УДК 004.652.42

С. О. Кулик, В.К. Овсяк
Українська академія друкарства

ОПИС СТРУКТУРНОЇ ЧАСТИНИ РЕЛЯЦІЙНОЇ МОДЕЛІ БАЗ ДАНИХ ЗАСОБАМИ АЛГЕБРИ АЛГОРИТМІВ

Описана структурна частина реляційної моделі баз даних засобами алгебри алгоритмів. Мовою алгебри алгоритмів описано поняття домену, кортежу, відношення, ключа.

Description of structural part of relational model by means algebra of algorithms are considered here . The concept of domain, cortege, relation and key are described by language of algebra of algorithms

Постановка проблеми

Абсолютна більшість баз даних сучасних інформаційних систем використовують у тій чи іншій мірі реляційну модель даних. Найбільш поширені системи управління базами даних (СУБД) Oracle, MySQL, MS-SQL Server, PostgreSQL, IBM DB2, Informix та інші. Будь-яка реляційна СУБД використовує для опису операцій у базах даних (БД) один із діалектів мови структурованих запитів SQL — Structured Query Language. Будь-який діалект чи версія реалізації SQL грунтується на операціях реляційної алгебри але втілює альтернативний підхід, який називається реляційним численням. Реляційна алгебра визначає у явному вигляді набір операцій для отримання із наявних відношень потрібного відношення. Реляційне числення на відміну від реляційної алгебри задає систему позначень для визначення потрібного відношення у термінах існуючих відношень. Реляційна алгебра і реляційне числення логічно еквівалентні. Кожному виразу в алгебрі відповідає еквівалентний вираз в численні, і кожному виразу числення відповідає еквівалентний вираз в алгебрі.

Опис реляційної моделі засобами алгебри алгоритмів [1] дасть можливість розв'язати задачі синтезу, аналізу та дослідження алгоритмів роботи баз даних. Єдиний універсальний опис дає можливість створити інформаційну технологію автоматизованої комп'ютерної реалізації бази даних та інформаційної системи взагалі на будь-яких платформах, СУБД та операційних системах.

Складові баз даних і елементи алгебри алгоритмів

Згідно К. Дейту [2, с.104] реляційна модель баз даних складається з трьох частин:

1. Структурної частини.
2. Цілісної частини.
3. Маніпуляційної частини.

Структурна частина описує, які об'єкти розглядаються реляційною моделлю. Постулюється, що єдиною структурою даних, використовуваною в реляційній моделі, є нормалізовані *n* - нарні відношення [3].

Цілісна частина описує обмеження спеціального вигляду, які виконуватися для будь-яких відношень у довільних реляційних базах даних. Це цілісність суті та цілісність зовнішніх ключів.

Маніпуляційна частина описує два еквівалентних способи маніпулювання реляційними даними - реляційну алгебру та реляційне числення.

Структурна частина є основою реляційної алгебри, яку складають фундаментальні поняття домену, відношення та кортежу [4, 5].

Основу мови алгебри алгоритмів складають унітерми та операції. Унітерми - це вирази мови, якими позначають об'єкти [1, 6]. Наприклад, унітермами є 1, 7, 13, *a, k, l, x, y, z, x, y=x, y=кx, y>x, y-x2, z=x&y, z=xvy, p:=q*, 0+1, *x–y, f(x,y), A(x), F(x,y,z)*.

Унітерми, які не залежать ні від однієї змінної поділяються на константи (наприклад, 0, 2), параметри (наприклад, *a, k, l*) і змінні (*x, y, z*).

Унітерми, які залежать від однієї або декількох змінних поділяються на предметні (наприклад, *p: =q, y=x, y=a+кx, y=x , z-x&y, x, y=x+z*) та змінні (наприклад, *F(x), S(l,x), R(x,y), P(x,y,z)*). Змінні унітерми, які залежать від однієї або декількох змінних, позначаються великими літерами латинського алфавіту з будь-якими індексами або без них.

Алгебра алгоритмів має такі операції [1, 6, 7]:


секвентування;


елімінування;

реверсування;


паралелення;


циклічного секвентування;


циклічного елімінування;


циклічного паралелення;

Формалізація понять структурної частини баз даних

У реляційної моделі кожне значення, змінна, параметр, оператор та особливо кожний реляційний атрибут відноситься до того чи іншого типу.

Едгар Кодд для типу використовував поняття домен. Співавтор теорії реляційних БД та послідовник Е. Кодда, Кристофер Дейт в останній роботі [2, с.166] вже використовує замість домену термін тип. Дейт пояснює це тим, що тип є ближчим до мов програмування, тобто до реалізації БД.

У реляційній алгебрі поняття домену та типу повністю збігаються. У даній роботі будемо дотримуватись терміна домен, та позначати його літерою *D*.

Подібно типам у програмуванні будь-який домен є або визначеним системою або визначеним користувачем. Крім того подібно типам у мовах програмування з кожним доменом пов'язана множина операторів, які можна застосовувати до множини значень даного кортежу. Це означає що, операції із значеннями вказаного типу можуть виконуватися виключно за допомогою операторів означених для цього типу.

Математично домен є допустимою потенційною обмеженою множиною значень даного типу.

Тип є множина значень означена в комп'ютерній системі. Тип даних позначимо літерою Т.

Тип накладає певні обмеження на значення та змінні. До прикладів таких типів відносяться INTEGER (множина всіх цілих чисел), CHAR (множина всіх символьних рядків), і так далі.

Область значень кожного типу даних можна визначити за допомогою секвентних областей значень [6].

Наприклад, заламо такі секвентні області значень:

$$Q_1 = \overbrace{A, B, C, \ldots, Z; a, b, c, \ldots, z}$$

$$Q_2 = \overbrace{А, Б, В, \ldots, Я; а, б, в, \ldots, я}$$

$$Q_3 = \overbrace{0; 1; 2; \ldots; 9}$$

$$Q_4 = \overbrace{+, -, \wedge, *, /, =}$$

$$Q_5 = \overbrace{(), \ldots, :, \ldots, -, ?, !, \_}$$

$$Q_6 = \overbrace{\&, |, <, >, \neq}$$

$$Q_7 = \overbrace{@, \$, \#, \%, \sim}$$

де Q_1 – латинський алфавіт, Q_2 – укр. кирилиця, Q_3 – цифри десяткової системи числення, Q_4 – знаки математичних операцій, Q_5 – знаки пунктуації, Q_6 – знаки математичної логіки і відношень, Q_7 – спеціальні символи.

Тоді типи CHAR та INTEGER будуть означені як:

$$T(CHAR) \in \overbrace{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7}$$

$$T(INTEGER) \in \overbrace{Q_3, -}$$

В алгебрі алгоритмів для опису доменів є **секвентний домен**, тобто домен D, який в алгебрі алгоритмів є впорядкованою секвентною областю елементів $d_1, d_2, d_3, \ldots, d_n$

$$D = \overbrace{d_1, d_2, d_3, \ldots, d_n}$$

Кожний елемент секвентного домену є унітерм–значення, який належить до типу, на якому означений домен:

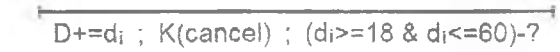
$$D \in T$$

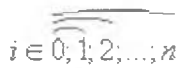
Домен можна розглядати як підмножина значень деякого типу даних таких, що мають певний сенс. Саме таке семантичне трактування домену відрізняє його від поняття математичної підмножини. Якщо тип даних можна рахувати безліччю всіх можливих значень даного типу, то домен нагадує підмножину в цій множині.

Наприклад, домен , що має сенс «вік співробітника» математично можна описати як така підмножина множини натуральних чисел:

$$D = \{n \in \mathbb{N} : n \geq 18 \text{ and } n \leq 60\}$$

Умова, обмежуюча додавання чи оновлення елементу d_i у секвентний домен D - «вік співробітника» буде виглядати так:





де, K – унітерм коментар «відміна дії», i - номер елемента домену.

Домен характеризується такими властивостями:

1. Домен має унікальну назву (в межах бази даних).
2. Домен визначений на деякому простому типі даних або на іншому домені.
3. Домен може мати деяку логічну умову, що забезпечує опис підмножини даних, допустимих для даного домену.
4. Домен несе певне смислове навантаження.

Перш за все домен відображає семантику, визначену предметною областю. Може бути декілька доменів, збіжних як підмножини, але такі що несуть різний зміст. Наприклад, домени "Вага деталі" і "Наявна кількість" можна однаково описати сукупністю невід'ємних цілих чисел, але зміст цих доменів буде різним, і це будуть різні домени.

Основне значення доменів полягає у тому, що домени обмежують порівняння. Некоректно з логічної точки зору порівнювати значення з різних доменів, навіть якщо вони мають однаковий тип. У цьому виявляється змістовне обмеження доменів. Синтаксично правильний запит "видати список всіх деталей, у яких вага деталі більше наявної кількості" не відповідає сенсу понять "кількість" і "вага".

Зауваження 1. Поняття домену допомагає правильно моделювати предметну область.

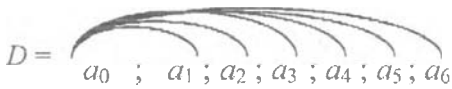
Зауваження 2. Не всі домени володіють логічною умовою, що обмежує можливі значення домену. У такому разі безліч можливих значень домену збігається з множиною можливих значень типу даних.

Зауваження 3. Не завжди очевидно, як задати логічну умову, що обмежує можливі значення домену.

Реалізувати опис домену комп'ютерної системи можна через набір атомарних властивостей домену, поданих як атрибути домену: назва, тип даних, розмір, умова, ключ, не нуль, автоінкремент.

Кожний елемент $d_i \in D$ відповідає атрибутам $a_0, a_1, \ldots, a_6$.

Засобами алгебри алгоритмів атрибути (властивості) домену D описуються як секвенція виду:



де: a_0 – атрибут назва, a_1 – атрибут тип даних, a_2 – атрибут розмір (байт), a_3 – атрибут умова, a_4 – атрибут ключ, a_5 – атрибут не нуль, a_6 – атрибут автоінкремент. Атрибути описуються так:

$a_0 \in \overline{\overline{Q_1, Q_2}, Q_3 ; \_; \$}$

$a_1 \in \begin{cases} \overline{\overline{T(INTEGER), T(FLOAT)}; T(DOUBLE)} \\ ; \\ \overline{\overline{T(CHAR)}; T(TEXT)}; T(BOOL) \\ ; \\ \overline{T(DATE); T(TIME)} \end{cases}$

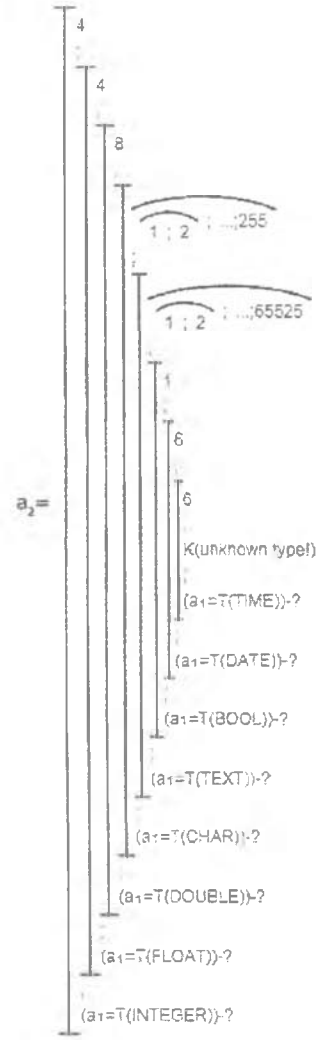
$a_3 = (u_k - ?)$

$a_4 \in \overline{0; 1; 2}$

$a_5 \in \overline{0; 1}$

$a_6 \in \overline{0, 1}$

де, a_0 може складатись із сєквенцій символів латинського та українського алфавіту, символів _ та $; a_1 набуває значення із сєквентної області типів даних означених в системі; a_2 визначається залежно від атрибуту a_1 (тип даних); a_3 є умова u_k – логічний вираз, який є предикатом від змінної d_i і набуває значення ІСТИНА (TRUE) на всьому домені D; a_4 набуває таких значень: 0 – не ключове, 1 – первинний ключ, 2 – вторинний ключ; a_5 може набувати значень 0 – або NULL-значення, 1 – не може приймати NULL-значення; a_6 (0 – значення елементу вільне; 1 – для кожного елементу виконується залежність $d_i=d_{i-1}+1$).



Тепер розглянемо точне визначення терміну ***кортеж***. Якщо дана колекція доменів D_i ($i = 1, 2, \ldots, n$), які не обов'язково всі мають бути різними, то значенням кортежу (або коротко *кортежем),* визначеним за допомогою цих доменів (позначимо його t – від англ. «tuple»), є множина впорядкованих трійок у формі $<A_i, D_i, v_i>$, де A_i — назва атрибуту, D_i — назва домну та v_i — значення домену D_i.

Трійки $<A_i, D_i, v_i>$ в алгебрі алгоритмів описуються секвенціями виду

$$\overbrace{\overbrace{A_i ; D_i} ; v_i}$$

Тоді кортеж t є секвенція виду:

$$t = \left(\left(\left(A_1 ; D_1 \right) ; v_1 \right) ; \left(\left(A_2 ; D_2 \right) ; v_2 \right) ; \left(\left(A_3 ; D_3 \right) ; v_3 \right) ; \ldots ; \left(\left(A_n ; D_n \right) ; v_n \right) \right)$$

Крім того, кортеж t повинен відповідати приведеним нижче умовам.

1. Значення n є ступенем, або арністю кортежу t.

2. Секвентна трійка $\overbrace{\overbrace{A_i , D_i} , v_i}$ є компонентом кортежу t.

3. Секвентна пара унітермів $\overbrace{A_i ; D_i}$ є атрибутом кортежу t і однозначно визначається назвою атрибуту A_i (назви атрибутів A_i і A_j збігаються, тільки якщо $i=j$).

4. Унітерм v_i – це значення атрибуту, відповідне атрибуту A_i кортежу t.

5. Домен D_i — це відповідний домен атрибуту.

6. Повна множина атрибутів складає заголовок t.

7. Тип кортежу t визначений заголовком t, а сам заголовок і тип кортежу мають такі ж атрибути (тому такі ж назви і типи атрибутів) й такий же ступінь, як і t.

Властивості кортежів:

1.Кожен кортеж містить точно одне значення (відповідного типу) для кожного із своїх атрибутів.

2. Для компонентів кортежу впорядкування, не є суттєвим.

3. Кожною підмножиною кортежу є кортеж (а кожна підмножина заголовка є заголовком). Більш того, ці властивості є істинними якщо застосовані, до порожніх підмножин.

Кортеж першого ступеня називається одноелементним, кортеж другого ступеня — двоелементним, кортеж третього ступеня — трьохелементним (і так далі). Кортеж n ступеня називають n-елементним. Кортеж нульового ступеня (тобто кортеж без компонентів) називають нульелементним або нуль арним.

Фундамент реляційної моделі даних складає поняття ***відношення***.

Нехай є n ($n > 0$) не обов'язково різних доменів $D_1, D_2 ..., D_n$.

Декартовий добуток $\times\{Di: i = 1, 2 ..., n\}$ – це n - кортежів $<t_1, t_2 ..., t_n>$ таких, що $t_i \in D_i$ для всіх i. Відношення R визначається на цих n доменах, якщо воно є підмножиною їх декартового добутку. Говорять, що таке відношення має ступінь n.

Визначення 1. Атрибут відношення є пара виду <Назва_атрибута: Назва_домена>. Назви атрибутів мають бути унікальні в межах відношення. Часто назви атрибутів відношення збігаються з назвами відповідних доменів.

Визначення 2. Відношення R, визначене на n доменах $D_1, D_2,\ldots,D_n$ (не обов'язково різних), містить дві частини: заголовок та тіло. Заголовок відношення містить фіксовану кількість атрибутів відношення:

$$\overbrace{A_1 : D_1 ; A_2 : D_2 ; \ldots ; A_n : D_n}$$

Тіло відношення містить множину кортежів відношення. Кожним кортежем відношення є множина пар виду <Назва_атрибута: Значення_атрібута>:

$$\overbrace{A_1 : v_1 ; A_2 : v_2 ; \ldots ; A_n : v_n}$$

таких що значення V_i атрибуту A_i належить домену D_i.

Замість множини індексів (1, 2,..., n) ми можемо використовувати будь-яку невпорядковану множину, якщо ми асоціюємо з кожним компонентом кортежу не тільки його домен, але також й індекс, що відрізняє його, який ми надалі називатимемо атрибутом кортежу. Відповідно, n різних атрибутів відношення ступеня n характеризують n різних застосувань доменів, на яких визначено дане відношення [5]. Відношення записується у вигляді:

$$R=\overbrace{A_1 : D_1 ; A_2 : D_2 ; \ldots ; A_n : D_n}$$

або коротше

$$R=\overbrace{D_1 ; D_2 ; \ldots ; D_n}$$

або просто R.

Число атрибутів у відношенні називають ступенем (або арністю) відношення. Потужність множини кортежів відношення називають потужністю відношення.

Таким чином, відношення складається з множини кортежів, і кожен кортеж має одну і ту ж множину атрибутів. Якщо всі домени є простими, то таке відношення має **табличне** зображення з такими властивостями:

1. Не існує дублікатів рядків (кортежів).
2. Порядок рядків є несуттєвим.
3. Порядок стовпців (атрибутів) є несуттєвим.
4. Всі елементи таблиці є атомарними значеннями.

Реляційна база даних – це сукупність даних, що змінюється в часі, допускає доступ до них всіх і їх оновлення таким чином, ніби вони були організовані у вигляді сукупності табличних (не ієрархічних) відношень відповідних ступенів, визначених на заданому наборі простих доменів, що змінюються в часі.

Базові відношення (base relation) – це такі відношення, які визначаються незалежно від інших відношень в базі даних у тому сенсі, що ніяке базове відношення повністю не виводиться (незалежно від часу) з яких-небудь інших базових відношень.

Вивідні відношення (derived relation), – це такі відношення, які можуть повністю виводитися з базових відношень. Це такий вид відношень, які зазвичай служать для забезпечення користувачів прикладних програм їх власними зображеннями (view) бази даних. Оголошені відношення включають всі базові відношення, а також можуть включати відношення, що виводяться.

Бази даних мають таку термінологію, за якої одні і ті самі об'єкти мають декілька назв. [5, с. 121] Прийнято розрізняти три термінології: реляційна (математична модель), таблична (логічне зображення) та файлова (фізична реалізація). Порівняння термінологій наведено в таблиці 1.

Таблиця 1. Порівняння термінологій баз даних.

Реляційний термін	**Табличний термін**	**Файлова термінологія**
База даних	Набір таблиць	Каталог
Схема бази даних	Набір заголовків таблиць	Структура каталогу
Відношення	Таблиця	Файл
Заголовок відношення	Заголовок таблиці	Метадані файлу
Тіло відношення	Тіло таблиці	
Атрибут відношення	Найменування стовпця таблиці	Поле
Кортеж відношення	Рядок таблиці	Запис
Ступінь (арність) відношення	Кількість стовпців таблиці	
Потужність відношення	Кількість рядків таблиці	
Домени і типи даних	Типи даних у комірках табл.	

Між табличними відношеннями не існує яких-небудь структурних зв'язків, таких як покажчики. Асоціації між відношеннями подаються виключно за допомогою значень.

З кожним відношенням асоціюється безліч можливих **ключів**. Сукупність атрибутів K відношення R називається можливим ключем (candidate key) R, якщо він володіє наступними незалежними від часу властивостями.

1. Ніякі два рядки R не містять один і той же K-компонент.
2. Якщо який-нсбудь атрибут виключається з K, то властивість унікальності втрачається.

Для кожного базового відношення один з можливих ключів вибирається у якості первинного ключа (primary key). Для заданої бази даних ті домени, на яких визначаються прості (тобто що складаються з одного атрибуту) первинні ключі, називаються первинними доменами (primary domain) цієї бази даних. Не всі атрибути компоненту складеного (тобто що складається з декількох атрибу-

тів) первинного ключа обов'язково мають бути визначені на первинних доменах. Первинні домени мають важливе значення для підтримки деяких транзакцій.

Всі операції вставки, оновлення і видалення, що виконуються над базовими відношеннями, обмежуються двома наступними правилами:

Правило 1 (цілісність суті): Не допускаються ситуації, коли первинний ключ якого-небудь базового відношення має невизначене значення (null) або містить хоч би один компонент з невизначеним значенням.

Правило 2 (цілісність по посиланнях): Допустимо, що деякий атрибут *A* складеного (тобто що складається з декількох атрибутів) первинного ключа відношення *R* визначений на первинному домені *D*. Тоді у будь-який момент часу для кожного значення *v* атрибуту *A* відносно *R* має існувати базове відношення (скажімо, *S*) з простим первинним ключем (наприклад, *B*) таке, що *v* є значенням *B* в *S*.

Висновок.

У роботі дані означення, властивості та опис засобами алгебри алгоритмів понять домену, кортежу, відношення та ключів, які складають основу структурної частини реляційної моделі.

1. Овсяк В.К. Засоби еквівалентних перетворень алгоритмів інформаційно-технологічних систем. / Овсяк В.К.//Доповіді Національної академії наук України, 1996,№9. -С. 83-89.

2. Дейт К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом "Вильямс", 2005. — 1328 с.

3. Кодд Э. Ф. Реляционная модель данных для больших совместно используемых банков данных/Э. Ф. Кодджурнал Системы Управления Базами Данных № 1/1995. — Переклад російською з оригіналу //E.F. Codd. A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, Volume 13, Number 6, June, 1970.

4. Кодд Э. Ф. Расширение реляционной модели для лучшего отражения семантики. /Э. Ф. Кодд //Системы управления базами данных 5/1996.— Переклад російською з оригіналу E.F. Codd. Extending the Database Relational Model to Capture More Meaning. //ACM

5. Коннолли Т., Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е издание. : Пер. с англ. — М. : Издательский дом "Вильяме", 2003. — 1440 с.

6. Owsiak W., Teoria algorytmov abstrakcyjnych i modeowanie atematyczne systemow informacyjnych./ Qwsiak A., Owsiak J.// −Opole: Studia i monografie. z. 176. "Politechnika opolska", 2005. − 275 s. Transactions on Database Systems, Vol. 4,№ 4, December 1979.

7. Ovsyak V.K. Computation models and algebra of algorithms /Ovsyak V.K.// Інформаційні системи та мережі. Вісник Національного університету «Львівська політехніка». 2008. − №621. − с. 3-18.