Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Мохаммед Карам Джасім Мохаммед

УДК 004.655

РОЗШИРЕННЯ ТАБЛИЧНИХ АЛГЕБР МНОЖИННИМ УСПАДКУВАННЯМ

Спеціальність 01.05.03 – математичне та програмне забезпечення обчислювальних машин і систем

АВТОРЕФЕРАТ
дисертації на здобуття наукового ступеня
кандидата фізико-математичних наук

Київ-2017

Дисертацією є рукопис.

Робота виконана на кафедрі теорії та технології програмування факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка, МОН України.

Науковий керівник – доктор фізико-математичних наук, професор
Буй Дмитро Борисович,
Київський національний університет імені Тараса Шевченка,
професор кафедри теорії та технології програмування.

Офіційні опоненти: доктор фізико-математичних наук,
старший науковий співробітник
Стецюк Петро Іванович,
Інститут кібернетики імені В. М. Глушкова НАН України,
завідувач відділу методів негладкої оптимізації.

кандидат фізико-математичних наук, доцент
Глибовець Андрій Миколайович,
Національний університет "Києво-Могилянська академія",
доцент кафедри мережних технологій.

Захист відбудеться "21" вересня 2017 р. о 14.00 годині на засіданні спеціалізованої вченої ради Д 26.001.09 Київського національного університету імені Тараса Шевченка за адресою: 03680, м. Київ, проспект академіка Глушкова, 4д, ауд. 40.

З дисертацією можна ознайомитись у Науковій бібліотеці ім.М.Максимовича Київського національного університету імені Тараса Шевченка за адресою: 01601, м. Київ, вул. Володимирська, 58, зал № 12.

Автореферат розісланий "21" серпня2017 р.

Учений секретар
спеціалізованої вченої ради

В.П. Шевченко

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність теми дослідження. Перші реляційні СУБД були розроблені ще в 70-ті роки минулого століття. На початку 90-х років, до реляційних баз даних були застосовані ідеї об'єктно-орієнтованого підходу, що привело до появи об'єктно-реляційних систем управління базами даних (ОРБД), які поєднують функціональність реляційних СУБД з концепціями та ідеями взятими з об'єктно-орієнтованого програмування. В стандартах SQL:1999 - SQL:2016 в реляційну модель даних були додані численні об'єктно-орієнтовані риси. Це значно змінило модель даних, що лежить в основі останніх версій СУБД, базованих на цих стандартах. На сьогодні, незважаючи на появу інших моделей даних, реляційні та об'єктно-реляційні бази даних залишаються основою інформаційних систем підприємств та організацій.

Семантиці SQL присвячені численні теоретичні роботи, починаючи, наприклад, з роботи в якій була побудована формальна модель запитів, названа EAVPC[1], та закінчуючи публікацією в 2017 році денотаційної семантики SQL, названої HoTTSQL[2]. Потужність реляційних СУБД значною мірою пояснюється тим, що вони базуються на строгих математичних основах, викладених ще Кодом[3]. В той же час, всі ці роботи не приділяють уваги об'єктно-орієнтованим властивостям СУБД, зосередившись, на структурах даних та операціях над ними.

На кафедрі теорії та технології програмування Київського національного університету імені Тараса Шевченка, ще з минулого тисячоліття розвивається побудова семантики SQL на основі спеціальної програмної алгебри, названої табличною алгеброю семантичних функцій, яка має ряд переваг над іншими підходами. А саме, в табличній алгебрі виділяється два типи функцій – функції на даних, зокрема таблицях, та функції на функціях, які називаються операторами, як це прийнято в інших мовах програмування. Семантика складних виразів SQL в цій алгебрі природнім чином будується з функцій на даних і операторів.

Слід відмітити, що такий підхід до визначення семантики мов програмування був запропонований в роботах по системам алгоритмічних алгебр (САА) В.М. Глушкова, Г.Е. Цейтліна, Е.Л. Ющенко, а згодом в роботах

---

[1] M. Negri, G. Pelagatti, L. Sbattella: Formal semantics of SQL queries. ACM Trans. Database Syst. 16(3), 513-534 (1991)

[2] Shumo Chu, Konstantin Weitz, Alvin Cheung, Dan Suciu: HoTTSQL: proving query rewrites with univalent SQL semantics. Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation, P. 510-524 (Barcelona, Spain — June 18 - 23, 2017)

[3] E. F. Codd: A relational model of data for large shared data banks. Communications of the ACM Volume 13 Issue 6, June 1970. P. 377-387

по композиційній семантиці В.Н. Редька, Д.Б. Буя та М.С. Нікітченка. Ці праці стали теоретичним підґрунтям Київської школи програмування (http://www.computer-museum.ru/galglory/27.htm).

Потреба у розвитку теорії до рівня сучасних ОРБД, зокрема семантики запитів мови SQL, та семантики об'єктних розширень реляційної моделі даних, і визначає актуальність даної дисертаційної роботи в який розширюється таблична модель даних за рахунок множинного успадкування таблиць, та удосконалена таблична алгебра семантичних функцій SQL, зокрема перевизначені оператори внутрішнього та зовнішнього з'єднання.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота є складовою частиною наукових робіт, проведених на кафедрі теорії та технології програмування факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка при виконанні фундаментальної теми "Формальні специфікації та методи розробки надійних програмних систем" (№ 0111U007052, 2011-2015 рр.) і теми «Теорія і методи розробки інтелектуальних інформаційних технологій та систем» (№16КФ015-02).

Мета і задачі дисертаційного дослідження. *Метою* дисертаційної роботи є аналіз алгебри семантичних функцій ядра мови SQL, яка називається табличною алгеброю, на відповідність реалізаціям SQL в поширених СУБД, зокрема операцій з'єднання та теоретико-множинних операцій; знаходження розбіжностей, тобто випадків, коли результати обчислень семантичних функцій відрізняються від результатів, які повертають відповідні запити реалізовані в мові SQL; модифікація алгебри семантичних функцій для уникнення таких розбіжностей; розширення реляційної моделі даних конструкцією одиночного і множинного успадкування таблиць.

З огляду на мету в роботі ставляться такі *задачі*:

- провести перевірку табличної алгебри ядра мови SQL. Перевірку провести на версії SQL, втіленої в об'єктно-реляційної СУБД PostgreSQL, яка є однією з найбільш точних та повних реалізацій стандарту SQL. При цьому особу увагу приділити обчисленню запитів на таблицях, які мають співпадаючи імена стовбців, або які є пустими;
- перевизначити семантику операторів внутрішнього та зовнішнього з'єднання реляцій таким чином, щоб результати з'єднання для реляцій, імена атрибутів яких частково або повністю співпадають, повертали такі самі реляції як і в мові SQL, по можливості, усунути також інші розбіжності
- перевірити семантичні функції, які задають семантику фрази ORDER BY. Дослідити властивості семантичних функцій, які відповідають за впорядкування рядків таблиць

- дослідити алгоритми автоматичного вирішення конфліктів імен атрибутів при множинному успадкуванні, та застосувати ці алгоритми до успадкування таблиць
- дослідити властивості методу лінеаризації, який використовується в алгоритмах множинного наслідування

Об'єктом дослідження є реляційні бази даних та об'єктно-реляційні бази даних.

Предметом дослідження є реляційна модель даних.

Наукова новизна одержаних результатів. Проведене порівняння результатів запитів ядра мови SQL та результатів обчислень відповідних їм семантичних функцій табличної алгебри.

Виявлені розбіжності в результатах виконання запитів, які містять операції зовнішнього та внутрішнього з'єднання та відповідним їм семантичним функціям на таблицях, які мають спільні імена атрибутів. Також виявлені розбіжності при виконанні теоретико-множинних операцій на таблицях, пов'язані з тим, що в SQL операції об'єднання, перетину та різниці таблиць враховують порядок стовбців заданий в запитах, а в семантичній алгебрі враховуються імена атрибутів стовбців, а порядок не є важливим.

- Запропоновані нові визначення для операцій внутрішнього та зовнішнього з'єднання та теоретико множинних операцій, визначена нова форма оператору суперпозиції, більш зручна для задання семантики складних запитів. В результаті, поновлена алгебра семантичних функцій більш точно описує семантику ядра мови SQL та є більш зручною в використанні.
- Проведена перевірка семантики фрази ORDER BY мови SQL. Виявлено, що семантичні функції задають частковий порядок на кортежах, в той час як фраза SQL ORDER BY завжди повертає лінійний порядок на рядках таблиці. Сформульована та доведена теорема, яка визначає умови, при яких семантичні функції задають точну семантику ORDER BY.
- Введено поняття ієрархії успадкування, яке описує як одиночне так і множинне успадкування таблиць. Сформульовані та доведені теореми про характеристичні ознаки одиночного успадкування. Надані формальні математичні визначення для таких властивостей методу лінеаризації ієрархії таблиць, як монотонність та збереження локального порядку успадкування.
- Що стосується практичної частини, то розроблені алгоритми перевірки, чи є ієрархія успадкування одиночним успадкуванням чи множинним, та чи є ієрархія одиночного успадкування простою.
- Надане формальне математичне визначення алгоритму MRO C3 для автоматичного вирішення конфлікту співпадаючих імен атрибутів

при множинному успадкуванні таблиць. Доведені теореми про монотонність алгоритму та його завершуваність.

Теоретичне і практичне значення одержаних результатів. Дисертація має теоретико-прикладну спрямованість. Результати роботи можуть використовуватись для задання семантики запитів мови SQL, для оптимізації запитів, досліджені властивостей мови SQL, доведенні коректності запитів, для еквівалентного перетворенні запитів та доведення еквівалентності запитів, проектуванні баз даних.

Результати роботи були впроваджені у навчальний процес за спеціальністю "Інформатика" на факультеті кібернетики КНУ (нормативні курси "Прикладна логіка", "Композиційна семантика SQL-подібних мов"; спеціальний курс "Вступ до реляційних баз даних"), а також у Кіровоградському державному педагогічному університеті імені Володимира Винниченка (нормативний курс "Табличні алгебри та SQL подібні-мови").

Особистий внесок здобувача. Всі результати, які складають суть дисертаційної роботи, отримані здобувачем самостійно. З праць, виконаних зі співавторами, на захист виносяться лише результати, отримані особисто здобувачем. У спільно виконаних роботах науковому курівнику Д.Б. Бую належить загальна постановка проблеми, обговорення та інтерпретація результатів.

Апробація результатів дослідження. Основні положення та висновки дисертаційного дослідження обговорювалися на наукових семінарах кафедри теорії та технології програмування КНУ, республіканському семінарі "Програмологія та її застосування".

Результати дисертаційного дослідження оприлюднені у доповідях і повідомленнях на Міжнародних та Всеукраїнських наукових конференціях, семінарах:

- XI Міжнародна науково-практична конференція: ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ. – Київ, 15-17 грудня 2014 р.
- I Международный научно-практический форум «Наука и бизнес»: – Черновцы, 2015.
- 13th International Scientific Conference on Informatics. – Poprad, Slovakia, 18-20 Nov., 2015.
- XII Міжнародна науково-практична конференція ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ. – Київ, 23-26 грудня 2015 року.
- XIII Міжнародна науково-практична конференція ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ. – Київ, 5-9 грудня 2016 року.

- XII International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH). – 2016.
- Міжнародний науково-практичний семінар – "Комбінаторні конфігурації та їх застосування" – 2016.
- Четвёртая конференция «КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В НАУКОЕМКИХ ТЕХНОЛОГИЯХ» (КМНТ-2016). – Харьков, 26-31 мая 2016 г

Публікації. Результати дисертації опубліковані у 7 статтях [1, 2, 3, 4, 5, 6, 7] в наукових журналах і збірниках наукових праць, 8 тезах конференцій [8, 9, 10, 11, 12, 13, 14, 15]; з них 5 статей опубліковані у наукових фахових виданнях, , одна стаття опублікована в міжнародному журналі.

Структура та обсяг дисертації. Дисертаційна робота з анотації, переліку умовних позначень, вступу, чотирьох розділів, висновків, списку використаних джерел. Загальний обсяг дисертації становить 160 с, основний текст 110 с., список використаних джерел 121 найменування на 15 сторінках.

ОСНОВНИЙ ЗМІСТ РОБОТИ

У **вступі** обґрунтовується актуальність теми дисертації, визначається об’єкт, мета та завдання дисертаційної роботи, методи дослідження, наукова новизна одержаних результатів та їх теоретико-практичне значення, відображаються апробації та публікації результатів дисертаційного дослідження.

У **першому розділі** «Змістовна семантика ядра мови SQL» надається огляд базових операторів мові SQL, формальна семантика яких буде описана в наступних розділах. Описуються оператори створення та успадкування таблиць, операції фільтрації на основі предикатів трьохзначної логіки, зовнішнього та внутрішнього з’єднання, агрегатні функції, задається оператор групування. Описуються скалярні та табличні підзапити, та корельовані підзапити.

Другий розділ «Уточнена алгебра семантичних функцій запитів ядра мови SQL» присвячений побудові алгебри функцій визначених на таблицях, які є семантичним ядром запитів мови SQL. В ній надано нове визначення операторів внутрішнього та зовнішнього з’єднання, теоретико-множинних операцій, оператора суперпозиції та розглянуті їх властивості.

Підрозділ 2.1 «Математичні основи сучасних реляційних СУБД» присвячений деяким ключовим аспектам математичних підстав сучасних реляційних СУБД: мультимножини, теоретико-множинні основи (конструкції повного образу, обмеження відношень, узагальненого декартова добутку, бінарне відношення сумісності, бінарне відношення конфінальності), некласичні логіки (сильна тризначна логіка Кліні), предпорядки і порядки.

В підрозділі 2.2 «Визначення основних структур даних та операцій на них» надаються визначення рядків, таблиць та їх схем, базових операцій на рядках, таблицях та схемах.

Нехай задані множина імен атрибутів Atr та універсальний домен Dom, який містить множину атомарних даних. Накладемо вимоги, що ці множини є непустими і в множині атомарних даних існує спеціальний елемент $null$, який трактується як невизначене значення. *Схемою* будемо називати будь яку скінченну підмножину імен атрибутів $S \subseteq Atr$. Множину всіх схем будемо позначати через Sch

Рядком, який задовольняє схемі S , або просто рядком схеми S будемо називати скінченну множину пар $\{(a_i, d_i) | a_i \in S,\ d_i \in Dom, i = 1..n\}$ таку, що $a_i \neq a_j$, якщо $i \neq j$, де $i, j = 1..n$. Позначимо операцію проекції рядку r по його першій компоненті через $pr_1(r)$, а схему рядку r будемо записувати як $S(r)$. Рядки можна також, більш компактно, визначити як скінченні відображення з множини імен атрибутів в універсальний домен.

Під *таблицею* схеми S будемо розуміти пару (t, S), де t є скінченою множиною рядків схеми S, яка називається *станом таблиці або просто станом*. В більшості випадків схема таблиці S може бути отримана зі схем рядків, за виключенням випадку, коли стан є пустою множиною. Множину всіх рядків (таблиць) схеми S будемо позначати як $Rec(S)$ (відповідно $Tbl(S)$), а множину всіх рядків (таблиць) – Rec (відповідно Tbl). Так $Rec(S) \stackrel{\text{def}}{=} \{r | pr_1(r) = S\}$, $Tbl(S) \stackrel{\text{def}}{=} \{(t, S) | t \in 2^{Rec(S)}\}$, $Rec \stackrel{\text{def}}{=} \bigcup_{S \subseteq Atr} Rec(S)$, $Tbl \stackrel{\text{def}}{=} \bigcup_{S \subseteq Atr} Tbl(S)$.

Запис вигляду $r(a)$, $r \in Rec, a \in Atr$ означає значення атрибуту з іменем a на рядку r. Домовимося схему рядку r позначати $S(r)$

Схема таблиці може бути пустою $\emptyset$. Для такої схеми існує рівно один рядок який позначимо як ε, і два стана – $\{\varepsilon\}$ та $\emptyset$. Домовимося позначати рядки через $r, r_1, r_2, \ldots$, таблиці – $T, T_1, T_2, \ldots$, стани таблиць через $t, t_1, t_2, \ldots$, а схеми через $S, S_1, S_2, \ldots$

Бінарне відношення *сумісності рядків* $r_1 \approx r_2$ визначається за формулою $\forall a(a \in S(r_1) \& a \in S(r_2) \Longrightarrow r_1(a) = r_2(a))$. Іншими словами два рядки є сумісними якщо значення атрибутів з співпадаючими іменами теж співпадають.

Рядки, схеми яких не перетинаються завжди є сумісними. Бінарна операція об'єднання сумісних рядків $\cup: Rec \times Rec \longrightarrow Rec$ визначається за формулою $r_1 \cup r_2 = \{(a, d) | (a, d) \in r_1 \vee (a, d) \in r_2\}$. Її областю визначення буде множина $dom \cup \stackrel{\text{def}}{=} \{(r_1, r_2) | r_1, r_2 \in Rec \,\&\, r_1 \approx r_2\}$

Разом з простими іменами атрибутів будуть використовуватися складені імена вигляду $a_1.a_2 \ldots a_n$, $a_i \in Atr, i = 1..n$. Множину всіх складених атрибутів позначимо Atr^+. Такі імена будуть використовуватися для того, щоб не порушувати вимогу унікальності імен атрибутів при об'єднані рядків. Відмітимо, що $Atr \subseteq Atr^+$. Далі визначимо спеціальну функцію

$a \Rightarrow: Rec \longrightarrow Rec$ параметризовану по імені атрибута a. Вона визначається за формулою $a \Rightarrow (r) = \{(a.a_i, d_i)|(a_i, d_i) \in r\}$. Таку функцію назвемо *функцією префіксації*, а ім'я атрибуту a будемо називати *префіксом*.

Протилежна функція, яка вилучає першу компоненту складених атрибутів, називається функцією *депрефіксації* і визначається за формулою $\Leftarrow (r) = \{(b, d)|(a.b, d) \in r \,\&\, a \in Atr \,\&\, b \in Atr^+\}$. Вона визначена на множені рядків, імена атрибутів яких складаються, принаймні з двох компонент, причому, якщо імена деяких атрибутів співпадають з точністю до перших компонент, то їх значення повинні теж співпадати. В протилежному випадку функція буде невизначеною. Тобто депрефіксація є частковою функцією на відміну від префіксації. Має місце наступна рівність: $r = \Leftarrow (a \Rightarrow (r))$

Операція *об'єднання рядків з префіксацією* $\cup_{a_1,a_2}: Rec \times Rec \longrightarrow Rec$, $a_1 \neq a_2$ є операцією параметризованою по іменам префіксів a_1 і a_2. Вона визначається наступним чином: $r_1 \cup_{a_1,a_2} r_2 = \{r_1' \cup r_2' | r_1' = (a_1 \Rightarrow r_1) \& r_2' = (a_2 \Rightarrow r_2)\}$

Позначимо через e пусте ім'я. Тоді складене ім'я $e.a$ будемо вважати співпадаючим з іменем a. Це означає, що, наприклад, $r_1 \cup_{e,a_2} r_2 = \{r_1' \cup r_2' \,|\, r_1' = r_1 \& r_2' = (a_2 \Rightarrow r_2)\} = \{r_1 \cup r_2' \,|\, r_2' = (a_2 \Rightarrow r_2)\}$.

Якщо в якості області визначеності розглядати множини рядків, імена атрибутів яких є простими, то операція буде всюди визначеною.

Аналогічним чином визначимо префіксацію для схем $S_1 \cup_{a_1,a_2} S_2 = a_1 \Rightarrow (S_1) \cup a_2 \Rightarrow (S_2)$, де $a \Rightarrow (S) \stackrel{\text{def}}{=} \{a.a_i | a_i \in S\}$

Будемо говорити, що рядок належить таблиці, якщо він належить стану цієї таблиці: $r \in T \stackrel{\text{def}}{=} \exists t \exists S(T = (t, S) \wedge r \in t)$.

В підрозділі 2.3 «Теоретико-множинні операції на таблицях» задані нові версії теоретико-множинних операцій на реляціях. На відміну від попередніх визначень, відповідність стовбців двох таблиць визначається за позицією, а не за ім'ям, таким самим чином, як це робиться в SQL.

Будемо говорити, що таблиці T_1 і T_2 є *однотипними* якщо потужності їхніх схем співпадають. Позначимо таке відношення через $T_1 \cong T_2$. Відношення *однотипності* рядків визначається аналогічним чином: $r_1 \cong r_2 \stackrel{\text{def}}{=} |S(r_1)| = |S(r_2)|$

Позначимо через $REN_B^A(t)$ унарну функцію перейменування атрибутів. Тут A і B – послідовності імен атрибутів однакової довжини. Функція заміняє в рядку r всі імена атрибутів з послідовності A на відповідні імена з послідовності B. Функція є частковою, так як після перейменування деякі атрибути можуть отримати однакові імена. Для таблиць ця функція буде перейменовувати кожний рядок, тобто $REN_B^A((t,S)) = (t', S')$, де $t' = \{r' | r' = REN_B^A(r) \& r \in t\}$, $S' = (S \backslash A) \cup B$

Позначимо через T_1 таблицю (t_1, S_1), а через T_2 іншу таблицю (t_2, S_2), однотипну першій, тобто $|S_1| = |S_2|$. Зафіксуємо послідовності, які складаються з імен атрибутів схеми S_1 та S_2, та позначимо їх через $< S_1 >$ та $< S_2 >$ відповідно.

Тоді теоретико-множинні операції задаються наступним чином:

Об'єднання таблиць $T_1 \Cup_{S_2}^{S_1} T_2 = (t_1 \cup \mathrm{REN}_{S_2}^{S_1}(t_2), S_1)$. Областю визначеня операції $\Cup_{S_2}^{S_1}$ буде множина пар однотипних таблиць, схема першої з яких співпадає з S_1, а схема другої співпадає з S_2: $\mathrm{dom}\, T_1 \Cup_{S_2}^{S_1} T_2 = \{(T_1, T_2) | T_1 \cong T_2 \,\&\, S(T_1) = S_1 \& S(T_2) = S_2\}$.

Домовимося операції вигляду $\Cup_{S_1}^{S_1}$ записувати як $\Cup$. Така операція діє як обмеження звичайної теоретико-множинної операції об'єднання на випадок, коли схеми таблиць співпадають.

Перетин таблиць: $T_1 \Cap_{S_2}^{S_1} T_2 = (t_1 \cap \mathrm{REN}_{S_2}^{S_1}(t_2), S_1)$. Область визначення задається аналогічно, як і для об'єднання.

Різниця таблиць: $T_1 \setminus\setminus_{S_2}^{S_1} T_2 = (t_1 \setminus \mathrm{REN}_{S_2}^{S_1}(t_2), S_1)$. Область визначення задається аналогічно, як і для об'єднання.

Як випливає з означення, на відміну від звичайних теоретико-множинних операцій, ці операції не є симетричними.

В підрозділі 2.4 «Операції з'єднання таблиць» визначаються операції декартова з'єднання, природнього з'єднання, та з'єднання за атрибутами.

Операція *декартова з'єднання* $\times_{a_1,a_2}$ є бінарною функцією типу $\times_{a_1,a_2}: Tbl \times Tbl \longrightarrow Tbl$ яка параметризована по префіксах атрибутів a_1, a_2, де $a_1 \neq a_2$.

Позначимо через T_1 таблицю (t_1, S_1) та через T_2 таблицю (t_2, S_2). Тоді декартове з'єднання визначається по формулі $\times_{a_1,a_2}(T_1, T_2) = (t_3, S_1 \cup_{a_1,a_2} S_2)$, де $t_3 = \{r_1 \cup_{a_1,a_2} r_2 | r_1 \in t_1 \& r_2 \in t_2\}$

Операція *природнього з'єднання або еквіз'єднання* $\otimes$ є всюди визначеною бінарною функцією типу $\otimes: Tbl \times Tbl \to Tbl$. Позначимо через T_1 таблицю (t_1, S_1) і через T_2 таблицю (t_2, S_2). Тоді природне з'єднання визначається по формулі $\otimes(T_1, T_2) = (t_3, S_1 \cup S_2)$, де $t_3 = \{r_1 \,\bar{\cup}\, r_2 | r_1 \in t_1 \& r_2 \in t_2 \& r_1 \approx r_2\}$,

Наступною операцією є *з'єднання за атрибутами* $\otimes_{a_1,a_2}^{B}$ яке має тип $Tbl \times Tbl \to Tbl$, де B є множиною імен атрибутів, a_1, a_2 є префіксами імен атрибутів, такими що $a_1 \neq a_2$. З'єднання за атрибутами є бінарною функцією з областю визначення $dom\ \otimes_{a_1,a_2}^{B} \stackrel{\text{def}}{=} \{(T_1, T_2) | B \subseteq S(T_1) \cap S(T_2)\}$. Вона визначається за формулою $\otimes_{a_1,a_2}^{B}(T_1, T_2) = (t_3, S_1 \cup_{a_1,a_2} S_2)$, де $t_3 = \{r_1 \cup_{a_1,a_2} r_2 | r_1 \in t_1 \& r_2 \in t_2 \& r_1(B) = r_2(B)\}$

В підрозділі 2.5 «Визначення операторів на таблицях» визначаються оператори (функціонали), які будують запити. Задаються чотири оператора. Це оператор фільтрації , оператор відображення, оператори внутрішнього та зовнішнього з'єднання. Характерною рисою цих операторів є те що вони розповсюджують функції на рядках на реляції. Крім того вводиться оператор суперпозиції.

Нехай $p: Rec \to Bool$ буде, взагалі кажучи частковим, предикатом, визначеним на рядках, який використовує тризначну логіку Кліні. Позначимо булевські значення через $true, false, unknown$. Множину всіх предикатів позначимо через P.

Назвемо множину функцій типу $Tbl^n \to Tbl$, $n = 0,1,2 \ldots$ n-арними табличними функціями. Позначимо її як TF^n. Тоді множина всіх табличних функцій TF будується як об'єднання n-арних табличних функцій $TF = \bigcup_{i=0,1,2\ldots} TF^i$

Оператор суперпозиції Λ

Нехай задані послідовність імен $B = (b_1, b_2, \ldots, b_n)$, і послідовність даних $D = (d_1, d_2, \ldots, d_n)$, причому $|B| = |D|$. Тоді операція побудови номінативної множини з послідовності даних та імен $B \twoheadrightarrow D$ визначається за формулою $B \twoheadrightarrow D = \{b_1 : d_1, b_2 : d_2, \ldots, b_n : d_n\}$.

Нехай задані номінативна множина $M = \{b_1 : d_1, b_2 : d_2, \ldots, b_n : d_n\}$, та послідовність імен $B' = (\mathrm{b}_{i_1}, \mathrm{b}_{i_2}, \ldots, \mathrm{b}_{i_k})$, така що всі імена з цієї послідовності належать множині імен з M. Тоді операція побудови впорядкованого зрізу з номінативної множини $\mathrm{M}|B'$ визначається як операція, що будує послідовність $(d_{i_1}, d_{i_2}, \ldots, d_{i_k})$, таку, що $(b_{i_p} : d_{i_p}) \in M, p = 1..k$.

Вираз $(B \twoheadrightarrow D)|B'$ будує зріз послідовності D.

Нехай задані послідовність імен $B = (b_1, b_2, \ldots, b_n)$, послідовності B_1, $B_2, \ldots B_m$, які містять імена з B і послідовність даних $D = (d_1, d_2, \ldots, d_n)$.

Оператор суперпозиції Λ визначається як функціонал типу $TF^n \to TF$, де $n = 0,1,2, \ldots$, по формулі:

$\Lambda(B): g\big(f_1(B_1), f_2(B_2), \ldots, f_m(B_m)\big)(D) = g(f_1(B \twoheadrightarrow D)|B_1, \ldots, f_m(B \twoheadrightarrow D)|B_m)$

, де $B, B_1, B_2, \ldots, B_m$ є параметрами оператору, а $g, f_1, f_2, \ldots, f_m$ – табличні функції.

Оператор фільтрації Φ

Нехай T є таблицею (t, S). Оператор фільтрації визначається як функціонал типу $\Phi: P \to TF^1$ що будує унарну табличну функцію з предиката на рядках за наступною формулою:

$\Phi(p)(T) = (t', S)$, де $t' = \{r | r \in t \& p(t) = true\}$, а схема S співпадає зі схемою таблиці T. Про такі оператори будемо говорити, що вони зберігають схему таблиці.

Оператор відображення M

Функцією на рядках будемо називати часткову функцію f_S типу $Rec \to Rec(S)$, де $Rec(S)$ – множина рядків схеми S, параметризовану по схемі $S \subseteq Atr$, таку що для будь якого вхідного рядку вона повертає рядок схеми S або результат є невизначеним. Позначимо множину функцій на рядках як RF.

Нехай $f_{S'}$ є функцією на рядках і T є таблицею (t, S). Тоді оператор відображення визначається як функціонал типу $RF \to TF^1$ який будує унарну табличну функцію з функції на рядках, за формулою:

$\mathrm{M}(f_{S'})(T) = (t', S')$, де $t' = \{r' | r \in t \& r' = f_{S'}(r)\}$

Оператор внутрішнього з'єднання Θ_{a_1,a_2}.

Оператор внутрішнього з'єднання за предикатом p визначається як оператор Θ_{a_1,a_2}типу $P \to TF$, параметризований по префіксам імен атрибутів a_1, a_2,де $a_1 \neq a_2$. Оператор визначається за формулою $\Theta_{a_1,a_2}(p)(T_1, T_2) = (t_3, S_1 \cup_{a_1,a_2} S_2)$, де $t_3 = \{r | r = (r_1 \cup_{a_1,a_2} r_2) \& r_1 \in t_1 \& r_2 \in t_2 \& p(r) = true\}$

Позначимо через $State(T)$ стан таблиці T. Нехай $dom\ p$ є областю визначення предиката p. Тоді область визначення побудованої функції буде $dom\ \Theta_{a_1,a_2}(p) \stackrel{\text{def}}{=} \{(T_1, T_2) | State(T_1 \times_{a_1,a_2} T_2) \subseteq dom\ p\}$.

Операції декартового з'єднання, природнього з'єднання та з'єднання за атрибутами є похідними від оператора з'єднання та деякого предикату.

Оператор лівостороннього зовнішнього з'єднання $\overleftarrow{\Theta}_{a_1,a_2}$.

Оператор лівостороннього зовнішнього з'єднання за предикатом p визначається як оператор $\overleftarrow{\Theta}_{a_1,a_2}$типу $P \to TF$, параметризований по префіксам імен атрибутів a_1, a_2,де $a_1 \neq a_2$.

Визначимо допоміжний оператор лівостороннього доповнення $\ltimes_{a_1,a_2}(p)(T_1, T_2) = (t_3, S_3)$, де $S_3 = S_1$ і $t_3 = \{r_1 | r_1 \in t_1 \& \forall r_2 (r_2 \in t_2 \Rightarrow p(r_1 \cup_{a_1,a_2} r_2) \neq true)\}$

Введемо операцію побудови всюди невизначеної таблиці зі схемою S яка будується за формулою $Null(S) = (t_{Null}, S)$, і має тип $Sch \to Tbl$. Нехай схема S дорівнює $\{a_1, a_2, \dots, a_n\}$, тоді $t_{Null} = \{\{a_1 : null, a_2 : null, \dots, a_n : null\}\}$, тобто таблиця складається з одного рядку з невизначеними значеннями.

Тоді оператор лівостороннього зовнішнього з'єднання визначається за формулою:

$\overleftarrow{\Theta}_{a_1,a_2}(p)(T_1, T_2) = \Theta_{a_1,a_2}(p)(T_1, T_2) \Cup (\ltimes_{a_1,a_2}(p)(T_1, T_2) \times_{a_1,a_2} Null(S(T_2)))$

Аналогічним чином визначаються оператори правостороннього зовнішнього з'єднання $\overrightarrow{\Theta}_{a_1,a_2}$та повного зовнішнього з'єднання $\overleftrightarrow{\Theta}_{a_1,a_2}$

В підрозділі 2.6 «Відношення конфінальності, передпорядки та порядки» розглядаються допоміжні властивості реляцій, які потім використовуються для розгляду семантики фрази ORDER BY в наступному підрозділі

Підрозділ 2.7 «Семантика фрази ORDER BY запитів SQL-подібних мов»

Семантика фрази ORDER BY запитів SQL-подібних мов була розглянута в монографії Редько В.Н., Буй Д.Б., та ін. «Реляційні бази даних: табличні алгебри та SQL-подібні мови», де було показано, що бінарне відношення, яке уточнює вказану фразу, є в загальному випадку передпорядком, а не порядком на рядках таблиці.

В дисертації наведений критерій того, що відповідне бінарне відношення є не тільки передпорядком, а і порядком

Нехай D - універсальна множина, яка вважається лінійно впорядкованою з порядком $\leq$. Нехай $< A_1, \dots, A_n >$ – кортеж атрибутів схеми S (тобто $A_1, \dots, A_n \in S$). На рядках таблиці вводиться наступне параметризоване за кортежем атрибутів $< A_1, \dots, A_n >$ бінарне відношення

$$s \preccurlyeq s' \overset{def}{\Longleftrightarrow} s(A_1) < s'(A_1) \vee \big(s(A_1) = s'(A_1) \& s(A_2) < s'(A_2)\big) \vee \dots \vee \big(\big(s(A_1) = s'(A_1) \wedge \dots \wedge s(A_{n-1}) = s'(A_{n-1})\big) \& s(A_n) < s'(A_n)\big) \vee \big(\big(s(A_1) = s'(A_1) \& \dots \& s(A_{n-1}) = s'(A_{n-1})\big) \& s(A_n) \leq s'(A_n)\big)$$

.

В наведеному означенні $s(A)$ – значення атрибуту A в рядку s, а $<$ – строга нерівність, тобто $d < d' \overset{def}{\Longleftrightarrow} d \leq d' \& d \neq d'$.

Множина атрибутів X називається ключем таблиці t , якщо виконується наступне твердження $\forall s \forall s' (s, s' \in t \& s|X = s'|X \Rightarrow s = s')$.

В цьому означенні $s|X$ – обмеження рядка s (як функції) на множину атрибутів X.

Теорема 2.2 (критерій передпорядку на рядках $\preccurlyeq$ бути порядком). Параметричне відношення передпорядку на рядках буде порядком тоді і тільки тоді, коли множина атрибутів $\{A_1, \dots, A_n\}$ містить ключ.

Третій розділ «Математичні основи алгоритмів лінеаризації» присвячений математичним основам алгоритмів лінеаризації – одного з методів розв‘язання конфлікту імен, що виникає при множинному успадкуванні в об’єктно-реляційних базах даних, та об‘єктно-орієнтованих мовах програмування. Основним об‘єктом дослідження є рефлексивно-транзитивне замикання бінарного відношення. Встановлені основні властивості цього замикання.

В підрозділі 3.1 «Основні властивості операції лінеаризації» розглянуті основні результати цього підрозділу. А саме: встановлено властивості двох допоміжних операцій (інверсії бінарного відносини і добутки бінарних

відносин), а також рефлексивно-транзитивного замикання бінарного відношення.

Зафіксуємо універсум D, елементи якого позначим $x, y, z, \ldots$; бінарні відношення на D позначим $U, V, \ldots$

В підрозділі 3.2 «Допоміжні операції інверсії і добутку бінарних відносин» розглянуті операції інверсії і добутку та їх властивості.

В підрозділі 3.3 «Допоміжні результати: рефлексивність, антисиметричність і транзитивність» розглянуті властивості відношень рефлексивності, антисиметричності та транзитивності.

В підрозділі 3.4 «Рефлексивно-транзитивне замикання бінарного відношення» розглянуте основне поняття цього розділу – рефлексивно-транзитивному замиканню відношення U, яке має наступне явне визначення:

$U^* = \bigcup_{i=0,1,2,\ldots} U^i$, де $U^0 = \Delta_D$, $U^1 = U$, $U^{i+1} = U \circ U^i$, $i = 1,2,\ldots$ де Δ_D – діагональ на універсумі, а $\circ$ – добудок бінарних відносин

Теорема 3.1 (критерій часткового порядку для рефлексивно-транзитивного замикання). Виконується еквівалентність:

U^* – частковий порядок $\Leftrightarrow \forall i, j = 1,2,\ldots U^i \cap \left(U^{-1}\right)^j \subseteq \Delta_D$. ■

Саме цей результат важливий для алгоритмів лінеаризації, коли по ієрархії класів, що задовольняє по суті умові відсутності циклів у відповідному індуцированому графі на класах, будується частковий порядок на множині класів (вимога $\forall i, j = 1,2,\ldots U^i \cap \left(U^{-1}\right)^j \subseteq \Delta_D$ і говорить по суті про відсутність циклів).

В підрозділі 3.5 «Рефлексивно-транзитивне замикання бінарного відношення: характеристичні денотативні властивості» розглянуті властивості рефлексивно-транзитивного замикання, важливі для внутрішньої проблематики теорії бінарних відносин. Представлені два завдання рефлексивно транзитивного відношення в термінах його властивостей.

В підрозділі 3.6 «Рефлексивно-транзитивне замикання бінарного відношення як найменше рішення характеристичного рівняння» надано третю денотативну характеристику рефлексивно-транзитивного замикання, а саме як найменшого рішення відповідного природного рівняння.

У четвертому розділі «Множинне успадкування таблиць з використанням алгоритмів лінеаризації» розглядається алгоритм розв'язання конфліктів імен при одинарному та множинному успадкуванню таблиць в об'єктно-реляційних базах даних MRO СЗ.

В підрозділі 4.1 «Основні визначення та поняття успадкування таблиць» надаються наступні визначення.

Введемо поняття класу таблиці, під яким будемо розуміти ім'я таблиці та множину атрибутів таблиці, які мають унікальні імена в межах одного класу, хоча ці імена можуть і повторюватися в інших класах. Класи таблиць будемо позначати латинськими буквами, а множину всіх класів таблиць позначимо через $TCls$. Множину всіх атрибутів позначимо через Atr, а окремі атрибути – через $a_1, a_2, \dots$ Тоді клас таблиці, або просто клас, буде визначатися як пара (C, atr), де C – ім'я класу, а atr – множина атрибутів $a_1, a_2, \dots, a_n$. Такий клас таблиці будемо записувати у вигляді $C(a_1, a_2, \dots, a_n)$, або просто C, якщо відомо, про які атрибути йде мова, або конкретний набор атрибутів неважливий.

Між класами існує відношення успадкування, коли один клас успадковує атрибути іншого класу, або, навіть, сукупності класів. Якщо клас C_1 успадковується від класу C_2 то домовимося таку пару записувати у вигляді $C_1 \to C_2$, а, якщо деякий клас C_1 успадковується від кількох класів $C_{21}, C_{22}, \dots C_{2n}$, то будемо це записувати у вигляді $C_1 \to (C_{21}, C_{22}, \dots, C_{2n})$ де $C_{2i} \neq C_{2j}, i \neq j$, та $C_1 \neq C_{2i}, i = 1..n$. Порядок, в якому класи $C_{21}, C_{22}, \dots, C_{2n}$ перераховуються є важливим, тобто, якщо поміняти місцями, наприклад, перший і другий класи $C_1 \to (C_{22}, C_{21}, \dots, C_{2n})$ то ми отримуємо інше успадкування. Якщо клас успадковується рівно від одного класу, то назвемо це одиночним успадкуванням, якщо ж він успадковується більш ніж від одного класу, то назвемо це множинним успадкуванням. Клас, який успадковується від інших класів, називається нащадком або сином, а клас, від якого успадковуються інші класи таблиць, називається предком або батьком.

Підмножина відношення успадкування утворює ієрархію класів таблиць. Наприклад $\{C \to (A, B),\ E \to (B, D)\}$ задає ієрархію класів таблиць. Але не будь яка підмножина відношення успадкування утворює ієрархію класів таблиць.

Сформулюємо умови, за яких підмножина відношення успадкування утворює ієрархію класів таблиць. Нехай є дві пари наслідування класів $A \to (B_1, B_2, \dots, B_k)$ і $C \to (D_1, D_2, \dots, D_l)$. Якщо існує таке B_i, що $C = B_i$, то назвемо такі пари зв'язаними. Послідовність пар успадкування $I_1, I_2, \dots, I_m$ утворює ланцюг успадкування, якщо будь яких два послідовних елемента є зв'язаними, тобто I_j, I_{j+1}, j=1..m-1 є зв'язаними. Якщо, крім того, I_m зв'язана з I_1 то такий ланцюг успадкування утворює цикл. Підмножина відношення успадкування називається ациклічною, якщо будь який ланцюг успадкування з цієї підмножини не утворює цикл.

Для того, щоб уникнути неоднозначності при перерахуванні предків деякого класу, достатньо вимагати, щоб вибрана підмножина відношення успадкування була функціональною, тобто для будь яких пар $a \to b$ і $c \to d$, з цієї підмножини, $a \neq b$.

Таким чином ієрархією класів таблиць, або просто ієрархією класів назвемо пару (S, I) де S – множина класів таблиць, а I - підмножина відношення успадкування, яка є ациклічною, функціональною і яка утворюється з класів з множини S.

Для деякого класу $C(a_1, a_2, \dots, a_k)$ звернення до атрибуту a_i будемо записувати у вигляді $C.a_i$. У випадку, коли клас входить до деякої ієрархії класів, це звернення трактується у розширеному сенсі. А саме, якщо атрибут a відсутній безпосередньо в класі C, то він шукається в предках цього класу. Але може бути ситуация, коли атрибут зустрічається в кількох батьківських класах. Тоді виникає питання, який з цих двох атрибутів вибрати. Для цього використовується спеціальний метод, який називається лінеаризацією. Введемо кілька означень. Нехай в ієрархії наслідування приймають участь тільки ті пари наслідування, в яких в якості предку використовується тільки один клас Такі ієрархії будемо називати ієрархіями одиночного успадкування класів. Якщо ж є хоча б одна пара з кількістю предків більших за одиницю, то будемо називати її ієрархією множинного успадкування класів. Якщо в ієрархії одиночного успадкування з всіх пар успадкування можна утворити один ланцюг, який містить всі класи, то таку ієрархію назвемо простою ієрархією одиночного успадкування класів.

За аналогією з деревами, клас, який не має жодного нащадку будемо називати листом, а клас, який не має жодного предка – коренем. Відмітимо, що для ієрархії одиночного успадкування класів лист має не більш ніж одного предка.

В підрозділі 4.2 «Одиночне успадкування таблиць» розглядаються деякі властивості ієрархії класів з одиночним успадкуванням таблиць

Теорема 4.1: Непуста ієрархія одиночного успадкування класів таблиць буде простою тоді і тільки тоді, коли в ній існує рівно лист.

Теорема 4.2. Якщо в ієрархії класів таблиць з одиночним успадкуванням існує k листів, то тоді мінімальна потужність множини ланцюгів успадкування які містять всі класи, дорівнює k.

Теорема 4.3. Якщо в ієрархії класів таблиць з одиночним успадкуванням існує k коренів, то тоді мінімальна потужність множини всіх ланцюгів успадкування, які містять всі класи ієрархії не може бути менша ніж k .

Наслідок 4.1. В ієрархії одиночного наслідування кількість листів завжди більша або дорівнює кількості коренів.

В підрозділі 4.3 «Множинне успадкування таблиць» розглядається ієрархія множинного успадкування класів таблиць.

Для вирішення проблеми вибору найбільш підходящого атрибуту з множини атрибутів з однаковими іменами, застосовується такий же самий підхід, як і для одиночного успадкування, а саме метод лінеаризації, коли

рухаємося від найближчого класу до того, який знаходиться на більший відстані, якщо під відстанню розуміти довжину ланцюга успадкування.

Проблема полягає в тому, що при множинному успадкуванні може бути кілька класів, які знаходяться на однаковій відстані. Тому, для лінеаризації множинного успадкування треба вводити додаткові критерії порівняння класів. Одним із таких критеріїв є локальний лінійний порядок. А саме, якщо в ієрархії класів існує пара успадкування $A \to (A_1, A_2, \dots, A_n)$, то будимо говорити, що A_i менший за A_j відносно класу A, якщо $i \leq j$ і записувати це у вигляді $A_i \leq_A A_j$. Це відношення назвемо локальним порядком успадкування класів.

Іншою важливою властивістю є монотонність. Змістовно кажучи, монотонність означає, що лінеаризація не порушує відносний порядок слідування класів при переході від деякого класу до будь якого його потомку. Іншими словами, якщо в лінеаризації деякого класу A клас B знаходиться ліворуч від класу C то і в лінеаризації будь якого потомка класу A ця властивість зберігається.

Дамо кілька визначень. Операцією лінеаризації будемо називати функцію, яка для ієрархії класів і деякого класу з цієї ієрархії, повертає список класів, який починається з класу, переданого в якості параметру, та всіх його предків, які розміщуються в списку без повторень. Позначимо операцію лінеаризації через L. Отриманий список будемо називати лінеаризацією класу.

Якщо в списку l елемент x знаходиться ліворуч від y то будемо позначати це через $x \prec_l y$. Будемо говорити, що список l_1 включається в список l_2 зі збереженням порядку, якщо всі елементи l_1 входять до списку l_2, причому для будь яких двох елементів x_1 x_2 які входять до l_1 таких що $x_1 \prec_{l_1} x_2$ має місце $x_1 \prec_{l_2} x_2$. Таке включення позначимо через $l_1 \sqsubseteq l_2$.

Операція лінеаризації називається монотонною, якщо для будь якої ієрархії класів I_C , довільного класу A який входить до цієї ієрархії, та будь якого його потомка B, виконується $L(I_C, A) \sqsubseteq L(I_C, B)$.

Базуючись, на введених означеннях, в дисертації надається формальне визначення одного з найбільш поширених алгоритмів лінеаризації, який називається MRO C3.

Теорема 4.4 Алгоритм MRO C3 є монотонним

Твердження 4.2 Алгоритм лінеаризації на коректних вхідних даних завжди завершується.

Існують приклади, які показують, що алгоритм не зберігає локальний порядок успадкування.

ВИСНОВКИ

Головним результатом дисертації є розширення табличної алгебри семантичних функцій мови SQL одиночним та множинним успадкуванням таблиць з автоматичним розв'язанням конфліктів імен атрибутів за допомогою алгоритму лінеаризації, відомого під назвою MRO C3. Це розширення розв'язує важливу задачу побудови математичної моделі об'єктно-реляційних баз даних, що має велике теоретичне та практичне значення для розробки перспективних комп'ютерних інформаційних систем.

Основні результати включають в себе:

1. Для більш точного опису семантики ядра мови SQL та більшої зручності у використанні замість старих визначень в табличній алгебрі семантичних функцій SQL запропоновані нові визначення операцій внутрішнього та зовнішнього з'єднання, а також теоретико множинних операцій.
2. Доведена теорема, яка визначає умови, при яких семантичні функції задають точну семантику оператора ORDER BY. Це показало, що повна семантика ORDER BY може бути задана лише недетермінованими функціями.
3. Доведені теореми про характеристичні ознаки одиночного успадкування. На їх базі розроблені алгоритми перевірки, чи є ієрархія успадкування одиночним чи множинним успадкуванням, та чи є ієрархія одиночного успадкування простою.
4. Формально визначені такі важливі властивості методу лінеаризації ієрархії таблиць, як монотонність та збереження локального порядку успадкування, що дає можливість оцінити той чи інший метод лінеаризації та визначити, наскільки він є безпечним.
5. Побудовано алгоритм MRO C3 для автоматичного вирішення конфлікту співпадаючих імен атрибутів при множинному успадкуванні таблиць. Це дозволило перевіряти властивості алгоритму на строгому математичному рівні. Зокрема доведено теореми про такі властивості алгоритму, як монотонність та його завершуваність. Алгоритм розроблювався на псевдокоді, що зробило його незалежним від мови програмування. Додатково він був реалізований на мові програмування Python. Проведене тестування програмної реалізації також підтвердило коректність доведених теорем.

СПИСОК ПРАЦЬ, ОПУБЛІКОВАНИХ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1. Мохаммед К.Д. Алгоритми CLOS та LOOPS лінеаризації класів в мовах програмування: формальна побудова / К.Д. Мохаммед, Д.Б. Буй, С.В. Компан, С.А. Поляков // Вісник Київського національного університету

імені Тараса Шевченка. Серія: Фізико-математичні науки. – 2015. – Вип. 2. – С. 99-102.
http://www.library.univ.kiev.ua/ukr/host/10.23.10.100/db/ftp/visnyk/fiz_mat_2_2015.pdf

2. Мохаммед К.Д. Відношення конфінальності, передпорядки та порядки, семантика фрази ORDER BY запитів SQL-подібних мов[Електронний ресурс] / Д. Б. Буй, Н. Д. Кахута, О. В. Шишацька, Sunmade Fabunmi, К.Д. Мохаммед // Вісник Київського національного університету імені Тараса Шевченка. Серія: Фізико-математичні науки. – 2015. – Вип. 4. – С. 88-95. – Режим доступу: http://nbuv.gov.ua/UJRN/VKNU_fiz_mat_2015_4_16
3. Мохаммед К.Д. Логики частичных предикатов, индуцированные трехзначными логиками Клини / Д.Б. Буй, Е.В. Шишацкая, Ф. Санмейд, К.Д. Мохаммед // Штучний інтелект.– 2015. – №3-4. – С. 84-88. (Режим доступу: http://nbuv.gov.ua/UJRN/II_2015_3-4_10)
4. Мохаммед К.Д. Математические основания множественного наследования: рефлексивно-транзитивное замыкание. / Д.Б. Буй, Е.В. Шишацкая, Ф. Санмейд, К.Д. Мохаммед // Вісник Харківського національного університету ім. В.Н. Каразіна. – 2016. – №28. - С .19-33 http://periodicals.karazin.ua/mia/article/viewFile/6549/6058
5. Мохаммед К.Д. Рефлексивно-транзитивные замыкания бинарных отношений. / К.Д. Мохаммед, Д.Б. Буй, Е.В. Шишацкая, Ф. Санмейд // Электротехнические и компьютерные системы. – 2016. – №22(98). – С.272-276. http://www.etks.opu.ua/?fetch=articles&with=info&id=806
6. Мохаммед К.Д. Характерестичні властивості одиночного успадкування класів. / К.Д. Мохаммед // Вісник Київського національного університету імені Тараса Шевченка. Серія: Фізико-математичні науки. – 2016. – Вип. 4. – С. 104-107.
7. Mohammed K. Join Operations in Relational Databases with Automatic Attributes Renaming / Poliakov S, Buy D, Mohammed K., Israa Jasim AL.kalafa // Scholars Journal of Engineering and Technology. – 2017. –№5(6) - P.254-257. http://saspublisher.com/sjet-56/
8. Мохаммед К.Д. Огляд типів рекомендаційних систем та використання баз даних для підвищення їх продуктивності / К.Д. Мохаммед, Буй Д.Б., Компан С.В., Поляков С.А. // XI Міжнародна науково-практична конференція: ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ. – Київ, 15-17 грудня 2014 р. - с 29-34- http://taapsd.at.ua/TAAPSD-2014.pdf
9. Мохаммед К.Д. Математические основания современных реляционных СУБД / К.Д. Мохаммед, Д. Б. Буй, , Н. Д. Кахута / I Международный научно-практический форум «Наука и бизнес»: Тезисы докладов. –

Черновцы, 2015. – С. 52-55- http://library.krok.edu.ua/media/library/category/statti/kakhuta_0008.pdf

10. Mohammed K. Linearization algorithms CLOS and LOOPS of the classes in programming languages: the formal definitions / K. Mohammed, D. Buy, J. Karam, S. Kompan, S. Polyakov.// 13th International Scientific Conference on Informatics. – Poprad, Slovakia, 18-20 Nov., 2015. – P. 63-66 (Print ISBN: 978-1-4673-9867-1, DOI 10.1109/Informatics.2015.7377809). http://ieeexplore.ieee.org/document/7377809/
11. Мохаммед К.Д. Реализация работы нестандартных типов данных в Framework Django на примере типа данных ltree / К.Д. Мохаммед, Д.Б.Буй, С.В.Компан, С.А. Поляков // XII Міжнародна науково-практична конференція ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ. – Київ, 23-26 грудня 2015 року. - с 26-32- http://taapsd.at.ua/
12. Мохаммед К.Д. Недетерминированные функции в SQL / К.Д. Мохаммед, Буй Д.Б., Поляков С.А. // XIII Міжнародна науково-практична конференція ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ. – Київ, 5-9 грудня 2016 рокАу. - с 44-49- http://phd.isofts.kiev.ua/taapsd-2016/
13. Mohmmed K. Mathematical Foundations of Multiple Inheritance: Reflexive-transitive Closure of the Binary Relations / Dmytro Buy, Olena Shyshatska, Sunmade Fabunmi, Karam Mohammed // Perspective Technologies and Methods in MEMS Design (MEMSTECH), 2016 XII International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH). – 2016. – С.192-195. (http://ieeexplore.ieee.org/document/7507540/)
14. Мохаммед К.Д. Математические основания множественного наследования: рефлексивно-транзитивное замыкание / К.Д. Мохаммед, Д.Б. Буй, Е.В. Шишацкая, Ф. Санмейд // Міжнародний науково-практичний семінар – "Комбінаторні конфігурації та їх застосування" – 2016. http://it-kntu.kr.ua/2016/04/18/results-cca-2016/#more-3782
15. Mohmmed K. Математические основы множественного наследования в ООП /K. Mohmmed, Буй Д.Б., Шишацкая Е.В., Fabunmi S. // Четвёртая конференция «КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В НАУКОЕМКИХ ТЕХНОЛОГИЯХ» (КМНТ-2016). – Харьков, 26-31 мая 2016 г (http://www.dsmmph.org.ua/kmnt.html)

АНОТАЦІЯ

Мохаммед К.Д. Розширення табличних алгебр множинним успадкуванням. – Рукопис.

Дисертація на здобуття наукового ступеня кандидата фізико-математичних наук за спеціальністю 01.05.03 – математичне та програмне

забезпечення обчислювальних машин і систем. – Київський національний університет імені Тараса Шевченка, МОН України. – Київ, 2017.

Хоча семантиці SQL присвячені численні теоретичні роботи, в них не приділяється достатньо уваги об'єктно-орієнтованим розширенням мови. Однією з характеристичних особливостей об’єктних розширень є можливість одиночного успадкування таблиць. У той же час існує потреба в множинному успадкуванні таблиць. При множинному успадкуванні виникає проблема конфлікту імен колонок. Для вирішення проблеми було запропоновано застосувати спеціальний алгоритм лінеаризації відомий під назвою MRO C3. У дисертації алгоритм був модифікований для застосування до таблиць і заданий строго, з використанням псевдокоду. Формально доведена його завершуванність.

Розглянуті такі важливі для лінеаризації властивості як монотонність і збереження локального порядку успадкування. Доведена монотонність алгоритму MRO C3. Алгоритм не зберігає локальний порядок успадкування, що було показано на відповідному прикладі.

Проведено уточнення табличної алгебри семантичних функцій SQL. Запропоновано нові визначення для операцій внутрішнього і зовнішнього з'єднання і теоретико множинних операцій. В результаті, уточнена алгебра більш точно описує семантику ядра мови SQL і є більш зручною у використанні.

Проведена перевірка семантики оператора ORDER BY мови SQL. Виявлено, що семантичні функції задають частковий порядок на рядках, в той час як фраза SQL ORDER BY завжди повертає лінійний порядок на рядках таблиці. Сформульовано і доведено теорему, яка визначає умови, при яких семантичні функції задають точну семантику оператора ORDER BY.

Ключові слова: SQL, реляційні бази даних, табличні алгебри, семантика мов програмування, програмні алгебри, лінеаризація, успадкування таблиць.

АННОТАЦИЯ

Мохаммед К.Д. Расширение табличных алгебр множественным наследованием. – Рукопись.

Диссертация на соискание ученой степени кандидата физико-математических наук по специальности 01.05.03 - математическое и программное обеспечение вычислительных машин и систем. - Киевский национальный университет имени Тараса Шевченко, МОН Украины. – Киев, 2017.

Хотя семантике SQL посвящены многочисленные теоретические работы, в них не уделяется достаточно внимания объектно-ориентированным расширением языка. Одной из характеристических особенностей объектных расширений является возможность одиночного наследования таблиц. В то же

время существует потребность в множественном наследовании таблиц. При множественном наследовании возникает проблема конфликта имен колонок. Для решения проблемы было предложено применить специальный алгоритм линеаризации известный под названием MRO C3. В диссертации алгоритм был модифицирован для применения к таблицам и сформулирован строго, с использованием псевдокода. Формально доказана его завершаемость.

Рассмотрены такие важные для линеаризации свойства как монотонность и сохранения локального порядка наследования. Доказана монотонность алгоритма MRO C3. Алгоритм не сохраняет локальный порядок наследования, что было показано на соответствующем примере.

Проведено уточнение табличной алгебры семантических функций SQL. Предложены новые определения для операций внутреннего и внешнего соединения и теоретико-множественных операций. В результате, уточненная алгебра более точно описывает семантику ядра языка SQL и является более удобной в использовании.

Проведена проверка семантики оператора ORDER BY языка SQL. Выявлено, что семантические функции задают частичный порядок на строках, в то время как фраза SQL ORDER BY всегда возвращает линейный порядок на строках таблицы. Сформулированы и доказана теорема, которая определяет условия, при которых семантические функции задают точную семантику оператора ORDER BY.

Ключевые слова: SQL, реляционные базы данных, табличные алгебры, семантика языков программирования, программные алгебры, линеаризация, наследование таблиц.

ANNOTATION

Mohhamed K.J. Extending of the table algebra with multiple inheritance. – Manuscript.

Candidate's thesis on Physics and Mathematics, speciality 01.05.03 – Theoretical and software of computers and systems. – Taras Shevchenko National University of Kyiv. – Kyiv, 2017.

In the thesis, two goals were set. One goal was to formalize the algorithm for multiple inheritance of tables and study its properties. Another goal was to refine and further develop the table algebra of semantic functions of the SQL query language.

Although many theoretical works are devoted to SQL semantics, they do not pay enough attention to object-oriented language extensions. These extensions were included in the standard of language and became an important part of modern relational databases. Such DBMSs are called object-relational DBMS. One of the characteristics of object-relational DBMS is the ability to inherit tables. They use single table inheritance. At the same time, there is a practical need for multiple table inheritance. This would make it possible to more adequately model the domain and

describe the real world. With multiple inheritance, a name conflict problem occurs. It consists in the fact that in parent tables different columns can have the same names. In this case, the question arises which column to include in the child table. A similar problem, in its time, arose in object-oriented programming languages that support multiple inheritance. One way to solve the problem in object-oriented programming languages is to linearize the parent objects. In this method, a linear order is specified on the parent objects. Then the smallest object in the sense of this order is chosen. In other words, all parent objects are lined up. The attribute of the object that is located to the left of other objects that have attributes with the same names is selected in the child object. Numerous linearization algorithms have been developed. The most common among them was the MRO C3 algorithm. It was chosen as the basis for resolving name conflicts with multiple inheritance of tables. As it turned out, this algorithm was formulated at the semi-formal level, which made it impossible to formally prove its properties and correctness. In the thesis, the MRO C3 algorithm was modified to apply to tables and was specified strictly using a pseudocode. Formal proof of its ended is formally proven.

As for the properties of the algorithm, we considered such important for linearization properties as monotonicity and preservation of the local order of inheritance. For these properties also there were no mathematical definitions, which made it impossible to prove their presence or absence in the algorithm. To formally define the properties of the algorithm, it was necessary to introduce the concept of a table inheritance hierarchy that describes both single and multiple table inheritance. Theorems about characteristic of single inheritance are formulated and proved.

On the basis of the definitions introduced, the monotonicity of the algorithm MRO C3 was proved. The algorithm does not preserve the local inheritance order, which was shown in the corresponding example.

As for the semantics of SQL, the table algebra of SQL semantic functions was improved. Note that this algebra has advantages over other ways of specifying SQL semantics. Namely, in table algebra two types of functions are distinguished: functions on tables, and functions on functions, which are called operators, as is customary in other programming languages. The semantics of complex SQL expressions in this algebra are naturally constructed from functions on tables and operators.

The results of the SQL query and the results of the calculations of the corresponding semantic functions of table algebra were compared. Differences in the results of the execution of queries that contain the operations of outer and inner joins and the corresponding semantic functions on tables that have equal attribute names are detected. Also, there were discrepancies in the results of set-theoretic operations on tables, due to the fact that in SQL, union operations, intersections and table differences take into account the order of columns specified in queries, and semantic algebra takes into account the names of column attributes, and order is not important.

New definitions for operations of inner and outer join and the theoretical-set operations are proposed. As a result, the refined table algebra of semantic functions more accurately describes the semantics of the core of the SQL language and is more convenient to use.

The semantics of the SQL statement ORDER BY is checked. It has been revealed that semantic functions specify partial ordering on rows, while the SQL ORDER BY clause always returns a linear order on the rows of the table. A theorem is formulated and proved that defines the conditions under which semantic functions specify the exact semantics of the ORDER BY operator.

Key words: SQL, relational databases, table algebra, semantics of programming languages, program algebras, linearization, inheritance of tables.