УДК 004.658: 652.3

Дворецький М. Л.,
ст. викладач кафедри інженерії
програмного забезпечення,
mykhaylo.dvoretskyy@chmnu.edu.ua
Боровльова С. Ю.,
ст. викладач кафедри інженерії
програмного забезпечення,
svetlana.borovlyova@chmnu.edu.ua
Давиденко Є. О.,
канд. техн. наук, доцент (б.в.з.) кафедри
інженерії програмного забезпечення,
ЧНУ ім. Петра Могили,
м. Миколаїв, Україна,
davydenko@chmnu.edu.ua

ПРОЕКТУВАННЯ СТРУКТУРИ РОЗПОДІЛЕНОЇ БД НА БАЗІ ПАРСИНГУ SQL-ЗАПИТІВ

У рамках дослідження розглянута специфіка оперативного обліку, пов'язана із переходом від локальних до розподілених БД. Ця тенденція обумовлена наявністю різних типів обліку, наявністю географічно віддалених АРМ та необхідністю в консолідованому обліку. У світі необхідності використання розподіленої структури БД актуальним постає питання вибору оптимальної стратегії збереження для відповідних фрагментів даних розподіленої БД, а також методів та засобів їх подальшої синхронізації у випадку їх зміни та/або для виконання подальшого їх консолідованого аналізу підсистемами оперативно-аналітичної обробки даних.

Дослідження ставить на меті підвищення швидкості виконання локальних SQL-запитів із збереженням значення ступеня актуальності даних у межах допустимого значення за рахунок оптимізації структури віддаленого автоматизованого робочого місця на базі парсингу користувацьких SQL-запитів та кластеризації множини відношень центральної БД.

На першому етапі розроблено підсистему обліку користувацьких запитів із можливістю їх подальшої класифікації згідно приналежності до того чи іншого автоматизованого робочого місця, географічного розташування, ролі користувача та інших критеріїв. Класифікація, в залежності від параметру, здійснюється як в автоматичному, так і в ручному режимах. Окремим етапом є здійснення парсингу тексту запитів з метою виявлення списку відношень та атрибутів. Задачу парсингу коду SQL-запиту запропоновано вирішити за допомогою використання проекту з відкритим кодом ANTLR.

Остаточний аналіз, що виконується на базі отриманих даних дозволяє отримати список БД та відношень, що мають бути доступними, та ранжувати його за різними критеріями. Деякі відношення потребують окремої уваги при прийнятті рішення щодо їх розміщення у віддаленій БД. У цьому випадку мають бути розглянуті їх проекції, що використовуються в різних запитах, та горизонтальне розбиття відношення через вказання діапазону ключів.

Ключові слова: *інформаційна система; розподілена база даних; розподілена транзакція; консолідований облік; система керування базами даних; SQL-запит; автоматизоване робоче місце; парсинг; профайлінг; ANTLR; синхронізація даних; ETL-система.*

Постановка проблеми. Інформаційні системи (ІС), що є взаємозалежною сукупністю засобів, методів і персоналу, мають на меті зберігання, обробку та представлення інформації [1]. Створюючи базу даних, користувач прагне впорядкувати інформацію з різних ознак для швидкого отримання потрібних відомостей з довільним сполученням ознак. У розвитку сучасних інформаційних систем намітилася тенденція переходу від локальних баз даних до створення розподілених баз [2]. Це пов'язано із деякими особливостями ведення обліку на підприємстві [3].

По перше цей факт обумовлено необхідністю автоматизації різних типів обліку, таких як складський, бухгалтерський облік, облік кадрів, розробка інфор-

маційних порталів, систем відеоспостереження, контролю прав доступу та ін. Автоматизація всіх видів обліку в одній універсальній системі має свої переваги, але недоліків, пов'язаних із таким підходом, значно більше [4]. Серед них можна навести такі, як перевантаженість центральної БД великою кількістю даних та користувачів, низька відмовостійкість, вразливість системи та недостатньо розвинені механізми обліку більшості напрямків автоматизації.

Другим аспектом, що може обумовити необхідність створення та використання бази даних розподіленої структури, може бути холдингова структура об'єкта автоматизації та/або географічна віддаленість філіалів підприємства. Звичайно, в даному випадку також можливе використання центральної БД із роботою через виділений канал зв'язку або розміщення БД у хмарі [5, 6]. Але цей підхід, знову ж таки збільшує навантаження на центральну БД, канали зв'язку, та знижує відмовостійкість системи. Крім того, існують фрагменти даних, що мають бути оперативно доступними 24/7 незалежно від наявності зв'язку, навіть за рахунок втрати їх актуальності.

Використання різних систем автоматизації або наявність декількох віддалених філіалів компанії із окремими системами автоматизації обліку зумовлює необхідність консолідованого аналізу даних відносно об'єкту автоматизації в цілому. При використання декількох БД та у деяких випадках навіть декількох різних СКБД задача може бути вирішена за допомогою використання розподілених транзакцій [7, 8], але даний підхід не можна назвати оптимальним. Іншим підходом є використання підсистем синхронізації даних із різних оперативних джерел даних із використанням ETL-систем [9].

Вищенаведені фактори, а також деякі інші аспекти, зумовлюють актуальність розподілених баз даних. Розподілена база даних (DDB – distributed database) – це сукупність взаємопов'язаних баз даних, розподілених у комп'ютерній мережі. Система управління розподіленою базою даних визначається як програмна система, яка управляє базою даних у такий спосіб, щоб її розподіленість була прозорою для користувачів [10]. Прозорість – це поняття незалежності даних у розподілених системах, яке передбачає, що користувач у цій системі працює з розподіленою базою даних як з логічно цілісною сукупністю даних, тобто на його роботу не повинно впливати те, як дані розподілені між вузлами мережі. Отже, в розподіленій системі користувачеві надається логічно цілісне подання фізично розподіленої бази даних.

На ринку програмних засобів з'явились розподілені СКБД, які дають змогу підтримувати та обробляти базу даних у багатокористувацьких системах. Основною задачею розподіленої СКБД є забезпечення управління доступом до даних багатьох споживачів і цілісності й узгодженості даних в умовах використання мережі ЕОМ. Тобто основна функція таких СКБД – це координування спільної роботи багатьох користувачів з розподіленою інформацією [11]. Розв'язання проблеми автономності роботи користувачів розподіленої системи створює багато специфічних проблем в організації баз даних, оскільки різні користувачі можуть працювати паралельно з одними й тими самими даними, виконуючи з ними різні перетворення.

Враховуючи вищесказане актуальним постає питання вибору оптимальної стратегії збереження для відповідних фрагментів даних розподіленої БД, а також методів та засобів їх подальшої синхронізації у випадку їх зміни та/або для виконання подальшого їх консолідованого аналізу підсистемами оперативно-аналітичної обробки даних.

Аналіз останніх досліджень і публікацій. Існує декілька підходів щодо представлення даних у БД розподіленої структури. Розподілена стратегія без дублювання визначає дані, які потрібно зберігати в кожному вузлі мережі. Проектування даних за такої стратегії є складною задачею. Ключовим фактором, який впливає на надійність і доступність бази даних, є так звана локалізація посилань. Розглянута стратегія підходить для тих предметних областей, в яких практично немає дублювання даних у різних вузлах мережі і потрібна мінімальна кількість логічних посилань для виконання інформаційних взаємозв'язків вузлів одного з одним. Тобто користувач кожного вузла працює зі своїми файлами і досить рідко використовує дані інших вузлів мережі. Головним суттєвим недоліком цієї стратегії є необхідність використання розподілених транзакцій при умові існування взаємозалежних даних на різних вузлах мережі. По-перше, використання розподілених транзакцій передбачає одночасну доступність всіх вузлів у мережі. По-друге, блокування, накладені на кожному окремому вузлу будуть зняті лише після повного завершення роботи розподіленої транзакції [7, 11, 12, 13].

Розподілена (децентралізована) стратегія з дублюванням полягає в тому, що база даних проектується як за централізованого підходу, але фізично дублюється в кожному вузлі мережі. Кожний вузол має свою копію, продубльовану стільки разів, скільки вузлів у мережі. Стратегія розподілу з дублюванням найбільш ефективно розв'язує проблеми доступу та вибірки даних з мінімальними витратами часу. Цей підхід характеризується складністю адміністрування та розв'язання проблеми узгодженості файлів БД у різних вузлах мережі. Найбільш гостро ця проблема постає тоді, коли зв'язок у мережі порушується і в копії в різних вузлах виникають розбіжності. У цьому разі потрібно розробити спеціальний механізм для узгодження деяких копій бази даних [8, 12, 13].

Комбінована стратегія розподілу даних поєднує два підходи, пов'язані з розподілом без дублювання та з дублюванням даних, з метою використання їх переваг. Остання стратегія є найбільш виправданою із точки зору можливості поєднання переваг всіх попередніх. Але при її використанні, окрім задачі синхронізації дубльованої інформації, актуальною постає задача оптимального проектування структури БД з точки зору приналежності даних до категорії того чи іншого вузла мережі [11, 12, 13]. Крім того, продуктивність системи напряму буде залежати від прийняття рішення щодо необхідності часткового або повного дублювання даних.

Загальні рекомендації щодо вибору тієї чи іншої стратегії збереження та подальшої синхронізації да-

них або не враховують специфіку системи обліку об'єкта автоматизації, або особливості її використання на тому чи іншому підприємстві. Отже вважається за необхідне проведення попереднього розбиття множини відношень централізованої БД на підмножини на базі результатів парсингу користувацьких SQL-запитів із визначенням їх перетинів із подальшою їх мінімізацією, оскільки однією з основних проблем розподілених БД є задача синхронізації даних, що дублюються.

Постановка завдання. Метою дослідження є підвищення швидкості виконання локальних SQL-запитів із збереженням значення ступеня актуальності даних у межах допустимого значення за рахунок оптимізації структури віддаленого автоматизованого робочого місця на базі парсингу користувацьких SQL-запитів та кластеризації множини відношень центральної БД. Гіпотеза полягає у тому, що у результаті парсингу SQL-запитів, що надходять до централізованої СКБД від АРМ, попередньо класифікованих за ознаками приналежності до групи користувачів, фізичного розташування (віддаленості) та оцінкою необхідного ступеня актуальності даних, можливо отримати множину відношень БД інформаційної системи та провести її кластеризацію за ступенем «близькості» відношень у запитах. Виконаний аналіз дозволить провести оптимізацію структури БД віддаленого АМР, що у свою чергу призведе до підвищення швидкості та ефективності використання ПЗ інформаційної системи.

Досягнення поставленої мети вимагає: проведення аналізу предметної області для побудови коректної моделі класифікації користувацьких SQL-запитів за ознаками приналежності до групи користувачів, фізичного розташування (віддаленості) та оцінкою необхідного ступеня актуальності даних; дослідження методів створення і використання формальних граматик та розробка підсистеми парсингу коду SQL-запитів до центральної БД; виконання розбиття множини відношень (таблиць) на підмножини, виявлення перетинів отриманих підмножин, їх аналіз та виявлення ступенів актуальності та швидкості отримання даних; реалізація механізмів синхронізації даних між віддаленими вузлами РБД для підмножини відношень із асинхронним режимом оновлення даних.

Виклад основного матеріалу. Першим кроком даного дослідження є створення підсистеми обліку користувацьких запитів із можливістю їх подальшої класифікації згідно приналежності до того чи іншого автоматизованого робочого місця, географічного розташування, ролі користувача та інших критеріїв, що можливо додати до системи динамічно під час її використання згідно з особливостями тієї чи іншої предметної області.

На концептуальному рівні у вищезазначеній моделі модна виділити наступні сутності. Місце розташування – характеризується атрибутами назва, приналежність до БД, швидкість каналу зв'язку та його надійність (по шкалі від 1 до 10 – визначається згідно статистики відмов або, у разі її відсутності, за відгуками користувачів програмного забезпечення); робоча станція із атрибутами назва, місце розташування, приналежність до типу автоматизованого робочого місця та список програмного забезпечення; а також набір сутностей без додаткових атрибутів, таких як програмне забезпечення, типи автоматизованого робочого місця, БД, відношення та ін. Основним відношенням є список користувацьких запитів. Сутність користувацький запит характеризується такими атрибутами, як список відношень, робоча станція та програмне забезпечення. Логічну архітектуру моделі наведено на рис. 1.

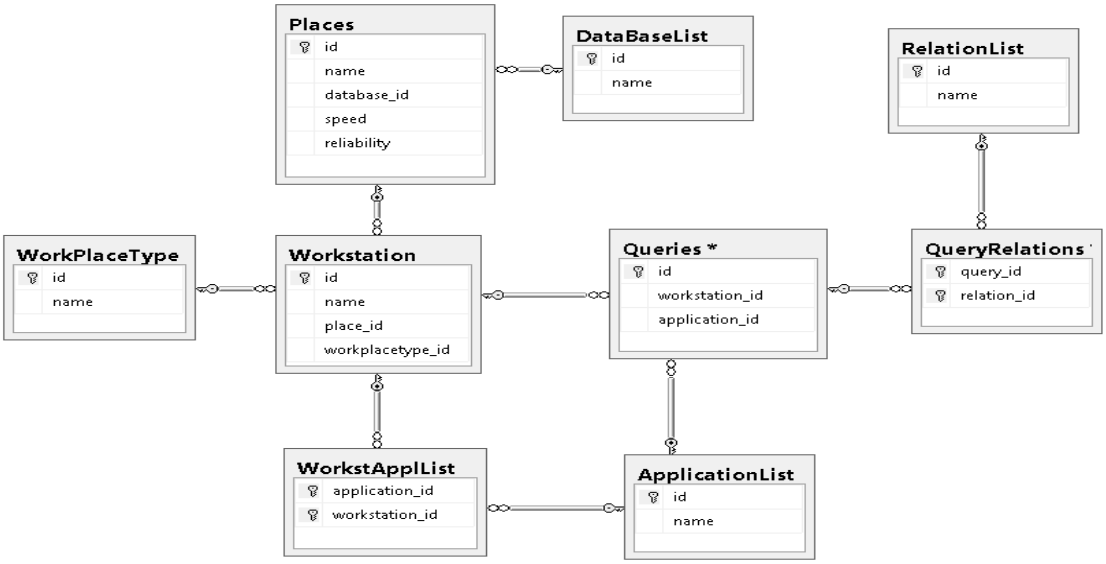


Рис. 1. Логічна архітектури підсистеми профайлингу користувацьких SQL-запитів

Підсистема має користувацький інтерфейс, що складається з форм вводу та редагування даних для таких сутностей, як типи автоматизованого робочого місця, робочі станції, місця розташування, бази даних та програмне забезпечення. Крім того, передбачені механізми імпорту даних із текстових та csv файлів для можливості взаємодії із сторонніми програмними продуктами при заповненні даних наприклад по таким сутностям, як програмне забезпечення або список робочих станцій. Ця інформація може бути отримана наприклад із ActiveDirectory або із використанням таких безкоштовних програмних продуктів, як

Advanced IP Scanner, NetWrix Inactive Users Tracker або WinAudit Freeware. Звісно, у випадку використання операційної системи, відмінної від Microsoft Windows існує багато інших адміністративних засобів, що дозволяють отримати такого роду інформацію.

Однак поряд із звичними формами вводу та засобами імпорту даних із стороннього ПЗ присутні сутності, що наповнюються даними виключно із використанням методів сканування структури БД. Так, наприклад, список відношень БД у випадку використання системи керування базами даних MS SQL Server може бути оновлено із використанням наступного скрипта SQL, наведеного на рис. 2.

```
delete from RelationList
where id not in(select query_id from QueryRelations)

declare @max_id int = (select isnull(max(id), 0) from RelationList)

insert into RelationList(id, name)
select ROW_NUMBER() over(order by object_id) + @max_id, name
from sys.objects
where type = 'U'
```

Рис. 2. T-SQL скрипт для оновлення даних в списку відношень бази даних

Цей фрагмент коду наведено для випадку використання однієї із версій MS SQL Server. Переважна більшість сучасних систем керування реляційними базами даних, таких як Oracle, MySQL, Postgree SQL, FireBird та ін., також мають у своєму розпорядженні механізми по отриманню даних списку користувацьких таблиць, але синтаксис команд при цьому буде різнитися. Виходячи із цього, програмно дана частина реалізована за допомогою композиції із використанням шаблону ООП «стратегія», отже достатньо лише обрати СКБД, що підтримується в налаштуваннях ПЗ, наступні ж дії для користувача не будуть відрізнятися ніяким чином.

Наступним кроком є заповнення основного відношення – списку користувацьких запитів із прив'язкою до робочої станції на програмного забезпечення, що генерує той чи інший запит. Сучасні СКБД мають у своєму розпорядженні досить розвинені механізми профайлінгу користувацької активності. В даній публікації розглянуто приклад отримання даних у випадку роботи із однією із версій MS SQL Server. Однак у випадку використання іншої СКБД, існують схожі за функціоналом утиліти, що можуть бути використані. На рис. 4–5. наведено екрану форма ПЗ MS SQL Profiler при налаштуванні та зборі даних щодо користувацьких SQL-запитів.

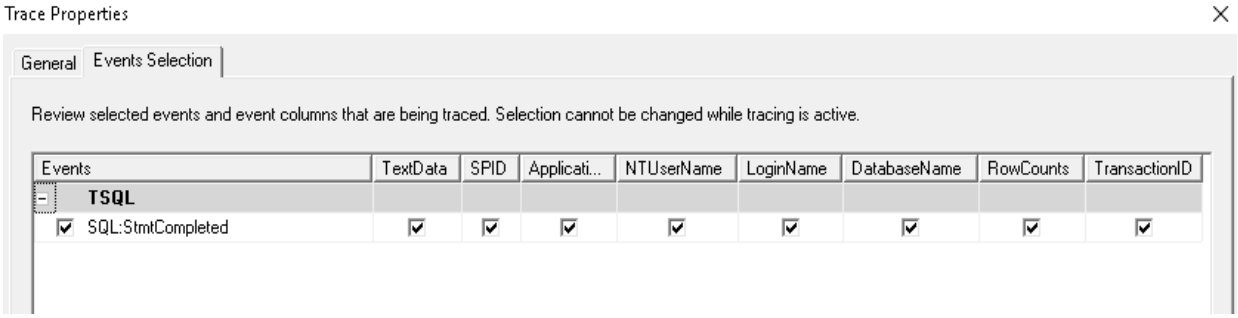
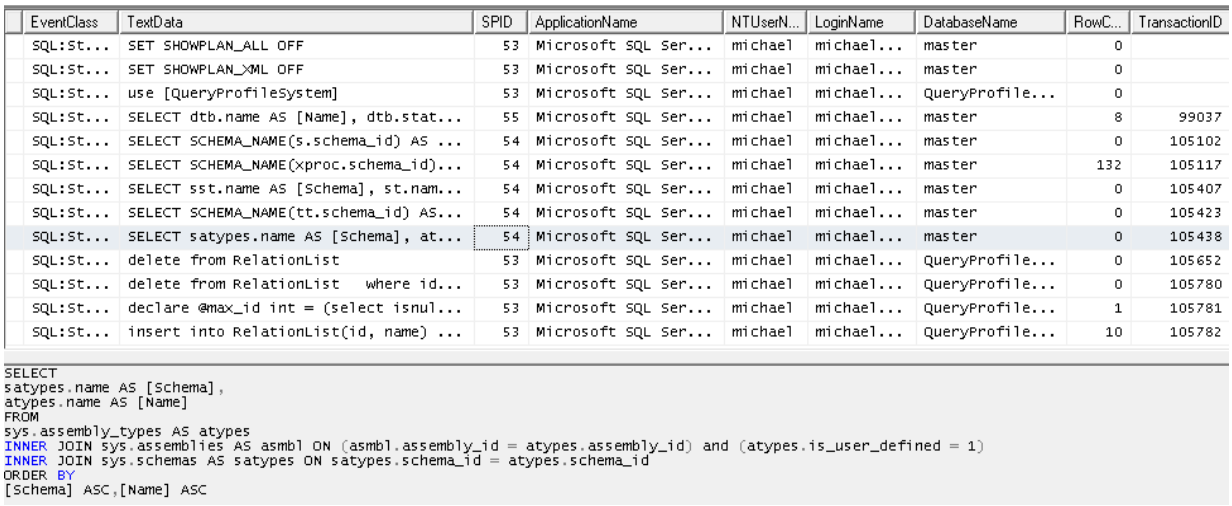


Рис. 4. Налаштування SQL Server Profiler для відстеження користувацьких SQL-запитів



EventClass	TextData	SPID	ApplicationName	NTUserN...	LoginName	DatabaseName	RowC...	TransactionID
SQL:St...	SET SHOWPLAN_ALL OFF	53	Microsoft SQL Ser...	michael	michael...	master	0	
SQL:St...	SET SHOWPLAN_XML OFF	53	Microsoft SQL Ser...	michael	michael...	master	0	
SQL:St...	use [QueryProfileSystem]	53	Microsoft SQL Ser...	michael	michael...	QueryProfile...	0	
SQL:St...	SELECT dtb.name AS [Name], dtb.stat...	55	Microsoft SQL Ser...	michael	michael...	master	8	99037
SQL:St...	SELECT SCHEMA_NAME(s.schema_id) AS ...	54	Microsoft SQL Ser...	michael	michael...	master	0	105102
SQL:St...	SELECT SCHEMA_NAME(xproc.schema_id)...	54	Microsoft SQL Ser...	michael	michael...	master	132	105117
SQL:St...	SELECT sst.name AS [Schema], st.nam...	54	Microsoft SQL Ser...	michael	michael...	master	0	105407
SQL:St...	SELECT SCHEMA_NAME(tt.schema_id) AS...	54	Microsoft SQL Ser...	michael	michael...	master	0	105423
SQL:St...	SELECT satypes.name AS [Schema], at...	54	Microsoft SQL Ser...	michael	michael...	master	0	105438
SQL:St...	delete from RelationList	53	Microsoft SQL Ser...	michael	michael...	QueryProfile...	0	105652
SQL:St...	delete from RelationList where id...	53	Microsoft SQL Ser...	michael	michael...	QueryProfile...	0	105780
SQL:St...	declare @max_id int = (select isnul...	53	Microsoft SQL Ser...	michael	michael...	QueryProfile...	1	105781
SQL:St...	insert into RelationList(id, name) ...	53	Microsoft SQL Ser...	michael	michael...	QueryProfile...	10	105782

```
SELECT
satypes.name AS [Schema],
atypes.name AS [Name]
FROM
sys.assembly_types AS atypes
INNER JOIN sys.assemblies AS asmbl ON (asmbl.assembly_id = atypes.assembly_id) and (atypes.is_user_defined = 1)
INNER JOIN sys.schemas AS satypes ON satypes.schema_id = atypes.schema_id
ORDER BY
[Schema] ASC,[Name] ASC
```

Рис. 5. Відстеження користувацьких SQL-запитів в SQL Server Profiler

Як можна побачити, прив'язати запит та транзакцію до робочої станції та програмного забезпечення не представляє великої складності, оскільки ці дані вже явним чином представлені у таблиці, що надаються профайлінговими програмами. Однак отримати список відношень, що задіяні у запиті, не є зовсім тривіальною задачею, оскільки вимагає попереднього парсингу коду SQL-запиту та виділення з нього списку відношень, які були задіяні при зверненні до даних.

Задачу парсингу коду SQL-запиту запропоновано вирішити за допомогою використання проекту з відкритим кодом ANTLR. ANTLR – буквально англ. Another Tool For Language Recognition – генератор парсерів, дозволяє автоматично створювати програму-парсер (як і лексичний аналізатор) однією з декількох цільових мов програмування (Java, C++, C#, Python, Ruby) за описом LL(*)-граматики мовою, близькою до EBNF. Дозволяє конструювати компілятори, інтерпретатори, транслятори з різних формальних мов. Також, надає зручні засоби для відновлення після помилок, і повідомлення про них. ANTLR – продовження PCCTS (Purdue Compiler Construction Tool Set), який було розроблено 1989 року. (https://uk.wikipedia.org/wiki/ANTLR).

Загальний синтаксис команди select мови T-SQL наведено на рис. 6.

```
-- Syntax for SQL Server and Azure SQL Database

<SELECT statement> ::=
    [ WITH { [ XMLNAMESPACES ,] [ <common_table_expression> [,...n] ] } ]
    <query_expression>
    [ ORDER BY { order_by_expression | column_position [ ASC | DESC ] }
  [ ,...n ] ]
    [ <FOR Clause>]
    [ OPTION ( <query_hint> [ ,...n ] ) ]
<query_expression> ::=
    { <query_specification> | ( <query_expression> ) }
    [  { UNION [ ALL ] | EXCEPT | INTERSECT }
        <query_specification> | ( <query_expression> ) [...n ] ]
<query_specification> ::=
SELECT [ ALL | DISTINCT ]
    [TOP ( expression ) [PERCENT] [ WITH TIES ] ]
    < select_list >
    [ INTO new_table ]
    [ FROM { <table_source> } [ ,...n ] ]
    [ WHERE <search_condition> ]
    [ <GROUP BY> ]
    [ HAVING < search_condition > ]
```

Рис. 6. Синтаксис команди select для SQL Server

Реалізація підсистеми парсингу T-SQL запитів починається із створення граматики. Правила пишуться в граматиці на спеціальній мові, заснованій на регулярних виразах. Синтаксис ANTLR знайомий більшості програмістів, тому що він схожий на синтаксис C і його похідні з деякими розширеннями для опису граматики. На рис. 7. наведено фрагмент вихідного коду створеної граматики що відповідає за команду select.

```
select_statement
    : with_expression? query_expression order_by_clause? for_clause? option_clause? ';'?
    ;

query_expression
    : (query_specification | '(' query_expression ')') union*
    ;

query_specification
    : SELECT (ALL | DISTINCT)? (TOP expression PERCENT? (WITH TIES)?)?
      select_list
      // https://msdn.microsoft.com/en-us/library/ms188029.aspx
      (INTO table_name)?
      (FROM table_sources)?
      (WHERE where=search_condition)?
      // https://msdn.microsoft.com/en-us/library/ms177673.aspx
      (GROUP BY group_by_item (',' group_by_item)*)?
      (HAVING having=search_condition)?
    ;
```

Рис. 7. Фрагмент реалізації команди select SQL Server у середовищі ANTLR v4.

Для реалізації програмного забезпечення на базі створеної граматики була обрана мова програмування Java SE 1.8 та середовище созробки IDE Eclipse, що дозволяє створювати, редагувати граматику та автоматично генеровути класи лексора та парсера. Крім того IDE Eclipse надає у розворядження розробника можливість переглядати граматику у вигляді синтаксичної діаграми. На рис. 8–9 наведені фрагменти реалізації реалізації команди select SQL Server у середовищі ANTLR v4 у вигляді синтаксичної діаграми.

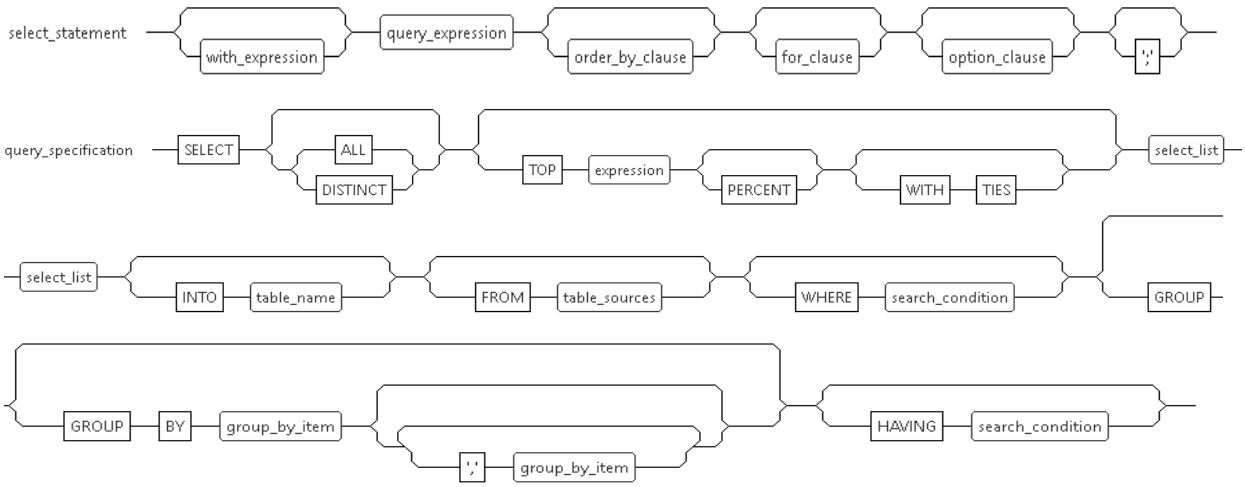


Рис. 8. Синтаксичні діаграми фрагменту реалізації команди select SQL Server у середовищі ANTLR v4 (част. 1)

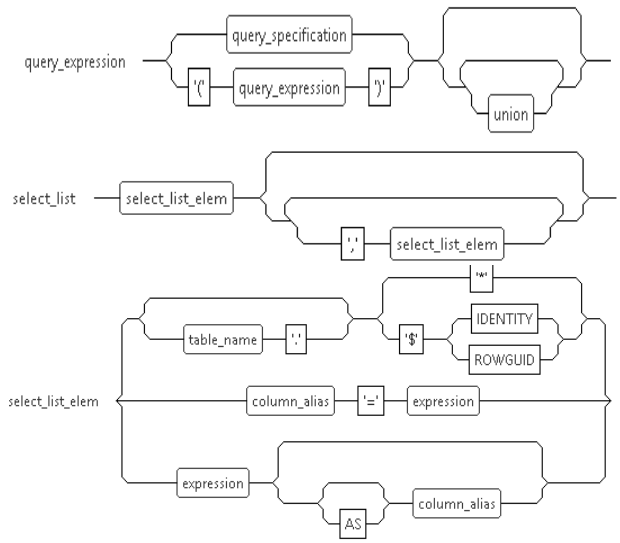


Рис. 9. Синтаксична ліаграми фрагменту реалізації команди select SQL Server у середовищі ANTLR v4 (част.2)

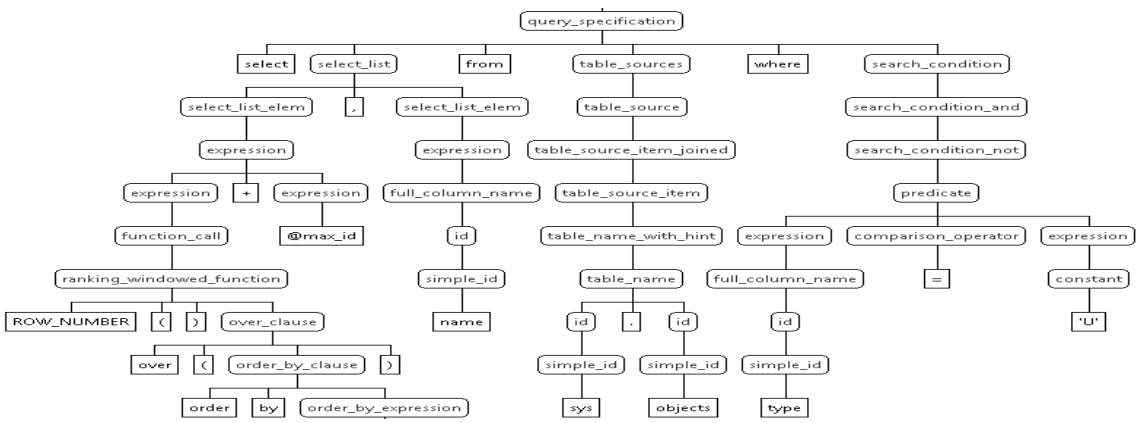


Рис. 10. Дерево парсингу тестового запиту T-SQL

У результаті виконання профайлінгу користувацьких SQL-запитів та парсингу їх тексту для визначення списку відношень, маємо таблицю наступного вигляду (рис.11). Результуюча таблиця отримується на базі запиту до БД, структура якої наведена на рис.1. та представлена наступними атрибутами: тип робочого місця, база даних, місце розташування, програмне забезпечення, відношення (таблиця БД), запит, транзакція, тривалість виконання запиту та об'єм прочитаних даних у мб.

Тип робочого місця	База даних	Місце розташування	Програмне забезпечення	Відношення	Запит (id)	Транзакція (id)	Тривалість м.с.	Об'єм, мб

Рис. 11. Результуюча таблиця користувацької активності

Остаточний аналіз, що виконується на базі отриманих даних, може бути проведено у різний спосіб. Так, наприклад, маючи постановку задачі про відкриття віддаленого філіала та список програмного забезпечення і типів робочих місць, що буде використовуватись, можна отримати список БД та відношень, що мають бути доступними для виконання користувачами поточних операцій. Список відношень БД може бути ранжовано відносно загальної та відносної тривалості виконання, об'єму прочитаних даних а також відносній представленості у загальній кількості запитів. Аналіз може проводитись у межах запиту, або у межах транзакції, що об'єднує у собі декілька запитів (команд), що сприймаються СКБД як єдиний неподільний сеанс активності.

Далі кожне з відношень отриманого списку аналізується на предмет доцільності розташування у віддаленій БД із подальшою необхідністю у синхронізації із центральною або використання напряму у центральній БД у рамках розподіленої транзакції. При прийнятті рішення враховуються наступні фактори. По перше, впливати буде ступінь використовуваності відношення у загальній кількості запитів віддаленого АРМ. Чим вище показник використовуваності, тим краще буде розташувати це відношення локально, тим самим підвищивши відмовостійкість системи. Другим впливовим чинником є кількість перетинів на відношенні з іншими віддаленими АРМ, що працюють із власною локальною копією даних. Крім того, окремо слід враховувати відносну кількість команд модивікації даних до загальної кількості звернень. У випадку великої кількості перетинів та операцій модифікації, більш бажаним є варіант звернення до центральної БД, оскільки досить складною постає задача подальшої синхронізації. Останнім із найбільш впливовох факторів є ступінь антуальності даних, що вимагається від того чи іншого відношення да віддаленому АРМ. Чим вище ступінь антуальності, тим інтенсивнішим має бути обмін даними із центральної БД, а отже варіант розподіленої транзакції в даному випадку може мати свої переваги.

Відношення із максимальним ступенев використовуваності, великою кількістю перетинів та операцій модифікації даних та ступенем актуальності даних звісно потребують окремої уваги при прийнятті рішення щодо їх розміщення у віддаленій БД. По перше, мають бути розглянуті проекції, що використовуються в різних запитах, тобто проведений аналіз щодо частоти використання окремихатрибутів, особливо в операціях модифікації даних. За рахунок цього, кількість перетинім на відношенні може бути значним чином зменшена за рахунок їх заміщення перетинами на підмножинах проекцій відношення. Якщо ж зменшити кількість перетинів за рахунок введення проекцій не вдається, можливе гризонтальне розбиття відношення через вказання діапазону ключів. Але таке розбиття стає можливим тільки у випадку використання в різних вузлах різних фрагментів даних відношення та вимагає більш детального аналізу звернень до відношення на рівні деталізації запис.

Висновки та перспективи подальших досліджень. У ході виконання дослідження було розроблено підсистему обліку та класифікації користувацьких SQL-запитів за такими ознаками, як приналежність до робочої станції, дислокації, бази даних, типу автоматизованого робочого місця та програмного забезпечення. Закладена можливість створювати користувацькі характеристики класифікації для можливості при необхідності введення додаткових аналітичних розрізів аналізу.

Для виявлення списку відношень та їх атрибутів із загального тексту отриманого SQL-запиту було реалізовано підсистему парсингу на базі формальних граматик та за допомогою використання проекту з відкритим кодом ANTLR. Також закладені механізми по автоматичному заповненню даними деяких сутностей. Ці операції можуть бути виконані через проміжні структури типу txt або csv, сформовані сторонніми ПЗ, наприклад на базі системних адміністративних утиліт. Дані механізми можуть бути кориснми при оновленні інформації таких сутностей, як списки програмного забезпечення, робочих станцій, користувачів та ін.

Отримані дані аналізуються на предмет визначення для відношень БД ступенів використання та перетинів з іншими АРМ для формулювання рекомендацій щодо представлення копії відношення у локальній БД із подальшою синхронізацією із головною версією центральної БД, або роботи із ним у рамках розподіленої транзакції. При отриманні суперечливих результатів запропоновано проведення поглибленого аналізу відношення шляхом вивчення проекцій та фільтрів за діапазонами ключів, що використовуються в запитах.

Результатом дослідження є підвищення швидкості виконання локальних SQL-запитів із збереженням значення ступеня актуальності даних у межах допустимого значення за рахунок оптимізації структури віддаленого автоматизованого робочого місця на базі парсингу користувацьких SQL-запитів та кластеризації множини відношень центральної БД. Виконаний аналіз дозволяє провести оптимізацію структури БД віддаленого АМР, що у свою чергу призводить до підвищення швидкості та ефективності використання ПЗ інформаційної системи.

ЛІТЕРАТУРА

1. Сутність інформаційних технологій й інформаційних систем [Електроний ресурс]. Режим доступу : URL: http://studopedia.org/10-121662.html.
2. Фісун М. Т. Аналіз та вибір моделей даних при створенні систем автоматизованого проектування / М. Т. Фісун, Є. О. Давиденко // Збірник наукових праць НУК. – Миколаїв : НУК, 2013 – №2(447). – С. 89–94.
3. Исаев Г. Н. Проектирование информационных систем. Учебное пособие. Омега-Л, 2015 г. – 424 с.
4. Гладкий А. Складской учет на компьютере. Лучшие программы, включая 1С 8.2. Литрес, 2013. – 410с.
5. Десять самых полезных сервисов облачных баз данных [Електроний ресурс]. – Режим доступу : URL: https://www.databases.com.ua/article/10-cloud-services-databases.
6. Edward Mahon. Transitioning the Enterprise to the Cloud: A Business Approach. 2015. – 178 p.
7. Управление транзакциями в ORACLE [Електроний ресурс]. – Режим доступу : URL: http://www.novsu.ru/file/96492.
8. Петкович Д. Microsoft SQL Server 2008. Руководство для начинающих. Пер.с англ. – СПб. : БХВ-Питербург, 2009. – 752 с.
9. Ralph Kimball, Joe Caserta. The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data. 2014. – 528 p.
10. M. Tamer Özsu, Patrick Valduriez. Principles of Distributed Database Systems 3rd ed. Springler, 2011. – 845 p.
11. М. Тамер Оззу, Патрик Валдуриз. Распределенные и параллельные системы баз данных. [Електроний ресурс]. – Режим доступу : URL: http://citforum.ru/database/classics/distr_and_paral_sdb/.
12. Автоматическая синхронизация распределенных баз данных в разделенном режиме [Електроний ресурс]. – Режим доступу : http://stimul.kiev.ua/materialy.htm?a=avtomaticheskaya_sinkhronizatsiya_raspredelennykh_baz_dannykh_v_razdelennom_rezh.
13. Использование синхронных и асинхронных операций базы данных [Електроний ресурс]. – Режим доступу : URL:http://help.adobe.com/ru_RU/as3/dev/WS5b3cc c516d4fbf351e63e3d118666ade46-7d39.html.

М. Л. Дворецкий,
С. Ю. Боровльова,
Е. А. Давыденко,
Черноморский национальный университет
им. Петра Могилы,
г. Николаев, Украина

ПРОЕКТИРОВАНИЕ СТРУКТУРЫ РАСПРЕДЕЛЕННОЙ БД НА БАЗЕ ПАРСИНГА SQL-ЗАПРОСА

В рамках исследования рассмотрена специфика оперативного учета, связанная с переходом от локальных к распределенным. Данная тенденция обусловлена наличием разных типов учета, наличием географически удаленных АРМ и необходимостью в консолидированном учете. В свете необходимости использования распределенной структуры БД актуальным становится вопрос выбора оптимальной стратегии сохранения для соответствующих фрагментов данных распределенной БД, а также методов и средств их дальнейшей синхронизации в случае их изменения и/или для выполнения дальнейшего их консолидированного анализа подсистемами оперативно-аналитической обработки данных.

Исследование ставит целью повышение скорости выполнения локальных SQL-запросов с сохранением значения степени актуальности данных в пределах допустимого значения за счет оптимизации структуры удаленного автоматизированного рабочего места на базе парсинга пользовательских SQL-запросов и кластеризации множества отношений центральной БД.

На первом этапе разработана подсистема учета пользовательских запросов с возможностью их дальнейшей классификации в соответствии принадлежности к тому или иному автоматизированному рабочему месту, географического расположения, роли пользователя и других критериев. Классификация, в зависимости от параметра, осуществляется как в автоматическом, так и в ручном режимах. Отдельным этапом является осуществление парсинга текста запросов с целью выявления списка отношений и атрибутов. Задачу парсинга кода SQL-запроса предложено решить с помощью использования проекта с открытым кодом ANTLR.

Окончательный анализ, выполняемый на базе полученных данных позволяет получить список БД и отношений, которые должны быть доступными, и ранжировать его по разным критериям. Некоторые отношения требуют особого внимания при принятии решения по их размещению в отдаленной БД. В этом случае должны быть рассмотрены их проекции, используемых в различных запросах, и горизонтальное разбиение отношения через указания диапазона ключей.

Ключевые слова: *информационная система; распределенная база данных; распределенная транзакция; консолидированный учет; система управления базами данных; SQL-запрос; автоматизированное рабочее место; парсинг; профайлинг; ANTLR; синхронизация данных; ETL-система.*

M. Dvoretskiy,
S. Borovlyova,
E. Davydenko,
Petro Mohyla Black Sea National University,
Mykolayiv, Ukraine

DISTRIBUTED DATABASE STRUCTURE DESIGN BASED ON SQL-QUERY PARSING

In measures of research, the specificity of operational records associated with the transition from local to distributed database was discussed. This trend is motivated by the presence of different accounting types, by existence of workstations that are geographically remoted and, of course, by the need of consolidating data analysis. In view of distributed database structure usage necessarily is actual question of best data storage strategy for different parts of data in distributed database. In addition, it is important to choose right way and technique of its future synchronizing in case of its changing or may be for making consolidated analysis by on-line analytical processing subsystems.

Research aims to improve execution speed of local SQL-queries with keeping value of data relevance in acceptable measures by optimizing the database structure of remote workstation place. It makes possible by users SQL-queries parsing with future clustering and classification of central DB relations set.

The first stage is creation of user queries accounting with the possibility of its future classification depending on what workstation place it belongs, where is it placed (geographical location), what user role it belongs, and other criteria's. According to the parameter, classification is been doing automatically as well as manually. Particularly attention is payed to separate stage of queries text parsing which is aided to determine relations and attributes list. SQL-query code parsing task is suggested to solve by using open source project ANTLR.

Final analysis that is made based on previously received data allows achieving databases and relations list. This list have to be available on-line and can be ranged by different criteria's. Some relations need of special attention during making a decision of its placing in remote database. In this case relation projections that are used in different queries need to be considered as well as horizontal division based on range of keys.

Key words: *information system; distributed database; distributed transaction; the consolidated accounting; database management system; SQL-query; workstation; parsing; profiling; ANTLR; data synchronization; ETL-system.*

Рецензенти: д. т. н., проф. ***М. Т. Фісун;***
д. т. н., проф. ***І. І. Коваленко.***

Дата надходження статті до редколегії 19.10.16