв'язаний зі школою 17).

Міф 3, найпоширеніший і найстрашніший. «Таблиці A і B з'єднано зв'язком «один-до-одного», якщо одному запису A відповідає один запис B. Таблиці A і B з'єднано зв'язком «один-до-багатьох», якщо одному запису A відповідає багато записів B. Таблиці A і B з'єднано зв'язком «багато-до-багатьох», якщо багатьом записам A відповідає багато записів B».

Хибність такого означення зв'язку «багато-до-багатьох» стає очевидною, якщо зауважити, що і зв'язки «один-до-одного» та «один-до-багатьох» йому також задовольняють. Справді, як би не були зв'язані таблиці, а ***багатьом*** записам однієї таблиці завжди відповідатимуть багато записів іншої.

Хибність наведеного означення зв’язку «один-до-багатьох» стане очевидною, якщо зауважити, що йому задовольняє і зв’язок «багато-до-багатьох». Наприклад, є зв’язок «читач прочитав книжку» – зв’язок типу «багато-до-багатьох» між читачем і книжкою. Але, оскільки один читач прочитав багато книжок, цей зв’язок, згідно з наведеним вище означенням, можна було б класифікувати як зв’язок типу «один-до-багатьох».

Наведене вище означення зв’язку «один-до-одного» є неповним (порівняйте його з поданим нижче коректним означенням).

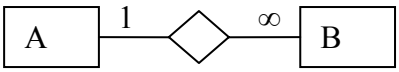
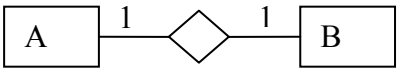
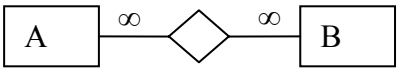
На жаль, спілкування автора з учителями інформатики свідчить, що більшість із них дають учням саме ці категорично хибні, неприпустимі означення, вони наведені і в деяких рекомендованих МОНМС підручниках. Звичайно, якщо у фундамент будови закладати такі міни, ні про яке коректне та ґрунтовне подальше вивчення баз даних не може йтися.

Для коректного визначення типу зв’язку між сутностями (таблицями) A і B слід відповісти на два запитання:

- Зі скількома об’єктами сутності A може бути зв’язано один об’єкт сутності B?
- Зі скількома об’єктами сутності B може бути зв’язано один об’єкт сутності A?

Як відповіді на ці запитання визначають типи зв’язків, і як вони графічно позначаються на моделі «сутність-зв’язок», показано в табл. 1.

Таблиця 1. Різновиди зв’язків

Зі скількома об’єктами A може бути зв’язано один об’єкт B?	Зі скількома об’єктами B може бути зв’язано один об’єкт A?	Тип зв’язку	Графічне позначення
з одним	з багатьма	один-до-багатьох	
з одним	з одним	один-до-одного	
з багатьма	з багатьма	багато-до-багатьох	

Найпоширеніша помилка з цього питання у шкільній інформатиці полягає в тому, що замість двох запитань для визначення типу зв’язку ставлять одне. Насправді відповідь на одне запитання визначає множинність зв’язку тільки з одного боку (тільки один символ "1" чи "∞"). Тому коректні означення типів зв’язку мають враховувати відповіді на два запитання і можуть бути такими.

- Сутності A і B з’єднано зв’язком «один-до-одного», якщо кожному об’єкту A відповідає не більше одного об’єкта B, а кожному об’єкту B – не більше одного об’єкта A.
- Сутності A і B з’єднано зв’язком «один-до-багатьох» (зображеним у першому рядку табл. 1), ***якщо кожному об’єкту B відповідає не більше одного об’єкта A***, але одному об’єкту A може відповідати довільна кількість об’єктів B.
- Сутності A і B з’єднано зв’язком «багато-до-багатьох», якщо одному об’єкту A може відповідати довільна кількість об’єктів B, а одному об’єкту B – довільна кількість об’єктів A.

Найчастіше «забувають» виділену жирним частину означення зв'язку «один-до-багатьох», хоча вона є важливішою за іншу його частину. Щоб зрозуміти це, поглянемо на зв'язки з іншої точки зору: зв'язкам «один-до-одного» та «один-до-багатьох» відповідають певні обмеження на кількість об'єктів, які можуть бути зв'язані з одним об'єктом іншої сутності, причому накладаються вони саме множинністю «1». Так, обмеження, що накладається зв'язком «один-до-багатьох», полягає в тому, що кожен об'єкт B не можна зв'язувати більше, ніж з одним об'єктом A. А у зворотний бік (символ «∞») обмеження немає: кількість об'єктів B, з яким може бути зв'язано кожен об'єкт A, необмежена. Тому зв'язки можна було б класифікувати так: у зв'язку «один-до-одного» обмеження на кількість об'єктів, з якими зв'язано один об'єкт іншої сутності, діє в обидва боки, у зв'язку «один-до-багатьох» – в один бік, а у зв'язку «багато-до-багатьох» – не діє зовсім. Це обмеження є різновидом обмежень цілісності для бази даних і тип зв'язку визначається саме тим, як воно підтримується чи не підтримується (тому на схемі даних у MS Access тип зв'язку відображуються тільки після встановлення прапорця «з підтримкою обмежень цілісності»). Годі й казати, що в означенні типу зв'язку не можна забувати про визначальну ознаку цього типу.

Зауважимо також, що в загальному випадку, коли зв'язок між сутностями A і B має з боку сутності A множинність «один», кожен об'єкт B зв'язано не точно з одним, а не більше, ніж з одним об'єктом A (тобто об'єкт B зв'язано з об'єктом A не обов'язково). Якщо ж кожен об'єкт B повинен бути зв'язаний з якимось об'єктом A, зв'язок називається обов'язковим з боку сутності B. У MS Access обов'язковість зв'язку задається за допомогою властивості «Обов'язкове поле» зовнішнього ключа тієї сутності, з боку якої зв'язок обов'язковий.

Міф 4, термінологічний. Зовнішній ключ – це ключ іншої таблиці, з якою зв'язано дану таблицю.

Насправді зовнішній ключ – це додаткове поле (або набір полів), значення якого належать множині значень первинного ключа іншої таблиці. Наприклад, якщо є зв'язані таблиці «Учні» та «Класи», то зовнішнім ключем таблиці «Учні» буде не ключ таблиці «Класи», а поле «клас» таблиці учнів (див. рис. 2 і 3). Зовнішні ключі буде детально розглянуто далі, а зараз зазначимо, що причиною помилкового тлумачення цього поняття є його невдала назва: природне значення слова «зовнішній» (англ. foreign) – той, який не належить розглядуваному об'єкту – у даному випадку незастосовне. Правильніше зовнішні ключі було б назвати полями зв'язку.

Тепер перейдемо до розгляду деяких методичних принципів викладання основ курсу «Бази даних».

Принцип 1. Перші 3–5 уроків курсу «Основи баз даних» слід присвятити побудові моделей «сутність-зв'язок» предметних областей «на папері».

Це теоретичні уроки, які мають проходити або з вимкнутими комп'ютерами, або з використанням редактора моделей «сутність-зв'язок», наприклад http://www.gliffy.com. Вивчення баз даних не можна починати з роботи в середовищі СКБД, оскільки тоді технологічні аспекти замулюють сутність справи: учень напевно запам'ятає склад меню «Файл» у MS Access і буде тішитися з того, але не зрозуміє принцип визначення зв'язків між сутностями предметної області, не усвідомлюючи, що друге незрівнянно важливіше від першого.

Крім того, як уже зазначалося, модель предметної області абстрагована від особливостей тієї чи іншої моделі даних. Побудова моделі предметної області – це найперший крок у розробці БД. Модель має бути описана в термінах сутностей, об'єктів,

параметрів, а не таблиць, записів, полів і вже потім може бути реалізована як в реляційній СКБД, так і в нереляційній, де таблиць, записів і полів немає. На моделі предметної області сутності та зв'язки позначаються умовно – прямокутниками та ромбами і виявити зв'язок між сутностями означає саме «виявити зв'язок», а не «перетягнути на схемі даних ключове поле однієї таблиці на зовнішній ключ іншої». Вивчати СКБД, не засвоївши попередньо моделювання предметних областей, – це вивчати механізми реалізації певної концепції у програмному середовищі, не зрозумівши самої концепції.

Принцип 2. Учні мають зрозуміти та навчитися застосовувати критерій правильності моделі «сутність-зв'язок» предметної області.

Зазвичай моделюванню предметних областей ми навчаємо на конкретних прикладах (і це правильно), кажемо: учень вчиться у класі – значить, між сутностями «учень» і «клас» існує зв'язок, учитель викладає у класі – значить, сутності «учитель» і «клас» також зв'язані. Але навіть за такою простою предметною областю з трьома сутностями постають «незручні» запитання, наприклад, як бути з безпосереднім зв'язком між учителем та учнем? Адже такий зв'язок, очевидно, існує (вчитель навчає учня), однак інтуїтивно ми розуміємо, що відображати його на моделі не потрібно, якщо тільки не йдеться про позакласні індивідуальні заняття (рис. 1). А чому? Або інший приклад: у предметній області розглядаються організації та їхні підрозділи. Ми розуміємо, що назва організації, її адреса, форма власності тощо – це атрибути сутності «організація», а назва підрозділу, кількість працівників у ньому та ін. – атрибути іншої сутності, «підрозділ». Але чому не можна віднести назву підрозділу і кількість працівників у підрозділі до атрибутів організації, не виділяючи окремої сутності «підрозділ»? Які критерії поділу набору атрибутів на кілька сутностей чи об'єднання їх в одній сутності? Які критерії необхідності відображення на моделі тих чи інших зв'язків?

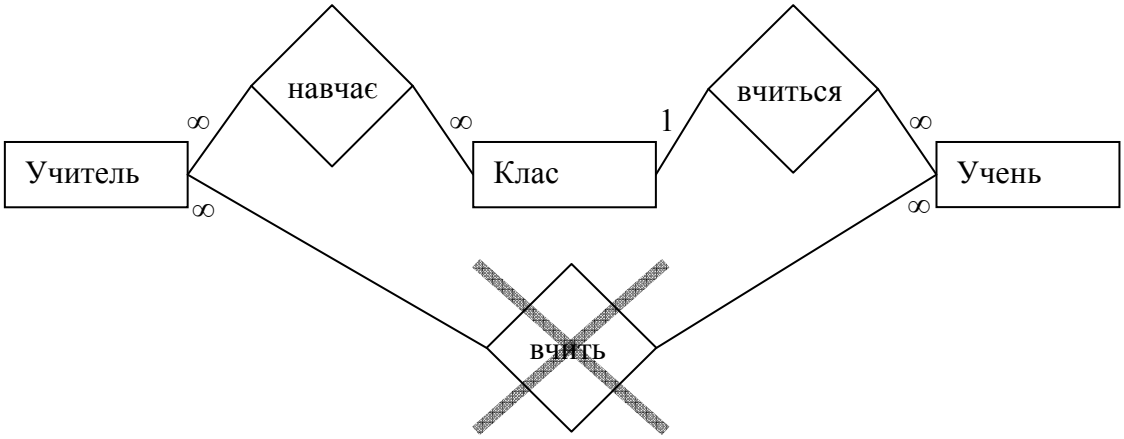


Рис. 1. Доречні та зайві зв'язки на моделі «сутність-зв'язок»

Існує єдиний простий універсальний критерій правильності моделі «сутність-зв'язок» предметної області (а отже, і схеми бази даних), який можна назвати принципом ненадлишковості або головним принципом семантичного моделювання: *модель «сутність-зв'язок» повинна дозволяти зберігання будь-яких відомостей лише в одному місці.* Якщо застосувати цей критерій до розглянутих прикладів, то прямий зв'язок між сутностями вчителів і учнів виявиться зайвим, оскільки відомості про те, які вчителі яких учнів навчають, зберігаються і завдяки зв'язкам «учень-клас» та «вчитель-клас», а об'єднання в одну сутність підрозділів і організацій призведе до багаторазового

дублювання атрибутів організації (назви, адреси тощо), які доведеться зазначати для кожного підрозділу.

Розуміння принципу ненадлишковості є критично важливим з точки зору досягнення основних навчальних цілей курсу «Основи баз даних» і тому учні мають виконати кілька прикладів на застосування цього принципу для перевірки коректності моделей предметних областей і визначення помилок у них.

Принцип 3. Під час навчання створенню зв'язків у реляційній СКБД найважливішим є поняття зовнішнього ключа.

Як уже зазначалося, зовнішній ключ – це додаткове поле (або набір полів), значення якого належать множині значень первинного ключа іншої таблиці. Слово «додаткове» ми вжили в тому розумінні, що в «паперовій» моделі «сутність-зв'язок» поля зовнішніх ключів не зазначаються, вони не властиві сутностям як таким і створюються тільки під час реалізації моделі «сутність-зв'язок» у реляційній (і ні в якій іншій) базі даних. Наприклад, поле «клас» не є властивістю учня як такого, однак ми його додаємо до таблиці «Учні», щоб промоделювати зв'язок між учнем і класом. Зв'язаними вважаються ті записи таблиць «Учні» та «Класи», у яких значення полів «клас» (зовнішній ключ) та «назва» (первинний ключ) однакові (рис. 2). З останнього твердження стає зрозумілим означення зовнішнього ключа: якщо значення поля «клас» таблиці «Учні» належать множині значень поля «назва» таблиці «Класи», то промоделювати зв'язок між учнями так класами можливо, якщо ж у поле «клас» ввести неіснуюче в полі «назва» значення, скажімо, «32Д», то такого учня буде «зв'язано» незрозуміло з яким класом, тобто поле «клас» перестане виконувати функцію моделювання зв'язку.

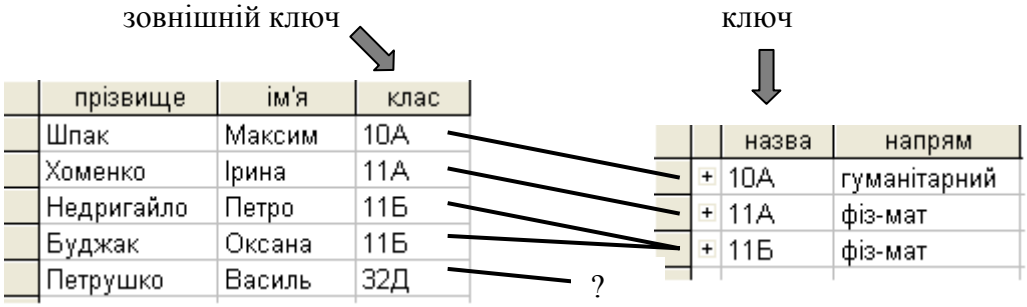


Рис. 2. Зв'язування записів за допомогою зовнішнього ключа

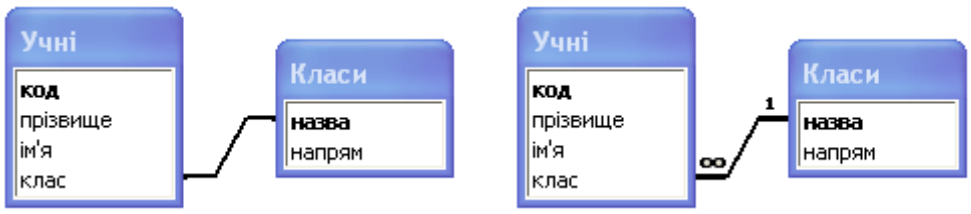
Зовнішні ключі – головний і єдиний механізм створення зв'язків між таблицями в реляційній БД, і тому не дати означення цього поняття абсолютно неможливо. Практика свідчить, що учні та студенти, які його не засвоїли, зв'язують на схемі даних паспорт учителя з назвою класу і т.п. Щоб унеможливити подібні помилки, крім викладання поняття зовнішнього ключа, потрібно дотримуватися ще одного принципу.

Принцип 4. Учні повинні створювати зв'язки не лише на схемі даних, а й безпосередньо між записами, за допомогою самих даних.

Схема даних у MS Access є тільки унаочненням, графічним відображенням зв'язків, вона не містить зв'язків як таких. Створити зв'язок між учнем та класом означає ввести в поле «клас» запису таблиці «Учні», що стосується учня Петрова, значення «10А», а схему даних можна було б не використовувати взагалі, адже в більшості комерційних реляційних СКБД такого засобу немає (до речі, термін «схема даних» некоректний, вона мала б називатися *схемою бази даних*).

Однак насправді схема даних в MS Access – якраз зручний і наочний навчальний засіб, ігнорувати його нерозумно. Але потрібно чітко усвідомлювати, що саме відбувається з базою даних, коли ви виконуєте ті чи інші дії з її схемою.

- Коли створюється зв'язок між таблицями без встановлення прапорця «з підтримкою обмежень цілісності», на схемі даних з'являється лінія зв'язку без зазначення множинностей (рис. 3а), але з самою базою даних не відбувається рівним чином нічого. Ця лінія є лише підказкою для розробника БД, а також для майстрів і конструкторів MS Access, що дає їм змогу визначити, які записи вважати зв'язаними під час автоматизованої побудови запитів, форм або звітів за кількома таблицями. А саме, якщо розглядати рис. 3а, зв'язаними вважатимуться ті записи таблиць «Учні» та «Класи», для яких значення поля «клас» у таблиці «Учні» (тобто значення зовнішнього ключа) дорівнює значенню поля «назва» в таблиці «Класи» (тобто значенню первинного ключа іншої таблиці).
- Якщо під час побудови зв'язку встановити прапорець «з підтримкою обмежень цілісності» (рис. 3б), на схемі даних біля кінців зв'язку з'являться позначення множинностей (символи «1» або «∞»), а в самій базі даних буде створено обмеження цілісності: *у поле зовнішнього ключа тепер не можна буде ввести значення, якого немає серед значень того первинного ключа, з яким цей зовнішній ключ зв'язано*.



а б

Рис. 3. Відображення зв'язків на схемі даних у MS Access: а – без підтримки обмежень цілісності; б – з підтримкою обмежень цілісності

З точки зору певного зв'язку таблиця, яка містить зовнішній ключ, у MS Access називається зв'язаною, а інша — головною. Однак зазначимо, що така термінологія нестандартна і використовується лише в цій СКБД.

Яким же чином у реляційній БД визначається тип зв'язку? Подивимося уважно на рис. 2. У кожну клітинку поля «клас» (зовнішнього ключа) можна ввести тільки одне значення, тому будь-якого учня може бути зв'язано тільки з одним класом. Інакше кажучи, множинність зв'язку з боку головної таблиці (тієї, яка не містить зовнішнього ключа) завжди «один». Множинність зв'язку з іншого боку визначається властивостями поля зовнішнього ключа. Якщо значення в цьому полі можуть повторюватися, то множинність зв'язку з боку зв'язаної таблиці буде «багато», оскільки багато її записів зможуть посилатися на той самий запис головної таблиці – таку ситуацію зображено на рис. 2. Якщо ж значення. зовнішнього ключа не можуть повторюватися, множинність зв'язку з боку зв'язаної таблиці буде «один» (рис. 4).

Значення зовнішнього ключа не можуть повторюватися у двох випадках.

- Зовнішній ключ водночас є і первинним ключем (рис. 4а).

- Зовнішній ключ є індексованим полем без повторень (у полів таблиць MS Access є така властивість). Цю ситуацію зображено на рис. 4б; зовнішнім ключем є поле «двигун».

а б

Рис. 4. Моделювання зв’язку «один-до-одного»: а – зовнішній ключ є водночас і первинним ключем; б – зовнішній ключ є індексованим полем без повторень

Зауважимо, що зовнішній ключ є водночас і первинним тільки в разі моделювання так званого зв’язку «загальний вид – різновид» між певною загальною сутністю та сутністю-різновидом. Так, на рис. 4а таблиці зображують загальну сутність «транспортний засіб» та її різновид – сутність «трамвай». З першого погляду може видатися, що тип зв’язку між цими сутностями встановлено неправильно, однак якщо послуговатися наведеним вище критерієм визначення типу зв’язку, все стає на свої місця.

- Скількома транспортними засобами є один трамвай? – одним.
- Скількома трамваями може бути один транспортний засіб? – не більш, ніж одним.

В усіх інших випадках зв’язки «один-до-одного» моделюються за допомогою індексованих полів без повторень, які, фактично, утворюють додаткові ключі таблиць (зазначимо, що взагалі в реляційній таблиці може бути кілька ключів, але MS Access не підтримує такої можливості в явному вигляді). Також варто зазначити, що основне призначення індексованих полів – не моделювання зв’язків, а забезпечення швидкого пошуку даних, однак цей аспект виходить за межі шкільного курсу інформатики.

Постає питання: якщо множинність зв’язку з боку головної таблиці – завжди «один», то яким чином моделювати зв’язки «багато-до-багатьох»? Відповідь стає зрозумілою, якщо зауважити, що зв’язок «багато-до-багатьох» еквівалентний двом зв’язкам «один-до-багатьох» із допоміжною таблицею, таблицею зв’язку (на рис. 5 це таблиця «Викладання»). Одного вчителя може бути зв’язано з багатьма записами таблиці «Викладання», а отже, і з багатьма класами. Один клас також може бути зв’язано з багатьма записами таблиці «Викладання», а отже, і з багатьма вчителями. Інших, більш прямих, механізмів моделювання зв’язків «багато-до-багатьох» у реляційних базах даних не існує.

Рис. 5. Зв’язок «багато-до-багатьох» на схемі даних

Зауважимо, що первинний ключ таблиці зв'язку має складатися з усіх її зовнішніх ключів. Інакше порушуватиметься основний принцип семантичного моделювання: відомості про зв'язок тих самих двох об'єктів зможуть зберігатися багаторазово.

Крім визначення типу зв'язку за множинністю, існують інші способи класифікації та різновиди зв'язків між сутностями, які достатньо важливі, але майже ніколи не вивчаються в шкільному курсі інформатики. Однак їх включено в розширену версію курсу за вибором «Основи баз даних» і детально описано в [1] і [2]. Якщо казати про методику викладання курсу основ баз даних загалом, то питанням, яке заслуговує на окреме дослідження, є також конструювання запитів, що, як уже зазначалося, буде розглянуто в наступних статтях.

Література

1. Завадський І.О. Основи баз даних. Навчальний посібник. К., вид. Завадський І.О., 2011. – 192 с, http://zavadsky.at.ua.
2. Дейт К.Дж. Введение в системы баз данных. М.: Вильямс, 2005. – 1328 с.