Л3. Відношення, заголовок та тіло відношення

Поняття відношення (таблиці) є найбільш фундаментальним в реляційній моделі даних, тому що n-арне відношення є єдиною базовою структурою даних, що зберігаються в реляційній базі даних. Сам термін *відношення* не зовсім точний, виходячи з того, що, говорячи про будь-які дані, що зберігаються, потрібно мати на увазі *тип* цих даних, *значення* цього типу та *змінні,* в яких зберігаються значення. Відповідно, для уточнення терміну *відношення* виділяються поняття *заголовку відношення, значення відношення* та *змінної відношення.* Крім цього, додатково потрібне поняття *кортежу.*

Заголовком (або *схемою)* відношення *r (Hr)* називається скінчена множина впорядкованих пар вигляду *<A, T>,* де *A* має назву атрибута, а *T* позначає ім'я деякого базового типу або раніше визначеного домену (Нагадаю, що **доменом** називається набір значень елементів даних одного типу, що відповідають заданим умовам). За означенням необхідно, щоб усі імена атрибутів у заголовку відношення були різними. У прикладі на мал. 3.1 заголовком відношення СТУДЕНТИ є множина пар {<*№* особ.справи, Рядок_номер_справи_студента>, <*Прізвище,* Рядок_прізвище>, <*Ім'я*, Рядок_ім'я>, <*По-батькові*, Рядок_по-батькові>, <Дата народж., Дата>, <Група, Рядок_Зовнішній ключ>}.

№ особ. справи	Прізвище	Ім'я	По-батькові	Дата народж.	Група
16493	Сергєєв	Петро	Михайлович	01.01.76	К23
16593	Петрова	Анна	Володимирівна	15.03.75	К24
16693	Науменко	Андрій	Борисович	14.04.76	К25

Мал.3.1. Приклад реляційного відношення (таблиці) СТУДЕНТИ

Якщо усі атрибути заголовку таблиці-відношення визначені на різних доменах, то раціональним є застосування для найменування атрибутів імен відповідних доменів. Іншими словами у БД можуть бути заведені домени, імена яких будуть надаватися атрибутам довільних таблиць.

Кортежем відношення *tr*, що відповідає заголовку *Hr*, називається множина впорядкованих триплетів вигляду *<A, T, v>,* по одному такому триплету для кожного атрибуту в *Hr*. Третій елемент - *v* у триплеті *<A, T, v>* обов'язково повинен бути допустимим значенням типу даних або домену *T*. Заголовку відношення СТУДЕНТИ відповідає, наприклад, такий кортеж {<*№* особ.справи, Рядок_номер_справи_студента, (16493)>,

<*Прізвище,* Рядок_прізвище, (Сергєєв)>,<*Ім'я*, Рядок_ім'я, (Петро)>, <*По-батькові*, Рядок_по-батькові, (Михайлович)>, <Дата народж., (01.01.76)>, <Група, (К23)>}.

Тілом Br відношення *r* називається довільна множина кортежів *tr*. Частинний випадок тіла відношення СТУДЕНТИ показаний на мал. 3.1. Зауважимо, що у загальному випадку, можуть існувати такі кортежі *tr*, які відповідають *Hr*, але не входять до *Br* (це можливо, коли деяке значення атрибута є невизначеним).

Значенням Vr відношення *r* називається пара множин *Hr* і *Br*. Одне з допустимих значень відношення СТУДЕНТИ показано на мал. 3.1.

З наведених означень випливає поняття *схеми реляційної бази даних* - набір пар <*ім'я_VARr; Hr*>, де під *VARr* розуміється іменований контейнер, який може містити будь-яке допустиме значення *Vr*. Іншими словами, *схема* РБД включає імена та заголовки усіх змінних таблиці-відношення, які визначені в базі даних. Реляційна база даних - це набір пар <*VARr; Hr*>.

Зауважимо, що у класичних реляційних базах даних після визначення схеми бази даних можуть мінятися лише значення змінних відношень. Однак у більшості сучасних реалізацій дозволяється і зміна схеми бази даних: визначення нових та зміна заголовків існуючих змінних відношень. Такий процес прийнято називати *еволюцією схеми бази даних.*

Надлишковість даних та аномалії.

При розробці РБД можуть виникнути проблеми, що пов'язані з *надлишковістю* даних та *аномаліями*. Під надлишковістю розуміють дублювання даних у таблицях-відношеннях. Існує просте дублювання та надлишкове.

Просте дублювання є допустимим (на мал. 3.2 можна переглянути приклад простого дублювання).

СТУДЕНТИ

№ особ. справи	Місце проживання
16493	Гурт. 16
16593	Київ
16693	Гурт.16

Мал. 3.2. Приклад простого дублювання

Двоє студентів проживають у 16 гуртожитку. Якщо прибрати одне зі значень «Гурт. 16» разом з кортежем, то буде втрачено інформацію про одного з студентів. Це приклад аномалії видалення.

Якщо хтось з цих студентів поміняє своє місце проживання, то провести зміну значення у полі нескладно. Але, якщо припустити, що студенти мають переселитися у інший гуртожиток, або виселитися (з різних обставин), то потрібно оновлення зробити для усіх, хто мешкав у гуртожитку. Якщо цього не зробити, виникає невідповідність, яка пов'язана з аномалією оновлення.

Аномалія вводу (введення) даних виникає у випадку внесення нових записів (кортежів), у полях яких присутні недопустимі значення (наприклад, у нашій таблиці відсутнє значення або таке, що неприпустиме для позначення місця проживання).

Прикладом надлишкового дублювання може служити випадок таблиці СТУДЕНТИ, у якій крім місця проживання, буде фіксуватися Адреса (або номер кімнати і т.і.).

№ особ. справи	Місце проживання	Адреса
16493	Гурт. 16	Сеченова, 6
16593	Київ	Бойчука, 10, кв. 7
16693	Гурт.16	Сеченова, 6

Мал. 3.3. Приклад надлишкового дублювання

Якщо замість повторення поставити деяке спеціальне позначення (типу прочерку), виникають проблеми:

- Для Адреси «Сеченова, 6» виникне необхідність шукати значення поля «Місце проживання»
- У пам'яті все одно буде відведено відповідне місце
- Якщо прибрати студента з записом Адреси «Сеченова, 6», усі інші записи про студентів з 16 гуртожитку втрачають адресу.

Від дублювання звільняються розбиттям таблиці на декілька. У наведеному випадку можна запропонувати такий варіант

№ особ. справи	Місце проживання
16493	Гурт. 16
16593	Київ
16693	Гурт.16

Місце проживання	Адреса

Гурт. 16	Сеченова, 6
Київ	Бойчука, 10, кв. 7
…	…

Мал. 3.3. Приклад розбиття таблиці на 2 відношення

Одним з варіантів створення схеми РБД з декількома таблицями є застосування зв’язків між відношеннями.

Зв’язок "один до одного" (між двома відношеннями)

Розглянемо трохи інший приклад. Маємо таблицю, у якій зберігаються дані про клієнтів та їх замовлення. Припустимо, у конкретний момент часу один клієнт може зробити лише одне замовлення. У цьому випадку між таблицею КЛІЄНТ та ЗАМОВЛЕННЯ встановлюється взаємозв’язок "один до одного" (на схемах позначається одинарними стрілками).

Між даними, що зберігаються в таблицях КЛІЄНТ та ЗАМОВЛЕННЯ, буде існувати зв’язок, в якому кожен запис в одному відношенні буде однозначно вказувати на запис в іншому. Нескладно навести приклад такого взаємозв’язку між даними. У жодному з відношень не може існувати запис, не зв’язаний з деяким записом в іншому відношенні.

Зв’язок "один до багатьох" (між двома відношеннями)

У конкретний момент часу один клієнт може стати власником декількох моделей автомобілів, при цьому декілька клієнтів не можуть бути власниками одного автомобіля. Взаємозв’язок "один до багатьох" позначається за допомогою одинарної стрілки у напряму до "одного" і подвійної стрілки у напряму до "багатьох".

У цьому випадку одному запису даних першого відношення (його часто називають батьківським або основним) буде відповідати декілька записів другого відношення (дочірнього або підлеглого). Взаємозв’язок "один до багатьох" є дуже розповсюдженим при розробці реляційних баз даних. В якості батьківського відношення часто використовують довідник, а у дочірньому зберігаються унікальні ключі для доступу до записів довідника. У нашому прикладі в якості такого довідника можна представити таблицю КЛІЄНТ, у якій зберігаються відомості про усіх клієнтів. При зверненні до запису про конкретного клієнта доступним є список усіх покупок, які він зробив і відомості про які зберігаються в таблиці МОДЕЛЬ АВТОМОБІЛЯ. У випадку, якщо у дочірньому відношенні будуть деякі записи, для яких немає відповідних записів в таблиці КЛІЄНТ, то їх не буде видно. За таких обставин говорять, що таблиця містить загублені (одинокі, втрачені) записи. Це неприпустимо і у

подальшому потребує модифікації таблиці. Далі познайомимося, як запобігти таким ситуаціям.

Якщо переглядати записи таблиці МОДЕЛЬ АВТОМОБІЛЯ, то з поля КЛИЕНТ можна отримати дані про клієнта, який купив даний автомобіль. Для втрачених записів відомості про клієнта не буде отримано.

Зв’язок "багато до багатьох" (між двома відношеннями)

У прикладі кожен продавець може обслуговувати декількох клієнтів. З іншого боку, купуючи автомобілі в різний час, кожен клієнт може обслуговуватися різними продавцями. Між відношеннями КЛІЄНТ і ПРОДАВЕЦЬ існує взаємозв’язок "багато до багатьох". Такий взаємозв’язок позначається подвійними стрілками. При перегляді даних у таблиці КЛІЄНТ можна дізнатися, які продавці обслуговували конкретного клієнта. Однак в таблиці ПРОДАВЕЦЬ у цьому випадку ми маємо завести декілька записів для кожного продавця. Кожен рядок буде відповідати кожному обслуговуванню продавцем клієнта. При такому підході можна зіштовхнутися з великими проблемами. Наприклад, неможливо ввести в таблицю ПРОДАВЕЦЬ унікальний ключ для кожного продавця, тому що один продавець буде обслуговувати декількох клієнтів, і з’являється декілька записів для одного й того ж продавця.

Приклад відношення зі зв’язком «багато-до-багатьох»

КЛІЄНТ

Ключ	Назва Клієнта
1	Фасад
2	Трюм ЛТД
3	ТОВ Жоржетта

Мал. 3.4. Відношення Клієнт

Додаткова таблиця (Покупка)

Клієнт	Продавець
1	3
2	1
1	1

Мал. 3.5. Відношення Покупка

ПРОДАВЕЦЬ

Ключ	Продавець
1	Microsoft Trade Line
2	Oracle Union Limited
3	Idea Production

Мал. 3.5. Відношення Продавець